The 7th International Conference on Smart Blockchain (SmartBlock 2022)

Dec. 2 - 4, 2022
New York, USA

Proceedings

Proceedings

The 7th International Conference on Smart Computing and Communication (SmartCom 2022)

SmartBlock 1

- Neal N. Xiong, Wenyong He and Mingming Lu, *LowFreqAttack: An Frequency Attack Method in Time Series Prediction*
- Hui Zhao, Peng Su, Keke Gai and Meikang Qiu, *A Novel Machine Learning-Based Model for Reentrant Vulnerabilities Detection*
- Jingnian Liu, Weihong Huang, Lijun Xiao, Yingzi Huo, Huixuan Xiong, Xiong Li and Weidong Xiao, *Deep Learning Object Detection*
- Jun Liu, Qi Zhang, Ming Yue Xie and Ming Peng Chen, *A decision-making method for blockchain platforms using axiomatic design*

SmartBlock 2

- Yue Zhang, Keke Gai, Liehuang Zhu and Meikang Qiu, *An Auxiliary Classifier GAN-based DDoS Defense Solution in Blockchain-based Software Defined Industrial Network*
- Shan Jin, Yong Li, Xi Chen, Ruxian Li and Zhibin Shen, *Blockchain-based Fairness-Enhanced Federated Learning Scheme Against Data Poisoning Attack*
- Ying Long, Yinyan Gong, Jiahong Cai, Weihong Huang, Nengxiang Xu and Kuan-Ching Li, *Cryptography of Blockchain*
- Yuan Li, Ran Guo, Guopeng Wang, Lejun Zhang, Jing Qu, Shen Su, Yuan Liu, Guangxia Xu and Huiling Chen, *An Efficient Detection Model for Smart Contract Reentrancy Vulnerabilities*

SmartBlock 3

- An Wang, Shulin He, Congming Wei, Shaofei Sun, Yaoling Ding and Jiayao Wang, *Using Convolutional Neural Network to Redress Outliers in Clustering based Side-Channel Analysis on Cryptosystems*
- Xuan You, Changsong Zhou, Zeng Chen, Yu Gu, Rui Han and Guozi Sun, *Secure File Outsourcing Method Based on Consortium Blockchain*
- Feixiang Ren and Sujuan Qin, *Fabric smart contract read-after-write risk detection method based on key methods and call chains*
- Hui Zhao, Jiacheng Tan, Keke Gai and Meikang Qiu, *A Critical-Path-Based Vulnerability Detection Method for tx.origin Dependency of Smart Contract*

SmartBlock 4

- Hui Zhao, Xing Li, Keke Gai and Meikang Qiu, *A Dynamic Taint Analysis-based Smart Contract Testing Approach*
- Kaiyin Zhu, Neal N. Xiong and Mingming Lu, *A Survey of Weakly-supervised Semantic Segmentation*
- Xiaoli Huang, Andi Liu, Yizhong Liu, Li Li, Zhenglin Lv and Fan Wang, *Construction practice of cloud billing message based on stream native*
- Jiasheng Cui, Li Duan, Mengchen Li and Wei Wang, *A Fine-grained Access Control Framework for Data sharing in IoT based on IPFS and Cross-Blockchain*

  .

SmartBlock 5

- Wenyu Shao, Xueyang Liu, Wenhui Hu, Xiankui Zhang and Xiaodong Zeng,

Research on Diabetes Disease Development Prediction Algorithm Based on Model Fusion

- Zimu Hu, Wei-Tek Tsai and Li Zhang, *Smart-Contract Vulnerability Detection Method Based on Deep Learning*
- Peiyun Ran, Mingsheng Liu, Jianwu Zheng, Zakirul Alam Bhuiyan, Jianhua Li, Gang Li, Shiyuan Yu, Lifeng Wang and Song Tang*, Automatic Smart Contract Generation with Knowledge Extraction and Unified Modeling Language*
- Nengxiang Xu, Weihong Huang, Jiahong Cai, Yinyan Gong, Huan Zhang and Kuan-Chin Li, *Blockchain Scalability Technologies*

SmartBlock 6

- Shuo Wang, Zhiqi Lei, Zijun Wang, Dongjue Wang, Mohan Wang, Yang Gangqiang and Keke Gai*, Blockchain Applications in Smart City: A Survey*
- Chunyan Tong, Zhanwei Xuan, Song Yang, Zheng Zhang, Hongfeng Zhang, Hao Wang and Hao Sun*, Topic-Aware Model for Early Cascade Population Prediction*
- Xiangyang Cui, Zhou Yan, Song Yang, Zheng Zhang, Hongfeng Zhang, Hao Wang and Qi Nie, *GeoNet: Artificial neural network based on geometric network*
- Hongze Wang and Qinying Zhang*, Research on Blockchain-based Smart Contract Technology*

SmartBlock 7

- Tianxiu Xie, Hong Zhang, Yiwei Feng, Jing Qi, Chennan Guo and Keke Gai*, Cross-chain-based Distributed Digital Identity: A Survey*
- Chen Zhang, Dong Mao, Lin Cui, Jiasai Sun, Fan Yang and Cong Cao*, Research on Power Border Firewall Policy Import and Optimization Tool*
- Tao Li, Haolong Wang, Yaozheng Fang, Zhaolong Jian, Zichun Wang and Xueshuo Xie, *Block-gram: Mining Knowledgeable Features for Smart Contract Vulnerability Detection*
- Renjie Niu, Zixiao Jia, Yizhong Liu, Jianhong Lin, Xiaoli Huang and Min Sun*, The enhanced 4A identity authentication center based on Super SIM technology*

SmartBlock 8

- Jiayi Li, Xinsheng Lei, Jieyu Su, Hui Zhao, Zhenyu Guan and Dawei Li*, Verifiable, Fair and Privacy-Preserving Outsourced Computation based on Blockchain and PUF*
- Xiangyu Gao, Meikang Qiu and Hui Zhao*, Architecture Search for Deep Neural Network*
- Siqi Xie, Jiahong Cai, Hangyu Zhu, Ce Yang, Lin Chen and Weidong Xiao*, Blockchain Development*
- Qinying Zhang and Hongze Wang*, Research on Blockchain-based Food Safety Traceability Technology*

SmartBlock 9

- Xiaodong Zeng, Wenhui Hu, Xueyang Liu, Yuhang Chen, Wenyu Shao and Lizhuang Sun*, Few-Shot Learning for Medical Numerical Understanding Based on Machine Reading Comprehension*
- Yawei Li, Yanli Liu, Heng Zhang and Neal Xiong*, A Feature Extraction Algorithm Based on Blockchain Storage that Combines ORB Feature Points and Quadtree Division*
- Weijie Chen, Ran Guo, Guopeng Wang, Lejun Zhang, Jing Qu, Shen Su, Yuan Liu, Guangxia Xu and Huiling Chen*, Smart contract vulnerability detection model based on Siamese network*
- Chunyan Tong, Kai Zhang, Song Yang, Zheng Zhang, Hongfeng Zhang, Hao Wang and Yerui Liu*, Context-User Dependent Model for Cascade Retweeter Prediction*
- Shiyu Wang, Jiahao Xie, Mingming Lu and Neal N. Xiong*, FedGraph-KD: An Effective Federated Graph Learning Scheme Based on Knowledge Distillation*

SmartBlock 10

- Ni Zhang, Baoquan Ma, Peng Wang, Xuhua Lei, Yejian Cheng, Jiaxin Li, Xiaoyong Huai, Zhiwei Shen, Ningning Song and Long Wang*, Heterogeneous system data storage and retrieval scheme based on blockchain*
- Dylan Dylan Yu, Ethan Yang, Alissa Shen, Dan Tamir and Naphtali Rishe*, Fibereum: A Novel Distributed Ledger Technology System*
- Pengyong Ding, Zian Jin, Yizhong Liu, Min Sun, Hong Liu, Li Li and Xin Zhang, *Study of Indistinguishable Obfuscated Encryption and Decryption based on Transformer Model*
- Jiahong Xiao, Wei Liang, Jiahong Cai, Hangyu Zhu, Xiong Li and Songyou Xie, *An Investigation of Blockchain-based Sharding*

LowFreqAttack: An Frequency Attack Method in Time Series Prediction

Neal N. Xiong[1], Wenyong He[2,*], and Mingming Lu[2]

[1] National Engineering Laboratory for Educational Big Data, Central China Normal University, Wuhan, China
[2] School of Computer Science and Engineering, Central South University, ChangSha, 410083, HuNan, China

Abstract. Time series prediction has become an important research direction in data mining because of the time-varying pattern of data in various fields. However, time series prediction suffers from the problem of vulnerability to adversarial example attack, which leads to models making wrong decisions in critical application scenarios and causing great losses to people's lives and properties. In addition, there is relatively little attacks research on time series prediction, and the existing attack methods simply migrate classical-attack methods in the image to time series prediction. On the one hand, it not only without fully considering the characteristics of temporal data but also without comparing and analyzing the effects of those classical-attack methods on time series prediction models. On the other hand, there is no comparative analysis of the effectiveness of these classical attack methods in different time series prediction methods. To address the above problems, this paper firstly compares the effectiveness of the attack methods on some time series prediction models and analyzes the inner mechanism of these time series prediction models. In addition, this paper finds that the defense ability of those models is related to their ability to portray the overall trend of time series data. Therefore, this paper further propose the new attack method, LowFreqAttack. The experimental results show that LowFreqAttack can attack the three existing time series prediction models better than the existing attach methods.

Keywords: Time Series Prediction · Adversarial Attack · Frequency.

1 Introduction

Time series prediction is an important research area in data mining and machine learning. On the one hand, time series prediction is widely used in various fields such as healthcare [1–6], traffic flow prediction [7–10], and energy consumption prediction [11]. On the other hand, time series prediction can reflect the intrinsic patterns of time-series data over time. In recent years, given the successful

* Corresponding author: hewenyong@csu.edu.cn

application of deep learning in fields such as computer vision and natural language processing. Some researchers have recently started to use it for time series prediction tasks as well [12–15].

However, deep learning is extremely vulnerable in adversarial settings. Attackers generate adversarial examples by finding perturbations that are not easily detectable by humans, and these examples can cause prediction models to produce erroneous outputs with high confidence, which can severely affect the decision-making of some critical applications. Recently, it has been shown that deep learning models are equally vulnerable to attacks on time series prediction tasks [18, 19]. For example, in a user health monitoring system, a fraudster can make the patient miss the best treatment time for the onset of the disease by faking the illusion that all health indicators of the onset patient are not abnormal, thus seriously endangering the patient's life [20–25]. However, there is relatively little research on time series prediction attacks. Therefore, this paper explores the time series-based counterattack. First, CNN, LSTM, and transformers [26] are used as the base models for time series prediction. The robustness is compared under the gradient-based attack approach. Experiments show that the ability of the three models to defend against attacks decreases in the order of transformer, CNN, and LSTM. Subsequently, to analyze the reasons for the different robustness of each model, this paper finds that the defense ability of these three models is related to their ability to portray the overall trend of the time series data by comparing and analyzing the inner mechanism of these time series prediction models.

To deeply analyze the reason for the robustness of the transformer, this paper finds inspiration from the application of the transformer in images. As early as in the field of vision, researchers points out that the transformer makes more use of the global information of the image for prediction [27], where the global information can be understood as the overall contour features in the image. Therefore, this paper proposes a more low-frequency component attack method Low-Frequency Attack (LowFreqAttack) based on this feature. And the attack performance of LowFreqAttack is compared with the existing attack methods on some existing defense methods. The experiments show that LowFreqAttack still guarantees better attack capability, thus verifying the effectiveness of the LowFreqAttack.

The remainder of this article is organized as follows. In Section II, we review the related literature on time series forecasting and transformer research in images. Section III elaborates some attack methods and the design method. We analysis the results in Section IV, and this article is concluded in Section V with a summary of potential future studies.

2 Related Work

2.1 Research on Time Series Forecasting

Time series prediction has become an important research direction in data mining and is now widely used in various areas of life [28, 29]. However, studies have

shown that deep learning models are vulnerable to attacks [16]. While there have been many significant studies on adversarial attacks on data such as images, text, and graph structures, the opposite is true for time series, for which there is a paucity of research. Forestier [18] pioneered the study of the vulnerability of deep learning models on time series adversarial examples by applying some attack methods on images [30, 31] to time sequence classification task, which reduces the accuracy of the classification model, thus pointing out the same vulnerability to attacks on the application of time series data. Mode studied the adversarial attacks on multivariate time series prediction and verified the vulnerability of deep learning models to adversarial attacks by conducting experiments on datasets in multiple scenarios. Recently, Wu [19] performed adversarial attacks on some recent time series prediction models and showed that even very small perturbations can equally easily cause significant degradation in the performance of the models. As for the defense methods for time series prediction, the existing defense methods are mainly developed from the perspective of data preprocessing. Juan [32] proposed an efficient data preprocessing method that takes advantage of the correlation of continuous time data and uses mean substitution for data noise reduction. Fahad Algarni [33] used the idea that input discretization can mitigate adversarial attack ideas, and then proposed a defense method based on thermometer coding, which effectively improves the robustness of the model.

2.2 Transformer Research in Images

Recently, the transformer, which has shown great potential in the field of natural language processing [26,34,35], can reach the same level of Convolutional Neural Network (CNN) in various tasks of computer vision [36,37], and the transformer is usually able to show better robustness. Alexey et al [36] first proposed the transformer architecture for image recognition, thus pioneering Vision Transformer (Vit), which not only has the same high accuracy as CNN but also shows better model robustness. Subsequently, Naseer [38] compared the robustness of three different architectures of the transformer, as well as some better performing CNN models considering the presence of severe occlusion, region shifting, spatial alignment, adversarial and natural perturbations in images, and again, found that the transformer has better robustness.

3 Method

3.1 Overview

Time Series The set $X = [x_1, x_2, ..., x_T], x_T \in R^n$ formed by the values corresponding to T consecutive unit times can be called a time series.

Time Series Prediction Input a time series $X = [x_1, x_2, ..., x_T]$, and predict the value $x_(T + h)$ of $(T + h)$, where h denotes the horizontal offset relative to the

current moment, which is used to control the number of units of time between the corresponding moment of the predicted value of the prediction task and the current moment, and different offsets are set according to different tasks.

Time Series Adversarial Example On a given time series X = $[x_1, x_2, ..., x_T]$, the attacker finds a small perturbation η based on the characteristics of the input time series, thus generating a new time series $X' = [x_1', x_2', ..., x_T']$, where $X' = X + \eta$, also called the adversarial sample. When the adversarial sample makes the model prediction accuracy decrease significantly, it means the attack is successful.

3.2 Gradient-based Attack Method

FGSM Goodfellow et al [30] proposed the Fast Gradient Sign Method (FGSM) for generating image adversarial examples, which successfully makes GoogleNet models predict errors. FGSM generates perturbations by computing the gradient of the input. In order to generate perturbations faster, the method computes the gradient of only one step. The computational procedure of FGSM for generating perturbations is shown Equation 1 and Equation 2.

$$\eta = \epsilon \cdot sign(\nabla_x J(X, \hat{Y})), \tag{1}$$

$$X' = X + \eta, \tag{2}$$

BIM Kurakin et al [31] proposed an iterative version (Basic Iterative Method) based on FGSM. The idea of BIM is to generate adversarial examples by moving smaller steps instead. Thus BIM iterates more times compared to the single-step perturbation generation method of FGSM. After each gradient-based generation of adversarial examples, cropping is performed, thus ensuring that the range obtained from each calculation lies in [X-α, X+α].

3.3 The Analysis of Attack Effect

The robustness of CNN, LSTM, and Transformer under the attack methods of FGSM and BIM were compared, and some of the results are shown in Figure 1.

As can be seen from Figure 1, the prediction performance of the transformer performs the best in both cases with the same perturbation factor ϵ. In particular, the performance of those is almost equal in the case of no perturbation. As the perturbation gradually increases, the performance of CNN and LSTM decreases dramatically, but the performance of the transformer decreases more slowly, which indicates that the transformer has better robustness under the same level of attack. In this paper, we argue that Transformer is better able to resist the noise of high-frequency signals because the window of its input data tends to be large, which is conducive to preserving the sequence correlation for

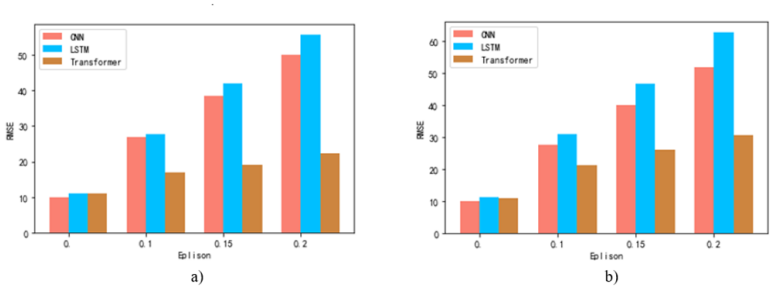


Fig. 1. a) The Comparison of results under FGSM attack [30]. b) The Comparison of results under BIM attack [31].

a longer period. In addition, the transformer learns the attention weights among the corresponding data at each time point through the self-attention mechanism, which makes it more dependent on the overall sequence to complete the prediction. However, noise tends to manifest itself locally in the few samples that are disturbed and have less impact on the global features on which the transformer relies to make decisions. Therefore, the transformer is still able to ensure a certain level of robustness even under large disturbances. On the contrary, CNN and LSTM focus more on the time series inside a segment window and are therefore much more affected than the transformer.

Based on the above phenomenon, this paper argues that transformer has more affinity for low-frequency features. Therefore, this paper then proposes a frequency domain-based perturbation attack method LowFreqAttack.

3.4 LowFreqAttack

The process of LowFreqAttack is divided into four processes. (1) Generating perturbation by attack methods FGSM or BIM. (2) Transformation of the perturbation into multiple components in the frequency domain. (3) Sieving the high-frequency components in the frequency domain. (4) Transforming the sieved perturbation back to the time domain to obtain a more low-frequency perturbation.

4 Result Ayalysis

4.1 The Overview of Experimental Data

The data used in this paper comes from Caltrans' Performance Measurement System (PeMS). This system collects traffic flow in real-time through detectors on the freeway. In this paper, we use two columns of this dataset (Time, Traffic). The time column represents the instantaneous moment of the detector feedback data, with 5 minutes as the feedback interval. The traffic flow column, on the

other hand, indicates the number of vehicles that passed through this detector during the interval from the previous moment to the current moment. Among them, the training set (2016/01/04 - 2016/02/29) includes 7776 data and the test set (2016/03/04 - 2016/03/31) contains a total of 4320 data.

4.2 The Effectiveness Comparison Under the Defense method

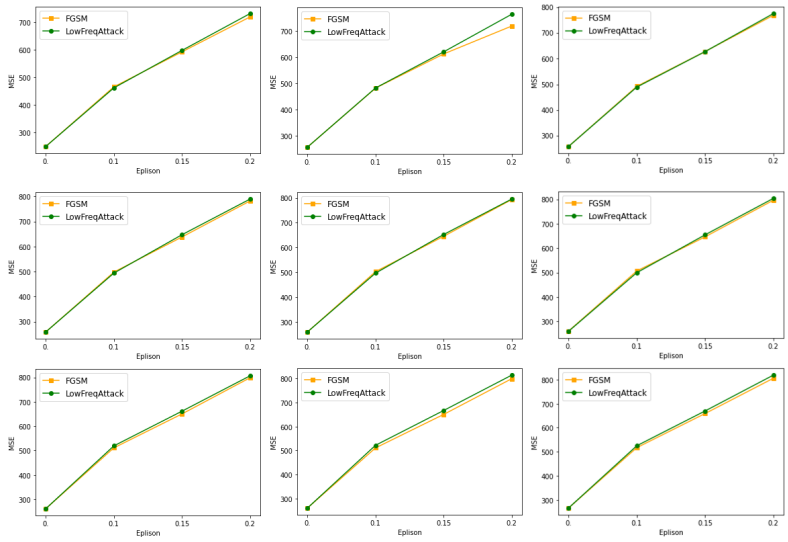


Fig. 2. The comparison under Kalman filtering.

To better show the performance between the lowfreqattack and other attack methods, we compare the attack effect by performing the attack under two defense methods. We use the two most commonly used time series defense methods: Kalman filter and wavelet transform. By inputting the generated adversarial samples into the time series prediction model, we obtain the experimental results under wavelet transform as shown in Figure 2. The nine plots in the figure represent the results of LowFreqAttack compared with the FGSM attack method under the wavelet transform defense method. The wavelet transform defense control factor is a quantile factor, ranging from 0.1 to 0.9, and the closer the value is to 1, the stronger the wavelet defense is. The horizontal coordinate in Figure 2 indicates the perturbation factor ϵ, with values ranging from 0.1, 0.15, 0.2, which is used to control the size of the perturbation, the larger the value the larger the perturbation. the vertical coordinate in Figure 2 is MSE, which is used to indicate the deviation of the prediction result. the larger the value, the better the attack effect, then the better the attack method. When the perturbation factor is 0.1, the LowFreqAttack attack effect exceeds the FGSM

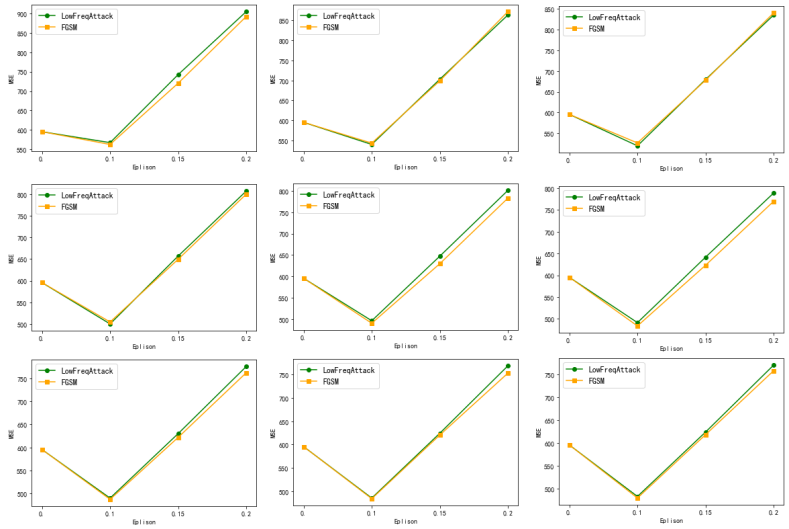


Fig. 3. The comparison under wavelet transform.

by about 0.63% on average. when the perturbation factor is 0.15, the LowFreqAttack attack effect exceeds the FGSM by about 1.52% on average. when the perturbation factor is 0.2, the LowFreqAttack attack effect exceeds the FGSM by about 1.07% on average. Therefore, based on the above experimental results, shows that LowFreqAttack is more aggressive than the FGSM method for the same wavelet defense strength.

Figure 3 shows the comparison of the experimental results under the Kalman filter defense method. The nine plots in Figure 3 represent the results of LowFreqAttack compared with the FGSM attack method under the Kalman filter defense method. The Kalman filtering defense control factor, Karman-level, ranges from 0.1 to 0.9, and the closer the value is to 0.1, the stronger the Kalman filtering defense is. In Figure 3, the horizontal coordinate is the perturbation factor ϵ, with values ranging from 0.1, 0.15, 0.2, which is used to control the size of the perturbation, the larger the value the larger the perturbation. the vertical coordinate is MSE, which is used to indicate the deviation of the prediction result. the larger the value, the better the attack effect. When the perturbation factor is 0.1, the LowFreqAttack attack effect exceeds the FGSM by about 0.73% on average. when the perturbation factor is 0.15, the LowFreqAttack attack effect exceeds the FGSM by about 1.25% on average. when the perturbation factor is 0.2, the LowFreqAttack attack effect exceeds the FGSM by about 1.20%. Therefore, based on the above experimental results, it shows that LowFreqAttack is more aggressive than the FGSM method with the same Kalman filtering defense strength.

5 Conclusion

In this paper, we compared and analysed the robustness of CNN, LSTM, and transformer on time series prediction tasks under the attack gradient-based approach. We found that the transformer is more robust. In addition, this paper analysed the reasons why the transformers are more robust in the frequency domain and also points out that the transformer is vulnerable to low-frequency perturbation attacks. Based on the vulnerability of transformers, this paper proposed a more low-frequency attack method, LowFreqAttack. Comparing LowFreqAttack with other attack methods under the same defense method, it is found that LowFreqAttack can still maintain better attack capability, thus verifying its effectiveness.

Acknowledgements This work was partially supported by the National Natural Science Foundation of China under Grant No. U20A20182 and 62177019.

References

1. Pranjul Yadav, Michael Steinbach, Vipin Kumar, and Gyorgy Simon. Mining electronic health records (ehrs) a survey. *ACM Computing Surveys (CSUR)*, 50(6):1–40, 2018.
2. Shruti Kaushik, Abhinav Choudhury, Pankaj Kumar Sheron, Nataraj Dasgupta, Sayee Natarajan, Larry A Pickett, and Varun Dutt. Ai in healthcare: time-series forecasting using statistical, neural, and ensemble architectures. *Frontiers in big data*, 3:4, 2020.
3. Feng Xia, Ruonan Hao, Jie Li, Naixue Xiong, Laurence T Yang, and Yan Zhang. Adaptive gts allocation in ieee 802.15. 4 for real-time wireless sensor networks. *Journal of Systems Architecture*, 59(10):1231–1242, 2013.
4. Prabhat Kumar, Randhir Kumar, Gautam Srivastava, Govind P Gupta, Rakesh Tripathi, Thippa Reddy Gadekallu, and Neal N Xiong. Ppsf: a privacy-preserving and secure framework using blockchain-based machine-learning for iot-driven smart cities. *IEEE Transactions on Network Science and Engineering*, 8(3):2326–2341, 2021.
5. Chunxue Wu, Chong Luo, Naixue Xiong, Wei Zhang, and Tai-Hoon Kim. A greedy deep learning method for medical disease analysis. *IEEE Access*, 6:20021–20030, 2018.
6. Hongju Cheng, Zhe Xie, Yushi Shi, and Naixue Xiong. Multi-step data prediction in wireless sensor networks based on one-dimensional cnn and bidirectional lstm. *IEEE Access*, 7:117883–117896, 2019.
7. Liang Zhang, Jianqing Wu, Jun Shen, Ming Chen, Rui Wang, Xinliang Zhou, Cankun Xu, Quankai Yao, and Qiang Wu. Satp-gan: Self-attention based generative adversarial network for traffic flow prediction. *Transportmetrica B: Transport Dynamics*, 9(1):552–568, 2021.
8. Yonglei Yao, Naixue Xiong, Jong Hyuk Park, Li Ma, and Jingfa Liu. Privacy-preserving max/min query in two-tiered wireless sensor networks. *Computers & Mathematics with Applications*, 65(9):1318–1325, 2013.

9. Chunxue Wu, Bobo Ju, Yan Wu, Xiao Lin, Naixue Xiong, Guangquan Xu, Hongyan Li, and Xuefeng Liang. Uav autonomous target search based on deep reinforcement learning in complex disaster scene. *IEEE Access*, 7:117227–117245, 2019.
10. Anmin Fu, Xianglong Zhang, Naixue Xiong, Yansong Gao, Huaqun Wang, and Jing Zhang. Vfl: a verifiable federated learning with privacy-preserving for big data in industrial iot. *IEEE Transactions on Industrial Informatics*, 2020.
11. Muhammad Shafiq, Zhihong Tian, Ali Kashif Bashir, Xiaojiang Du, and Mohsen Guizani. Corrauc: a malicious bot-iot traffic detection method in iot network using machine-learning techniques. *IEEE Internet of Things Journal*, 8(5):3242–3254, 2020.
12. Sourabh Shastri, Kuljeet Singh, Sachin Kumar, Paramjit Kour, and Vibhakar Mansotra. Time series forecasting of covid-19 using deep learning models: India-usa comparative case study. *Chaos, Solitons & Fractals*, 140:110227, 2020.
13. Meikang Qiu, Keke Gai, and Zenggang Xiong. Privacy-preserving wireless communications using bipartite matching in social big data. *Future Generation Computer Systems*, 87:772–781, 2018.
14. Han Qiu, Meikang Qiu, and Zhihui Lu. Selective encryption on ecg data in body sensor network based on supervised machine learning. *Information Fusion*, 55:59–67, 2020.
15. Meikang Qiu, Lei Zhang, Zhong Ming, Zhi Chen, Xiao Qin, and Laurence T Yang. Security-aware optimization for ubiquitous computing systems with seat graph approach. *Journal of Computer and System Sciences*, 79(5):518–529, 2013.
16. Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. A survey on adversarial attacks and defences. *CAAI Transactions on Intelligence Technology*, 6(1):25–45, 2021.
17. Fazle Karim, Somshubra Majumdar, and Houshang Darabi. Adversarial attacks on time series. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3309–3320, 2020.
18. Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Adversarial attacks on deep neural networks for time series classification. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.
19. Tao Wu, Xuechun Wang, Shaojie Qiao, Xingping Xian, Yanbing Liu, and Liang Zhang. Small perturbations are enough: Adversarial attacks on time series prediction. *Information Sciences*, 587:794–812, 2022.
20. Tengfei Ma, Cao Xiao, and Fei Wang. Health-atm: A deep architecture for multifaceted patient health record representation and risk prediction. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 261–269. SIAM, 2018.
21. Yewang Chen, Lida Zhou, Songwen Pei, Zhiwen Yu, Yi Chen, Xin Liu, Jixiang Du, and Naixue Xiong. Knn-block dbscan: Fast clustering for large-scale data. *IEEE transactions on systems, man, and cybernetics: systems*, 51(6):3939–3953, 2019.
22. Jin Zhao, Jifeng Huang, and Naixue Xiong. An effective exponential-based trust and reputation evaluation system in wireless sensor networks. *IEEE Access*, 7:33859–33869, 2019.
23. Yongbin Gao, Xuehao Xiang, Naixue Xiong, Bo Huang, Hyo Jong Lee, Rad Alrifai, Xiaoyan Jiang, and Zhijun Fang. Human action monitoring for healthcare based on deep learning. *Ieee Access*, 6:52277–52285, 2018.
24. Wei Zhang, Shiwei Zhu, Jian Tang, and Naixue Xiong. A novel trust management scheme based on dempster–shafer evidence theory for malicious nodes detection in wireless sensor networks. *The Journal of Supercomputing*, 74(4):1779–1801, 2018.

25. Shaobo Huang, Zhiwen Zeng, Kaoru Ota, Mianxiong Dong, Tian Wang, and Neal N Xiong. An intelligent collaboration trust interconnections system for mobile information control in ubiquitous 5g networks. *IEEE transactions on network science and engineering*, 8(1):347–365, 2020.
26. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
27. Shikhar Tuli, Ishita Dasgupta, Erin Grant, and Thomas L Griffiths. Are convolutional neural networks or transformers more like human vision? *arXiv preprint arXiv:2105.07197*, 2021.
28. Keyi Zhang, Ramazan Gençay, and M Ege Yazgan. Application of wavelet decomposition in time-series forecasting. *Economics Letters*, 158:41–46, 2017.
29. Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Evaluating surgical skills from kinematic data using convolutional neural networks. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 214–221. Springer, 2018.
30. Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
31. Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018.
32. Juan Antonio Cortés-Ibáñez, Sergio González, José Javier Valle-Alonso, Julián Luengo, Salvador García, and Francisco Herrera. Preprocessing methodology for time series: An industrial world application case study. *Information Sciences*, 514:385–401, 2020.
33. Zhongguo Yang, Irshad Ahmed Abbasi, Fahad Algarni, Sikandar Ali, and Mingzhu Zhang. An iot time series data security model for adversarial attack based on thermometer encoding. *Security and Communication Networks*, 2021, 2021.
34. J Devlin M Chang K Lee and K Toutanova. Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
35. Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
36. Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
37. Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.
38. Muhammad Muzammal Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. *Advances in Neural Information Processing Systems*, 34:23296–23308, 2021.

A Novel Machine Learning-Based Model for Reentrant Vulnerabilities Detection

Hui Zhao[1], Peng Su[2], Keke Gai[3] ✉, and Meikang Qiu[4]

[1] Educational Information Technology Laboratory, Henan University, Kaifeng, China
zhh@henu.edu.cn
[2] Software School, Henan University, Kaifeng, China
supeng@henu.edu.cn
[3] School of Cyberspace Science and Technology, Beijing Insititute of Technology, Beijing, China
gaikeke@bit.edu.cn ⋆
[4] Dakota State University, Midison, SD, USA
qiumeikang@ieee.org

Abstract. Machine learning-based models are one of the main methods for detecting reentrant vulnerabilities. However, these models extract smart contract features only from a single form, resulting in incompleteness and inaccuracy of features. To address this problem, we propose a novel machine learning-based model for reentrant vulnerabilities detection. We extract and fuse features from abstract syntax trees, opcodes, control flow graph basic blocks, and combine machine learning algorithms for reentrant vulnerabilities detection. Additionally, to address the time-consuming problem of manual labeling, we also propose an approach for automatically adding dataset labels. We perform experiments on Smartbugs and SolidiFi-benchmark datasets and results show that our model outperforms existing models.

Keywords: Smart Contracts · Machine Learning · Reentrant Vulnerabilities · Dataset Labels · Detection.

1 Introduction

In recent years, blockchain has become a hotspot of research and application. We begin to focus on information security and information privacy [7, 13, 21]. Privacy stealing and privacy protection are a long-term confrontation process [18]. Blockchain technology [2,6,8] has many applications in modern information technology and computing environments. This technology uses key features, such as distributed ledgers and tamper resistance, to use distributed computing resources [11, 14, 20]. As one of the key components of blockchain [5], smart contracts play a vital role in achieving automation functions. The concept of smart contracts was proposed by Nick Szabo [23]. When the condition is triggered, the smart contract will execute the corresponding transaction. Smart contracts

⋆ Keke Gai is corresponding author.

cannot find suitable application scenarios at that time. Therefore, the smart contract cannot develop quickly. Until the advent of blockchain technology, smart contracts begin to attract people's attention and gradually became a research hotspot. However, security problems of smart contracts are beginning to be exposed. One of the most destructive attacks is the reentrant attack.

Reentrant attacks are one of the most destructive attacks in blockchain systems [9, 22] caused by smart contract vulnerabilities [3]. When an attacker recursively calls the **draw(.)** of the target contract to withdraw funds from the target contract, a reentry attack will occur. When the contract cannot update its balance after sending the funds, the attacker can call the withdrawal function continuously to exhaust the contract funds. A well-known reentrant attack is "**TheDAO**" attack which caused a loss of 60 million US dollars [16].

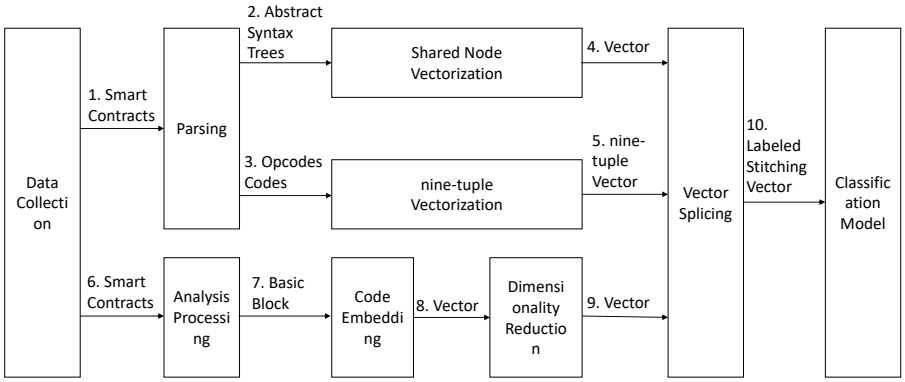
Machine learning-based vulnerabilities detection has been proven effective [10,19]. However, there are two drawbacks to this technique. First, the problem of noise code interference of similar contracts. At the current stage, smart contracts are mostly deployed on blockchains by programmers. To save time, programmers often use copy and paste while writing smart contracts. Therefore, this situation has produced a lot of similar contracts [26]. Vulnerabilities are usually caused by a few statements, but similar contracts contain many statements that are not related to the vulnerability. We call it noise codes [12]. Second, the time-consuming issue of manual marking. At the current stage, researchers usually use traditional methods such as Mythril and Slither to detect data sets. Then, they manually add tags to the data set according to the detection results [17].

In order to detect whether smart contracts have reentrant vulnerabilities, we propose a novel machine learning-based model for reentrant vulnerabilities' detection. Our proposed model parses smart contracts into abstract syntax trees and opcodes. By sharing node and opcode categorization techniques, we extract feature vectors from abstract syntax trees and opcodes. Our model parses smart contracts into control flow graphs and extracts basic blocks containing transaction features from them. We concatenate the basic blocks and perform word embeddings on them through the n-grams algorithm. Our model concatenates these three types of vectors to obtain the full feature vector for the smart contract. Our proposed model automatically labels vectors through code embedding and similarity detection techniques. Under six machine learning models, our method is compared with existing vulnerability detection methods. Experiments show that our proposed model improve the accuracy of reentrant vulnerabilities' detection. Moreover, our accuracy rate is much higher than some common smart contract detection tools [1]. Our proposed model automatically add labels to vectors, which solves the time-consuming problem of manually labeling data.

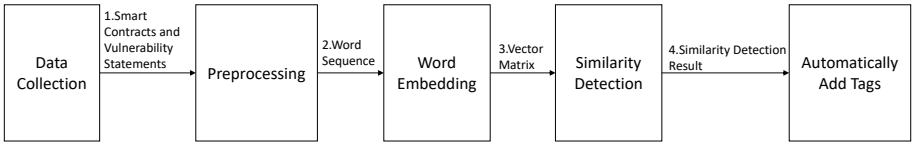
The main contributions of this paper are as follows. (1) We extract and fuse features from smart contract abstract syntax trees, opcodes, and control flow graph basic blocks, which enhances feature completeness and accuracy. (2) We automatically add labels to smart contracts through code embedding technology and similarity comparison technology, which solves the time-consuming manual labeling. (3) We perform vulnerability detection on the extracted smart contract

features under the Smartbugs and SolidiFi-benchmark datasets and 6 machine learning algorithms, and the results show that our model outperforms existing models.

The remainder of this paper is organized in the following order. Section 2 describes the design of our proposed model, which is applied to reentrant vulnerabilities detection. Section 3 describes our proposed algorithms. We analyze the experimental results in Section 4 and conclude the paper in Section 5.



(a) Collecting smart contracts and extracting smart contract features



(b) Collecting vulnerability statements and adding smart contract tags

Fig. 1. The overall workflow of our proposed model.

2 Proposed Model

2.1 Overview

Fig. 1(a) shows the step of collecting smart contracts and extracting smart contract features. The system parses smart contracts into abstract syntax trees and extracts features from smart contracts by solving shared nodes. The system parses the smart contract into opcodes and obtains the nine-tuple vector of the smart contract by traversing the opcodes. The system parses the smart

contract into a control flow graph, from which it extracts basic blocks containing transaction characteristics. These basic blocks are converted into vectors by word embedding technology and dimensionality reduction is performed on them. The system concatenates three types of vectors. Our proposed system uses six machine learning algorithms as classification models for experiments.

Fig. 1(b) shows the step of collecting vulnerability statements and adding smart contract tags. The system relies on code embedding and space vector comparison for automatic labeling. Through data preprocessing, smart contract statements and vulnerability statements are converted into word sequences. The system uses the **FastText** tool to train it into word vectors and sums to get the sentence vector matrix. Through the results of similarity detection, labels are automatically added to the smart contracts.

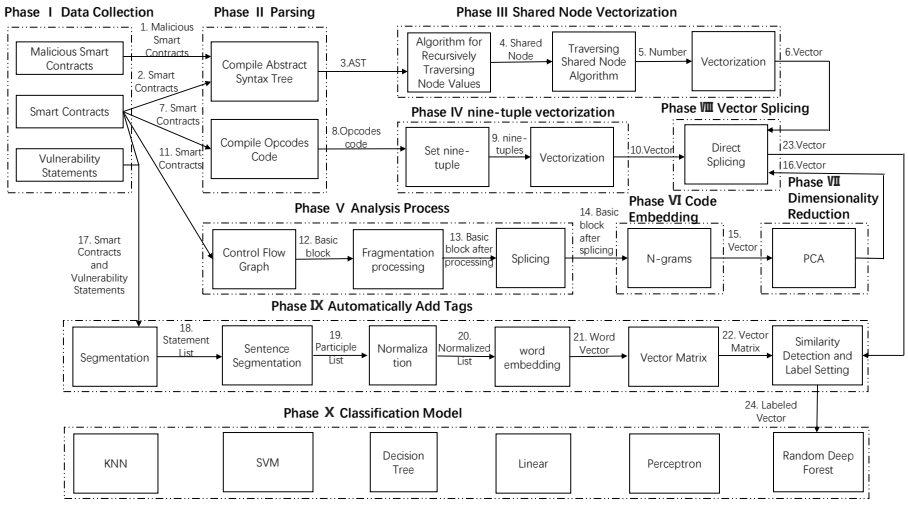


Fig. 2. The phase diagram of our proposed model.

2.2 Model Design

The design scheme of the model mainly includes 10 phases. Data collection will be completed in the first phase. The second phase is responsible for the parse of the smart contract. The third phase realizes the vectorization of shared nodes. The nine-tuple vectorization will be completed in the fourth phase. The fifth phase is to analyze and process the smart contract. The sixth phase is to perform code embeddings on basic blocks. The seventh phase is to reduce the dimensionality of the vector. The eighth phase is responsible for vector splicing. The ninth phase realizes automatically add tags. The tenth phase is responsible for the classification model. The schematic diagram of the model construction

method is shown in Fig. 2. Next, we will describe the core phases within our model.

In the third phase, due to our goal is to obtain the shared nodes between the malicious smart contract and the smart contract set, we first need to obtain the values of all nodes in the smart contract abstract syntax tree. In order to facilitate data manipulation, we convert the abstract syntax tree in JSON format to dictionary format. Each abstract syntax tree contains three types of nodes. The three types are **list**, **dictionary** and **string**. Our goal is to get the node value of all string types. Through recursive traversal list and dictionary, we get all the node values of the abstract syntax tree. Through traversal of shared node number, we obtain the number of shared nodes of the two abstract syntax trees.

In the fourth phase, we divide opcode instructions into 9 categories, namely PUSH, DUP, SWAP, LOG, OPERATION, LOGIC, COMPARISON, VALUE1, VALUE2. By counting the number of each category in the opcode instructions of the smart contract, we get a nine-tuple vector. To reduce the traversal time, we further simplified the opcode instructions. We delete the operands in the opcode instructions. To be precise, we delete the operand after the **PUSH** instruction. Next, we use the third-stage normalization formula to normalize the nine-tuple vector.

Phase 5 is divided into three parts: the smart contract is parsed into a control flow graph, sharding processing and basic block splicing. We use python third-party library **py-solc** to parse smart contracts into bytecodes. Then we parse the bytecodes into a control flow graph and obtain all the basic blocks from the control flow graph. Because reentrant attacks are mainly caused by transfer operations, we focus on the basic blocks that contain transactions. We extract these basic blocks that contain the keywords **'CALLVALUE'**, **'CALLDATALOAD'**, and **'CALLDATASIZE'**. We splice all the extracted basic blocks together to retain the continuous operation characteristics of the transaction.

Phase 9 refers to our previous work [25]. At this phase we propose an automatic data labeling algorithm, which is elaborated in Section III. This phase is divided into 7 parts: contract segmentation, sentence segmentation, normalization, word embedding, vector matrix, similarity detection, set vector labels.

In order to prove that the features we extracted are feasible, we perform experimental evaluation under six machine learning models. Through experimentation, we can observe whether our method works and which model is better for our method.

3 Algorithm

In this section, we will describe our proposed *Automatic Data Labeling* (ADL) Algorithm. The goal of the algorithm is to automatically add labels to datasets, thereby reducing the time-consuming manual labeling process.

The input to the Automatic Data Labeling Algorithm is a set of smart contracts, which we denote by **SCS**. The smart contract set includes training set, test set and malicious statement set. The output of this algorithm is the label

Algorithm 1 Automatic Data Labeling Algorithm

Input: SCS
Output: Label_set

```
function LABEL(SCS)
    op_1 ← Sentence segmentation and word segmentation are performed on SCS.
    op_2 ← Perform word embeddings on op_1.
    train_set,test_set,eyi_statement_set ← Divide the op_2.
    for smart_contract in train_set do
        for statement in smart_contract do
            for statement_eyi in eyi_statement_set do
                if The similarity between statement and statement_eyi is greater than the threshold then
                    smart_contract's label is set to 1.
                else
                    smart_contract's label is set to 0.
                end if
            end for
        end for
    end for
    Label_set ← labels for train_set and test_set. return Label_set
end function
```

set. This set includes labels for all smart contracts in the training and testing sets. We denote the output of this algorithm by **Label_set**.

The Automatic Data Labeling Algorithm is applied in the ninth phase of our proposed model. The goal of the algorithm is to obtain the labels of all smart contracts in the training and test sets. When the label is 1, it means that the smart contract has a vulnerability risk. When the label is 0, it means that the smart contract is a security contract.

The workflow of the Automatic Data Labeling Algorithm is: (1) Input smart contracts into the algorithm. (2) Sentence segmentation and word segmentation for smart contracts. (3) Perform word embedding on the segmented smart contracts. (4) Divide smart contracts into training set, test set and malicious statement set. (5) All the sentences of each smart contract in the training set are checked for similarity with the malicious sentence set. According to the comparison between the detection result and the threshold, the label of the contract is set. (6) The test set and the training set are treated the same way. (7) Save all labels in **Label_set**.

Time complexity. The algorithm uses a three-level **for** loop. The first **for** loop is used to traverse all smart contracts in the training set. The second level of **for** loop is used to traverse all statement vectors in the smart contract. The third layer of **for** loop is used to traverse all the sentence vectors in the malicious sentence set. The test set and training set operate in the same way. Therefore, the time complexity of this algorithm is $O(n^3)$.

Table 1. Experiment settings

Contract sources	reference [4]
Abstract syntax tree parser tool	py-solc
Opcode parsing tool	py-solc
Control flow graph parsing	EVM-CFG-BUILDER
Divider	newline (‘\n’)
Word segmentation tool	Jieba
Word embedding tool	FastText,n-grams
Training vector parameters	minCount =1 lr = 0.05 dim = 20 loss = ns
Threshold	0.92

4 Experiments and Results

4.1 Experiment Configuration

We ran the assessment in the following environment. We used Python's third-party library **jieba** as a word segmentation tool. We also used Python (v3.7) to implement our model in the Pychar IDE. The hardware platform was Lenovo ideapad 300-15ISK laptop, equipped with Windows 10 operating system, 2.3GHz CPU, i5 version kernel and 8 GB 2691MHz LPDDR3 memory [25]. The experiment settings were shown in Table 1.

Our proposed model adopted the n-grams algorithm for word embedding in the sixth phase. We conducted experiments when n was equal to 1, 2 and 3 respectively. We defined three different algorithms as 1-grams, 2-grams, 3-grams. We used these three names to denote the overall model with different algorithms. Selected existing methods included Xu et al.'s method [24], Wang et al.'s method [17], and NeuCheck [15]. NeuCheck [15] used three different n-grams algorithms to detect vulnerabilities. We defined as NeuCheck_1, NeuCheck_2, NeuCheck_3.

4.2 Experimental Results

Fig. 3 showed the experimental comparison between our method and existing methods under the KNN model. Among the existing methods, Xu et al.'s method had the highest precision rate, reaching 86%. Our method achieved 84% precision rate, which was 2% lower than existing methods. The precision rate of existing methods was stable at 70% to 86%. Among the existing methods, Wang et al.'s method had the highest macro average, reaching 87%. The macro average of our method was the highest, reaching 89%.

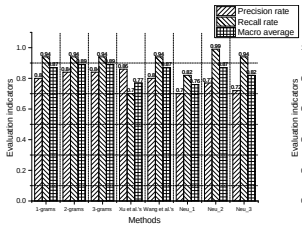


Fig. 3. Comparison of existing methods under KNN model.

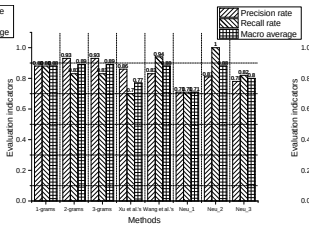


Fig. 4. Comparison of existing methods under SVM model.

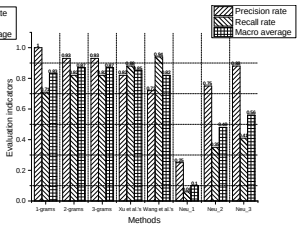


Fig. 5. Comparison of existing methods under Decision Tree model.

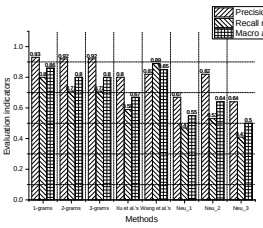


Fig. 6. Comparison of existing methods under Random Deep Forest model.

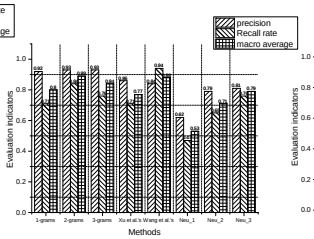


Fig. 7. Comparison of existing methods under Linear model.

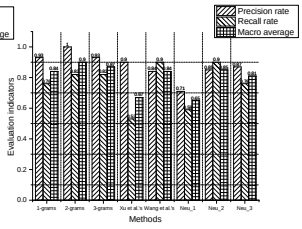


Fig. 8. Comparison of existing methods under Perceptron model.

Fig. 4 showed the experimental comparison between our method and existing methods under the SVM model. Among the existing methods, Xu et al.'s method had the highest precision rate, reaching 86%. Our method achieved 93% precision rate, which was much larger than existing methods. The precision rate of existing methods was stable at 71% to 93%. Among the existing methods, Wang et al.'s method had the highest macro average, reaching 88%. The macro average of our method was the highest, reaching 89%.

Fig. 5 illustrated an experimental comparison of existing methods under the Decision Tree model. The vulnerability detection precision rate of existing methods remained between 70% and 88%. The vulnerability detection precision rate of our methods were all maintained above 93%, with a maximum of 100%. Among the existing methods, the method of Xu et al. had the highest macro average, reaching 85%. Our method achieved an average of 87% macros, a 2% increase over existing methods.

Fig. 6 illustrated an experimental comparison of existing methods under the Random Deep Forest model. The vulnerability detection precision rate of existing methods remained between 64% and 82%. The vulnerability detection precision rate of our methods were all maintained above 92%, with a maximum of 93%. Among the existing methods, the method of Wang et al. had the highest macro

average, reaching 85%. Our method achieved an average of 86% macros, a 1% increase over existing methods.

Fig. 7 showed the experimental comparison of existing methods under the Linear model. In vulnerability detection precision rate, our method remained above 90%. The precision rate of the existing methods was below 90%. Xu et al.'s method had the highest precision rate, reaching 86%. Compared with existing methods, the precision rate of our method was increased by 7%. Macro values of existing methods remained between 53% and 88%. The macro average of our method was 89%. Compared with existing methods, our macro value increased by 1%.

Fig. 8 showed the experimental comparison of existing methods under the Perceptron model. In vulnerability detection precision rate, our method remained above 93%. The precision rate of the existing methods was below 90%. Xu et al.'s method had the highest precision rate, reaching 90%. Compared with existing methods, the precision rate of our method was increased by 10%. Macro values of existing methods remained between 65% and 85%. The macro average of our method was 90%. Compared with existing methods, our macro value increased by 5%.

5 Conclusions

In this paper, we proposed a novel machine learning-based model for reentrant vulnerabilities detection. Our model parsed smart contracts into abstract syntax trees, control flow graphs and opcodes. We extracted features from abstract syntax trees, control flow graphs and opcodes and concatenated the extracted features. To solve the time-consuming problem of manual labeling, our model implemented automatic labeling. With code embedding and similarity detection technology, our model automatically added tags to the smart contract set. We performed experiments on Smartbugs and SolidiFi-benchmark datasets and results showed that our model outperformed existing models.

Acknowledgment

Natural Science Foundation of Shandong Province (Grant No. ZR2020ZD01).

References

1. Badruddoja, S., Dantu, R., et al.: Making smart contracts smarter. In: IEEE Int'l Conf. on Blockchain and Crypto' (2021)
2. Claudia, P., Tudor, C., Marcel, A., et al.: Blockchain based decentralized management of demand response programs in smart energy grids. Sensors (2018)
3. Dong, C., Li, Y., Tan, L.: A new approach to prevent reentrant attack in solidity smart contracts. In: CCF China Blockchain Conf. (2019)

4. Durieux, T., Ferreira, J., Abreu, R., Cruz, P.: Empirical review of automated analysis tools on 47,587 ethereum smart contracts. ACM/IEEE 42nd Int'l conf. on software eng. (2019)
5. Gai, K., Guo, J., Zhu, L., Yu, S.: Blockchain meets cloud computing: A survey. IEEE Comm. Surveys & Tutorials (2020)
6. Gai, K., Qiu, M., Zhao, H., Tao, L., Zong, Z.: Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing. JNCA (2016)
7. Gai, K., Wu, Y., et al.: Privacy-preserving energy trading using consortium blockchain in smart grid. IEEE Trans. on Industrial Informatics (2019)
8. Gai, K., Wu, Y., et al.: Differential privacy-based blockchain for industrial internet-of-things. IEEE TII (2020)
9. Gai, K., Zhang, Y., et al.: Blockchain-enabled service optimizations in supply chain digital twin. IEEE TSC (2022)
10. Gao, X., Qiu, M.: Energy-based learning for preventing backdoor attack. In: KSEM (3). pp. 706–721 (2022)
11. Hu, F., Lakdawala, S., et al.: Low-power, intelligent sensor hardware interface for medical data preprocessing. IEEE TITB **13**(4), 656–663 (2009)
12. Jianjun, H., Songming, H., et al.: Hunting vulnerable smart contracts via graph embedding based bytecode matching. IEEE TIFS (2021)
13. Li, Y., Gai, K., et al.: Intercrossed access controls for secure financial services on multimedia big data in cloud systems. ACM TMCCA (2016)
14. Li, Y., Song, Y., et al.: Intelligent fault diagnosis by fusing domain adversarial training and maximum mean discrepancy via ensemble learning. IEEE TII **17**(4), 2833–2841 (2020)
15. Lu, N., Wang, B., et al.: Neucheck: A more practical ethereum smart contract security analysis tool. Software: Practice and Experience (2021)
16. Marta, M., Norberto, M.: Consecuencias penales y tributarias a la modificación fraudulenta de los smart contracts. especial referencia al caso the dao. CEFLegal: revista práctica de derecho. Comentarios y casos prácticos (2020)
17. Pouyan, M., Yu, W., Reza, S.: Machine learning model for smart contracts security analysis. In: 17th Int'l Conf. on Privacy, Security and Trust (2019)
18. Qiu, H., Kapusta, K., et al.: All-or-nothing data protection for ubiquitous communication: Challenges and perspectives. Information Sciences (2019)
19. Qiu, H., Qiu, M., Lu, R.: Secure V2X communication network based on intelligent PKI and edge computing. IEEE Network **34**(2), 172–178 (2019)
20. Qiu, H., Zheng, Q., et al.: Topological graph convolutional network-based urban traffic flow and density prediction. IEEE TITS **22**(7), 4560–4569 (2020)
21. Qiu, M., Qiu, H.: Review on image processing based adversarial example defenses in computer vision. In: IEEE 6th Intl Conf. BigDataSecurity. pp. 94–99 (2020)
22. Qiu, M., Qiu, H., et al.: Secure data sharing through untrusted clouds with blockchain-enabled key management. In: 3rd SmartBlock conf. pp. 11–16 (2020)
23. Szabo, N.: Formalizing and securing relationships on public networks. First Monday (1997)
24. Xu, Y., Hu, G., You, L., Cao, C.: A novel machine learning-based analysis model for smart contract vulnerability. Security and Commu. Networks (2021)
25. Zhao, H., Su, P., et al.: Gan-enabled code embedding for reentrant vulnerabilities detection. In: Int'l Conf. on Knowledge Science, Eng. and Management (2021)
26. Zhipeng, G., Vinoj, J., Lingxiao, J., et al.: Smartembed: A tool for clone and bug detection in smart contracts through structural code embedding. In: IEEE Int'l Conf. on Software Maintenance and Evolution (2019)

Deep Learning Object Detection

Jingnian Liu[1,2], Weihong Huang[1,2,*], Lijun Xiao[1,2], Yingzi Huo[1,2,3], Huixuan Xiong[1,2], Xiong Li[4] and Weidong Xiao[5]

[1] School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201,China
[2] Hunan Key Laboratory for Service computing and Novel Software Technology, Xiangtan 411201,China
[3] Guangdong Financial High-tech Zone "Blockchain +" Fintech Research Institute, Foshan 528253,China
[4] School of Computer Science and Engineering, University of Electronic Science and Technology of China, Xiangtan 411201,China
[5] School of Software Engineering, Xiamen University of Technology, Xiamen 361024, China
jingnianl1999@163.com,whhuang@hnust.edu.cn,ljxiao@hnust.edu.cn, 1805513056@qq.com, 15888985929@163.com, lixiongzhq@163.com, xiaoweidong@xmut.edu.cn

Abstract. Object detection techniques are a major part of computer vision research, with large-scale applications in industrial, scientific and other scenarios. Technologies such as face detection, medical image detection, autonomous driving, and traffic detection have played a significant role in people's lives. With the rapid development of deep learning, many application areas, such as image classification, text classification, machine translation, etc., have achieved breakthrough success in combination with deep learning. R-CNN brings object detection into the era of deep learning, and its advantage compared with traditional methods is that the former requires personnel to extract features manually, while the latter uses deep learning to extract features automatically, which greatly improves efficiency, simplifies operation, and opens a new era of object detection research. First, this paper provides an overview of deep learning-based object detection backbone networks, reviews and analyzes milestone object detection algorithms, compares commonly used datasets, summarizes applications, and finally concludes the paper.

Keywords: Computer Vision, Deep Learning, Neural Network, Object-Detection, R-CNN.

1 Introduction

The key task of object detection is to correctly identify objects (e.g., humans, animals, vehicles, and logo text) in a picture and to determine the location of the object [1]. By means of a rectangular edge box, to locate the detected object and to distinguish between classes of objects. Object detection has an important role in industrial scenes, scientific research, etc. And similarly, other tasks such as classification, segmentation,

motion estimation, and scene understanding are also fundamental problems in computer vision [2].

Most traditional object detection algorithms are constructed based on artificially constructed features [3], similar to the Viola-Jones detector [4], *Histogram of Oriented Gradients* (HOG) [5] etc. The structure of these early traditional algorithms is generally divided into three steps: informative region selection, feature extraction and classification [6], such models have obvious drawbacks and shortcomings, such as not fast convergence and poor migration ability on new datasets.

The development of deep learning [7-10] and storage technology [11-12] has promoted the breakthrough of target detection. DCNN has excellent feature extraction and data migration capabilities, and its emergence has changed the field of object detection. The DCNN network AlexNet [13] was introduced in 2012, which has since opened the era of deep learning research boom.

In this paper, deep learning-based object detection techniques are reviewed and sorted out, and the main part will be organized as described below. In Section 2, the main deep convolutional neural network models are reviewed, and their architectures and performances are concisely described. Section 3 summarizes important object detection algorithms from the past to the present, and their structures are carefully analyzed and compared. Section IV reviews the commonly used datasets and evaluation criteria in object detection. Section V summarizes the main current application. Section VI concludes the paper.

2 Backbone network for object detection

The rise of deep learning [14-16], big data [17-18], computer capability [19-21], and cloud computing [22-23], has also led to the development of object detection. In the object detection task, we usually use convolutional neural networks to extract features from images for subsequent recognition and localization of objects, which is a very important part of the object detection field. In the following, we will focus on reviewing landmark networks in deep learning.

2.1 AlexNet

AlexNet is one of the seminal works in the field of deep learning, which opened a new era of modern deep learning. The earliest convolutional neural network was LeNet [24], which perfectly solved the handwritten digit recognition task and achieved an average accuracy of 98% on the MNIST dataset. Alexet uses a deeper and wider network compared to LeNet, consisting of five convolutional layers, three maximum pooling layers, and three fully connected layers. Each convolution layer uses multiple channels to enhance information processing capability. The activation function between the intermediate layers is performed by relu to speed up the model convergence, while a new regularization technique dropout is used on the first two fully connected layers in order to cope with the overfitting problem. The last fully connected layer is passed through softmax, which produces a vector of size 1000 to represent the distribution of categories. AlexNet achieved the best result on the ImageNet LSVRC-2010 dataset at that time, with an accuracy rate of 62.5% and 83% on top-1 and top-5 classifications.

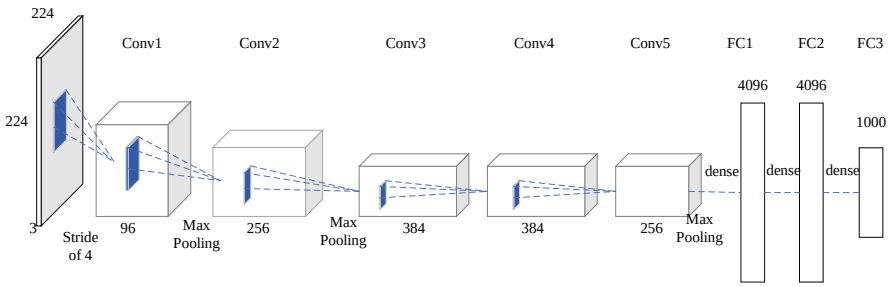


Fig.1. Schematic diagram of AlexNet structure.

2.2 VGG

VGG has deeper layers and more parameters than AlexNet. VGG uses a convolutional kernel of 3×3 with a step size of 1 to get more information and details about the object from the picture, and uses the ReLu Layer after each convolutional layer. The dominant structures are VGG16 and VGG19, the former consisting of 13 convolutional layers and 3 fully connected layers, and the latter consisting of 16 convolutional layers and 3 fully connected layers.[25] Combining multi-crop and dense evaluation, using scale jittering to resize the images, we achieved second place in the classification task of the ILSVRC-2014 challenge with an error rate of 7.3% and won first place in the localization task. VGG proved at that time that a deeper and wider network would lead to higher accuracy.

2.3 GoogLeNet

Since the start of the deep learning boom, convolutional networks have moved in a deeper and broader direction with more parameters. But larger models also require higher computational costs and are more likely to cause overfitting of data, so how to achieve high accuracy with lower computational costs is the core goal of GoogLeNet [26]. Inception block is an important concept in GoogLeNet. The structure of the Inception model consists of four parallel paths with different sizes of convolutional kernels (5×5, 3×3, 1×1) to extract information simultaneously, 1×1 convolutional kernel can change the dimension and reduce the parameters while achieving the purpose of deepening the network and interacting with information across channels [27]. GoogLeNet won the ILSVRC-2014 challenge, outperforming other networks of the same period with an error rate of 6% in the classification task 28.

2.4 Resnet

The deeper layers will cause gradient disappearance, gradient explosion, and degradation problems will occur, and 56 layers will not even perform as well as 26 layers. Resnet introduced Batch Normalization [29] to solve the gradient disappearance and gradient explosion, and proposed residual to solve the degradation problem. The residual module notates the mapping stacked by convolutional layers as H(x), and the first input of these layers is noted as x. By way of a shortcut connection, we can choose

to skip some layers. Rather than expect stacked layers to approximate H(x), we explicitly let these layers approximate a residual function F(x):=H(x)-x [30]. The advantage of this is that at least it does not make the parameters worse, it is able to train deeper models, effectively solving the degradation problem, and the depth of the model can even reach more than 1000 layers. Resnet won first place in all kinds of tasks in the ILSVRC 2015 competition [30]. Some subsequent networks, similar to Densenet [31], and ShuffLeNet [32], were also inspired by the ideas in Resnet and were born.

3 The architectures of object detection

Object detection is not limited to the identification of a particular object, but requires the detection and localization of many objects within an image. Traditional object detection algorithms usually require manual efforts to extract features, which has inherent drawbacks such as poor performance on new datasets and inefficient operation. DCNNs employ deep convolutional networks to automatically extract object features, greatly improving efficiency and speed. In the following, we review the important algorithms and models based on DCNN in the field of object detection.

Table 1. Comparison of mainstream algorithms for target detection.

Methods	Backbone	Highlights	Year
R-CNN	AlexNet	The first application of deep neural networks to object detection	2014
Fast R-CNN	VGG16	Put the whole image and its bounding boxes into the neural network	2015
Faster R-CNN	VGG16	Use the neural network to generate proposals to improve efficiency	2016
Yolo	GoogLeNet(Modified)	A neural network is used to complete the work of bounding box generation and feature extraction	2015
SSD	VGG-16	The accuracy is guaranteed while the speed is guaranteed.	2016

3.1 TWO STAGE

R-CNN. R-CNN [33] is the first model that successfully applies deep learning to object detection, and the module is designed as described below. 2000 region proposals are obtained by Selective search, fixing all region proposals to the same size, then applying Alext to each region proposal for feature extraction and outputting a vector of 4096 sizes, and finally classifying them by a trained SVM to remove the candidate bounding

boxes with IOU values larger than a threshold by NMS. Finally, a trained regression model is used to predict the correction of its bounding box by training four parameters, centroid, height and width.

R-CNN obtained the best results in the then VOC2007, VOC2010 and other object detection challenge competitions. However, R-CNN has to perform feature extraction for all 2000 region proposals, and there are many crossovers between the region proposals and redundant feature extraction operations, resulting in slow speed, large space occupation, and the need to train AlexNet, SVM and regressor individually.

Fast R-CNN. In response to a series of problems with R-CNN, Fast R-CNN was born in 2015. Fast R-CNN uses VGG16 as the Backbone of the network, which not only makes a breakthrough in speed but also improves the accuracy rate than before.

The specific operation is region selection first (this step is consistent with R-CNN), unlike R-CNN which puts each region proposal for feature extraction, Fast R-CNN extracts features by putting the whole image and the region proposals on the image into VGG16 at once. The Rol pooling layer [34] is used for selecting the region of interest and feeding the resulting feature vectors into two fully connected layers. One of which is responsible for discriminating the category and one is responsible for locating the anchor box to the correct position. The Fast RCNN uses multi-task loss jointly train classification and bounding-box regression so that two tasks share convolution features.

Faster R-CNN. Although Fast R-CNN has been effective, traditional methods of generating candidate bounding boxes such as selective search still have the problem of taking a long time. Faster R-CNN generates detection boxes directly using RPN [35] based on Fast R-CNN (RPN is a fully connected neural network), which further improves the running speed. This is done as follows: (1) generating a large number of anchors (2) RPN determines whether all the anchors contain objects, but not their categories, and (3) adjusting the positions of the anchors to get more reasonable proposals. the ROI pooling layer is used to adjust the vectors to a uniform size, and then output to the fully connected layer. the Faster R-CNNN is different from the Compared with previous R-CNN series, the region selection task also uses a deep learning approach, which greatly improves the operational efficiency.

3.2 ONE STAGE

Yolo. Traditional two-stage models need two steps to generate a bounding-box and predict object types. In contrast, Yolo is an end-to-end model in which the predicted bounding boxes and object classes are obtained by only one network.

Yolo's Backbone framework is inspired by the GoogLeNet model [36] and consists of 24 convolutional layers, and 2 fully connected layers, but instead of using Inception blocks, a 1×1 convolution is used behind a 3×3 convolution. The full connected layer vector in Yolo is Reshaped into a three-dimensional tensor with a size of $7 \times 7 \times 30$ [36], which is responsible for predicting the object if its center falls in a cell. 7×7 represents the division of the image into 7×7 cells, and 30 represents the generation

of two bounding boxes per cell, each predicting five values (c, x, y, w, h), and 20 categories.

Yolo is inferior to Faster R-CNN in terms of accuracy, but faster than Faster R-CNN. The Yolo positioning accuracy is not enough, multiple targets are close together, the target is too small, and the detection effect is not good. J. Redmon et al. subsequently proposed yolov2 [37], yolov3 [38].

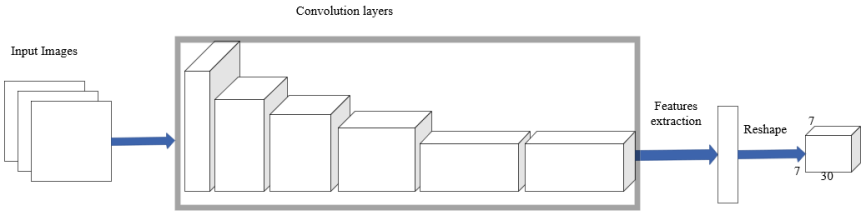


Fig.2. Schematic diagram of Yolo structure.

SSD. In order to ensure the speed and accuracy at the same time, Liu W et al. proposed SSD, which is the same as the popular detection model nowadays, SSD combines the whole detection process into a single deep neural network, combining the advantages of Faster R-CNN and Yolo, and its speed is faster than the one stage model of the same period, yolo v1 is faster and has higher accuracy, 59 FPS with MAP 74.3 % on VOC2007 test, vs Faster R-CNN 7 FPS with MAP 73.2 % or YOLO 45 FPS with MAP 63.4 % [39].

The Backbone of SSD is VGG16, which uses a base network to extract features, generating multiple anchor frames at each pixel on a feature map at different scales, predicting bounding boxes and categories for each anchor frame. The convolution layer halves the height and width of the input image to arrive at fitting small objects with the bottom layer and large objects with the top layer.

4 Datasets and evaluation criteria

4.1 Datasets

Datasets are an important part of the object detection task to train parameters and evaluate models. This subsection will systematically review the classic datasets that have made outstanding contributions and advanced research in the field of object detection.

The creation of the VOC (2005-2012) challenge made an important contribution to the development of the computer vision field. The most commonly used ones are VOC07 and VOC12, which have been extended to 20 classes of objects compared to VOC05. The training set size of VOC07 [40] has been increased to 5k and has more than 12k labeled objects. In contrast, the training set size of VOC12 reached 11k and had 16k labeled objects [41].

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (2010-2017) [42], which had made irreplaceable contributions to the development of image

classification, target detection and other fields. The ImageNet dataset, created under the auspices of Stanford professor Feifei Li, contains over 14 million tagged images, and 1000 classes of objects, including over 500k images for the target detection class and 200 classes.

MS-COCO is the most challenging dataset at present, with a huge scale and a high status in the industry, mainly used for object detection, instance segmentation and other scenarios.MS-COCO has fewer categories than ImageNet, but more instances per category, with more than 2.5 million tags in 320,000 images, containing 91 common object categories, 82 of which have more than 5,000 tagged instances [43].

Open images is a dataset built by Google that launched its first version in 2016 and includes about 6,000 categories and over 9 million images. In 2018, Google launched Open Images V4, which contains 15.4 million border boxes for 600 categories on 1.9 million images [44].

4.2 Evaluation criteria

Evaluation criteria are used to measure how good the network is on the dataset. There are many different kinds of evaluation criteria in the object detection task, such as recall, accuracy, mean average precision (MAP), FPS, etc. The following is an analysis of the evaluation criteria in object detection data.

Intersection over union (IOU) is the ratio of intersection and union of predicted and true bounding-box. If the IOU is greater than the threshold value, the prediction is considered as True Positive(TP), and if the IOU is less than the threshold value, the prediction is considered as False Positive(FP). If the object in the bounding box is not detected by the model, it is recorded as False Negative (FN). Precision measures the percentage of correct predictions while recall measures the correct predictions with respect to the ground truth 2.

$$Precision = TP/(TP + FP) \tag{1}$$

$$Recall = TP/(TP + FN) \tag{2}$$

Based on the above equation, Average Precision is calculated for each class separately. Average Precision of all classes is averaged to obtain *mean Average Precision* (mAP) and MAP is used to compare the performance between detectors.

5 Applications

5.1 Face Detection

Face detection has been an important application scenario in the field of object detection, where the task goal is to find out the face in the image and determine its location, and the traditional face detection is mainly done by manually extracting the face features and then using a sliding window to match out the face in the image, where the representative algorithm is VJ detector [45]. Object detection has achieved great success since it entered the era of deep learning, and face detection algorithms are

inextricably linked with general-purpose object detection algorithms such as the RCNN family. A cascaded CNN containing multiple cascaded DCNN classifiers was proposed [46], which improves the speed of face detection and solves the problems caused by illumination and angle in some realistic applications. To improve the problem of multi-pose and face occlusion recognition, [47] was proposed subsequently.

5.2 Text Detection

Text is one of the most important information carriers in human society and is a necessary part of people's lives. Text detection in images has important applications in many aspects, such as intelligent traffic, recognizing road signs and slogans. Used for information extraction, automatic recognition of text in natural scenes can save a lot of resources and protect customer privacy.

In the early days, text detection was usually extracted manually, but in the era of deep learning, features are usually extracted automatically using neural networks. This has greatly improved efficiency and simplified the workflow.

Mainstream text detection is divided into two ways, one is to first detect the text with generic object detection and then identify the text content, including image pre-processing, feature representation, sequence modeling (or character segmentation recognition), and prediction. Among the representative algorithms are [48] and others. And one is the end-to-end recognition approach, in which text detection and text recognition were previously divided into two separate problems, while end-to-end systems unite them into one, and recently, building real-time and efficient end-to-end systems has become a new trend in the community [49].

6 Conclusions

This paper reviewed the evolution of object detection, focusing on the contribution of deep learning-based object detection to the industry and research development, as well as comparing its advantages and where it has advanced compared to traditional approaches. The architecture of landmark backbone networks for target detection, such as AlexNet, VGG, GoogleNet, and ResNet was analyzed. Deep learning-based target detection algorithms such as R-CNN, Fast R-CNN, and Faster R-CNN were summarized and reviewed, and their differences from traditional object detection algorithms are analyzed and their characteristics were compared. The architectures of One Stage algorithms YOLO and SSD were concisely outlined, and their advantages and disadvantages were compared with Two Stage algorithms, and their operational effects were illustrated. Four major datasets in the field of object detection were introduced and the evaluation criteria of the models were parsed. Finally, a summary of the classic applications in target detection.

References

1. Xiao Y, Tian Z, Yu J, et al. A review of object detection based on deep learning[J]. Multimedia Tools and Applications, 2020, 79(33): 23729-23791.

2. Zaidi S S A, Ansari M S, Aslam A, et al. A survey of modern deep learning based object detection models[J]. Digital Signal Processing, 2022: 103514.
3. Zou Z, Shi Z, Guo Y, et al. Object detection in 20 years: A survey[J]. arXiv preprint arXiv:1905.05055, 2019.
4. Viola P, Jones M. Rapid object detection using a boosted cascade of simple features, IEEE CVPR 2001, 1: I-I.
5. Dalal N, Triggs B. Histograms of oriented gradients for human detection[C]//2005 IEEE CVPR, 2005, 1: 886-893.3
6. Zhao Z Q, Zheng P, Xu S, et al. Object detection with deep learning: A review[J]. IEEE transactions on neural networks and learning systems, 2019, 30(11): 3212-3232.
7. Liang W, Long J, Li K C, et al. A fast defogging image recognition algorithm based on bilateral hybrid filtering[J]. ACM TOMM, 2021, 17(2): 1-16.
8. Diao C, Zhang D, Liang W, et al. A Novel Spatial-Temporal Multi-Scale Alignment Graph Neural Network Security Model for Vehicles Prediction. IEEE TITS 2022.
9. Peng L, Peng M, Liao B, et al. Improved low-rank matrix recovery method for predicting miRNA-disease association[J]. Scientific reports, 2017, 7(1): 1-10.
10. Xiao W, Tang Z, Yang C, et al. ASM-VoFDehaze: a real-time defogging method of zinc froth image[J]. Connection Science, 2022, 34(1): 709-731.
11. Wang J, Luo W, Liang W, et al. Locally minimum storage regenerating codes in distributed cloud storage systems[J]. China Communications, 2017, 14(11): 82-91.
12. Liang W, Huang Y, Xu J, et al. A distributed data secure transmission scheme in wireless sensor network[J]. Int'l J. of Distrib. Sensor Netw., 2017, 13(4): 1550147717705552.
13. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classifification with deep convolutional neural networks," in NIPS, 2012.
14. H. Qiu, T. Dong, et al., "Adversarial attacks against network intrusion detection in IoT systems," IEEE Internet of Things Journal 8(13), 10327-10335, 2020
15. H. Qiu, Q. Zheng, et al., "Topological graph convolutional network-based urban traffic flow and density prediction", IEEE Trans. on ITS, 2020
16. F. Hu, S. Lakdawala, et al., Low-power, intelligent sensor hardware interface for medical data preprocessing, IEEE Trans. on Info. Tech. in Biome. 13 (4), 656-663, 2009
17. K. Gai, M. Qiu, S. Elnagdy, "A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance," IEEE BigDataSecurity 2016
18. Y. Li, K. Gai, et al. "Intercrossed access controls for secure financial services on multimedia big data in cloud systems", ACM TMCCA, 2016
19. M. Qiu, Z. Chen, Z. Ming, X. Qin, J. Niu, "Energy-aware data allocation with hybrid memory for mobile cloud systems", IEEE Systems J., 11 (2), 813-822, 2014
20. M. Qiu, C. Xue, Z. Shao, E. Sha, "Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems," IEEE DATE Conf., 1-6, 2007
21. M. Qiu, Z. Jia, et al., "Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP", JSPS, Spinger, 2007
22. J. Niu, Y. Gao, M. Qiu, Z. Ming, "Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability", JPDC, 72(12), 1565-1575, 2012
23. J. Li, Z. Ming, et al., "Resource allocation robustness in multi-embedded systems with inaccurate information", Journal of Systems Architecture 57 (9), ore e840-849, 2011
24. LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.
25. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.
26. Dhillon A, Verma G K. Convolutional neural network: a review of models, methodologies

and applications to object detection, Progress in AI, 2020, 9(2): 85-112.
27.Liu L, Ouyang W, Wang X, et al. Deep learning for generic object detection: A survey[J]. International journal of computer vision, 2020, 128(2): 261-318.
28.Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 1-9.
29.Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift, Int'l conf. on mach. learning. PMLR, 2015: 448-456.
30.He K, Zhang X, Ren S, et al. Deep residual learning for image recognition, IEEE CVPR. 2016: 770-778.
31.Huang G, Liu Z, Van Der Maaten L, et al. Densely connected convolutional networks, IEEE CVPR. 2017: 4700-4708.
32.Zhang X, Zhou X, Lin M, et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices, IEEE CVPR. 2018: 6848-6856.
33.Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation, IEEE CVPR. 2014: 580-587.
34.Girshick R. Fast r-cnn[C]//Proceedings of the IEEE international conference on computer vision. 2015: 1440-1448.
35.Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[J]. Advances in neural info. processing systems, 2015, 28.
36.Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection, IEEE conf. on computer vision and pattern recognition. 2016: 779-788.
37.Redmon J, Farhadi A. YOLO9000: better, faster, stronger[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 7263-7271.
38.Redmon J, Farhadi A. Yolov3: An incremental improvement[J]. arXiv preprint arXiv:1804.02767, 2018.
39.Liu W, Anguelov D, Erhan D, et al. Ssd: Single shot multibox detector[C]//European conference on computer vision. Springer, Cham, 2016: 21-37.
40.Everingham M, Winn J. The PASCAL visual object classes challenge 2007 (VOC2007) development kit[J]. Int. J. Comput. Vis, 2010, 88(2): 303-338.
41.Everingham M, Winn J. The PASCAL visual object classes challenge 2012 (VOC2012) development kit. Pattern Anal. Stat. Model. Comp. Learn., Tech. Rep, 2012, 2007: 1-45.
42.Russakovsky O, Deng J, Su H, et al. Imagenet large scale visual recognition challenge[J]. International journal of computer vision, 2015, 115(3): 211-252.
43.Lin T Y, Maire M, Belongie S, et al. Microsoft coco: Common objects in context[C]//European conference on computer vision. Springer, Cham, 2014: 740-755.
44.Kuznetsova A, Rom H, Alldrin N, et al. The open images dataset v4[J]. International Journal of Computer Vision, 2020, 128(7): 1956-1981.
45.Viola P, Jones M. Rapid object detection using a boosted cascade of simple features, IEEE CVPR 2001. 1: I-I.
46.Li H, Lin Z, Shen X, et al. A convolutional neural network cascade for face detection, IEEE CVP. 2015: 5325-5334.
47.Shi X, Shan S, Kan M, et al. Real-time rotation-invariant face detection with progressive calibration networks, IEEE CVPR. 2018: 2295-2303.
48.Wang T, Wu D J, Coates A, et al. End-to-end text recognition with convolutional neural networks, 21st IEEE ICPR, 2012: 3304-3308.
49.Chen X, Jin L, Zhu Y, et al. Text recognition in the wild: A survey[J]. ACM Computing Surveys (CSUR), 2021, 54(2): 1-35.

A Decision-Making Method for Blockchain Platforms Using Axiomatic Design

Jun Liu[1[0000-0001-7390-8958]] , Qi Zhang[1[0000-0003-4443-0056]], Ming-Yue Xie[1[0000-0001-7390-8958]] and Ming-Peng Chen[1[0000-0001-7390-8958]]

Chongqing University of Posts and Telecommunications, Chongqing 400065, China

Abstract. For companies using blockchain technology, it is critical to select the most suitable blockchain platform to develop enterprise applications. However, it is still a challenge for enterprises. As an important part of modern decision science, multi-criteria decision-making can well solve the problem of blockchain platform selection. Blockchain platforms integrate various blockchain technologies, and enterprises also need to consider multiple different criteria in the decision-making process. Therefore, this paper will use a heterogeneous multi-criteria decision-making method to solve the blockchain platform selection problem. First, the blockchain platform alternatives and evaluation criteria used for decision-making are identified. Second, blockchain platform alternatives are evaluated with appropriate fuzzy numbers based on defined evaluation criteria. Then, the original evaluations are consistency and normalized to obtain a normalized evaluation. Next, the improved information content formulas of the axiomatic design is proposed to obtain the information content of each normalized evaluation. Then, the weights of all evaluated criteria are obtained using the entropy weight method. Finally, the total weighted information content of each blockchain platform alternative is obtained. With validation, the decision-making model of blockchain platform proposed in this paper has a strong reference value.

Keywords: Blockchain platform, Multi-criteria decision-making, Heterogeneous information, Axiomatic design, Information content, Entropy Weight Method.

1 Introduction

As a novel technology combination integrating existing technologies such as computing power [1-3], peer-to-peer communication [4-6], cryptography [7-9], and distributed data storage [10-12], blockchain [13-14] is widely used by numerous enterprises because of its characteristics of anonymity, openness, security, information immutability, and decentralization [15-17]. Blockchain has gone through three generations since 2008. The latest generation of blockchain is not limited to the financial industry [18], but has been widely used in various industries, such as the Internet of Things, logistics [19], public services, digital rights, and insurance [20-22]. In these contexts, developing blockchain applications for various industries is the

current focus [23-25]. As a result, industries are increasingly inseparable from blockchain technology [26,27]. Blockchain platforms integrate various blockchain technologies and can shorten the development and deployment cycle of blockchain applications in various industries. Therefore, it is more beneficial to use existing blockchain platforms to develop blockchain applications.

With the development of blockchain technology, there are a large number of blockchain platforms on the market and are used in various businesses of enterprises. However, with the rapid increase in the number of blockchain platforms, it has become a challenge for enterprises to select the suitable blockchain platform. Different blockchain platforms provide different functions, and the functions required by enterprises in different industries are also different. As a decision-making method, multi-criteria decision-making can well solve the decision-making problem of blockchain platforms. Supporting this argument, paper [28] proposed a decision-making model for blockchain platforms, [29] proposed a decision-making framework for business blockchain platforms, and [30] proposed a decision-making model for public blockchain platforms.

In order to solve the problem of blockchain platform selection, this paper proposes an axiomatic design-based blockchain platform decision-making method with the following main contributions:

- Transforming the blockchain platform selection problem into a heterogeneous multi-criteria decision-making.
- Considering that one type of fuzzy number [31-36] is not sufficient to express multiple criteria, this paper provides interval number, real number, and triangular fuzzy number to express the evaluation information based on different criteria.
- The formula for the information content of the axiomatic design is improved, and its value falls between 0 and 1, and the larger the better. It can avoid the situation that the weights of the evaluated criteria cannot be calculated and the ranking of the solved alternatives is not reasonable when the information content is ∞ [36-38].

The rest of the paper is organized as follows. Section 2 introduces related concepts, including entropy weight method, and axiomatic design. Section 3 describes the decision-making process of blockchain platform based on axiomatic design, and Section 4 summarizes the whole paper.

2 Preliminaries

In this section, we will briefly introduce some necessary basics related to this thesis. Firstly, the concept of entropy weight method will be introduced. Finally, the concepts related to axiomatic design will be introduced.

2.1 The concept of entropy weight method

The entropy weight method is an objective method to determine the weight of criteria. In the entropy weight method, according to the definition of information entropy, for a certain index, the entropy value can be used to determine the dispersion degree of a certain index, the smaller the information entropy value, the greater the dispersion degree of the index, the greater the influence (i.e. weight) of the index on the comprehensive evaluation, if all the values of a certain index are equal, the index does not work in the comprehensive evaluation. The calculation steps of the entropy weight method are as follows.

Step 1: Obtain the normalized matrix with n rows and m columns.

Step 2: Calculate the entropy value e_j of criterion *j*.

$$e_j = -l\sum_{i=1}^{n} p_{ij} \ln p_{ij}, p_{ij} = \frac{x_{ij}}{\sum_{i=1}^{n} x_{ij}}, l = \frac{1}{\ln n}, 0 \le e_j \le 1 \tag{1}$$

Step 3: Calculate the degree of divergence(d_j) of average intrinsic information contained by criterion *j*.

$$d_j = 1 - e_j \tag{2}$$

Step 4: Calculate the weight W_j for criterion *j*.

$$W_j = \frac{d_j}{\sum_{j=1}^{m} d_j} \tag{3}$$

2.2 The concept of axiomatic design

Axiomatic design is proposed by Suh, which can provide a theoretical basis for designers to choose the most reasonable design solutions and establish scientific and systematic standards for design activities.

The most important concept in axiomatic design is the existence of design axioms, the first one being the axiom of independence and the second one being the axiom of information. These axioms are expressed as follows:

1. Independence Axiom: Maintain the independence of functional requirements (FRs).
2. Information axiom: Minimize information content.

The independence axiom states that the independence of functional requirements (FRs) must always be maintained, and that FRs are defined as the smallest set of independent requirements that describe the design goals.

The information axiom states that the design with the least information content is the best design among the designs that satisfy the independent axiom. The information content is the probability of satisfying a given functional requirement. For a functional requirement FR_m , its information content I_m is defined as follows:

$$I_m = \log_2(\frac{1}{P_m}) \tag{4}$$

P_m is the probability of achieving the functional requirement FR_m , the notation "log" means that the logarithm is base 2 logarithm. The probability of achieving a functional requirement is determined by both the degree to which the designer wants to achieve it (the design range) and the degree to which the design can achieve it (the system range). The common part of the design range and system range is the area where an acceptable solution exists. As shown in Fig. 1. Therefore, under the uniform probability density function:

$$P_m = \frac{common\ range}{system\ range} \tag{5}$$

Then the information content I_m can be defined as:

$$I_m = \log_2(\frac{system\ range}{common\ range}) \tag{6}$$

Based on the axiomatic design, we can get that in a multi-criteria decision-making, the design range of benefit criteria is the maximum evaluated value of the alternative under the corresponding criteria. The cost criteria can be transformed into benefit criteria. The independence axiom of axiomatic design requires that functional requirements are independent of each other, and therefore the criteria in this research are independent of each other.

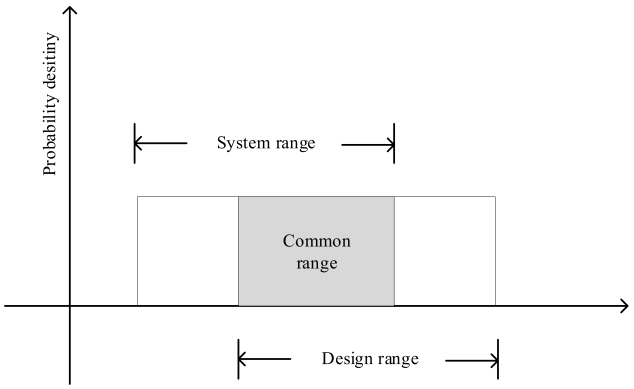


Fig. 1. Design range, system range, common range and probability density function of FR

3 Decision-making process for blockchain platforms

In this section, we will introduce the decision-making process for blockchain platforms. The steps of decision-making include: determining the blockchain platform alternatives and evaluation criteria, evaluating all blockchain platforms based on each evaluation criterion to get the original evaluation matrix, consistency and normalized the original evaluation matrix to get the normalized evaluation matrix, calculating the information content of each element in the normalized evaluation matrix, using the entropy weight method to get the weight of each evaluation criterion, calculate the total weighted information content of each blockchain platform alternative, and rank the blockchain platform alternatives. The decision-making process is described in Fig. 2, and the detailed decision-making process is described below.

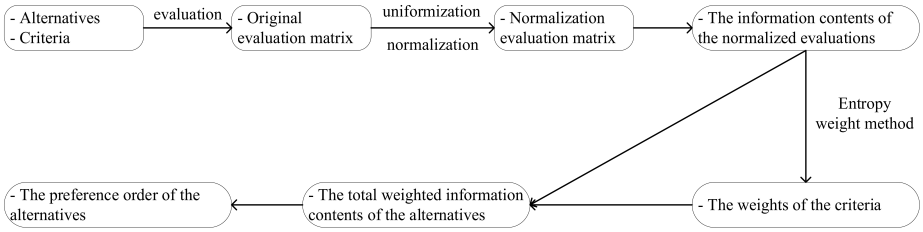


Fig. 2. Decision-making Process for Blockchain Platform Alternatives

Step 1: Let the n blockchain platform alternatives be $\{A_1, A_2, ..., A_n\}$ and the m evaluation criteria be $\{C_1, C_2, ..., C_m\}$.

Step 2: Constructing the original evaluation matrix.

The evaluation of n blockchain platform alternatives based on m evaluation criteria leads to the following decision matrix.

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{pmatrix}$$

a_{nm} denotes the evaluation of the nth blockchain platform alternative A_n based on the mth evaluation criterion C_m, where $0 \le a_{ij} \le 1, 1 \le i \le n, 1 \le j \le m$.

Step 3: Constructing the normalized evaluation matrix.

In this paper, the evaluation criteria can be divided into benefit-type and cost-type. It can be seen that the elements in the original evaluation matrix A are divided into two categories, one belonging to the cost-type and the rest belonging to the benefit-

type. Next, the elements in the matrix A are consistent and normalized, all the elements are transformed into benefit-type elements.

For the interval number $a_{ij} = [k_{ij}, l_{ij}]$ in the original evaluation matrix, where $0 \le k_{ij} \le l_{ij} \le 1$, the following equation can be obtained, where b_{ij} is the normalized evaluation of $a_{ij} = [k_{ij}, l_{ij}]$.

$$b_{ij} = \begin{cases} [\dfrac{k_{ij} - k_j^{\min}}{l_j^{\max} - k_j^{\min}}, \dfrac{l_{ij} - k_j^{\min}}{l_j^{\max} - k_j^{\min}}] & (benefit\ criterion) \\ [\dfrac{l_j^{\max} - l_{ij}}{l_j^{\max} - k_j^{\min}}, \dfrac{l_j^{\max} - k_{ij}}{l_j^{\max} - k_j^{\min}}] & (\text{cost}\ criterion) \end{cases} \tag{7}$$

For the real number $a_{ij} = r_{ij}$ in the original evaluation matrix, where $0 \le r_{ij} \le 1$, the following equation can be obtained, where b_{ij} is the normalized evaluation of $a_{ij} = r_{ij}$.

$$b_{ij} = \begin{cases} \dfrac{r_{ij} - r_j^{\min}}{r_j^{\max} - r_j^{\min}} & (benefit\ criterion) \\ \dfrac{r_j^{\max} - r_{ij}}{r_j^{\max} - r_j^{\min}} & (\text{cost}\ criterion) \end{cases} \tag{8}$$

For the triangular fuzzy number $a_{ij} = (m_{ij}, n_{ij}, p_{ij})$ in the original evaluation matrix, where $0 \le m_{ij} \le n_{ij} \le p_{ij} \le 1$, the following equation can be obtained, where b_{ij} is the normalized evaluation of $a_{ij} = (m_{ij}, n_{ij}, p_{ij})$.

$$b_{ij} = \begin{cases} (\dfrac{m_{ij} - m_j^{\min}}{p_j^{\max} - m_j^{\min}}, \dfrac{n_{ij} - m_j^{\min}}{p_j^{\max} - m_j^{\min}}, \dfrac{p_{ij} - m_j^{\min}}{p_j^{\max} - m_j^{\min}}) & (benefit\ criterion) \\ (\dfrac{p_j^{\max} - p_{ij}}{p_j^{\max} - m_j^{\min}}, \dfrac{p_j^{\max} - n_{ij}}{p_j^{\max} - m_j^{\min}}, \dfrac{p_j^{\max} - m_{ij}}{p_j^{\max} - m_j^{\min}}) & (\text{cost}\ criterion) \end{cases} \tag{9}$$

Finally, the following decision matrix can be obtained.

$$B = \begin{pmatrix} b_{11} & \cdots & b_{1m} \\ \vdots & \ddots & \vdots \\ b_{n1} & \cdots & b_{nm} \end{pmatrix}$$

where b_{nm} denotes the normalized evaluation obtained after consistent and normalized processing of the a_{nm} in the original evaluation matrix, where $0 \le b_{ij} \le 1, 1 \le i \le n, 1 \le j \le m$.

Step 4: Calculate the information content in each element of the normalized evaluation matrix.

For the interval numbers $b_{ij} = [\overline{k_{ij}}, \overline{l_{ij}}]$ and $b_j^* = [\overline{k_{ij\,\max}}, \overline{l_{ij\,\max}}]$, where $0 \le \overline{k_{ij}} \le \overline{l_{ij}} \le 1$, $0 \le \overline{k_{ij\,\max}} \le \overline{l_{ij\,\max}} \le 1$, the following equation can be obtained, where $I'(b_{ij}, b_j^*)$ is the information content of $b_{ij} = [\overline{k_{ij}}, \overline{l_{ij}}]$.

$$I'(b_{ij}, b_j^*) = \begin{cases} \log_2\left(1 + \dfrac{\overline{l_{ij}} - \overline{k_{ij\,\max}}}{\overline{l_{ij}} - \overline{k_{ij}}}\right) & (\overline{k_{ij\,\max}} < \overline{l_{ij}}) \\ 0 & (\overline{l_{ij}} \le \overline{k_{ij\,\max}}) \end{cases} \tag{10}$$

For the real numbers $b_{ij} = \overline{r_{ij}}$ and $b_j^* = \overline{r_{ij\,\max}}$, where $0 \le \overline{r_{ij}} \le 1$, $0 \le \overline{r_{ij\,\max}} \le 1$, the following equation can be obtained, where $I'(b_{ij}, b_j^*)$ is the information content of $b_{ij} = \overline{r_{ij}}$.

$$I'(b_{ij}, b_j^*) = \log_2\left(1 + \frac{\overline{r_{ij}}}{\overline{r_{ij\,\max}}}\right) \quad (\overline{r_{ij}} \le \overline{r_{ij\,\max}}) \tag{11}$$

For the triangular fuzzy numbers $b_{ij} = (\overline{m_{ij}}, \overline{n_{ij}}, \overline{p_{ij}})$ and $b_j^* = (\overline{m_{ij\,\max}}, \overline{n_{ij\,\max}}, \overline{p_{ij\,\max}})$, where $0 \le \overline{m_{ij}} \le \overline{n_{ij}} \le \overline{p_{ij}} \le 1$, $0 \le \overline{m_{ij\,\max}} \le \overline{n_{ij\,\max}} \le \overline{p_{ij\,\max}} \le 1$, the following equation can be obtained, where $I'(b_{ij}, b_j^*)$ is the information content of $b_{ij} = (\overline{m_{ij}}, \overline{n_{ij}}, \overline{p_{ij}})$.

$$I'(b_{ij}, b_j^*) = \begin{cases} \log_2\left(1 + \dfrac{(\overline{p_{ij}} - \overline{m_{ij\,\max}})^2}{(\overline{p_{ij}} - \overline{m_{ij}})(\overline{p_{ij}} - \overline{n_{ij}} + \overline{n_{ij\,\max}} - \overline{m_{ij\,\max}})}\right) & (\overline{p_{ij}} > \overline{m_{ij\,\max}}) \\ 0 & (\overline{p_{ij}} \le \overline{m_{ij\,\max}}) \end{cases} \tag{12}$$

Step 5: Calculate the weight of each evaluation criterion using the entropy weight method.

Step 5.1: Constructing the matrix for the calculation, the matrix can be obtained from Step 4 as follows:

$$I^{'}=\begin{pmatrix} I_{11}^{'} & \cdots & I_{1m}^{'} \\ \vdots & \ddots & \vdots \\ I_{n1}^{'} & \cdots & I_{nm}^{'} \end{pmatrix}$$

where $I_{nm}^{'}$ denotes the information content of b_{nm} in the normalized evaluation matrix, where $0 \le I_{ij}^{'} \le 1, 1 \le i \le n, 1 \le j \le m$.

Step 5.2: Calculate the entropy of each evaluation criterion.

$$e_j = -h\sum_{i=1}^{n} p_{ij} \ln p_{ij}, p_{ij} = \frac{I_{ij}^{'}}{\sum_{i=1}^{n} I_{ij}^{'}}, h = \frac{1}{\ln n}, 0 \le e_j \le 1 \tag{13}$$

Step 5.3: Calculate the degree of divergence of average intrinsic information contained for each evaluation criterion.

$$d_j = 1 - e_j \tag{14}$$

Step 5.4: Calculate the weights for each evaluation criterion.

$$W_j = \frac{d_j}{\sum_{j=1}^{m} d_j} \tag{15}$$

Step 6: Calculate the total weighted information content for each blockchain platform alternative, where $I^{'}(A_i)$ is the total weighted information content of the alternative A_i.

$$I^{'}(A_i) = \sum_{j=1}^{m} I_{ij}^{'} W_j \tag{16}$$

Step 7: Ranking of all blockchain platform alternatives.

Each blockchain platform alternative A_i is ranked according to $I^{'}(A_i)$, and the larger the $I^{'}(A_i)$, the better the blockchain platform alternative A_i.

4 Conclusion

With the continuous development of blockchain technology, the number of blockchain platforms in the market is increasing. However, selecting the suitable blockchain platform is still a problem. To address the above background, this paper proposed a heterogeneous multi-criteria decision-making method based on axiomatic design to solve the blockchain platform selection problem. The blockchain platform decision-making method proposed in this paper can help relevant practitioners to select the suitable blockchain platform more quickly and efficiently.

References

1. M. Qiu, Z. Jia, et al., Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP, J. Signal Processing Systems, 2007
2. M. Qiu, L. Yang, Z. Shao, E. Sha, Dynamic and leakage energy minimization with soft real-time loop scheduling and voltage assignment, IEEE TVLSI, 18 (3), 501-504, 2009
3. M. Qiu, C. Xue, Z. Shao, E. Sha, Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems, IEEE DATE Conf., 1-6, 2007
4. J. Niu, Y. Gao, *et al.,* Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability, JPDC, 72(12), 1565-1575, 2012
5. M. Qiu, C Xue, Z Shao, *et al*., Efficient algorithm of energy minimization for heterogeneous wireless sensor network, IEEE EUC, 25-34, 2006
6. M. Qiu, H. Li, E. Sha, Heterogeneous real-time embedded software optimization considering hardware platform, ACM sym. on Applied Comp., 1637-1641, 2009
7. H. Qiu, T. Dong, T. Zhang, et al., Adversarial attacks against network intrusion detection in IoT systems, IEEE Internet of Things Journal 8(13), 10327-10335, 2020
8. K. Gai, M. Qiu, S. Elnagdy, A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance, IEEE BigDataSecurity 2016
9. X. Gao and M. Qiu, Energy-Based Learning for Preventing Backdoor Attack, KSEM (3) 2022: 706-721
10. M. Qiu, Z. Chen, Z. Ming, X. Qin, J. Niu, Energy-aware data allocation with hybrid memory for mobile cloud systems, IEEE Systems J., 11 (2), 813-822, 2014
11. M. Qiu, H. Qiu, *et al*.: Secure Data Sharing Through Untrusted Clouds with Blockchain-enabled Key Management, SmartBlock 2020, pp. 11-16, 2020, China.
12. J. Li, Z. Ming, *et al*., Resource allocation robustness in multi-core embedded systems with inaccurate information, Journal of Systems Arch. 57 (9), 840-849, 2011
13. M. Qiu and H. Qiu, Review on Image Processing Based Adversarial Example Defenses in Computer Vision, IEEE BigDataSecurity, pp. 94-99, Baltimore, USA, 2020.
14. K. Gai, Y. Zhang, *et al*.: Blockchain-enabled Service Optimizations in Supply Chain Digital Twin, IEEE Transactions on Service Computing, 2022
15. Xie, M.-Y., Liu, J.: A Survey on Blockchain Consensus Mechanism: Research Overview, Current Advances, and Future Directions. *International Journal of Intelligent Computing and Cybernetics*. 1-27 (2022).
16. Xie, M.-Y., Liu, J., Chen, S.-Y., Xu, G.-X., Lin, M.-W.: Primary node election based on probabilistic linguistic term set with confidence interval in the PBFT consensus mechanism for blockchain. *Complex & Intelligent Systems* doi: https://doi.org/10.1007/s40747-022-00857-9, 2022.
17. Liu, J., Xie, M.-Y., Chen, S.-Y., Ma, C., Gong, Q.-H.: An improved DPoS consensus mechanism in blockchain based on PLTS for the smart autonomous multi-robot system. *Information Sciences*. 575, 528-541 (2021).
18. Y. Li, K. Gai*, et al.,* Intercrossed access controls for secure financial services on multimedia big data in cloud systems, ACM Trans. on Multimedia Comp., Comm., and App., 2016
19. Liu, J., Zhao, J., Huang, H.-H., Xu, G.-X.: A novel logistics data privacy protection method based on blockchain. *Multimedia Tools and Applications*. 81(17), 23867-23877 (2022).
20. M., Qiu, H., Zheng, *et al.*: Topological graph convolutional network-based urban traffic flow and density prediction. *IEEE Transactions on Intelligent Transportation Systems*. 99 (2020).

21. Li, Y.-B., Song, Y., Jia, L., et al.: Intelligent fault diagnosis by fusing domain adversarial training and maximum mean discrepancy via ensemble learning. *IEEE Transactions on Industrial Informatics*. 17(4), 2833-2841 (2021).
22. Hu, F., Lakdawala, S., Hao, Q., et al.: Low-Power, Intelligent Sensor Hardware Interface for Medical Data Preprocessing. *IEEE Transactions on Information Technology in Biomedicine*. 13(4), 656-663 (2009).
23. Gai, K.-K., Wu, Y.-L., Zhu, L.-H., et al.: Permissioned Blockchain and Edge Computing Empowered Privacy-Preserving Smart Grid Networks. *IEEE Internet of Things Journal*. 6(5), 7992-8004 (2019).
24. Qiu, M.-K., Gai, K.-K., Xiong, Z.-G.: Privacy-preserving wireless communications using bipartite matching in social big data. *Future Generation Computer Systems-The International Journal of eScience*. 87, 772-781 (2018).
25. Gai, K.-K., Fang, Z.-K., Wang, R.-L., et al.: Edge Computing and Lightning Network Empowered Secure Food Supply Management. *IEEE Internet of Things Journal*. 9(16), 14247-14259 (2022).
26. Hijazi, A.A., Perera, S., Alashwal, A.M., et al.: Enabling a single source of truth through BIM and blockchain integration. In: *International Conference on Innovation, Technology, Enterprise, and Entrepreneurship 2019*. pp. 24-25(2019).
27. Perera, S., Nanayakkara, S., Rodrigo, M., et al.: Blockchain technology: Is it hype or real in the construction industry?. *Journal of Industrial Information Integration*. 17, 2020.
28. Farshidi, S., *et al.*: Decision support for blockchain platform selection: Three industry case studies. *IEEE Transactions on Engineering Management*. 67(4), 1109-1128 (2020).
29. Büyüközkan, G., Tüfekçi, G.: A decision-making framework for evaluating appropriate business blockchain platforms using multiple preference formats and VIKOR. *Information Sciences*. 571, 337-357 (2021).
30. Tang, H.-M., Shi, Y., Dong, P.-W.: Public blockchain evaluation using entropy and TOPSIS. *Expert Systems with Applications*. 117, 204-210 (2019).
31. Chen, C.-H.: A novel multi-criteria decision-making model for building material supplier selection based on entropy-AHP weighted TOPSIS. *Entropy*. 22(2), 2020.
32. Kumar, R., Bilga, P.-S., Singh, S.: Multi objective optimization using different methods of assigning weights to energy consumption responses, surface roughness and material removal rate during rough turning operation. *Journal of Cleaner Production*. 164, 45-57 (2017).
33. Chen, P.-Y.: Effects of the entropy weight on TOPSIS. *Expert Systems with Applications*. 168, 2021.
34. Khan, M.J., *et al.*: The renewable energy source selection by remoteness index-based VIKOR method for generalized intuitionistic fuzzy soft sets. *Symmetry-Basel*. 12(6), 2020.
35. Ghadikolaei, A.S., Madhoushi, M., Divsalar, M.: Extension of the VIKOR method for group decision making with extended hesitant fuzzy linguistic information. *Neural Computing & Applications*. 30(12), 3589-3602 (2018).
36. Feng, J.-H., Xu, S.-X., Li, M.: A novel multi-criteria decision-making method for selecting the site of an electric-vehicle charging station from a sustainable perspective. *Sustainable Cities and Society*. 65, 2021.
37. Chen, X., *et al.*: Matching demanders and suppliers in knowledge service: A method based on fuzzy axiomatic design. *Information Sciences*. 346, 130-145 (2016).
38. Büyüközkan, G., Karabulut, Y., Arsenyan, J.: RFID service provider selection: An integrated fuzzy MCDM approach. *Measurement*. 112, 88-98 (2017).

An Auxiliary Classifier GAN-based DDoS Defense Solution in Blockchain-based Software Defined Industrial Network

Yue Zhang[1][0000−0002−0431−6390], Keke Gai[2][0000−0001−6784−0221]⋆, Liehuang Zhu[2][0000−0003−3277−3887], and Meikang Qiu[3][0000−0002−1004−0140]

[1] School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China. 3220201017@bit.edu.cn
[2] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China. {gaikeke, liehuangz}@bit.edu.cn
[3] Beacom College of Computer and Cyber Sciences, Dakota State University, Madison, SD 57042, USA. qiumeikang@ieee.org

Abstract. As an emerging technology, *Software-Defined Industrial Networks* (SDIN) appears to be a vital technical approach for powering up new manufacturing modes due to its higher-level scalability and controllability. However, a few threats are still restricting the implementation of SDIN and *Distributed Denial of Service* (DDoS) is one of the common attacks. In this paper, we focus on the DDoS issue in SDIN, and propose a blockchain-empower SDIN scheme and an *Auxiliary Classifier Generative Adversarial Networks* (AC-GAN)-based DDoS attack detection model. Our experiment evaluations have demonstrated the effectiveness and performance of our proposed approach.

Keywords: Blockchain · Software Defined Industrial Networks · Distributed Denial of Service · Auxiliary Classifier Generative Adversarial Networks · Artificial Intelligence · security

1 Introduction

Software Defined Industrial Network (SDIN) is a novel technical paradigm that provides flexible and manageable network control capabilities for complex industrial network systems, especially heterogeneous networks deployed in cross-organizations [9]. SDIN separates the network control and data plane, overcoming limitations such as unscalability and low efficiency caused by static configuration in traditional industrial networks [2].

Despite multiple observable advantages offered by SDIN, vulnerabilities still exist in the industrial network context. Prior study [11] has argued that *Distributed Denial of Service* (DDoS) attack is one of critical threats for SDIN. First, considering the application plane in SDIN, the programmable interface is often exploited by attackers. Second, the centralized network controller is easy

⋆ Corresponding author: Keke Gai (gaikeke@bit.edu.cn)

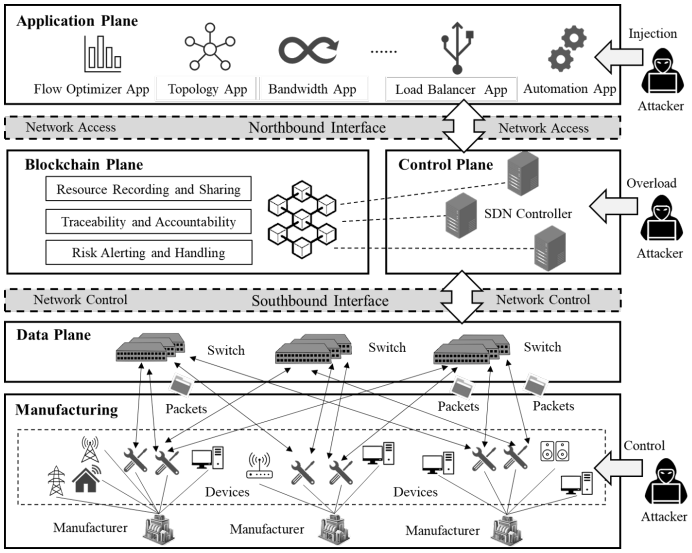


Fig. 1. The architecture of the blockchain-empowered SDIN.

to become the attack target. Finally, the threat is further enlarged along with the large amount of *Internet of Things* (IoT) devices deploying in the industrial network, as IoT devices generally offer limited defense capability [14] and are easily controlled by attackers.

To address above security issues, a variety of DDoS attack detection methods have emerged [3, 12]. Among them, deep learning has gained widespread attention due to its autonomy and accuracy. However, deep learning models tend to become sensitive and inaccurate when detecting adversarial DDoS attacks [4]. Since the distributed and traceable features of blockchain [5, 6, 8] provide novel ideas for improving SDIN security, blockchain-based security solutions have also gained popularity [1,7,13]. However, there is currently no specific defense method against DDoS attacks in the blockchain-enabled SDIN architecture.

In this work, we focus on adversarial DDoS attack detection and mitigation in the blockchain-empowered SDIN system. As illustrated in Fig. 1, we propose the architecture of blockchain-empowered SDIN, in which we also mark locations where adversaries may launch DDoS attacks. Multiple controllers act as participants in the blockchain to record application flows and network behaviors as transactions into the distributed ledger. After distributing the control plane to multiple controllers, it is necessary to ensure resource consensus among all controllers. In addition, deriving prior achievements, we propose an *Auxiliary Classifier Generative Adversarial Networks* (AC-GAN) scheme to detect adver-

[4] Adversarial attacks refer to that the attacker deliberately adds certain imperceptible interference to input samples, causing the misjudgment of the prediction model.

sarial DDoS attacks and adopt varied data flow strategies via the controller to lower down the attack opportunities.

Main contributions of this work are threefold, as given in the followings:

- This paper proposes a scheme using deep learning to defend against DDoS attacks in blockchain-empowered SDIN.
- We present a blockchain-based SDIN architecture that decentralizes the control plane to multiple controllers, thus preventing single points of failure. The behavior of application flows can be tracked and audited via the blockchain, helping to monitor and replay network state for debugging and recovery.
- We design an AC-GAN algorithm to detect three types of DDoS attacks in blockchain-based SDIN environment, including TCP, UDP and ICMP flood attacks. In particular, AC-GAN generates adversarial samples for data enhancement, which can effectively improve the detection accuracy.

The rest of this work are organized in the following order. Sections 2 and 3 describe the proposed method and core algorithms, respectively. Then, Section 4 provide our experiment evaluations and results. Finally, conclusions of this work are drawn in Section 5.

2 Model Design

2.1 Blockchain-based SDIN Model

Fig. 1 illustrates the higher-level architecture of the blockchain-based SDIN model, which consists of five parts, including application plane, control plane, blockchain plane, data plane and physical plane.

Since a single controller in SDIN is vulnerable to DDoS attacks and causes a single point of failure, we decentralize the control plane to multiple controllers based on blockchain technology [15]. Specifically, multiple controllers in SDIN serve as clients of the blockchain platform for resource sharing and transaction recording. Resource sharing means that the network resources managed by each controller can be shared after reaching a consensus among all controllers. Transaction recording refers to the storage of network traffic, behavior and state into the blockchain in the form of transactions.

In the proposed model, we design the following three types of transactions.

- **Entity registration transaction.** Each entity participating in SDIN, including applications, controllers, and switches, needs to register identity information before joining the blockchain. Through the authentication access mechanism, the attack cost of the adversary is increased.
- **Application flow transaction.** Transactions are generated when an application sends flow information to a controller, which is mainly used to record and monitor the behavior of the application.
- **Network event transaction.** Network events provided by switches also generate transactions uploaded to the blockchain, which are mainly dynamic states triggered by application events and $Packet\_IN$ messages. This kind

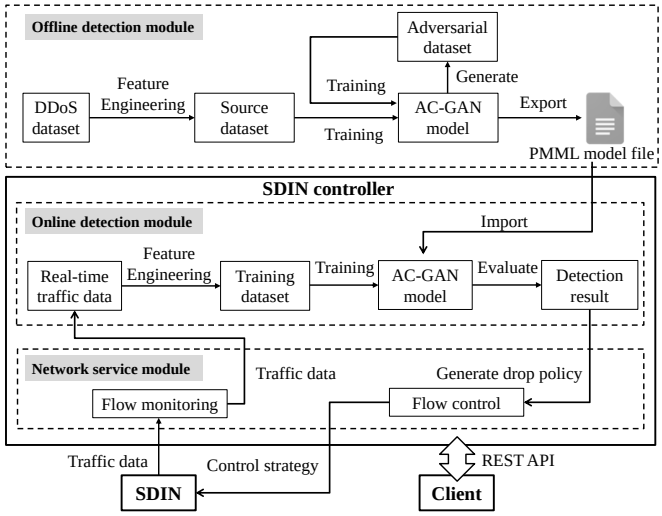


Fig. 2. Processing flow of DDoS attack prevention model based on AC-GAN.

of transaction records the state of network in time, so as to facilitate the timely detection of network issues.

Through above three transactions, entities, information flows and network events in SDIN are all recorded in blockchain, which facilitates tracking and auditing of network status. Besides, smart contracts can be designed at blockchain plane to further enhance the security of SDIN system. For example, when a large number of abnormal flows are detected, smart contracts can issue policies to relieve the burden on the controller or alert the controller to failure in time.

2.2 AC-GAN-based Anti-DDoS Model

Fig. 2 illustrates the process flow of AC-GAN-based anti-DDoS model in the blockchain-empowered SDIN system, including data pre-processing, feature engineering, traffic acquisition, DDoS detection and attack mitigation.

Data preprocessing is responsible for preliminary processing of the original DDoS dataset, including data cleaning, interpolation and other operations.

Feature engineering is mainly responsible for feature construction and selection. Traffic characteristics used in the proposed model include attribute (categorical) features [4] and statistical features. As for attribute features, we use the *Random Forest* (RF) method to rank the importance of original categorical features, so as to select the top effective characteristics. Each analysis interval T needs to group and count flow attributes. The statistical features constructed by the proposed model are as follows.

– **Avg_Packets** means the average packets number of flow in T.

- **Avg_Bytes** means avetage packets bytes of flow in T.
- **E(srcIP)** represents the entropy of the source IP address during T, which can be calculated in Eq. 1. $srcIP_i$ is the number of occurrences of the source IP address within the given time interval T. Hence, N is the total number of occurrences of all source IP addresses, defined in Eq. 2. In the same way, we can obtain the **E(dstIP)**, **E(srcPort)**, and **E(dstPort)**.

$$E(srcIP) = -\sum_{i=1}^{n}(\frac{srcIP_i}{N})log_2(\frac{srcIP_i}{N}). \tag{1}$$

$$N = -\sum_{i=1}^{n} srcIP_i. \tag{2}$$

- **PGR** represents the port growth rate, which can be defined as $PGR = \Delta Port/T$, where the $\Delta Port$ is the change number of ports during T.
- **IPGR** represents IP address growth rate, which is defined as $IPGR = \Delta IP/T$, where the ΔIP is the change number of IP addresses during T.

Model training refers to building the deep learning classifier to detect adversarial DDoS attacks. AC-GAN consits a generator and a discriminator. The input of generator is composed of the one-hot encoding tensor of the training data classification information **c** and the tensor of the noise data **z**. Eq. 3 represents the false data output by the generator.

$$X_{fake} = G(\boldsymbol{c}, \boldsymbol{z}). \tag{3}$$

Eq.s 4 to 6 formalizes the optimization objective of generator loss function, i.e. maximizing the difference between classification loss L_C and authenticity judgment loss L_S.

$$\boldsymbol{Max}\ L_G = \boldsymbol{Max}\ (L_C - L_S). \tag{4}$$

$$L_C = E[logP(C = c|X_{fake})]. \tag{5}$$

$$L_S = E[logP(S = fake|X_{fake})]. \tag{6}$$

As shown in Eq.s 7 to 9, the optimization goal of discriminator is to distinguish the real data from the generated data as much as possible, while being able to effectively classify data. The input of discriminator includes real data X_{real}, fake data X_{fake} (generated by the generator) and data label c. The discriminator outputs P(C|X) and P(S|X), where P(C|X) is the probability distribution of input data with respect to classification labels, and P(S|X) is the probability distribution of source data authenticity judgment (whether the data is real or fake). The last layer of the discriminator is two parallel dense (fully connected) layers to obtain the results of P(C|X) and P(S|X), respectively.

$$\boldsymbol{Max}\ L_D = \boldsymbol{Max}\ (L'_C + L'_S). \tag{7}$$

$$L'_C = E[logP(C = c|X_{real})] + E[logP(C = c|X_{fake})]. \tag{8}$$

$$L'_S = E[logP(S = real|X_{real})] + E[logP(S = fake|X_{fake})]. \quad (9)$$

L'_C is loss function of data classification accuracy, which is defined as logarithmic probability of correct classification. L'_S is loss function for judging the authenticity of data, which is defined as log-likelihood of the correct source data.

Traffic acquisition is mainly responsible for obtaining the key features required for traffic classification through the SDIN controller.

DDoS attack detection is completed by calling the trained machine learning model. These obtained traffic characteristics are used as the input of the classification prediction module, and the prediction result of whether the traffic is a DDoS attack flow is output.

DDoS attack mitigation means that the controller takes corresponding measures according to the detection results. When a traffic is detected as normal, it will be transmitted safely according to the original forwarding rules. Once the detection module detects the attack traffic, the proposed model automatically activates the DDoS attack mitigation module.

3 AC-GAN Training and Classification Algorithm

The *AC-GAN Training and Classification* (ATC) algorithm conducts model training for offline DDoS attack detection. Main phases of Alg. 1 are described as follows:

1. The generator model and discriminator model of AC-GAN are defined in Steps 1 to 11. The main function of generator is using noise data and label information to generate adversarial samples. The discriminator is mainly responsible for judging the authenticity of the data and giving the classification result of the DDoS attack flow.
2. Steps 12 to 23 define loss functions of the generator and discriminator.
3. Steps 24 to 32 complete data processing and feature selection process. Read in offline DDoS dataset and perform feature processing.
4. The training process of the AC-GAN model, which calls the G, D, G_Loss and D_Loss functions defined above, is given in Steps 33 to 44. In step 45, we evaluate the performance of proposed model on testing dataset.
5. Save training parameters to PMML file to improve the detection efficiency when calling the model across platforms.

$FLOP$ is usually used to measure the time complexity of a deep learning model, which determines the time for model training and prediction. Dense layers and convolutional layers are used in our model. The $FLOP$ of dense layers is $\sum_{i=1}^{D}(2 \times In_i - 1) \times Out_i$, where D represents the number of dense layers, In represents the input dimension of the hidden layer, and Out represents the output dimension of the hidden layer. The $FLOP$ of convolutional layers is $\sum_{j=1}^{L} M_j^2 \times K_j^2 \times C_{j-1} \times C_j$, where L represents the number of convolutional layers, M represents the side length of feature map, K represents the side length of kernel, and C_j represents the number of convolution kernels in the j_{th} layer. Total time complexity of training model is approximately the sum of the above.

Algorithm 1 AC-GAN-enabled Training and Classification Algorithm

Require: Generator model, Discriminator model, DDoS training dataset, Data classification label

Ensure: Trained model in PMML format

/*Define the generator model of AC-GAN*/
function G(Classified traffic $\boldsymbol{c}$, Noise information $\boldsymbol{z}$)
    Define network structure of generator model G
    Use $\boldsymbol{c}$ and $\boldsymbol{z}$ to generate adversarial samples X_{fake}
    return X_{fake}
end function
/*Define the discriminator model of AC-GAN*/
function D(X_{train}, $\boldsymbol{c}$)
    Define network structure of discriminator model D
    Determine whether the data is real or fake
    Determine the classification of traffic data
    return isReal_out, DDoS_class_out
end function
/*Define the loss function of generator*/
function *G_Loss*(fake_out, fake_class_out, $\boldsymbol{c}$)
    Calculate fake losss L_S by Eq. 6
    Calculate classification loss L_C by Eq. 5
    $L_G \leftarrow L_C$ - L_S
    return L_G
end function
/*Define the loss function of discriminator*/
function *D_Loss*(real_out, real_class_out, fake_out, $\boldsymbol{c}$)
    Calculate authenticity loss L'_S by Eq. 9
    Calculate classification loss L'_C by Eq. 8
    $L_D \leftarrow L'_C + L'_S$
    return L_D
end function
/*Data reading and processing*/
Read the DDoS dataset with classification label $\boldsymbol{c}$
Data preprocessing
for $\forall$ characteristic columns in the dataset **do**
    for $\forall$ characteristic X **do**
        X $\leftarrow (X - X_{min}) \div (X_{max} - X_{min})$
    end for
end for
Feature engineering for feature selection and extraction
Divide the dataset to X_{train}, Y_{train}, X_{test}, Y_{test}
/*AC-GAN model training process*/
Generate noise $\boldsymbol{z}$ based on the distribution of X_{train}
while i $\leq$ preset iterations & stop condition not met **do**
    for each epoch **do**
        X_{fake} = G(Y_{train},$\boldsymbol{z}$)
        fake_out, fake_class_out = D(X_{fake}, Y_{train})
        real_out, real_class_out = D(X_{real}, Y_{train})
        g_loss = G_Loss(fake_out, fake_class_out, Y_{train})
        d_loss = D_Loss(real_out, real_class_out, fake_out, Y_{train})
        Calculate the gradient of g_loss and d_loss
    end for
    i $\leftarrow$ i + 1
end while
Validate model performance on X_{test}, Y_{test}
Export AC-GAN model to PMML file **return** PMML

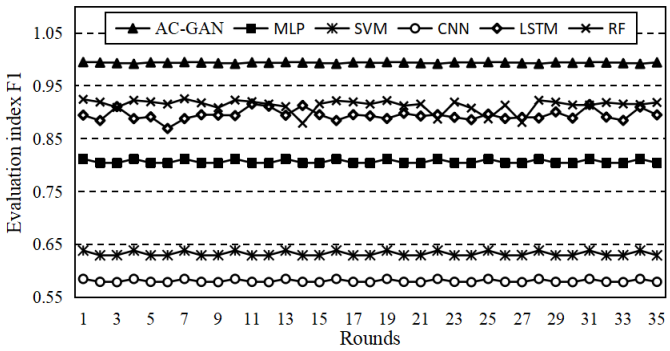


Fig. 3. F1 results of algorithms under multiple rounds of experiments.

4 Experiments and the Results

4.1 Experiment Environment and Configuration

When simulating the setting up of blockchain-based SDIN scenario, we use Hyperledger Fabric 1.4 blockchain platform and Mininet 2.3 emulator to customize network topology with Python 3.7 on the VirtualBox virtual machine with the Ubuntu 18.04 operating system. In order to improve development efficiency, the virtual network is connected to the Floodlight 1.2 controller.

The original dataset used for offline model training are TCP SYN flood flow, UDP flood flow and $TCMP$ flood flow in $CICDDoS2019$, as well as *Benign* normal flow in $CICDS2017$. The adversarial data samples generated by the AN-GAN network are also used for training.

4.2 Experiment Results

We compare the performance of *Multi-layer Perceptron* (MLP) [10], *Convolutional Neural Networks* (CNN) [3], *Long-Short Term Memory* (LSTM) [3], *Support Vector Machines* (SVM), basic *Random Forest* (RF) with the designed AC-GAN algorithm on the adversarial DDoS attack dataset from four perspectives, namely *Precision*, *Recall*, *Accuracy* and *F1*.

We repeat 35 rounds of model trainging as well as verification, and the $F1$ results of AC-GAN and other algorithms are shown in Fig. 3. It can be concluded that the performance of the AC-GAN algorithm is stable and performs well on $F1$. We give the performance of above algorithms in specific percentage in Table. 1. It can be found that AC-GAN has achieved better performance in *Precision*, *Accuracy* and $F1$. The *Recall* rate of CNN can reach 100%, and its *Accuracy* performance is poor on adversarial attack dataset. In addition, the performance of MLP and SVM in DDoS detection is not satisfactory. LSTM and RF methods show high *Recall*, and both have poor performance in *Accuracy*.

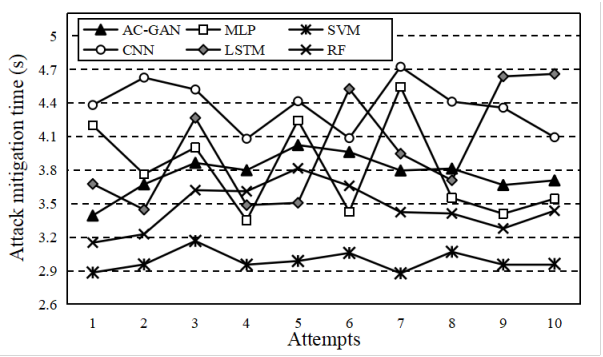


Fig. 4. Comparison of mitigation time under TCP flood attack.

Table 1. Comparison of different models on four detection indicators

Algorithm	Precision	Recall	Accuracy	F1 Score
AC-GAN	**99.658**%	99.318%	**99.402**%	**99.488**%
MLP	89.549%	79.557%	82.610%	84.258%
CNN	58.4952%	**100.000**%	58.495%	73.813%
LSTM	86.464%	99.574%	90.633%	92.557%
SVM	77.588%	54.258%	64.075%	63.859%
RF	86.642%	**100.000**%	90.981%	92.843%

After launching the attack, the SDIN controller detects the attack and issues mitigation strategies to the flow table in time. Fig. 4 compares the mitigation time of adopting different algorithms when launching 10 TCP Flood attacks. Although the mitigation time of SVM is slightly faster than other methods, its detection accuracy is flawed. Moreover, there is little difference in mitigation time between AC-GAN and other algorithms.

In summary, experiment evaluations depict that the AC-GAN approach can effectively detect and mitigate DDoS attacks in SDIN environment. The performance of the scheme have been evidenced and adoptable results are received.

5 Conclusions

Aiming at security issues in SDIN system, this paper proposes a blockchain-empowered SDIN scheme and a deep learning-based attack detection model to defend adversarial DDoS attacks. The blockchain records all entities, information flows and network events in SDIN, monitoring the state of devices and networks to improve security. We also develop an AC-GAN method to generate adversarial attack samples, which improves the detection accuracy of DDoS attacks by increasing the model sensitivity. The performance of our proposed scheme had been evaluated and demonstrated to be an adoptable approach in SDIN.

Acknowledgements

This work is partially supported by the National Key Research and Development Program of China (Grant No. 2021YFB2701300), Shandong Provincial Key Research and Development Program (Grant No. 2021CXGC010106).

References

1. Chattaraj, D., et al.: On the design of blockchain-based access control scheme for software defined networks. In: 39th IEEE Conference on Computer Communications, INFOCOM Workshops. pp. 237–242. IEEE (2020)
2. Chen, J., et al.: SDATP: An SDN-based traffic-adaptive and service-oriented transmission protocol. IEEE TCCN **6**(2), 756–770 (2019)
3. Gadze, J.D., Bamfo-Asante, A.A., Agyemang, J.O., Nunoo-Mensah, H., Opare, K.A.: An investigation into the application of deep learning in the detection and mitigation of DDoS attack on SDN controllers. Technologies **9**(1), 14 (2021)
4. Gai, K., Qiu, M., Thuraisingham, B.M., Tao, L.: Proactive attribute-based secure data schema for mobile cloud in financial industry. In: 17th IEEE HPCC. pp. 1332–1337. IEEE, New York, USA (2015)
5. Gai, K., Wu, Y., et al.: Differential privacy-based blockchain for industrial Internet-of-Things. IEEE Trans. Ind. Informatics **16**(6), 4156–4165 (2020)
6. Gai, K., Zhang, Y., Qiu, M., Thuraisingham, B.: Blockchain-enabled service optimizations in supply chain digital twin. IEEE Transactions on Services Computing **PP**(99), 1–12 (2022)
7. Gao, Y., Chen, Y., Lin, H., Rodrigues, J.J.P.C.: Blockchain based secure IoT data sharing framework for SDN-enabled smart communities. In: 39th IEEE Conference on Computer Communications, INFOCOM Workshops. pp. 514–519. IEEE (2020)
8. Guo, Y., et al.: A privacy-preserving auditable approach using threshold tag-based encryption in consortium blockchain. In: International Conference on Smart Computing and Communication. pp. 265–275. Springer (2021)
9. Liu, K., Xiao, K., Dai, P., Lee, V., Guo, S., Cao, J.: Fog computing empowered data dissemination in software defined heterogeneous VANETs. IEEE Transactions on Mobile Computing **PP**(99), 1 (2020)
10. Nugraha, B., et al.: Deep learning-based slow DDoS attack detection in sdn-based networks. In: 2020 IEEE NFV-SDN. pp. 51–56. IEEE, Madrid, Spain (2020)
11. Tayfour, O., Marsono, M.: Collaborative detection and mitigation of distributed Denial-of-Service attacks on Software-Defined-Network. Mobile Networks and Applications **25**, 1338–1347 (2020)
12. Wang, R., et al.: An entropy-based distributed DDoS detection mechanism in Software-Defined-Networking. In: 2015 IEEE Trustcom/BigDataSE/ISPA. vol. 1, pp. 310–317. IEEE (2015)
13. Weng, J., Weng, J., Liu, J., Zhang, Y.: Secure Software-Defined Networking based on blockchain (2019), `http://arxiv.org/abs/1906.04342`
14. Xu, Y., Liu, Y.: DDoS attack detection under SDN context. In: IEEE INFOCOM 2016-the 35th annual IEEE ICCC. pp. 1–9. IEEE, San Francisco, CA, USA (2016)
15. Zhang, Y., Gai, K., Xiao, J., Zhu, L., Choo, K.R.: Blockchain-empowered efficient data sharing in Internet of Things settings. IEEE Journal on Selected Areas in Communications **PP**(99), 1–1 (2022)

Blockchain-based Fairness-Enhanced Federated Learning Scheme Against Data Poisoning Attack

Shan Jin[1][0000-0002-0689-6544], Yong Li[1][0000-0002-1419-6257], Xi Chen[2][0000-0001-8553-827X], Ruxian Li[2] and Zhibin Shen[2]

[1] School of Electronic and Information Engineering,
Beijing Jiaotong University, Beijing 100081, China.

[2] Linklogis, Shenzhen 518063, China.

liyong@bjtu.edu.cn

Abstract. The federated learning technology provides a new method for data integration, which realizes sharing of a global model and prevent the leakage of user's original data information. In order to resist data poisoning attack from some participants, ensure reliability and accuracy of the global model, and ensure fairness of the aggregation process in federated learning, we propose a blockchain−based fairness enhanced federated learning scheme. The *accuracy* of global model and *fairness* of the aggregation process is guaranteed by an adaptive aggregation algorithm which can defense data poisoning attack. The *reliability* of federated learning process is ensured by recording the entire process of the model training on the blockchain and using digital signatures. The *privacy* of each participant of federated learning is protected by public key encryption combined with the use of random numbers. Theoretical analysis and experiments show that the scheme can protect privacy of each participant, mitigate data poisoning attack and ensure the reliability and fairness of the entire federated learning process.

Keywords: Blockchain, Data Poisoning Attack, Federated Learning, Privacy Protection.

1 Introduction

Nowadays, the existence of data silos has seriously affected the integration of data. The existence of data silos and the strengthening of data privacy regulation presents challenges to artificial intelligence. In this conflict between data availability and privacy protection [1-3], "*privacy-preserving computing*" technology can promote the process of data consolidation. Federated learning (FL) [4][5], one privacy-preserving computing technology, supports secure multi-party machine learning: local data is maintained on the participant's device and the updates to the model are aggregated through a secure aggregation algorithm. Federated learning generally includes a central server and various participants. Participants upload their local models or gradients to the central server, afterwards the server aggregates the global model and then sends it to each participant to continue the next iteration.

However, traditional federated learning assumes a trusted central server for coordination. Due to the centralized form of federated learning, it faces some problems such as the single point of failure problem, the lack of verifiability of the aggregation results, etc. In addition, some malicious participants can harm the performance of the global model by carrying out a poisoning attack [6][7]. Considering above problems, some studies [9-12] have tried to combine blockchain [8] with federated learning, and complete federated learning tasks by storing model updates on the blockchain.

Most schemes which considered poisoning attacks only discuss the scheme's performance in the presence of poisoning attacks. However, there is another situation in practice, that is, all participants are honest, and no one carries out poisoning attacks. In this case, most schemes may misjudge an honest participant as a poisoner, making the scheme unfair. It's necessary to design a scheme that can ensure the fairness of federated learning as well as resist poisoning attack.

In this paper, we propose a blockchain-based federated learning framework and an A-BFL (*Adaptive Blockchain based Federated Learning*) algorithm which can resist data poisoning attack and ensure the fairness of the aggregation process. In the end, we perform experiments on the MNIST [13] and Fashion-MNIST [14] datasets to test performance of our scheme. The contributions of our work are as follows:

- A blockchain-based federated learning framework is proposed, which combines the traditional federated learning with blockchain to ensure the training process and results are traceable and cannot be tampered with.
- We design an A-BFL aggregation algorithm to resist data poisoning attack, which can identify most of the poisoners, reduce the probability of misjudgment and ensure the fairness of aggregation.
- We use the deposit mechanism in the blockchain to ensure the task requester's reliability. If the task requester performs malicious actions, such as uploading a wrong accuracy on the test dataset, etc., its deposit will be deducted as a penalty.
- Public key encryption and digital signature are used to ensure that no one except the task requester can obtain the local models and to prevent local models from being stolen.

The rest of this paper is organized as follows. In section 2, we review related works. Then, we discuss the design of the scheme in section 3 and test the performance of our scheme through experiments on the actual datasets in section 4. Finally, we summarize the paper in section 5.

2 Related Works

In the process of FL, some participants may carry out data poisoning attack to drop the accuracy of the global model. There is no proper method for resisting poisoning attack in schemes [9][10][15] which combined blockchain with FL. Schemes [16][17] focused on the quality of the global model, which can prevent poisoning attacks to a certain extent.

Algorithms used in traditional machine learning such as Krum/Multi-Krum [18], Zeno [19], etc. can also be used to resist data poisoning attack in FL. For example, the first P2P machine learning system [20] proposed by Shayan et al. filtered malicious model updates through multi-Krum algorithm. These traditional algorithms require to input the number of poisoners in advance, however it's difficult to do this in reality. Unlike the traditional algorithms, schemes [21] [22] filtered out poisoned models by setting the threshold. However, in the absence of poisoning attacks, some normal models will also be filter out, making these schemes unfair.

In addition to filter out models, schemes [23] [24] adjusted the weights of models during the aggregation process according to the quality of models. If a local model is of low quality, it will be weighted less when aggregating. However, homomorphic encryption and the calculation under ciphertext make them non-universal.

In addition, some algorithms used in traditional machine learning such as Median [25] (take the median value of the local models or gradients submitted by the participants) are also suitable for FL.

3 Blockchain Based Federated Learning

Fig.1 shows the pipeline of the proposed framework. The framework includes task requester, task participants, blockchain, and cloud platform.

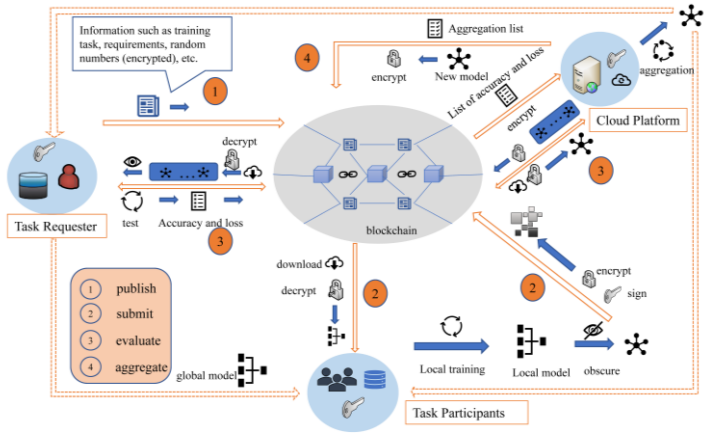


Fig. 1. An overview of the proposed blockchain based federated learning framework.

- Task Requester (TR for short): The requester of the federated learning task, responsible for publishing the task to the blockchain and collaborating with the cloud platform to complete the aggregation of global model. TR can obtain all the local models submitted by task participants. To reward task participants and cloud platform, TR should pay for them.
- Cloud Platform (CP): It is responsible for calculating the distance between models submitted by participants and cooperating with TR to complete model aggregation.
- Task Participants (P): They are responsible for training local models, who can be from different institutions in the same industry with similar datasets. Task participants should own eligible local datasets. (Assuming that the local datasets of all participants are independent and identically distributed).
- Blockchain: It records the process of model submission and aggregation.

For ease of reference, the symbols that appeared in this paper and corresponding descriptions are listed in Table.1.

Table 1. Symbols and their descriptions

Symbol	Description	Symbol	Description
SGD	stochastic gradient descent	m	dimension of the model
N	number of participants	$\boldsymbol{r}$	m random numbers
T	max iteration of FL	$\boldsymbol{G}$	gradient
L	local training epochs	$\boldsymbol{R}$	obscured model
η	learning rate	sk	private key
$\boldsymbol{x}$	training data vector	pk	public key
y	training data labels	pop	percentage of poisoners
B	small batches of training data	iop	intensity of poisoning attack
$\boldsymbol{\omega}$	model	I	number of iterations

3.1 Assumption

TR has a very small dataset and limited computing power, and wants to obtain a better model. Some participants may be malicious, they will carry out data poisoning attack to drop the accuracy of the global model; in addition, they may steal other participants' models from blockchain, and then use other people's models to obtain rewards. CP is semi-honest, and it may infer some relevant model information while completing computing task honestly. There is no collusion among TR, task participants, and CP, as well as among the task participants. Other nodes outside the federated learning process on the blockchain will not carry out attacks that will affect the federated learning process.

3.2 Blockchain-based federated learning framework

In this paper, we mainly focus on representative poisoning attacks: *label-flipping attack*. For example, malicious participants incorrectly mark the samples with the true label to the wrong label for misclassification. Our main ideas are as follows:

The federated learning process can proceed only if the data is successfully uploaded to the blockchain. However, uploading the data to blockchain also takes some time. In order to achieve high accuracy with fewer iterations and reduce waiting time for participants, task participants need to complete several training epochs locally before uploading local models to the blockchain. Inspired by the previous work [23], we combine asymmetric encryption with random numbers to ensure that CP and anyone in blockchain who do not participant in the federated learning task cannot obtain the original model data.

To resist data poisoning attack, we proposed an A-BFL aggregation algorithm that can not only identify the poisoners as much as possible, but also guarantee fairness when there are no poisoners. The aggregation algorithm mainly considers the following factors: the distance (use the same method as in [18]) between the local models, and the performance (accuracy and loss) of the local models on the test dataset. Each step is described in detail as follows, and the A-BFL algorithm is shown in Fig.2.

3.3 Details of our scheme

1. *Publishing the task*

TR determines the training task and training parameters, uploads the following content to the blockchain.

(1) Training task, the target accuracy of global model or the maximum training iteration T; the participant's local training epochs L, the waiting time for the submission of the local model for each iteration, etc.

(2) The hash value of the test dataset D_{test} (each participant should get the original test dataset after the overall training process to verify that the TR has uploaded the correct results when testing the accuracy and loss of the local models).

(3) Initial global model $[\![\boldsymbol{\omega}^0]\!]_{pk_n}(n \in [1, N])$ (encrypted with each participant's public key). The initial m random numbers $\boldsymbol{r}^0 = \{r_i, i \in [1, m]\}$ are selected according to the dimension m of the initial global model, encrypted with the public keys of each participant.

In addition, TR needs to pay a deposit to the blockchain, and the deposit will be redeemed only when more than half of the participants agree with the training results at the end of the training.

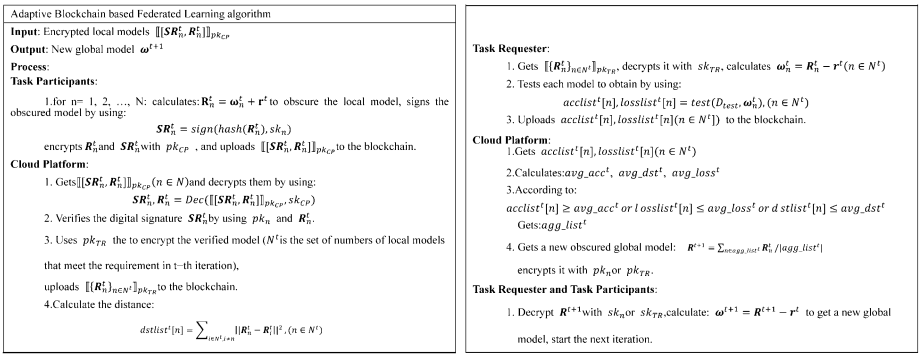


Fig. 2. A-BFL algorithm

2. *Submitting the model*

Take the first iteration as an example:

Participants randomly select the local data of small batch B and calculate the loss (cross entropy loss function): $L_f(B, \boldsymbol{\omega}) = \frac{1}{|B|}\sum_{\langle \mathbf{x}_i, y_i\rangle \in B} L_f(\boldsymbol{\omega}, \mathbf{x}_i, y_i)$. Then each participant calculates the gradient of the loss function: $\boldsymbol{G} = \nabla_{\boldsymbol{\omega}} L_f(B, \boldsymbol{\omega})$. Finally, the new local model is calculated: $\boldsymbol{\omega}^l = \boldsymbol{\omega}^{l-1} - \eta \cdot \mathbf{G}$.

Each participant repeats the above training process until the number of training epochs reaches L. The local model at this time is recorded as $\boldsymbol{\omega}_n^0 = \boldsymbol{\omega}^L$. Then each participant obscures the local model with m random numbers: $\mathbf{R}_n^0 = \boldsymbol{\omega}_n^0 + \mathbf{r}^0$, hashes the obscured model and sign it with its private key: $\boldsymbol{SR}_n^0 = Sign(hash(\boldsymbol{R}_n^0), sk_n)$. Then participants encrypt their respective models by using the public key of CP: $[\![\boldsymbol{SR}_n^0, \boldsymbol{R}_n^0]\!]_{pk_{CP}} = Enc([\boldsymbol{SR}_n^0, \boldsymbol{R}_n^0], pk_{CP})$, upload $[\![\boldsymbol{SR}_n^0, \boldsymbol{R}_n^0]\!]_{pk_{CP}}$ to the blockchain.

3. *Model evaluation and aggregation*

Taking the t-th iteration as an example, within the specified waiting time, CP gets $\boldsymbol{SR}_n^t, \boldsymbol{R}_n^t = Dec([\![\boldsymbol{SR}_n^t, \boldsymbol{R}_n^t]\!]_{pk_{CP}}, sk_{CP})$.

TR, task participants and CP cooperate with each other to evaluate and aggregate the local models by executing the A-BFL algorithm shown in Fig.2. In the A-BFL algorithm , CP can calculate: $avg\_acc^t = (\max(acclist^t[n]) + \min(acclist^t[n]))/2$, $avg\_loss^t = (\max(losslist^t[n]) + \min(losslist^t[n]))/2$ and $avg\_dst^t = (\sum_{n\in N^t} dstlist^t[n])/|N^t|$. Then CP uploads the public key list agg_list^t (participants whose local model meet (1)) and the public key list of all participants all_list^t (all participants who submit model in the t-th iteration) to the blockchain. The corresponding task participants in the agg_list^t can get paid from blockchain.

$$acclist^t[n] \geq avg\_acc^t or\ losslist^t[n] \leq avg\_loss^t or\ dstlist^t[n] \leq avg\_dst^t \quad (1)$$

Next, CP can aggregate the models that meet the requirement (1) to obtain a new obscured global model: $\mathbf{R}^{t+1}$. TR and task participants can get the new global model $\boldsymbol{\omega}^{t+1}$ by using the random numbers $\boldsymbol{r}^t$. After this, TR tests the accuracy and loss of the global model, uploads the accuracy and loss to the blockchain, and continues to generate m random numbers $\boldsymbol{r}^{t+1}$ for the next iteration, encrypted $\boldsymbol{r}^{t+1}$ with the public key of the participant in all_list^t, uploads the encrypted $\boldsymbol{r}^{t+1}$ to the blockchain. Then task participants continue to train the local models and repeat above steps until

the maximum iteration is reached or the accuracy of the global model meets the requirement. Besides, CP gets paid from blockchain.

4. *End of the federated learning task*

(1) When the new global model meets the training requirement or the iteration reaches T, TR announces the end of the training, and at the same time uploads the relevant information of the test dataset to the blockchain (for example, if it is stored in the cloud, the link of the test dataset needs to be uploaded to the blockchain). After the task participants obtain the dataset, they can verify whether the hash value is the same as the hash value uploaded by the TR in the beginning.

(2) Each participant can use the historical submitted models on the blockchain to verify the accuracy and loss on the test dataset, and compares them with the results uploaded by TR on the blockchain. If more than half of the participants believe that there is no problem with the aggregation process, then the deposit will be redeemed to TR; otherwise, as a penalty, TR will not be able to get the deposit. At this point, the training process is finished, the TR and eligible task participants have obtained the global model.

4 Experiments Analysis and Comparison

The experiments were run on a PC equipped with a 4-core I5-6300H processor at 2.30 GHz and 12 GB RAM. We implemented label flipping attack and defense algorithms in Python, using the TensorFlow and sklearn packages. In our experiments, participants train the local model using the Multilayer Perceptron (MLP) network in TensorFlow. We compare the performance of schemes [21][25] with our A-BFL, the evaluation criteria include the accuracy in test dataset and the misjudgment rate. Regarding the misjudgment rate, the description is as follows: It mainly consider two aspects, one is the probability that the local models submitted by poisoners are regarded as the normal models: WTC (Wrong to Correct), on the other hand is the probability of excluding global models submitted by honest participants when the global model is aggregated: CTW (Correct to Wrong).

Suppose the set of all participants is expressed as $\boldsymbol{P} = \{\boldsymbol{P}_h, \boldsymbol{P}_p\}$, the total amount $N = |\boldsymbol{P}|$, the number of honest participants is $N_h = |\boldsymbol{P}_h|$, and the number of poisoners is $N_p = |\boldsymbol{P}_p|$. The set of the participants selected to participate in the aggregation in aggregation process is donated as $\boldsymbol{P}_{chosen}$.Then, $WTC = (\boldsymbol{P}_{chosen} \cap \boldsymbol{P}_p)/N_p$, $CTW = ((\mathbf{P} - \mathbf{P}_{chosen}) \cap \mathbf{P}_h)/N_h$ .Total misjudgment rate can be calculated: $err\_rate = WTC + CTW$.

The federated average algorithm is denoted as: “Fed-Avg”, the aggregation algorithm proposed in [21] is denoted as “SFL”, and the method of taking the median [25] value of the model (or gradient) is denoted as “Median”. We use the percentage of the modified labels to indicate the intensity of data poisoning attack. The symbols used in the following description are described in section 3.

1) Performance under data poisoning attack

It is assumed that $N = 20, L = 5, pop = 0.3, iop = 1$. “Fed-Avg” is set as the baseline (the dotted line in the figure). As shown in Fig.3.1, when some participants carry out data poisoning attack, compared with the baseline, the accuracy of Fed-Avg dropped by nearly 10% in the first few iterations. The Median algorithm requires

more parameter interaction rounds to achieve better accuracy. SFL and A-BFL algorithm can achieve an accuracy comparable to the baseline. The total misjudgment rate of SFL and A-BFL is almost 0, so that A-BFL can identify most of the poisoners and exclude them to get a better global model (Because the total misjudgment rate is always 0 in most time when there exist poisoners, we do not show the experiment results here).

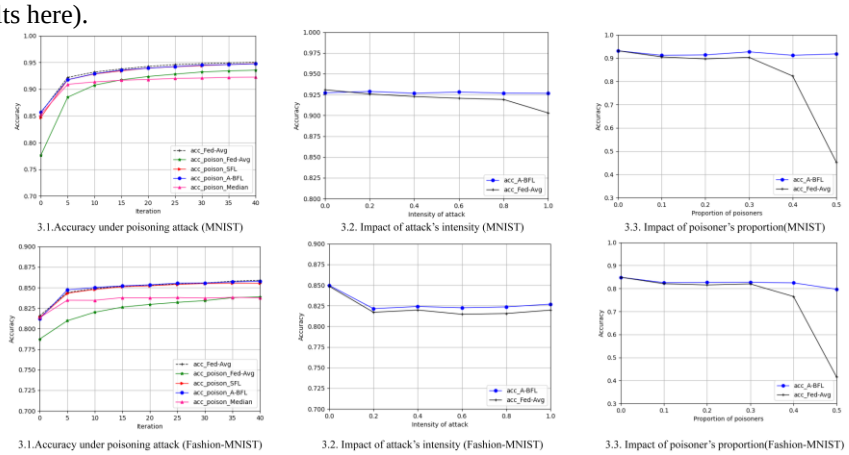


Fig. 3. Experiments under poisoning attack

When setting $N = 20, pop = 0.3, I = 10$, the impact of the poisoning attack intensity (iop) on the accuracy of the global model is shown in Fig. 3.2. Compared with the baseline, the A-BFL algorithm can always maintain a high accuracy, which ensures that TR can obtain the desired training results.

When setting $N = 20, iop = 1, I = 10$, the impact of the proportion of the poisoners (pop) on the accuracy of the global model is shown in Fig.3.3. When the proportion of poisoners is less than 50%, with the increase of the proportion of poisoners, the accuracy of Fed-Avg algorithm drops sharply. In contrast, A-BFL algorithm can still maintain a high accuracy. In the case of more than half of the participants become the poisoners, the accuracy of A-BFL algorithm will be seriously reduced. Because in the model evaluation stage, the local models submitted by honest participants are very different from the poisoning local models, which will make the honest participants have a high probability of being judged as poisoners, resulting in a decrease in the accuracy (It's not shown here).

2) Performance without data poisoning attack

In addition to considering how to identify and exclude the poisoned models, we also consider how to prevent local models submitted by honest participants from being regarded as the poisoned models when all the participants are honest. Fig.4.1 shows that without the poisoners, the performance of the proposed A-BFL algorithm is close to the baseline. In addition, we consider how to ensure the fairness of the algorithm when there is no poisoning attack. Because the honest participant may be identified as poisoners, which leads to normal local models cannot participate in the aggregation, we take the probability of misjudgment into consideration. Fig.4.2 shows that compared to SFL algorithm, A-BFL algorithm can greatly reduce the probability of misjudgment. Besides, from Fig.4.2 we can see that the misjudge rate of SFL varies

greatly, that is because SFL algorithm uses the accuracy of the model as the only criterion for identifying the poisoned models.

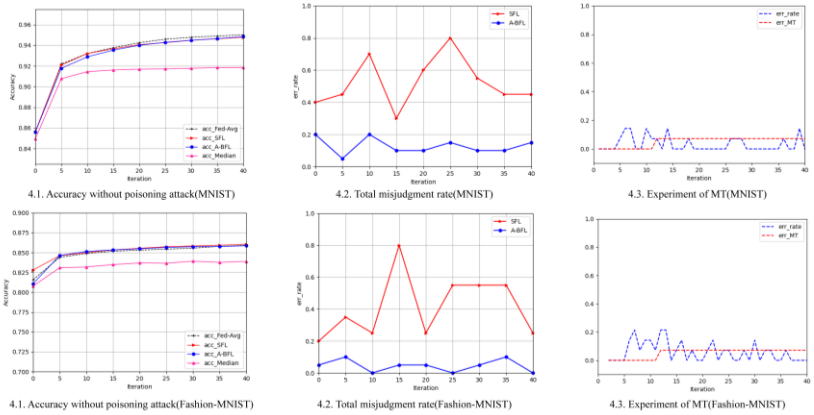


Fig. 4. Experiments of misjudgment rate and performance without poisoning attack

In addition, TR can choose to set a tolerable maximum number of non-participation aggregation times MT, and participants who exceed this number of times will not be eligible to continue participating in the subsequent training process. CP makes judgment based on the aggregation history recorded on the blockchain. If the local model submitted by a participant does not meet the requirement in consecutive MT iterations of training, then the public key of that participant will be uploaded to the blockchain. And after this iteration, the local model uploaded by that participant will no longer be aggregated, and TR also will no longer send m random numbers for obscuring to that participant.

3) Experiments of MT

Under the setting of MT, if one participant whose local model does not meet the requirements in consecutive MT iterations, then that participant will be excluded from the federated learning task. If one participant whose local model does not meet the requirements in a single iteration, it will not be excluded.

In addition to the total misjudgment rate mentioned above, we use a parameter $err\_MT = (\boldsymbol{P}_{ex} \cap \boldsymbol{P}_h)/N_h$ to represent the probability of honest participants being excluded from the federated learning task ($\boldsymbol{P}_{ex}$ is the set of participants who have excluded from the federated learning task).

It is assumed that $N = 20, L = 5, pop = 0.3, iop = 1$ and MT is 4. As shown in Fig.4.3, the poisoners are excluded in the first MT iterations. After that, even if some honest participants are regarded as the poisoner in some iteration, they will not be excluded from the federated learning task, unless they are regarded as the poisoner in a consecutive MT iteration. The setting of MT can make our scheme fairer, prevent misjudge the honest participants as the poisoners as much as possible.

4) Time overhead

Some cryptographic algorithms such as hash function are used in our scheme. We perform experiments to illustrate the time spent by our scheme. RSA cryptosystem is used in the experiment to encrypt, decrypt, sign, and verify. The length of the private key is set to 1024. We do ten times of experiment and take the average as the results.

The final results are shown in Table.2. The decryption of the model takes the longest time. Other processes take less time compared to the process of training one local model which is necessary in traditional federated learning.

Table 2. Time overhead

Process	hash	sign	verify	encrypt	decrypt	train	test
Time	8.07ms	1.71ms	8.51ms	2.28s	48.11s	17.32s	0.47s

Remark on decryption time overhead. TR only need to perform one decryption operation in an iteration to obtain all local models. However, when the number of task participants is too large, TR will spend too much time in decrypting the model set. This problem may be solved in the future by combining public key encryption and symmetric encryption.

Remark on security analysis. Due to page limitation, security analysis of robustness, privacy, fairness is omitted and will be provided in the full version of the paper.

5 Conclusion

In this paper, we proposed a blockchain-based federated learning scheme, focusing on the defense against data poisoning attack and the privacy protection of each participant. In practice, TR can be specific agency, such as government, who need a lot of data to train the model but don't have sufficient data. And the proposed scheme can be applied in multiple scenarios such as Internet of Things (IoT) and Mobile Edge Computing (MEC) [26][27].

Through experiments on public datasets, it can be seen that the proposed scheme can effectively resist data poisoning attack, and in the absence of data poisoning attack, the scheme can also ensure a low probability of misjudgment. In addition, our scheme can protect the privacy of each participant, guarantee the fairness of federated learning and reliability of the TR, ensure high accuracy of the global model.

References

1. Qiu M, Gai K, Xiong Z.: Privacy-preserving wireless communications using bipartite matching in social big data. Future Generation Computer Systems, 87: 772-781 (2018).
2. Gai K, Qiu M.: Blend arithmetic operations on tensor-based fully homomorphic encryption over real numbers. IEEE Transactions on Industrial Informatics, 14(8): 3590-3598 (2017).
3. Qiu H, Qiu M, Lu Z.: Selective encryption on ECG data in body sensor network based on supervised machine learning. Information Fusion, 55: 59-67 (2020).
4. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas.: Communication-efficient learning of deep networks from decentralized data. In: AISTATS, pp. 1273–1282 (2017).
5. T. Li, A. K. Sahu, A. Talwalkar and V. Smith.: Federated Learning: Challenges, Methods, and Future Directions. IEEE Signal Processing Magazine, vol. 37, no. 3, pp. 50-60 (2020).
6. M. Goldblum et al.: Dataset Security for Machine Learning: Data Poisoning, Backdoor Attacks, and Defenses. IEEE Transactions on Pattern Analysis and Machine Intelligence (2022).
7. Biggio B, Nelson B, Laskov P.: Poisoning attacks against support vector machines. arXiv preprint. arXiv:1206.6389 (2012).
8. Nakamoto, Satoshi.: Bitcoin: A peer-to-peer electronic cash system. Decentralized Business Review: 21260 (2008).

9. Z. Zhang, T. Yang, and Y. Liu. : SABlockFL: A blockchain-based smart agent system architecture and its application in federated learning. Int. J. Crowd Sci., vol. 4, no. 2, pp. 133–147 (2020).
10. P. C. M. Arachchige, P. Bertok, I. Khalil, D. Liu, S. Camtepe and M. Atiquzzaman.: A Trustworthy Privacy Preserving Framework for Machine Learning in Industrial IoT Systems. IEEE Transactions on Industrial Informatics, vol. 16, no. 9, pp. 6092-6102 (2020).
11. Zhilin Wang and Qin Hu. Blockchain-based federated learning: A comprehensive survey. arXiv preprint. arXiv:2110.02182 (2021).
12. Y. Qu et al.: Decentralized Privacy Using Blockchain-Enabled Federated Learning in Fog Computing. IEEE Internet of Things Journal, *vol*. 7, no. 6, pp. 5171-5183 (2020).
13. LeCun Y, Bottou L, Bengio Y, et al.: Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11): 2278-2324 (1998).
14. Xiao, Han, Kashif Rasul, and Roland Vollgraf.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint. arXiv:1708.07747 (2017).
15. Mugunthan V, Rahman R, Kagal L.: BlockFLow: An Accountable and Privacy-Preserving Solution for Federated Learning. arXiv preprint. arXiv:2007.03856 (2020).
16. Y. Lu, X. Huang, Y. Dai, S. Maharjan and Y. Zhang.: Blockchain and Federated Learning for Privacy-Preserved Data Sharing in Industrial IoT. IEEE Transactions on Industrial Informatics, vol. 16, no. 6, pp. 4177-4186 (2020).
17. Z. Li, J. Liu, J. Hao, H. Wang, and M. Xian.: CrowdSFL: A secure crowd computing framework based on blockchain and federated learning. Electronics, vol. 9, no. 773, pp. 1–21 (2020).
18. P. Blanchard et al.: Machine learning with adversaries: Byzantine tolerant gradient descent. In: Proc. Neural Inf. Process. Syst. (NeurIPS), pp. 119–129 (2017).
19. Xie, Cong, Sanmi Koyejo, and Indranil Gupta.: Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance. In: 36th International Conference on Machine Learning. PMLR 97: pp. 6893-6901 (2019).
20. M. Shayan, C. Fung, C. J. M. Yoon and I. Beschastnikh.: Biscotti: A Blockchain System for Private and Secure Federated Learning. IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 7, pp. 1513-1525 (2021).
21. Liu Y, Peng J, Kang J, et al.: A Secure Federated Learning Framework for 5G Networks. IEEE Wireless Communications, vol. 27, no. 4, pp. 24-31 (2020).
22. Yuanhang Qi, M. Shamim Hossain, Jiangtian Nie, Xuandi Li.: Privacy-preserving blockchain-based federated learning for traffic flow prediction. Future Generation Computer Systems 117: 328-337 (2021).
23. X. Liu, H. Li, G. Xu, Z. Chen, X. Huang and R. Lu.: Privacy-Enhanced Federated Learning Against Poisoning Adversaries. IEEE Transactions on Information Forensics and Security, vol. 16, pp. 4574-4588 (2021).
24. Z. Ma, J. Ma, Y. Miao, Y. Li and R. H. Deng.: ShieldFL: Mitigating Model Poisoning Attacks in Privacy-Preserving Federated Learning. IEEE Transactions on Information Forensics and Security, vol. 17, pp. 1639-1654 (2022).
25. D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett.: Byzantine-robust distributed learning: Towards optimal statistical rates. In: 35th International Conference on Machine Learning, PMLR 80: pp. 5650–5659 (2018).
26. Gai K, Wu Y, Zhu L, et al.: Differential privacy-based blockchain for industrial internet-of-things. IEEE Transactions on Industrial Informatics, 16(6): 4156-4165 (2019).
27. Xu Y, Lu Z, Gai K, et al.: BESIFL: Blockchain Empowered Secure and Incentive Federated Learning Paradigm in IoT. IEEE Internet of Things Journal, pp. 2327-4662 (2021).

Cryptography of Blockchain

Ying Long[1,2],Yinyan Gong[1,2*], Weihong Huang[1,2], Jiahong Cai[1,2], Nengxiang Xu[1,2], Kuan-ching Li[1,2]

1 School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China

2 Hunan Key Laboratory for Service computing and Novel Software Technology, Xiangtan 411201, China

G18873530267@163.com,2820558906@qq.com,whhuang@hnust.edu.cn,jiahongcai@mail.hnust.edu.cn, 1113482768@qq.com, 2949380334@qq.com

Abstract. With the development of digital currencies and 5G technology, blockchain has gained widespread attention and is being used in areas such as healthcare, industry and smart vehicles. Many security issues have also been exposed in the course of blockchain applications. Cryptography can ensure the security of data on the blockchain, the integrity and validity of data as well as the ability to authenticate users and anonymize them. This article therefore examines the cryptography underlying blockchain security issues, providing an overview of cryptographic homomorphic encryption, zero-knowledge proofs and secure multi-party computation commonly used in blockchains. At the same time, the development of quantum computing is bound to affect existing cryptographic systems, and blockchains applying these cryptographic systems are bound to be hit hard, so this article discusses four of the most promising post-quantum cryptography techniques available: hash-based public key cryptography, code-based public key cryptography, multivariate public key cryptography, and lattice-based public key cryptography.

Keywords: Blockchain, Homomorphic encryption, Post quantum cryptography, Secure multi-party computation, Zero-knowledge proof.

1 Introduction

In 2008, Satoshi Nakamoto introduced the concept of Bitcoin, a decentralized virtual currency, in his published paper "Bitcoin: A Peer-to-Peer Electronic Money System" [1]. Blockchain is a data structure that organizes blocks of data in a chain in chronological order and is capable of verifying, tracing, and reliably storing data on the chain through cryptography to ensure that the data on the blockchain is not tampered with and cannot be forged. The consistency of data on the blockchain is ensured through transaction signatures, consensus mechanisms and cross-chain technologies [2]. Blockchain technology is decentralized, traceable, tamper-proof, complete and open and transparent [3], which have attracted the attention of academia and industry. Moreover,

blockchain technology will be a revolutionary technology to solve the trust crisis in the future society [4].

As blockchain continues to develop, there is a greater demand for data protection, anonymity and untraceability [5] in many fields. Blockchain is no longer only used for virtual currencies [6] but is also being extended to various fields such as healthcare, copyright protection and finance. For example, blockchain is currently the most effective solution for personal privacy protection and sharing [7]. As part of the big data trend [8], the growing scale of the Internet of Things (IoT) [9] and the sharing of its data requires the use of blockchain. Many insider attacks are caused by trust issues, and blockchain can solve the problem of trust between untrustworthy users [10]. However, the rise of blockchain technology in various fields has brought about many security issues. For example, privacy protection and transaction protection. Also, with the development of quantum computing, which may break many cryptographic systems [11], the cryptography of blockchain will be severely challenged. With the emergence of various attacks, various cryptography-based blockchain security protection techniques are gradually developed [12]. Therefore, this paper will study the cryptography techniques in blockchain.

2 Blockchain Overview

2.1 The data structure of Bitcoin

In blockchain technology, a great deal of cryptographic knowledge is used to ensure the system's security. Cryptography greatly protects the privacy of data on the blockchain [13]. The blockchain is actually made up of blocks connected one by one, each block generating a hash value from the previous block [14]. The user can then verify the correctness of the data on the chain. Furthermore, if an attacker wants to tamper with the data on the blockchain, he must change all the blocks after that one.

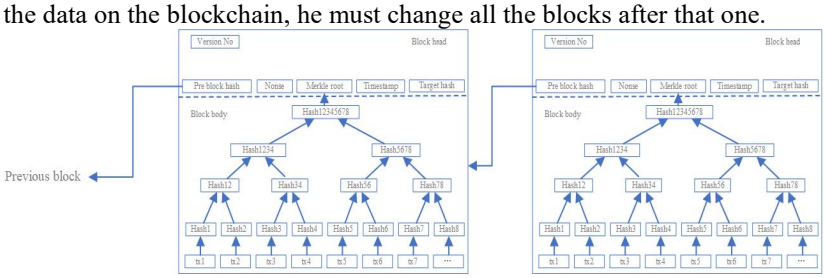


Fig. 1. Bitcoin structure diagram.

A block contains a block header and a block body. The block header holds the previous block's hash value, version number, random number Nonce, timestamp, Merkle root and the target hash value. In the block body is a Merkle tree, a typical binary tree. Its root is formed by the hashes of all the transactions in the block; the leaf nodes of the Merkle tree are the hashes generated by the transactions packed into the block, and the values of the non-leaf nodes are generated by concatenating the hashes of their children into a string and then hashing them, in this way working from the bottom up to generate the hash of the Merkle root. This structure allows a quick look at whether the

transactions packed in the block have been altered and, if found to have been altered, a quick way to locate the altered transaction.

2.2 Security challenges facing blockchain

Blockchain is a promising and growing technology but also faces many challenges. These challenges arise from the existing computer system [15-17] and network architecture [18-20], the consensus mechanisms used in the blockchain and the need for data protection [21-23]. With the development of blockchain and the development and promotion of the application of 5G technology, blockchain is gradually applied to various industries such as healthcare [24], industry [25], and finance [26]. While blockchain is widely used, it also raises a series of security and privacy issues. The digital currencies used in blockchain have also suffered many security threats, with attacks on trading platforms, theft of currencies and crimes committed by hackers and criminals using blockchain's anonymous transactions occurring frequently. At the same time, privacy breaches [27-28] in blockchain can also make the skeptical public of blockchain. These challenges are very detrimental to the development and innovation of blockchain.

3 Typical cryptography

3.1 Homomorphic encryption

The idea of homomorphic encryption was first introduced by Rivest, Adleman and Detouzos [29] (the R and A in "RSA") in 1978. Homomorphic data encryption allows direct manipulation of the encrypted data without the need for preliminary decryption of the operands. The effect of manipulating the encrypted data is the same as manipulating the data before encryption. In a blockchain, *FHE (fully homomorphic encryption)* ensures that the ledger information is not compromised but can be manipulated, even if the blockchain is attacked. *FHE* is a good solution to the problem of data being used on remote devices [30]. In 2009, Gentry [31] proposed a secure and reasonable *FHE* system that performs arbitrary addition and multiplication operations on the encrypted data while also acting on the pre-encrypted data. However, the performance of *FHE* algorithms is so poor that they are difficult to use in practice. In 2011, Brakerski et al. [32] proposed a new *FHE* algorithm, BVG, based on *Learning With Errors* (LWE), an alternative assumption to lattice encryption. The BGV system uses a somewhat more practical LWE assumption than the system proposed by Gentry in 2009. In 2013, Gentry [33] et al. proposed a simpler, *FHE* algorithm GSW based on LWE. In their scheme, they proposed a way to construct *FHE* of a new technique known as the approximate eigenvector method.

3.2 Zero-knowledge proofs

Zero-knowledge proof means that the prover does not need to reveal anything about the verification to the verifier who can also do the verification. With the use of zero-

knowledge proofs in blockchain, other nodes can verify the legitimacy and correctness of a transaction even if both parties do not reveal any information about the transaction. Zero-knowledge proofs are divided into interactive zero-knowledge proofs and non-interactive zero-knowledge proofs. Interactive zero-knowledge proofs require multiple interactions between the verifier and the prover, and the verifier improves the trustworthiness of the prover by performing multiple verifications to the prover. Non-interactive zero-knowledge proofs, on the other hand, allow the verifier and the prover to interact once with the aid of a machine. An overview of the development of zero-knowledge proofs is given next.

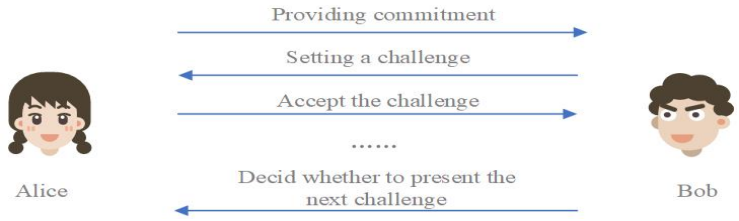


Fig. 2. Interactive zero-knowledge proof.

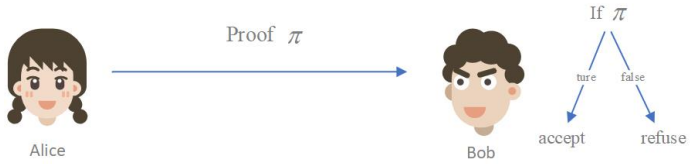


Fig. 3. Non-interactive zero-knowledge proof.

The concept of zero-knowledge proofs was introduced by S. Goldwasser, S. Micali and C. Rackoff [34] in 1988. After introducing this concept, zero-knowledge proofs have also been present in the overview of the theory. The emergence of blockchains and the need for data confidentiality has facilitated scholarly research on zero-knowledge proofs, which can address the difficulty of aligning blockchain privacy protection with data transparency [35]. In 2010, Groth proposed the key theory of zero-knowledge proofs ZK-SNARK [36] (zero-knowledge succinct non-interactive knowledge proofs). The provers can prove the correctness of their provided proofs mathematically to the verifiers without providing information about the proofs as the verifiers do. Subsequent scholars have worked on ZK-SNARK to reduce verification time and improve efficiency. the Pinocchio [37] protocol, proposed in 2013, is an improved version of ZK-SNARK, and in 2015 the blockchain application Zcash was used to build ZK-SNARK [38], a widespread application of zero-knowledge proofs. In 2016, Groth [39] proposed Groth16, which is also based on an improved version of ZK-SNARK with asymmetric pairing, and the proof will be more efficient.

ZK-SNARKS requires public reference strings for provers and verifiers for trustworthy settings when performing zero-knowledge proofs, but these public strings are again provided by a small group of people, and thus are vulnerable to attack by some malicious nodes [39]. As a result, research now prefers to discard trusted settings. In

2018, the Bulletproofs algorithm was introduced to eliminate the need for trusted settings. It is a more efficient algorithm that produces proofs of logarithmically transformed size, which would be very beneficial for storing proofs in the blockchain. Moreover, Bulletproofs can also merge and compress proofs of the same scope, reducing the size of the space occupied by the blockchain. In 2018, a new zero-knowledge proof scheme ZK-STARKS was also proposed [40], a zero-knowledge proof scheme that does not require trustworthy settings. ZK-STARKS has better scalability, and its proof and verification times are linearly and logarithmically related to the initial computation time, respectively. As the initial size increases, its proof and verification times do not increase significantly. The more widely used non-interactive zero-knowledge proofs for blockchain applications are ZK-SNARKS, Bulletproofs and ZK-STARKS.

3.3 Secure Muti-party Computation

Secure Muti-party Computation (SMPC) is derived from the "millionaire problem" proposed by Professor Yao in 1982, i.e.. Collaborative multi-party computing with third-party guarantees may carry the risk of information leakage from third-party organizations. SMPC enables distributed parties to jointly compute arbitrary functions without revealing their own private inputs and outputs. In the SMPC scenario, there are $n(n \geq 2)$ participants performing multi-party collaboration to compute an objective function $f(x_1, x_2, ..., x_n) = (y_1, y_2, ..., y_n)$, where $x_1, x_2, ..., x_n$ are the input information of each party. When the computation is finished, each participant does not get any other information except its own corresponding output y_i, Also no input information can be deduced from the input results.

Fig. 4. Secure multi-party calculation.

SMPC provides input correctness, computational correctness and output independence to analyze and capture the value in the data while protecting the privacy of the data. Similar to blockchain, secure multi-party computing supports collaborative computing by uniting untrusted users without a trusted third party. SMPC can collaborate with untrusted users in the blockchain without compromising their privacy to perform analytical calculations, analytical modelling, etc., on some sensitive data. It is also beneficial for blockchain applications to industries that require data analysis and storage, such as healthcare and industry. SMPC can be used in multi-signature, secret sharing, and random number generation, and the wallet ZenGo [41], a wallet released by Kzen,

does not require the use of mnemonic word and keys, but rather uses a gated signature method that combines the advantages of multi-signature and secret sharing. However, there are still difficulties that need to be addressed in the application of SMPC in blockchain, such as the fact that SMPC requires the participation of multiple honest nodes, malicious nodes may collude in the computation [42], and the efficiency of SMPC is low when the network transmission rate is low.

4 Post-Quantum Cryptography

If quantum computing develops as expected, it will inevitably disrupt existing cryptographic systems. For example, Bernstein and Daniel J. [43] point out that quantum computing's well-known Shor [44] algorithm and Grover [45] algorithm will have an impact on existing cryptographic systems. The Shor algorithm can theoretically solve the underlying mathematical problems on which the security of public key cryptographic algorithms depends, such as discrete logarithms and large integer decomposition problems. So these public-key algorithms, such as RSA and DSA, should not be secure. And Grover's algorithm will halve the security effect of some symmetric cryptographic algorithms and hash algorithms, requiring an increase in key length. Although it is only for theoretical threats, to take a long-term view, it is necessary to research early cryptography that can resist quantum attacks.

Post-Quantum Cryptography (PQC) resists an attacker even if the attacker has a quantum computer, also known as anti-quantum cryptography. The current mainstream schemes for PQC are hash-based public key cryptography, code-based public key cryptography, multivariate public key cryptography and lattice-based public key cryptography.

4.1 Lattice based cryptography

A lattice is a set of points in a high-dimensional space. Let $b_1, b_2, ..., b_n$ be a linearly independent set of bases ($n \le m$) in R^m and the lattice be the set of all linear combinations of integer coefficients of this set of bases. That is.

$$L(B) = \left\{ \sum_{i=1}^{n} x_i b_i, x_i \in Z, i = 1, 2, ..., n \right\} \tag{1}$$

The security of cryptographic algorithms relies on the underlying mathematical problem. Furthermore, the two main difficulties in lattice problems: are the difficulty of solving the shortest vector and the difficulty of solving the nearest vector. These problems have worst-case difficulty [46]. Many scholars have conducted many studies on lattice problems. The most famous algorithm is the LLL proposed by H. Lenstra, A. Lenstra and Lovasz in 1982 [47], however it can only solve the shortest vector in polynomial time with an approximation factor $(1+\varepsilon)\sqrt{4/\sqrt{3}}^{(n-1)/2}$ of (where it is a constant). Thus lattice-based cryptography is quantum resistant.

In the "Third Status Report on the Post-NIST Quantum Cryptographic Standardization Process" - NISTIR 8413 published in July 2022, four algorithms to be standardized were announced. And three of these are all lattice-based cryptographic schemes. The lattice-based cryptography algorithms have a better balance of security, public and private key size, and computational speed and are considered one of the most promising post-quantum cryptographic algorithms [48].

4.2 Hash-based signature algorithm

The hash-based signature algorithm was proposed by Leslie Lamport in 1979, but compared to other signature schemes, it did not to be widely used because it could produce relatively long signatures. With the arrival of the threat of quantum computing, it is gradually gaining attention again because hash-based signature counting has quantum-resistant properties, such as being resistant to attacks by Shor algorithms. It is one of the algorithms that have the potential to replace the traditional signature algorithm [49]. The one proposed by Leslie Lamport is a single hash signature, which cannot sign multiple messages, and was later improved by Ralph Merkle to form a multiple signature algorithm based on the Merkle tree. The public key is the root of the Merkle, and the key is each leaf node in the Merkle tree. The quantum resistance of Hash-based signature algorithm is based on the collision resistance of the Hash function because the current quantum algorithms cannot find the collision of the Hash. Swati Kumari [50] proposed an enhanced hash-based post-quantum cipher (PQC) architecture called signature-based Merkle hash multiplication (SMHM) algorithm. The hash Merkle signature-based algorithm is enhanced by using the Bernoulli-Karatsuba multiplication algorithm. Konstantinos Chalkias [51] proposed a scalable post-quantum cryptography scheme based on Merkle tree signatures suitable for blockchains and distributed ledgers, which can utilize dedicated chains or image structures to reduce the cost of key generation, signing, verification, and the size of signatures.

4.3 Code based cryptography

The code-based cryptosystem is derived from McEliece [52]. The algorithm is based on the integrable binary Goppa code called classical McEliece. The encryption and decryption of the McEliece cryptosystem are fast and secure. However, it is rarely used in practice because of the large size of the key, so one of the subsequent directions of research on code-based cryptography is to reduce the size of its key. The general linear decoding hard problem on which McEliece cryptographic algorithm is based is the NP-hard problem [53], so coding-based cryptography is very promising in quantum-resistant cryptography. Moreover, the NIST post-quantum cryptographic algorithm standard collection has coding-based cryptography second only to lattice-based cryptography. It is mainly used in public key encryption algorithms and only two for signature algorithms.

4.4 Multivariate-based Cryptography Regime

The security of the multivariate-based cryptography regime relies on solving the mathematical problem of solving a system of random multivariate quadratic polynomial equations over a finite field, which is nondeterministic polynomial time-hard. There is no finite algorithm for solving this problem. The *multivariate quadratic* polynomial problem is to find a solution in a system of quadratic polynomial equations in a given finite field. Since multivariate based cryptographic systems emerged late, they still need a lot of research and experiments to prove their security [54]. Although earlier multivariable-based signature systems have been breached and are no longer secure, multivariable-based signature algorithms are small in signature size and fast in inflammation. Therefore, multivariate based signature schemes are still very promising, and multivariate based signature algorithms are the most numerous in the NIST post-quantum cryptographic algorithm standards collection.

5 Conclusion

With the application of the blockchain, the blockchain needs to meet various different needs for data protection, multi-party participation and collaboration, and identity authentication in the face of different scenarios, and cryptography is crucial to the development of blockchain applications. In this paper, some classical cryptography and post-quantum cryptography in blockchain were studied. First, the origin of blockchain and its concepts were introduced, and the structure of Bitcoin and the security challenges it faces were presented. Subsequently, some classical cryptographic homomorphic encryption, zero-knowledge proofs and secure multi-party computation used in blockchains were investigated. Finally, four more promising post-quantum cryptograms were introduced for quantum computing attacks.

References

1. Nakamoto S. Bitcoin: A peer-to-peer electronic cash system[J]. Decentralized Business Review, 2008: 21260.
2. Liang W, Xiao L, Zhang K, et al. Data fusion approach for collaborative anomaly intrusion detection in blockchain-based systems[J]. IEEE Internet of Things Journal, 2021.
3. P Kumar, R Kumar, et al., PPSF: a privacy-preserving and secure framework using blockchain-based machine-learning for IoT-driven smart cities, IEEE Transactions on Network Science and Engineering 8 (3), 2326-2341, 2021.
4. He W, Zheng H. Literature Review on Block Chain: Technology, Principle and Development[C]//Journal of Physics: Conference Series. IOP Publishing, 2021, 1848(1): 012166.
5. Xu Z, Liang W, Li K C, et al. A Time-sensitive Token-Based Anonymous Authentication and Dynamic Group Key Agreement Scheme for Industry 5.0[J]. IEEE TII, 2021.
6. Gorkhali A, Li L, Shrestha A. Blockchain: A literature review[J]. Journal of Management Analytics, 2020, 7(3): 321-343.
7. W. Liang, Y. Yang, C. Yang, Y. Hu, S. Xie, K. C. Li, and J. Cao, "PDPChain: A Consortium Blockchain-Based Privacy Protection Scheme for Personal Data," IEEE Transactions on Reliability, pp. 1-13, 2022, doi: 10.1109/TR.2022.3190932.

8. Long J, Liang W, Li K C, et al. A Regularized Cross-Layer Ladder Network for Intrusion Detection in Industrial Internet-of-Things[J]. IEEE Trans. on Industrial Informatics, 2022.
9. Liang W, Xie S, Cai J, et al. Novel private data access control scheme suitable for mobile edge computing[J]. China Communications, 2021, 18(11): 92-103.
10. J. Zhao, J. Huang, et al., An effective exponential-based trust and reputation evaluation system in wireless sensor networks, IEEE Access 7, 33859-33869, 2019.
11. Nejatollahi H, Dutt N, Ray S, et al. Post-quantum lattice-based cryptography implementations: A survey[J]. ACM Computing Surveys (CSUR), 2019, 51(6): 1-41.
12. Li X, Liao J, et al. A New Dynamic ID-Based User Authentication Scheme Using Mobile Device: Cryptanalysis, the Principles and Design. Wire. Per. Comm, 2015, 85(1): 263-288.
13. Liang W, Xie S, Cai J, et al. Deep neural network security collaborative filtering scheme for service recommendation in intelligent cyber-physical systems. IEEE IoT J., 2021.
14. Liang W, Ning Z, Xie S, et al. Secure fusion approach for the internet of things in smart autonomous multi-robot systems[J]. Information Sciences, 2021, 579: 468-482.
15. M. Qiu, Z. Jia, et al., "Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP", J. of Signal Proc. Systems, 2007
16. M. Qiu, L. Yang, et al., "Dynamic and leakage energy minimization with soft real-time loop scheduling and voltage assignment", IEEE TVLSI, 18 (3), 501-504, 2009
17. M. Qiu, C. Xue, et al., “Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems,” IEEE DATE Conf., 1-6, 2007
18. M. Qiu, Z. Chen, et al., "Energy-aware data allocation with hybrid memory for mobile cloud systems", IEEE Systems J., 11 (2), 813-822, 2014
19. M. Qiu, C Xue, Z Shao, et al., "Efficient algorithm of energy minimization for heterogeneous wireless sensor network", IEEE EUC, 25-34, 2006
20. J. Li, Z. Ming, et al., "Resource allocation robustness in multi-core embedded systems with inaccurate information", Journal of Systems Architecture 57 (9), 840-849, 2011
21. Raikwar M, Gligoroski D, Kralevska K. SoK of used cryptography in blockchain[J]. IEEE Access, 2019, 7: 148550-148575.
22. H. Qiu, T. Dong, et al, "Adversarial attacks against network intrusion detection in IoT systems," IEEE Internet of Things Journal 8(13), 10327-10335, 2020
23. K. Gai, M. Qiu, S. Elnagdy, “A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance,” IEEE BigDataSecurity 2016
24. F. Hu, S. Lakdawala, et al., Low-power, intelligent sensor hardware interface for medical data preprocessing, IEEE Trans. on Info. Tech. in Biomedicine 13 (4), 656-663, 2009
25. H. Qiu, Q. Zheng, et al., "Topological graph convolutional network-based urban traffic flow and density prediction", IEEE Trans. on ITS, 2020
26. Y. Li, K. Gai, et al., "Intercrossed access controls for secure financial services on multimedia big data in cloud systems", ACM Trans. on Multimedia Comp., Comm., and App., 2016
27. M. Qiu, H. Qiu, et al., “Secure Data Sharing Through Untrusted Clouds with Blockchain-enabled Key Management”, the 3rd SmartBlock pp. 11-16, Oct. 2020, Zhengzhou, China.
28. K. Gai, Y. Zhang, et al., “Blockchain-enabled Service Optimizations in Supply Chain Digital Twin”, IEEE Transactions on Service Computing, 2022
29. Rivest R L, Adleman L, Dertouzos M L. On data banks and privacy homomorphisms[J]. Foundations of secure computation, 1978, 4(11): 169-180.
30. Gai K, Qiu M. Blend arithmetic operations on tensor-based fully homomorphic encryption over real numbers[J]. IEEE Transactions on Industrial Informatics, 2017, 14(8): 3590-3598.
31. Gentry C. Fully homomorphic encryption using ideal lattices[C]//Proceedings of the forty-first annual ACM symposium on Theory of computing. 2009: 169-178.

32. Brakerski Z, Vaikuntanathan V. Fully homomorphic encryption from ring-LWE and security for key dependent messages, Crypto. Conf., Springer, Heidelberg, 2011: 505-524.
33. Gentry C, Sahai A, Waters B. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based[C]//Annual Cryptology Conference. Springer, Berlin, Heidelberg, 2013: 75-92.
34. Goldwasser S, Micali S, Rackoff C. The knowledge complexity of interactive proof-systems, Providing Sound Found. for Crypto.: On the Work of Shafi Goldwasser and Silvio Micali. 2019: 203-225.
35. Chor B, Goldwasser S, Micali S, et al. Verifiable secret sharing and achieving simultaneity in the presence of faults, 26th IEEE Sym. on Found. of Comp. Sci. (SFCS), 1985: 383-395.
36. Groth J. Short pairing-based non-interactive zero-knowledge arguments, Int'l Conf. on the Theory and Appl. of Crypt. and Info. Security. Springer, Berlin, Heidelberg, 2010: 321-340.
37. Parno B, Howell J, Gentry C, et al. Pinocchio: Nearly practical verifiable computation[J]. Communications of the ACM, 2016, 59(2): 103-112.
38. Banerjee A, Clear M, Tewari H. Demystifying the Role of zk-SNARKs in Zcash, IEEE conf. on application, info. and network security (AINS), 2020: 12-19.
39. Groth J. On the size of pairing-based non-interactive arguments, Int'l conf. on the theory and applications of crypto. techniques. Springer, Berlin, Heidelberg, 2016: 305-326.
40. Sasson E B, Chiesa A, Garman C, et al. Zerocash: Decentralized anonymous payments from bitcoin[C]//2014 IEEE symposium on security and privacy. IEEE, 2014: 459-474.
41. Lindell Y. Fast secure two-party ECDSA signing[C]//Annual International Cryptology Conference. Springer, Cham, 2017: 613-644.
42. Wang Z, Cheung S C S, Luo Y. Information-theoretic secure multi-party computation with collusion deterrence, IEEE Trans. on Info. Forensics and Security, 2016, 12(4): 980-995.
43. Bernstein D J, Lange T. Post-quantum cryptography[J]. Nature, 2017, 549(7671): 188-194.
44. Shor P W. Algorithms for quantum computation: discrete logarithms and factoring, 35th annual symposium on foundations of computer science. IEEE, 1994: 124-134.
45. Grover L K. A fast quantum mechanical algorithm for database search[C]//Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. 1996: 212-219.
46. Esgin M F, Steinfeld R, et al. Short lattice-based one-out-of-many proofs and applications to ring signatures, Int'l Conf. on Applied Crypto. and Netw. Secu., Springer, 2019: 67-88.
47. Lenstra A K, Lenstra H W, Lovász L. Factoring polynomials with rational coefficients[J]. Mathematische annalen, 1982, 261(ARTICLE): 515-534.
48. Micciancio D, Regev O. Lattice-based cryptography[M]//Post-quantum cryptography. Springer, Berlin, Heidelberg, 2009: 147-191.
49. Merkle R C. Secrecy, authentication, and public key systems[M]. Stanford university, 1979.
50. Kumari S, Singh M, Singh R, et al. Signature based Merkle Hash Multiplication algorithm to secure the communication in IoT devices, Knowledge-Based Syst., 2022, 253: 109543.
51. Chalkias K, Brown J, Hearn M, et al. Blockchained post-quantum signatures, IEEE iThings/GreenCom/CPSCom/SmartData, 2018: 1196-1203.
52. McEliece R J. A public-key cryptosystem based on algebraic[J]. Coding Thv, 1978, 4244: 114-116.
53. Chaulet J, Sendrier N. Worst case QC-MDPC decoder for McEliece cryptosystem[C]//2016 IEEE International Symposium on Information Theory (ISIT). IEEE, 2016: 1366-1370.
54. Ding J, Yang B Y. Multivariate public key cryptography[M]//Post-quantum cryptography. Springer, Berlin, Heidelberg, 2009: 193-241., last accessed 2016/11/21.

An Efficient Detection Model for Smart Contract Reentrancy Vulnerabilities

Yuan Li[1], Ran Guo[2*], Guopeng Wang[3*], Lejun Zhang[1,2*], Jing Qiu[2], Shen Su[2], Yuan Liu[2], Guangxia Xu[2], and Huiling Chen[4]

[1] College of Information Engineering, Yangzhou University, Yangzhou, 225127, China
[2] Cyberspace Institute Advanced Technology, Guangzhou University, Guangzhou, 510006, China
[3] Engineering Research Center of Integration and Application of Digital Learning Technology, Ministry of Education, Beijing, 100039, China
[4] Department of Computer Science and Artificial Intelligence, Wenzhou University, Wenzhou 325035, China
MX120200526@yzu.edu.cn; guoran@gzhu.edu.cn; wangguopeng@sohu.com; zhanglejun@yzu.edu.cn; qiujing@gzhu.edu.cn; johnsuhit@gmail.com; yuanliu@gzhu.edu.cn; xugx@gzhu.edu.cn; chenhuiling.jlu@gmail.com

Abstract. We propose a novel smart contract re-entry vulnerability detection model based on BiGAS. The model combines a BiGRU neural network that introduces an attention mechanism with an SVM. We start from the data features of smart contracts, learn the model layer by layer to achieve feature extraction and vulnerability identification, introduce batch normalization, Dropout processing and use improved model classifiers to improve the vulnerability identification accuracy, model convergence speed and generalization capability of smart contracts. We had conducted numerous experiments, and the experimental results showed that BiGAS Detection Model has a strong vulnerability detection ability. The accuracy of vulnerability detection reached 93.24%, and the F1-score was 93.17%. We compared our approach with advanced automated audit tools and other deep learning-based vulnerability detection methods. The conclusion was that our method is significantly better than the existing advanced methods in detecting smart contract reentrancy vulnerabilities.

Keywords: smart contract, reentrancy vulnerability, bidirectional gated recurrent neural network, SVM

1 Introduction

With the development of blockchain technology, smart contracts have attracted a lot of attention in recent years. They are widely used because they can reduce the cost of trust compared with traditional contracts. Smart contracts are made up of computer code written by humans, which inevitably carries the risk of flaws and vulnerabilities. If the vulnerabilities are exploited by hackers, they will lead to the theft of assets that cannot

be recovered, ultimately causing huge losses to users. Therefore, the current research on smart contract vulnerability detection is particularly important.

A smart contract represents the transaction requirements of two parties to a contract in the form of a digital rule, which usually contains a set of predefined states. When a user on the blockchain makes a transaction (calling the address of a contract), the contract can perform a transfer of state driven by the transaction [1]. However, there is no effective way to guarantee the security of smart contract code [2]. Smart contracts can simplify and accelerate the development of various applications, but they also bring some problems [3]. In 2016, hackers attacked decentralized autonomous organization (DAO) using a reentrancy vulnerability, resulting in the theft of over 3.6 million ETH. Since then, Ether split into two chains, ETH and ETC [4]. In May 2021, the flash.sx smart contract suffered a reentrancy attack in which approximately 1.2 million EOS and 462,000 USDT were stolen one after another. Although it passed a security audit before then, the reentrancy attack was not detected in time. We propose a new approach that goes beyond the rule-based framework. The contributions of the proposed model are listed below.

1. The BiGAS Detection Model is proposed to solve the problem that the use of Softmax as a recurrent neural network classifier leads to the lack of generalization ability of the smart contract vulnerability identification model and cannot be better applied to vulnerability detection and classification.
2. BiGAS Detection Model combines bidirectional gated recurrent neural network (BiGRU), attention mechanism and SVM. The loss function of SVM is squared hinge loss (or maximum edge loss). For the problem of judging whether there is a vulnerability, using the hinge loss does have good performance, and using the squared hinge loss will make the model more robust on this basis.
3. BiGAS Detection Model is used for smart contract vulnerability detection. The performance is compared with existing traditional vulnerability detection tools and vulnerability detection tools incorporating deep learning. The experimental results show that our deep learning-based approach outperforms state-of-the-art smart contract vulnerability detection tools and our method has high practicality and application value in smart contract reentrancy vulnerability detection.

The article is divided into five chapters, which are structured as follows. The Introduction section introduces the security issues of smart contract reentrancy vulnerabilities in recent years and the significance and contribution of our research. The Related Work section presents existing practical research combining SVM with neural networks. The construction and principles of the model are described in Materials and Methods. The Results and Discussion section describes the experimental procedure and analysis of the results. We compared the proposed method with other currently available methods in the experimental section. The Conclusion section summarizes the strengths, weaknesses and future research directions of our study.

2 Related Work

There are many existing studies that combine SVM with neural networks for various applications. Han Qiu et al. propose a machine learning-based selective encryption design that combines SVM with BSNs [5]. A novel approach to component assembly inspection based on mask R-CNN and Support Vector Machines was proposed by Huang et al. [6]. The method combines CNN and SVM to construct an SVM classification model to identify assembly defects. Pedro F. et al. proposed an object detection method based on discriminatively trained Part-Base V models [7]. They combine a margin-sensitive approach for data-mining hard negative examples with a formalism they call latent SVM. Agarap A F combines convolutional neural network and SVM to construct image classification architecture [8]. Ross Girshick et al. pro-posed rich feature hierarchies for accurate object detection and semantic segmentation [9]. Abdulrahman proposed a time series classification method combining SVM with an echo state network (ESN), replacing the linear readout func-tion of the output layer with the radial basis function kernel of the SVM [10]. Yichuan Tang proposed Deep Learning using Linear Support Vector Machines demonstrate a small but consistent advantage of replacing the softmax layer with a linear support vector machine [11]. Agarap et al. proposed a neural network architecture combining gated recurrent unit and support vector machine for intrusion detection in network traffic data [12].

3 Materials and Methods

3.1 Vulnerability detection Processing

The process of vulnerability detection for smart contracts by the model we designed (BiGAS Detection Model) is as follows. Fig. 1 shows the diagram of the model we designed.

(1) Perform data clean up on the original smart contract. Remove blank lines, spaces, comments, non-ASCII values, and other information that is not relevant to the reentrancy vulnerability. Then, the code statements related to specific operations in the program code are extracted, and these semantically and functionally related code fragments are combined into small code fragments for analysis.

(2) Convert the smart contract into a smart contract fragment after data cleaning has been performed. First, the user-defined variables and functions are mapped to symbolic names, respectively. Information such as keywords and III operators of the smart contract are to be preserved in this process. After that, the contract fragments represented by the symbols are divided into a series of token lists preserving the semantic order by word partition analysis and converted to vector representation by word2vec.

(3) The data are randomly initialized and the training and test sets are divided in a ratio of 8:2. The data from the training set is fed into a BiGRU model with a self-attentive mechanism for feature extraction learning. BiGRU can extract deep-level features from the input vectors. The BiGRU model can be considered as consisting of two

parts, the forward GRU and the reverse GRU, and the unit state of the GRU is calculated from the model input and its learning parameter values.

(4) The loss function of the SVM is used in the classification phase to compare the input data (i.e., predicted values) with the target. The decision function of SVM is used to calculate the prediction labels, and the calculated loss values are used to measure how well the network predictions match the expected results.

(5) We use an optimization algorithm to minimize the losses, and we use the Adam optimizer, which has significant advantages [13]. It is simple to implement and computationally efficient. The parameters are updated without the scaling transformation of the gradient, the hyperparameters are well interpreted and the learning rate is usually adjusted automatically without adjustment or with only little fine-tuning. It is well suited for scenarios with large-scale data and parameters for unstable objective functions. The optimizer uses the loss values to update the weights of the network.

(6) The above steps are repeated in the experiment by multiple epochs for the purpose of training the model and outputting the optimized BiGAS detection model after the training.

(7) The test set is fed into the trained model to predict the labels of contract fragments to determine the presence of re-entry vulnerabilities, and the predictions are compared with the target data to obtain the evaluation metrics for the model.

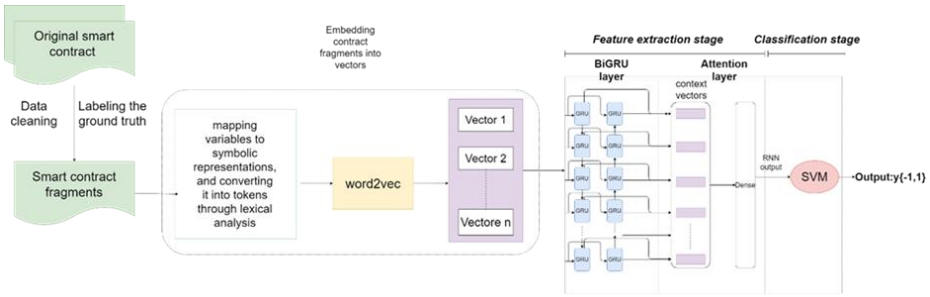


Fig. 1. Vulnerability detection Processing.

3.2 BiGAS Detection Model

BiGRU Neural Network. The following procedure calculates the hidden state of the GRU at the time step t. We use $h_t = GRU(x_t, h_{t-1})$ to symbolize the following process. Our number of samples is n, the number of inputs is d, the number of hidden cells is h, the input vector $x_t \in R^{n*d}$,$h_{t-1} \in R^{n*h}$ at the tth time step, the reset gate r_t and the update gate z_t are expressed as below.

$$r_t = \sigma(x_t W_{xr} + h_{t-1} W_{hr} + b_r) \tag{1}$$

$$z_t = \sigma(x_t W_{xz} + h_{t-1} W_{hz} + b_z) \tag{2}$$

$$\widetilde{h_t} = \tanh(x_t W_{xh} + (r_t \odot h_{t-1}) W_{hh} + b_h) \tag{3}$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \widetilde{h_t} \tag{4}$$

Self-attention mechanism. To highlight the importance of the impact of different keywords on the reentrancy vulnerability, a self-attentive mechanism is introduced after the BiGRU layer of the model to assign weights to different words. The input of the Attention mechanism layer is h_t, which is the hidden vector of the output processed by the BiGRU neural network layer in the previous layer.

First, we input the hidden layer vector to the fully connected layer (with weight W and bias b) to obtain u_t, which is stated in (5).

$$u_t = \tanh\,(Wh_t + b) \tag{5}$$

Then, we use this vector to calculate the calibration vector α_t (weights normalized by the attention mechanism), which is stated in (6).

$$\alpha_t = \frac{\exp\,(u^T u)}{\sum_t(\exp\,(u^t u))} \tag{6}$$

The u^t in the above equation is the average optimal vector corresponding to the current time step t, which is different for each time step. This vector is obtained by training. The output of the attention mechanism layer we express in (7).

$$s_t = \sum_{i=1}^{t} \alpha_t h_i \tag{7}$$

Classification Modul. The strategy of SVM is to minimize the loss function, and depending on the classification task our problem can be converted into (8).

$$L = \min_w \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \max\big(0,\ \ 1 - y_i(w^T x_i + b_i)\big) \tag{8}$$

Briefly, when SVM is introduced as the last layer of a neural network model, the parameters will be learned by optimizing the objective function of the SVM. In addition, the process uses the loss function of SVM instead of the cross-entropy function to calculate the loss values. In the above equation, L is the loss value, w is the weight of the output layer, n denotes the length of the output vector, i.e., the number of labels, y_i denotes the factual label of the unique thermal encoding, x_i denotes the input, b_i denotes the bias, C is the penalty level of the error sample, and T is the transposition character.The double-regularized SVM is denoted as L2-SVM, one of the improved algorithms of SVM, converts soft intervals into hard intervals, i.e., linearly indistinguishable into linearly distinguishable. In this case, the objective function is simply a constant factor 1/C added to the diagonal of the kernel function, which can be treated as a minor modification of the kernel function. It is differentiable and more stable compared to L1-SVM, so L2-SVM is used for the network structure in this paper, and its mathematical representation is as (9).

$$L = \min_{w} \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n} \max\left(0,\ 1 - y_i(w^T x_i + b_i)\right)^2 \quad (9)$$

The decision function of the support vector machine generates a score vector for each classification, in order to obtain a mathematical representation of the predicted label of the data as (10).

$$predicted_{class} = argmax\left(sign(wx + b)\right) \quad (10)$$

4 Results and Discussion

4.1 Evaluation Metrics

The whole experiment is divided into a training phase and a testing phase. Our job in the training phase is to optimize the model parameters by learning the loopholes to get a trained model. The testing phase of our work is to use the test data as input to the trained model and output the prediction results of vulnerability detection. The prediction results are compared with the real tags to measure the performance of our model. In this paper, we used widely used metrics, including accuracy (ACC), true positive rate or recall (TPR), false positive rate (FPR), precision (PRE), and F1- score (F1).

The whole experiment is divided into a training phase and a testing phase. Our job in the training phase is to optimize the model parameters by learning the loopholes to get a trained model. The testing phase of our work is to use the test data as input to the trained model and output the prediction results of vulnerability detection. The prediction results are compared with the real tags to measure the performance of our model. In this paper, we used widely used metrics, including accuracy (ACC), true positive rate or recall (TPR), false positive rate (FPR), precision (PRE), and F1- score (F1). The experimental model is built on a computer with Intel Core (TM) i7- 10875H CPU, NVIDIA GeForce GTX 2060 GPU, and 16GB RAM.As for the parameter settings, for each model, we use a 10-fold cross-validation method to select and train the best parameter values for training, corresponding to the effectiveness of the reentrancy vulnerability detection. The settings of all parameters we give in Tab. 1.

Table 1. Setting of hyperparameters

Hyper-parameters	BIGSA
Batch size	64
Cell size	300
Dropout rate	0.3
Recurrent dropout rate	0.5
Epochs	120
Learning rate	0.001
SVM C	0.5

4.2 Experimental Results and Performance Comparison

We compare the performance of our model with an advanced automatic safety analysis. Tab. 4 shows the comparison of experimental results data. The experimental results show that :(1) The vulnerability detection performance of existing automated security analysis needs to be improved. The Oyente[14] with the highest accuracy reached only 71.50%, the Oyente with the highest recall (TPR) reached only 50.84%, and the Securify [15] with the highest F1 value reached only 52.79%. (2) Our model (BiGAS Detection Model) has a high accuracy compared to the state-of-the-art tools. The accuracy of our model is 93.75%, which is 22.25% higher than that of the state-of-the-art Oyente. In addition, F1-score of our model is 93.44%, which is 40.65% higher than the F1-score of Securify. (3) Our model achieves a recall of 98.27%, which is much higher than that of advanced automatic safety analysis. This indicates that our model has a lower false positive rate. (4) All the metrics of our model outperform advanced automated security analysis. The experimental data shows that our approach has great potential for smart contract reentrancy vulnerability detection. We plot the results in Fig. 2.

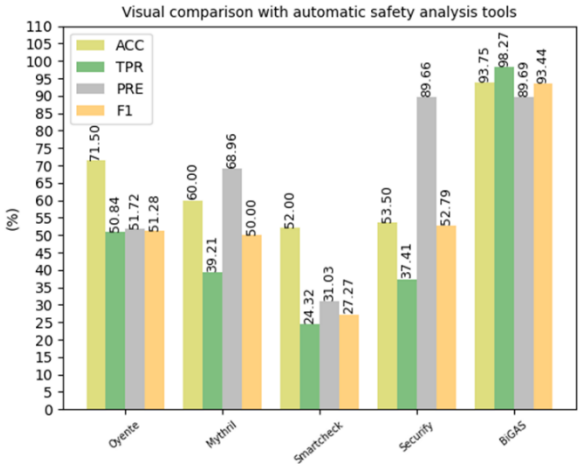


Fig. 2. Visual comparison with existing automatic safety analysis tools

To verify the effectiveness of our model, we also compared it with other neural network-based vulnerability detection methods, and the experimental results are shown in Tab. 2. Based on the data in Tab. 2, the following conclusions can be drawn. (1) According to models 1-5 in the table (sequential model): for our dataset, GRU performs slightly better than LSTM; bidirectional recurrent neural network is slightly better than unidirectional; introducing attention mechanism in the model will improve the model performance. (2) All metric values of our method are higher than existing neural network-based vulnerability detection methods, indicating the research significance and development potential of our method in combining deep learning with smart contract vulnerability detection. (3) Comparing our model with the model replaced with Softmax classifier, the accuracy of our model (the model combined with SVM) is 4.47% higher than that combined with Softmax, the recall of our model is 8.86% higher than

that combined with Softmax, the precision of our model is 3.27% higher than that combined with Softmax, and the F1-score of our model is 5.29% higher than that combined with Softmax. From the experimental data of these two models, we can conclude that our choice of SVM as the classifier of the model greatly improves the performance of the model. From these two sets of values, SVM as a classifier gives our BIGAS Dection Model a higher vulnerability detection capability. And even without SVM (i.e., BiGRU-ATT-Softmax model) shows that it is not inferior to the vulnerability detection capability of existing more advanced methods, which indicates that the flexible framework we build is more suitable for the task of detecting reentrancy vulnerabilities in smart contracts.

Table 2. Performance comparison with Neural Network Based Methods.

Models \ Metrics	ACC (%)	TPR (%)	PRE (%)	F1 (%)
RNN	78.84	75.65	82.10	79.56
GRU	83.33	73.28	81.66	81.79
LSTM	82.78	76.41	85.26	80.06
BLSTM	85.36	85.57	85.23	84.56
BLSTM-ATT [16]	87.27	88.48	86.45	87.14
DR-GCN [17]	81.47	81.80	72.36	76.39
TMP [17]	84.48	82.63	74.06	78.11
CGE [18]	89.15	87.62	85.24	86.41
BiGRU-ATT-Softmax	89.28	89.41	86.42	88.15
BiGAS	93.75	98.27	89.69	93.44

As shown in Fig. 3, we plot the ROC curves of different methods under the same threshold value. We use it to compare the classification effect of different methods for vulnerability detection. The area under the ROC curve is also called the AUC value, and the larger the value, the better the classification effect of the model. From the figure, our method has the largest AUC value of 0.9474. This indicates that our method has the best classification of vulnerable and non-vulnerable, i.e., the best vulnerability detection performance.

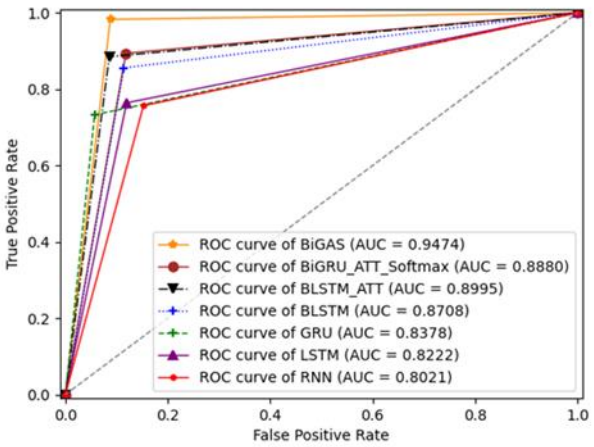


Fig. 3. ROC curves of Recurrent Neural Networks, this figure symbolizes the performance visualization results of different method

5 Conclusions

In this paper, we proposed a novel smart contract reentrancy vulnerability detection model based on BiGAS (which combines BiGRU-ATT and SVM). Starting from the original data features, the model learns layer by layer to achieve feature extraction and vulnerability identification, introducing batch normalization, Dropout processing and using SVM to improve the model classifier to improve the vulnerability identification accuracy, model convergence speed and generalization capability of smart contracts. In addition, we compared with existing security analysis and deep learning-based vulnerability detection methods. The results of the experiments showed that BiGAS Detection Model has better performance in terms of prediction accuracy, F1- score and classification effect. All the experimental data confirmed the effectiveness and practicality of our approach in dealing with the smart contract reentrancy vulnerability detection problem, which has certain advantages over the existing models.

The shortcoming of our research is that the model was only used for smart contract reentrancy vulnerability detection. Our future work will consider combining the expert knowledge model to detect more types of smart contract vulnerabilities such as timestamp dependencies, integer overflows, and other vulnerabilities on top of that. Our goal is to achieve a feasible and efficient model with wider applications.

Funding Statement This work is sponsored by the National Natural Science Foundation of China under grant number No. 62172353. And Innovation Fund Program of the Engineering Research Center for Integration and Application of Digital Learning Technology of Ministry of Education under grant number No.1221045.

References

1. K. F. Zhang, S. L. Zhang, S. Jin: The Security Research of Blockchain Smart Contract. Journal of Information Security Research, vol. 5, no. 3, pp. 192-206 (2019).
2. W. Q. Zou, D. Lo, P.S. Kochhar: Smart contract development: Challenges and opportunities. IEEE Transactions on Software Engineering, vol. 47, pp. 2084-2106 (2019). DOI:10.1109/TSE.2019.2942301.
3. T. Hu, X. Liu, T. Chen: Transaction-based classification and detection approach for Ethereum smart contract. Information Processing Management, vol. 58, no. 2, pp. 102462 (2021). DOI:10.1016/j.ipm.2020.102462.
4. N. AMIET: Blockchain Vulnerabilities in Practice. ACM Digital Library, vol. 2, no. 2, article 8, 2021
5. H. Qiu, M. Qiu, Z. Lu: Selective encryption on ECG data in body sensor network based on supervised machine learning. Information Fusion, ,55: 59-67(2020).
6. H. Huang, Z. Wei, L. Yao: A Novel Approach to Component Assembly Inspection Based on Mask R-CNN and Support Vector Machines. Information 2019, 10, 282 (2019).
7. P. F. Felzenszwalb, R. B. Girshick, D. McAllester and D. Ramanan: Object Detection with Discriminatively Trained Part-Based Models. in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 9, pp. 1627-1645 (2010). https://doi.org/10.3390/info10090282.
8. A. F. Agarap: An architecture combining convolutional neural network (CNN) and support vector machine (SVM) for image classification. Computer Science (2017).
9. R. Girshick, J. Donahue, T. Darrell and J. Malik: Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. 2014 IEEE Conference on Computer Vision and Pattern Recognition.580-587 (2014).
10. A. Alalshekmubarak, L. S. Smith: A novel approach combining recurrent neural network and support vector machines for time series classification. 2013 9th International Conference. Proceedings: Innovations in Information Technology (IIT). Al Ain, United Arab Emirates, 42-47 (2013).
11. Tang, Y.: Deep learning using linear support vector machines.(2013).
12. A. F. M. Agarap: A neural network architecture combining gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data. Proceedings: the 2018 10th international conference on machine learning and computing (ICMLC). 26-30 (2018).
13. Kingma, D. P., Ba, J: Adam: A method for stochastic optimization. (2014).
14. Luu L, Chu D H, Olickel H: Making smart contracts smarter. the 2016 ACM SIGSAC Conference. Proceedings: Computer and Communications Security (CCS). New York City, NY, USA, 254-269 (2016).
15. P. Tsankov, A. Dan, D. Drachsler-Cohen: Securify: Practical security analysis of smart contracts. the 2018 ACM SIGSAC Conference. Proceedings: Computer and Communications Security. Toronto, Canada,67-82 (2018).
16. P. Qian, Z. Liu, Q. He: Towards automated reentrancy detection for smart contracts based on sequential models. IEEE Access, vol. 8, pp. 19685-19695 (2020).
17. Y. Zhuang, Z. Liu, P. Qian: Smart Contract Vulnerability Detection using Graph Neural Network. the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI.3283-3290 (2020).
18. Z. Liu, P. Qian, X. Wang: Combining graph neural networks with expert knowledge for smart contract vulnerability detection. IEEE Transactions on Knowledge and Data Engineering, (2021).

Using Convolutional Neural Network to Redress Outliers in Clustering based Side-Channel Analysis on Cryptosystem

An Wang[1,2], Shulin He[3], Congming Wei[1,*], Shaofei Sun[1], Yaoling Ding[1,4], and Jiayao Wang[5]

[1] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China
[2] State Key Laboratory of Cryptology, P.O. Box 5159 Beijing 100878, China
[3] School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China
[4] State Key Laboratory of Information Security (Institute of Information Engineering, Chinese Academy of Sciences), Beijing 100093, China.
[5] The 31461 Unit of the Chinese People's Liberation Army, Shenyang 110001, China
{wanganl,hsl21,7520220100,sfsun,dyl19}@bit.edu.cn, 355688762@qq.com
*Corresponding author: Congming Wei

Abstract. Blockchain, designed with cryptographic technology, is widely used in the financial area, such as digital billing and cross-border payments. Digital signature is the core technology in it. However, digital signatures in public key cryptosystems face the threat of simple power analysis in Side-Channel Analysis (SCA). The state-of-the-art simple power analysis based on clustering mostly will appear outliers in the process of analysis, which will reduce success rate of key recover. In this paper, we propose a new SCA method with clustering algorithm Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and deep learning technology Convolutional Neural Network (CNN), called DBSCAN-CNN, to analyze public key cryptosystems. We cluster data with DBSCAN firstly. Then we train a CNN model based on the trusted clustering results. Finally, we classify the outliers of clustering results by the trained model. We mount the proposed method to analyze an FPGA-based elliptic curve scalar multiplication power trace which is desynchronized by simulating random delay. The experimental results show that the error rate of the proposed method is at least 69.23% lower than that of the classical clustering method in SCA.

Keywords: Side-Channel Analysis · Outlier Detection · DBSCAN · Convolutional Neural Network · Public-key cryptosystems

1 Introduction

Blockchain is widely used in various fields [3,4], which digital signature algorithm is one of the core technologies in it. Side-Channel Analysis (SCA) [6] is a method

in which an analyst uses physical signal changes to recover secret information, such as secret key used in digital signature algorithm. One of the analysis scenarios in SCA is non-profiled. It's consistent with the actual situation, because it's that the analyst can only obtain side-channel information on the target device, e.g., Simple Power Analysis (SPA) [6].

In recent years, machine learning has a great impact on some well-known countermeasures after being introduced into SCA [8,13]. For example, clustering-based SCA can relieve countermeasures against SPA, which clusters data in an unsupervised scenario according to "Samples of the same clusters are like each other while samples of different clusters are significantly different".

The most used clustering algorithm in SCA is K-Means [13,14], which requires the number of clusters to be specified. K-Means is a hard clustering algorithm based on the distance between cluster centers and sample points. So, it only deals with convex clusters data and cannot find outliers. These problems increase the computational complexity of cryptographic operations classification. So, we propose to use Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [12] instead of K-Means. Because DBSCAN can well avoid the above problems.

Clustering generally requires Principal Components Analysis (PCA) to perform a projection into a lower dimensional space for computational reasons. In this process, the loss of information may cause outliers which possibly have error labels after clustering. These outliers still contain feature information which can be used to distinguish between clusters. In addition, side-channel countermeasures, such as random delay, may cause the time samples corresponding to the same operation on the power trace to be misaligned, and bring some outliers [1], which will have a significant impact on the accuracy of clustering in SCA.

In order to eliminate the influence of outliers in clustering, outlier detection and processing are usually taken to replace outliers in the preprocessing of SCA [9,10]. However, if there are a lot of outliers, it will lead to error classification of the cryptographic operations. We choose deep learning to mine effective features in outliers and improve the accuracy of clustering. Applying deep learning to execute non-profiled SCA is a new way [7].

Among them, Convolutional Neural Network (CNN) is a suited model in SCA because of its ability to desynchronize. In 2019, Carbone et al. [2] used CNN models to attack the hardware RSA implementation and succeeded. Therefore, we use CNN to redress the error labels of outliers, which will improve the accuracy of clustering-based SCA.

Nascimento et al. [9] improved the unsupervised RSA horizontal clustering attack framework. They discussed the problem of outlier processing. In order to ensure high efficiency, they used the median of normal data to replace outliers that detected by distance from mean or Turkey's range test. Perin et al. [10] proposed to use deep learning to correct the error bits generated after K-Means clustering horizontal attack. For outlier detection and handling, they used the Turkey test method to detect and replace the outliers with the median value. But the approach of median replacement makes the analyst lose some information.

Because they are normal data perturbed by unconventional behaviors, outliers should contain some characteristics of normal data to a certain extent.

Contributions In this paper we propose a new SCA method DBSCAN-CNN. It only needs to collect a power trace on the target device, and then performs analysis using unsupervised clustering algorithm and deep learning.

- Combining clustering and deep learning, we propose DBSCAN-CNN SCA. Our method is unsupervised overall. Firstly, we use DBSCAN to cluster the lower-dimensional power segments after PCA projection. Then the original power segments is divided into training set and testing set according to the *clustered* labels and *outlier* labels. Finally, we train CNN with training set and test it with testing set. Through repeatability experiments, we prove that the accuracy of key recovery through our method is about 99%.
- Deep learning is used to extract effective features in outliers. Different from outlier processing methods such as median replacement, elimination, and increase samples to reduce noise impact in previous work, we use the effective feature mining method "deep learning" to help classifying outliers.
- We compare our proposed method with the classical clustering methods (i.e. K-Means and DBSCAN) in SCA. The results show that the error rate of our method is at least 69.23% lower than that of the classical clustering methods.

2 Preliminaries

2.1 Public-key Cryptosystems and Simple Power Analysis

In public-key cryptosystem, the communication parties use different keys: public key and private key. The sender encrypts the message with the receiver's public key. The receiver decrypts the message with its own private key. The security of existing public key cryptosystems depends on difficult mathematical problems, which makes it impossible to obtain the corresponding private key by calculation when the public key is known. Representative algorithms include: RSA, ElGamal, and ECC.

The core of ECC is scalar multiplication (Algorithm 1). Among them, R and P represent points on the elliptic curve and k is an integer operating parameter.

Algorithm 1 Scalar Multiplication

Input: P, $k = (k_{n-1}k_{n-2}\dots k_1k_0)$
Output: $R = kP$
$R = P$
for $i = n-1$ ***down to*** 0 **do**
  $R = 2R$
  if $k_i = 1$ **then**
    $R = R + P$
  end
end
return R

In Algorithm 1, there are two different operations: Double and Add. When k_i is 0, one times Double operation. When k_i is 1, one times Double operation and Add operation. Therefore, we can have following statements: (1) There are more Double operations than Add operations; (2) There is no continuous Add operations; (3) There is continuous Double operations; (4) An add operation always follows behind a Double operation.

SPA differentiates different cryptographic operations based on the difference in amplitude of the power trace. It's often used to analyze the square-and-multiply algorithm of RSA and the double-and-add algorithm of ECC. Next, we take an ECC power trace as an example to introduce how to perform SPA.

Fig 1 is a fragment of power trace during scalar multiplication calculation in ECC algorithm. There are obviously two patterns, one for Double operation "D" and one for Add operation "A". Double operation corresponds to a higher energy value while Add operation corresponds to a lower energy value. According to the above statements about two kinds of operations, we can uniquely obtain the binary scalar sequence corresponding to the operation sequence, to restore the scalar value used in the scalar multiplication process.

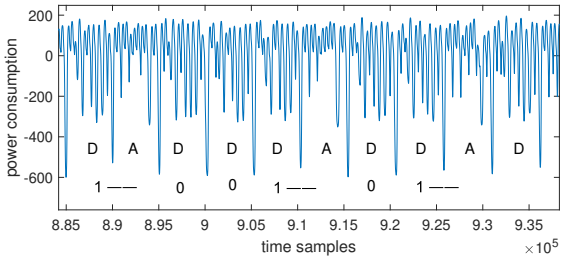


Fig. 1: SPA on An ECC Power Trace

When cryptanalysts want to obtain the private key in public key cryptosystem through SPA, they need to have certain cryptographic theory and analysis experience. But the keys for public-key cryptosystem are usually very long, SPA is time-consuming and laborious. In practice, cryptographic products will adopt some countermeasures against SPA and protect core chip security. Therefore, artificial intelligence methods such as machine learning and deep learning are introduced to improve state-of-the-art SCA techniques for the analysis of public-key cryptosystem.

2.2 Outlier Detection and Clustering

Outlier detection is an active research field in data processing, whose goal is to distinguish between normal and abnormal data [5]. Clustering is a common unsupervised machine learning method in outlier detection and SCA.

In this paper, we chose DBSCAN as clustering method to analyze ECC. DBSCAN is a density clustering algorithm [12]. This method doesn't need to specify the number of clusters. And it can pick out abnormal data points. The parameters ϵ and θ are used to control the size of determination range of the

search data points in clustering. The parameter ϵ represents the domain search radius around the data point. The parameter θ on behalf of the minimum number of data points within the search radius. According to this group of parameters, DBSCAN find core points, border points and noise from all points. Border points have the same clustered labels as their own core points.

Obviously, DBSCAN can make the points in the high-density area cluster into the same classes, and marks points that stray far outside the clusters as outliers. Therefore, by debugging the parameters ϵ and θ, we can make the number of correct labels of clustered data as much as possible and ensure that the cryptographic operations with the obtained clustered labels is not ambiguous. Then we we can get reliable training set and further analyze outlier data points picked out, to reduce the computational complexity.

3 DBSCAN-CNN Side-Channel Analysis Method

In view of the shortcomings of the median replacement and rejection of outliers in the previous work, we propose DBSCAN-CNN SCA method. Because random delay will lead to outliers in clustering, CNN can desynchronize and extract features. So, DBSCAN-CNN can effectively solve the effect of random delay.

This paper focus on the ECC implementation. We assume that the analyst can collect a power trace of the target device when it is running. And the network depth used in DBSCAN-CNN can be adjusted according to the actual situation of the dataset.

The framework of DBSCAN-CNN method in this paper is shown in Figure 2, which is mainly divided into three stages: preprocessing, DBSCAN clustering and CNN classification. The specific process of each stage is as follows:

Preprocessing Step 1.1 Collect a power trace T from the target device. The trace contains L time samples.

Step 1.2 Cut the trace T into several power segments $\{t_0, ..., t_u\}$, according to previous knowledge in Sect. 2 about the number of operations in ECC.

DBSCAN Clustering

Step 2.1 Reduce the high-dimensional segments to three-dimensional by PCA, get the three-dimensional data points $\{p_0, ..., p_u\}$, the coordinate of p_u is represented by (PC_1, PC_2, PC_3).

Step 2.2 Cluster data points $\{p_0, ..., p_u\}$ through DBSCAN. Adjust parameters ϵ and θ and get clustering results: normal clusters and outliers. There is $K(K \geq 2)$ classes in normal clusters.

Step 2.3 Divide the data into training set and testing set according to clustering results. The segments corresponding to normal clusters $\{t_0, ..., t_r\}$ is training data and their labels is $\{y_0, ..., y_r\}$, here $y_i \in [1, ..., K]$ for $i \in [0, new]$. The segments corresponding to outliers $\{t_0, ..., t_s\}$ is testing data. Also, $r + s + 1 = u$.

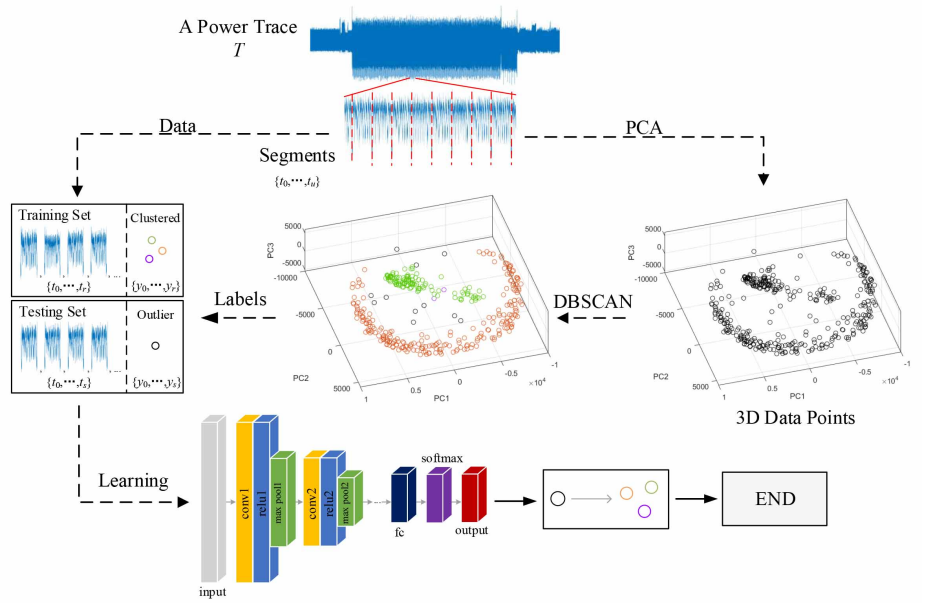


Fig. 2: DBSCAN-CNN Framework

CNN Classifying

Step 3.1 Extend the training set by data augmentation i.e., data $\{t_0, ..., t_{new}\}$ and labels $\{y_0, ..., y_{new}\}$, here $y_i \in [1, ..., K]$ for $i \in [0, new]$.

Step 3.2 Build CNN model with depth D. Its conventional kernel size is 3*3 and padding is 1.

Step 3.3 Train the CNN model on training set obtained in Step 3.1.

Step 3.4 Test the trained model with testing set. The output is K classification labels.

In [5], Järvinen et al. have given the method about how to recover key when they have K kinds of labels. In this case, there are $K!$ possible mapping π between labels and cryptographic operations. If K is small, the attacker can try all mappings. Then they find the correct best mapping, and finally recover the key bits.

4 Experimental Results

In this section, we take an FPGA-based ECC implementation as example to verify the performance of DBSCAN-CNN. All experiments were performed on MATLAB R2022a, and the CNN we used is designed using the Deep Network Designer App. Our experimental device is a laptop with 16GB memory and 2.4GHz CPU.

4.1 Experimental Data

The dataset we used is collected from a terminal chip with unprotected ECC implementation.

Figure 3(a) is the power trace collected during the ECC decryption process. Figure 3(b) is the result after zooming in on a part of it. As mentioned in Sect. 2.1, there is a repeating pattern in the power trace, each pattern corresponds to a double or add operation. However, it is obvious that leakage cannot be found visually like Figure 1. Then we divide the trace into segments according to the negative peaks of the trace as shown in Figure 3(c). There are 372 segments. Each segment corresponds to a double or add operation.

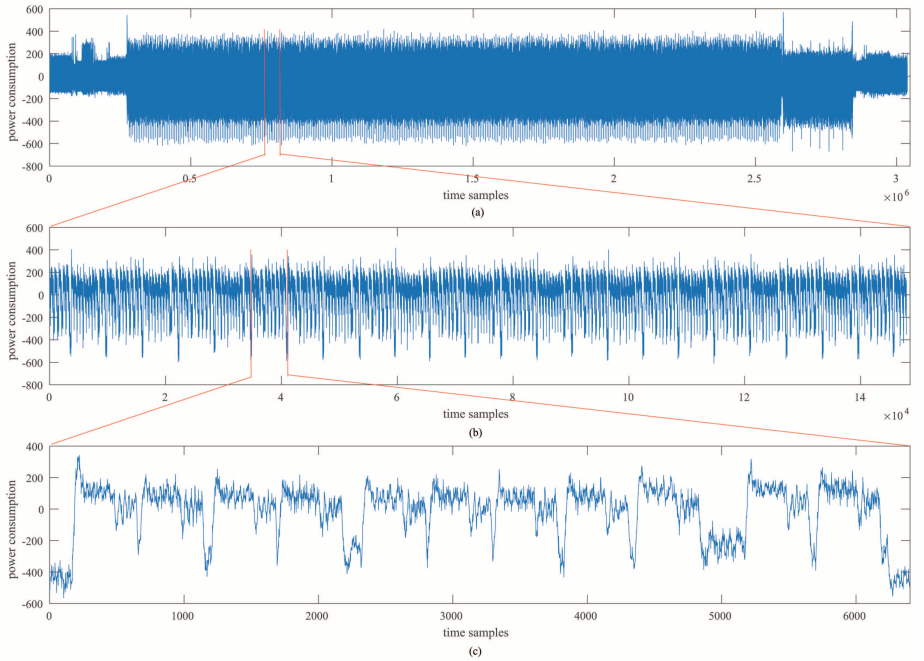


Fig. 3: ECC Power Trace

It is common to simulate the effect of countermeasures by modifying the dataset [11]. In order to show a random delay scenario, we desynchronized the power trace to simulate random delay countermeasure by inserting redundant traces into segments. Here's how we simulate:

We randomly select 20 segments from the segment set and insert square waves with different signal-to-noise ratio (SNR), duty cycle and sample length at random position. Some of the square waves we used are shown in Figure 4.

The simulation results are shown in Figure 5. Red stars indicate the insertion position.

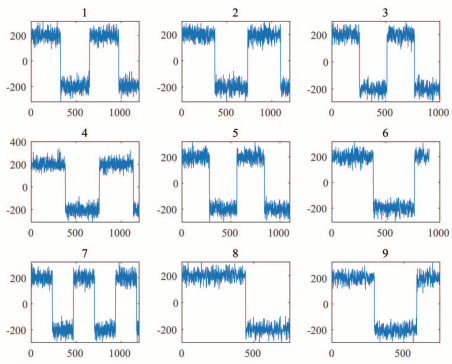


Fig. 4: The Redundant Square Waves

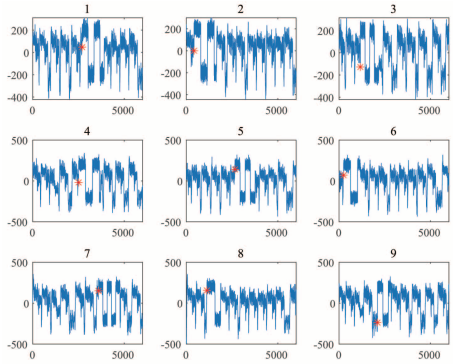
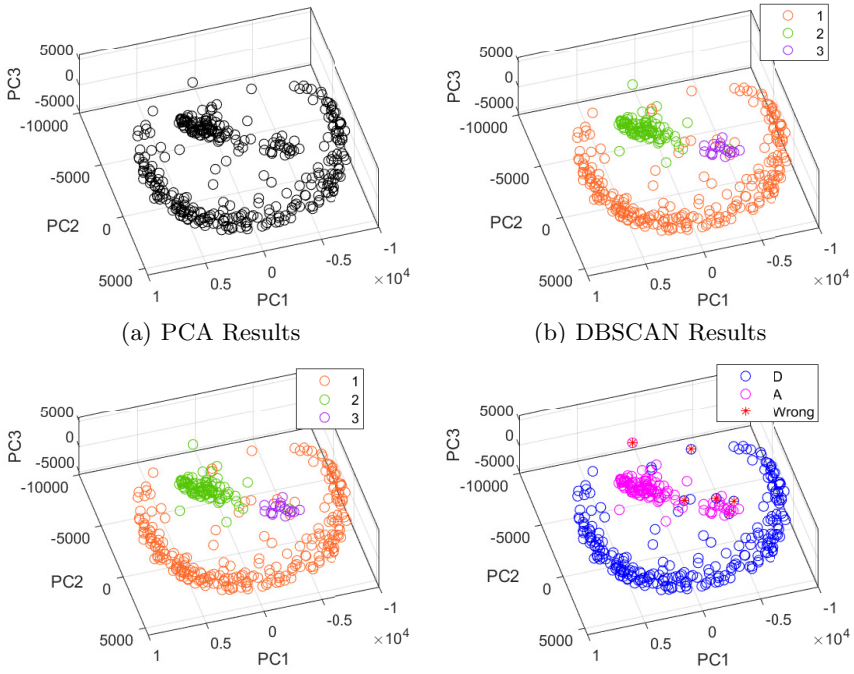


Fig. 5: Power Segments after Random Delay Simulation

4.2 Experiment Results

In subsequent chapters, we use "D" representing Double operations and "A" representing Add operations.

The results of DBSCAN-CNN are shown in Figure 6. As Figure 6(a) shows, after reducing the data to 3 dimensions using PCA, the data presents a distribution of two main clusters and a large number of outliers. Setting the parameters of DBSCAN as $\varepsilon = 1900$ and $\theta = 15$, DBSCAN clustering results are 3 classes



(a) PCA Results

(b) DBSCAN Results

(c) CNN Results

(d) CNN Misclassification Points

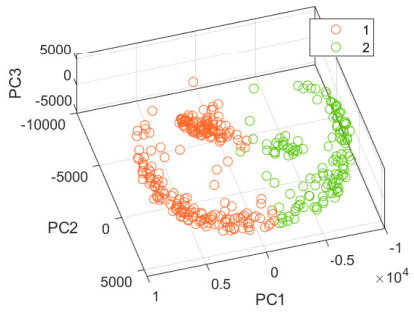
Fig. 6: DBSCAN-CNN Experiment Results

and 33 outliers as shown in Figure 6(b). We train a CNN on classes $1, 2, 3$ of DBSCAN clustering results and test on outliers. The CNN we used has 9 convolutional layers and learning rate 0.001 . As shown in Figure 6(c), CNN classifies all data into 3 classes. We mark all points that are misclassified by CNN under the premise that the mapping relationship between cryptographic operations and labels is known. As shown in Figure 6(d), CNN misclassified 6 points.

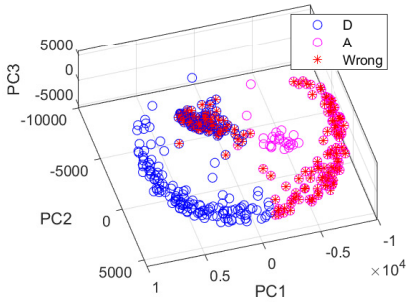
We repeated the experiment 9 times, redress the labels of misclassified points by voting and the results are shown in Table 1. In Table 1, we count the number of classification errors of 33 outliers after CNN classification. The CNN accuracy is the proportion of the overall number of outliers correctly classified by the trained model. Here, the DBSCAN-CNN classification accuracy obtained by voting is improved to 87.9%, and the number of misclassification are reduced to 4.

Table 1: Repeated Experimental Results

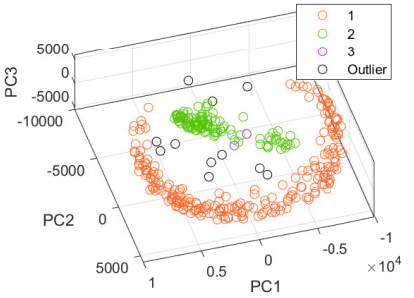
Experiment Index	1	2	3	4	5	6	7	8	9	**Vote**
CNN Misclassification Count	3	7	5	5	8	6	5	6	4	**4**
CNN Accuracy	0.909	0.788	0.848	0.848	0.758	0.818	0.848	0.818	0.879	**0.879**



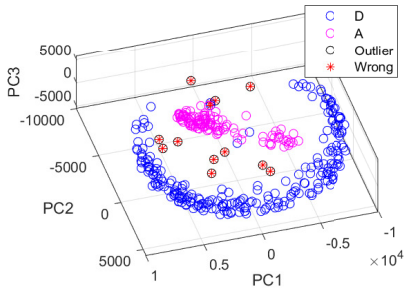
(a) K-Means clustering results



(b) K-Means clustering error points



(c) DBSCAN clustering results



(d) DBSCAN clustering error points

Fig. 7: Comparative Experiment

4.3 Comparisons

Finally, we also compare DBSCAN-CNN with classical clustering SCA methods K-Means and DBSCAN.

K-Means result is shown in Figure 7(a), points are directly divided into two clusters from the middle.

As shown in Figure 7(b), there are many misclassified points. DBSCAN result is shown in Figure 7(c) with parameters as $\varepsilon = 2200$ and $\theta = 2$. As shown in Figure 7(d), there are few misclassified points. We repeated the experiment 8 times: the average number of K-Means errors is 165 and the average number of DBSCAN errors is 13. It can be found from the comparative experiments that under the same dataset, the accuracy of K-Means is 55.65%, that of DBSCAN is 96.51%, and that of DBSCAN-CNN is 98.92%. The method proposed in this paper is better than single clustering SCA method when there are outliers during clustering.

In Table 2, we list error numbers, error rate and time consumption for three methods. Compared with classical clustering SCA, DBSCAN-CNN always has fewer misclassified points number, and the misclassification rate is reduced by at least 69.23%, which confirms that our proposed method is effective for clustering with outliers. Our method takes more time than classical methods because this time includes the training duration of the CNN model. But it's still within an acceptable range. Overall, if an attacker use our proposed method, he can complete the correction of errors and eventually recover secret information with less brute force complexity while spending some time.

Table 2: Average Error Nums, Error Rate and Time Consumption

Method	**Error Nums**	**Error Rate**	**Time Consumption**
K-Means	165	0.4435	0.451695 s
DBSCAN	13	0.0349	0.403635 s
DBSCAN-CNN	4	0.0101	303.97 s

5 Conclusion

This paper proposed a new SCA method for public-key cryptosystems in the scenario, which used deep learning to recover outliers in clustering results. As mentioned before, the accuracy of clustering results was limited by abnormal (but correct) data, so attackers needed to process these abnormal data. The method we proposed can extract correct features contained in abnormal data and label these data correctly. Among classical clustering SCA methods, K-Means can't distinguish outliers, DBSCAN can only distinguish outliers but can't process them. Our proposed DBSCAN-CNN trained CNN on normal data and tested on abnormal data to improve the classification accuracy of abnormal data. The experimental results showed that compared with classical clustering SCA, our method reduced error rate of clustering results by at least 69.23%.

Acknowledgement

This work is supported by National Key R&D Program of China (Nos. 2022YFB310 3800, 2021YFB3101500), National Natural Science Foundation of China (Nos. 62272047, 62002021), Beijing Institute of Technology Research Fund Program for Young Scholars, and Cryptographic Application Industry Chain Supply and Demand Docking Platform of New Energy and Intelligent Connected Vehicle Industry (2021-0181-1-1).

References

1. Basora, L., Olive, X., Dubot, T.: Recent advances in anomaly detection methods applied to aviation. Aerospace **6**(11), 117 (2019)
2. Carbone, M., Conin, V., Cornelie, M.A., Dassance, F., Dufresne, G., Dumas, C., Prouff, E., Venelli, A.: Deep learning to evaluate secure rsa implementations. IACR Transactions on Cryptographic Hardware and Embedded Systems pp. 132–161 (2019)
3. Gai, K., Guo, J., Zhu, L., Yu, S.: Blockchain meets cloud computing: a survey. IEEE Communications Surveys & Tutorials **22**(3), 2009–2030 (2020)
4. Gai, K., Wu, Y., Zhu, L., Zhang, Z., Qiu, M.: Differential privacy-based blockchain for industrial internet-of-things. IEEE Transactions on Industrial Informatics **16**(6), 4156–4165 (2019)
5. Järvinen, K., Balasch, J.: Single-trace side-channel attacks on scalar multiplications with precomputations. In: International Conference on Smart Card Research and Advanced Applications. pp. 137–155. Springer (2016)
6. Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Annual international cryptology conference. pp. 388–397. Springer (1999)
7. Kwon, D., Kim, H., Hong, S.: Non-profiled deep learning-based side-channel preprocessing with autoencoders. IEEE Access **9**, 57692–57703 (2021)
8. Lerman, L., Poussier, R., Markowitch, O., Standaert, F.X.: Template attacks versus machine learning revisited and the curse of dimensionality in side-channel analysis: extended version. Journal of Cryptographic Engineering **8**(4), 301–313 (2018)
9. Nascimento, E., Chmielewski, Ł.: Applying horizontal clustering side-channel attacks on embedded ecc implementations. In: International Conference on Smart Card Research and Advanced Applications. pp. 213–231. Springer (2017)
10. Perin, G., Chmielewski, Ł., Batina, L., Picek, S.: Keep it unsupervised: Horizontal attacks meet deep learning. IACR Transactions on Cryptographic Hardware and Embedded Systems pp. 343–372 (2021)
11. Picek, S., Perin, G., Mariot, L., Wu, L., Batina, L.: Sok: Deep learning-based physical side-channel analysis. Cryptology ePrint Archive (2021)
12. Ram, A., Jalal, S., Jalal, A.S., Kumar, M.: A density based algorithm for discovering density varied clusters in large spatial databases. International Journal of Computer Applications **3**(6), 1–4 (June 2010)
13. Ravi, P., Jungk, B., Bhasin, S.: Single trace electromagnetic side-channel attacks on fpga implementation of elliptic curve cryptography. In: 2019 Joint International Symposium on Electromagnetic Compatibility, Sapporo and Asia-Pacific International Symposium on Electromagnetic Compatibility (EMC Sapporo/APEMC). pp. 1–4. IEEE (2019)
14. Tang, M., Luo, M., Zhou, J., Yang, Z., Guo, Z., Yan, F., Liu, L.: Side-channel attacks in a real scenario. Tsinghua Science and Technology **23**(5), 586–598 (2018)

Secure File Outsourcing Method Based on Consortium Blockchain

Xuan You[1], Changsong Zhou[1], Zeng Chen[2], Yu Gu[1], Rui Han[1], and Guozi Sun[1](✉)

[1] School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing, 210023, Jiangsu, China. sun@njupt.edu.cn
[2] Jiangsu cancer hospital, Jiangsu, China.

Abstract. With the development of the big data era, the amount of data has entered an explosive growth phase. Limited by the constraints of cost, efficiency, and security of self-built storage systems, enterprises are forced to outsource files to cloud storage systems. However, the lack of file security and auditability in cloud storage systems continues to threaten the security of outsourced files. This paper designs and implements BFSOut, which is a secure file outsourcing method based on consortium blockchain. It uses Hyperledger Fabric and Interplanetary File System (IPFS) as the underlying storage engine, which solves the problem of cloud storage security issues. In BFSOut, in order to ensure the security of outsource files, the client-side offline block encryption is used. Furthermore, a dynamic hybrid encryption scheme is adopted, making the overall encryption effect more Efficient. Experimental performance analysis show that the system has good performance.

Keywords: Secure file outsourcing; Hybrid encryption; Hyperledger Fabric.

1 Introduction

With the rapid development of big data and cloud computing, more and more enterprises are using cloud storage services. Therefore, a large amount of data in the enterprise is stored in the cloud platform, such as employee record sheets, company financial data, and so on. When sharing files, users only need to share the link through the client software provided by the cloud platform, such as Dropbox and Google Drive. However, this way of outsourcing data faces great security threats and privacy leaks[1,2]. The user loses control of a file when it is uploaded to the cloud storage platform in plaintext. Although the platform provides the function of deleting the file, this deletion is executed in a soft method,

1. This work was supported by the National Natural Science Foundation of China (No.61906099), the Open Fund of Key Laboratory of Urban Land Resources Monitoring and Simulation, Ministry of Natural Resources (No.KF-2019-04-065).
2. Corresponding author: Guozi Sun, sun@njupt.edu.cn

as the automatic backup mechanism of the cloud platform will automatically retain a copy of the file. Which is worse, cloud service providers may leak user data to unauthorized entities to obtain illegal profits. In addition, third-party cloud platforms also have the problem of opaque auditing and the inability to guarantee integrity[3].

For the problems existing in a cloud platform, blockchain technology is an ideal solution[4,5]. Blockchain is a sharable database, and participants in different networks, sharing data by synchronizing ledgers. The ledger has the characteristics of traceability, decentralization, and immutable, which promotes credible interaction in an untrusted environment[6,7]. The blockchain uses smart contracts to execute the transaction process. Smart contracts are automatically executable codes that control operations according to contracts or agreements. It has transaction logic that can control the entire business life cycle[8]. Through the use of smart contracts, transactions on the blockchain can be fully automated and run without any human intervention. The use of blockchain technology[9] in the file-sharing system can make the file-sharing process more transparent, and its immutable feature ensures the integrity and traceability of the data. There are many methods to establish a blockchain network. Public blockchains require a lot of computing power to ensure the fairness of transactions. The privacy of transactions is extremely low or there is even no privacy at all[10], which are important considerations for enterprises. The consortium blockchain is controlled by multiple organizations, and these organizations can share the responsibility of maintaining the blockchain. When all participants need to obtain permission and are responsible for the blockchain, the consortium blockchain is the ideal choice[11]. In addition, the consortium blockchain has higher throughput and lower transaction delay, which can improve the overall performance of the system.

From the perspective of file itself, a feasible way to ensure the security of the sensitive information is to use some encryption scheme for encrypting before uploading the file to an untrusted server, while owner of the file is responsible for it. The security strength of the file depends on the effectiveness of the encryption key and the complexity of the encryption algorithm[12]. Common systems use a dynamic encryption method to encrypt all files in a unified manner as file is unstructured data.

Our contributions. The main contributions of this paper are demonstrated as follows:

1. This paper designs a file security outsourcing method based on consortium blockchain, to ensure the balance of file security and efficiency in the process of file outsourcing. It also explains the process of file outsourcing.
2. In the process of file outsourcing, a dynamic hybrid encryption scheme based on the combination of symmetric and asymmetric encryption algorithms to provides fine-grained protection for a single file.
3. For each operation step of the BFSOut system, its performance and consumption are tested, and the block generation of blockchain is tested.

Paper organization. The remainder of this paper is structured as follows. The second part reviews some work related to the method. The third part explains the specific details. The fourth part tests the method from multiple angles. Finally, the conclusion is in the fifth part.

2 Related work

2.1 Distributed blockchain storage systems

Azaria et al. developed a system called MedRec[13], which effectively manages and stores medical records based on blockchain. Using unique blockchain properties, MedRec can also be used for identity verification, accountability, and data sharing. Li Z et al. designed a distributed peer-to-peer network architecture based on the blockchain protocol[14] to ensure the security and stability of cloud data transmission and sharing. Moses et al. aiming at the problem that fingerprint templates are vulnerable to security attacks due to their asymmetry[15], proposed to protect encrypted fingerprint templates through the symmetric peer-to-peer network and symmetric encryption.

2.2 Data outsoucing and sharing based on blockchain

Li et al. designed a medical data fusion distributed privacy management system based on Fabric and IPFS[16] to overcome the disadvantages of traditional hospital data separation and solve the problems of electronic medical record data sharing and tracking difficulties. Tang et al. designed a data storage model based on Ethereum[17]. The purpose is to use smart contracts to implement some backend service functions, ensuring the transparency, openness, and traceability of services while effectively resisting DDoS attacks. Shen et al. proposed a secure access control scheme based on CP-ABE[18] to solve the problems of cloud storage decentralization and data sharing.

3 Secure file outsourcing method

In this section, we will first introduce BFSOut's secure file outsourcing model in detail. Then we will demonstrate the dynamic hybrid encryption model.

3.1 Secure file outsourcing model

This model designs a file outsourcing method with decentralized features based on the consortium blockchain, which allows data owners to upload or share files to any address that exists on the blockchain, while the file receiver can safely download the files in their address. Its model is shown in Figure 1.

In the file outsourcing model, it is mainly divided into the sender, receiver, dynamic hybrid encryption and decryption algorithm, smart contract and Fabric and IPFS network. The main process of this model is listed as follows:

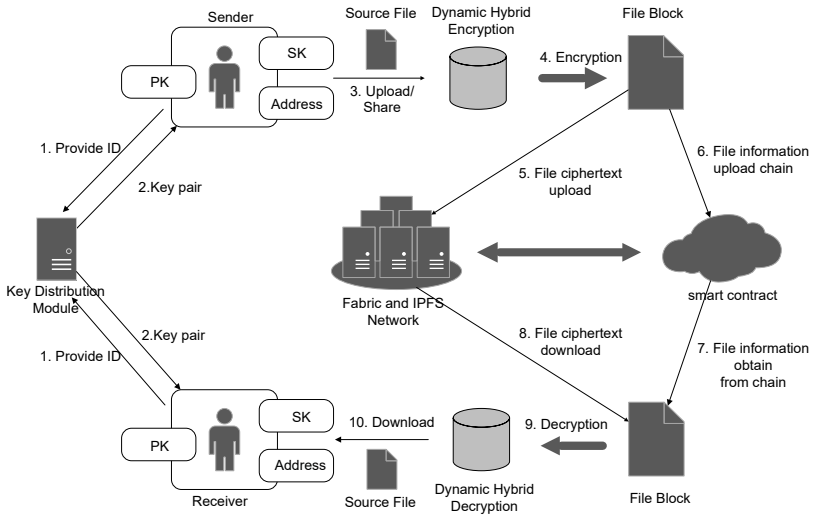


Fig. 1. Secure file outsourcing model

1. **System initialization.** The sender and receiver use the BFSOut client to generate a unique ID (step 1) and obtain the user's public and private key pair through the built-in key distribution module of the client (step 2). Then the public key is converted into a 272-bit address by the address conversion algorithm. At the same time, the public key and address are stored in the Fabric platform through a smart contract.
2. **File upload.** In the file upload process, the sender first initiates a file upload and sharing request (step 3). The client generates file blocks using the dynamic hybrid encryption algorithm (step 4). The file block is divided into file stream ciphertext(f_{sc}) and file digest ciphertext(f_{dc}). The f_{sc} is stored in the IPFS network in the form of slices (step 5). The f_{dc} should be written to the receiver's address through a smart contract and is permanently stored as a file record(step 6).
3. **File sharing.** The smart contract automatically determines whether the address of the file upload is the receiver's address. If the address is confirmed to be correct, the smart contract will write the f_{dc} to the corresponding receiver's address.
4. **File download.** The receiver confirms that the sender shared the file by checking its address. The smart contract will be called first to obtain the information of this file (step 7). Then download the f_{sc} through the file Hash to the IPFS network (step 8). Next, the original file is decrypted through the decryption module (step 9). Finally, the system will check the validity and correctness of the file, download and save the file in the local disk (step 10).

3.2 Dynamic hybrid encryption model

In the hybrid encryption scheme, the symmetric encryption algorithm is AES, and the asymmetric encryption uses the ECC. Symmetric encryption is usually faster than asymmetric encryption, but its security is lower, so they are often used in combination. The method uses AES encrypt real data, and ECC to encrypt parameters such as the key to the symmetric encryption algorithm. This method is more efficient in encryption, and the key is easier to manage. To enhance the file security level, we take the block-based encryption method. The file is divided into several blocks of the same size, while each block is responsible for different encryption processing.

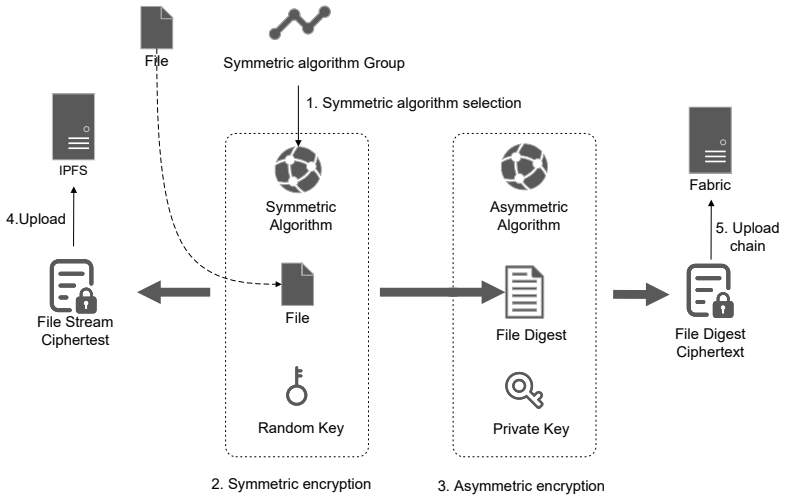


Fig. 2. Dynamic hybrid encryption model

When a file is uploaded through the client program, the client's encryption module will be triggered. To prevent file transmission from being stolen, the file encryption and decryption process are completely offline. The entire encryption process is shown in Figure 2. When file encryption is performed, it is first choose symmetric algorithm from Symmetric algorithm group (step 1). Then the system will generate a random symmetric key, use the key to perform symmetric encryption on the source file to generate the file stream ciphertext f_{sc} (step 2), and then the model will generate the file digest ciphertext f_{dc} by asymmetric encryption with some parameters, including symmetric encryption algorithm, symmetric key and random vector in block encryption (step 3). Finally, the model will perform a persistence operation, upload thef_{sc} to the IPFS server for persistent storage (step 4), and store the f_{dc} in the Fabric through a smart contract (step 5). The algorithm description is as follows.

Algorithm 1 describes the encryption process of this method. First, an original encryption key f_{s_k} is randomly generated when the encryption module is invoked after the file (f) is read. Then,f_{s_k} and f_{id} are combined into file encryption key f_k by Hash-based Message Authentication Code ($HMAC$). To prevent the key from being tampered with, the algorithm used by $HMAC$ is $HMAC-MD5$. In the block encryption process, each file block (B) will firstly generate a unique counter (ctr). Then f_{id} and ctr_i generate a unique IV_i vector associated with the file block through the XOR function. Then it uses the encryption algorithm (em) to encrypt the file block to get the file ciphertext block (CB_i). The purpose is to prevent the same or similar file blocks from being encrypted into the same ciphertext output, so that if an attacker decrypts one file, he cannot decrypt the others, ensuring that the block encryption is unique. Next, all file ciphertext blocks (CB) are composed of the f_{sc}, and the em, f_k, and ctr parameters are composed of the f_d. Finally, the pk and ECC algorithm are used to generate the f_{dc}. To analyze the time complexity of this algorithm, it is assumed that the length of a single file is n. The time consumption of this algorithm mainly lies in the block encryption part, that is, a symmetric encryption algorithm is required for each file block. Since the time complexity of AES is $O(n)$, the time complexity of this encryption algorithm is $O(n^2)$.

Algorithm 1 File Encryption

input: f_{id}, f, em
output: f_{dc}, f_{sc}

function FileEncryption(f_{id}, f, em)
  $f_{s_k} \leftarrow GenerateRandom()$
  $f_k \leftarrow HMAC(f_{s_k}, f_{id})$
  $B \leftarrow Call\ CreatBlocks(f)$
  //Generate random counters
  for $\forall B_i \in B$ **do**
    $ctr_i \leftarrow GenerateRandom()$
  end for
  //Blocks encryption using encryption method
  for $\forall B_i \in B$ **do**
    $IV_i \leftarrow f_{id} \bigoplus ctr_i$
    $CB_i \leftarrow Encrypt(em(B_i, f_k, IV_i))$
  end for
  $f_{sc} := (CB_1, ..., CB_n)$
  $f_d := (em, f_k, ctr)$ // generate file digest
  $f_{dc} \leftarrow Encrypt(ECC(fd, pk))$ // Keys encrytion using ECC
  return f_{dc}, f_{sc}
end function

The decryption process is reversed. Algorithm 2 describes the decryption steps of the file. The user takes the private key (sk) to decrypt f_{dc} into related parameters, and then uses these parameters to decrypt the file block CB and

Algorithm 2 File Decryption

input: f_{id}, f_{sc}, f_{dc}
output: f

function FileDecryption(f_{id}, f_{sc}, f_{dc})
  //Keys decryption using ECC
  $em, f_k, ctr \leftarrow Decrypt(ECC(f_{dc}, sk))$
  //Blocks decryption using encryption method
  for $\forall B_i \in B$ **do**
    $IV_i \leftarrow f_{id} \bigoplus ctr_i$
    $B_i \leftarrow Decrypt(em(CB_i, f_k, IV_i))$
  end for
  $f := (B_1, ..., B_n)$
  return f
end function

compose the file. Again, since decryption requires one decryption for each file block, all time complexity is $O(n^2)$.

4 Experimental evaluation

In this section, we analyze and test BFSOut's block generation and the system performance is tested.

4.1 Blockchain block analysis

The paper uses Fabric 2.0, the time for a single block generation is adjusted to 0.1s, the maximum number of transactions in a block is 10, and the smart contract is written in the Go language. In the BFSOut system, there are system initialization phases, file upload phases, and file sharing phases that will generate blocks. The public key and address will be uploaded during the system initialization phase, and a block will be generated in this phase. The file upload stage will generate the f_{dc} and Hash, and at the same time generate the encryption related parameters. The information of the file itself will generate a block, and the storage of other parameters will generate a block. So the stage will generate 2 blocks. During the file-sharing phase, the parameters are rewritten into the receiver address. So the file-sharing process will generate a blocks.

The blockchain system in BFSOut builds a pseudo-cluster composed of two organizations and a single channel as a test environment. Each organization has two Peer nodes. The blockchain system is initialized into 7 blocks, including 1 genesis block, 2 update anchor node blocks, and 4 instantiation chain code blocks.

4.2 System performance test

The performance test of BFSOut is to test the upload and download performance of various files of different sizes through experiments. The experiment

is carried out in an experimental environment composed of 3 computers. One of the machines is used as the server of IPFS and Fabric, and the other two machines take the responsibility of the file sender and receiver respectively. The server machine is running on the Centos 7.9 system and the client machine is running on Windows 10 system. The server machine is configured with Intel Xeon E-2225G processor, the IPFS uses version 0.4.23, and all components of the client are written using Go1.14.4.

The execution time is measured by considering the entire process of uploading and sharing on BFSOut. The test file uses a data set made of text and compressed files, and the data size ranges from 1M to 1000M. The result of each experiment is the average of 10 runs to reduce the randomness of a single test. The size of the page cache is one page, usually 4K. When linux reads and writes a file, it is used to cache the logical content of the file, thus speeding up access to the disk's images and data. In order to ensure the accuracy of the results obtained, the page cache is refreshed between each test.

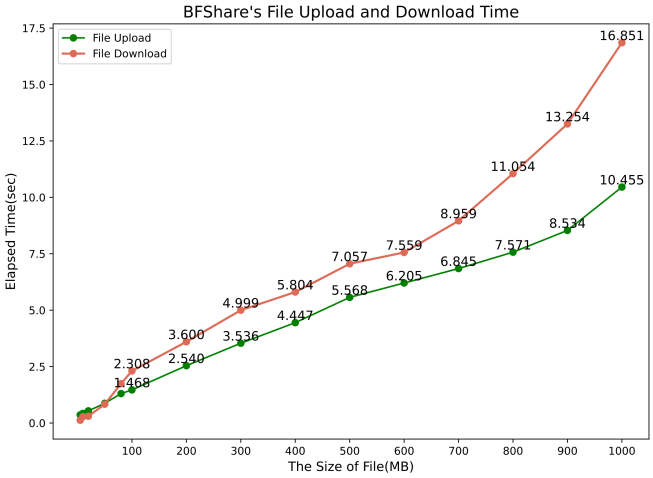


Fig. 3. BFSOut file upload and download time consumption

First, we evaluate the performance of BFSOut in uploading and downloading files of different sizes. Figure 3 shows a comparison of the time taken to write and read files to the server using the BFSOut. This performance data includes the entire file operation process, that is, file hybrid encryption and decryption, file storage and download, and related information insertion and query on the blockchain during file reading and writing. In the whole process, the time-consuming file upload and download increase linearly with the file size. When the file is less than 100M, the uploading time is slightly longer than the downloading time. This is mainly because the file uploading process will generate new blocks

Table 1. BFSOut file upload and download time consumption (sec)

Method	File Upload				File Download			
	100MB	200MB	500MB	1000MB	100MB	200MB	500MB	1000MB
Read/Write	0.063	0.212	0.222	0.839	0.002	0.148	0.325	0.558
File Enc/Dec	0.189	0.365	0.912	1.804	0.161	0.326	0.782	1.799
Keys Enc/Dec	0.046	0.085	0.167	0.316	0.039	0.072	0.153	0.307
Query/Insert	0.276	0.261	0.274	0.216	0.069	0.062	0.069	0.074
IPFS Up/Down	0.894	1.617	3.993	7.280	2.037	2.992	5.728	14.113
Sum	1.468	2.540	5.568	10.455	2.308	3.600	7.057	16.851

and increase the time consumption. In general, the download time is higher than the upload process, and the gap becomes more obvious as the file size increases.

Table 1 shows the measured time of each process during uploading, downloading, and file sharing using BFSOut. Among them, the file encryption algorithm uses AES, and the asymmetric encryption uses the ECC-512. From the table, it can be concluded that most of the time taken for file upload and download from IPFS and file encryption, the proxy re-encryption process and the asymmetric encryption process also linearly increase with the size of the file.

5 Conclusion

This paper introduced BFSOut, a secure file outsourcing system based on the blockchain Hyperledger Fabric. It was mainly to guaranteed the security and efficiency of outsourcing files to third-party untrusted servers and file sharing. On the premise of ensuring file security, BFSOut proposed a dynamic hybrid encryption scheme based on symmetric and asymmetric encryption algorithms to made the overall encryption effect more efficient in the process of file outsourcing.

In future work, this paper intended to use machine learning to model the dynamic encryption algorithm selection part to made the algorithm selection more accurate[20]. And explored the applicability of parallel encryption or file stream encryption in BFShare to further improved the encryption efficiency and resource consumption of the system.

References

1. A. Sanchez-Gomez, J. Diaz, L. Hernandez-Encinas, and D. Arroyo, Review of the main security threats and challenges in free-access public cloud storage servers, in Computer and Network Security Essentials. Springer, 2018, pp. 263–281.
2. M. Qiu, K. Gai, and Z. Xiong, Privacy-preserving wireless communications using bipartite matching in social big data, Future Generation Computer Systems, vol. 87, pp. 772–781, 2018.
3. Q. Gan, X. Wang, J. Li, J. Yan, and S. Li, Enabling online/offline remote data auditing for secure cloud storage, Cluster Computing, pp. 1–15, 2021.

4. J. Stodt, D. Schönle, C. Reich, F. Ghovanlooy Ghajar, D. Welte, and A. Sikora, Security audit of a blockchain-based industrial application platform, Algorithms, vol. 14, no. 4, p. 121, 2021.
5. K. Gai, Y. Wu, L. Zhu, Z. Zhang, and M. Qiu, Differential privacy-based blockchain for industrial internet-of-things, IEEE Transactions on Industrial Informatics, vol. 16, no. 6, pp. 4156–4165, 2019.
6. N. Z. Aitzhan and D. Svetinovic, Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams, IEEE Transactions on Dependable and Secure Computing,vol. 15, no. 5, pp. 840–852, 2016.
7. C. Sun, C. Huang, H. Zhang, B. Chen, F. An, L. Wang, and T. Yun, Individual tree crown segmentation and crown width extraction from a heightmap derived from aerial laser scanning data using a deep learning framework, Frontiers in Plant Science, vol. 13, 2022.
8. M. H. Nasir, J. Arshad, M. M. Khan, M. Fatima, K. Salah, and R. Jayaraman, Scalable blockchains—a systematic review, Future Generation Computer Systems, vol. 126, pp. 136–162, 2022.
9. K. Gai, J. Guo, L. Zhu, and S. Yu, Blockchain meets cloud computing: a survey, IEEE Communications Surveys & Tutorials, vol. 22, no. 3, pp. 2009–2030, 2020.
10. S. Ghesmati, W. Fdhila, and E. Weippl, Studying bitcoin privacy attacks and their impact on bitcoin-based identity methods, in International Conference on Business Process Management. Springer, 2021, pp. 85–101.
11. S. Aggarwal and N. Kumar, Chapter sixteen-hyperledger. Adv. Comput. vol. 121, pp. 323–343, 2021.
12. K. Sinha, A. Priya, and P. Paul, K-rsa: Secure data storage technique for multimedia in cloud data server, Journal of Intelligent & Fuzzy Systems, vol. 39, no. 3, pp. 3297–3314, 2020.
13. A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, Medrec: Using blockchain for medical data access and permission management, in 2016 2nd international conference on open and big data (OBD). IEEE, 2016, pp. 25–30.
14. Z. Li, A. V. Barenji, and G. Q. Huang, Toward a blockchain cloud manufacturing system as a peer to peer distributed network platform, Robotics and computer-integrated manufacturing, vol. 54, pp. 133–144, 2018.
15. M. A. Acquah, N. Chen, J.-S. Pan, H.-M. Yang, and B. Yan, Securing fingerprint template using blockchain and distributed storage system, Symmetry, vol. 12, no. 6, p. 951, 2020.
16. L. Li, Z. Yue, and G. Wu, Electronic medical record sharing system based on hyperledger fabric and interplanetary file system, in 2021 The 5th International Conference on Compute and Data Analysis, 2021, pp. 149–154.
17. X. Tang, H. Guo, H. Li, Y. Yuan, J. Wang, and J. Cheng, A dapp business data storage model based on blockchain and ipfs, in International Conference on Artificial Intelligence and Security. Springer, 2021, pp. 219–230.
18. J. Lai, R. H. Deng, and Y. Li, Fully secure cipertext-policy hiding cp-abe, in International conference on information security practice and experience. Springer, 2011, pp.24–39.
19. J. Shao and Z. Cao, Cca-secure proxy re-encryption without pairings, in *International Workshop on Public Key Cryptography*, pp. 357–376,Springer, 2009.
20. H. Qiu, M. Qiu, and Z. Lu, Selective encryption on ecg data in body sensor network based on supervised machine learning, Information Fusion, vol. 55, pp. 59–67, 2020.

Fabric smart contract read-after-write risk detection method based on key methods and call chains

Feixiang Ren[1] and Sujuan Qin[1]

[1] State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, 100876, China;
qsujuan@bupt.edu.cn

Abstract. Fabric is currently the most popular consortium chain platform with a modular architecture that provides high security, elasticity, flexibility and scalability. Smart contracts realize the automatic execution of transactions and the operation of reconciliation data. The Fabric platform supports general programming languages to write smart contracts. However, in the development process of smart contracts, due to insufficient understanding of the underlying operating logic of smart contracts, developers are prone to introduce some risky operations, resulting in a mismatch between the execution logic of smart contracts and business logic, resulting in a lot of losses. The read-after-write risk is a relatively complex and common security risk in smart contracts. Currently, many detection tools cannot detect this risk. There is an urgent need for a solution that can quickly and accurately detect the read-after-write risk in smart contracts. This paper proposes a static analysis smart contract read-after-write risk detection method based on key methods and call chains. The scheme extracts key method patterns on the abstract syntax tree, identifies and locates key methods with risks, greatly reduces the interference of useless nodes on detection, and realizes rapid detection. By constructing the key method call chain, the real call scene is restored according to the call type and attribute of the key method. After experimental verification, compared with the current popular smart contract risk detection tool Revive^CC, the tool proposed in this paper has higher detection accuracy and can more accurately locate the read-after-write risk in smart contracts.

Keywords: Fabric Blockchain, Smart Contract, Static Analysis, Read-after-Write Risk.

1 Introduction

Fabric is an open-source permissioned blockchain platform founded by the Linux Foundation. Since its inception, the Fabric platform has dominated enterprise-level blockchain deployments, at least 13 companies with over 1 billion dollars in revenue have built their strategic net-works on Fabric platform, far more than any other technology or platform in this type of deployment [1]. A wide range of applications in the

Internet of Things [2], supply chain [3], privacy data [4] and other fields have been deployed on the Fabric platform.

Smart contract is a digital agreement that uses algorithms and programs to formulate contract terms, which is deployed on the blockchain, and can be automatically executed according to the rules [5]. Smart contracts allow for trusted transactions without third parties that are traceable and irreversible. Smart contracts are generally used to manage digital assets, have extremely high economic value [6], and are also the most vulnerable to attack. A research report pointed out [7] that the smart contract layer is the weakest link in the blockchain system and one of the important incentives for blockchain security incidents in recent years. The most representative case is TheDao Attack in 2016 [8], where attackers used a reentrancy vulnerability [9] on smart contracts to steal 3.64 million ETH, causing a loss of about $60 million.

Fabric provides a trusted execution environment for smart contracts that can be written in general-purpose programming languages [10]. The digital assets in Fabric are stored in the state database, and are mainly updated in an appending manner through the read-write interface of the smart contract [11], so there are a large number of read-write operations in the smart contract. However, as a distributed system, Fabric does not meet read and write consistency [12], so there is a risk of read-after-write. The read-after-write risk specifically refers to the fact that the data can-not be read immediately after a data is successfully written during the operation of the smart contract, and the value observed at this time is still the value that was not updated before writing. The read-after-write risk brings severe challenges to the business on the Fabric blockchain. In the process of processing assets, the operations performed may not match the actual assets, resulting in serious errors in business logic. At present, the tools for the read-after-write risk detection of Fabric smart contracts have the shortcomings of low detection accuracy and single detection scene, and cannot cover a large number of read and write logic in actual projects. Therefore, it is urgent to find a solution that can detect the risk of write-after-write more comprehensively. This paper uses static analysis technology to implement a read-after-write risk detection method based on key methods and key method call chains, and through experimental verification, the detection method proposed in this paper makes up for the shortcomings of current tools for read-after-write risk detection.

The rest of the paper is organized as follows. Section 2 introduces the related work of smart contract security risk detection. Section 3 analyzes the execution process of the Fabric smart contract and the principle of the read-after-write risk. Section 4 presents a read-after-write risk detection scheme based on key methods and method call chains. Section 5 conducts relevant experiments on the scheme of this paper, and provides a comparison between the scheme of this paper and the Revive^CC [13] tool. Finally, Section 6 draws our conclusions.

2 Related Work

At present, the static detection work for Fabric smart contracts mainly focuses on feature matching. By analyzing the smart contract source code or intermediate code

form, fast matching is realized on the basis of a predefined feature library to detect whether there is a corresponding smart contract. security risks. In 2019, K. Yamashita et al. [14] proposed to form a signature library through static analysis of smart contracts to detect security risks in smart contracts. This scheme converts the source code of the smart contract into an abstract syntax tree, and preliminarily determines whether there is a security risk by detecting whether the package and the defined variables introduced by the smart contract are related to the defined risk. It is a coarse-grained detection scheme. P. Lv, Y. Wang, et al. [15] followed the idea of K. Yamashita et al., and refined the process of feature matching into three parts: packet detection, instruction detection, and logic detection. The package detection module detects whether there is a package that can introduce random sources through package dependency analysis. The instruction detection module mainly detects the existence of global variables and function calls that can cause security risks based on the form of intermediate code. The logic detection module detects by analyzing the function call relationship. Whether there is logic for read after write risk. Then, a signature database is formed according to the static code features of the three security risks, and the type and location of the risks are determined by matching the signature database. Since this scheme loads the entire abstract syntax tree during static analysis, the overall detection efficiency is low. Revive^CC is an open source detection tool for the security risks of Fabric smart contracts, which is widely used in the community. Revive^CC can support multiple security risk detection, and has independent detection logic for each smart contract security risk. Unfortunately, Revive^CC is designed with a single function in a smart contract as the detection target, which leads to certain limitations in the detection of smart contract security risks spread across functions.

Based on the above analysis, it can be seen that the existing smart contract security risk detection schemes and tools cannot detect the read-after-write risk well, and have the problems of low detection accuracy and single detection scenario.

3 Read-after-write Risk on Fabric

3.1 Read-after-write Risk Root Cause Analysis

The read-after-write risk refers to the fact that the data cannot be read immediately after a data is successfully written during the operation of the smart contract, and the value observed at this time is still the value that was not updated before writing.

The generation of read-after-write risk is closely related to the Fabric transaction process. In Fabric, transactions follow the execute-order-validate phases. Any operation that writes transaction data on the blockchain to the ledger database is performed after the transaction is completed and the result of the transaction is verified. To maintain data consistency, Fabric does not support read-after-write operations which means that in the same transaction of Fabric, the data updated recently cannot be read.

As shown in Fig.1, assuming that the initial state of the ledger database is {"key":"0"}. In line 6 of the code, the function reads after writing to update the value of the key to "1" by calling the write method, and then in line 8 of the code attempts

to obtain this updated value by calling the read method. However, because the two actions of reading and writing are in the execution stage of the same transaction, the update operation of the ledger database will not occur during the period, and the value of the key will become "1" only after the transaction is committed, so the return value obtained at this time is the original state. Therefore, there is a risk of inconsistency in read and write, resulting in errors in business logic.

```
func ReadYourWrite(){
    //Value of key in database befor update is 0
    key:="key"
    data:="1"

    err:=stub.PutState(key,[]byte(data))
    //through PutState method setting value of key to 1
    response,_:=GetState(key)
    //through GetState method getting value of key,cause result has not
    //been submitted,value of key is still 0
}
```

Fig. 1. Read-after-write risk code sample

3.2 Read-after-write Risk Detection Analysis

Although the existing Fabric smart contract security risk detection has achieved good results, there are still some problems. These problems mainly focus on the detection of read-after-write risks, and there are a large number of false negatives in the detection of read-after-write risks in existing solutions. Unlike other risks on Fabric smart contracts, read-after-write risks have two notable characteristics:

1. Read-after-write risk is a risk caused by both the write method and the read method. Other risks on smart contracts such as range query risk are only associated with one method.
2. Read-after-write risk has strict requirements on the execution order of the write method and the read method, that is, executing the write method first and then the read method following.

After researching the current smart contract security risk detection tools and solutions, this paper analyzes the two main factors that cause the poor performance of read-after-write risk detection:

1. Combination of multiple functions leads to read-after-write risks.
   In actual projects, it is very common to decouple the read and write operations of the database. In the process of writing business code, developers call the read and write interfaces in different functions, which makes the risk of read-after-write not easy to be detected. In Fig. 2, the function Cross calls function A and function B respectively, resulting in indirect calls to the write method and the read method, which together lead to the risk of read-after-write.
2. Special statements affect the execution order of multiple methods.
   The read-after-write risk needs to satisfy that the write method has priority over the read method in the execution order. Existing detection tools do this by comparing where the two methods are located in the code. As shown in the Fig. 3, the physical location of the write method is before the read method. However, due to

the particularity of the delayed execution keyword, the delayed execution of the code block will add a method call to the function call list. This method is always called after the function returns, which means that the write method will actually be called after the read method, so the write-after-reading risks will not arise.

```
func A(key,data){
    stub.PutState(key,data)
} //function A through PutState method set value of key
func B(key){
    stub.GetState(key)
} //function B through GetState method get value of key
func Cross(){
    key:="key"
    data:="data"
    A(key,[]byte(data))
    B(key)
} //function Cross indirectly call function A and function B
```

Fig. 2. Example of read-after-write risk caused by multi-functions

```
func DeferFunc(){
    key:="key"
    data:="data"
    defer stub,PutState(key,[]byte(data))
    //Affected by defer keyword, PutState method execute later than GetState method
    stub.GetState(key)
}
```

Fig. 3. An example of the effect of a particular statement on the read and write order

None of the existing detection schemes can well support these two scenarios, so a detection scheme is needed that can cover these two scenarios and improve the detection accuracy of read-after-write risks.

4 Detection scheme based on key method and key method call chain

Aiming at the current security risk problem of read after write in Fabric smart contracts, this paper uses the technology of static analysis to implement a detection method based on key methods and key method call chains.

4.1 Overview

The overall detection scheme consists of three parts: preprocess, identify key method, and construct key method call chain, as shown in Fig. 4. Our scheme first defines key methods, and uses pattern matching to extract features related to read-after-write risks from the abstract syntax tree form of smart contracts. Further, by summarizing the different trigger scenarios of read-after-write risk, and constructing the key method invocation chain according to the multiple attributes of the key method, to reflect the real execution sequence of the key method in the process of invoking the smart contract.

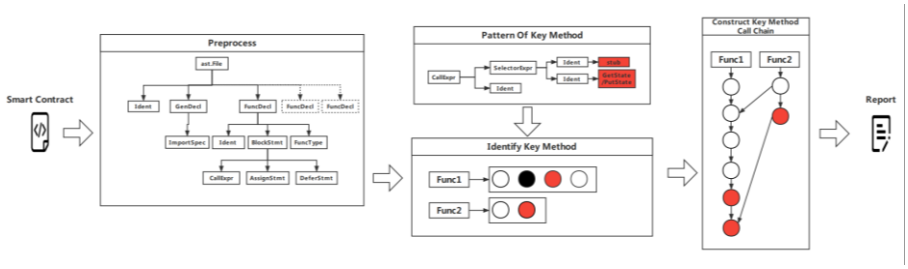


Fig. 4. Overall process of read-after-write risk detection

4.2 Preprocessing

This paper obtains rich semantic and grammatical information of smart contracts through preprocessing. Preprocessing includes two parts:

1. Convert the smart contract source code into an abstract syntax tree
   Use Golang official toolkits Scanner and Parser [16] to perform lexical and semantic analysis on smart contracts written in Golang and generate abstract syntax trees, as figure 5 shows.
2. Prune the abstract syntax tree
   Prune the converted abstract syntax tree including externally pruning nodes that are not related to analysis outside the body of functions such as GenDecl and its sub-nodes and internally pruning unrelated nodes such as Literal nodes inside the function body.

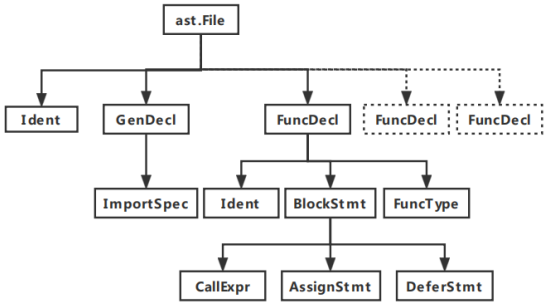


Fig. 5. Abstract syntax tree form of smart contract

4.3 Identify key methods

The key method is used to define the method invocation information related to the read-after-write risk on the Fabric smart contract. As shown in Table 1, the key methods defined in this paper mainly include the following attributes:

(1) Name represents the name of the key method
(2) Type indicates the key method whether under defer statement.
(3) Pos represents the position in the code where the key method is located.
(4) Param represents the parameters of the key method call.

(5) Function indicates whether the key method is a function call.

Table 1. Attributes of key methods

Attribute	Value Type	Content
Name	String	PutState, GetState or function name
Type	Int	1 represents ordinary method or 2 in special method
Pos	token.Pos	Indicates the location of key method in the source code
Param	[]byte{}	Parameters representing key method calls
Function	Bool	True means it is a function call, False is the opposite

The purpose of defining key methods is to describe the read-after-write risk in smart contracts more clearly. Identifying key methods is the first step in detecting read-after-write risks. After the abstract syntax tree converted from the smart contract source code is preprocessed, there are still a large number of redundant nodes. This paper uses pattern matching to quickly identify key methods in the abstract syntax tree. According to the different calling methods, the key methods can be divided into two categories:

1. Direct call

   Direct call refers to the read and write methods that are directly called in the function. For the key method of directly calling the class, there is always a three-level sub-tree structure on the abstract syntax tree as shown in Fig. 6, which is called the key method pattern in this paper. The three-layer sub-tree structure is embodied as follows: the root node is the CallExpr node, the successor node is the SelectorExpr node, and the successor nodes of the SelectorExpr node are two Ident nodes, the specific content is stub and PutState or GetState, which means PutState or GetState two read and write methods. It can be seen that the type and structure of the abstract syntax tree nodes that can represent the read and write methods are very fixed. Key methods can be quickly identified by matching such structures in an abstract syntax tree. It should be noted that when a structure of key method is found, in order to determine whether the key method exists in the special statement, it is necessary to trace back to the parent node of the CallExpr node. If the parent node of the CallExpr node is the DeferStmt node, it means that the key method exists in the special statement, and the Type value is set to 2.

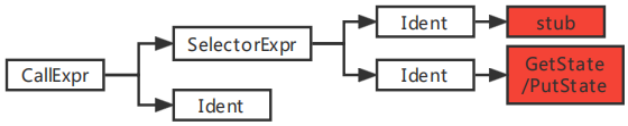


Fig. 6. Pattern of key method

2. Indirect call

   Indirect call refers to functional call. Different from the direct call, the indirect call implements the call to the read and write methods by calling the function. To identify indirect call, traverse all FuncDecl child nodes in turn starting from the root node of the abstract syntax tree, that is, traverse the first layer nodes of the

abstract syntax tree (the root node is layer 0). Extract the function name from the FuncDecl function child node structure, and maintain a global list of function names. Starting from the node declared with the FuncDecl function, traverse the subtree with the FuncDecl node as the root node. During the traversal process, if the node type meets the CallExpr call expression, extract the Ident identifier in the CallExpr node. If the Ident identifier exists in the global function name list, indicating that there is a calling relationship between functions.

4.4 Construct key method call chains

The key method call chain is a one-way linked list composed of a collection of key methods, used to represent the actual calling sequence of key methods in the smart contract.

Through the identification of key methods, all the key methods in each function body are collected, including the position attribute of the key method, but there may be problems if the position attribute is simply used to construct the call chain of the key method.

Figure 7 shows the call chain constructed using only the position value. In the function Func1, according to the context of the position, a logical sequence of read-after-write is formed. However, if the writing method exists in the defer statement, due to the characteristics of the defer statement, the delayed execution will delay the execution of the writing method, so that the sequence of read-after-write is not formed. The actual execution sequence is shown in Fig. 8.

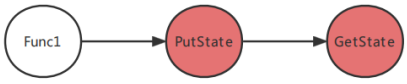


Fig. 7. Constructed by position attribute

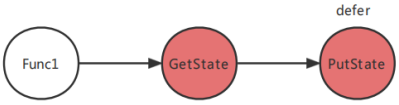


Fig. 8. Under defer statement

Another problem that needs to be solved is the impact of invocation within functions on the calling sequence of key methods. There may be a mutual calling relationship between functions. As shown in Fig. 9, the function body of Func1 executes the write method first, and then calls another function, Func2. In the Func2, the read method is executed. In this way, the logic of writing first and then reading will be formed through this indirect call, so it is necessary to consider the call between functions.

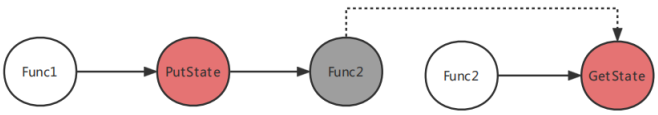


Fig. 9. Read-after-write risk caused by indirect call

Considering the execution characteristics of the defer statement, the defer statement will be executed when the function exits, that is to say, the defer statement will be later than ordinary statement. Another point worth noting is that when there are

multiple defer statements, they will be executed according to the order of execution of the stack, that is, the order of first-in-last-out, that is to say, the first defer statement defined by defer will be executed last. In order to represent the actual call sequence of key methods in the function body, the method call chain cannot be constructed simply based on the position attribute, but should be considered in combination with the position attribute, type attribute and function attribute. The process of constructing the key method call chain is as follows:

1. Scan key methods
   According to the types of key methods, the set of key methods is divided into two sub-sets. If the key method type is ordinary, add the key method to set 1; if the key method type is special, add the key method to set 2.
2. Construct sub-chain
   For the elements in set 1, according to the position attribute of the key method, the sub-chain S1 is formed from small to large. For the elements in set 2, the sub-chain S2 is formed from large to small according to the position attribute of the key method.
3. Union sub-chains
   Connect the obtained sub-chain S2 to the sub-chain S1, and get the key method call chain S0.
4. Expand call chain
   Node is expanded into the key method call chain corresponding to the function attribute until there is no key method marked as true in the key method call chain, and the key method call chain S is returned.

5 Evaluation

This section compares the proposed scheme with the current static detection scheme Revive^CC to demonstrate the advantages of the scheme.

5.1 Dataset

This experiment collects 120 smart contracts for the actual application of Fabric from the opensource platform GitHub. Among them, the read-after-write risk samples accounted for 24 samples. Specifically, the number of read-after-write risk samples in the multi-function scenario was 5, and the read-after-write risk caused by special statements accounted for 7 samples.

5.2 Comparison With Revive^CC

Revive^CC is one of the existing Fabric smart contract static analysis tools designed to detect security vulnerabilities related to blockchain to help developers write clean and secure smart contracts. This paper sets up two groups of experiments, one group adopts the detection scheme based on the key method and the key method call chain in this paper, and the other group adopts Revive^CC for control experiments. By per-

forming read-after-write risk detection on the collected 120 smart contracts, and counting the number of detected risk samples, the number of false negatives and false positives in the samples is manually screened, and finally the overall accuracy rate is calculated.

Table 2 shows the difference in read-after-write risk between the two detection schemes. The number of risk samples, false negatives, false positives, and accuracy detected by Revive^CC are 14, 5, 2, and 85.71%, respectively, while the number of risk samples, false negatives, false positives, and accuracy rates are 25, 2, 3, and 88%, respectively. The number of read-after-write risks detected by the scheme proposed in this paper is 1.8 times that of Revive^CC, and the accuracy rate is improved by 2.3%, which is significantly better than Revive^CC. Both in the special sentence scenario and multi-function scenario, the false negative rate and accuracy rate of the tool proposed in this paper are better than Revive^CC.

Table 2. Comparison of our tool and Revive^CC on read-after-write risk detection

Type	Tool	Risk samples	False negatives	False positives	Accuracy rate
Read-after-write Risk	**Revive^CC**	14	**5**	2	85.71%
	Our Tool	**25**	**2**	**3**	**88.00%**
Special Sentence	**Revive^CC**	2	**4**	1	50.00%
	Our Tool	**6**	**0**	**1**	**83.33%**
Multi-function	**Revive^CC**	0	**7**	0	0.00%
	Our Tool	**7**	**1**	**1**	**85.71%**

The accuracy and false negative rate of our scheme are better than Revive^CC. But the overall false positive rate is slightly higher than Revive^CC. The main reason is that, by observing the distribution of read-after-write risk samples, we can see that one false positive sample was introduced when detecting the read-after-write risk caused by multi-functions, resulting in the overall number of false positives changing from 2 to 3. Through research, this paper finds that the false positive samples are generated because when the risk of read after write is detected, the parameters of the read and write methods may have aliasing problems, resulting in that the read and write methods do not operate on the same object. One problem is that the alias relationship of the method parameters cannot be identified and analyzed, which leads to the misunderstanding that the behavior of read after write occurs.

6 Conclusion

In this paper, we proposed a static analysis-based detection scheme for read-after-write risk in Fabric smart contracts. By identifying key methods, it quickly located nodes which were closely linked that may cause read-after-write risk. This paper also proposed to construct a chain of key method calls to reflect the actual execution sequence between key methods, which covered more read-after-write risk scenarios. Experiments showed that the solution proposed in this paper made up for the short-

comings of the existing mainstream smart contract detection tools when detecting the risk of read-after-write and showed certain application value.

Acknowledgments. This work is supported by National Key R&D Program of China under Grant 2021YFB2700400

References

1. Web3.0 Prospective Research Report [R]TBI，2022
2. Qiu, H., Qiu, M., Memmi, G., Ming, Z., Liu, M. (2018). A Dynamic Scalable Blockchain Based Communication Architecture for IoT. In: Qiu, M. (eds) Smart Blockchain. SmartBlock 2018.
3. Li, H., Gai, K., Zhu, L., Jiang, P., Qiu, M. (2020). Reputation-Based Trustworthy Supply Chain Management Using Smart Contract. In: Qiu, M. (eds) Algorithms and Architectures for Parallel Processing. ICA3PP 2020. Lecture Notes in Computer Science(), vol 12454. Springer, Cham.
4. K. Gai, Y. Wu, L. Zhu, Z. Zhang and M. Qiu, "Differential Privacy-Based Blockchain for Industrial Internet-of-Things," in IEEE Transactions on Industrial Informatics, vol. 16, no. 6, pp. 4156-4165, June 2020, doi: 10.1109/TII.2019.2948094.
5. Shao Qifeng, Jin Cheqing, Zhang zhao, Qian Weining, Zhou Aoyin.Blockchain: Architecture and Research progess[J]. Chinese Journal of computers,2018,41(05):969-988.
6. X. Wu, H. Qiu, S. Zhang, et al "ChainIDE 2.0: Facilitating Smart Contract Development for Consortium Blockchain," IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2020, pp. 388-393
7. Blockchain security capability evaluation and analysis report [R]CAICT，2021
8. Y. Chinen, N. Yanai, J. P. Cruz and S. Okamura, "RA: Hunting for Re-Entrancy Attacks in Ethereum Smart Contracts via Static Analysis," 2020 IEEE International Conference on Blockchain (Blockchain), 2020, pp. 327-336, doi: 10.1109/Blockchain50366.2020.00048.
9. X. Yan, S. Wang and K. Gai, "A Semantic Analysis-Based Method for Smart Contract Vulnerability," 2022 IEEE 8th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), 2022, pp. 23-28
10. Elli Androulaki, Artem Barger, Vita Bortnikov, et al. 2018. Hyperledger fabric: a distributed operating system for permissioned blockchains. In Proceedings of the Thirteenth EuroSys Conference (EuroSys '18). Association for Computing Machinery, New York, NY, USA, Article 30, 1–15. https://doi.org/10.1145/3190508.3190538
11. FabricLedger[EB/OL]https://hyperledger-fabric.readthedocs.io/en/latest/ledger/ledger.html, 2022
12. Vukolic, Marko. “Rethinking Permissioned Blockchains.” Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts (2017): n. pag.
13. Revive^CC [EB/OL] https://github.com/sivachokkapu/revive-cc,2019
14. K. Yamashita, Y. Nomura, E. Zhou, B. Pi and S. Jun, "Potential Risks of Hyperledger Fabric Smart Contracts," 2019 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE), 2019, pp. 1-10, doi: 10.1109/IWBOSE.2019.8666486.
15. P. Lv, Y. Wang, Y. Wang and Q. Zhou, "Potential Risk Detection System of Hyperledger Fabric Smart Contract based on Static Analysis," 2021 IEEE Symposium on Computers and Communications (ISCC), 2021, pp. 1-7, doi: 10.1109/ISCC53001.2021.9631249.
16. Go-parser [EB/OL] https://pkg.go.dev/go/parser,2022

A Critical-Path-Based Vulnerability Detection Method for tx.origin Dependency of Smart Contract

Hui Zhao[1], Jiacheng Tan[2], Keke Gai[3]✉, and Meikang Qiu[4]

[1] Educational Information Technology Laboratory, Henan University, Kaifeng, China
zhh@henu.edu.cn
[2] Software School, Henan University, Kaifeng, China
15515439053@163.com
[3] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China
gaikeke@bit.edu.cn
[4] Dakota State University, Madison, SD, USA
qiumeikang@ieee.org

Abstract. Smart contracts are one of the most successfully applied technologies on the blockchain, which are decentralized and immutable. Smart contracts cannot be modified once deployed. Therefore, security detection of smart contracts before deployment is essential. Some smart contracts may have tx.origin dependency vulnerabilities. In this paper, we propose a critical path vulnerability detection method for detecting tx.origin dependency vulnerabilities in smart contracts. Then, in order to solve the problem that the traditional search algorithm cannot determine the critical path, we propose a path determination method based on path priority. Our method determines the critical path in the control flow graph, which enables us to detect the vulnerabilities existing in smart contracts more quickly. The experimental results show that our method is more efficient than the existing technology and the false positive rate is lower.

Keywords: smart contract · critical path · vulnerability detection · control flow graph.

1 Introduction

Encrypted currency is widely considered one of the most disruptive technologies of the past few years [1]. With the rapid development of computer hardware [2–4], parallel and distributed computing [5–7], and networks [8–10], blockchain has emerging as a promising technique. Smart contracts are programs running on the blockchain [11, 12] that store code and state on the ledger and can send and receive virtual currency [13]. Smart contracts can be viewed as code-based contracts that execute transactions without third-party supervision. These transaction records are trustworthy and traceable, but contract transactions are irreversible once completed [14]. The node user interacts with the smart contract by

calling the internal functions of the smart contract [15]. The security of smart contracts is a hot research topic today [16]. Due to the irreversible and immutable nature of smart contracts, making it critical to check their correctness before deployment [17]. Existing vulnerability detection methods include formal verification, symbolic execution, intermediate relation representation, fuzzing, and deep learning. However, most smart contract detection tools still have a high false positive rate and a long audit time. This paper proposes a method to determine the critical path through dangerous instructions to detect tx.origin dependency vulnerabilities, which can effectively improve the detection efficiency and reduce the false positive rate.
We summarize our core contributions as follows:

- We propose a method to identify critical paths by dangerous instructions to discover tx.origin dependency vulnerabilities in smart contracts.
- We propose an algorithm to judge the critical path to identify tx.origin dependency vulnerabilities in smart contracts.

The structure of this paper is organized as follows: Section 2 provides an introduction to related work. Section 3 analyzes our focus on the issues. Section 4 introduces our proposed method in detail. Section 5 gives a detailed description of the core algorithms used in our method. Section 6 conducts experiments comparing our method with other existing methods. Section 7 summarizes this paper.

2 Related work

Since 2016, due to the rapid growing of big data [18–20] and cybersecurity [21, 22] techniques, frequent attacks on smart contract vulnerabilities and resulting economic losses. The security [23, 24] of smart contracts has received extensive attention [16]. Checking based on symbolic execution. Oyente is the first symbolic execution tool to perform vulnerability analysis [25]. Muller et al. proposed a symbolic execution-based tool Mythril, which is mainly used to detect common smart contract security problems. Securify analyzes the dependency graph of smart contracts through symbolic analysis [26]. It extracts precise contract semantic information from code.

Based on fuzzy testing. ContractFuzzer is the first fuzzification framework based on Ethereum platform. Regurad is a fuzzing analysis tool mainly for smart contract reentrancy vulnerabilities [27]. sFuzz combines the strategies in AFL blurrs with adaptive strategies for branches that are hard to find [28]. Based on the intermediate method representation. Slither transforms the Solidity source code of smart contracts into an intermediate representation of SlithIR [29]. Vandal performs abstract interpretation and transformation of bytecodes in a logical manner through a decompiler. [30]. Ethir is also an analysis tool based on the EVM bytecode level [31]. SmartCheck detects smart contract vulnerabilities by transforming the Smart Contract Solidity source code into an XML intermediate representation relationship and exploiting XPath patterns .

3 Problem Definition

In this section, we describe the research problem. Analyze how the problem was identified. Describes how our critical paths are determined.

3.1 Problem Description

There is a global variable **tx.origin** in Ethereum smart contracts. It can backtrack the entire call stack to return the address of the contract that originally initiated the call. When the smart contract uses this variable for user authentication or authorization, the attacker can use the characteristics of **tx.origin** to create a corresponding attack contract to steal ether. For example, the attacker calls the withdrawal function of the victim contract in his own Fallback function, and induces the victim contract to transfer ether to the attacker contract. Undetectable exception due to authentication via **tx.origin**. Thus, all the ether in the victim's contract is transferred to the attacker's contract account.

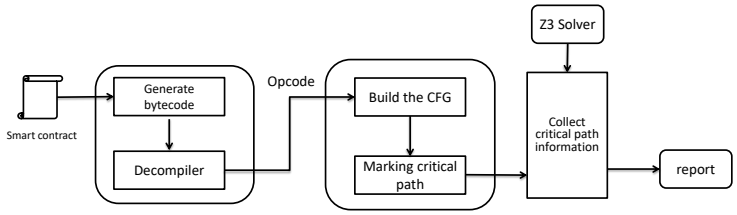


Fig. 1. GLP Smart Contract Vulnerability Check

3.2 Problem Analysis

We analyze the contract paths related to the receipt and sending of money. For example, if the contract never sends any Ether. To determine if the contract can receive ether, we check to see if the contract has a payment function. Starting with Solidity version 0.4.x, contracts are allowed to receive ether only if the public functions in the contract are declared with the keyword payable.

When the Solidity compiler compiles a non-payble function. It inserts a sequence of opcode instructions before the function body to indicate that ether cannot be accepted. Therefore, we check whether the contract allows to receive ether. We check that each path contains the above sequence of instructions.

3.3 Critical Path Determination

To allow for more effective critical path analysis, we prioritize paths based on their likelihood of revealing vulnerabilities.For each path, we compute a criticality score and determine the path based on the score. The criticality score is

calculated as follows: Let Pa be a path and V be the set of properties that Pa violates.

$$criticalpath(Pa) = \frac{\sum_{s \in V} \cdot \alpha_s}{\epsilon \cdot depth(Pa)} \quad (1)$$

where α_s is a constant representing the severity of the violation of property s. *depth(Pa)* is the depth bound of the path Pa, the number of times the function is called. ϵ is a positive constant. Intuitively, the critical score is designed such that the more critical the property violated by the path, the higher the score. And the more properties it violates, the higher the score.

4 Method

In response to the above problems, we propose a method to detect tx.origin dependency vulnerabilities based on critical paths. In this section, we introduce our method based on control flow graph critical path detection (GLP). Fig. 1 is the flow of our method GLP.

4.1 Preprocessing

Smart contracts are executed in the form of bytecode in the blockchain. However, the bytecode is obtained by compiling the source code of the smart contract. Ethereum bytecode is composed of 144 opcodes, each of which is encoded as a byte and represented in hexadecimal format. In the decompilation operation, use the solc compiler to compile the smart contract source code .sol file to obtain the EVM virtual machine bytecode of the contract. The generated bytecode is divided into deployment code, runtime bytecode and auxdata. Then decompile the runtime bytecode. Starting from the first byte of the runtime bytecode, compare each byte in the runtime bytecode in turn to obtain the instruction value corresponding to each bytecode.

4.2 Build Control Flow Graph

A control flow graph is a directed graph structure. Each of the vertices corresponds to the basic block of the program. There are no branching jump instructions in this basic block, and the basic block starts with a branch purpose and ends with a branch.

The Ethereum Virtual Machine currently supports 144 instructions. Among them, the two instructions JUMP and JUMPI represent unconditional jump and conditional jump respectively. They can generate branches of execution paths. The instructions JUMP and JUMPI will be used as the end of the basic block. They indicate a jump to the next basic block. The instruction JUMPDEST identifies the destination offset for JUMP and JUMPI jumps. And the jump destination offset of the instructions JUMP and JUMPI can only be the offset where the JUMPDEST instruction is located. Therefore, JUMPDEST needs to be the beginning of the basic block. REVERT, RETURN and INVALID in EVM

indicate termination. They are also the end of the basic block. We prioritize the basic blocks in the control flow graph according to JUMPDEST. Then perform more detailed block partitioning according to JUMP and JUMPI instructions.

4.3 Critical Path Generation

In determining the critical path, we mainly mark the paths that may have vulnerabilities, and then detect the paths that may have vulnerabilities. We do not need to instrument all paths in the generated control flow graph. Through this method, we can improve the detection efficiency and reduce the average detection time of a single contract.

When using the global variable tx.orgin for authentication and authorization operations, the smart contract is at risk of being attacked. We get the bytecode of the contract by preprocessing the contract. The control flow graph of the contract is created through bytecode to simulate the running process of the smart contract. When generating the contract control flow graph, we judge the information in the generated block in each path. We mainly focus on dangerous instructions such as contract calls and reading and modification of balances, such as CALL, CALLVALUE, CALLDATALOAD, SLOAD, SSTORE instructions, etc. We label the paths with this information and get the critical path. We label the paths with this information and get the critical path. Then we judge the instruction information of the block in the critical path, whether there is the use of ORIGIN for authentication and authorization. When it exists, the contract may be vulnerable. The z3 solver is called on the path with dangerous information to judge the reachability of the path. Finally output the detection result.

5 Algorithms

This section mainly describes the algorithms for critical path determination. As shown in Algorithm 1. The algorithm for this purpose is applied to the collection of critical path sections. The algorithm mainly introduces the process of determining the critical path. The algorithm takes the generated control flow graph as input, and then judges the block information generated in the control flow graph. When the current block contains dangerous instructions, such as CALL, CALLVALUE, CALLDATALOAD, SLOAD, SSTORE, etc. Then mark the current path, and then traverse the marked path. When the marked path has an instruction to use ORIGIN for authentication and authorization, the current path is added as a critical path. Finally output the critical path.

In Algorithm 2. The algorithm is applied to the tx.origin dependency vulnerability detection section. The algorithm mainly introduces the use of the generated critical path information to judge whether the contract has vulnerabilities. The algorithm takes the generated critical path as input. First, we traverse all the generated critical paths. And make judgments on all critical paths. If the current path contains dangerous instructions that meet the requirements, collect

Algorithm 1 Select critical path.

```
Input CFG.
Onput path.
for blocks[i] in CFG.blocks do
    if blocks[i].instruction in CFG.blocks.instruction then
        a ← blocks[i]
    end if
    if boleanjudge(a) then
        newpath ← a
    end if
end for
for newpath in CFG.blocks do
    if boleanMark(newpath) then
        pathAdd(newpath)
    end if
end for
return path
```

the current path constraints. Continue to traverse the information of the current path. Determine whether the current path contains the opcode feature of the tx.origin dependency vulnerability. If the current path contains the opcode characteristics of tx.origin dependency vulnerability. Then call the solver to solve the constraint of the current path to judge the accessibility of the current path. Finally, return the detection result.

boleanjudge(n) is A bool value. When the current command contains dangerous commands such as CALL, CALLVALUE, CALLDATALOAD, SLOAD, SSTORE, etc., the bool value is true. *boleanMark(p)* is A bool value. When the marked path basic block contains the ORIGIN directive, the bool value is true. *Collect(p)* represents a collection. When the judgment conditions are met, the constraints of the current path are collected. *boleanFinal(p)* is A bool value. Determine whether the current path contains the opcode feature of the tx.origin dependency vulnerability. *Solver(p)* is A bool value. Call the solver to judge the reachability of the current path.

6 Evaluation

This section includes the environment in which we conduct experiments, data sources, comparative experiments with other detection tools, and analysis of results. We first introduce the data sources and experimental configuration, and then evaluate the effectiveness of our method GLP in detecting tx.origin dependency vulnerabilities in real smart contracts. We conducted experiments on the accuracy of Mythril, SmartCheck, Vandal, and ContractGuard in detecting this vulnerability, and compared GLP with these four tools in terms of accuracy.

Algorithm 2 Vulnerability detection.

```
Input path.
Onput result
for CurrentPath in path do
    if current opcode is dangerous instruction. then
        Code ← current opcode
        Code.Path ← CurrentPath
    end if
end for
for opcode in code do
    if opcode is a characteristic of tx.origin . then
        Collect(Code.Path)
    end if
    if boleanFinal(Code.Path) then
        result ← Solver(CurrentPath)
    end if
end for
return result
```

6.1 Experiment Configuration

The dataset we use for evaluation is 6500 real smart contracts collected from Ethereum.We divided the collected smart contracts into three datasets. The number of contracts in the first dataset is 500, including 100 vulnerable contracts. The number of contracts in the second dataset is 1,000, of which 150 contain vulnerable contracts. The number of contracts in the third dataset is 5000, including 200 vulnerable contracts. We also added smart contracts that correctly use the global variable tx.origin to the dataset. This experiment mainly compares the three aspects of true positive, false positive and false negative of each tool. Finally, the average time to detect vulnerabilities by each tool is compared. We use these three datasets to make an empirical evaluation of GLP.We compare the performance of GLP with that of Mythril, SmartCheck, Vandal, and ContractGuard and show that GLP outperforms all systems in terms of accuracy and runtime. The experiment is mainly implemented in the Linux system.

6.2 Experiments and Results

We compared the smart contract detection accuracy of GLP and four detection tools, Mythril, SmartCheck, Vandal and ContractGuard, under the same experimental environment. We conducted three sets of experiments for verification. As shown in Fig. 2. In the figure, we use TP to represent the number of true positives, FN to represent the number of false negatives, and FP to represent the number of false positives. We use the first dataset for experiments. We observed that Mythril, SmartCheck, Vandal and ContractGuard have different degrees of false positives and false negatives, and Mythril and SmartCheck have the most false positives and false negatives. Our method outperforms all tools in terms of contract detection accuracy.

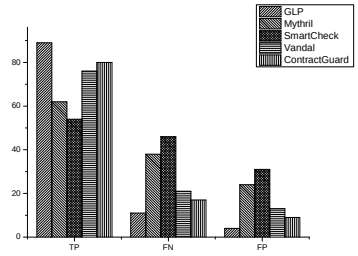


Fig. 2. The comparison of TP FN and FP on setting one.

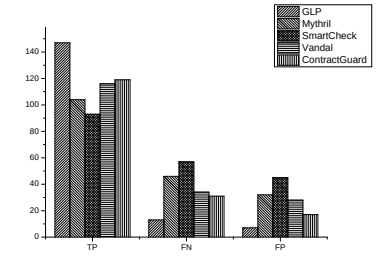


Fig. 3. The comparison of TP FN and FP on setting two.

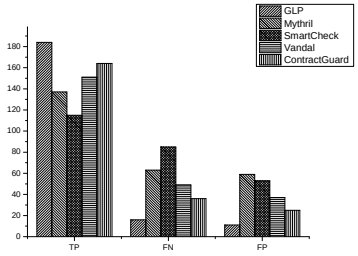


Fig. 4. The comparison of TP FN and FP on setting three.

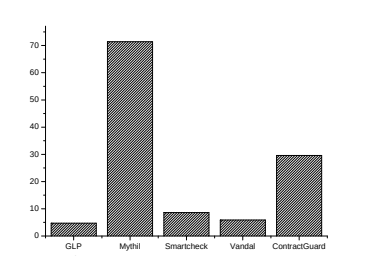


Fig. 5. Average detection time (seconds).

Fig. 3 shows the results of the second group of experiments were compared. As the number of smart contracts increases. Mythril, SmartCheck, Vandal and ContractGuard experienced a significant increase in false positives and false negatives. We see that GLP has the highest accuracy rate and lowest false positive rate compared to all other tools.Fig. 4 shows the results of the second group of experiments were compared. In the last set of experiments, we added 200 contracts with vulnerabilities to 5000 smart contracts. We see that SmartCheck detected the least number of vulnerable contracts, followed by Mythril. Out of these tools, we observed that GLP flagged more vulnerable contracts and had a small number of false positives and false negatives. As the number of smart contracts increases. Accuracy rates of Vandal and ContractGuard detections decreased, while false positive rates increased. The experimental results show that Vandal and ContractGuard are superior to Mythril and SmartCheck in detection accuracy. Experiments show that our method outperforms other tools in accuracy.

Finally, we also compared the time taken for detection. The average detection time comparison results are shown in Fig. 5. Mythril was used the longest in average detection time. The average detection time of Mythril was 71.4s. The second is ContractGuard, which has an average detection time of 29.6s. The average detection times of SmartCheck and Vandal are 8.6s and 5.9s, respectively. The average detection time of GLP is the shortest compared to other tools. The average detection time of GLP is 4.7s. In terms of total detection time, Mythril and ContractGuard are the two longest-used tools. The total detection time of Mythril was 129.1 h. The total detection time of ContractGuard is 53.5h. GLP

is the least used in total detection time compared to other tools. The total detection time of GLP is 8.4h. Comparing processing time, GLP uses the least time. Experiments show that compared with other tools, GLP can effectively reduce the false positive rate and false positive rate, and can effectively reduce the detection time.

Overall, our experimental results show that GLP is effective in discovering tx.origin dependency vulnerabilities and determining smart contract correctness. Compared with existing detection methods, GLP is overall better than existing detection methods in terms of accuracy and detection time.

7 Conclusion

We proposed a method to detect **tx.origin** dependency vulnerabilities existing in smart contracts by determining the critical path. We show examples of vulnerable contracts and how to attack them, and propose a critical path method to detect vulnerabilities. Finally, we experimented with real smart contracts on Ethereum. Compared with existing methods, our method performs better overall in detection efficiency and false positive rate, indicating that our method is both efficient and effective.

Acknowledgement

Natural Science Foundation of Shandong Province (Grant No. ZR2020ZD01).

References

1. Z. Zheng, S. Xie, *et al.*, "Blockchain challenges and opportunities," *Int'l J. of Web and Grid Services*, vol. 14, no. 4, pp. 352–375, 2018.
2. M. Qiu, H. Li, and E. Sha, "Heterogeneous real-time embedded software optimization considering hardware platform," in *ACM SAC*, pp. 1637–1641, 2009.
3. M. Qiu, Z. Jia, *et al.*, "Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP," *J. of Signal Proc. Sys.*, 2007.
4. M. Qiu, L. Yang, *et al.*, "Dynamic and leakage energy minimization with soft real-time loop scheduling and voltage assignment," *IEEE TVLSI*, vol. 18, no. 3, pp. 501–504, 2009.
5. M. Qiu, C. Xue, *et al.*, "Efficient algorithm of energy minimization for heterogeneous wireless sensor network," in *IEEE EUC*, pp. 25–34, 2006.
6. M. Qiu, C. Xue, *et al.*, "Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems," in *IEEE DATE*, pp. 1–6, 2007.
7. L. Zhang, M. Qiu, *et al.*, "Variable partitioning and scheduling for MPSoC with virtually shared scratch pad memory," *JSPS.*, vol. 58, no. 2, pp. 247–265, 2018.
8. J. Niu, Y. Gao, *et al.*, "Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability," *JPDC*, vol. 72, no. 12, pp. 1565–1575, 2012.

9. M. Qiu, Z. Chen, *et al.*, "Energy-aware data allocation with hybrid memory for mobile cloud systems," *IEEE Systems J.*, vol. 11, no. 2, pp. 813–822, 2014.
10. Y. Li, K. Gai, *et al.*, "Intercrossed access controls for secure financial services on multimedia big data in cloud systems," *ACM TMCCA*, 2016.
11. M. Qiu, H. Qiu, *et al.*, "Secure data sharing through untrusted clouds with blockchain-enabled key management," in *3rd SmartBlock conf.*, pp. 11–16, 2020.
12. K. Gai, Y. Zhang, *et al.*, "Blockchain-enabled service optimizations in supply chain digital twin," *IEEE TSC*, 2022.
13. L. W. Ouyang, S. Wang, Y. Yuan, N. I. Xiao-Chun, and F. Y. Wang, "Smart contracts:architecture and research progresses," *Acta Automatica Sinica*, 2019.
14. M. Qiu, K. Gai, and Z. Xiong, "Privacy-preserving wireless communications using bipartite matching in social big data," *FGCS*, vol. 87, pp. 772–781, 2018.
15. M. Qiu, L. Zhang, *et al.*, "Security-aware optimization for ubiquitous computing systems with seat graph approach," *Journal of Computer and System Sciences*, vol. 79, no. 5, pp. 518–529, 2013.
16. F. U. Menglin, W. U. Lifa, Z. Hong, and W. Feng, "Research on vulnerability mining technique for smart contracts," *Journal of Computer Applications*, 2019.
17. H. Qiu, M. Qiu, and Z. Lu, "Selective encryption on ECG data in body sensor network based on supervised machine learning," *Information Fusion*, vol. 55, pp. 59–67, 2020.
18. J. Li, Z. Ming, *et al.*, "Resource allocation robustness in multi-core embedded systems with inaccurate information," *J. of Systems Arch.*, vol. 57, no. 9, pp. 840–849, 2011.
19. F. Hu, S. Lakdawala, *et al.*, "Low-power, intelligent sensor hardware interface for medical data preprocessing," *IEEE Trans. on Infor. Tech. in Biomedicine*, vol. 13, no. 4, pp. 656–663, 2009.
20. H. Qiu, Q. Zheng, *et al.*, "Topological graph convolutional network-based urban traffic flow and density prediction," *IEEE Trans. on ITS*, 2020.
21. H. Qiu, T. Dong, *et al.*, "Adversarial attacks against network intrusion detection in IoT systems," *IEEE IoT J.*, vol. 8, no. 13, pp. 10327–10335, 2020.
22. K. Gai, M. Qiu, and S. Elnagdy, "A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance," in *IEEE BigDataSecurity conf.*, 2016.
23. X. Gao and M. Qiu, "Energy-based learning for preventing backdoor attack," in *KSEM (3)*, pp. 706–721, 2022.
24. M. Qiu and H. Qiu, "Review on image processing based adversarial example defenses in computer vision," in *IEEE 6th BigDataSecurity*, pp. 94–99, 2020.
25. L. Luu, D. H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *the 2016 ACM SIGSAC Conference*, 2016.
26. P. Tsankov, A. Dan, D. Drachsler-Cohen, A. Gervais, F. BXfc, nzli, and M. T. Vechev, "Securify: Practical security analysis of smart contracts," *ACM*, 2018.
27. C. Liu, H. Liu, *et al.*, "Reguard: Finding reentrancy bugs in smart contracts," in *IEEE/ACM Int'l Conf. on Software Eng.: Companion*.
28. D. N. Tai, H. P. Long, *et al.*, "sfuzz: An efficient adaptive fuzzer for solidity smart contracts," in *42nd ICSE*, 2020.
29. J. Feist, G. Greico, and A. Groce, "Slither: A static analysis framework for smart contracts," in *IEEE/ACM 2nd WETSEB*, 2019.
30. L. Brent, A. Jurisevic, M. Kong, E. Liu, and B. Scholz, "Vandal: A scalable security analysis framework for smart contracts," 2018.
31. E. Albert, P. Gordillo, B. Livshits, A. Rubio, and I. Sergey, "Ethir: A framework for high-level analysis of ethereum bytecode," 2018.

A Dynamic Taint Analysis-based Smart Contract Testing Approach

Hui Zhao[1], Xing Li[2], Keke Gai[3] ✉, and Meikang Qiu[4]

[1] Educational Information Technology Laboratory, Henan University, Kaifeng, China
zhh@henu.edu.cn
[2] Software School, Henan University, Kaifeng, China
lx@henu.edu.cn
[3] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China
gaikeke@bit.edu.cn ⋆
[4] Dakota State University, Midison, SD, USA
qiumeikang@ieee.org

Abstract. Due to the unique global state and transaction sequence characteristics of smart contracts, the detection method based on a single test case cannot improve the vulnerability detection rate during contract detection. The current contract testing methods based on genetic algorithms have not yet solved the problems caused by these characteristics. Therefore, we propose an adaptive fuzzing method based on dynamic taint analysis and genetic algorithm, SDTGfuzzer. SDTGfuzzer focuses on dynamic taint analysis to collect runtime information as feedback, and focuses on solving the challenges brought by global variables and transaction sequences for contract testing. Genetic Algorithms work well in test case generation for fuzzing. Therefore, SDTGfuzzer optimizes the genetic algorithm based on an efficient and lightweight multi-objective adaptive strategy, focusing on solving the problem that the contract constraints cannot be covered due to the global state. Experimental results show that our method has a higher vulnerability detection rate than other tools for detecting contract vulnerabilities.

Keywords: Smart Contracts · Vulnerability Detection · Fuzzing · Genetic Algorithms · Taint Analysis.

1 INTRODUCTION

With the development of computer capability [14, 16] and distributed cloud [6,13], blockchain [1,15] is emerging as very promising techniques for various applications. Fuzz testing, as a common vulnerability detection method, is widely used in program detection. Smart contracts are different from traditional *C* language and other languages in terms of operating environment and program characteristics. Therefore, when designing a fuzzing method for smart contracts, it is necessary to consider the properties specific to smart contracts.

⋆ Keke Gai is corresponding author.

There are three main challenges when fuzzing contracts. 1) Influence of transaction sequence on global variables 2) Influence of global variables on constraints. 3) The impact of transaction-related variables on constraints. Although various fuzzy testing tools have been proposed to detect contract vulnerabilities, these problems are still not well solved.

This paper mainly solves the problems brought by the above three challenges for contract testing. We propose SDTGfuzzer, a fuzzing method combining dynamic taint analysis and genetic algorithm. SDTGfuzzer uses dynamic taint analysis to obtain the pendencies of variables in the contract, and improves the efficient and lightweight multi-objective adaptive strategy based on sfuzz [11].

Our main contributions are as follows:

- For the impact of transaction sequences on contract testing, we divide transaction sequence generation into two steps, and use feedback to generate transaction sequences that are easier to find potential vulnerabilities.
- For the challenges brought by global variables for fuzzing, we modify the genetic algorithm based on an efficient and lightweight multi-objective adaptive strategy. It can better solve the problem of contract testing under the condition that the value of global variables is uncertain.
- We propose a smart contract dynamic taint analysis and genetic algorithm fuzzing method, SDTGfuzzer. The model centers on dynamic taint analysis, captures runtime feedback, and optimizes for features such as smart contract special transaction variables.

The remainder of the paper is organized as follows: We introduced the relevant work in section 2 . We introduces the system architecture of SDTGfuzzer in Section 3. Section 4 Algorithms for Methods In section 5, we conduct experiments to compare our method with other methods. Concludes is in Section 6.

2 RELATED WORK

Various tools are proposed to detect contract vulnerabilities. teEther [5], SGUARD [10] , Sereum [17] ,Osiris [18], Mythril [9] use symbolic execution methods for contract testing. But the symbol execution has the problem of facing the path explosion. Although machine learning [3, 7, 12] is used for contract detection, it is itself less interpretable. Static analysis produces many false positives

Fuzzing is widely used for contract detection due to low false positives. ContractFuzzer [4] uses black-box testing to detect contracts. Reguard [8] converts specific smart contracts into the traditional language C++. Ethploit [21] constructs pollution graphs to generate target sequences. sfuzz [11] combines policies in AFL fuzzers and lightweight multi-objective adaptive policies. ADF-GA [20] uses the exception stop statement as the basis to generate test cases. In addition, Harvey [19] uses gray box testing for contract fuzzing, and ILF [2] uses a combination of deep learning and fuzz testing for vulnerability detection.

Many fuzzing methods are used for fuzzing of smart contracts, but the research on the problems caused by the transaction sequence and global variables faced by fuzzing is still insufficient. In addition, we found that although the genetic algorithm is widely used in contract fuzzing, it does not solve the problems caused by the global state and transaction sequence. Therefore, we introduce dynamic taint analysis and improve the genetic algorithm of lightweight multi-objective adaptive strategy. Use dynamic taint analysis to obtain feedback, and solve the problem of global state and transaction sequence faced by genetic algorithm applied to fuzzing through program feedback.

3 SDTGFUZZER METHOD

The main task of this section is to construct transactions. A smart contract transaction consists of four parts: FROM, TO, VALUE, and DATA. Fig. 1 shows the logic for test parameter generation. In order to solve the FROM, VALUE, DATA required for transaction generation.

3.1 Feedback stage

Since storage is stored in the form of $key - value$ pairs, it is possible to obtain read and write operations on those storage variables in the function. Dependencies between constraints and storage variables are available through constraints and storage variables. Due to the influence of the storage variable, individual transactions directly affect each other. When the global variable directly or indirectly affects the branch constraints, the coverage of the branch cannot be completely dependent on the unit test of the function, and the calling order between functions needs to be considered. Take SLOAD as source and JUMPI opcode as sink. Record the *key* value in the storage variable to SLOAD. The storage variable flowing into the branch will have an effect on the operands of the opcode, which will affect the test.

3.2 Transaction Sequence Generation

The first step in generating a transaction sequence is to look for a danger transaction function. The second step generates a sequence of transactions based on the selected function.

transaction function selection The first step in transaction sequence generation is transaction function selection. The method selects functions by taking coverage, dangerous opcodes and abnormally stopped branch statements as important influencing factors. We first combine the dangerous opcode and block coverage as the first set of parameters. The more dangerous opcodes in uncovered blocks, the greater the probability of a vulnerability. We combine the Abnormally stopped branch and branch coverage as the second set of parameters. The more Abnormally stopped branch that are not covered, the harder

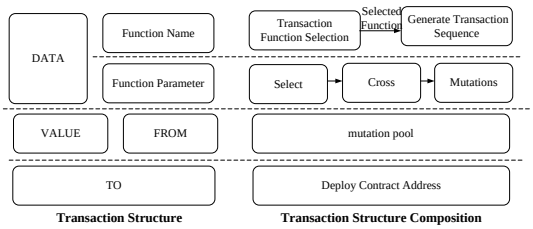


Fig. 1. Overview of invocation model parameter generation.

it is to code overwrite the contract We uses the roulette algorithm to pick the probabilities.

generate transaction sequence The second step in transaction sequence generation is generate transaction sequence. Smart transaction may modify global variables, thereby affecting the coverage of contract branches. It is for the above reasons that some contracts can only execute the logic in a specific order. The storage variable is a variable stored globally, which greatly affects the transaction sequence of the contract. To this end, it is necessary to obtain the situation of the storage variables associated with the uncovered branches, and then obtain the dependencies on the storage variables.

At this stage, the functions obtained in the Transaction Function Selection stage are first analyzed. There may be multiple storage variables in the function, which will have different effects on the Constraint branch. Different storage variables affect different Constraint branches. In order to reduce the impact of irrelevant storage variables on JUMPI constraints, and generate transaction sequences more accurately. When performing dynamic taint analysis, use SLOAD as the taint source and JUMPI as the taint sink to record the *key* of the storage variable that affects JUMPI. That is, construct < constraints, Storage variables >. Only these storage variables directly or indirectly affect the constraints of the function Constraint, and they are regarded as global variables that affect the transaction sequence. The storage variable is composed of *key* and value, and a storage variable can be located by *key*. According to the collected < Storage variables, function name>, that is, the SLOAD and SSTORE information in each function. In addition, due to the principle of assigning first and then using. the SSTORE of the *key* flowing into JUMPI will have a greater impact on the contract. Collection of functions that write to SSTORE variables, which help to generate transaction sequences.

Let the transaction sequence be $Tra = \langle fun_1 ... fun_n \rangle$. where Fun_n is the danger function chosen from transaction function selection. Analyze Fun_n to get the JUMPI constraint that SLOAD flows into. Obtain a JUMPI whose storage quantity is greater than 0 from Fun_n's uncovered JUMPI, and then obtain the set of *keys* that affect the storage variable of this branch. Search for the func-

tion that writes the key set in $< key, SLOAD, SSTORE >$. and then make the function part of the transaction sequence $\langle fun_1 ... fun_{n-1} \rangle$. At the same time, the sequences with increased coverage are stored.

3.3 Genetic Algorithm-Based Test Cases

A complete transaction call should include functions and their corresponding parameters. The second step of the invocation module faces the problem of test case generation. Efficient and lightweight multi-objective adaptive strategies in sFuzz had been shown to be very effective for generating test cases. However, the algorithm used in sFuzz does not consider the influence of the uncertainty of the value of global variables, which makes some vulnerabilities undetectable. Factors that affect branching in a contract may be parameters of functions or due to global variables. Therefore, on the basis of considering the transaction sequence, it is necessary to consider the case where branch constraints depend on global variables. For the case where the branch constraint does not depend on the global variable in the function. We take the method used by sfuzz, combined with the factor of coverage, to generate test cases for uncovered branches.

Genetic algorithm first considers generative fitness. Two kinds of fitness are adopted, one for the branch constraints that do not depend on global variables, and one for considering branch constraints that depend on global variables. The first method does not consider global variables. The fitness algorithm has better coverage for hard constraints. According to the branch constraints, the branch distance is determined by using the comparison opcode in the EVM. The smaller the branch distance, the more suitable the test case is to survive, and it is taken as the selected test case t. In this method, the branch distance is no longer obtained for the covered branch, so that the algorithm is more focused on covering the uncovered code.

Second, consider the impact of global variables, as shown in Algorithm 1. During each execution of the genetic algorithm, for the branch constraints that are not covered and have global variables flowing in, the branch distance is obtained in the same way as the first fitness. However, the test cases with small distance from the branch do not necessarily contribute to the coverage of the branch. Rather, other transactions indirectly affect the branch by affecting global variables. Therefore, for the fitness of this branch, functions with global variable dependencies need to be considered. In order to get the test cases that depend on the function, it is necessary to traverse the historical transactions. Get the position of the current test case in the whole testing process, look for the dependent function from this position forward, and record the test case of the dependent function that affects it.

After obtaining the fitness, the selection operation needs to be performed according to the fitness. There are two options. The first option: a function whose branching constraints do not depend on global variables uses branching distance as an important factor to select test cases suitable for survival. For each branch

Algorithm 1 getTestCase

Require: fun
Ensure: returnParameter, relatedFunTestCase

```
if There is an uncovered branch in fun then
  if storageDependencyNum[fun] > 0 then
    if η then
      relationFuns ← [ ]
      for key in compareStorage [comparepc] do
        relationFuns.append (writefun(key))
      end for
      totalLength ← λ
      for totalLength>0 do
        if iscfun(relationFuns) then
          relatedFunTestCase[cfun] ← (cfun related test case, fun current branch
          distance)
        end if
      end for
    end if
  else
    GeneticAlgorithm ()
    return returnParameter,relatedFunTestCase
  end if
else
  outFunctionGeneticAlgorithm()
  return returnParameter
end if
```

in the function, a test case with the smallest branch distance is selected as the fit for survival test case. The second option: For the sequence $< f_a, f_b >$, the current function f_a indirectly affects the function f_b by affecting the global variable. While using the first method to select test cases suitable for covering branches in f_a, it is also necessary to select test cases suitable for covering branches in f_b. In this method, the test case of f_a which makes the branch distance of f_b small is regarded as the test case suitable for survival.

3.4 Build Mutation Pool

In order to test the contract more quickly, we built a mutation pool and used it in the mutation phase of the genetic algorithm. We build mutation pools for hard-coded and transaction-related global variables. Hardcoding helps in some cases to generate test cases.The system uses dynamic taint analysis to obtain hard codes. Take CALLDATALOAD and CALLDATACOPY as the source to obtain the direct dependencies between transaction parameters and constraints. With comparison opcodes as tainted sinks, comparison opcodes usually exist as constraints. The operands to get the comparison opcode are used to generate hardcoding. The transaction-related global variables of the blockchain have a

greater impact on the coverage of fuzzing. The system uses dynamic taint analysis to build a special global variable mutation pool to quickly cover constraints that are dependent on special global variables.

4 ALGORITHM

In order to deal with the impact of global variables on fuzzing, the impact of global variables on branching needs to be considered in the genetic algorithm. To do this, the fitness between functions needs to be calculated, and the algorithm is shown in Algorithm 1. This algorithm is used for the second fitness generation. The algorithm mainly introduces how to generate survival-fit test cases in relationFuns that affect fun coverage. The algorithm takes *fun* that reaches the genetic algorithm condition as input, outputs the test case returned by *returnParameter*, and *relatedFunTestCase*, which is the test case suitable for survival in the correlation function. *callFunHistory* is the call status of transactions in history, and *storageDependencyNum* is the number of dependencies between global variables and branches. The read-write relationship of the storage variable in the *funStorage* is in the form of { *key*: {'SLOAD': function name, 'SSTORE': the function name}}, and the direct relationship between storage and the comparison operator in the *compareStorage* is in the form of {functionName: { comparePC:{*key*}}}. *comparePC* is the pc for comparing opcodes.

η is a bool value that is true when a branch *comparepc* is selected from the branch, and the branch contains a global variable dependency . *writefun(key)* is the name of the function in *funStorage* that performs the SSTORE operation on the *key*. *writefun(key)* is the relationship between the storage variable and the function name .λ is the subscript of the position where the test case is found from back to front in *callFunHistory*. *callFunHistory* is the historical transaction information *.iscfun(relationFuns)* is a bool value. Traverse *callFunHistory* from λ forward, when the function cfun in relationFuns is found, the bool value is true .

5 EXPERIMENT EVALUATION

In this section, we introduces the content of the experiment, and verifies the effectiveness of this method by comparing it with advanced fuzzing tools.

5.1 Experiment Configuration

We used dataset tests the accuracy of our method for a given labeled contract. For the design of the dataset, We adopted smartbugs' SB Curated dataset. and we used 4 types of vulnerabilities to test for unchecked return value exceptions, block-dependent and timestamp-dependent vulnerabilities, and Arithmetic. For contracts with inheritance relationship, we only tested the last contract. For

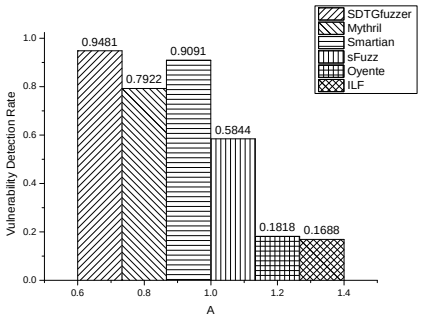


Fig. 2. Contract detection rate for dataset.

datasets marked by smartbugs, Arithmetic, Bad Randomness and Time Manipulation each have a contract that does not directly cause the vulnerability.

We writing system in python and runs on the win10 system. The system uses the solc version of 0.4.25+commit.59dbf8f1.Windows.msvc,and use Ganache as a private chain platform. The CPU is Intel(R) Core(TM) i5-9500 CPU @ 3.00GHz and the memory is DDR4, 8G. We ran each method for up to 10 minutes

5.2 Results And Analysis

In this section, We test each tool based on the number of vulnerable contracts detected for each contract.

Testing on the dataset We tested each method in the database. Compared with sFuzz, SDTGfuzzer has been greatly improved in contract Vulnerability. The results are shown in Fig. 2. Since Smartian, sFuzz, and Mythril simultaneously support vulnerabilities in all datasets, we calculate their accuracy under the dataset. Compared to smatian, we can detect 3.9% more vulnerabilities in contracts. The reason for the low detection rates of Oyente and ILF is that they do not fully support these vulnerabilities. Oyente does not support Unchecked Low Level Calls, and ILF does not support Arithmetic. Fig. 2 shows the detection rates of these methods in the smartbugs dataset.

For the test in Arithmetic. The detection result is shown in part (a) of Fig. 3. Since sFuzz does not perform vulnerability analysis for multiplication, the multiplication related vulnerabilities cannot be detected. The vulnerability detection method proposed by DTGfuzzer for Arithmetic refers to the method of Osiris, but the method of Osiris does not carry out detailed analysis on integer overflow between functions, so it cannot detect cross-contract vulnerabilities. Our method can further analyze cross-contract vulnerabilities by designing taint analysis tools.

Because SDTGfuzzer adopts dynamic taint analysis, it can avoid the problem that sFuzz must use interactive contracts to cause exceptions to detect unchecked

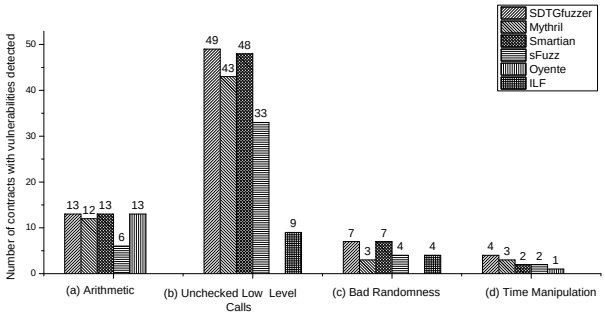


Fig. 3. Detection results of each method in dataset.

return value vulnerability. Compared with Smartian, SDTGfuzzer can still obtain a better vulnerability detection rate within a certain period of time. We found that smartian is not good at detecting contracts that require a specific address to call. SDTGfuzzer can easily find the correct address of the calling contract by using the mutation pool, and unlock the contract address on the private chain. It is possible to override contracts that need to be called from a specific address. The detection result of unchecked return value vulnerability is shown in part (b) of Fig. 3.

Both Bad Randomness and Time Manipulation are essentially dependent on the blockchain. For these dependencies, sFuzz may have false positives. Since sFuzz does not analyze the data flow, it cannot correctly judge the flow of variables, and false positives may be generated. There may be situations where an environment variable is not used by a dangerous operation, but sFuzz reports a vulnerability. Since our tool uses dynamic taint analysis, it can track the opcode and determine whether it flows into JUMPI or CALL to determine vulnerability, thereby avoiding false positives. Meanwhile, through the mutation pool, our method can quickly cover the branches constrained by environmental variables. Parts (c) and (d) of Fig. 3 are the detection situations of each tool. Compared to smartian, our method builds mutation pools for special global variables. For some constraints that require a specific address to call the contract, our method can pass these conditions smoothly.

6 CONCLUSIONS

In this work, We proposed SDTG fuzzy model. It mainly solves the problems caused by the global state and transaction sequence. The experimental results showed that the proposed method can effectively detect vulnerability in contract.

Acknowledgement

Natural Science Foundation of Shandong Province (Grant No. ZR2020ZD01).

References

1. Gai, K., Zhang, Y., et al.: Blockchain-enabled service optimizations in supply chain digital twin. IEEE TSC (2022)
2. He, J., Balunović, M., et al.: Learning to fuzz from symbolic execution with application to smart contracts. In: ACM CCS. pp. 531–548 (2019)
3. Hu, F., Lakdawala, S., et al.: Low-power, intelligent sensor hardware interface for medical data preprocessing. IEEE TITB **13**(4), 656–663 (2009)
4. Jiang, B., Liu, Y., Chan, W.: Contractfuzzer: Fuzzing smart contracts for vulnerability detection. In: 33rd IEEE/ACM Int'l Conf. ASE. pp. 259–269 (2018)
5. Krupp, J., Rossow, C.: {teEther}: Gnawing at ethereum to automatically exploit smart contracts. In: 27th USENIX Security Symposium (USENIX Security 18). pp. 1317–1333 (2018)
6. Li, Y., Gai, K., et al.: Intercrossed access controls for secure financial services on multimedia big data in cloud systems. ACM TMCCA (2016)
7. Li, Y., Song, Y., et al.: Intelligent fault diagnosis by fusing domain adversarial training and maximum mean discrepancy via ensemble learning. IEEE TII **17**(4), 2833–2841 (2020)
8. Liu, C., Liu, H., et al.: Reguard: finding reentrancy bugs in smart contracts. In: 2IEEE/ACM 40th Int'l Conf. ICSE-Companion. pp. 65–68 (2018)
9. Mueller, B.: A framework for bug hunting on the ethereum blockchain (2017)
10. Nguyen, T.D., Pham, L.H., Sun, J.: Sguard: Towards fixing vulnerable smart contracts automatically. In: IEEE Sym. on Sec. and Pri. (SP). pp. 1215–1229 (2021)
11. Nguyen, T.D., Pham, L.H., Sun, J., Lin, Y., Minh, Q.T.: sfuzz: An efficient adaptive fuzzer for solidity smart contracts. In: Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering. pp. 778–788 (2020)
12. Qiu, H., Zheng, Q., et al.: Topological graph convolutional network-based urban traffic flow and density prediction. IEEE TITS **22**(7), 4560–4569 (2020)
13. Qiu, M., Chen, Z., et al.: Energy-aware data allocation with hybrid memory for mobile cloud systems. IEEE Systems J. **11**(2), 813–822 (2014)
14. Qiu, M., Jia, Z., et al.: Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP. J. of Signal Proc. Sys. (2007)
15. Qiu, M., Qiu, H., et al.: Secure data sharing through untrusted clouds with blockchain-enabled key management. In: 3rd SmartBlock conf. pp. 11–16 (2020)
16. Qiu, M., Yang, L., et al.: Dynamic and leakage energy minimization with soft real-time loop scheduling and voltage assignment. IEEE TVLSI **18**(3), 501–504 (2009)
17. Rodler, M., Li, W., Karame, G.O., Davi, L.: Sereum: Protecting existing smart contracts against re-entrancy attacks. arXiv preprint arXiv:1812.05934 (2018)
18. Torres, C.F., Schütte, J., State, R.: Osiris: Hunting for integer bugs in ethereum smart contracts. In: Proceedings of the 34th Annual Computer Security Applications Conference. pp. 664–676 (2018)
19. Wüstholz, V., Christakis, M.: Harvey: A greybox fuzzer for smart contracts. In: 28th ACM European Software Eng. Conf. and Sym. on the Foundations of Software Eng. pp. 1398–1409 (2020)
20. Zhang, P., Yu, J., Ji, S.: Adf-ga: data flow criterion based test case generation for ethereum smart contracts. In: Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops. pp. 754–761 (2020)
21. Zhang, Q., Wang, Y., et al.: Ethploit: From fuzzing to efficient exploit generation against smart contracts. In: IEEE 27th Int'l Conf. on Software Analysis, Evolution and Reeng. (SANER). pp. 116–126 (2020)

A Survey of Weakly-supervised Semantic Segmentation

Kaiyin Zhu[1*], Neal N. Xiong[2], and Mingming Lu[1]

[1] School of Computer Science and Engineering, Central South University, ChangSha, 410083, HuNan, China zhukaiyin@csu.edu.cn, mingminglu@csu.edu.cn
[2] Department of Computer, Mathematical and Physical Sciences, Sul Ross State University, Alpine, TX 79830, USA, nxiong@cs.gsu.edu

Abstract. Weakly-supervised image semantic segmentation is a popular technology in computer vision and deep learning today. The main goal of weakly-supervised semantic segmentation is to train a model by images with only coarse or sparse annotations. Specifically, it assigns a label to each pixel through coarse label refinement or sparse label propagation, etc. The existing semantic segmentation has a wide range of applications, which includes pedestrian detection, autonomous driving, medical image segmentation, etc. However, fully-supervised semantic segmentation requires pixel-level annotation, which is expensive in manpower and time, and more and more works have focused on weakly-supervised semantic segmentation in recent years. Thus, this paper provides a review of weakly supervised semantic segmentation. Firstly, this paper summarizes the state-of-the-art research results of weakly-supervised semantic segmentation. Secondly, the widely-used datasets and semantic segmentation models are introduced. Finally, this paper analyzes the existing problems and future development directions in the field of weakly-supervised semantic segmentation.

Keywords: Semantic segmentation · Weakly-supervised · Sparse annotations.

1 Introduction

The main goal of semantic segmentation is to divide an image into multiple parts by semantic information, and each part has its own semantic label. Specifically, the pixels belonging to the same object have the same label due to assigning one label to each pixel.

Semantic segmentation has a wide range of application scenarios. On the one hand, it can be implemented for traffic field, such as pedestrian detection [1],automatic driving [2] and smart city [3], as it can provide vehicles with scene information such as streets and pedestrians through semantic segmentation. On the other hand, Semantic segmentation is also used wildly in the medical field. As

* Corresponding author: zhukaiyin@csu.edu.cn

Medical image segmentation [4], including abdominal organ segmentation, cardiac internal structure segmentation, etc., provides support for follow-up disease monitoring and other works [5]. All of these applications are based on semantic segmentation technology.

Recently, with the development of computer capability [6–8], system [9, 10], cloud computing [11–18], and big data [19–23], deep neural networks [24–27] have been applied to various applications including semantic segmentation. With its excellent deep feature mining ability and fitting ability, the fully supervised semantic segmentation models have been greatly improved. The *Fully Convolutional Networks* (FCN) proposed by Long et al. [28] has a 20% improvement compared to traditional non-deep learning methods. The U-Net [29] proposed by Ronneberger et al. provides a basic encoder-decoder framework for subsequent semantic segmentation models. The DeepLab models [30–33] mark that fully supervised semantic segmentation based on deep learning has achieved very good practical results.

The emergence of weakly-supervised semantic segmentation is to solve the relieve of the high annotation cost of fully supervised semantic segmentation. The training of fully supervised semantic segmentation models requires a large number of pixel-level annotations, which is a huge labor and time-consuming process. Despite the unsupervised semantic segmentation can relieve this problem to some extend, unsupervised semantic segmentation remains difficult to achieve good practical results due to the lack of information such as the position and edge of objects. As a compromise between these two methods, weak supervision has interested many researchers, and it also has got great breakthroughs in recent years. To better understand weakly-supervised learning, this paper classifies weakly-supervised semantic segmentation models from the perspective of annotation methods and sorts out the latest weakly-supervised segmentation models.

The remainder of this article is organized. Section 2 introduces some commonly used datasets for weakly-supervised semantic segmentation. Section 3 introduces the classic fully supervised and state-of-the-art weakly supervised semantic segmentation models in the order of full, image-level, point, scribble, and box-bounding annotations. Finally in section 4, this paper points out the existing problems in weakly-supervised segmentation and analyzes the future-research directions from the perspective of solutions.

2 Datasets for semantic segmentation

This section summarizes some common semantic segmentation datasets. Pascal VOC 2012 [34] is the most commonly used dataset for semantic segmentation currently. It is also commonly used as an evaluation criterion for models in this field. The dataset contains 20 categories with the themes of people, animals, traffic, and indoor scene. The training set contains 2913 images , which contain 6929 objects. The test set contains 1452 images.

Pascal Context [35] is an extension of the Pascal VOC 2010 dataset.It annotated the entire training set in Pascal VOC 2010 at the pixel level, which includes 540 categories of annotations and 10103 images. Semantic boundaries dataset (SBD) [36] is an extension to Pascal VOC 2011, which annotates all unlabeled images. and contains 11355 fully annotated images as training set. Due to its huge data volume, SBD is currently replacing Pascal VOC as the most commonly used dataset in the field of semantic segmentation. ScribbleSup [37] is a scribble annotation on the Pascal VOC dataset made by Lin 's team. It is widely used in scribble-based weakly-supervised semantic segmentation training.

Cityscapes [38] is a large-scale dataset containing 19 categories of road traffic scenes. It contains 3,475 finely annotated images and more than 20,000 coarsely annotated images. KITTI [39] contains pictures of traffic scenes captured by various types of sensors. Many subsequent works have annotated some of the images in this dataset. Microsoft COCO [40] is a large-scale image recognition dataset, containing more than 80 categories. 82,783 training images, 40,504 validation images, and more than 80,000 test images are provided. In the field of semantic segmentation, it is generally used to pre-train models.

3 Semantic segmentation models

This section first introduces the fully supervised semantic segmentation models. Although the focus of this paper is on weak supervision, most weakly-supervised models generate pseudo-labels through some technical means, and then use the classic fully supervised segmentation model for training. Then the state-of-the-art weakly-supervised segmentation models based on image-level, point, scribble and bounding-box annotations are summarized in order of labeling cost.

3.1 Fully supervised semantic segmentation models

The boom of fully supervised semantic segmentation in the field of deep learning started with the proposal of FCN [28] and U-Net [29] models in 2015. To cope with the limitations of convolutional networks in restoring image details, Long et al. proposed a Fully Convolutional Networks(FCN) [28] . The main idea is to convert the fully connected layer into convolutional layers, and perform upsampling through deconvolution to obtain features with the same size as the input image. This network structure has become the basic model for subsequent semantic segmentation. The U-Net model [29] proposed by Ronneberger et al. consists of a contraction path for extracting image information and an expansion path for precise positioning, which is the encoder-decoder structure used in many subsequent works.

The DeepLab series has become the most widely used semantic segmentation model due to its excellent performance. DeepLab v1 [30] is actually an FCN model. The breakthrough is that it adds a conditional random field at the end of the model to make the edge of the classification result more refined. DeepLabv2 [31] and Deeplabv3 [32] propose Atrous Spacial Pyramid Pooling(ASPP) and add several layers of atrous convolution on the basis of v1, which

Table 1. Fully supervised semantic segmentation models. The mIOU column is the results on the Pascal VOC dataset.

Method	Year	mIOU
FCN [28]	2015	62.2%
U-Net [29]	2015	-
DeepLabv1 [30]	2016	71.6%
DeepLabv2 [31]	2017	79.7%
DeepLabv3 [32]	2017	86.9%
DeepLabv3+ [33]	2018	89%
SegNeXt [41]	2022	90.6%

makes the segmentation more robust by controlling the sampling rate and the field of view of the filter. DeepLabv3+ [33] combines the ASPP module with the encoder-decoder structure. 1 compares the classic models of fully supervised semantic segmentation with the current state-of-the-art models.

3.2 Weakly-supervised semantic segmentation models

Weakly supervised semantic segmentation is divided into the following four categories in terms of labeling complexity, as shown in Figure 1.

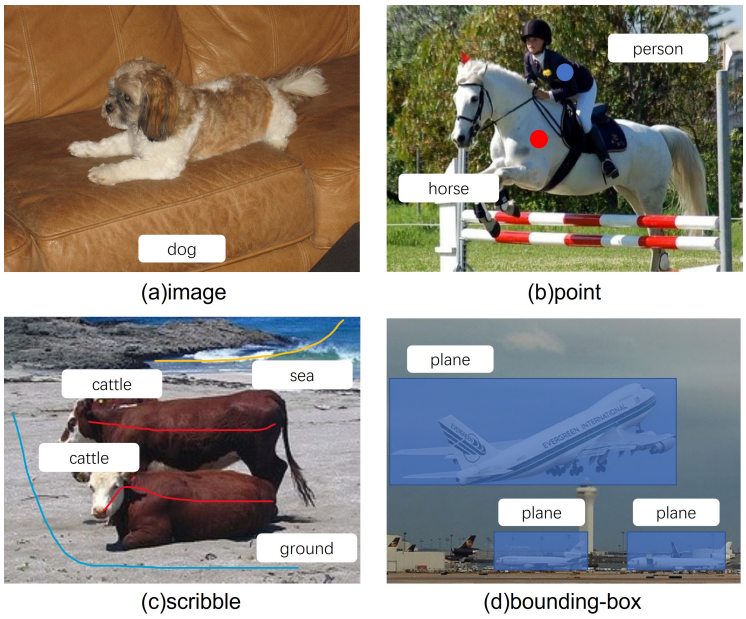


Fig. 1. Four weakly-supervised Labeling Methods.

The first type is image-level labels. According to the number of objects in the image, one or more labels should be given. It has the lowest labeling cost, but does not give the boundaries or locations of objects. The second type of point is to mark a point on the subject of each object in the picture, which is a very sparse annotation. The third type of scribble draws a scribble on the main body of each object in the image to represent the location and approximate area of the object. The fourth type of bounding-box marks a rectangular box for each object to indicate the boundary and position of the object. This labeling method has a slightly higher overhead, but it gives edge information and eliminates a lot of background interference.

This section will introduce the research results of weakly-supervised semantic segmentation in recent years from these four directions.

Segmentation algorithm based on image-level labels The mainstream method of weak supervision for image labeling is to first use the classic classification model to generate image heatmaps, class activation maps, etc., as the initial coarse labels for semantic segmentation, and then gradually refine the labels through various methods to obtain the image's location and edge information.

The *Class Activation Map* (CAM) [42] proposed by Zhou et al. provides a technical means for generating coarse labels for many subsequent works. It locates the image regions that can be used for distinction by projecting the output weights obtained by global draw pooling in the convolutional neural network onto the feature map, and obtains the class activation map. This positioning technique is also widely used in subsequent weak supervision segmentation. The IRNet proposed by Ahn,Jiwoon et al. [43] learns the center bias and boundary of objects based on CAM, and obtains instance regions according to the random walk algorithm. Yao et al. [44] introduced a self-attention mechanism to generate pseudo-labels.

Puzzle-CAM [45] proposed by Sanghyun Jo et al. improves the CAM mechanism so that it no longer only focuses on discriminative areas, but on the most integrated areas in the object, shifting the focus of CAM from classification problems to segmentation problems. The RecurSeed [46] they proposed subsequently reduces the false-detection phenomenon of small-scale objects and non-detection of large-scale objects through recursive iteration. The results of the above methods will be compared in 2.

Table 2. Image-level weakly-supervised semantic segmentation models.

Method	Year	mIOU
CAM [42]	2015	-
IRNet [43]	2019	64.8%
SGAN [44]	2019	67.2%
Puzzle-CAM [45]	2021	72.2%
RS+EPM [46]	2022	73.6%

Segmentation algorithm based on point Point annotation is inspired by the way of locating objects in human life. Humans are accustomed to pointing a point with a finger to indicate the approximate location of an object. Point-based research idea has attracted wide attention in many fields, but it has not been well studied in semantic segmentation. Amy Bearman et al. [47] used point supervision for semantic segmentation for the first time and achieved 12.9% mIOU higher than the current image-level segmentation. R. Austin McEver et al. [48] proposed a method to generate pseudo-labels using a combination of class activation maps and label propagation.

Segmentation algorithm based on scribble The mainstream method used in scribble annotations is to propagate sparse pixel annotations through technical means such as conditional random fields to generate pseudo-annotations.

Inspired by interactive commercial software, Lin et al. [37] used scribble annotation in weakly-supervised semantic segmentation for the first time. They made scribble annotation for several datasets of Pascal VOC. Tang et al. [49] proposed a normalized cut loss to optimize the training process. Ke et al. [50] used the metric learning method to shorten the distance between the same semantic pixels and increase the distance between different semantic pixels, to map each pixel of the image into the feature space containing semantics. This method can achieve better results on datasets with any annotation form.

Liang et al. [51] proposed a tree energy loss, which uses a minimum spanning tree algorithm to obtain semantic information between pixels so that single-stage training can achieve better results than previous work. Zhang et al. [53] proposed a new regularization loss and used a vision transformer as the backbone network to extract feature relationships at different levels. In the research work in the field of semantic segmentation in recent years, the transformer model has received more and more attention.

Segmentation algorithm based on bounding-box The bounding-box annotation uses the bounding-box label as the coarse label of the object to train the model, and gradually optimizes the edges.

Song et al. [56] proposed a box-driven class-wise masking model(BCM) to remove regions within the bounding box that are irrelevant to objects, and trained with an adaptive loss guided by the fill rate within the bounding box for each class. Lee et al. [57] employed an object detector to identify subjects within the bounding box, and used the results as pseudo-labels for fully supervised training. Kim et al. [58] proposed background-aware pooling to separate foreground and background information in bounding boxes, and used the noise-aware loss to remove noise in the results. Zhang et al. [55] proposed an affinity attention graph neural network, which first generates a confidence seed, calculates the semantic relationship between pixels through the affinity in the graph network, propagates the seed to unlabeled pixels, and uses the bounding-box annotation as a subsequent constraint.

The point, scribble, and bounding-box annotations will be given in 3.

Table 3. Semantic segmentation models based on point, scribble and bounding-box. In the Sup. column, P stands for Point, S for scribble, and B for bounding-box.

Method	Sup.	Year	mIOU
What's the Point [47]	P	2016	46.1%
PCAM [48]	P	2020	70.5%
SPML [50]	P	2021	73.2%
TEL [51]	P	2022	74.2%
ScribbleSup [37]	S	2016	63.1%
Normalized cut [49]	S	2018	74.5%
BPG [52]	S	2019	76.0%
SPML [50]	S	2021	76.1%
TEL [51]	S	2022	77.3%
DFR [53]	S	2021	82.9%
BCM [56]	B	2019	70.2%
Box2Seg [54]	B	2020	76.4%
SPML [50]	B	2021	74.7%
A2GNN [55]	B	2022	75.2%

4 Conclusions

This article systematically summarized the state-of-the-art weakly-supervised semantic segmentation models in recent years and introduced the commonly used datasets and classic fully-supervised models in the field of semantic segmentation. To sum up, great progress has been made in the field of weakly-supervised semantic segmentation in recent years, but there are still some problems.

First, the limitations of dataset scenarios. Due to the huge annotation cost of semantic segmentation, most of the datasets are based on mainstream research fields such as road traffic and medical imaging, lacking rich practical scenarios, and the application fields and application effects are also limited. Second, the size of the model is too large. Many research results use multi-stage training and design a large-scale deep learning model, which requires high equipment and computing power. Third, the training and inference speed is slow. It is difficult to meet the real-time requirements in practical applications.

In response to the above problems, researchers can make an effort to the following perspectives. Firstly, for the dataset scene limitations, we can utilize the domain adaptation to extend the semantic segmentation models to more application scenarios. Secondly, for the model scale and inference speed, we can consider how to improve the loss function, and take single-stage and lightweight models.

Acknowledgements This work was partially supported by the National Natural Science Foundation of China under Grant No. U20A20182 and 62177019.

References

1. Chunxue Wu, Bobo Ju, Yan Wu, et al. Uav autonomous target search based on deep reinforcement learning in complex disaster scene. *IEEE Access*, 7:117227–117245, 2019.
2. M YANC. Review on semantic segmentation of road scenes. *Laser & Optoelectronics Progress*, 58(12):36–58, 2021.
3. Prabhat Kumar, Randhir Kumar, et al. Ppsf: a privacy-preserving and secure framework using blockchain-based machine-learning for iot-driven smart cities. *IEEE Trans. on Network Science and Eng.*, 8(3):2326–2341, 2021.
4. Mohammad Hesam Hesamian, Wenjing Jia, et al. Deep learning techniques for medical image segmentation: achievements and challenges. *J. of digital imaging*, 32(4):582–596, 2019.
5. Chunxue Wu, Chong Luo, et al. A greedy deep learning method for medical disease analysis. *IEEE Access*, 6:20021–20030, 2018.
6. M. Qiu, H. Li, E. Sha. Heterogeneous real-time embedded software optimization considering hardware platform. *ACM SAC*, 1637-1641, 2009.
7. M. Qiu, C. Xue, et al. Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems. *IEEE DATE*, 1-6, 2007.
8. Shaobo Huang, Zhiwen Zeng, et al. An intelligent collaboration trust interconnections system for mobile information control in ubiquitous 5g networks. *IEEE T. on network science and eng.*, 8(1):347–365, 2020.
9. M. Qiu, L. Yang, et al. Dynamic and leakage energy minimization with soft real-time loop scheduling and voltage assignment. *IEEE TVLSI*, 18 (3), 501-504, 2009.
10. M. Qiu, Z. Jia, et al. Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP. *JSPS*, 2007.
11. J. Niu, Y. Gao, et al. Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability. *JPDC*, 72(12), 1565-1575, 2012.
12. M. Qiu, Z. Chen, et al. Energy-aware data allocation with hybrid memory for mobile cloud systems. *IEEE Systems J.*, 11 (2), 813-822, 2014
13. M. Qiu, C Xue, Z Shao, et al. Efficient algorithm of energy minimization for heterogeneous wireless sensor network. *IEEE EUC*, 25-34, 2006
14. Feng Xia, Ruonan Hao, et al. Adaptive gts allocation in ieee 802.15. 4 for real-time wireless sensor networks. *J. of Sys. Arch.*, 59(10):1231–1242, 2013.
15. Hongju Cheng, Zhe Xie, et al. Multi-step data prediction in wireless sensor networks based on one-dimensional CNN and bidirectional LSTM. *IEEE Access*, 7:117883–117896, 2019.
16. Yonglei Yao, Naixue Xiong, et al. Privacy-preserving max/min query in two-tiered wireless sensor networks. *Comp. & Math. with App.*, 65(9):1318–1325, 2013.
17. Jin Zhao, Jifeng Huang, and Naixue Xiong. An effective exponential-based trust and reputation evaluation system in wireless sensor networks. *IEEE Access*, 7:33859–33869, 2019.
18. Wei Zhang, Shiwei Zhu, et al. A novel trust management scheme based on dempster–shafer evidence theory for malicious nodes detection in wireless sensor networks. *The J. of Supercom.*, 74(4):1779–1801, 2018.
19. J. Li, Z. Ming, et al. Resource allocation robustness in multi-core embedded systems with inaccurate information. *J. of Systems Arch.* , 57 (9), 840-849, 2011
20. Y. Li, K. Gai, et al. Intercrossed access controls for secure financial services on multimedia big data in cloud systems. *ACM TMCCA* , 2016

21. K. Gai, M. Qiu, S. Elnagdy A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance. *IEEE BigDataSecurity*, 2016
22. Anmin Fu, Xianglong Zhang, et al. Vfl: a verifiable federated learning with privacy-preserving for big data in industrial iot. *IEEE TII*, 2020.
23. Yewang Chen, Lida Zhou, et al. Knn-block dbscan: Fast clustering for large-scale data. *IEEE TSMC: systems*, 51(6):3939–3953, 2019.
24. H. Qiu, T. Dong, et al. Adversarial attacks against network intrusion detection in IoT systems. *IEEE IoT J.*, 8(13), 10327-10335, 2020
25. H. Qiu, Q. Zheng, et al. Topological graph convolutional network-based urban traffic flow and density prediction. *IEEE TITS*, 2020
26. F. Hu, S. Lakdawala, et al. Low-power, intelligent sensor hardware interface for medical data preprocessing. *IEEE TITB*, 13 (4), 656-663, 2009
27. Yongbin Gao, Xuehao Xiang, et al. Human action monitoring for healthcare based on deep learning. *Ieee Access*, 6:52277–52285, 2018.
28. Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE CVPR*, pages 3431–3440, 2015.
29. Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Int'l Conf. on Medical image comp, and computer-assisted inter.*, pages 234–241. Springer, 2015.
30. Liang-Chieh Chen, George Papandreou, et al. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.
31. Liang-Chieh Chen, George Papandreou, et al. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE TPAMI*, 40(4):834–848, 2017.
32. Liang-Chieh Chen, George Papandreou, et al. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.
33. Liang-Chieh Chen, Yukun Zhu, et al. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Euro. conf. on comp. vision (ECCV)*, pages 801–818, 2018.
34. Mark Everingham and John Winn. The pascal visual object classes challenge 2012 (voc2012) development kit. *Pattern Anal. Stat. Model. Comput. Learn., Tech. Rep*, 2007:1–45, 2012.
35. Roozbeh Mottaghi, Xianjie Chen, et al. The role of context for object detection and semantic segmentation in the wild. In *IEEE CVPR*, pages 891–898, 2014.
36. Alberto Garcia-Garcia, Sergio Orts-Escolano, et al. A review on deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857*, 2017.
37. Di Lin, Jifeng Dai, Jiaya Jia, et al. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *IEEE CVPR*, pages 3159–3167, 2016.
38. Marius Cordts, Mohamed Omran, et al. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.
39. Andreas Geiger, Philip Lenz, et al. Vision meets robotics: The kitti dataset. *The Int'l J. of Robotics Research*, 32(11):1231–1237, 2013.
40. Tsung-Yi Lin, Michael Maire, et al. Microsoft coco: Common objects in context. In *Euro. conf. on comp. vision*, pages 740–755. Springer, 2014.
41. Meng-Hao Guo, Cheng-Ze Lu, et al. Segnext: Rethinking convolutional attention design for semantic segmentation. *arXiv preprint arXiv:2209.08575*, 2022.
42. Bolei Zhou, Aditya Khosla, et al. Learning deep features for discriminative localization. In *IEEE CVPR*, pages 2921–2929, 2016.

43. Jiwoon Ahn, Sunghyun Cho, and Suha Kwak. Weakly supervised learning of instance segmentation with inter-pixel relations. In *IEEE/CVF CVPR*, pages 2209–2218, 2019.
44. Qi Yao and Xiaojin Gong. Saliency guided self-attention network for weakly and semi-supervised semantic segmentation. *IEEE Access*, 8:14413–14423, 2020.
45. Sanghyun Jo and In-Jae Yu. Puzzle-cam: Improved localization via matching partial and full features. In *IEEE Conf. on Image Proc. (ICIP)*, pages 639–643. IEEE, 2021.
46. Sang Hyun Jo, In Jae Yu, and Kyung-Su Kim. Recurseed and certainmix for weakly supervised semantic segmentation. *arXiv preprint arXiv:2204.06754*, 2022.
47. Amy Bearman, Olga Russakovsky, et al. What's the point: Semantic segmentation with point supervision. In *Euro. conf. on comp. vision*, pages 549–565. Springer, 2016.
48. R Austin McEver and BS Manjunath. Pcams: Weakly supervised semantic segmentation using point supervision. *arXiv preprint arXiv:2007.05615*, 2020.
49. Meng Tang, Abdelaziz Djelouah, et al. Normalized cut loss for weakly-supervised cnn segmentation. In *IEEE CVPR*, pages 1818–1827, 2018.
50. Tsung-Wei Ke, Jyh-Jing Hwang, and Stella X Yu. Universal weakly supervised segmentation by pixel-to-segment contrastive learning. *arXiv preprint arXiv:2105.00957*, 2021.
51. Zhiyuan Liang, Tiancai Wang, et al. Tree energy loss: Towards sparsely annotated semantic segmentation. In *IEEE/CVF CVPR*, pages 16907–16916, 2022.
52. Bin Wang, Guojun Qi, Sheng Tang, et al. Boundary perception guidance: A scribble-supervised semantic segmentation approach. In *IJCAI conf. on AI*, 2019.
53. Bingfeng Zhang, Jimin Xiao, and Yao Zhao. Dynamic feature regularized loss for weakly supervised semantic segmentation. *arXiv preprint arXiv:2108.01296*, 2021.
54. Viveka Kulharia, Siddhartha Chandra, et al. Box2seg: Attention weighted loss and discriminative feature learning for weakly supervised segmentation. In *Euro. Conf. on Comp. Vision*, pages 290–308. Springer, 2020.
55. Bingfeng Zhang, Jimin Xiao, et al. Affinity attention graph neural network for weakly supervised semantic segmentation. *IEEE TPAMI*, 2021.
56. Chunfeng Song, Yan Huang, et al. Box-driven class-wise region masking and filling rate guided loss for weakly supervised semantic segmentation. In *IEEE/CVF CVPR*, pages 3136–3145, 2019.
57. Jungbeom Lee, Jihun Yi, et al. Bbam: Bounding box attribution map for weakly supervised semantic and instance segmentation. In *IEEE/CVF CVPR*, pages 2643–2652, 2021.
58. Youngmin Oh, Beomjun Kim, and Bumsub Ham. Background-aware pooling and noise-aware loss for weakly-supervised semantic segmentation. In *IEEE/CVF CVPR*, pages 6913–6922, 2021.

Construction Practice of Cloud Billing Message Based On Stream Native

Xiaoli Huang[1], Andi Liu[2], Yizhong Liu[2⋆], Li Li[1], Zhenglin Lv[1], and Fan Wang[1]

[1] China Mobile Information Technology Co., Ltd, Shenzhen, China
[2] School of Cyber Science and Technology, Beihang University, Beijing, China
huangxiaoliit@chinamobile.com, liuandi@buaa.edu.cn,
liuyizhong@buaa.edu.cn, liliit@chinamobile.com,
lvzhenglin@chinamobile.com, wangfanit@chinamobile.com

Abstract. It is necessary to accelerate digital development to make digital economy, society and government shine brightly in the future. Digital technology and the real economy will be deeply integrated, and a large number of new industries and new models will emerge. Cloud computing is an important industry to create further advantages in the digital economy. As the key to the development of enterprise business, cloud computing is the decisive factor. The ability to support billing for the entire cloud is a vital link, and it is the core capability of the whole system. It requires high accuracy and performance. Based on cloud billing, this paper analyzes the challenges cloud computing service support faces. It studies the architecture upgrade, builds the cloud service billing support capability based on stream native technology, meets the flexible billing needs of cloud services, and realizes the rapid and efficient operation support of cloud services. This system has been put into production and played a significant role strongly supporting the high-quality development needs of cloud business billing.

Keywords: Stream Native, Cloud Computing Billing, Billing Message

1 Introduction

With the requirement of digital economy, society and government, a large number of new industries and new models appear and develop quickly. Cloud computing [1, 2] has recently emerged as a buzzword in the distributed computing community and one of the scenarios of the digital outline. Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services). It can be rapidly provisioned and released with minimal management effort or service provider interaction [3].

There are five essential elements of cloud computing: on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service

⋆ Corresponding author.

[4]. Depending on the different types of cloud service, cloud community has three service models: *Software as a service* (SaaS), *Platform as a Service* (PaaS) and *Infrastructure as a Service* (IaaS). Furthermore, cloud community can also be classified as private cloud, community cloud, public cloud and hybrid cloud according to the deployment.

However, the development of cloud computing technology also has many bottlenecks [5,6]. The existing cloud computing market is highly centralized. A few technology giants dominate the market share. Relying on their highly centralized server resources, they monopolize the entire cloud computing market and enjoy high profits, leading to high computing service prices. Blockchain-based distributed cloud computing infrastructure will allow on-demand, secure, low-cost access to the most competitive computing infrastructure. Blockchain technology achieves the ledger's consistency among distributed nodes through a specific consensus [7,8]. Distributed cloud computing based on blockchains can arouse transactions between participants triggered by off-chain behaviors, such as providing data sets in real-time, transferring files, performing calculations, and providing professional services [9]. Besides, sharding blockchains [10–12] are prominent solving tools to realize scalability for cloud computing. Also, blockchain-based access control methods [13] could be adopted to keep data safe in cloud environments.

Current researches focus on accelerating the iterative upgrade of cloud operating systems, promoting technological innovations such as ultra-large-scale distributed storage [14,15], elastic computing, data virtual isolation and improving cloud security [16, 17]. They are the basis of hybrid cloud industry solutions, system integration and maintenance management. Cloud computing is an important industry to create new advantages of digital economy. By giving full play to the benefits of cloud computing's massive data, artificial intelligence and rich application scenarios, it can promote the deep integration of digital technology and the real economy, enable the transformation and upgrading of traditional industries, spawn new industries, new formats and new models, and strengthen new engines for economic development.

Our contributions. This paper builds a cloud service support system to support stream native billing capabilities. It surpasses the application of cloud computing technology and meets the diverse needs of different vertical industries. In Section 2, we introduce the structure and each part property of our cloud support system. The system charges cloud service with microservice and docker technologies. In Section 3, we list the challenges to build a cloud service billing support and build our support system in Section 4. The system can carry out multi-dimensional and differentiated billing designs for users. Finally, we summarize our system capabilities and show its combination with state-of-art blockchain technologies in Section 5.

2 Cloud Service Support System

2.1 Cloud Support Technology Architecture Based On Cloud-Native

The system adopts a "cloud-native" architecture with microservices + containerization as the core technical support. It can flexibly scale the support capability

according to the system's real-time business volume, effectively improving each business's processing efficiency and performance. Fig. 1 is the technical architecture of cloud support system.

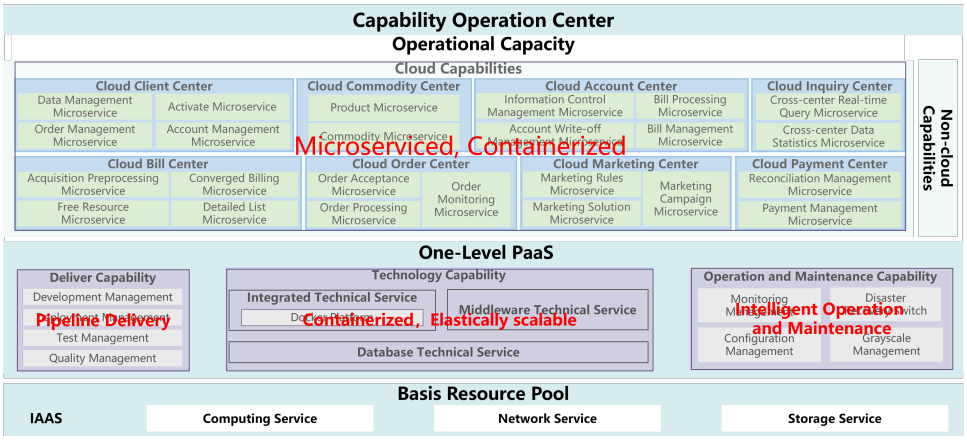


Fig. 1. Technical Architecture of Cloud Support

Cloud-Native Definition Cloud-native includes a set of applied patterns that help enterprises deliver business software quickly, continuously, reliably, and at scale. Cloud-native consists of microservice architecture, DevOps, and agile infrastructure represented by containers.
Cloud-native applications changed the application architecture, development method, deployment and maintenance technology. It achieves cloud elasticity, dynamic scheduling, automatic scaling and other capabilities that traditional IT does not have. The most significant feature of cloud-native applications is that new services can rapidly deploy.

Cloud-Native Features There are three classic cloud-native features: microserviced, containerized and DevOps.

(1) Microserviced

Microservice is a style of software architecture that composes large, complex applications in a modular fashion based on small functional blocks focused on a single responsibility and function. It follows the principle of a single operation, independent deployment of services, decoupling of functions, high scalability, simplified development testing and deployment, and friendly to development teams. Improve the overall agility and maintainability of the application through loose coupling. An application container is a lightweight runtime environment that provides applications with files, variables, and information bases required for their operation, thereby maximizing portability, supporting distributed deployment, elastic scaling, and massive business access.

(2) Containerized

Containers provide applications with an isolated running space: each container contains an entire, complete user environment space, and changes in one container will not affect the running environment of other containers. Container technology uses namespaces for space isolation, in which the container can access files through the mount point of the file system and how many resources each container can use through groups.

(3) DevOps

DevOps (the combination of Development and Operations) is a set of best practice methodologies that provides agile collaboration, code management, build engine, automated testing, automated deployment and other capabilities. Facilitates collaboration and communication among IT stakeholders (including development, operations, and testing) throughout the application and service lifecycle, resulting in:

- Continuous Integration: Easily switch from development to testing and operations.
- Continuous Deployment: Release continuously or as often as possible.
- Continuous Feedback: Seek rapid feedback at all application and service lifecycle stages.

2.2 Key Technologies and Implementation Schemes

Microservice Technology Framework and Implementation The system microservice framework follows the principle of "high cohesion and loose coupling". It divides the cloud business support services into microservices and microservices transformation. It builds a microservice governance system to improve the governance capabilities of services. A critical problem that needs to be solved is the communication and invocation problem between microservices (service discovery and service routing). The cloud support system adopts the ServiceMesh microservice framework. It is a new generation of microservice architecture currently promoted by CNCF (*Cloud-Native Computing Foundation*). ServiceMesh is a complex infrastructure layer for handling communication between services. It is essentially a service network composed of network agents . These agents are deployed next to the user's application, and the application's code is unaware of their presence. Through the sidecar of ServiceMesh, the service application itself and the microservice governance framework are decoupled entirely. Business developers can focus on the business itself. Microservice framework engineers focus on microservice orchestration and governance to solve cloud-native applications. The complexity of the microservice architecture ensures reliable and fast delivery of applications. ServiceMesh is an intelligent service network, and the sidecar is a critical component in this intelligent network — a smart router. Relying on Sidecar, ServiceMesh provides a complete set of microservice architecture solutions, including advanced functions such as service routing, load balancing, traffic control, and circuit breaker.The ServiceMesh components are more streamlined, the interaction efficiency between services is

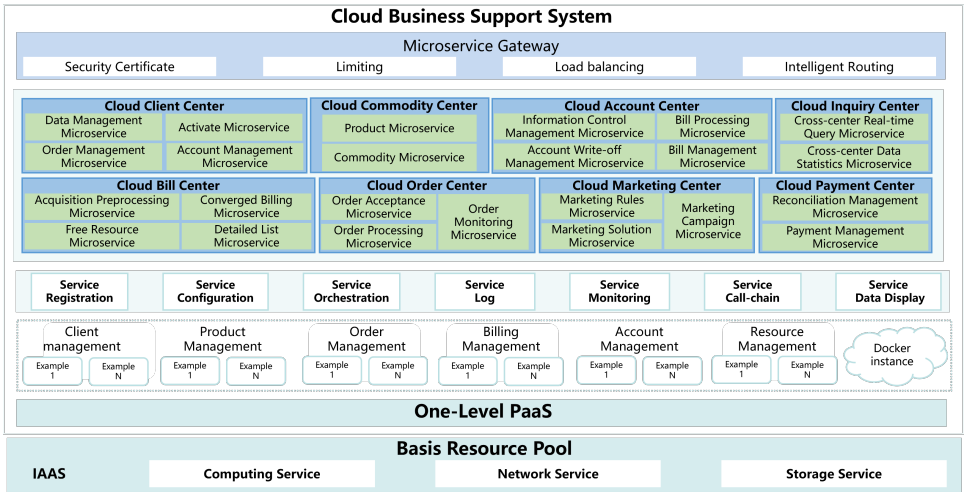


Fig. 2. Cloud Supports Microservice Business Architecture

improved, and the business response is faster. The microservice architecture for the cloud support system is shown in Fig 3. It includes application service layer, microservice PaaS layer and IaaS layer.

Application service layer First, we introduce the application service layer.

(1) CRM microservice division

Our system divided CRM into 11 microservices:

1. Account management microservice: Realize cloud customer account opening, customer data synchronization, customer manager change and other customer-related management services.
2. Order management microservice: Realize the generation of cloud business orders, order instance management, and order attribute management services.
3. Activation-type microservice: When ordering cloud basic password products, it synchronizes EC enterprise information to the cloud platform and sends basic password products to the cloud platform for activation services.
4. Marketing rule microservice: Implement management services for marketing rule elements, templates and relationships.
5. Marketing activity microservice: Realize management services for marketing activity information, marketing target groups, and marketing scenarios.
6. Order acceptance microservice: Realize order creation, order modification, order cancellation, order closing, order feedback and other services. They are the core services supporting channel business sales acceptance.
7. Order management microservice: realize the functions of order decomposition, scheduling, merging, completion, work order dispatching, feedback, and filling, and complete the execution of external or workflow according to the business, and realize the core scheduling service of sales performance.

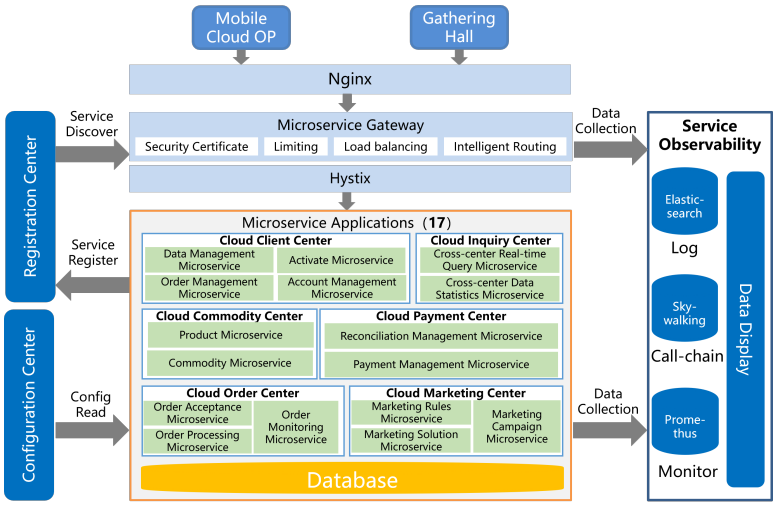


Fig. 3. Cloud Supports Microservice Technology Architecture

8. Product microservice: Realize configuration management of related attributes that constitute products, including configuration services for product creation, product catalogs, product attributes, and product changes.
9. Commodity microservice: Realize total life cycle management of commodities and commodity-related configuration services.
10. Cross-center query microservice: Realize data cross-center common query to solve the situation after the system is microserviced. Each center is independent of the other and cannot be directly related to query access.
11. Payment management microservice: When cloud Internet customers make payments, the cloud business supports the service of invoking unified payment capabilities.

(2) Microservice division of billing and accounting system

The billing and accounting system of the cloud support system is divided into 12 microservices:

1. Collection microservice: Realize the acquisition of original offline bills from the cloud platform and bill reconciliation auditing.
2. Offline gateway microservice: Parse and convert offline bills into files and convert bill records into messages.
3. Preprocessing microservice: Format offline CDR messages and convert them into messages meeting the billing format requirements.
4. Duplication microservice: Verify offline CDR messages using the defined key fields extracted and stored in the duplicate checking table.
5. Element retrieval microservice: Verify and restore CDR elements. Return the call result of the CDR processing node (normal/abnormal order and data transfer). The bureau data and customer data are obtained from the memory bank.

6. Rating microservice: Implement various checks, calculations, records, updates, and execution of event processes for the traffic and costs of received bill messages. Provide necessary data basis for downstream billing services.
7. Deduction microservice: Obtain the billing account according to the evaluation result of the bill. Get the account, account book, balance and other information according to the customer id and deduct the account book cyclically.
8. Billing processing microservice: Generate monthly bills for customers based on accumulated charges and fixed charges based on customer bills. Provide discounts and guarantees based on customer order statements.
9. Billing management microservice: Provide external billing queries, billing data statistics and other services according to the bills generated by the billing processing.
10. Account write-off management microservice: Obtain customer payment records from the centralized ERP system and match them against unwritten bills. Provide the matching results to the front page for display. According to the displayed results, the account manager updates billing record including unpaid amount, bill status, etc., and triggers the signal control startup service after confirming the write-off.
11. Credit control management microservice: Remind customers who exceed the payment cycle and operate shutdown according to the customer's credit control level. Receive the startup request of account write-off management microservice and perform the startup operation.
12. Balance management microservice: Receive customer balance recharge requests and change the balance. Receive the deduction request sent by the batching microservice and change the balance.

Microservice PaaS layer Next, we introduce the microservice PasS layer.

The service registration center is realized through the PaaS platform as well as the circuit breaker, current limit, and downgrade of services. PaaS platform also provides functions such as service deployment, port, domain name management, and automatic capacity expansion.

IaaS layer Finally, we describe the IaaS layer.

IaaS layer is provided by IT Ningbo resource pool, including physical equipment, power security, equipment access control, equipment security configuration optimization, and regular security inspections of physical equipment.

Containerization Technology Framework and Implementation The system adopts a containerized architecture based on k8s+docker, packages applications into containers, and releases them to run on the PaaS platform, realizing the transformation from traditional resource-centric to application-centric. Applications do not need to care about the process running on the platform. On which host only resource requirements need to be raised. Kubernetes is used to manage Docker containers, containers are isolated from each other, and microservice applications form clusters through Docker to provide high-availability

services. Kubernetes can manage each container effectively. It has the following characteristics.

1. Separation of application and configuration

Applications use port numbers, IPs, etc. in the service configuration center. The service configuration center provides unified storage, change, and version maintenance for all configuration files and management. In the subsequent application startup and deployment, the operation and maintenance personnel can associate the application configuration with the application to complete the deployment to avoid repackaging the image every time when the configuration is modified.

2. Application image asset construction

The basis for container operation is the image, and each application writes a docker file to generate a container image.

3. Ability Arrangement

The atomic service interfaces are combined according to the expected output capabilities, and the service orchestration layer realizes the control of the calling sequence and service status between the atomic service interfaces to ensure the integrity and consistency of the output capabilities. Service routing can realize multiple instances of the original aggregation capability. It ensures each atomic service request and parameter of the orchestration call correctly passed to the appropriate service instance and returns the result and status.

4. The application provides a health check interface

The application provides a survival and service availability check interface for the K8S to call. Once the application is unavailable, the K8S will kill the original container and restart the new container to improve the reliability of the application.

5. Application log

Stream collection method transmits the logs generated by the computing nodes to ELK or big data platform in real-time, which realizes the real-time display of business report data.

Cloud Service Charging Framework and Implementation In the model of cloud service billing 1.0, each module interacts thru files and processes offline CDR in the traditional way shown in Fig 4.

Traditional offline CDR file processing must go through multiple steps, such as

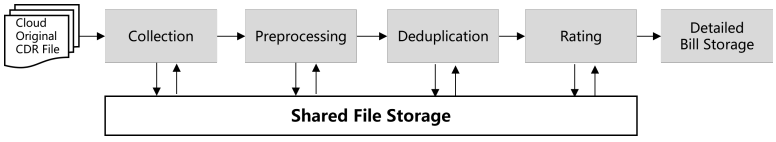


Fig. 4. Traditional Offline CDR File Processing

collection, preprocessing, deduplication and rating. It relies on a high-performance

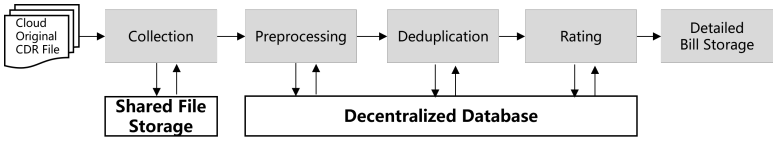


Fig. 5. Cloud-based and Distributed Billing

distributed shared file system with large demand storage and high cost. Dealing with traditional offline file processing flaws, cloud service billing 2.0 in Fig 5 meets cloud-based and distributed processing. It generally realized X86 and elastic expansion under a service-oriented, centralized capacity building, in line with the development trend of the industry. The bottom layer adopts distributed database technology to support the storage of many bills, achieving high reliability, maintainability and online capacity expansion.

3 Challenges Faced by Cloud Service Billing Support

In business development, the cloud is the key to the success of government and enterprise affairs in the cloud. The billing capability is an essential part of the cloud business and the core capability of the entire system, which requires high accuracy and performance. At the same time, the number of cloud business customers, subscriptions and bills continue to grow rapidly, placing higher requirements on billing efficiency and system resource utilization. The cloud business has thrived with various products, increasing business complexity, and complex and diverse billing methods. Cloud services have higher and higher requirements on billing scenarios and complexity and require billing systems to have higher support efficiency and more substantial support capabilities.

4 Cloud Service Billing Support

4.1 Stream Native Architecture

The new generation of stream native (Pulsar) technology architecture is based on the interactive data flow mode and adopts a flexible and unified message processing model. It realizes the message nation of the whole system process and solves the high I/O brought by the existing file processing method, which achieves high throughput and low latency CDR processing capability.

1. *Acceleration of Message Processing.* We build a distributed stream billing system based on the new stream native queue component combined with the stream processing concept. It realizes the full message bearing of the billing process and reduces the number of file landings to improve the overall execution efficiency. 2. *Consolidation and streamlining of business links.* By re-abstracting the existing billing business process, dividing the functions and merging the links, we encapsulate the business implementation with a functional and service-oriented

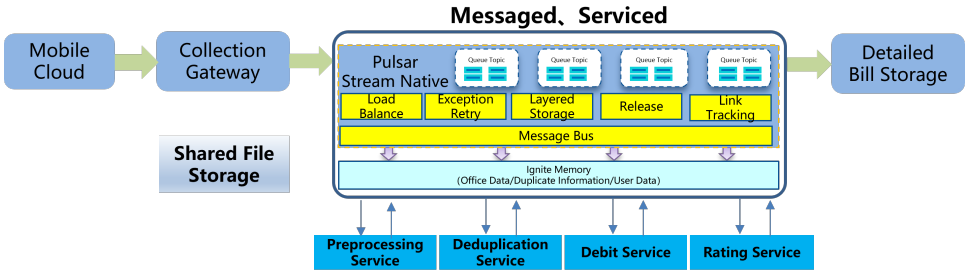


Fig. 6. Stream Native Application Framework Diagram in Billing System

architecture that supports multi-mode business carrying. 3. *Visualized process standard* By using the call chain tracking and APM components to track the link call relationship and performance in units of billing "request packets". The standardized and general RPC protocol is used between business components, which is convenient for upgrades and function expansion. 4. *Balanced and hierarchical resource storage*. The system introduces distributed message storage components to strengthen the utilization of local disk resources. With existing memory libraries, distributed file storage, and shared storage, they formed a five-layer storage architecture that decouples high-performance shared storage dependencies and reduces storage expansion costs. 5. *Center-level cluster takeover*. The cross-regional replication and synchronization of message data support the switching between message processing clusters, allowing quick taking over other centers with billing center-level exceptions. 6. *Non-interruption online business*. With the release capability of pulsar components, the billing process will not be interrupted to realize the ability of lightweight and fast support for billing when the version is upgraded. A set of billing logic can realize various supports for service, batch and real-time processing.

The billing module of the cloud support system connects to the Pulsar stream to input and output data streams, and implements functions such as message adaptation, process engine, service management, and configuration management through the service control CN node. The business carries modules such as preprocessing, duplicate checking, pricing, deduction, and auditing. The Pulsar streaming native technology implements streaming processing, load balancing, and exception retry for billing. The offline gateway microservice converts the bill file into a message and puts it in the Pulsar stream native message queue for business control acquisition and processing. The business control performs message adaptation according to the type of the CDR message, and the process execution engine calls different microservices for message processing. After the message processing is completed, the message is put into the Pulsar stream native message queue for business control calls for further processing. After the billing message is processed, the Pulsar stream natively outputs the processed message as a file for subsequent detailed bill storage and other applications. The ability to support lightweight and fast billing is realized through the applica-

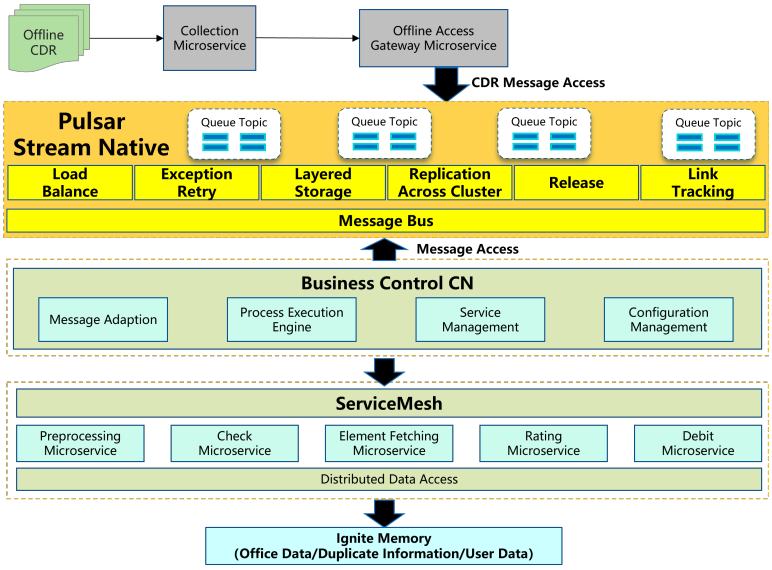


Fig. 7. Stream Native Application Implementation Diagram in Billing System

tion of the streaming native architecture in the billing system. A set of billing logic can realize multiple support modes of service, batch, and real-time processing. At the same time, the service-oriented business access capability can better support the development of vertical industries and provides various external service capabilities, including authority billing, pay-per-view, and differentiated rate billing.

4.2 Security Policy of Data Transmission Based on Stream Native Billing Capability

In the process of data transmission using pulsar, based on the security needs of the project itself, it is necessary to encrypt the transmitted data. There are usually two types of encryption algorithms: symmetric and asymmetric. In symmetric encryption, the same key is used for encryption and decryption; in asymmetric encryption, two keys are used. Generally, the public key is used for encryption and the private key is used for decryption. Symmetric encryption and decryption are relatively fast, while asymmetric encryption and decryption take a long time and the speed is relatively slow. Because stream native has extremely high real-time requirements for data processing, asymmetric encryption algorithms cannot meet the needs in terms of processing efficiency. Symmetric encryption is used here. The stream native in the EBOSS system includes a large amount of data information, such as the current user's order data, the message body itself is large, and the message length is further expanded after AES encrypts the data. The encryption takes a long time, which affects the timeliness of message delivery and increases. It increases the network load pressure of the streaming native system. To solve this problem, we construct Huffman coding trees to improve

the efficiency of data encryption and decryption. It reduces the stress of data network transmission. In the traditional Huffman algorithm, data compression is based on character statistics of the current data to be compressed and then encoded. However, in combination with stream native application scenarios, if each stream data is constructed with a separate Huffman coding tree, the cost of processing massive stream data will be very high, and the transmission efficiency will also be affected. Considering that in our business system, different single stream data have high similarity and repetition, we use pre-timed to construct a Huffman coding tree based on global stream data and load it into each stream native application node. The compression method that only encodes and decodes based on the existing code tree maximizes the data compression ratio on the premise of ensuring compression performance.

In the future, cloud business support will continue to study system implementation based on technologies such as artificial intelligence, neural networks, and blockchain systems, continuously optimizing system capabilities in terms of intelligence. With blockchain consensuses, such as PBFT [18], Hotstuff [19] and SSHC [20], different clouds can communicate as nodes in blockchain system and achieve agreement on the computing results. As one of the most promising and appealing notions in blockchain technology, the self-enforcing and event-driven features of smart contracts is also adapted to cloud computing application deployment [21]. Some digital rights management [22], cross-chain technology [23], and cloud storage [24] could be combined with cloud computing [25] to realize a better application. At the same time, we will continuously improve the professionalism and flexibility of cloud business support based on in-depth support and understanding of cloud business.

5 Conclusion

This paper systematically studied the architecture upgrade and built the cloud service billing support capability based on stream native technology. The upgraded cloud service billing capability can meet the multi-scenario billing needs of the cloud business and realize the intelligent operation support of the cloud business. The system had been put into production and had achieved remarkable results, which strongly supported the high-quality development of cloud business.

Acknowledgment

This paper is supported by the National Natural Science Foundation of China (U21B2021, 62202027, 61972018, 61932014), Yunnan Key Laboratory of Blockchain Application Technology (202105AG070005-YNB202206).

References

1. J. Zou, D. He, et al., Integrated blockchain and cloud computing systems: A systematic survey, solutions, and challenges, ACM Comput. Surv. 54 (8) (2022) 160:1–160:36.

2. S. D. Okegbile, J. Cai, A. S. Alfa, Performance analysis of blockchain-enabled data-sharing scheme in cloud-edge computing-based iot networks, IEEE IoT J. 9 (21) (2022) 21520–21536.
3. P. Mell, T. Grance, Draft nist working definition of cloud computing, Referenced on June. 3rd 15 (32) (2009) 2.
4. T. Dillon, C. Wu, E. Chang, Cloud computing: issues and challenges, in: 24th IEEE conf. on advanced info. netw. and applications, 2010, pp. 27–33.
5. M. Qiu, Z. Chen, et al., Energy-aware data allocation with hybrid memory for mobile cloud systems, IEEE Systems J. 11 (2) (2014) 813–822.
6. K. Gai, M. Qiu, S. Elnagdy, A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance, in: IEEE BigDataSecurity, 2016.
7. Y. Liu, J. Liu, Z. Zhang, T. Xu, H. Yu, Overview on consensus mechanism of blockchain technology, Journal of Cryptologic Research 6 (4) (2019) 395–432.
8. Y. Liu, J. Liu, Z. Zhang, H. Yu, A fair selection protocol for committee-based permissionless blockchains, Computers & Security (2020) 101718.
9. Y. Liu, J. Liu, Y. Hei, Y. Xia, Q. Wu, A secure cross-shard view-change protocol for sharding blockchains, in: ACISP 2021, Vol. 13083, Springer, 2021, pp. 372–390.
10. Y. Liu, J. Liu, et al., Building blocks of sharding blockchain systems: Concepts, approaches, and open problems, Computer Sci. Review 46 (2022) 100513.
11. Y. Liu, J. Liu, J. Yin, G. Li, H. Yu, Q. Wu, Cross-shard transaction processing in sharding blockchains, in: ICA3PP 2020, 2020, pp. 324–339.
12. E. Kokoris-Kogias, P. Jovanovic, et al., Omniledger: A secure, scale-out, decentralized ledger via sharding, in: IEEE SP, 2018, pp. 583–598.
13. Y. Liu, M. Qiu, J. Liu, M. Liu, Blockchain-based access control approaches, in: IEEE CSCloud, 2021, pp. 127–132.
14. J. Li, Z. Ming, et al., Resource allocation robustness in multi-core embedded systems with inaccurate information, J. of Systems Arch. 57 (9) (2011) 840–849.
15. F. Hu, S. Lakdawala, et al., Low-power, intelligent sensor hardware interface for medical data preprocessing, IEEE TITB 13 (4) (2009) 656–663.
16. Y. Li, K. Gai, et al., Intercrossed access controls for secure financial services on multimedia big data in cloud systems, ACM TMCCA (2016).
17. M. Qiu, H. Qiu, et al., Secure data sharing through untrusted clouds with blockchain-enabled key management, in: 3rd SmartBlock conf., 2020, pp. 11–16.
18. M. Castro, B. Liskov, et al., Practical byzantine fault tolerance, in: OsDI, Vol. 99, 1999, pp. 173–186.
19. M. Yin, D. Malkhi, et al., Hotstuff: Bft consensus with linearity and responsiveness, in: ACM Symp. on Principles of Distr. Computing, 2019, pp. 347–356.
20. Y. Liu, J. Liu, et al., SSHC: A secure and scalable hybrid consensus protocol for sharding blockchains with a formal security framework, IEEE Trans. Dependable Secur. Comput. 19 (3) (2022) 2070–2088.
21. B. Hu, Z. Zhang, et al., A comprehensive survey on smart contract construction and execution: paradigms, tools, and systems, Patterns 2 (2) (2021) 100179.
22. Y. Hei, J. Liu, et al., Making MA-ABE fully accountable: A blockchain-based approach for secure digital right management, Comput. Networks 191 (2021) 108029.
23. Y. Hei, D. Li, C. Zhang, J. Liu, Y. Liu, Q. Wu, Practical agentchain: A compatible cross-chain exchange system, Future Gener. Comput. Syst. 130 (2022) 207–218.
24. Y. Hei, Y. Liu, D. Li, J. Liu, Q. Wu, Themis: An accountable blockchain-based P2P cloud storage scheme, Peer-to-Peer Netw. Appl. 14 (1) (2021) 225–239.
25. C. Qiu, H. Yao, C. Jiang, S. Guo, F. Xu, Cloud computing assisted blockchain-enabled internet of things, IEEE Trans. Cloud Comput. 10 (1) (2022) 247–257.

A Fine-grained Access Control Framework for Data sharing in IoT based on IPFS and Cross-Blockchain Technology

Jiasheng Cui[1], Li Duan[1,*], Mengchen Li[1], and Wei Wang[1]

Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing, China,
{21120469, duanli*, mengchen.li, wangwei1}@bjtu.edu.cn

Abstract. *Internet of Things* (IoT) data from different trust domains is usually shared to assist in providing more services, where privacy sensitive information of shared data will be leaked or accessed without authorization. The traditional centralized access control method is difficult to adapt to the current dynamic and distributed large-scale IoT environment, and there is a risk of the single point of failure. To address these challenges, we propose a fine-grained access control framework for shared data based on cross-blockchain technology and *Interplanetary File System* (IPFS). In this framework, we firstly introduce a cross-blockchain module to realize cross-domain data sharing and solve the problem of data isolation between different data domains in IoT. Then IPFS is used to store the shared data, avoiding the risk of centralized storage. Combining symmetric encryption algorithm with ciphertext policy attribute based encryption (CP-ABE) algorithm, the fine-grained access control of shared data is guaranteed. In addition, the blockchain is applied to store the decryption key and the storage address of the original data, which records the authorization operation of access transactions and audits the access behavior of users. Experimental results show that the proposed scheme can provide higher performance compared to centralized access control methods.

Keywords: cross-blockchain · IPFS · data sharing · CP-ABE · fine-grained access control.

1 INTRODUCTION

With the development and implementation of 5G, *Internet of things* (IoT), big data, artificial intelligence and other technologies, especially with the continuous increase of IoT device [1], a large amount of data has also been generated, which needs to be stored, processed and analyzed to create value [2][3]. The traditional cloud based access control schemes [4] cannot prevent malicious cloud servers from disclosing users' data, the privacy of the stored data is damaged [5][6].

At present, the blockchain-based cloud data management schemes [7] that have been proposed still have problems such as the risk of cloud server centralization and the low storage capacity of blockchain network nodes. What is more, the data domains of the IoT [8] are not interconnected, which also results in different degrees of data isolation. Cross-blockchain technology [9] is an important

method to realize the data interaction between different blockchains, and can be used to construct the communication infrastructure between different blockchain networks composed of IoT devices in different trust domains.

At the same time, secure data sharing has high requirements for the privacy [10], confidentiality and security [11]. Therefore, fine-grained access control has become an important means to protect user data privacy and share data efficiently and securely.

Interplanetary File System (IPFS) [12] is a network transmission protocol aimed at creating persistent and distributed storage and sharing files. As a distributed storage solution, IPFS has the characteristics of low cost, high efficiency and high security. Compared with cloud server, IPFS is a distributed storage solution, which solves the single point of failure problem caused by cloud storage.

As an open, transparent, tamper proof and traceable emerging decentralized network, blockchain technology is widely used in various fields. Distributed blockchain network can well solve the defects of centralized access control scheme. The exit of any accounting node in the blockchain will not affect the stability of the whole system. The blockchain programmable smart contract [13] can realize data sharing in the state of encryption, while automatically processing user data access requests, effectively avoid the disclosure of sensitive information caused by the openness and transparency of the blockchain.

In this paper, we use the cross-blockchain technology and the IPFS to design a novel data sharing scheme in the IoT scenario. In this method, IPFS is used to realize distributed data storage, blockchain is used to realize shared data storage, cross-blockchain module is introduced to realize cross-domain data sharing. Since CP-ABE technology [14] can make specific access policies, only the data users who meet the attributes can access the data successfully, which can flexibly manage user's access rights. The main contributions of this paper are as follows:

- This scheme adopts the IPFS to realize the distributed storage of shared data, avoiding the risk of centralized storage.
- This scheme uses blockchain to realize the on-chain storage of shared data. The introduced cross-blockchain module can realize cross-blockchain data sharing, which solves the problem of data isolation between different trust domains in traditional IoT.
- This scheme uses CP-ABE technology to encrypt the key, which can flexibly specify the access authority of a single user. Only when the attribute meets the access policy can the access be successful. It solves the problem of fine-grained access control of shared data.

The rest of this paper is organized as follows. In Section 2, some related work about the existing data access control methods is presented. The proposed model, data access process and security analysis are proposed in Section 3. To demonstrate the validity of our method, experiment analysis is presented in Sections 4. The application scenario of the proposed scheme is introduced in Sections 5. Finally, the conclusion is shown in Section 6.

2 Related work

In order to achieve data secure sharing and privacy protection [15], a personal data management system based on blockchain technology [16] was proposed. This system combines blockchain and non-blockchain storage to construct a privacy-focused personal data management platform, which can better protect users' data privacy [17]. Unfortunately, this scheme still uses a centralized cloud to store data. In order to solve the confidentiality and privacy protection of data in the IoT system, a blockchain architecture system for IoT [18] was proposed, which used *Attribute Based Encryption* (ABE) technology to achieve data access control to solve the privacy and confidentiality of data shared in the blockchain IoT ecosystem. Among them, the data recorded by the sensors of the IoT needed to be transmitted to the cluster head for centralized processing and encryption, which still has the risk of centralization.

In addition, in order to solve the problem that traditional access control methods cannot support the security of private data access control process in the current IoT, an auditable and attribute-based access control system based on blockchain [19] was proposed. [20] proposed a privacy-protecting medical data sharing scheme based on blockchain, which is supplemented by proxy re-encryption and zero-knowledge proof technology. At the same time, a decentralized access control mechanism based on blockchain and ABE [21] was proposed. This scheme re-designs blockchain transaction, token encryption, token initialization and token update schemes to achieve cross-domain, fine-grained and flexible permission management.

In [22], a fine-grained access control scheme for attribute revocation in the blockchain IoT system was proposed. This scheme combines chameleon hash and ABE technology on the multi-layer blockchain scheme to realize the dynamic update of attributes. However, the key generation and distribution of this scheme heavily relies on a centralized authority. [23] proposed an intelligent usage-based insurance system premium competition scheme with privacy protection based on cross-chain. This scheme uses cross-chain technology to connect multiple blockchains to form an open multi-chain premium competition ecosystem. In order to solve the interconnection of the system and the need of data sharing among multiple organizations in cross-organizational collaboration, a blockchain-based access control scheme [24] was proposed. This method uses the consortium blockchain to establish a trusted environment. Then the *Role-Based Access Control* (RBAC) model is deployed in this environment using multi-signature protocol and smart contract approach.

3 The Proposed Scheme

3.1 System Model

In this section, we describe a data sharing scheme based on IPFS and cross-blockchain. The system model is shown in Fig.1. The model includes four entities: source blockchain, target blockchain, cross-blockchain module, and IPFS. Among them, the source blockchain contains the data owner, who owns the data can be shared. The target blockchain contains the data user, who needs to use the

shared data and can interact with the cross-blockchain module. Besides, IPFS can generate Content Identifier (CID), which is the storage address of the original data. In our scheme, assume that IPFS is not trusted, the execution process of the designed system is as follows:

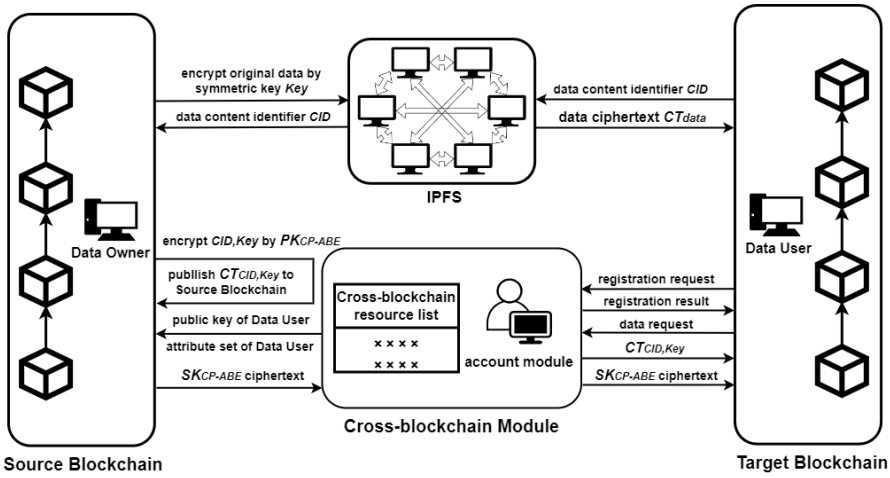


Fig. 1. The model of cross-blockchain data sharing access control.

- System initialization: The data user sends a registration request to the cross-blockchain account module. After the cross-clockchain account module verifies the registration request, it helps the user complete the registration process and return the registration result. Then, the data owner generates the system public key PK_{CP-ABE} and system master key MK_{CP-ABE}.
- Data publishing: The data owner encrypts the original data using a symmetric key Key to obtain the ciphertext CT_{data}. The data owner uploads the ciphertext CT_{data} to IPFS and receives the unique content identifier CID corresponding to the data. Then, the data owner encrypts CID and the symmetric key Key using the system public key PK_{CP-ABE} in conjunction with the access policy

$$CT_{CID,Key} = Enc(PK_{CP-ABE}, Policy, CID, Key) \tag{1}$$

  The ciphertext is associated with the access policy, and can be decrypted only if the attributes meet the access policy in the ciphertext. Then the data owner publishes the ciphertext of data $CT_{CID,Key}$ and access control policy to the source blockchain.
- Data request: The data user on the target blockchain logs in the cross-blockchain account module, checks the cross-blockchain resource list, and sends a data request to data user by calling the relevant contract. If the contract is called successfully and the attribute set of the data user meets the access policy, then the smart contract will send the ciphertext $CT_{CID,Key}$ of the CID and symmetric key to the data user, and send the public key

and attribute set of the data user to the data owner. The data owner uses the attribute set to generate the attribute private key SK_{CP-ABE} and uses public key of the data user to encrypt the attribute private key to generate ciphertext of attribute private key. The data owner sends the cpabe-key ciphertext to the cross-blockchain account of the data user through the secure channel. The data user can obtain the ciphertext of attribute private key by logging into the cross-blockchain account.

- Data acquisition: After obtaining two parts of the ciphertext, the data user uses his own private key to decrypt the ciphertext of cpabe-key and obtain the attribute private key SK_{CP-ABE}. Among them, the ciphertext of CID and symmetric key is associated with the access policy formulated by the data owner. Only when the attributes meet the access policy can the ciphertext be decrypted. Therefore, if the attributes of the data user meet the access policy, the attribute private key can be used to decrypt the ciphertext and then obtain the CID and symmetric key.

$$(CID, Key) = Dec(CT_{CID,Key}, SK_{CP-ABE}) \tag{2}$$

Finally, the data user uses CID to find and obtain the original ciphertext of original data CT_{data} on the IPFS, and then uses the symmetric key to decrypt CT_{data} and obtain the original data.

3.2 Data Access Process

In our proposed scheme, the data access process is realised by the following five algorithms, including *Setup*(), *KeyGen*(), *Encrypt*(), *Match*() and *Decrypt*(), the detailed process is as follows:

***Setup*() :** The input of the initialization algorithm is set as the secure parameter λ. Firstly, the algorithm randomly selects the bilinear group G_0 and the bilinear map $e : G_0 \times G_1 \to G_T$ whose generators are g and order p. The whole set of attributes $\Omega = \{a_1, a_2, \cdots, a_n\}$ and random elements $t_1, t_2, \cdots, t_n, t_{n+1}, t_{n+m} \in Z_p^*$ are then generated. The algorithm randomly selects parameters $\alpha_1, \beta_1 \in Z_p^*$ to generate the main private key $MK_{CP-ABE} = (\beta_1, g^{a_1})$ and the main public key $PK_{CP-ABE} = \{G_0, g, h_1 = g^{\beta_1}, e(g, g)^{\alpha_1}\}$.

***KeyGen*() :** The key generation algorithm first takes the user's attribute set U as input. The algorithm first selects a random number $r_1 \in Z_p^*$, figures out $d_1 = g^{a_1 - r_1}$, then randomly select $r_j \in Z_p^*$ for each property in attribute set U, figures out $d_j = g^{a_j - r_j}$ and generates the attribute private key $SK_{CP-ABE} = (d_1 = g^{a_1 - r_1}, \forall j \in U : d_j = g^{a_j - r_j})$.

***Encrypt*() :** The encryption algorithm is run by the data owner and consists of the following two sub-algorithms:

- *DataEncrypt().* The data owner randomly selects a string from the key space as the symmetric key Key and figures out $CT_{data} = Enc_{Key}(data)$, which indicates that *data* is encrypted by AES algorithm, and the encryption key is Key. The data owner uploads the ciphertext to IPFS and records the CID returned by IPFS.

- *KeyEncrypt().* In order to encrypt the Key and CID under the access policy T, the algorithm first selects a random number $s_1 \in Z_p$ and assigns it as the root node of tree T, then it assigns a shared secret to each non-leaf node in T in a recursive way. Given that the value of this node is s_1, if this node is an AND operation and its leaf node is not marked, mark this node and assign a random value $s_i (1 \leq s_i \leq p-1)$ to each of its child nodes. Set the last child node to $s_t = s_1 - \sum_{i=1}^{t-1} s_i \ mod\ p$. If the node is an OR operation, mark the node and set the value of all its children to s_1. For every leaf node $a_{j,i} \in T$, it calculates $c_{j,i} = g_{t_j s_i}$, where i is the serial number of the attribute in the tree. The data plaintext that needs to be encrypted is $M = (CID, Key)$, and then it caculates $c^* = Me(g,g)^{a_1 s_1}$, $c_1 = g^{s_1}$ respectively. Finally, it can output $CT_{CID,Key} = (T, c^*, c_1, \forall\ \mathrm{a}_{j,i} \in T)$ as ciphertext.

***Match*() :** The smart contract first determines whether the attributes of the user U meet the access control policy T. If the attributes do not meet the access control policy, the access fails. Otherwise, $result = CT_{CID,Key}$, the $result$ is returned to the user.

***Decrypt*() :** The decryption algorithm is run by the data user and consists of the following two sub-algorithms:

- *KeyDecrypt().* If the user attribute set U does not satisfy T, decryption fails. Otherwise, a minimum subset of U satisfying T is selected for calculation $\prod e(c_{j,i}, d_j) = \prod e(g^{t_j s_i}, g^{r t_j - 1}) = e(g,g)^{r_1 s_1}, e(c_1, d_1) \cdot e(g,g)^{r_1 s_1} = e(g^{s_1}, g^{a_1})$, and calculate the plaintext $\frac{c^*}{e(g^{s_1}, g^{a_1})} = M = (CID, Key)$.
- *DataDecrypt().* Data user downloads the ciphertext of original data CT_{data} from IPFS according to CID and figures out $data = Dec_{Key}(CT_{data})$, which indicates that CT_{data} is decrypted by a symmetric key.

In order to ensure the correctness of the model, our original data is encrypted and stored on IPFS, and the storage address of the data, namely CID, uniquely corresponds to the ciphertext of the original data. Therefore, as long as the user obtains the correct CID, he can accurately search and obtain the required ciphertext of original data on IPFS. At the same time, in order to achieve the requirement of fine-grained access control, our access control strategy adopts an access structure tree composed of many attributes. In the access structure tree, each leaf node is an attribute set by the data owner, and the attribute value is the secret value passed to the node by the parent node. Only when the attributes of the user accurately meet the access structure tree, the ciphertext of CID and the symmetric key can be obtained and decrypted, and then the ciphertext of the original data can be decrypted to obtain the original data.

3.3 Security Analysis

In this section, we will describe the security analysis of our scheme from the aspects of confidentiality and privacy.

1) Confidentiality. To ensure that no adversary can decrypt the published data, we use the symmetric encryption algorithm $DataEncrypt()$ and CP-ABE algorithm to ensure confidentiality. In the model, the data owner first encrypts

the data by a symmetric key and then uploads it to IPFS. The symmetric key is stored on the blockchain. Even if IPFS is malicious, it cannot obtain the clear text of the data. The reason is that its attributes cannot match the access policy, the ciphertext of symmetric key cannot be obtained and decrypted. In addition, the data owner uses CP-ABE algorithm $KeyEncrypt()$ to encrypt the symmetric key, and then uploads the ciphertext to the blockchain. Only data user with attribute private key can decrypt the ciphertext, which makes the confidentiality of the entire system more complete.

2) Privacy. In order to ensure that the privacy of the data is well satisfied, the data owner encrypts the data before uploading the data to IPFS, and then uploads the ciphertext. The key information is stored on the blockchain, and only the users who meet the access policy can successfully obtain it. Even if the IPFS can obtain the ciphertext data uploaded by the data owner, it cannot obtain any useful information. In order to achieve secure access control, the CP-ABE attribute private key is generated by $KeyGen()$ algorithm. Only the data owner know it. This part avoids the problem of exposing access policies and realizes the privacy protection of attributes in the access policy. In addition, the symmetric key and CID are encrypted by $KeyEncrypt()$ before being uploaded to the blockchain to ensure their privacy.

4 Experiment Analysis

In terms of experiments, we implemented a prototype to analyze the feasibility and performance of the scheme. The specific configuration of the experimental platform and environment are as follows: the operating system is Ubuntu 20.04.2 LTS, the processor is AMD Ryzen 7 5800U with Radeon Graphics, 4GB memory, and the programming languages are C++ and Solidity. The cross-blockchain module is mainly implemented through WeCross [25], which enables interoperability between heterogeneous blockchains.

Table 1. Comparison of upload speed between cloud server and IPFS

size of data (KB)	1000	5000	10000	15000	20000
the upload speed of cloud (M/s)	1.01	0.95	0.99	1.01	1.03
the upload speed of IPFS (M/s)	10.95	37.25	52.51	64.28	70.57

Table 2. Comparison of download speed between cloud server and IPFS

size of data (KB)	1000	5000	10000	15000	20000
the download speed of cloud (M/s)	31.26	35.84	26.14	30.56	35.26
the download speed of IPFS (M/s)	10.98	47.54	74.67	97.06	122.75

Firstly, we generated a group of test data on Linux to compare the upload and download speeds of IPFS and cloud servers. The experiment had five groups of tests in total. Table 1 and Table 2 show the size of test data, upload speed and download speed of the two schemes respectively. Each group of test was carried out 100 experiments on IPFS and cloud server respectively. Finally, we calculated the average value of the 100 results. The test results are shown in Fig.2. Since

IPFS uses content-based addresses instead of domain-based addresses and just needs to simply validate the hash of the content, so it can get faster transfer speeds than cloud server.

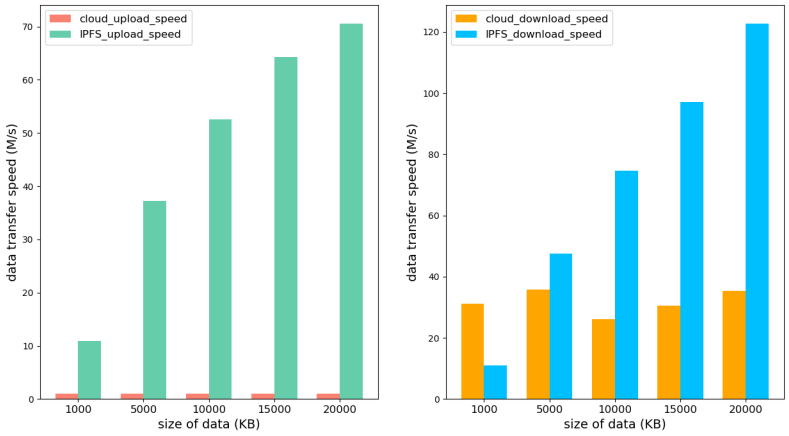


Fig. 2. (a) The comparison of upload speed between cloud and IPFS, and (b) the comparison of download speed between cloud server and IPFS.

From the experimental results, we can see that the upload speed of data on IPFS is much faster than that on cloud server. And when the data size is about 1000KB, the download speed of data on IPFS is slower than that of the cloud server. But with the data size increasing, the download speed of data on IPFS is much faster than that of the cloud server. In other words, when the data is very small, the cloud server may be faster than IPFS download speed, and when the data is larger, the IPFS download speed is more advantageous. Therefore, the data transmission speed of IPFS can well meet the needs of users.

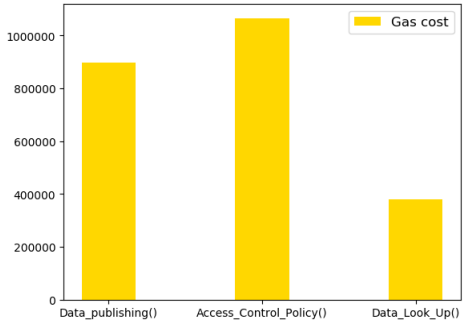


Fig. 3. The gas cost of deploying smart contracts.

In addition, we also test the gas cost of three smart contracts on Remix. The test results are shown in Fig.3. The data publishing contract is responsible for storing data that needs to be published to the blockchain. The function of the access control policy contract is to determine whether the user's attributes meet the policy. The data look-up contract is responsible for retrieving the data requested by the user.

5 Application Scenario

The application of IoT and the continuous expansion of data scale have put forward higher requirements for the sharing of IoT data. At present, the selection and combination of hardware modules in the IoT industry are very diverse, and the supporting capabilities of the blockchain platform are not the same. Once the hardware deployment is completed, it is difficult to update. What is more, a single blockchain platform will inevitably encounter bottlenecks when connecting diversified IoT devices. However, the cross-blockchain module of our scheme supports the cross-blockchain expansion of IoT devices. The blockchain connecting multiple IoT devices can be securely integrated to realize cross-platform linkage of IoT devices. For example, hospitals in different regions use different underlying blockchain systems to record medical data. For medical data sharing in different regions, our scheme uses CP-ABE access control strategy, which can provide effective privacy and security protection for medical data and realize secure sharing of patient medical data between hospitals in different regions. In addition, when the IoT devices in one region need to obtain and use the data of IoT devices in another region, the cross-blockchain data access control function of this scheme can realize the secure data sharing between IoT devices in different regions.

6 Conclusion

In this paper, we proposed a data sharing access control model based on IPFS and cross-blockchain, which realizes the secure sharing of cross-blockchain data between users in different data domains. Secondly, we designed an access control scheme based on CP-ABE, and control the access rights of data users through the permission of smart contracts. The IPFS-based data storage method reduces the storage overhead on the blockchain and avoids the risk of centralized storage. Finally, the experiment shows that the scheme can achieve the autonomous access control of data on the blockchain, and can realize the secure data sharing between heterogeneous IoT domain.

Acknowledgments

This work was supported by the National Key R&D Program of China under Grant 2020YFB1005604, the National Natural Science Foundation of China under Grant No.61902021 and No.62272031, the Beijing Natural Science Foundation under Grant No.4212008, the Open Foundation of Information Security Evaluation Center of Civil Aviation, Civil Aviation University of China under Grant No. ISECCA-202101.

References

1. Gai K, Choo K K R, et al. Privacy-preserving content-oriented wireless communication in internet-of-things[J]. IEEE IoT J., 2018, 5(4): 3059-3067.
2. M. Qiu, Z. Chen, et al., "Energy-aware data allocation with hybrid memory for mobile cloud systems", IEEE Systems J., 11 (2), 813-822, 2014

3. J. Li, Z. Ming, et al., "Resource allocation robustness in multi-core embedded systems with inaccurate information", J. of Systems Arch., 57 (9), 840-849, 2011
4. Fan K, Tian Q, Wang J, et al. Privacy protection based access control scheme in cloud-based services[J]. China Comm., 2017, 14(1): 61-71.
5. Y. Li, K. Gai, et al., "Intercrossed access controls for secure financial services on multimedia big data in cloud systems", ACM Trans. on Multimedia Comp., Comm., and App., 2016
6. H. Qiu, T. Dong, et al., "Adversarial attacks against network intrusion detection in IoT systems," IEEE IoT J. 8(13), 10327-10335, 2020
7. Zhu L, Wu Y, Gai K, et al. Controllable and trustworthy blockchain-based cloud data management[J]. Future Gene. Comp. Systems, 2019, 91: 527-535.
8. Gai K, Wu Y, Zhu L, et al. Differential privacy-based blockchain for industrial internet-of-things[J]. IEEE TII, 2019, 16(6): 4156-4165.
9. Borkowski M, Frauenthaler P, Sigwart M, et al. Cross-blockchain technologies: Review, state of the art, and outlook[J]. White paper, 2019.
10. Liu X, Liu J, Zhu S, et al. Privacy risk analysis and mitigation of analytics libraries in the android ecosystem[J]. IEEE TMC, 2019, 19(5): 1184-1199.
11. Li L, Liu J, Cheng L, et al. Creditcoin: A privacy-preserving blockchain-based incentive announcement network for communications of smart vehicles[J]. IEEE TITS, 2018, 19(7): 2204-2220.
12. IPFS Homepage, https://ipfs.tech/. Last accessed 29 Oct 2022
13. Wang W, Song J, Xu G, et al. Contractward: Automated vulnerability detection models for Ethereum smart contracts[J]. IEEE TNSE, 2020, 8(2): 1133-1144.
14. Bethencourt J, Sahai A, Waters B. Ciphertext-policy attribute-based encryption, IEEE sym. on secu. and pri. (SP'07). 2007: 321-334.
15. Qiu M, Gai K, Xiong Z. Privacy-preserving wireless communications using bipartite matching in social big data[J]. Future Gene. Computer Sys., 2018, 87: 772-781.
16. Zyskind G, Nathan O. Decentralizing privacy: Using blockchain to protect personal data, IEEE Security and Pri. Workshops. 2015: 180-184.
17. Wang W, Shang Y, He Y, et al. BotMark: Automated botnet detection with hybrid analysis of flow-based and graph-based traffic behaviors[J]. Information Sciences, 2020, 511: 284-296.
18. Rahulamathavan Y, Phan R C W, Rajarajan M, et al. Privacy-preserving blockchain based IoT ecosystem using attribute-based encryption, IEEE Int'l Conf. on Advanced Netw. and Telecomm. Systems (ANTS). 2017: 1-6.
19. Han D, Zhu Y, Li D, et al. A blockchain-based auditable access control system for private data in service-centric IoT environments[J]. IEEE TII, 2021, 18(5): 3530-3540.
20. Li T, Wang H, He D, et al. Blockchain-based privacy-preserving and rewarding private data sharing for IoT[J]. IEEE IoT J., 2022.
21. Ren W, Sun Y, Luo H, et al. SILedger: A blockchain and ABE-based access control for applications in SDN-IoT networks[J]. IEEE TNSM, 2021, 18(4): 4406-4419.
22. Yu G, Zha X, Wang X, et al. Enabling attribute revocation for fine-grained access control in blockchain-IoT systems. IEEE T. Eng. Manag., 2020, 67(4): 1213-1230.
23. Yi L, Sun Y, et al. CCUBI: A cross-chain based premium competition scheme with privacy preservation for usage-based insurance. Int'l J. of Intell. Syst., 2022.
24. Gai K, She Y, Zhu L, et al. A Blockchain-based Access Control Scheme for Zero Trust Cross-organizational Data Sharing[J]. ACM TOIT, 2022.
25. WeCross Homepage, https://wecross.readthedocs.io/zh_CN/latest/. Last accessed 29 Oct 2022

Research on Diabetes Disease Development Prediction Algorithm Based on Model Fusion

Wenyu Shao[1*], Xueyang Liu[2], Wenhui Hu[3], Xiankui Zhang[4], and Xiaodong Zeng[5]

[1]School of Software and Microelectronics, Peking University, Beijing, China

swy1798@stu.pku.edu.cn

[2]National Engineering Research Center for Software Engineering Peking University, Beijing, China

liuxueyang@pku.edu.cn

[3]National Engineering Research Center for Software Engineering Peking University, Beijing, China

huwenhui@pku.edu.cn

[4]School of Software and Microelectronics, Peking University, Beijing, China

zxkui1999@stu.pku.edu.cn

[5]School of Software and Microelectronics, Peking University, Beijing, China

zengxiaodong@stu.pku.edu.cn

Abstract. In today's world, with the deepening of population aging, chronic diseases have become the main diseases which affecting human health. Diabetes is a common chronic disease. Its incidence rate is high and rising year by year. For patients with diabetes, it is very important to predict the development of the disease and the possible complications for their follow-up treatment and recovery. However, the existing prediction of diabetes is mostly limited to the prediction of the incidence rate of patients, and only uses collaborative filtering or features for prediction, and rarely uses the patient's condition information to construct sequences and predict the development of the disease. Our aim is to make a accurate prediction on the development of diabetes patients and the possible complications. We need to use the historical development information of patients. Therefore, we propose a sequence based model fusion prediction algorithm, which effectively fuses the sequence and feature information. We use the high-order Markov Chains with attention mechanism as the basic learner for learning sequence information, and we also use XGBoost and CatBoost as the basic learner for learning feature information. Finally, LightGBM is used as a meta learner to fuse the output of the base learner. Experiments on the data of diabetes patients show that our method achieves better results than the original sub learner.

Keywords: Diabetes Patients Development Prediction · Markov Chains · CatBoost · Sequence Prediction · Model Fusion

1 Introduction

Diabetes is a chronic disease with a large number of patients and a high incidence rate. It is of positive significance in disease control and complication prevention to predict the disease development and possible complications of diabetes patients. With the development of medical informatization, various medical institutions have also accumulated a large number of patient's medical electronic data in the process of patient diagnosis and treatment. Making full use of these data and mining valuable laws from

them will help doctors adjust treatment plans in time according to the development of patients' conditions, improve treatment efficiency, and help patients better control their own disease development and conduct follow-up recovery and treatment more scientifically [1].

In recent years, researchers have made many explorations in the research and prediction of diabetes. Peijie Du used statistical methods to collect the statistical data of diabetes incidence rate, prevalence and risk factors of community residents over 40 years old in Zhengzhou, China, and summarized the main factors leading to the disease; Pal et al. used naive Bayes, decision tree, SVM and other machine learning algorithms to establish a prediction model for retinopathy, and analyzed its prediction performance and so on. However, most of the existing diabetes prediction is only used to predict the incidence rate. They mainly use collaborative filtering or human characteristics to predict the incidence rate, rather than using the diagnosis and treatment sequence of diabetes patients to predict the disease development [2].

Sequence prediction is to learn the change rule of user' behavior or preference from the user' behavior sequence, so as to predict the subsequent behavior of the user [3]. In the prediction of the development of diabetes, namely learning the rule of the change of the patient's condition from the historical diagnosis and treatment sequence of diabetes patients, so as to predict the development degree of the patient's condition. Sequence prediction can make good use of rich historical data and mine corresponding rules. In this paper, the high-order Markov Chains with attention mechanism is used to learn the sequence information of the disease development of diabetes patients, which helps to achieve a more accurate prediction of the disease development of diabetes [4].

Model fusion is a method to complete learning tasks by building multiple learners and combining them. A learner is a model. Different models have their own advantages. By combining multiple models, model fusion gives play to the advantages of each model, and obtains a learner with stronger effects. Stacking is a relatively advanced model fusion method. Its idea is to train multiple base learners based on the original data, and then combine the prediction results of the base learners into a new training set to train a new learner. The new learner is also called a meta learner, so as to reduce the dependence on a single learner and improve the performance of the model [5]. In this paper, high-order Markov Chains with attention mechanism, XGBoost and CatBoost are selected as the base learners of model fusion, and LightGBM is selected as the meta learner for the experiment. In feature engineering, XGBoost is used to obtain the highest accuracy, CatBoost is used to process category features, and LightGBM is used to achieve the highest accuracy as quickly as possible. From the analysis of the experimental results, it can be seen that the effect of the fused model is better than any single model.

The content organization structure of this article is as follows:

Chapter 1: Introduction. First, the status quo of diabetes disease prediction is introduced. Secondly, the applicability of sequence prediction in diabetes prediction is analyzed, and the method of model fusion in this paper are introduced. Finally, the main work of this paper is introduced.

Chapter 2: Related work. Firstly, the disadvantages of traditional recommendation algorithms are introduced. Secondly, the advantages of sequence based recommendation algorithm are introduced. Finally, we introduce the high-order Markov algorithm combined with the attention mechanism and the model fusion technology.

Chapter 3: Our Methods. First, we introduce the high-order Markov algorithm combined with attention mechanism. Secondly, the basic learner of the feature-based prediction algorithm for the development of diabetes is introduced. Finally, the specific scheme design of model fusion is introduced.

Chapter 4: Experiment and result analysis. The experimental environment is introduced, and the experimental results are displayed and analyzed.

Chapter 5: Conclusion and future work. First, the work of this paper is summarized, and then the future research direction is proposed.

2 Related Work

In recent years, with the rapid development of the Internet and the popularity of mobile devices, the amount of information that people are exposed to has reached an explosive level. The recommendation system can help people find the information they need accurately and quickly from the redundant information. The recommendation algorithm is the core factor that affects the accuracy of the recommendation system. There is a large amount of historical preference information of users in such a large amount of data. Although traditional recommendation algorithms, such as content-based recommendation algorithm and collaborative filtering algorithm, have achieved good results on the whole, they have the problem of insufficient model expression ability. In particular, it can only simulate the interaction between users and projects statically, and can only capture the general preferences of users, and can not make full use of the existing historical preference information of users. While the sequence based recommendation algorithm generally considers that the user's behavior at a certain time is determined by the user's historical behavior. It regards the user's historical behavior or preference as a dynamic and sequential sequence, and through such serialization, it understands and learns the user's historical behavior, so as to predict the user's subsequent behavior or preference more accurately [6]. Sequence recommendation algorithms include Markov chain based recommendation algorithm and deep learning based recommendation algorithm. Both of them make prediction and recommendation based on user historical behavior sequence.

2.1 High Order Markov Chains With Attention Mechanism

Markov chain is a kind of random time series with no aftereffect. It considers that the sequence state of a random sequence process at a certain time is only related to the state of the previous time, and not related to the state at an earlier time. It is a mainstream sequence modeling method. This method is widely used in the research of population quality of life, disease prediction and prevention, and has good effect. The factorized Markov Chains (FPMC) model was proposed in 2010 as a method for sequence prediction by combining matrix decomposition and Markov Chains by Rendle et al. The model learns a corresponding transfer matrix for each user, and then uses maximum likelihood estimation to solve the parameters. However, it has sparsity and long tail distribution on many data sets [7]. Fusing similarity models with Markov Chains (Fossil) model is a high-order Markov Chains model proposed by Ruining He et al in 2016 [8]. The model smoothly combines similarity based methods with Markov Chains, uses similarity based methods to model users' long-term preferences, and uses high-order Markov Chains to model users' short-term preferences, Then, the combination of long-term and short-term preferences realizes personalized sequence prediction for sparse data and long tail data sets. The meaning of high-order is to extend the traditional view that a state at a certain time is only related to the state at the previous time to the state at the previous multiple times. The number of related states is reflected as the order. This parameter can be set by yourself. Therefore, we can make full use of the user's historical behavior sequence, learn more information through the high-order Markov Chains, and high-order Markov Chains have stronger practical significance. On this basis, we think that we should consider that there may be differences in the degree of influence of the states at the previous times on the current state. Therefore, we add the attention mechanism to the algorithm, which can help adaptively learn the weight

of the influence of the past states on the current state, so as to achieve stronger practical significance and more accurate recommendation effect [9].

2.2 Model Fusion

Model fusion is to train multiple learners according to a certain method, that is, to train multiple models. It has been proved mathematically that with the increase of the number of individual classifiers, the error rate of the integration will decrease exponentially, and eventually it will be zero. Model fusion can make different models learn from each other and achieve the effect of model optimization. Stacking is a relatively advanced model fusion method, which uses raw data to train multiple base learners, and then combines the prediction results of the base learners into a new training set to train a new learner. The new learner is also called a meta learner. The most common stacking structure is that only one layer of meta learners is stacked on the base learner, and the single-layer stacking can flexibly select their favorite models on the base learner and the meta learner. This structure improves the effect of the model at the cost of increasing relatively small complexity [10].

3 Our Method

In order to predict the development of diabetes patients, the historical diagnosis and treatment information of patients is very important. We use the high-order Markov Chains with attention mechanism to learn the historical diagnosis and treatment sequence information of diabetes patients. The characteristics of patients are also essential for the prediction of disease development. Therefore, we selected several classical machine learning algorithms to learn the characteristics of patients and then predict the disease development [11]. Finally, we use model fusion to fuse the above sub models to obtain the final model.

3.1 High Order Markov Chains With Attention Mechanism

In this paper, the high-order Markov Chains with attention mechanism is selected as one of the basic learners. The operation steps of the model are shown in Figure 1.

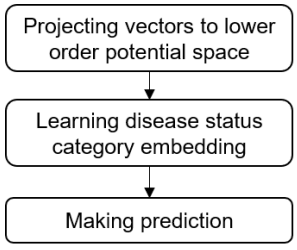


Fig. 1. Running steps of the model

First, the high-order Markov Chains with attention mechanism will project the patient's historical state of illness category vector into the low-order potential space, and then the patient's long-term state of illness information is modeled.

$$p_u(j \mid i) \propto \underbrace{\sum_{j' \in J_u^+ \setminus \{j\}} < \vec{P}_i, \vec{Q}_j >}_{\text{modeling of long-term condition information}} \tag{1}$$

$\vec{Q}_j$ represents the state of disease at the final time to be predicted in each disease sequence, $\vec{P}_i$ represents the patient's condition at each historical moment before the final moment. Then, the high-order Markov Chains is used to model the short-term condition state information of the patient, and the probability that the condition state j becomes the state of the patient's next condition under the given condition state

sequence is calculated.

$$p_u(j \mid \underbrace{\mathcal{S}_{t-1}^u, \mathcal{S}_{t-2}^u, \ldots, \mathcal{S}_{t-L}^u}_{\text{patient short-term state sequence}}) \propto \left\langle \underbrace{\sum_{k=1}^{L} \overbrace{(\eta_k + \eta_k^u)}^{\text{Personalized weighting factor}} \cdot \vec{P}_{\mathcal{S}_{t-k}^u}, \vec{Q}_j}_{\text{modeling of short-term condition information}} \right\rangle \quad (2)$$

$\mathcal{S}_{t-1}^u, \mathcal{S}_{t-2}^u, \ldots, \mathcal{S}_{t-L}^u$ represents the patient's short-term state sequence, and each patient has a global deviation vector $\eta_1^u, \eta_2^u, \ldots, \eta_L^u$. The basic idea is that each patient's previous state of illness has different weights for the high-order smoothness.

The attention mechanism is realized by calculating the similarity between the historical state of illness and the target state. The specific method is to calculate the implicit vector of the historical state of illness category in the learning sequence and the implicit vector inner product of the state to be predicted, and replace the original vector with the newly generated implicit vector of the historical state of illness category. Thus, different weights are assigned to the state of illness at different times in the history of diabetes patients, It reflects the influence degree of the historical state of illness at different times on the target state.

$$\vec{a}_{i,j} = < \vec{P}_i, \vec{Q}_j > \quad (3)$$

$$p_u(j \mid i) \propto \underbrace{\sum_{j' \in \mathcal{J}_u^+ \setminus \{j\}} \left\langle \overbrace{\vec{a}_{i,j}}^{\text{attention weight}} \cdot \vec{P}_i, \vec{Q}_j \right\rangle}_{\text{modeling of long-term condition information}} \quad (4)$$

$$\vec{a}_{\mathcal{S}_{t-k}^u, j} = < \vec{P}_{\mathcal{S}_{t-k}^u}, \vec{Q}_j > \quad (5)$$

$$p_u(j \mid \underbrace{\mathcal{S}_{t-1}^u, \mathcal{S}_{t-2}^u, \ldots, \mathcal{S}_{t-L}^u}_{\text{patient short-term state sequence}}) \propto \left\langle \underbrace{\sum_{k=1}^{L} (\eta_k + \eta_k^u) \cdot \overbrace{\vec{a}_{\mathcal{S}_{t-k}^u, j}}^{\text{attention weight}} \cdot \vec{P}_{\mathcal{S}_{t-k}^u}, \vec{Q}_j}_{\text{modeling of short-term condition information}} \right\rangle \quad (6)$$

u represents the current patient, j represents the state of the patient at the target time, i and $\mathcal{S}_t^u$ represents the patient's state of illness at different historical times, $\mathcal{J}_u^+$ indicates the patient's condition state set, $\vec{P}_i$ represents the hidden vector of the patient's state of illness at the historical time, $\vec{Q}_j$ represents the hidden vector of the patient's disease state at the target time, $\vec{a}_{i,j}$ denotes attention weight, η Represents a global parameter shared by all patients, η^u represents a personalized weighted scalar for a specific patient, which is used to control the relative weight of the long-term and short-term patient's state of illness.

Finally, the maximum a posteriori probability (MAP) estimation and stochastic gradient descent (SGD) method are used to solve the model parameters, and the algorithm training and model optimization are carried out to obtain the final model. By calculating the conditional transition probability of the target disease state, the matching probability of the candidate disease state to be recommended is calculated using the softmax function and the obtained prediction probability to generate the K disease states with the highest probability as the final prediction result. The final prediction result formula of the high-order Markov model combined with the attention mechanism is as follows, β_j is the offset term used to normalize the long-term dynamic component.

$$\hat{p}_{u,t,j} = \underbrace{\beta_j}_{\text{offset term}} + \underbrace{\sum_{j' \in J_u^+ \setminus \{j\}} \left\langle \overbrace{\vec{a}_{i,j}}^{\text{attention weight}} \cdot \vec{P}_i, \vec{Q}_j \right\rangle}_{\text{modeling of long-term condition information}} +$$

$$\underbrace{\left\langle \sum_{k=1}^{L} (\eta_k + \eta_k^u) \cdot \overbrace{\vec{a}_{S_{t-k}^u, j}}^{\text{attention weight}} \cdot \vec{P}_{S_{t-k}^u}, \vec{Q}_j \right\rangle}_{\text{modeling of short-term condition information}} \tag{7}$$

3.2 Feature-based Basic Learners For Predicting The Development Of Diabetes

We use feature engineering technology for feature selection firstly, and then use two classical machine learning models as the basic learners for predicting the development of diabetes.

3.2.1 Feature Engineering

Feature engineering is particularly important in machine learning. In this experiment, the characteristics of diabetes patients have a very important impact on the development of diabetes. Through the observation and analysis of the data set, we can obtain the personal information of diabetes patients, including basic information, previous history, family history, personal history, marriage and childbearing history, etc, the basic information including the patient's age, height, weight, occupation, etc. For the needs of the later experiments, we standardized continuous digital features, and coded discrete classification features using one-hot code [12].

After preliminary screening, we selected 81 dimensions of features. In order to ensure the effectiveness of features, we use logistic regression and XGBoost to process features of patients. XGBoost is an integrated learning method. Its corresponding model is multiple CART trees with dependency. By adding CART trees that can improve the overall effect, it uses features to split, fit the residuals of the last prediction, and continuously reduce losses. At the same time, in order to reduce the risk of over fitting, it limits the number of leaf nodes by adding penalties in the objective function. The algorithm finally adds the predicted value of each tree together as the final predicted value [13]. CatBoost is a GBDT framework with fewer parameters and high accuracy that supports category variables based on symmetric decision tree based learners. It uses combined category features and can take advantage of the relationship between features, which greatly enriches feature dimensions and can efficiently and reasonably handle category features [14].

3.2.2 Learning and training

By inputting the personal information of patients with diabetes, we can output the probability corresponding to the development status of diabetes patients in the next stage. About the objective function, we choose the cross entropy loss function as the objective function, and the calculation formula is as follows.

$$L(\hat{y}) = -\sum_{i=1}^{m} y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \tag{8}$$

$\hat{y}$ is the matching probability of the disease development state of the diabetes patient in the next stage predicted by the base learner, and m is the number of candidate states.

3.3 Model Fusion

The method framework of this paper is shown in Figure 2. Our method is a model fusion algorithm based on stacking, and the method steps are as follows.

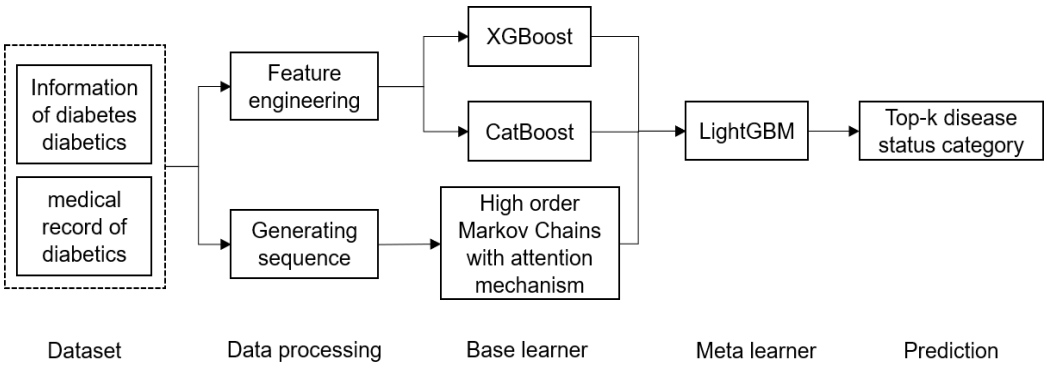


Fig. 2. Framework diagram of our method

- Firstly, preprocess the original data to obtain the characteristics of diabetes patients and the development sequence of their state of illness.
- Secondly, we choose the high-order Markov Chains with attention mechanism, XGBoost and CatBoost as the base learners; Among them, the inputs of XGBoost and CatBoost are the characteristics of diabetes patients, and the inputs of the high-order Markov Chains with attention mechanism are the state sequence of diabetes patients. The output result of these basic learners is the matching probability of the development state of diabetes patients in the next stage.
- Then, we choose LightGBM as the meta learner to obtain the highest accuracy in the shortest possible model training time; We take the output of the base learner as the input of the meta learner and learn through the meta learner LightGBM; The output result of the meta learner is the final matching probability of the development state of diabetes patients in the next stage.
- Finally, according to the final matching probability, predicting the development status of the Top-k next stage of diabetes patients.

4 Experiment And Result Analyzing

4.1 Experimental Setting

The data set in this paper comes from the patient visit data of some hospitals in a certain area. The final experimental data set is obtained by manual labeling, rule screening and personal information data masking. Our experimental data set includes two types of tables: diabetes patient information table and diabetes patient diagnosis and treatment record table. The information table of diabetes patients records the relevant information of diabetes patients, including the patient's age, height, weight, occupation, previous history, family history, personal history, marriage and childbirth history, etc; The diagnosis and treatment record of diabetes patients is the diagnosis and treatment information of the diabetes patients from admission to discharge. We obtain the historical disease development sequence of each diabetes patient from the diagnosis and treatment record table of diabetes patients, and select the influencing

factors of disease development from the information table of diabetes patients as the characteristics of the algorithm.

4.1.1 Classification Of Disease State

According to the development stage of diabetes, the patient's condition was classified. The state of the disease is mapped into a two bit numeric code. Among them, the first one represents the major category of diabetes to indicate whether the patient has diabetes complications and the types of complications. The second one represents the severity of the disease corresponding to the current major category of diabetes. The specific method is to normalize each index examined by the patient, convert it into a numerical value, and then carry out a weighted average. Finally, sixty disease status labels are obtained.

4.1.2 Generating Disease State Sequence

Based on the diagnosis and treatment record table of diabetes patients and the classification results of the disease status, the corresponding codes of the current disease status of the patients are calculated according to the classification of the disease status and the current detection indicators of the patients, thus generating the sequence of the patient's disease development. Then for sequence s, assuming its total length is n, s [k] is the prediction target (k $\in$ [0, n]) of sequence s [0: k]. Therefore, a sequence with a total length of n can generate n-1 pieces of training data. Among all the data, 60% of the data of diabetes patients were used for training, 20% for validation, and the remaining 20% for testing.

4.2 Results And Analysis

We chose Recall and NDCG (normalized discounted cumulative gain) to evaluate the effect of the experiment.

Recall is also called recall rate, which means the probability of being predicted as positive samples in the actual positive samples, so as to find out how many of the actual positive samples are predicted as positive.

$$Recall = \frac{TP}{TP+FN} \tag{9}$$

TP represents the number of positive samples predicted to be positive, and $TP + FN$ represents the total number of positive samples actually.

NDCG (normalized cumulative gain) uses the ratio of DCG and IDCG of each patient as the normalized score, and DCG (Discounted cumulative gain) can evaluate the recommended list. Its core idea is that when there is a relatively high correlation result in the lower position of the search result list, punishment should be imposed on the evaluation score, and the punishment proportion is related to the logarithmic value of the corresponding result location. Therefore, it takes into account the factor of sorting order. IDCG is the maximum DCG value under ideal conditions.

$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i-1}}{\log_2(i+1)} \tag{10}$$

$$IDCG_p = \sum_{i=1}^{|REL|} \frac{2^{rel_i}-1}{\log_2(i+1)} \tag{11}$$

$$nDCG_p = \frac{DCG_p}{IDCG_p} \tag{12}$$

In these formulas, i indicates different sorting positions in the recommendation results, and rel_i represents the correlation at i, p represents the recommended number of results, and $|REL|$ indicates the optimal ranking of recommendation results.

In order to evaluate the performance of our method, we conducted experiments on data.The experimental results are shown in Table 1. It can be seen from the results that our method is superior to the other base learner in terms of Recall and NDCG, and we have achieved better prediction performance. Model fusion combines Markov's good at processing sequence data, XGBoost and CatBoost's use of rich feature information, makes each sub model give full play to its advantages, uses other sub models to make up for its shortcomings, and finally achieves better results than a single model [15]. Affected by the relatively detailed classification of disease state, the overall accuracy of the model is not particularly ideal. However, in general, the model has a high accuracy in predicting the development of diabetes patients and has good practical significance.

Table 1. Experimental Results Of Our Method And Baseline

Method	Top-5		Top-10		Top-20	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
XGBoost	0.7529	0.5483	0.8064	0.5681	0.8237	0.5735
CatBoost	0.7697	0.5452	0.8023	0.5729	0.8317	0.5798
High order Marcov Chains with attention mechanism	0.7876	0.5782	0.8325	0.5977	0.8754	0.6201
Our method	0.8254	0.5949	0.8751	0.6198	0.9192	0.6317

5 Conclusion And Future Work

In this experiment, we quantified the examination indicators of diabetes patients and obtained the disease degree value of the patients. Then we used the high-order Markov Chains with attention mechanism to learn the disease development sequence of diabetes patients, and used XGBoost and CatBoost to learn the influence of the characteristics of the patient's personal information on the patient's disease development. We choose these three algorithms as the base learners, and we chose LightGBM as the meta learner for model fusion. We took the output of the base learner as the input of the meta learner and output the final prediction result through the meta learner. The experimental results show that compared with using a single model, the model fusion method has obvious advantages. Moreover, Markov model and model fusion technology are widely used in the research of population quality of life, disease prediction and prevention, and have good effects. This experiment proves that Markov model and model fusion technology also have good effects in using the patient's disease progression sequence to predict the patient's disease progression. In future work, we will try the combination of other algorithms in model

fusion, including changing the number and types of learners, so as to explore whether we can further enhance the effect of the model and explore the application of this model fusion in other scenes.

References

1. Guomin J, Liu Y. Research on Development Opportunities and Countermeasures of "Internet Plus Chronic Disease Management" [J]. China medical insurance, 2022 (07): 46-52

2. He B. Prediction of Diabetes Based on Convolutional Neural Network[D]. Southwest University, 2019

3. Wangrui J, Peixi K, Zhuju Y, et al. Multivariable time series forecasting using model fusion[J]. Information Sciences,2022,585

4. Wangjia M, Liuqiu P, Zhangming L, et al. Effectiveness of different screening strategies for type 2 diabete on preventing cardiovascular diseases in a community-based Chinese population using a decision-analytic Markov model[J]. Journal of Peking University (Medical Edition), 2022,54 (03): 450-457

5. Liu W, Zouwei H, Luyan J, et al. Research on image recognition of Chinese medicinal materials based on transfer learning and model fusion[J]. Journal of Hunan University of traditional Chinese medicine, 2022,42 (05): 809-814

6. Puhong F, Shaojian F, Zhangxiao W, et al. Sequence recommendation of fusing dynamic interest preference and feature information[J]. Journal of Yunnan University (NATURAL SCIENCE EDITION), 2022,44 (04): 708-717

7. Steffen Rendle, Christoph Freudenthaler, Lars Schmidt-Thieme. Factorizing personalized Markov chains for next-basket recommendation[P]. World wide web,2010

8. Ruining He, Julian McAuley. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation[J]. CoRR,2016

9. Wujian F. Research of attentional mechanisms involved in the recognition of diabetes retinopathy[D]. Guangdong Normal University of technology, 2022

10. Duanqian W. Personal credit risk assessment based on stacking fusion model[D]. Shandong University, 2021

11. Chensi H, Zhangyun Q. Machine learning-based prediction model of type 2 diabetes mellitus complications[J]. Chinese Journal of Medical Library and information, 2020,29 (11): 31-38

12. Dengxiu Q, Xiewei H, Liufu C, et al. A Prediction Model for Advertising Click Conversion Rate Based on Feature Engineering[J]. Data collection and processing, 2020,35 (05): 842-849

13. Yang Jian,Guan Jinhan. A Heart Disease Prediction Model Based on Feature Optimization and Smote-Xgboost Algorithm[J]. Information,2022,13(10)

14. Paul-Adrian Câlburean,Paul Grebenişan,Victor Vacariu, et al. Prediction of 3-Year All-Cause Death in a Percutaneous Coronary Intervention Registry using Machine Learning: A Comparison Between Random Forest and CatBoost Algorithms[J]. Applied Medical Informatics,2021,43

15. Ma S, Cuijian F, Xiaowei D, et al. Deep Learning-Based Data Augmentation and Model Fusion for Automatic Arrhythmia Identification and Classification Algorithms[J]. Computational Intelligence and Neuroscience,2022,2022

Smart-Contract Vulnerability Detection Method Based on Deep Learning

Zimu Hu[1], Wei-Tek Tsai[1,2], Li Zhang[1]

[1] Digital Society & Blockchain Laboratory, Beihang University, Beijing, China
{huzmu049, leezhang} @buaa.edu.cn
[2] Beijing Tiande Technologies, Beijing, China
tsai@tiandetech.com

Abstract. With the rapid development of blockchain technology, smart contracts (SCs) applied in digital currency transactions have been widely used. However, SCs often have vulnerability in their code that allow criminals to exploit them to steal associated digital assets. Benefiting from the development of machine learning technology and the improvement of hardware performance, one can use deep learning techniques to analyze code and detect vulnerabilities. This paper proposes an innovative combination of opcode sequences and abstract syntax trees for source code parsing. And a method based on the combination of self-attention mechanism and bidirectional long-short term memory neural network is proposed to detect the vulnerability of SCs after word embedding. Experimentation results show that the two parsing methods can complement each other and effectively improve the accuracy of vulnerability detection.

Keywords: Blockchain, Smart Contract, Deep Learning, Vulnerability Detection

1 Introduction

With the development of blockchain (BC), smart contracts (SCs) running on top of a BC have received significant attention. Since 2009, BC has gone through several significant innovations with different architectures, consensus protocols, and transaction processes. For example, instead of single BC systems completing all the steps of financial transactions, new BC systems often perform only one step of a financial transaction, such as KYC (Know Your Customers), trading, or settlement. As a BC is responsible for only one step of a financial transaction, the complexity of the involved BC is significantly reduced. Thus, the overall system can have much better scalability, optimization, and performance. Passed through the first generation of BC system Bitcoin, the second generation of BC system Ethereum, and then to the third generation of BC with multi-chain structure ChainNet [1]. Nowadays, with the emergence of concepts such as the metaverse, BC is undergoing innovation, and there will be a more powerful and well-regulated BC system architecture. SC, as an electronic agreement that can execute contract terms without mutual trusts, is of great significance in both legal theory and the development of computer technology.

SC code is stored and run on a BC. Due to the BC characteristics, it is difficult to change, and the content of BC can be seen by all participants. If the code with vulnerability is submitted to the chain, it will be easy to be maliciously exploited. Sometimes it can even lead to serious losses. The DAO incident, an attack against SC [2], appeared in the Ethereum, and caused huge economic losses. Attackers used a reentrancy vulnerability to steal more than 3.6 million ETH, causing losses of up to $70 million. Because of this, vulnerability detection of SC has become an important research topic.

Some studies have used methods of symbolic execution [3,4] and fuzzing [8,9]. However, the efficiency of detection is slow, and it relies too much on expert knowledge. There are also some studies that use machine learning-based methods [10,12,13]. Some of them use source code as inputs while others use opcode. There are 145 different opcodes designed in the Ethereum Yellow Book. Each opcode corresponds to an operation.

This paper proposes a deep-learning-based vulnerability-detection method for SCs. This approach parses the SC source code into opcode and abstract syntax tree (AST). And the two vectors are represented respectively, and the neural network model is used for feature learning. Then, the vectors are fused, and the obtained fusion features are classified to obtain whether the code has vulnerabilities. In the deep learning method, we use RNN, GRU, and LSTM (Long Short-Term Memory) to learn the features. Meanwhile, we add the attention mechanism.

This paper is organized as follows: Section 2 introduces related work; in Section 3 proposes a vulnerability detection method based on the combination of self-attention mechanism and BiLSTM neural network; Section 4 discusses experimental results; Section 5 finally concludes this paper.

2 Related Work

The existing research mainly focuses on three general directions, which are symbolic execution, fuzzing, and machine learning. Symbolic execution can use symbols to analyze unknown variables, and it can use the information stored on the BC outside the SC to deduce it as symbols. Slither [3] proposed by Feist, S-gram [4] proposed by Liu, Manticore [5] proposed by Mossberg and Osiris [6] proposed by Torres all use symbolic execution to detect SC vulnerability. Although symbolic execution can detect vulnerabilities in SCs, it may not be able to solve with the constraints of the program are too complex. At the same time, it cannot get rid of the heavy dependence on the prior knowledge of experts. The time required to detect a SC often ranges from tens to hundreds of seconds, depending on the complexity of the code.

Fuzzing is another detection method. For example, Torres's study [7], Sereum [8], and Jiang's Contractfuzzer [9]. These dynamic analysis tools can execute the code. However, with dynamic analysis, analysts need to implement and execute attack patterns to prevent vulnerabilities in advance, so analysts need to be fully aware of attacks on SCs. Therefore, these dynamic analysis tools rely heavily on the rule setting, and on the prior knowledge of experts. At the same time, dynamic analysis takes a long time to execute, because the probability of detecting a contract vulnerability can be higher only if more inputs are executed. Such detection methods often need to run for hundreds of seconds to detect vulnerabilities.

With the improvement of machine learning technology, SC vulnerability detection methods based on machine learning have been proposed. It uses machine learning to learn the features of SCs, to classify the code into two cases: vulnerability and no vulnerability. Ashizawa proposes Eth2Vec [10], a neural network training tool based on the PV-DM model [11]. There are also some SC analysis tools that cannot automatically extract features and require human input. For example, the studies of Momeni [12] and Wang [13] use classical machine learning algorithms for analysis, but they may output wrong results when the code to be analyzed is rewritten. Qian proposed Vul-DeeSmartContract [14], an automatic feature extraction method based on Word2Vec and LSTM [16], but it can target re-entrancy vulnerability only.

3 Methodology

3.1 Overview

This paper proposes SC vulnerability detection method based on deep learning is mainly divided into the following contents: First, the SC code is preprocessed into opcode sequences and abstract syntax tree sequences. Next, the Word2Vec [15] model is used to process the input into vectors. Then, a method based on the combination of self-attention mechanism [17] and BiLSTM [16] is used for feature learning, and finally a fully connected network is used to classify vulnerabilities.

There are many types vulnerabilities of SCs, and this paper mainly analyzes four of them: integer overflow, integer underflow, timestamp dependency, and reentrancy vulnerability. Integer vulnerability is divided into two cases: integer overflow and underflow. Overflow or underflow occurs if the stored integer is larger or smaller than the maximum or minimum value of the data type. If an attacker exploits the overflow or underflow, it may cause problems such as infinite loops or failure of execution conditions [18]. Timestamp dependency vulnerability is the use of timestamp related variables in the SC [2]. Some malicious miners will set the timestamp within the allowed range to benefit the SC and even steal the digital asset. The reentrancy is exploited in the DAO incident [2]. A reentrancy can be achieved if the attacker designs the callback function so that some function of the contract, such as a money transfer function, is called again.

The opcode sequences generated are generated from source code. Firstly, the SC source code .sol file should be compiled to generate the opcode file. After that, according to the division of each contract in the source code, the opcode to each contract is stored separately. In SCs written in Solidity, each .sol file can contain multiple contracts. This can be compared to the structure of multiple classes in a source file in other object-oriented programming languages. The structure of the SC .sol file is shown in Figure 1.

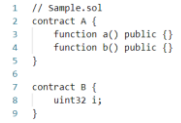


Fig. 1. The structure of .sol file

Instead of storing all opcodes in a source file together as many existing studies, this paper chooses to divide according to contracts. This can provide a finer granularity for

code vulnerability detection, and can detect which contract is vulnerable in a source code file. The SC vulnerability detection method proposed in this paper is based on the BiLSTM neural network model, that can consider the connections between contexts in the SC. It can detect vulnerabilities with high accuracy in a short time, and combine the self-attention mechanism to further improve the effect of the model. Figure 2 illustrates the vulnerability detection algorithm.

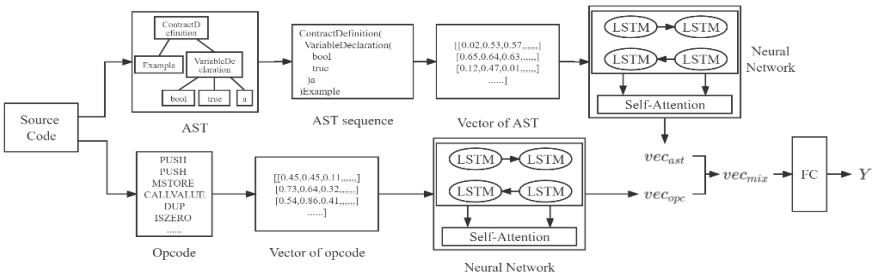


Fig. 2. Vulnerability Detection Algorithm

The vulnerability-detection algorithm consists of four steps: Firstly, the source code of the SC is preprocessed, and the abstract syntax tree sequence and opcode sequence are generated; Secondly, the word embedding operation is performed on the two sequences to generate the vectorized representations; Thirdly, the vectorized representation matrix put into the neural network for feature learning, and obtain vec_{opc} and vec_{ast}; Finally, binary classification detection was performed. To make full use of the extracted feature vectors, the algorithm concatenates vec_{opc} and vec_{ast} to obtain vec_{mix}. Then the vec_{mix} is input into the fully connected layer, and the classification result shows whether there is a vulnerability.

3.2 Smart Contract Preprocessing

SC code is written in Solidity, that contains a lot of redundant content, including comments and identifiers. Therefore, it is not appropriate to use the source code of SCs for vulnerability detection. This paper uses two different methods to preprocess the SC, compiling it into opcode and extracting the AST from it.

Since there are many kinds of opcodes in the Ethereum virtual machine, it needs to be simplified. By analyzing the function and structure of opcodes, this paper finds some opcodes that can be reduced. The simplification of these opcodes with similar semantics can reduce the features for vectorization, and reduce the complexity of feature learning. The opcodes are simplified according to the reduction rules, shown in Table 1.

By analyzing the characteristics of Solidity syntax, this paper designs a rule for structured traversal of Solidity AST. An AST does not represent every detail that in the code, but it preserves important parts of the source code. In Solidity, the structure represented by AST can be mainly divided into Definition, Expression, Statement and TypeName structure. For example, the Definition structure mainly includes Event, Function, Structure and Contract Definition. To better reflect the syntax structure information contained in Solidity, a rule for structured traversal sequence is formulated. First, each structure in the AST is represented by the structure type name and the node name, with the structure

name placed first. Then use parentheses to indicate the contents of the subtree, followed by the struct type name or node name. For the Definition structure, after the closing parenthesis, there is a definition identifier. In Expression structures, structure name is used at the end again.

Table. 1. Opcode simplification rules

Original opcode	Simplified opcode
PUSH1~PUSH32	PUSH
DUP1~DUP16	DUP
SWAP1~ SWAP16	SWAP
LOG0~LOG4	LOG

3.3 Embedding

After processing the SC code into opcode and AST, it is not possible to directly input this information into a neural network. The input must be in the form of vectors. Therefore, opcode and AST must be vectorized. This paper uses word embedding Word2Vec model. First, all the input will be decomposed to extract all the words and use one-hot encoding, but the dimension of one-hot encoding is large, and is not convenient to use. Therefore, the CBOW (Continuous Bag-of-Words) [15] model will be used to learn the word embedding matrix. In this way, all the words can be mapped to a short vector through the word embedding matrix, and the dimension of the vector can be reduced while the relationship between the identifiers is preserved. Using this method, all opcodes and AST can be vectorized.

3.4 Feature Learning

Once the code is processed into a word vector, it can be used for feature learning. This step uses the LSTM model. There are RNN, GRU, and LSTM in recurrent neural networks. GRU and LSTM, are both optimized models based on RNN. In terms of performance, both are able to capture long distance dependencies, unlike RNN, which can only pay attention to the previous word. And code vulnerabilities are not only related to the previous word, but may be related to several words in the context. Through analysis, it can be found that the number of parameters in the GRU is less than that of the LSTM. When the features to be learned are relatively simple, this difference is not obvious, and may even produce gains. But when the features are more complex, such as the features of the SC vulnerability, this difference will show up. At the same time, some studies have also found that when the amount of data to be learned is large, the effect of LSTM is also better than that of GRU, and this is also confirmed by our experimentation. As the number of parameters of GRU is small, the time cost required for training is also low. However, in the task of vulnerability detection, the effect of the model is prioritized over the training cost, so this paper finally chooses to use the LSTM.

LSTM is an improved recurrent neural network. There are gradient vanishing and gradient explosion problems in ordinary recurrent neural networks, which is also the reason why ordinary recurrent neural networks cannot deal with the problem of long-distance dependence. The idea of LSTM is to add a state C to preserve the long-term

state based on the original recurrent neural network. The hidden layer h_t can be calculated by the following formula.

$$\begin{cases} C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t \\ f_t = \sigma\left(W_f[h_{t-1}, x_t] + b_f\right) \\ \tilde{C}_t = tanh(W_C[h_{t-1}, x_t] + b_C) \\ i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \\ o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \\ h_t = o_t \times tanh(C_t) \end{cases} \tag{1}$$

Meanwhile, this method adds the attention mechanism, and chooses the Self-Attention mechanism. It is a variant of the attention mechanism [17], that can better find the correlation within the data features and deal with long-distance dependencies. Through this method, the relationship between the input sequences can be further analyzed. By adding the self-attention layer, the model can focus on the key words, so that the effect of the whole model is enhanced. This paper inputs all the output of the BiLSTM layer into the self-attention mechanism layer, so that the attention mechanism can fully learn. In the self-attention mechanism, the three parameters W, b and u need to be trained. W and b parameters are used to obtain u_t, and the calculation formula of vec is as follows.

$$u_t = tanh(W[h_{Lt}, h_{Rt}] + b) \tag{2}$$

$$a_t = softmax(u_t^T \cdot u) \tag{3}$$

$$vec = \sum_{t=1}^{T} a_t[h_{Lt}, h_{Rt}] \tag{4}$$

4 Experimental Verification

4.1 Evaluation Metrics

As a classification model, the SC vulnerability detection model needs to use certain metrics to evaluate its classification effect. The evaluation metric needs to be able to show the effect of the model accurately, and can compare the pros and cons of different models. There are four types of SC vulnerabilities studied in this paper, which are Integer overflow (IOF), Integer underflow (IUF), timestamp dependency (TD), and reentrancy (REN). Detecting each vulnerability is a binary classification. For the binary classification problem, the evaluation metrics used in this paper are accuracy (ACC), Recall and F1 score (F1). We calculate the metrics using the following formula: $ACC = \frac{TP+TN}{TP+TN+FP+FN}$, $Precision = \frac{TP}{TP+FP}$, $Recall = \frac{TP}{TP+FN}$ and $F1 = \frac{2*Precision*Recall}{Precision+Recall}$.

4.2 Data Sets

According to the method of selecting data sets in existing research, such as the work of Momeni [12] and Wang [13], this paper collects source code, makes labels, and builds its own data set Contract5000 in the same way. At the same time, the datasets in the study of Eth2Vec [10] are used for experimentation. The base case statistics of the two datasets are shown in Table 2. By collecting a large number of SC source code from Etherscan, the Contract5000 dataset retains 5321 SC codes that are different from each

other at the opcode level by compiling to opcode and removing duplication. Among them, there are 20327 contracts, 8560 contracts have IOF, 3396 contracts have IUF, 452 contracts have TD, and 96 contracts have REN. We label each of these contracts separately, with the presence of vulnerabilities marked as 1 and the absence of vulnerabilities marked as 0. The Eth2Vec dataset contains 5000 SC source code, which contains a total of 17237 contracts. There are 8730 IOF, 3594 IUF, 323 TD, and 58 REN.

Table. 2. Dataset statistics

Data Set	Source Code	Contracts	IOF	IUF	TD	REN
Eth2Vec	5000	17237	8730	3594	323	58
Contract5000	5321	20327	8560	3396	452	96

4.3 Training Environment and Parameter Settings

We use Keras and Tensorflow to build the model, and the language we use is Python. The hardware environment used for all the experiments is CPU: Intel Xeon at 2.50GHz, GPU at RTX 2080 Ti and 43GB Memory. In terms of parameter setting, for each dataset, we use 7:3 to divide the training set and test set, and choose binary cross entropy as the Loss function. The optimal gradient descent algorithm Adam was used in all the experiments. For the learning rate lr, we experimented with multiple values, and to prevent the model from overfitting, we added a dropout layer and set the parameter dr. We used [5,10,20] for the word vector dimension vop of opcodes, and [50,80,100] for the word vector dimension vast of abstract syntax trees. In the experimental results, we use the following parameters: lr=0.001, dr=0.2, batch=128, vop=10, vast=50.

4.4 Results and Comparison

In the experiment, we use the Eth2Vec and Contract5000 dataset to train and test the model. During the training process of Eth2Vec and Contract5000, the changes of Acc, Loss and F1-score of the training set are shown in Figure 3 and Figure 4.

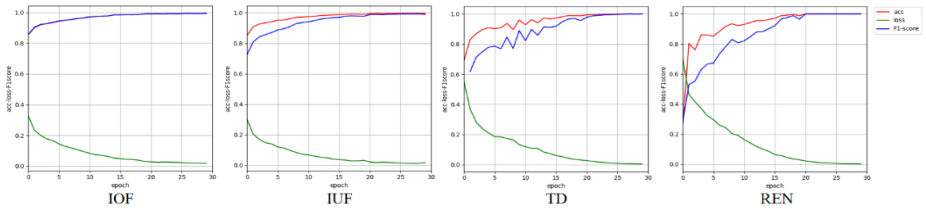


Fig. 3. Eth2Vec training set ACC, Loss, and F1 curves

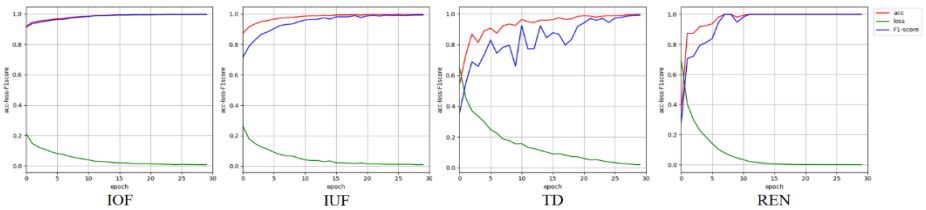


Fig. 4. Contract5000 training set ACC, Loss, and F1 curves

The final effect in the test set of the two datasets is shown in Table 3. In the experiments, we respectively experiment the effects of RNN, BiGRU, BiLSTM and Att-BiLSTM models on various vulnerabilities in this part, each model will input the opcode sequence and the abstract syntax tree sequence for learning at the same time, and the recurrent neural network hidden layer hyperparameter Settings of each model are the same. In the self-attention mechanism layer, we pass the output sequence of the hidden layer of BiLSTM at different moments for learning, so that the self-attention layer learns the whole sequence. The other parameters of the model remain the same, and the Eth2Vec dataset is uniformly used for the corresponding experiments. The division of training and test set is 7:3, and the specific experimental effects are shown in Table 4.

Table. 3. Test set effect of Eth2Vec and Contract5000

Dataset / Metrics	Eth2Vec				Contract5000			
	IOF	IUF	TD	REN	IOF	IUF	TD	REN
Acc	0.9331	0.9476	0.9233	0.9048	0.9569	0.9688	0.9079	0.8844
Recall	0.9542	0.9365	0.8912	0.8333	0.9626	0.9195	0.8897	0.8621
F1-score	0.9404	0.8913	0.7994	0.7500	0.9495	0.9079	0.7634	0.7143

Table. 4. Results for different models

Type / Models	IOF		IUF		TD		REN	
	Recall	F1	Recall	F1	Recall	F1	Recall	F1
RNN	0.793	0.762	0.493	0.391	0.320	0.383	0.647	0.386
BiGRU	0.946	0.928	0.912	0.848	0.866	0.740	0.706	0.511
BiLSTM	0.954	0.933	0.921	0.882	0.887	0.785	0.765	0.684
AttBiLSTM	**0.954**	**0.940**	**0.936**	**0.891**	**0.891**	**0.799**	**0.833**	**0.750**

From the data in the table, one can see that when simple RNN is used for vulnerability detection, except for the integer overflow vulnerability, the detection performance of the others of vulnerabilities is relatively poor, and the features of vulnerabilities cannot be well identified. However, the model using BiGRU and BiLSTM can achieve good detection effects on the four kinds of vulnerabilities, which also shows that the characteristics of vulnerabilities are indeed context-related. Only by taking full account of the context, can we achieve more accurate detection of vulnerabilities. When comparing the effect of BiGRU and BiLSTM, we find that the effect of using BiLSTM model is slightly higher than that of BiGRU. Although GRU is improved based on LSTM, it has fewer parameters than LSTM, so it is normal that the actual effect is slightly worse than LSTM. Considering the time of model training, GRU takes less time than LSTM. Since the problem of this paper is the vulnerability detection of SCs, the final effect of the model is more important, and the requirement for the time of model training is not very high. Therefore, the LSTM neural network can better meet the requirements of this paper.

AttBiLSTM represents the BiLSTM network using self-attention mechanism. Att-BiLSTM is the best one among the four network models in the table. In the IOF and IUF, the advantage of AttBiLSTM model is not obvious. However, on the REN, the effect of the network model with the self-attention mechanism has a more prominent performance, which indicates that the self-attention mechanism layer has learned more information, which improves the performance of the model. Combined with the whole comparison test, we can see that adding the self-attention mechanism can indeed improve the classification effect of the model, especially for the REN, that proves that the method proposed

in this paper is effective. To verify the effectiveness of the vulnerability detection method proposed in this paper, this paper selects five related works for comparison, and the comparison results with related works are shown in Table 5.

Table. 5. Comparison result with related work

Type / Models	IOF		IUF		TD		REN	
	Recall	F1	Recall	F1	Recall	F1	Recall	F1
SVM[12]	0.731	0.783	0.392	0.500	0.028	0.053	0.078	0.123
Eth2Vec[10]	0.576	0.701	0.560	0.637	0.170	0.273	0.548	0.615
ABCNN[18]	0.830	0.858	0.903	0.886	0.832	0.870	0.862	0.820
DR-GCN[19]	—	—	—	—	0.789	0.749	0.809	0.764
CGE[20]	—	—	—	—	0.881	**0.878**	**0.876**	**0.864**
AttBiLSTM	**0.954**	**0.940**	**0.936**	**0.891**	**0.891**	0.799	0.833	0.750

Through the experimental data, the effect of this paper is better than that of SVM and Eth2Vec. The method proposed in this paper can effectively detect vulnerabilities. Within comparison with ABCNN, we find that the method in this paper is superior to the comparison work in the detection of IOF and IUF, which indicates that under the condition of sufficient learning, BiLSTM has better ability to extract sequence features than TextCNN. However, it is slightly inferior to the comparison work on the REN. This is since the number of positive samples of the REN category in the dataset is small, and it is difficult to learn the features of the REN, which affects the final effect of the model. Within comparison with DR-GCN and CGE, since these two works both detect TD and REN, only the effects of these two vulnerabilities can be compared. The method in this paper is better than DR-GCN, but it is slightly inferior to CGE. Under the same conditions, the expert mechanism is used in CGE, which theoretically improves the effect of the model. In addition, CGE uses the callback point and edge specifically for the REN, and further optimizes the detection of the REN. However, the work of DR-GCN and CGE uses graph neural network, which cannot detect long code and needs to use specific methods to intercept fragments in the code, so it is not more universal than the method in this paper. Through the comparison of related work, the method proposed in this paper has a good performance in IOF and IUF, and can also show good results in TD and REN. In comparison with other methods, the method in this paper also has certain advantages, which can well complete the task of SC vulnerability detection.

5 Conclusion

To detect Ethereum SC vulnerabilities efficiently, this paper proposes a SC vulnerability detection method based on self-attention mechanism and BiLSTM neural network that integrates multiple code extraction techniques. Experiments show that the SC vulnerability detection method proposed in this paper can detect vulnerability with great coverage and efficiency. While this paper analyzes four kinds of vulnerability, the same method can be used with a different dataset to detect other kinds of vulnerability.

Acknowledgment. This work is supported by Chinese Ministry of Science and Technology (Grant No. 2018YFB1402700).

References

1. Wei-Tek Tsai et al. ChainNet, Oriental Publishing House, Beijing (2020).
2. N. Atzei, M. Bartoletti, and T. Cimoli. A survey of attacks on Ethereum smart contracts (sok). In Proc. of POST 2017, volume 10204 of LNCS, 10 pages 164–186. Springer, 2017.
3. Feist J, Greico G , Groce A . Slither: A Static Analysis Framework For Smart Contracts[C]// 2019 IEEE/ACM 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB). IEEE, 2019.
4. Liu H, Liu C, Zhao W, et al. Smashing smart: towards semantic-aware security auditing for Ethereum smart contracts[C]//2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 2018: 814-819.
5. Mossberg M, Manzano F, Hennenfent E, et al. Manticore: A user-friendly symbolic execution framework for binaries and smart contracts[C]//2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 2019: 1186-1189.
6. Torres C F, Schütte J, State R. Osiris: Hunting for integer bugs in Ethereum smart contracts[C]//Proceedings of the 34th Annual Computer Security Applications Conference. 2018: 664-676.
7. Torres C F , Baden M , Norvill R , et al. AEGIS: Shielding Vulnerable Smart Contracts Against Attacks[J]. 2020.
8. Rodler M, Li W, Karame G O, et al. Sereum: Protecting existing smart contracts against re-entrancy attacks[J]. arXiv preprint arXiv:1812.05934, 2018.
9. Jiang B, Liu Y, Chan W K. Contractfuzzer: Fuzzing smart contracts for vulnerability detection[C]//2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 2018: 259-269.
10. Ashizawa N, Yanai N, Cruz J P, et al. Eth2Vec: Learning contract-wide code representations for vulnerability detection on Ethereum smart contracts[C]//Proceedings of the 3rd ACM International Symposium on Blockchain and Secure Critical Infrastructure. 2021: 47-59.
11. Le Q, Mikolov T. Distributed representations of sentences and documents[C]//International conference on machine learning. PMLR, 2014: 1188-1196.
12. Momeni P, Wang Y, Samavi R. Machine learning model for smart contracts security analysis[C]//2019 17th International Conference on Privacy, Security and Trust (PST). IEEE, 2019: 1-6.
13. Wang W, Song J, Xu G, et al. Contractward: Automated vulnerability detection models for ethereum smart contracts[J]. IEEE Transactions on Network Science and Engineering, 2020.
14. Qian P, Liu Z, He Q, et al. Towards automated reentrancy detection for smart contracts based on sequential models[J]. IEEE Access, 2020, 8: 19685-19695.
15. Mikolov T , Chen K , Corrado G , et al. Efficient Estimation of Word Representations in Vector Space[J]. Computer Science, 2013.
16. Hochreiter S , Schmidhuber J . Long Short-Term Memory[J]. Neural Computation, 1997, 9(8):1735-1780.
17. Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.
18. Sun Y, Gu L. Attention-based machine learning model for smart contract vulnerability detection[C]//Journal of Physics: Conference Series. IOP Publishing, 2021, 1820(1): 012004.
19. Zhuang Y, Liu Z, Qian P, et al. Smart Contract Vulnerability Detection using Graph Neural Network[C]//IJCAI. 2020: 3283-3290.
20. Liu Z, Qian P, Wang X, et al. Combining graph neural networks with expert knowledge for smart contract vulnerability detection[J]. IEEE Transactions on Knowledge and Data Engineering, 2021.

Automatic Smart Contract Generation with Knowledge Extraction and Unified Modeling Language

Peiyun Ran[1], Mingsheng Liu[2⋆], Jianwu Zheng[3⋆], Zakirul Alam Bhuiyan[4], Jianhua Li[5], Gang Li[6], Shiyuan Yu[7], Lifeng Wang[8], Song Tang[9], and Peng Zhao[10]

[1] School of Cyberspace Security of Beihang University ,Beijing, China
rpy233@buaa.edu.cn
[2] Shijiazhuang Institute of Railway Technology, Shijiazhuang 050041 ,Hebei Province, China liums601001@sina.com
[3] Key Laboratory of Traffic Safety and Control of Hebei Province, China
School of Transportation, Shijiazhuang Tiedao University, Hebei Province, China
zhengjw@stdu.edu.cn
[4] Department of Computer and Information Sciences Fordham University JMH 334, E Fordham Road, Bronx, NY 10458 USA mbhuiyan3@fordham.edu
[5] School of Information Science and Technology, Shijiazhuang TieDao University, No.17 East North Second Ring Road, Changan District, Shijiazhuang, China
lijh128@163.com
[6] Zhongke Zidong Information Technology (Beijing) Co., Ltd ligang@people-ai.cn
[7] Beijing Public Security Burea, China 2778849841@qq.com
[8] Hebei Huaye Jike Information Technology Co., Ltd 904757835@qq.com
[9] Institute of Applied Mathematics of Hebei Academy of Sciences 574465982@qq.com
[10] The First Hospital of Hebei Medical University zhaopeng@jyyy.com.cn

Abstract. Since the launch of Ethereum in 2013, the smart contract has been a momentous part of the blockchain systems due to its character of automatic execution. The generation of smart contracts has also attracted extensive attention from the academic community. However, the preparation and generation of smart contracts are still mainly manual so far, which limits the scalability of the smart contracts. In this paper, we put forward a new method to generate smart contracts automatically based on knowledge extraction and Unified Modeling Language(UML), which can significantly accelerate the generation of smart contracts. We will describe this method in more detail based on the logistics supply chain.

Keywords: blockchain, smart contract, automatic code generation, knowledge extraction, unified modeling language

⋆ Corresponding authors

1 Introduction

Blockchain has attracted broad academic attention since its inception. With decentralized and tamper-proof characteristics, blockchain can establish a distributed ledger and help solve the trust issue. Cryptography technology is primarily used to ensure data security in a decentralized scenario, including, but not limited to, encryption, digital signatures, and hash functions. Nowadays blockchain technology has been widely applied in many scenarios such as finance, agriculture, the medical industry, logistics, and so on. The focus of research is often on improving its scalability[1].

In a blockchain system, smart contracts are common. Smart contract was first proposed in the 1990s. It is essentially a type of computer program that can be executed automatically when certain conditions are met[2]. Due to this attribute of smart contracts, they can be observed and executed by multiple untrusted nodes in a distributed system. The characteristics of blockchain are quite consistent with those of smart contracts. The smart contracts, combined with cryptography technology which ensures the difficulty of tampering or intervening in the content and execution of contracts, help to establish a set of guidelines for distrustful nodes to follow together.

Smart contracts have been successfully implemented in multiple blockchain systems such as Ethereum, Hyperledger Fabric, and so on. While making a splash in the blockchain, smart contracts are also faced with a series of problems, the most crucial among which is its generation. There are already some Turing complete languages, such as Go and Solidity that can be used for its generation. However, when it comes to smart contract coding, since the absence of a unified standard for developing decentralized solutions for business processes, the development of smart contracts, especially their automatic generation is challenging[3]. In addition, the tamper resistance of smart contracts brings difficulties to error correction. All these issues limit the scalability of smart contracts and the overall efficiency of the blockchain system. Therefore, researches on the automatic generation and security check of smart contracts have drawn more attention in recent years[4].

To fundamentally solve the aforementioned problem, we still need to start with the natural language (NL) contract. In this paper, we introduced natural language processing (NLP) technology to preprocess the original NL contracts and extract knowledge from them. Based on the result of knowledge extraction, we can further understand the various state transition processes experienced during the operation of smart contracts. In addition, the entities involved in the contract and the relationship between them will also be extracted. Using the extracted knowledge, we can first construct the smart contract template of the involved fields. Since the execution of smart contracts is essentially the running process of a finite state machine, the template will be presented in the form of a state diagram. Based on this template, the template will be further filled according to the specific contract content. When the state diagram template is populated, we can further convert the state diagram into a formal smart contract code. In [3], Smart contract code is proven to be generated from UML

state diagrams. In this paper, we use the UML modeling method to model and populate the smart contract template. Specifically, we first define all possible states of the contract, use these states to create a state diagram template, and then supplement the events that may occur in each state, especially those that may cause state transition, according to the various actions that the subject may take in each state.

We will take the application of blockchain in the logistics field as an example to show the process of automatically generating smart contracts according to our method. In the logistics industry, goods generally pass through the process of shipment, transportation and unloading inspection. Due to the relative lack of mandatory supervision, there may be many violations in this process. The enforceability of smart contracts helps to better monitor both parties and reduces the occurrence of defaults.

2 RELATED WORK

Automatic code generation is considered an effective method to improve the degree of automation and the final quality of software development and is widely used across academia and industry. At present, the technology of automatic code generation is mainly used in fields such as automatic code completion, and its purpose is to help programmers write programs.

In this paper, we aim to automatically generate smart contract codes according to NL contracts. This is automatic code generation based on function description, which is more challenging. In[5], a computer algorithm named Deepcoder is proposed, which can be used for automatic code writing. To generate codes, deep learning was used along with a graphical user interface (GUI). [6] takes advantage of UML and abstract syntax tree(AST) to generate JAVA codes. When it comes to smart contracts, [7] provides a method based on domain-specific ontologies and semantic rules to generate templates of smart contracts first, then insert constraints of the contracts by manipulating an AST. Moreover, [8] introduces the concept of controlled NL and uses it to rewrite the traditional NL contract, so that the computer can understand the contract content more accurately. In addition, this paper also proposes using a state diagram to describe the execution process of smart contracts. In [9], a code development platform is introduced to auto-generate and deploy the smart contract, which simplifies the deployment of a supply chain management (SCM) solution on the blockchain. The creator (the user who creates a supply chain) only needs to enter personal details and company or organization details through the user interface, and then submit product details, including its name and personalized attributes.

3 Main Technologies

3.1 Knowledge Extraction

Knowledge extraction, whose target is to extract knowledge from unstructured data from different sources and store it in the knowledge map, is an impor-

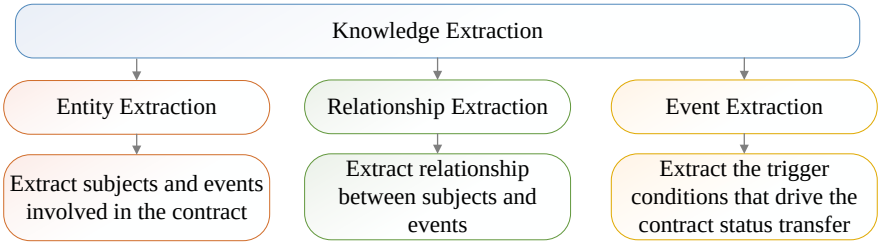


Fig. 1. Knowledge extraction classification and its role in processing NL contracts

tant technology to realize the automatic construction of large-scale knowledge atlases in NLP. Knowledge extraction mainly includes three sub-tasks: named entity recognition, relationship extraction, and event extraction. The goals of the three are to determine the entities in the text, determine the relationship between entities[10], detect the occurrence of specific types of events, and extract parameters respectively[11].At present, there are many methods for knowledge extraction, especially event extraction.The methods based on are mainly graph convolutional networks[12], heterogeneous information networks[13], reinforcement learning[14], and so on. There are similar methods for entity and relationship extraction using graph neural networks[15] and heterogeneous information networks[16].

As shown in Fig. 1, in our method, we mainly use entity extraction and event extraction to analyze contract participants and events that trigger contract terms. We also use relationship extraction to analyze the relationship between multiple entities that may be involved in the contract. After this step, the contract written in NL will be simplified and structured.

Contracts written in NL generally exist in an unstructured form, and contract terms will be converted into structured data after knowledge extraction. This has laid the foundation for the establishment and filling of the smart contract template. With these structured data, it will be easier to analyze the execution process of smart contracts. For example, the contract specifies the possible breach of contract and the punishment method. After event extraction, the breach of contract and the punishment method can be extracted completely. In addition, unnecessary and redundant words can be deleted, which will make the computer more accurate in understanding and implementing the contract terms. In addition, the probability of errors in the process of generating the contract state diagram and even the code will be reduced accordingly.

3.2 UML Class Diagrams Generation

UML class diagram is a powerful modeling tool. As a result, it is closely related to object-oriented (OO) programming and can help developers fully understand relevant modeling for businesses. UML class diagrams show a set of classes, in-

terfaces, collaborations, and their relationships. Before modeling with the UML class diagram, we must specify the business requirements. In the context of smart contracts, pre-written NL contracts are essential. We can start with NL contracts, first describe the requirements in NL, and then define the objects, classes, and methods involved in the contracts through object-oriented programming. Then we can model the implementation process of smart contracts on this basis.

Though mapping from user requirements to UML diagrams provides great convenience, it is difficult to complete the direct conversion from contract requirements written in NL to UML class diagrams, since almost all NLs contain redundancy, fuzziness, and imprecision. In addition, artificially prepared contracts may sometimes even have omissions, resulting in more serious consequences. Therefore, this research has always been a hot spot in the academic circle, especially in the field of NLP[17]. Some methods have been proposed to solve this difficulty. In[17], an approach which can maps user's requirements to UML class diagram based on the MDA is proposed. With the rule-based NLP approach, the processed text will be mapped into an XMI file first, then the XMI file will be converted to UML diagram by a CASE tool such as ArgoUMLFurther, [18] proposed a kind of heuristic rule based on NLP. They developed a set of pre-defined rules to extract OO concepts such as classes, attributes, methods, and relationships to generate a UML class diagram from the given requirements. The paper also referred to the reconstruction and normalization of NL text, whicprovided inspiration for our preprocessing.

There is one thing in common between [17] and [18]. Both of them reconstruct the NL contract based on certain semantic or grammatical rules and then construct the classes and objects in the UML class diagram. We do not subscribe to this idea. Instead, we use knowledge extraction to directly structure the contract content, which saves the trouble of defining semantic rules.

3.3 UML Class Diagrams to Smart Contract

There have been previous studies on knowledge graphs embedding[19]. As mentioned earlier, since the smart contract will execute itself after meeting the trigger conditions, we can regard it as a finite state machine, and the process of executing a smart contract as a state transition process when a finite state machine runs. Coincidentally, the UML class diagram is also a powerful tool to describe the running process of finite state machines. Therefore, it is not difficult to imagine that transforming UML class diagrams into smart contracts will be an effective way to automatically generate smart contracts.

In [3], the algorithm for model transformation and generation of executable smart contracts is outlined. This paper also compares their generated smart contract with the original contract, and the results are highly similar, which further illustrates the feasibility of transforming UML class diagram into smart contract. Besides, the modeling method for smart contract state diagram is not unique. In addition to UML class diagram, [9] used another modeling language named State Chart XML(SCXML). Although the modeling languages used are

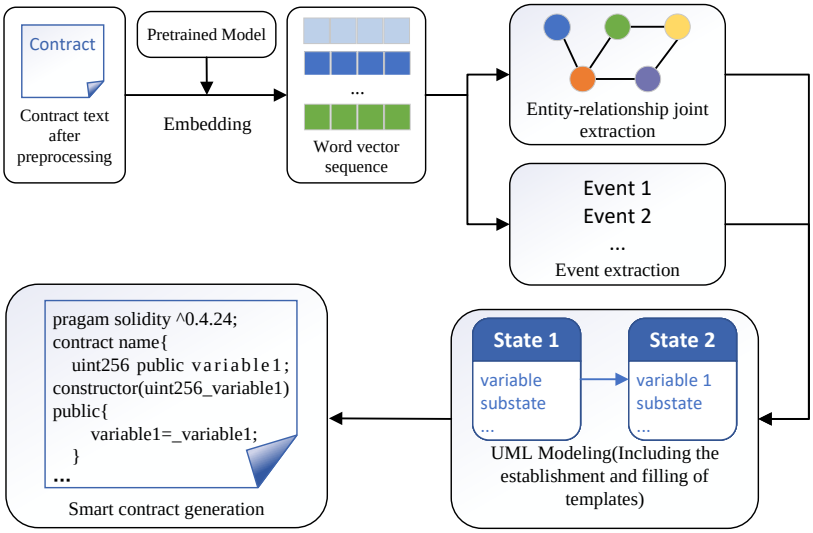


Fig. 2. Flow chart of the framework

different, these methods are based on the state diagram to realize the automatic generation of smart contracts. The method proposed in this paper is also based on this idea, using the state diagram constructed by UML to generate smart contracts.

4 Methods

In this section, we will introduce the whole process of automatically generating smart contracts from natural language contracts under our method. Fig. 2 gives a flow chart of the framework.

4.1 Preparation and Preprocessing of NL Contracts

Firstly, we need to Clarify the rights and obligations of all parties involved in the contract to avoid possible mistakes in natural language contracts. As previously stated, almost all the NLs have more or less redundancy, fuzziness, and imprecision. Therefore, it is necessary for us to preprocess the natural language contract after its content is determined. For example, we can observe the following two contract terms. One term is 'Party B will be fined if it releases false information and other acts that damage Party A's reputation'. The other term is 'Party A shall test the goods (iron ore) and compare them with the specified indicators. If the iron grade difference is greater than 1.5 and not greater than 2.5, 500 yuan/vehicle will be deducted each time. If the difference is greater than 2.5 and

not greater than 3.5, all the vehicle freight will be deducted'. It is straightforward that the former term's description is hard for computers to understand because it does not clearly define the so-called "false information" and "damage Party A's reputation". So, this kind of terms are not suitable for smart contract execution. In contrast, the latter term has a much clearer definition of the default of substandard goods, which is also much easier for computers to understand.

The above example illustrates the importance of the accurate description of contract terms to computer understanding of contracts. Therefore, in the process of preprocessing NL contracts, we need to focus on the regulations that are not fully and clearly described in the contract and modify or delete them so that the regulations can be fully understood by the computer.

4.2 Data Structured Representation

After preprocessing the NL contracts, we can further analyze the contract with the method of knowledge extraction. Specifically, we need to use entity extraction and relationship extraction to extract the parties involved in the contract (such as contract participants) and their relationships, and use event extraction to extract all events that may occur in the process of contract execution.

In practice, according to [10] and [20], we can jointly extract entities and their relationships. We will introduce a multi-round question and answer mechanism to "poll" the contract content, so as to obtain the parties involved in the contract and their relationships. Given that the smart contract application scenario is dominated by blockchain, which is a distributed ledger, the subjects involved can also be roughly considered as contract participants and their property or employees. Before multiple rounds of Q&A, we can set the question template in advance, and then fine-tune this template according to the changes in the application scenarios. For example, in the scenario of the logistics industry, in addition to the contract between Party A and Party B, the subjects involved also involve a series of entities such as the goods transported, the employees involved in the transportation, and the assets of the entrusting party, and their relationship networks need to be clarified without any omission. If we change scenarios, such as car leasing, then the entities we involve include not only the lessee and the leasing company, but also the information about related vehicles, the qualifications of the leasing company, the driver's license of the lessee, etc. Although the entities involved in the above two scenarios are different, the common parts can be summarized, such as the rights and obligations of Party A and Party B, both parties, certain items to be traded, and the qualifications or characteristics.

Similarly, inspired by [11] and [21] Event extraction can also take the form of similar multi-round question and answer. The only difference is that it focuses more on the process of contract execution. It needs to take into account every event in the contract execution process, and further consider the contract state transition when an event occurs. Therefore, the accuracy of NL contracts description directly determines the quality of event extraction results. In addition, when designing the template for multiple rounds of question and answer,

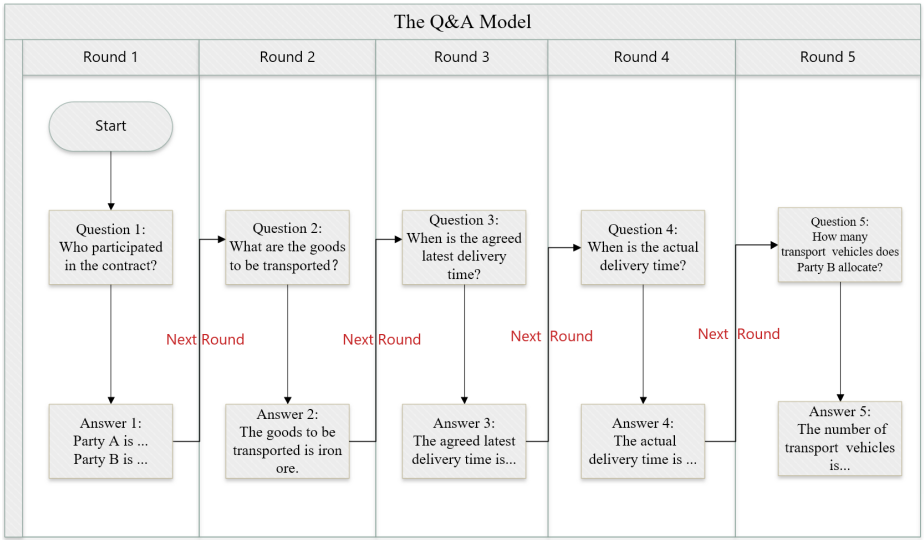


Fig. 3. QA for extracting the event in the loading stage of the logistics contract

we also need to focus on the state changes during the implementation of the smart contract. For each state, we need to ask about the possible behavior of the subject in the state and the state changes that may be caused by it. Finally, the results of multiple rounds of inquiry are output in a structured form of triple groups of subject, behavior, and state changes. Fig. 3 is an example of some questions and answers involved in the loading phase of the logistics contract for event extraction.

As for the generation of dialogue, we learned from the mode in [11], which is divided into two modules. One is to generate a question and answer based on the current contract statement, and the other is to generate the predictive answer based on the existing statement. In different scenarios, Module 1 will generate a question and answer dialogue template according to the characteristics of the scenario. Correspondingly, module 2 extracts parameters that may be used as responses on the basis of existing contract statements to fill in the template. Specifically, the filling process needs to use the statement embedding vector obtained in the previous stage to locate the parameters to be selected (such as the keywords used for answering). We can use the following probability formula to calculate the probability of the k-th word (P_k) being selected:[11]

$$P_k = \frac{exp(H \cdot WE_k)}{\sum_{i=1}^{N} exp(H \cdot WE_i)} \tag{1}$$

In the formula, WE refers to the final representation of the i-th word embedded in the sentence, H refers to the vector that maps a word vector to a scalar. H will change due to different scenarios.

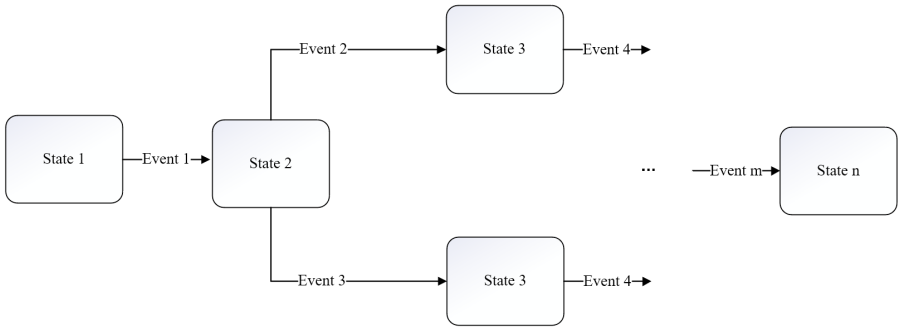


Fig. 4. A template of state diagram

4.3 Construction of Smart Contract State Diagram

With the help of the structured contract information obtained from the previous knowledge extraction, we have clarified the subject and state information involved in the smart contract. Next, we will build the state diagram of the smart contract. We will use a UML class diagram to describe the change of contract state. Therefore, in the process of UML modeling, the established objects will also focus on different states of the contract. The involved entity objects can perform certain activities in each state and cause corresponding state changes.

The construction of a smart contract state diagram based on structured data is mainly divided into two steps. First, it is necessary to define all the state changes experienced in the whole process of smart contract execution and the sequence of states, and then draw the general framework of the UML state diagram. The state graph drawn at this stage only contains the states experienced by the contract. Different nodes of the graph represent different states, and the edges point to the order of the surface states. This completes the construction of the state diagram template. Fig. 4 shows a template of a state diagram. The specific contents need to be further supplemented.

Then we will further fill in the state diagram template according to the entities involved in the contract and their possible behaviors in each phase. Specifically, for a certain subject involved in the contract, he may play different roles in different stages of contract execution and lead the execution of the contract to different directions. For example, for the goods that need to be transported in the logistics contract, they are in the state of being unloaded for transportation in the preparation stage before transportation. Whether they are delivered on time or not will lead the contract execution into two different states: "normal delivery, normal contract execution" and "default of the carrier, overdue delivery". In the transportation stage, whether the goods are in good condition during transportation also determines the trend of the contract status. If the goods are in good condition, the contract will be executed normally. If the goods are damaged or lost, it will be deemed that the carrier is in breach of contract. Similarly, in the unloading inspection stage, the quality inspection results of

the goods will also lead to the bifurcation of the contract status. Therefore, in the process of filling in the state diagram template, it is necessary to take into account all entities and their relationships, and consider all possible impacts of each entity on the contract state trend.

When all the structured data corresponding to the contract related subjects are filled into the UML state diagram template, the state diagram should cover the complete process of smart contract execution, including but not limited to the start state, various intermediate states, and end states. These states will exist as nodes in the diagram, and they will be directly or indirectly related by directed edges. Each connected edge corresponds to one or more events that cause state transitions. When all the structured data corresponding to the contract related subjects are filled into the UML state diagram template, the state diagram should cover the complete process of smart contract execution, including but not limited to the start state, various intermediate states, and end states. These states will exist as nodes in the diagram, and they will be directly or indirectly linked by directed edges. Each connected edge corresponds to one or more events that cause state transitions. It is worth mentioning that the starting state of each state diagram is unique, but the ending state may not be unique. This indicates that the execution of the contract may have different trends.

4.4 Transformation from State Diagram to Code

After the UML state diagram describing the smart contract is constructed, we can further convert it into smart contract code. Since UML class diagrams are closely related to object-oriented programming, UML class diagrams were used to generate JAVA code at an early stage[22][23], based on object-oriented methods. Their ideas are much the same, almost all of them use the state changes depicted by UML class diagrams to build the hierarchical state, concurrent state, and historical state of the corresponding code[24].

For smart contracts, the mainstream method is based on Model Driven Architecture(MDA). The MDA-based approach makes the development of smart contracts more systematic and helps to change the dilemma that there is no unified approach to smart contract development.[25].

In addition, there are tools that can directly translate UML state diagrams into the Solidity codes. For example, Eclipse has introduced an extension package called UML to Solidity. It contains a UML profile and a set of Acceleo code generators to model smart contracts in UML and generate Solidity code. Usable with the Papyrus UML modeler for Eclipse. In addition, the UML state diagram here is drawn using Papyrus, which is a UML 2 modeling tool based on the Eclipse platform and supports the UML 2 standard specified by the Object Management Group (OMG) and the second-generation Diagram Interchange (DI2) standard.

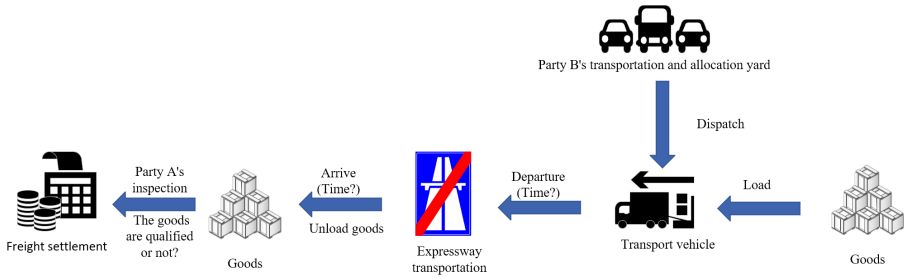


Fig. 5. A general process of logistics transportation

5 Example

In this section, we will give an example of automatic generation of smart contracts for logistics and transportation to explain our method in more detail. Fig. 5 illustrates the general process of logistics and transportation.

In a logistics contract, Party A and Party B are generally the shipper and carrier respectively. The contract will first include the basic information of both parties, such as company name, legal representative, address, and then the period of validity of the contract. Both Party A and Party B are participants, which can be considered two different objects of the same category. The validity period of the contract indicates the starting and ending time of the code.

When a logistics contract is formally implemented, it is generally considered that it can be divided into preparation stage, transportation stage, arrival inspection stage and freight settlement stage. In addition, a special default status should be set to deal with the default of both parties. Each stage can be considered as a state that the contract needs to pass through.

In the preparation stage, the goods (assumed to be iron ore) have not been loaded. In this state, goods and trucks are the main players. In addition, the latest departure time and the destination of goods is also involved, which can be used as a variable. If the goods are delivered on time, it will enter the next stage (transportation stage); otherwise, it will be treated as Party B's breach of contract and enter the default state.

During the transport stage, the main body involved is typically the goods being transported. It is the state of the goods during the transportation process that determines whether they enter the default state. In addition, weather, road conditions, and road fees need to be considered. These will all exist in the form of variables.

If the goods are delivered to the place specified by Party A on time, it will enter the goods inspection stage. At this stage, Party A will first weigh the received goods (iron ore) to check whether the weight is up to the standard. If it is up to standard, samples will be taken for testing, focusing on the difference between the weight of iron ore and the iron grade value and the standard value,

and determine whether freight is deducted according to the difference. If the difference is significantly more than a certain critical value, it will directly enter the default state. Specific weight standards and iron grade standards will be uploaded in the form of predefined parameters before the deployment of smart contracts.

Upon passing the quality inspection, the contract will enter the freight settlement phase. In the settlement status, Party A needs to settle the freight according to the tonnage of the goods arrived this time and the unit quality freight agreed by both parties in advance, and then pay the freight to Party B before the specified payment date. If Party A fails to pay when due, the smart contract will automatically execute the payment process and transfer the freight payable by Party A to Party B's account. The variables to be set at this stage are mainly freight per ton of goods, payment period of Party A, etc.

The above is the normal state experienced during the execution of the contract, while the abnormal state is the default state mentioned previously. The abnormal state will be divided into several sub states, each of which corresponds to the possible breach of contract at a certain stage of contract execution. For example, in the preparation stage, if the goods are not delivered on time, it will enter default sub state 1, which will be regarded as inheriting the sub class of default status, and the other sub states are the same. In each sub class, the handling methods for violating the corresponding provisions of the contract will be clearly specified. This will include but not limited to a certain amount of liquidated damages, fines, and even termination of the contract.

To date, the state refinement of the logistics contract has been completed. This process will be completed by knowledge extraction based on multiple rounds of questions and answers. It should be emphasized that data preprocessing before refining is essential. The entities and variables involved in the contract mentioned above must be explicitly written into the NL contract after data preprocessing. According to the structured data extracted from knowledge, we can generate the state diagram as shown in the figure. After the smart contract state diagram is determined, we can directly use the Eclipse plug-in to convert it into Solidity code. So far, the source code of the smart contract has been generated.

6 Conclusion

In this paper, we propose an automatic generation method of smart contracts based on NLP technology and UML. Based on the knowledge extraction technology in NLP, we used multi-round dialogue to extract the entity relationship joint extraction and event extraction of the preprocessed NL contract, used the obtained structured contract data for UML state diagram modeling, and finally transformed the UML state diagram into a smart contract.

The method of knowledge extraction can eliminate the relatively complex definition of semantic rules in the process of transformation from natural language to UML class diagram, and the use of UML, a unified language, to model smart contract state diagram has a similar effect, without the need to define ad-

ditional domain-specific languages for each application scenario. Therefore, we have reason to believe that the automatic generation of smart contracts based on these two technologies will further shorten the development cycle of smart contracts. This will improve the scalability of smart contracts based on existing methods.

Acknowledgements This work was supported by the S&T Program of Hebei through grant 20310101D.

References

1. Lin Liu, Wei-Tek Tsai, Mingsheng Liu, Ze Gao, Shumei Ji, Hao Peng, and Zakirul Alam Bhuiyan. A recursive reinforced blockchain performance evaluation and improvement architecture: Maximising diversity to improve scalability. *International Journal of Communication Systems*, page e5315, 2022.
2. Zibin Zheng, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang. Blockchain challenges and opportunities: A survey. *International journal of web and grid services*, 14(4):352–375, 2018.
3. Mantas Jurgelaitis, Lina čeponienė, and Rita Butkienė. Solidity code generation from uml state machines in model-driven smart contract development. *IEEE Access*, 10:33465–33481, 2022.
4. Lin Liu, Wei-Tek Tsai, Md Zakirul Alam Bhuiyan, Hao Peng, and Mingsheng Liu. Blockchain-enabled fraud discovery through abnormal smart contract detection on ethereum. *Future Generation Computer Systems*, 128:158–166, 2022.
5. Matej Balog, Alexander L Gaunt, Marc Brockschmidt, Sebastian Nowozin, and Daniel Tarlow. Deepcoder: Learning to write programs. *arXiv preprint arXiv:1611.01989*, 2016.
6. Arun Veeramani, Kausik Venkatesan, and K Nalinadevi. Abstract syntax tree based unified modeling language to object oriented code conversion. In *Proceedings of the 2014 International Conference on Interdisciplinary Advances in Applied Computing*, pages 1–8, 2014.
7. Olivia Choudhury, Nolan Rudolph, Issa Sylla, Noor Fairoza, and Amar Das. Auto-generation of smart contracts from domain-specific ontologies and semantic rules. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 963–970. IEEE, 2018.
8. Takaaki Tateishi, Sachiko Yoshihama, Naoto Sato, and Shin Saito. Automatic smart contract generation using controlled natural language and template. *IBM Journal of Research and Development*, 63(2/3):6–1, 2019.
9. Khushi Asawa, Shubham Kukreja, and Rishi Gondkar. An ncdp for developing a blockchain based dynamic supply chain management with auto-generation of smart contract. In *2021 26th International Conference on Automation and Computing (ICAC)*, pages 1–6. IEEE, 2021.
10. Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. Entity-relation extraction as multi-turn question answering. *arXiv preprint arXiv:1905.05529*, 2019.

11. Qian Li, Hao Peng, Jianxin Li, Jia Wu, Yuanxing Ning, Lihong Wang, S Yu Philip, and Zheng Wang. Reinforcement learning-based dialogue guided event extraction to exploit argument relations. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:520–533, 2021.
12. Hao Peng, Jianxin Li, Qiran Gong, Yangqiu Song, Yuanxing Ning, Kunfeng Lai, and Philip S Yu. Fine-grained event categorization with heterogeneous graph convolutional networks. *arXiv preprint arXiv:1906.04580*, 2019.
13. Hao Peng, Jianxin Li, Yangqiu Song, Renyu Yang, Rajiv Ranjan, Philip S. Yu, and Lifang He. Streaming social event detection and evolution discovery in heterogeneous information networks. *ACM Trans. Knowl. Discov. Data*, 15(5), may 2021.
14. Hao Peng, Ruitong Zhang, Shaoning Li, Yuwei Cao, Shirui Pan, and Philip Yu. Reinforced, incremental and cross-lingual event detection from social messages. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.
15. Hao Peng, Ruitong Zhang, Yingtong Dou, Renyu Yang, Jingyi Zhang, and Philip S Yu. Reinforced neighborhood selection guided multi-relational graph neural networks. *ACM Transactions on Information Systems (TOIS)*, 40(4):1–46, 2021.
16. Hao Peng, Renyu Yang, Zheng Wang, Jianxin Li, Lifang He, S Yu Philip, Albert Y Zomaya, and Rajiv Ranjan. Lime: Low-cost and incremental learning for dynamic heterogeneous information networks. *IEEE Transactions on Computers*, 71(3):628–642, 2021.
17. Wahiba Ben Abdessalem Karaa, Zeineb Ben Azzouz, Aarti Singh, Nilanjan Dey, Amira S. Ashour, and Henda Ben Ghazala. Automatic builder of class diagram (abcd): an application of uml generation from functional requirements. *Software: Practice and Experience*, 46(11):1443–1458, 2016.
18. Esra A. Abdelnabi, Abdelsalam M. Maatuk, Tawfig M. Abdelaziz, and Salwa M. Elakeili. Generating uml class diagram using nlp techniques and heuristic rules. In *2020 20th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)*, pages 277–282, 2020.
19. Hao Peng, Haoran Li, Yangqiu Song, Vincent Zheng, and Jianxin Li. Differentially private federated knowledge graphs embedding. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 1416–1425, 2021.
20. Markus Eberts and Adrian Ulges. Span-based joint entity and relation extraction with transformer pre-training. *arXiv preprint arXiv:1909.07755*, 2019.
21. Xinya Du and Claire Cardie. Event extraction by answering (almost) natural questions. *arXiv preprint arXiv:2004.13625*, 2020.
22. Iftikhar Azim Niaz, Jiro Tanaka, et al. Mapping uml statecharts to java code. In *IASTED Conf. on Software Engineering*, pages 111–116, 2004.
23. Muhammad Usman and Aamer Nadeem. Automatic generation of java code from uml diagrams using ujector. *International Journal of Software Engineering and Its Applications*, 3(2):21–37, 2009.
24. Sunitha E. V. and Philip Samuel. Automatic code generation from uml state chart diagrams. *IEEE Access*, 7:8591–8608, 2019.
25. Kai Hu, Jian Zhu, Yi Ding, Xiaomin Bai, and Jiehua Huang. Smart contract engineering. *Electronics*, 9(12):2042, 2020.

Blockchain Scalability Technologies

Nengxiang Xu[1,2],Jiahong Cai[1,2], Yinyan Gong[1,2], Huan Zhang[1,2], Weihong Huang[1,2,*], Kuan-ching Li[1,2]

[1] School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China
[2] Hunan Key Laboratory for Service computing and Novel Software Technology, Xiangtan 411201, China
1113482768@qq.com,jiahongcai@mail.hnust.edu.cn,G18873530267@163.com,2820558906@qq.com,whhuang@hnust.edu.cn,rob5li@outlook.com

Abstract. As the underlying implementation technology of the current main-stream digital currency, blockchain can establish a trusted distributed system without relying on third-party trusted institutions or a privacy-protect system. The decentralized characteristics of blockchain have broad application scenarios, such as the Internet of Things, financial technology and other fields. However, the scalability of the current blockchain is seriously insufficient, such as limited throughput in performance, small storage capacity, and difficulty in scaling functions. This paper introduces the definition and technical classification of scalability, analyzes the current problems of scalability and briefly introduces the current main-stream expandable technologies such as sharding, on-chain and off-chain storage, off-chain payment channel and cross-chain technology from two aspects of performance and function, as well as the principles and ideas of these technologies. Finally, the research progress of current blockchain extension technology is summarized, and the problems faced by the current extension scheme are pointed out, which provides a direction for future research work.

Keywords: Blockchain, Cross-chain, Off-chain, Scalability, Sharding Introduction.

1 Introduction

In 2008, Nakamoto published a technical white paper [1], Bitcoin: ``a point-to-point electronic cash payment system``. The Bitcoin system first proposed and implemented a decentralized digital payment system that uses blockchain technology as the underlying support technology and does not rely on trusted third parties in an open network [2]. The security trust model in Bitcoin system is greatly different from the existing commercial digital payment system [3]. In a Bitcoin system, trust between users comes from trust in the entire payment system [4]. The continuous development of blockchain has brought more possibilities in many areas. However, the development of blockchain technology is still subject to the trilemma [5]. The throughput of Bitcoin is 7 transactions per second [6], and Ethereum is 10-20 transactions per second. Scalability has become the biggest bottleneck for blockchain applications [7], as shown in Fig.1. Aiming at the problems faced by blockchain scalability, this paper introduces the current

mainstream scalability technologies in Section 3, analyzes their characteristics and compares them. In Section 4, the technical bottlenecks of these problems are analyzed, and the future research directions are pointed out.

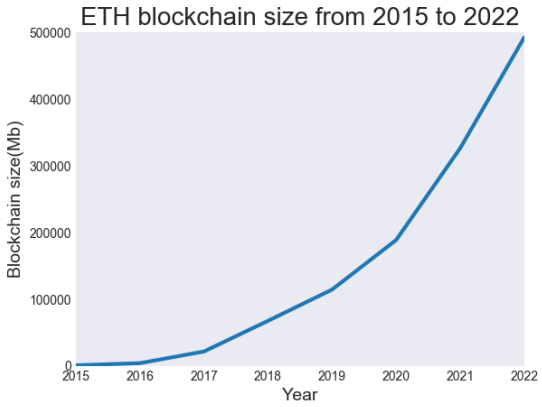


Fig.1. ETH block size from 2015 to 2022 in Mb.

2 Correlation technique

With the development of computer hardware [8-10], networks [11-13], and new algorithms [14-16], large amounts of data [17-19] can be generated in a short period. Data mining and machine learning [20-22] techniques had been applied in various datasets. However, the concerns about privacy and security [23-25] become a big challenge. Blockchain is a promising tool for solving this problem [26-28]. The current mainstream methods for improving blockchain scalability contains two aspects.

2.1 Sharding technology

In the blockchain system based on sharding technology, the formation of a committee for transaction verification and consensus has reduced the storage burden of nodes to a certain extent and achieved the effect of scalability.

2.2 Payment channel off chain

Spilman proposed the micro-payment channel protocol [29], which establishes one-way channel cumulative transactions for small high-frequency transactions in the blockchain without the need for blockchain consensus.

3 Research statue

This chapter will begin to introduce several blockchain extension methods mentioned above.

3.1 Scalable performance

Performance-scalable methods include sharding and off-chain payment channels. The sharding mechanism was first applied to traditional distributed databases,and Elastico [30] and Zilliqa innovatively applied this technology to the consensus mechanism of the blockchain. Several typical sharding methods and offline payment channels will be introduced below.

Elastico. The core idea of the Elastico sharding protocol is to divide the nodes in the blockchain network into committees. Each committee has the same number of members. The disjoint transaction subsets in the blockchain are assigned to the committee and the PBFT consensus protocol is independently performed internally. Elastico defines an intra-fragment consensus process as an Epoch, in which the agreement is carried out in five steps, as shown in Fig.2. There are several key points in the Elastico protocol. First, in the authentication and confirmation committee grouping phase, each processor now locally selects their own identity information group(IP,PK) [31], representing their IP address and public key, respectively [7]. In order for the network to accept their identity, each processor must also find a PoW solution(nonce), which must correspond to their own chosen identity, and everyone in the system can verify the nonce. Using this method can effectively avoid sybil attack. But this is not enough; to prevent the node from calculating the nonce value in advance and then submitting it in the current era, the system also introduces an epochRandomness variable as a random source for an era. This random source was published after the last step of the previous era was generated. The Nonce value calculated by the node needs to meet the following conditions [30].

$$H(epochRandomness|PK||IP|nonce) \leq 2^{(r-D)} \quad (1)$$

Among them, D is a pre-set workload proof difficulty, r refers to the number of output bits generated by calculating the hash value H, and the final calculated hash value will also be used as the ID of the node. The last s bits of the node ID will determine the committee grouping of the node. When assigning the committee, if each member independently confirms the nodes in the committee through broadcast, the allocated message overhead $O(n^2)$, Elastico establishes a directory committee to determine the committee grouping results, and broadcasts them to each node, so that the message overhead is controlled at $O(n^c)$, where c is the number of committee nodes. The directory committee consists of the first c nodes that first calculate the nonce value.

In the consensus phase, the committee group conducts partial consensus through the PBFT protocol [32]. The final consensus is carried out by the final committee established by Elastico. The final committee is randomly selected and responsible for broadcasting the consensus results to the blockchain network nodes and calculating the next random source to generate new nonce values.

As the first sharding scheme, Elastico provides many guiding ideas for the next research. In terms of scalability, the Elastico scheme greatly improves the blockchain throughput, which is almost linear growth. However, Elastico also has some defects. For example, the PBFT protocol used by the consensus within the committee has too

many communication rounds. It is a protocol with extremely high communication complexity, so the protocol delay is as high as 100 seconds. At the same time every time after an era of redistribution committee also brought a lot of time consumption.

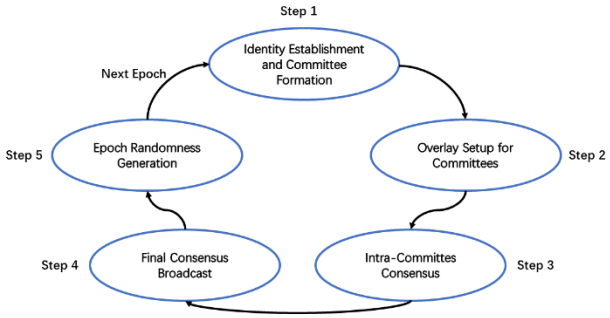


Fig.2. Elastico workflow.

OmniLedger. OmniLedger [33] was proposed by Philipp et al. The OmniLedger protocol improves the sharding tendency of the Elastico protocol and the lack of cross-shard transaction guarantees, while increasing transaction throughput. First, the Elastico sharding mechanism has a nonce value calculation to determine the grouping. Even if the random source is announced after the end of an era, it still cannot avoid the aggregation of malicious nodes. OmniLedger determines the location of the slice based on the randomly generated rndn, and rndn uses the unbiased random number generation scheme RandHound, which avoids the centralization problem caused by third-party participation, and also solves the tendency sharding problem caused by calculating the nonce value, as shown in Fig.3. RandHound requires a Leader to generate unbiased random numbers, which is determined by the VRF-based leader election algorithm. In the consensus algorithm [34], OmniLedger and Elastico protocol use the same PBFT consensus algorithm. Not the same with Elastico, OmniLedger divides the fragment into a sub-chain, and then uses the PBFT consensus on the fragment sub-chain. OmniLedger named this consensus algorithm ByzCoinX.

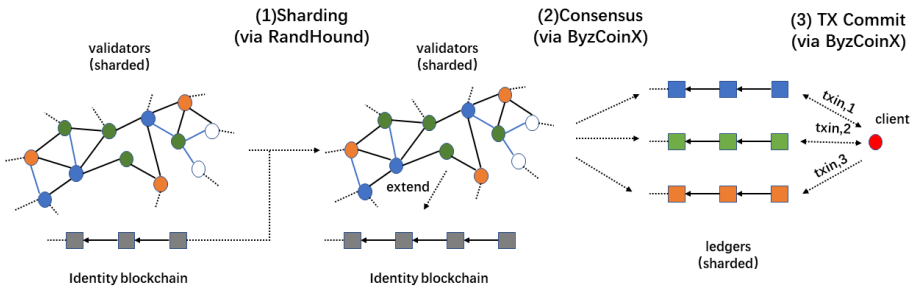


Fig.3. OmniLeder sharding structure.

In dealing with cross-shard transactions, OmniLedger is completed by the user(client). If it is necessary to transfer the unspent transaction output(UTXO) of different fragments to other fragments, the input fragments(ISs) only need to provide proof, and

the user locks these fragments. If all the input fragments provide proof, the user will send the proof to the output fragments(OSs). As long as one of the fragments rejects the transaction, unlock all the fragments and return these unspent transaction outputs. OmniLedger's cross-slice transaction processing method does not require communication between slices. The disadvantage is that users may be lazy in some application scenarios, resulting in the infeasibility of cross-slice transactions, as shown in Fig.4.

For small transactions in slices, OmniLedger adopts the 'trust-but-verify' authentication mechanism. When small transactions accumulate to a certain amount, a small number of nodes perform the first verification, and then the other part of the node performs the second verification. The two-layer verification ensures that malicious behavior can be detected in a timely manner, and users can choose to perform only one verification. The verification method of cumulative confirmation improves the transaction throughput to a certain extent. Table 1. compares Elastico with OmniLedger from three different aspects.

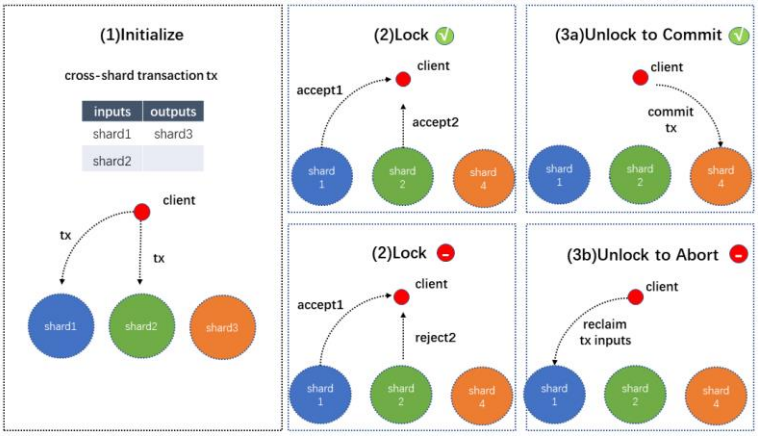


Fig.4. Cross-chain transaction progress of OmniLedger.

Table 1. Compare between Elastico and OmniLedger.

Approach	Sharding method	Consensus protocol	Efficiency
Elastico	Pow solution	PBFT	Depend on PBFT and duration of an epoch
OmniLedger	RandHound	ByzCoinX	Depend on ByzCoinx

Off-chain payment network. The payment channel network deals with high-frequency small transactions through off-chain processing, which indirectly improves the transaction throughput of the blockchain system [35]. Among them, the blockchain plays the role of an arbitration platform. In the process of channel payment, if the two

parties disagree on the status of the channel, the perpetrator can be punished by triggering a penalty transaction(the penalty transaction will be confirmed on the blockchain). The payment network under the chain mainly includes two protocols, payment channel protocol and cross channel payment protocol [36]. The payment channel protocol implements off-chain payment, which does not need to wait for confirmation from the blockchain. The cross-channel payment protocol is used to implement transactions between channels [37].

A typical implementation is the lighting network [38], which consists of two protocols: RSMC (*Recoverable Sequence Maturity Contract*) and HTLC (*Hashed Timelock Contract*). In the Lightning network system, RSMC implements a two-way payment function for both parties in the channel [39], that is, either party participating in the channel can send a transaction to the other party through the payment channel without the transaction being linked and confirm it immediately [40]. HTLC implements the function of cross-channel payment, that is, any two indirectly connected nodes can

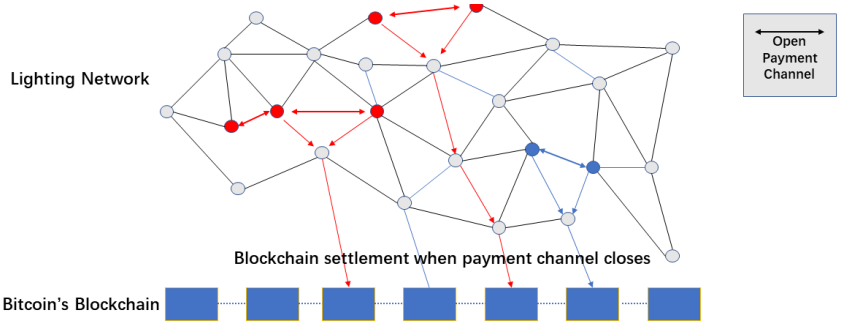


Fig.5. Payment path of lighting network.

transfer money through a payment channel connecting them. RMSC uses the timelock mechanism to make it take longer for one party to retrieve the assets in the channel than the other party, and introduces a penalty mechanism to suppress the possible malicious behavior of any party in the channel, as shown in Fig.5.

3.2 Function-expandable

At present, blockchain also has shortcomings in function scalability. At present, there is a lack of credible data exchange mechanism between different blockchain systems, which makes it impossible to share information between different blockchain systems, so as to realize more complex commercial applications. This paper introduces several mainstream cross-chain technologies.

Notary model. Notary Schemes is a cross-chain technology that is easy to maintain and scale. It mainly realizes data collection, transaction confirmation and verification by electing notaries between blockchains. The main representative program is Ripple [41].

Sidechain technology. In 2014, BlockStream proposed sidechain technology [42] to solve the scalability problem of Bitcoin system. It is a blockchain system independent of the main chain, using bidirectional anchoring technology to make the side chain attached to the main chain. The bidirectional anchoring performs the internal exchange of assets between the main chain and the side chain at a preset rate. The side chain technology uses the managed mode or the SPV mode to act as the connecting party of the main chain and the side chain. The managed mode is divided into two types: single managed and alliance managed.

3.3 Other storage solutions

The literature [43] proposes a blockchain light node verification protocol FlyClient. To reduce the number of blocks downloaded by light-node clients during a validation transaction, FlyClient uses a probabilistic validation mechanism to download the blockhead validation chain for log(n) (n is the number of blocks). If the blockchain is validated as a valid chain, the light-node client only needs to store one block information to verify that the blockchain has the transactions it looks for.

The literature [44] proposes a lightweight node ESPV. Because the request for new blocks in the blockchain system is large and the request for old blocks is small, ESPV adopts a fully redundant storage strategy to store new blocks. The new block is updated using the sliding window mechanism [45]. When a new block is generated, the last block in the window is deleted and the new block is added to the window.

The literature [46] proposes a layered edge cloud blockchain structure LayerChain. LayerChain stores blockchain data from IIoT (*Industry Internet of things*) [47] devices in a three-tier architecture of cloud blockchain layer [48], edge blockchain layer, and IIoT device layer [49].

4 Summarized and prospected

This section will point out the challenges faced by these methods and provide some possible directions for future research work.

4.1 Challenges for scalable programs

Efficiency. Under the chain payment network, the side chain technology performs transaction processing and data storage outside the blockchain. When data exchange with the blockchain is needed, the efficiency is not high [50].

Security. The Elastico sharding protocol uses the method of calculating the nonce value to confirm the committee grouping in the formation committee stage, so it is very likely that malicious nodes will gather [51]. Light nodes still rely on transaction data provided by all nodes when conducting transaction verification, which may also cause security problems.

4.2 Future research direction

Research on efficient data query model. In the side chain technology, the data sharing between the side chain and the main chain will cause a relatively large burden on the system. The transactions submission in the payment channel under the chain also needs to transmit data with the block chain. Therefore, the next step can be to study efficient data query mode.

Research on secure data protection mechanism. In the current blockchain technology research, security is still a very important direction. For example, in the Elastico sharding protocol, how the formation of the committee avoids the aggregation of malicious nodes. Therefore, the next step is to study secure data protection mechanisms [52].

5 Conclusion

This paper introduced some current technologies to improve the scalability of blockchain from two aspects: performance and function. These technologies have different emphases on scalability, security and decentralization. Finally, the challenges faced by the current blockchain scalability technology were analyzed, and the future direction was prospected.

Reference

1. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008. https://bitcoin.org/bitcoin.pdf
2. Yin Huang, Wei Liang, Jing Long, Jianbo Xu, and Kuan-Ching Li. A novel identity authentication for FPGA based IP designs. 17th IEEE TrustCom/BigDataSE, pp.1531–1536, 2018.
3. F. Xia, R. Hao, et al., Adaptive GTS allocation in IEEE 802.15. 4 for real-time wireless sensor networks, Journal of Systems Architecture 59 (10), 1231-1242, 2013.
4. Wei Liang, Yang Yang, Ce Yang, et al.: A consortium blockchain-based privacy protection scheme for personal data. IEEE Transactions on Reliability, pages 1–13, 2022.
5. Keke Gai, Ziyue Hu, Liehuang Zhu, et al.. Blockchain meets dag: A blockdag consensus mechanism. Conf. on Algor. and Arch. for Parallel Processing, pages 110–125, 2020.
6. Andreas M Antonopoulos. Mastering Bitcoin: unlocking digital cryptocurrencies. ” O'Reilly Media, Inc.”, 2014.
7. Zisang Xu, Wei Liang, Kuan-Ching Li, Jianbo Xu, Albert Y. Zomaya, and Jixin Zhang. A time-sensitive token-based anonymous authentication and dynamic group key agreement scheme for industry 5.0. IEEE TII, 18(10):7118–7127, 2022.
8. M. Qiu, Z. Jia, et al., "Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP", JSPS, 2007
9. M. Qiu, H. Li, E. Sha, "Heterogeneous real-time embedded software optimization considering hardware platform", ACM sym. on Applied Comp., 1637-1641, 2009
10. M. Qiu, C. Xue, Z. Shao, E. Sha, “Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems,” IEEE DATE Conf., 1-6, 2007

11. J. Niu, Y. Gao, et al., "Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability", JPDC, 72(12), 1565-1575, 2012
12. M. Qiu, Z. Chen, Z. Ming, X. Qin, J. Niu, "Energy-aware data allocation with hybrid memory for mobile cloud systems", IEEE Systems J., 11 (2), 813-822, 2014
13. M. Qiu, J. Liu, J. Li, et al., "A novel energy-aware fault tolerance mechanism for wireless sensor networks", IEEE/ACM Int'l Conf. on GCC. 2011
14. Z. Shao, M. Wang, et al., "Real-time dynamic voltage loop scheduling for multi-core embedded systems", IEEE Trans. on Circuits and Systems II, 54 (5), 445-449, 2007
15. M. Qiu, C Xue, Z Shao, et al., "Efficient algorithm of energy minimization for heterogeneous wireless sensor network", IEEE EUC, 25-34, 2006
16. M. Qiu, L. Yang, et al., "Dynamic and leakage energy minimization with soft real-time loop scheduling and voltage assignment", IEEE TVLSI, 18 (3), 501-504, 2009
17. J. Li, Z. Ming, et al., "Resource allocation robustness in multi-core embedded systems with inaccurate information", Journal of Systems Architecture 57 (9), 840-849, 2011
18. M. Qiu, E. Khisamutdinov, et al., "RNA nanotechnology for computer design and in vivo computation," Philosophical Transactions of the Royal Society A, 2013.
19. M. Qiu, E. Sha, M. Liu, et al., "Energy minimization with loop fusion and multi-functional-unit scheduling for multidimensional DSP", JPDC, 68 (4), 443-455, 2008
20. H. Qiu, Q. Zheng, et al., "Topological graph convolutional network-based urban traffic flow and density prediction", IEEE Trans. on ITS, 2020
21. M. Qiu and H. Qiu, "Review on Image Processing Based Adversarial Example Defenses in Computer Vision," IEEE 6th BigDataSecurity, pp. 94-99, Baltimore, MD, USA, 2020.
22. F. Hu, S. Lakdawala, et al., Low-power, intelligent sensor hardware interface for medical data preprocessing, IEEE Trans. on Info. Tech. in Biomedicine 13 (4), 656-663, 2009
23. H. Qiu, T. Dong, et al., "Adversarial attacks against network intrusion detection in IoT systems," IEEE Internet of Things Journal 8(13), 10327-10335, 2020
24. Y. Li, K. Gai, et al., "Intercrossed access controls for secure financial services on multimedia big data in cloud systems", ACM Trans. on Multimedia Comp., Comm., and App., 2016
25. K. Gai, M. Qiu, S. Elnagdy, "A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance," IEEE BigDataSecurity 2016
26. M. Qiu, H. Qiu, et al., "Secure Data Sharing Through Untrusted Clouds with Blockchain-enabled Key Management", the 3rd SmartBlock 2020, pp. 11-16, Oct. 2020,, China.
27. K. Gai, Y. Zhang, et al., "Blockchain-enabled Service Optimizations in Supply Chain Digital Twin", IEEE Transactions on Service Computing, 2022
28. X. Gao and M. Qiu, "Energy-Based Learning for Preventing Backdoor Attack". KSEM (3) 2022: 706-721
29. Eric Chan and Mohsen Lesani. Brief announcement: Brokering with hashed timelock contracts is np-hard. ACM PODC, Virtual Event, Italy, pages 199– 202. ACM, 2021.
30. Loi Luu, Viswesh Narayanan, Chaodong Zheng, Kunal Baweja, Seth Gilbert, and Prateek Saxena. A secure sharding protocol for open blockchains, ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, pages 17–30. ACM, 2016.
31. Wei Liang, Songyou Xie, Xiong Li, Jing Long, Yong Xie, and Kuan-Ching Li. A novel lightweight puf-based rfid mutual authentication protocol. 2017.
32. Feng Xia, Ruonan Hao, Jie Li, Naixue Xiong, Laurence T. Yang, and Yan Zhang. Adaptive GTS allocation in IEEE 802.15.4 for real-time wireless sensor networks. Journal of Systems Architecture, 59(10, Part D):1231–1242, 2013.
33. Eleftherios Kokoris-Kogias, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ewa Syta, and Bryan Ford. Omniledger: A secure, scale-out, decentralized ledger via sharding. IEEE Symposium on Security and Privacy, SP 2018, pages 583–598. 2018.

34. Hongju Cheng, Zhe Xie, et al. Multi-step data prediction in wireless sensor networks based on one-dimensional cnn and bidirectional lstm. IEEE Access, 7:117883–117896, 2019.
35. H. Cheng, Z. Xie, et al., Multi-step data prediction in wireless sensor networks based on one-dimensional CNN and bidirectional LSTM, IEEE Access 7, 117883-117896, 2019.
36. Y. Chen, L. Zhou, et al., KNN-BLOCK DBSCAN: Fast clustering for large-scale data, IEEE transactions on systems, man, and cybernetics: systems 51 (6), 3939-3953, 2019.
37. Chen Pan. Research on blockchain scalability technology. Master's thesis, Shanghaijiaotong University, 2019.
38. Z. Fang, K. Gai, et al.. LNBFSM: A food safety management system using blockchain and lightning network. Conf, Algo. and Arch. for Parallel Proc., pages 19–34, 2020. Springer
39. J. Zhao, J. Huang, et al., An effective exponential-based trust and reputation evaluation system in wireless sensor networks, IEEE Access 7, 33859-33869, 2019.
40. Y. Yao, N. Xiong, et al., Privacy-preserving max/min query in two-tiered wireless sensor networks, Computers & Mathematics with Applications 65 (9), 1318-1325, 2013.
41. Frederik Armknecht, Ghassan O. Karame, et al,, Overview and outlook. - 8th Int'l Conf. Trust and Trustworthy Computing, volume 9229, pages 163–180. Springer, 2015.
42. Adam Back, Matt Corallo, Luke Dashjr, et al., Enabling blockchain innovations with pegged sidechains. 2014.
43. Benedikt B¨unz, Lucianna Kiffer, et al., Super-light clients for cryptocurrencies. In 2020 IEEE Symposium on Security and Privacy, SP 2020, pages 928–946. 2020.
44. Yulong Zhao, Baoning Niu, Peng Li, and Xing Fan. A novel enhanced lightweight node for blockchain. 1st Int'l Conf, BlockSys, China, volume 1156, pages 137–149. Springer, 2019.
45. A. Fu, X. Zhang, et al., VFL: a verifiable federated learning with privacy-preserving for big data in industrial IoT, IEEE Transactions on Industrial Informatics, 2020.
46. Yao Yu, Shumei Liu, et al.: A hierarchical edge-cloud blockchain for large-scale low-delay industrial internet of things applications. IEEE TII, 17(7):5077–5086, 2021.
47. Yue Zhang, Keke Gai, et al., Blockchain-empowered efficient data sharing in internet of things settings. IEEE Journal on Selected Areas in Communications, pages 1–1, 2022.
48. Keke Gai, Jinnan Guo, Liehuang Zhu, and Shui Yu. Blockchain meets cloud computing: A survey. IEEE Communications Surveys & Tutorials, 22:2009–2030, 2020.
49. Jing Wang, Wei Luo, Wei Liang, et al.. Locally minimum storage regenerating codes in distributed cloud storage systems. China Communications, 14(11):82–91, 2017.
50. Hafid, Abdelatif et al. “Scaling Blockchains: A Comprehensive Survey.” IEEE Access 8 (2020): 125244-125262.
51. P Kumar, R Kumar, et al., PPSF: a privacy-preserving and secure framework using blockchain-based machine-learning for IoT-driven smart cities, IEEE Transactions on Network Science and Engineering 8 (3), 2326-2341, 2021.
52. Feng Gao, Liehuang Zhu, Keke Gai, Can Zhang, and Sheng Liu. Achieving a covert channel over an open blockchain network. IEEE Network, 34(2):6–13, 2020.

Blockchain Applications in Smart City: A Survey

Shuo Wang[12], Zhiqi Lei[12], Zijun Wang[1], Dongjue Wang[3], Mohan Wang[4], Gangqiang Yang[5], Keke Gai[12(✉)]

[1] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China
{3220215214,3220211023,3120221290,gaikeke}@bit.edu.cn
[2] Yangtze Delta Region Academy of Beijing Institute of Technology, Jiaxing, Zhejiang, China
[3] School of Cyber Engineering, Xidian University, Xi'an 710071, China
wdjxxcy@163.com
[4] Faculty of Information Science and Engineering, Ocean University of China, Qingdao 266400, China
wmh1561004@163.com
[5] School of Information Science and Engineering, Shandong University, Qingdao, Shandong, 266237, China
g37yang@sdu.edu.cn

Abstract. Smart cities bring new ideas to solve the social, economic, and environmental problems existing in traditional cities. However, the services of smart cities suffer from centralized data storage and untrustworthiness. As a decentralized distributed database, blockchain provides distributed and trusted infrastructure technology for the construction of smart cities. In this survey, we conduct a brief survey of the literature on blockchain applications in smart cities. We review the application of blockchain in smart cities from the four main application scenarios of energy, transportation, medical care, and manufacturing. Then, we discuss the realization of blockchain-based smart cities from the perspectives of privacy and storage. We believe this survey provides new ideas for the development of smart cities.

Keywords: Blockchain · Smart City · Smart Contract · Distributed Storage

1 Introduction

With the continuous increase of urban population, the medical level, convenient transportation, education level and living environment have been dramatically improved in cities. A large number of citizens are mitigating to bigger-sized cities for seeking a better living. However, a large growth of urban population also has brought many issues, for example, the limited energy resource, traffic congestion, air pollution, lack of medical resource, and manufacturing shortage. Constructing a digital transformation towards smart city, i.e., integrating Information Technology (IT)-based solutions with city infrastructure, is an alternative solution to solving some known city issues. The goal of the smart city

Table 1. Summary of the application of blockchain in specific fields

Ref.s	Fields	Description
[6]	Smart grid	Blockchain-based data protection model; strengthening the defense of smart grid against external network attacks
[7]	Medical	Hybrid storage model; improving the scalability of blockchain for the medical field.
[8]	Transportation	Blockchain-based Internet of Vehicles trust system; realizing transportation detection management.
[9]	Supply sys.	Ethereum-based supply chain system; ensuring the security and traceability of the information.

is to build a sustainable city through new technologies. New technologies (e.g., blockchain [1, 2]) provide technical supports for the construction of smart cities. This survey focuses on new adoptions of blockchain in smart city.

Blockchain [3–5] is known as a distributed data ledger that jointly maintains system tamper-proof, immutability, and traceability in a decentralized and trustless manner. Blockchain-based smart cities can solve some existing city issues. For example, decentralization and trustless mechanisms secure the execution of distributed environments without a trusted third party; immutability ensures security of data transmissions; traceability is conducive to the realization of the full life cycle management of block storage data; the deployment of smart contracts, moreover, realizes automated executions of transactions and expand applications in various fields in smart city.

Many prior researches have explored blockchain applications from various dimensions, for instance, as shown in Table. 1. However, investigations targeting at blockchain adoptions in smart city are rare. To make up for the application gap in this field, this survey mainly synthesizes blockchain-based smart city solutions from perspectives of transportation, medical, energy [10], and manufacturing. In addition, we further discuss major factors of blockchain-based smart city in order to provide a fundamental view of the blockchain adoptions in smart city.

The rest of this survey is organized in the following order. Section 2, 3, 4, 5 elaborate on the four main application scenarios of blockchain-based smart cities, namely medical, transportation, energy and manufacturing, respectively. The discussions of the smart city are presented in Section 6. Finally, we concluded the work in Section 7.

2 Blockchain-based Smart Energy

Energy is a crucial foundation for urban economic development. The traditional centralized energy system is no longer appropriate for the current situation, and the energy system is gradually evolving from centralized to dispersed [11]. Blockchain includes attributes like anonymity, decentralization, and traceability that can be utilized to change the way the energy system operates.

The smart grid [12] is a new grid infrastructure that supports the two-way delivery of energy and information. A smart grid must effectively manage the data related to distributed energy. To ensure the secure sharing of distributed energy data, a safe and auditable data exchange system [13] for smart grids based on blockchain was proposed, which establishes a framework for data exchange that includes fine-grained access control, computation of sensitive data, and data traceability. Furthermore, to ensure the security of private information, an out-of-chain smart contract based on the trusted execution environment is designed in this scheme.

The smart grid is vulnerable to cyberattacks because network communication technology based on TCP/IP and Ethernet protocols exposes it to the public network. To strengthen the smart grid's defenses against outside cyberattacks, a distributed data protection model [6] based on blockchain was presented. In this model, smart meters are used as nodes in the blockchain to capture and store power data to ensure data security. Aiming at the security problems of smart grids in an open environment, Bera et al. [14] proposed a blockchain-based access control for the smart grid, which provides anonymity and non-traceability.

The decentralization of blockchain is consistent with distributed energy trading. There are many designs that combine blockchain and energy trading [15–18]. To ensure privacy in smart grid energy transactions, Gai et al. [15] presented a consortium blockchain-based energy trading system. To defend against linking attacks and malicious data mining algorithm attacks, this scheme protects private data such as energy trading trends by building an account mapping mechanism on the smart contract.

In addition, some scholars have been interested in the efficiency of the energy trading process. Ali et al. [16] proposed an adaptive network model for P2P energy trading. In this model, the smart contract is used to create and manage prosumer groups, which improves the scalability of the prosumer grouping mechanism and eventually improves the overall system performance. For distributed energy, the use of P2P energy trading can reduce energy consumption costs. Khalid et al. [17] designed a hybrid P2P energy trading model based on blockchain. This scheme makes use of three smart contracts to realize a complex energy trading market. Among them, the master smart contract manages all energy trading operations, the P2P smart contract is used to control the relevant information between producers and consumers. The prosumer-to-grid (P2G) smart contract is used to manage the data related to the main power grid. AlSkaif et al. [18] constructed a full P2P energy trading model based on blockchain for residential energy systems with different distributed energy types. In this scheme, two new trading strategies are proposed, one is the matching strategy of supply and demand, and the other is the matching strategy based on distance. In the supply and demand matching strategy, the transaction coefficients are calculated according to the power demand and excess power. In the distance-based matching strategy, P2P trading between nearby participants is encouraged to reduce the loss of long-distance transmission, thus improving energy efficiency.

3 Blockchain-based Smart Medical

In current city life, medical data is basically stored and managed by each hospital separately. However, the centralized storage of medical data will cause certain security problems. As a distributed ledger, blockchain provides distributed and secure medical data storage.

Blockchain provides an efficient and low-latency method for data storage. For example, Gaetani et al. [19] use blockchain to ensure data integrity. They design a database based on blockchain to solve the low throughput and high latency in the original cloud computing environment. However, the medical data of patients are stored separately in each hospital, resulting in the fragmentation of the medical data storage of patients, which brings inconvenience for patients to seek medical treatment across hospitals.

To build a full life cycle of medical data storage management for patients, Yue et al. [20] proposed a Healthcare Data Gateway based on blockchain, which guarantees patients to control their private information. The complete data storage framework was built through the blockchain, which built a lifelong traceable and tamper-proof data record for patients. Since all data in a single blockchain system has been stored. However, the data storage capacity of the current blockchain is limited. To improve the scalability of blockchain, Sun et al. [7] proposed a hybrid storage mode based on blockchain pluses IPFS. Therefore, in the face of massive medical data, the hybrid storage model can safely and irreversibly store medical data in blockchain and IPFS without violating the concept of blockchain distributed and trusted storage, which satisfies big data storage requirements to a certain extent.

In addition, there is a problem with privacy disclosure in the process of sharing medical data. To realize the safe and reliable sharing of medical data under the premise of ensuring user privacy, Kumar et al. [21] proposed a patient-centered framework and a data access control mechanism for PHRs stored in semi-trusted servers. This scheme uses ABE technology to encrypt the PHR files of each patient. However, the ABE system needs additional computing overhead to perform attribute revocation and encrypt the data again. To reduce costs, Gu et al. [22] proposed a more effective ABS scheme using monotone predicates, but it cannot solve the problems caused by access control modifications. In this regard, Guo et al. [23] introduced a signature scheme with multiple permissions to ensure the effectiveness of medical data in the blockchain.

In the smart medical system, blockchain can effectively solve the problems of data isolation. In addition, it can tamper with the current medical data storage, and relieve the pressure on data storage caused by massive medical data.

4 Blockchain-based Smart Transportation

In this section, several of the newly developed blockchain applications that relate to the transportation industry will be thoroughly discussed. Transportation serves as both the urban development's infrastructure and the heartbeat of the

city. Promoting the establishment of smart cities requires the development of smart transportation. Smart transportation scenario refers to the industry developed by using cutting-edge technologies (e.g., blockchain etc.) combined with the traditional transportation industry.

Traditional transportation system has two main problems. First, the level of intelligence and digitalization of traditional transportation infrastructure is low, and the capability of controlling information is weak. Second, some data-centric transportation applications, such as vehicular ad-hoc networks (VANETs), suffer from the lack of a fixed infrastructure and dynamic information communication. Blockchain can assist in building a distributed, secure and trustworthy smart transportation system. Applying blockchain technology to the transportation industry can achieve borderless connectivity and flat sharing of transportation system data.

Currently, many related research [24, 8, 25] are applying blockchain technology to various aspects of transportation. In the aspect of big data management. To solve the serious road safety problem caused by the easy modification of car odometer data, Preikschat et al. [24] proposes a blockchain-based distributed database. This distributed database enables data sharing of odometers and ensures that data is protected from tampering. This scheme enhances the public transparency of odometer data to a certain extent.

In the aspect of transport detection management, Yang et al.[8] give an implementation of an IoV trust management system using blockchain. The scheme focuses on verifying the received information by the Bayesian model and calculating the trust offset value of the received information. It realizes the combination of data and blockchain. Additionally, Liu et al. [25] assert that blockchain technology might be used to plan the charging and discharging of electric vehicles. In this scheme, electric vehicles firstly place charging and discharging orders to the public blockchain trading platform of the smart grid. Then, matching orders are processed and authenticated by peer nodes within the blockchain network. Finally, both organizations save the confirmed requests in a distributed manner. All the above researches prove that blockchain provide some technical support to ITS, thus improving the distributed capability as well as the security of the system.

Some related research has shown that blockchain can help build vehicular VANETs and manage vehicle communications. Conventional VANETs suffer from the lack of fixed infrastructure and dynamic information communication. It has been demonstrated that dynamic VANETs can be safely managed using software-defined VANETs [26]. Following that, it is incorporated with blockchain technology to offer distributed control over the entire network. Moreover, To improve the interactive security of electric vehicles in VANETs, data coins and energy coins stimulated by blockchain can be defined as a new type of cryptocurrency and can be applied to the vehicles [27]. This literature focuses on workload proofs using data contribution frequencies and energy contribution amounts to achieve distributed consensus among vehicles. In addition, a significant problem that needs to be resolved is how to store and transmit data securely in VANETs

environment. It has been demonstrated [28] that the issue of safe data storage on VANETs can be resolved by implementing a distributed transaction storage method based on a blockchain.

Further, to address the consistency and non-tamperability of transmitted data, Zhang [29] proposed a data sharing and storage system based on blockchain. The system allows having digital signatures in a self-organizing on-board network. Roadside units (RSU) can also execute smart contracts to define the parameters of data exchange and store replica sensor data in a distributed method.

All the above applications prove that blockchain technology can bring certain technical support to the transportation field, thus making transportation under smart cities more efficient and safer.

5 Blockchain-based Smart Manufacturing

Smart manufacturing achieves production optimization and improves production efficiency by utilizing new-generation information technologies, including but not limited to the Industrial Internet of Things (IIoT) and automation [30, 31]. However, smart manufacturing is currently facing challenges such as security, interoperability, privacy protection, and traceability [32, 33]. As a decentralized and trustless distributed data ledger, blockchain provides new ideas for solving the problems existing in smart manufacturing. Next, the applications of blockchain in IIoT and supply chain management are discussed below.

IIoT is an application of the IoT in industry and one of the core technologies of smart manufacturing. However, the traditional IIoT has problems of high cost, high communication overhead, low security, and high latency. The distributed storage of blockchain can effectively solve the problems of a single point of failure in centralized IIoT and improve its robustness of it [34].

A variety of solutions have been proposed by researchers in response to the problems existing in IIoT. Huo et al. [35] proposed a trusted identifier co-governance architecture with the fusion of blockchain and Handle technology. The architecture is divided into three levels. In the second level, blockchain is introduced to provide efficient and stable identity resolution services. The proposed solution not only further improves the performance of identity resolution services, but also ensures data security and reliability. Although blockchain can make IIoT securer, there are still shortcomings in communication efficiency. Wang et al. [36] attempted to replace the Merkle tree in the blockchain with incremental aggregator subvector commitment, which realizes the aggregation proof of multiple data blocks. It effectively solves the problem of low communication efficiency caused by verifying data. In addition, the combination of blockchain and other technologies (e.g., cryptography, machine learning, artificial intelligence) can better ensure the security of IIoT. Tan et al. [37] proposed an IIoT key data protection scheme based on blockchain, where the private key is split and encrypted through the Shamir secret sharing algorithm. They also publish it on the blockchain. Besides, Mansour [38] proposed an Intrusion Detection System for IIoT, which not only improves the accuracy of intrusion

detection, but also uses blockchain to achieve secure data transmission. The above applications show that the deep integration of blockchain and IIoT can solve the problems of insecure data transmission, low communication efficiency, which improves the security and stability of the IIoT equipment.

Supply chain is an important part of smart manufacturing and is moving towards digitization. However, it faces challenges and problems in information sharing, data security, traceability and enterprise interconnection [39, 9]. Blockchain provides new ideas for optimizing the management of smart manufacturing supply chain.

Many studies focus on blockchain-based solutions to the supply chain. For example, Wu and Zhang [40] investigated the problem of supply chain trust management. They proposed a framework for trust management utilizing blockchain to make the interactions among supply chain entities more reliable. The improved EigenTrust algorithm used in it optimizes the trust management model. Specifically, the malicious nodes are identified by calculating the trust value of the node. Thus, the influence of malicious nodes, who may give a negative evaluation, can be reduced.

To achieve supply chain transparency and traceability, Xu et al. [9] put forward a supply chain management system based on Ethereum. The combination of traditional database technology and blockchain enables producers, managers, and customers to obtain the required information according to their needs, and ensures the correctness and credibility of the information. In addition, Weaterkamp et al. [41] designed a supply chain system with traceability utilizing smart contracts. In contract, the ingredients of products are defined as recipes. Every ingredient is a token, which represents a batch of real goods. The token is unique and unforgeable. When products are manufactured, the input tokens are consumed and new tokens are created. It is more convenient to trace the conversion process of products.

The above applications show that blockchain effectively solves problems in the supply chain. It makes the smart manufacturing supply chain more secure, reliable and transparent and improves its efficiency.

6 Discussions

While blockchain brings convenience to smart cities, we also hope to protect citizens' data security and privacy. However, there are some privacy leakage issues with the blockchain. For example, users participating in the construction of the blockchain cannot guarantee complete anonymity, resulting in the disclosure of user identity data. This is because the blockchain is open and transparent, and all network participants can see all the information stored on the blockchain. Although the blockchain maps network participants to pseudonymous addresses in order to ensure anonymity. However, attackers can use the combination of public information stored in the blockchain and external information to track the actions of users, so as to obtain the real identity of the user, resulting in the

disclosure of user privacy. Therefore, it is crucial to study the anonymity of the blockchain to ensure the privacy of users.

In addition, the scalable storage of smart blockchains is a prerequisite for future urban applications. In smart cities, various data collection devices will generate a large amount of data, which requires blockchain technology to process. However, each node in the traditional blockchain contains complete transaction information, which cannot store all transaction information due to the limited storage space of the blockchain. Therefore, it is impossible to directly apply the blockchain to smart cities. We need to study the scalability of the blockchain to meet the application needs of massive data in smart cities. In addition, we can use decentralized storage methods in the blockchain system to increase its storage, thereby providing the possibility for a wide range of applications in smart cities.

7 Conclusion

This survey explored the application of blockchain technology in smart cities. First, we discussed in detail the relevant applications of blockchain technology in smart cities from the perspectives of medical, transportation, energy and manufacturing. Further, we also discussed the privacy and scalability of blockchain-based applications in smart cities. To sum up, the application of blockchain to smart cities improve people's quality of life. Moreover, we also believe that this survey provides new ideas for the development of smart cities.

Acknowledgements

This work is partially supported by the National Key Research and Development Program of China (Grant No. 2021YFB2701300), Natural Science Foundation of Shandong Province (Grant No. ZR2020ZD01).

References

1. Gai, K., Wu, Y., Zhu, L., Choo, K.R., B. Xiao, B.: Blockchain-enabled trustworthy group communications in UAV networks. IEEE Transactions on Intelligent Transportation Systems **22**(7), 4118–4130 (2021)
2. Gai, K., Qiu, M.: Optimal resource allocation using reinforcement learning for IoT content-centric services. Applied Soft Computing **70**(1), 12–21 (2018)
3. Zhang, Y., Gai, K., Xiao, J.: Blockchain-empowered efficient data sharing in internet of things settings. J-SAC **PP**(99), 1–1 (2022)
4. Gai, K., Tang, H., Li, G.: Blockchain-based privacy-preserving positioning data sharing for IoT-enabled maritime transportation systems. IEEE Transactions on Intelligent Transportation Systems **PP**(9), 1–15 (2022)
5. Gai, K., Guo, J., Zhu, L., Yu, S.: Blockchain meets cloud computing: A survey. IEEE Communications Surveys & Tutorials **22**(3), 2009–2030 (2020)

6. Liang, G., Weller, S., Luo, F.: Distributed blockchain-based data protection framework for modern power systems against cyber attacks. IEEE Transactions on Smart Grid **10**(3), 3162–3173 (2018)
7. Sun, J., Yao, X., Wang, S., Wu, Y.: Blockchain-based secure storage and access scheme for electronic medical records in IPFS. IEEE Access **8**, 59389–59401 (2020)
8. Yang, Z., Yang, K., Lei, L., et al: Blockchain-based decentralized trust management in vehicular networks. IEEE Internet of Things Journal **6**(2), 1495–1505 (2018)
9. Xu, Z., Zhang, J., Song, Z., et al: A scheme for intelligent blockchain-based manufacturing industry supply chain management. Computing **103**(8), 1771–1790 (2021)
10. Gai, K., Qiu, M., Zhao, H., Tao, L., Zong, Z.: Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing. Journal of Network and Computer Applications **59**, 46–54 (2016)
11. Bao, J., He, D., Luo, M., Choo, K.K.R.: A survey of blockchain applications in the energy sector. IEEE Systems Journal **15**(3), 3370–3381 (2020)
12. Dileep, G.: A survey on smart grid technologies and applications. Renewable energy **146**, 2589–2625 (2020)
13. Wang, Y., Su, Z., Zhang, N.: Spds: A secure and auditable private data sharing scheme for smart grid based on blockchain. IEEE Transactions on Industrial Informatics **17**(11), 7688–7699 (2020)
14. Bera, B., Saha, S., Das, A.K.: Designing blockchain-based access control protocol in IoT-enabled smart-grid system. IEEE Internet of Things Journal **8**(7), 5744–5761 (2020)
15. Gai, K., Wu, Y., Zhu, L., Qiu, M., Shen, M.: Privacy-preserving energy trading using consortium blockchain in smart grid. IEEE Transactions on Industrial Informatics **15**(6), 3548–3558 (2019)
16. Ali, F.S., Bouachir, O., Özkasap, Ö.: Synergychain: Blockchain-assisted adaptive cyber-physical P2P energy trading. IEEE Transactions on Industrial Informatics **17**(8), 5769–5778 (2020)
17. Khalid, R., Javaid, N., Javaid, S., Imran, M., Naseer, N.: A blockchain-based decentralized energy management in a P2P trading system. In: IEEE International Conference on Communications. pp. 1–6. IEEE (2020)
18. AlSkaif, T., Crespo-Vazquez, J.L., Sekuloski, M., van Leeuwen, G., Catalão, J.P.S.: Blockchain-based fully peer-to-peer energy trading strategies for residential energy systems. IEEE Transactions on Industrial Informatics **18**(1), 231–241 (2021)
19. Gaetani, E., Aniello, L., Baldoni, R.: Blockchain-based database to ensure data integrity in cloud computing environments (2017)
20. Yue, X., Wang, H., Jin, D., et al: Healthcare data gateways: Found healthcare intelligence on blockchain with novel privacy risk control. Journal of medical systems **40**(10), 1–8 (2016)
21. Kuma, M.R., Fathima, M.D., Mahendran, M.: Personal health data storage protection on cloud using MA-ABE. International Journal of Computer Applications **75**(8), 11–16 (2013)
22. Gu, K., Jia, W., Wang, G., Wen, S.: Efficient and secure attribute-based signature for monotone predicates. Acta Informatica **54**(5), 521–541 (2017)
23. Guo, R., Shi, H., Zhao, Q., Zheng, D.: Secure attribute-based signature scheme with multiple authorities for blockchain in electronic health records systems. IEEE access **6**, 11676–11686 (2018)
24. Preikschat, K., Böhmecke-Schwafert, M., Buchwald, J., Stickel, C.: Trusted systems of records based on blockchain technology-a prototype for mileage storing in the automotive industry. Concurrency and Computation **33**(1), e5630 (2021)

25. Liu, C., Chai, K., Zhang, X., et al: Adaptive blockchain-based electric vehicle participation scheme in smart grid platform. IEEE access **6**, 25657–25665 (2018)
26. Zhang, D., Yu, F., Yang, R.: Blockchain-based distributed software-defined vehicular networks: A dueling deep q-learning approach. IEEE Transactions on Cognitive Communications and Networking **5**(4), 1086–1100 (2019)
27. Liu, H., Zhang, Y., Yang, T.: Blockchain-enabled security in electric vehicles cloud and edge computing. IEEE Network **32**(3), 78–83 (2018)
28. Zheng, D., Jing, C., Guo, R., Gao, S., Wang, L.: A traceable blockchain-based access authentication system with privacy preservation in vanets. IEEE Access **7**, 117716–117726 (2019)
29. Zhang, X., Chen, X.: Data security sharing and storage based on a consortium blockchain in a vehicular ad-hoc network. IEEE Access **7**, 58241–58254 (2019)
30. Leng, J., Ye, S., Zhou, M.: Blockchain-secured smart manufacturing in industry 4.0: A survey. T-SMC **51**(1), 237–252 (2021)
31. Thoben, K.D., Wiesner, S., Wuest, T.: "Industrie 4.0" and smart manufacturing – a review of research issues and application examples. International Journal of Automation Technology **11**(1), 4–16 (2017)
32. Phuyal, S., Bista, D., Bista, R.: Challenges, opportunities and future directions of smart manufacturing: A state of art review. Sustainable Futures **2**, 100023 (2020)
33. Zhou, K., Liu, T., Zhou, L.: Industry 4.0: Towards future industrial opportunities and challenges. In: 2015 12th International Conference on Fuzzy Systems and Knowledge Discovery. pp. 2147–2152 (2015)
34. Hassan, M.U., Rehmani, M.H., Chen, J.: Privacy preservation in blockchain based iot systems: Integration issues, prospects, challenges, and future research directions. Future Generation Computer Systems **97**, 512–529 (2019)
35. Huo, R., Zeng, S., Di, Y., Cheng, X., et al: A blockchain-enabled trusted identifier co-governance architecture for the industrial internet of things. IEEE Communications Magazine **60**(6), 66–72 (2022)
36. Wang, J., Chen, J., Ren, Y., Sharma, P.K., A., O., Tolba, A.: Data security storage mechanism based on blockchain industrial internet of things. Computers Industrial Engineering **164**, 107903 (2022)
37. Yu, K., Tan, L., Yang, C., Choo, K.R., et al: A blockchain-based shamir's threshold cryptography scheme for data protection in industrial internet of things settings. IEEE Internet of Things Journal **9**(11), 8154–8167 (2022)
38. Mansour, R.F.: Blockchain assisted clustering with intrusion detection system for industrial internet of things environment. Expert Syst. Appl. **207**, 117995 (2022)
39. Zuo, Y.: Making smart manufacturing smarter – a survey on blockchain technology in industry 4.0. Enterprise Information Systems **15**, 1323 – 1353 (2021)
40. Wu, Y., Zhang, Y.: An integrated framework for blockchain-enabled supply chain trust management towards smart manufacturing. Advanced Engineering Informatics **51**, 101522 (2022)
41. Westerkamp, M., Victor, F., Küpper, A.: Blockchain-based supply chain traceability: Token recipes model manufacturing processes. In: 2018 IEEE International Conference on Internet of Things. pp. 1595–1602. Halifax, NS, Canada (2018)

Topic-Aware Model for Early Cascade Population Prediction

Tong Chunyan[1], Xuan zhanwei[2], Yang Song[3], Zhang Zheng[4], Zhang Hongfeng[5], Wang Hao[6], Shuang Xinzhuo[7], and Sun Hao[8⋆]

[1] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
tongchunyan@people.cn
[2] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
xuanzhanwei@people.cn
[3] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
yangsong@people.cn
[4] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
zhangzheng@people.cn
[5] Wuhan Second Ship Design and Research Institute, Wuhan 430064, China,
2654565030@qq.com
[6] Wuhan Second Ship Design and Research Institute, Wuhan 430064, China
whuwanghao@163.com
[7] Fuxin Higher Vocational College, Liaoning 123000, China.
382010@qq.com
[8] Electronic Information School, Wuhan University, Wuhan, 430072, China
2021202120053@whu.edu.cn

Abstract. This paper introduces an early content propagation popularity prediction model based on graph neural network and variational inference topic dependent dynamic variational autoencoder model (CD-VAE). CD-VAE captures the dynamics in the content propagation process, aggregates the topological information in the information diffusion process using GraphSAGE, approaches the uncertainty in terms of time and node from the perspective of probability by introducing two variational autoencoders, considers the changes in semantic characteristics in the process by integrating natural language processing methods into the model, and therefore significantly improves its prediction performance.…

Keywords: Graph Attention Network, Attention Mechanism, Topic-dependent; Variational Autoencoder, Artificial Neural Network.

1 Introduction

With the growing prevalence of mobile devices, online social networks have become one of the most important service of the Internet [1]. A large number of

⋆ Corresponding author

famous social APP, such as Twitter, Facebook and Youtube, have appeared all over the world. This study is conducted on the Weibo, a famous microblogging network in China.

In Weibo network, users are connected by directed following relationships and the information can diffuse through users' reposting behavior from their followees. When users retweet or post information, the platform pushes the information to his/her fans, which will accelerate its diffusion.

Models that predict the popularity of content can be roughly divided into three categories. Over the past few years, feature-based methods are used to predict popularity, which emphasize devising effective festures and adopt classical machine-learning models [2–5], such as Naive Bayes, Support Vector Machine (SVM) and Multi-Layer Perceptron (MLP), for prediction. However, those methods rely heavily on the devised features. Then, generative models based on Hawkes process [6, 7] or Possion process [8, 9]. are used in predictive model Generative models are used to model the intensity of the information diffusion. However, those models assume that later retweeting is only affected by the previous publishing or retweeting, which is not satisfied in real diffusion process. Recently, deep learning models are developed to predic diffusion process [10–12]. In a deep learning model, a diffusion process is embedd into a vector, which can be used to predict user behavior or diffusion. However, the existing deep learning models only rely on time-series information, while other features such as user preference, text content and user relationship are not extracted.

In summary, the difficulties of information diffusion models include: (1) Insufficient use of structure, time, and text features, ignoring joint modeling at the text-structure-temporal level; (2) There is little information, and the uncertainty of users and time levels is large, making early prediction difficult; (3) The prediction model has poor migration and generalization ability.

Our work can be concluded as follows: a deep generative model—Topic-dependent Dynamic Variation-al Autoencoder (CD-VAE) is proposed to achieve early content propagation popularity prediction. Specifically, the model learns structural features using an inductive graph neural network GraphSAGE, uses Transformer network that is based on attention mechanism to learn time series information. Finally, CD-VAE uses two variational autoencoders to model the node-level and temporal-level uncertainty to achieve propagation prediction.

2 Preliminaries

2.1 Problem Definition

The popularity prediction discussed in this paper is defined in Definition.

Definition. Information Popularity Prediction. For information C_i, the observation time window is $[t_0, T]$, and the prediction time window is $(T, T+\tau)$.The diffusion process can be expressed as $G_i = G(E, V)$, in which $e_{ij} \in E$ means user u_j retweet the information from u_i. Based on the diffusion process G_i, text content of the tweet and the time information in the observation window,

the target is to predict the popularity increase size ΔP_i in the prediction time window.

2.2 Topic-wise propagation scale

Consider the conditions of time delay, time, user historical activity, edge historical contact, published text, and forwarded generated text, the content cascade is divided into different scale intervals to obtain the distribution Y. The user's historical activity is expressed by the number of contents published and forwarded by users in history, while the historical contact degree is measured by the number of contents forwarded between two users in history, that is, the frequency of side occurrence. They are divided into different intervals according to the frequency to get their distribution. In this section, published texts and forwarded generated texts are divided into 10 categories, namely sports, finance and economics, real estate, home furnishing, education, science and technology, fashion, politics, games and entertainment, and their distribution is obtained. According to different X, the obtained joint distribution is as shown in Figure 1:

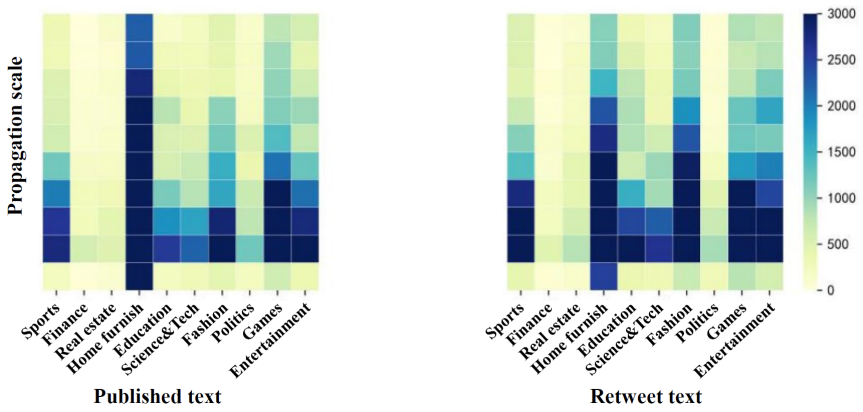


Fig. 1. Joint distribution of propagation scale of different topics

3 Model

3.1 Popularity Prediction Model

Framework CD-VAE tackles the challenges of existing prediction models by considering the following characterstics of information diffusion on social networks.

– • Topic dependence of user behavior. User social relationship and User Generated Content (UGC) are factors that influencing information diffusion and are topic dependent.

– • Structure dependence. When the retweeting structure changes, some users' retweeting behavior of the information will be changed.
– • Realtime dependence: The retweeting sequence is composed of a time sequence. In addition, information retweeting will decay over time. Information has a greater popularity scale during the day than at night, which can cause the time series to be unstable.

These three characteristics make our problem more complicated, and our model is to deal with the difficulties above.

We propose an end-to-end neural network framework to deal with these three characteristics. The popu-larity prediction model is mainly composed of 5 parts as is shown in figure 2:

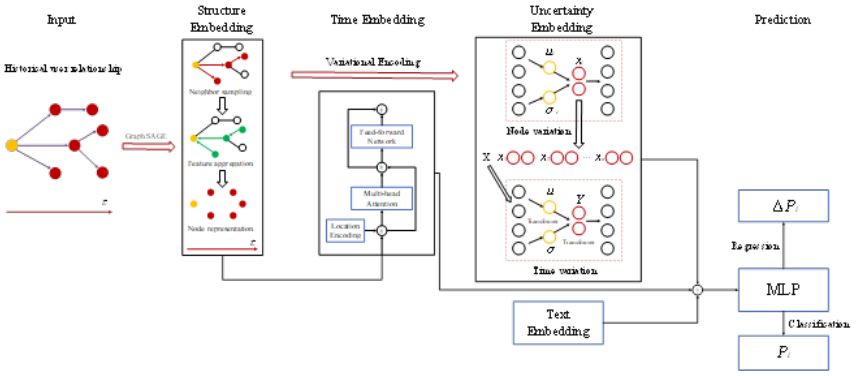


Fig. 2. CD-VAE model framework

- Structure embedding: CD-VAE mainly focuses on the structure patterns in the content propagation process and the potential relationships in the network. Specifically, the model uses an inductive graph neural network method GraphSAGE to embed nodes to learn the network structure.
- Time embedding: CD-VAE adopts Transformer network based on attention mechanism to embed time series information in the propagation process.
- Uncertainty embedding: CD-VAE uses the Variable Autoencoder (VAE) to model the node level and time level uncertainties in the content propagation process.
- Text embedding: CD-VAE uses natural language processing methods to represent non node attributes in text, and uses GraphSAGE to embed an undirected acyclic graph with attributes to obtain text representation.
- Prediction: CD-VAE combines Transformer, variational inference, and text embedding to get the final representation, and inputs it into the MLP for classification and regression prediction tasks.

Diffusion Dynamic Graph For a piece of information C_i, the first step is to jointly model the user, content, time, and structure characteristics in the observation time window into the diffusion dynamic graph G_i. The N retweeting sequences in the observation time window are represented as N subgraphs $G_i = \{g_i(t_0), g_i(t_1), \cdots, g_i(t_{N-1})\}$. The sequence set from t_0 to t_{N-1}describes the time characteristics. $g_i(t_j) = (V_i^{t_j}, E_i^{t_j}, D_i^{t_j}, F_i^{t_j})$ contains the set of nodes and edges, that is, describes the structural characteristics. User characteristics are described by attribute $D_i^{t_j}$, and content characteristics are described by attribute $F_i^{t_j}$. The user's behavioral characteristics and social characteristics are a process that depends on long-term formation, so we assume that the user's characteristics remain unchanged during the observation period. Social characteristics are presented in the form of edge attributes, so there is $D_i^{t_j} = S_i = \{s_i(0), s_i(1), \cdots, s_i(K_e - 1)\}$. User behavior characteristics and content characteristics belong to the node level, next we combine content characteristics and user behavior characteristics to node attributes.

Compared with the traditional method, we take retweeting generated text by each user as its text feature to form the user text content set $Text_i = \{text_i(0), text_i(1), \cdots, text_i(K_v-1)\}$, where $text_i(k)$ represents a word sequence. Then we use the text embedding method to embed the user text content. We use the self-attention content embedding model (SACE) shown in Figure 3 to learn text representation.

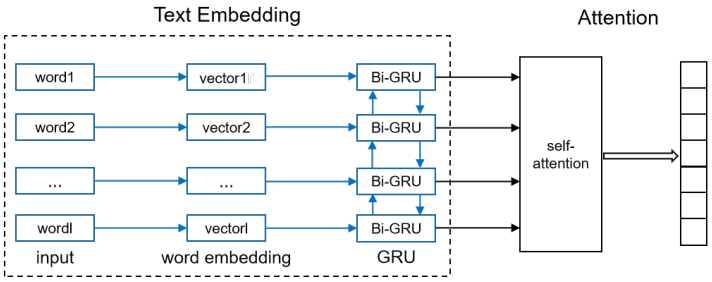


Fig. 3. SACE text embedding model

Spatio-Tempooral Dependence We model and represent information diffusion dynamics in a structure-time mode. In order to obtain structural information, we need to use graph embedding technology to learn the representation of diffusion dynamic graph G_i. User B retweeting information of user A appears to be a one-way process, and direction needs to be considered. Therefore, g is a directed acyclic graph with node attributes and edge attributes, and it is difficult to embed it effectively. Therefore, g is a directed acyclic graph with node attributes and edge attributes, and it is difficult to embed it effectively. We use graph attention network to achieve this, as shown in Figure 4. The GAT action process is divided into 3 steps: 1) Segmentation: divide the graph into 4

subgraphs according to direction, node and edge; 2) Aggregation: Use the attention mechanism to learn features for the graphs where the nodes and edges are located; 3) Combination: aggregate the sub-graphs representations.

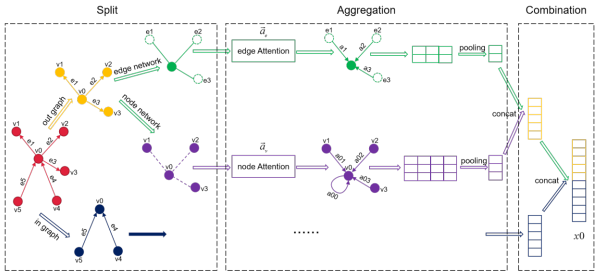


Fig. 4. Improved graph attention network: uses three layers of segmentation, aggregation, and combination to learn directed acyclic graphs with node attributes and edge at-tributes.

Prediction The result of represenation is a tensor containing three dimensions: node, time, and feature. We sum them to eliminate the node and time scale, as shown in formula (1).

$$Y_i = \sum_{t=t_0}^{t_{N-1}} \sum_{k=0}^{K_v-1} x''(t,k) \tag{1}$$

Y_i is the final representation of information C_i through the the proposed model, then we connect it to a multilayer perceptron (MLP) for prediction tasks.

$$\Delta P_i = f(C_i(User, Content, Structure, Time)) = MLP(Y_i) \tag{2}$$

Our goal is to obtain the popularity increment ΔP_i within the predicted time τ according to the user, content, structure, and time characteristics of the information C_i within the observation time T. The loss function is defined as formula (3):

$$loss = \frac{1}{c}\sum_{i}^{j}(\log \Delta P_i - \log \overline{\Delta P_i})^2 + \mu L_{reg} \tag{3}$$

Among them, c is the quantity of information, (ΔP_i) represents the predicted increment of information C_i, and $(\overline{\Delta P_i})$ represents the true increment. L_{reg} represents the L2 regularization term, which is to prevent overfitting during training, and is a hyperparameter.

4 Experiments

4.1 Dataset

We used Weibo, China's largest Weibo platform to obtain real-world data. Specifically, we tested CD-VAE on the dataset in the 2021 Artificial Intelligence Challenge contest hosted by People's Daily Online. These Weibo data are sampled. The posts with less than 10 retweets are filtered and we get a total of 18000 posts finally. For contrast, we also tested our model on the dataset that DeepHawkes used to compare the performance of CD-VAE and DeepHawkes. Figure 5 shows the relationship between the number of posts and the popularity of posts.

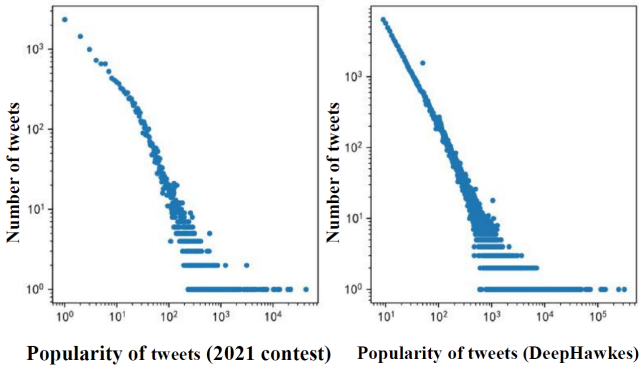


Fig. 5. Blog post popularity distribution relationship

User behavior characteristics and social characteristics will change over a relatively long period. Therefore, we consider user characteristics to remain unchanged in the short-term, we use the data to extract user-related priors characteristics.

Node and edge information of the dataset are listed in Table 1.

Table 1. Dataset statistics

Dataset	2021 Contest	DeepHawkes
Content count	18000	119313
Avg. nodes	63.51	142.24
Avg. edges	63.67	143.90

4.2 Baselines

In this part, we mainly refer to some of relevant models in the 3 types of information popularity prediction mentioned in the Introduction as the baseline model:
Feature-deep: A method based on feature design.
TopoLSTM:A method based on diffusion model [13].
DeepHawkes: A method based on generative models [14].
CasCN: A model based on deep learning [15].
NPP: A model based on deep learning [16].
CasFlow: A model based on deep learning and generative process [17].

4.3 Experimental Result

For popularity prediction, we define the information popularity prediction problem as a regression problem, and use MSLE for evaluation.

$$MSLE = \frac{1}{c}\sum_{i=0}^{c}(\log \Delta P_i - \log \overline{\Delta P_i})^2 \tag{4}$$

Among them, c is the amount of information, ΔP_i and $\overline{\Delta P_i}$ are the predicted popularity increment and true popularity increment. The smaller the MSLE, the better the effect.

We compare CD-VAE with the baseline models, and the results are shown in Table 2.

Table 2. Model performance result

	2021 Contest	Data	DeepHawkes	Data
Model	Acc	MSLE	Acc	MSLE
Feature-deep	52.03%	2.36	57.58%	3.58
DeepHawkes	64.84%	1.74	65.67%	2.67
CasCN	65.57%	1.72	66.27%	2.58
NPP	67.19%	1.62	66.52%	2.55
CasFlow	65.98%	1.64	65.12%	2.39
CD-VAE	74.02%	1.39	69.42%	2.23

From Table 2, we can see that Feature-deep model requires manual design of structure, time, content, and user characteristics, which is inefficient, and its performance is not as good as other methods. As described in Section 2, the method based on feature design relies heavily on feature selection and design. CD-VAE also uses structure, content, time, and user characteristics to achieve better results.

Although DeepHawkes method introduces a deep learning method on the basis of Hawkes, the focus is still on the learning of temporal characteristics. CasCN fully obtains temporal and structural information, but it does not consider the characteristics of users and content. In addition, CasCN learns network characteristics through the GCN layer. In contrast, our CD-VAE performs local graph learning during structural dependence modeling, which has better results. It can also handle directed acyclic graphs with attributes (nodes, edges). NPP fully mixes user, content, time characteristics, and establishes the relationship between content and user characteristics, showing better results. But in the process of information diffusion, the network topology changes with time, and NPP does not consider the structure-dependent information. From the above experimental results, we conclude that CD-VAE comprehensively considers user, content, structure, and time information, models user topic dependence and spatiotemporal dependence, and shows excellent performance on the Weibo datasets.

5 Conclusion And Future Work

We propose a topic-dependent dynamic variational autoencoder model (CD-VAE) to predict information popularity. CD-VAE is a model based on deep learning and Bayesian learning. It considers the characteristics of early propagation of Weibo content, and learns structural information via GraphSAGE, learns time information using Transformer, and deploys two variation autoencoders to learn the uncertainty of users and time in the content propagation process. The text information of users is embedded as node attributes using NLP methods, and text features are learned in a graph-dependent way. We use Weibo data to conduct experiments to predict the information popularity. The results show that CD-VAE has better results, and existing deep learning methods on different datasets.

Acknowledgement

This work was supported by the Open Funding Projects of the State Key Laboratory of Communication Content Cognition (No. 20K05 and No. A02107).

References

1. Jihe Wang, Meikang Qiu, and Bing Guo. Enabling real-time information service on telehealth system over cloud-based big data platform. *Journal of Systems Architecture*, 72:69–79, 2017.
2. Gabor Szabo and Bernardo A Huberman. Predicting the popularity of online content. *Communications of the ACM*, 53(8):80–88, 2010.

3. Zongyang Ma, Aixin Sun, and Gao Cong. On predicting the popularity of newly emerging hashtags in t witter. *Journal of the American Society for Information Science and Technology*, 64(7):1399–1410, 2013.
4. Eytan Bakshy, Jake M Hofman, Winter A Mason, and Duncan J Watts. Everyone's an influencer: quantifying influence on twitter. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 65–74, 2011.
5. Justin Cheng, Lada Adamic, P Alex Dow, Jon Michael Kleinberg, and Jure Leskovec. Can cascades be predicted? In *Proceedings of the 23rd international conference on World wide web*, pages 925–936, 2014.
6. Yasuko Matsubara, Yasushi Sakurai, B Aditya Prakash, Lei Li, and Christos Faloutsos. Rise and fall patterns of information diffusion: model and implications. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 6–14, 2012.
7. Swapnil Mishra, Marian-Andrei Rizoiu, and Lexing Xie. Feature driven and point process approaches for popularity prediction. In *Proceedings of the 25th ACM international on conference on information and knowledge management*, pages 1069–1078, 2016.
8. Huawei Shen, Dashun Wang, Chaoming Song, and Albert-László Barabási. Modeling and predicting popularity dynamics via reinforced poisson processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, 2014.
9. Shuai Gao, Jun Ma, and Zhumin Chen. Modeling and predicting retweeting dynamics on microblogging platforms. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 107–116, 2015.
10. Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.
11. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
12. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
13. Jia Wang, Vincent W Zheng, Zemin Liu, and Kevin Chen-Chuan Chang. Topological recurrent neural network for diffusion prediction. In *2017 IEEE international conference on data mining (ICDM)*, pages 475–484. IEEE, 2017.
14. Qi Cao, Huawei Shen, Keting Cen, Wentao Ouyang, and Xueqi Cheng. Deephawkes: Bridging the gap between prediction and understanding of information cascades. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1149–1158, 2017.
15. Xueqin Chen, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Fengli Zhang. Information diffusion prediction via recurrent cascades convolution. In *2019 IEEE 35th international conference on data engineering (ICDE)*, pages 770–781. IEEE, 2019.
16. Guandan Chen, Qingchao Kong, Nan Xu, and Wenji Mao. Npp: A neural popularity prediction model for social media content. *Neurocomputing*, 333:221–230, 2019.
17. Xovee Xu, Fan Zhou, Kunpeng Zhang, Siyuan Liu, and Goce Trajcevski. Casflow: Exploring hierarchical structures and propagation uncertainty for cascade prediction. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

GeoNet: Artificial neural network based on geometric network

Cui Xiangyang[1], Yan Zhou[2], Yang Song[3], Zhang Zheng[4], Zhang Hongfeng[5], Wang Hao[6], Shuang Xinzhuo[7], and Nie Qi[8⋆]

[1] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
cuixiangyang@people.cn
[2] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
yanzhou@people.cn
[3] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
yangsong@people.cn
[4] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
zhangzheng@people.cn
[5] Wuhan Second Ship Design and Research Institute, Wuhan 430064, China,
2654565030@qq.com
[6] Wuhan Second Ship Design and Research Institute, Wuhan 430064, China
whuwanghao@163.com
[7] Fuxin Higher Vocational College, Liaoning 123000, China.
382010@qq.com
[8] Electronic Information School, Wuhan University, Wuhan, 430072, China
nieqi@whu.edu.cn

Abstract. Artificial neural network has achieved great success in many fields. Considering the unique advantages of naturally generated networks, we combine the geometric complex network model with the existing neural network model to build a neural network with geometric space structure characteristics. We proposes a GeoNet neural network model based on a random geometric network structure and finds that the neural network with a natural structure has good classification performance, and the classification accuracy is higher than the widely used neural network structure.…

Keywords: Complex Network, Geometric Network Model, Network Generation, Residual Structure, Artificial Neural Network.

1 Introduction

Artificial neural network (ANN) has achieved great success in machine learning such as image recognition, object detection, computer vision, natural lan-

⋆ Corresponding author

guage processing, and so on, which drives deep learning to become a very popular research topic. As an important hyperparameter, the structure of a network is a significantly vital factor promoting the development of the neural network. Currently, ANNs have evolved from simple chain-like models to structures with multiple wiring paths such as ResNets [1], DenseNets [2], etc. In deep learning, how computational networks are wired is crucial for building intelligent machines.

A large number of studies show that the naturally formed network structure has many advantages such as high information transmission efficiency [3, 4], high economy [3, 5], and good robustness [6]. Therefore, we have reason to doubt the superiority of the traditional structure: whether the naturally formed network structure has more advantages than the current artificially designed structure.

In this paper, the network structure with natural characteristics is combined with the neural network, and the naturally formed complex network is applied to the structural design of the neural network. By comparing the artificial residual model with the real network structure, this paper believes that the structural features of efficient training naturally exist in the natural network. In this paper, stochastic geometric networks are used to describe the topological connection of artificial neural networks, so that the neural networks have the key characteristics of natural networks, such as small-world effect, powerlaw degree distribution, and high aggregation. According to the existing neural network technology, this paper proposes a GeoNet neural network model based on a random geometric network structure and finds that the neural network with a natural structure has good classification performance, and the classification accuracy is higher than the widely used neural network structure.

2 Related work

In 2006, the concept of deep learning accelerated the research of neural networks, in which convolutional neural networks are widely used. In 2012, AlexNet [7] convolutional neural network won the championship in the ImageNet image recognition competition. The network uses a Rectified Linear Unit (ReLU) instead of the traditional sigmoid activation, which alleviates the problem of gradient disappearance in the backpropagation of the neural network, and uses dropout technology to prevent the neural network from overfitting. Subsequently, the vision group of Oxford University proposed the VGG network [8] in 2014, which uses 3×3 The small-size convolution kernel replaces the large-size convolution kernel and improves the network representation ability by cascading. The research also explores the relationship between the depth of the convolution neural network and its performance, showing that the depth of the neural network is a key factor in the network performance. Because of VGG's excellent feature extraction ability, the VGG network has been widely used in image processing tasks such as style migration. In the same year, Google proposed GoogLeNet [9–11], which uses the multibranch Inception structure to increase the adaptability of the network to multi-scale features. In 2015, He

Kaiming and others proposed ResNet [**?**], aiming to alleviate the problem of gradient disappearance of network backpropagation training, so that networks can be stacked to a deeper depth. Although the residual structure of ResNet reduces the training difficulty of the network, there is evidence that the design will result in only a small number of residual blocks learning useful information, and this problem is described as dividing feature reuse [12]. To alleviate this problem, WideResNet [13] studied the impact of increasing the residual block width on the performance of the residual network. This study shows that increasing the number of convolution cores is beneficial to the performance of the residual neural network. PyramidNet [14] studied the scale change of network width and depth on this basis and achieved a good balance between them. DenseNet model [2] uses a dense connection structure to enable all layers in the network to accept the features of all previous convolution layers. This structure can effectively alleviate the problem of gradient disappearance and promote feature reuse.

The performance of the convolutional neural network is improved with the growth of model size. Generally, better network performance can be obtained by increasing the depth and expansion width. However, the high computational complexity, memory consumption, and delay limit the practical application of deep networks. MobileNet [15] uses depth separable convolution to reduce the redundant parameters of traditional 3D convolution and the computation required for convolution. At the same time, the network uses the RELU6 activation function to maintain the robustness of the network under low floating point numerical accuracy. MobileNet's low computational complexity and low latency make it suitable for mobile device deployment scenarios. In 2018, Google proposed MobileNet V2 [16], which uses expansion convolution compression reverse residuals and Linear Bottlenecks to retain feature information as much as possible. Recently, Google proposed MobileNet V3 [17]. By introducing lightweight attention [18] and the h-swish activation function, the network has significantly improved its performance in classification, semantic segmentation, and target detection tasks. In addition to manually designing the neural network structure, the neural network architecture search [19–23] (Neural Architecture Search, NAS) uses deep learning to automatically screen the neural network structure, which is also widely used in the current neural network design.

3 Geometric neural network model

Real networks are usually characterized by high concentration and small-worldness. Among them, the high aggregation is reflected in the triangular connection relationship in the network, which forms the same connection form as the shortcut connection of the residual network. At the macro scale, the small-world feature of the network makes the path length between network nodes shorter, which is conducive to slowing down the gradient attenuation phenomenon of neural network training. Figure 1 shows the structural similarity between the residual network and the real network. The real network naturally has structural characteristics that are easy to train.

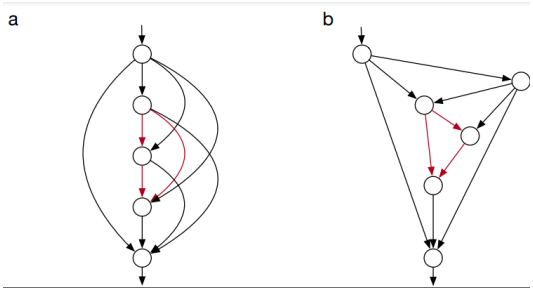


Fig. 1. Schematic diagram of residual network and real network. (a) Resnet network structure; (b) Real network structure. In the figure, red lines are used to identify a group of triangular connections in the network. The triangle connection in the real network has the same connection form as the shortcut structure in the residual network.

3.1 Network Connection Structure and Function

Connection structure of the network The generation network of the S^1 model has the characteristics of scale-freeness, high aggregation and small-worldness, and the network nodes have the characteristics of spatial distribution. Its model features are highly consistent with the structural features of the brain neural network [24, 25], so this paper uses S^1 model to simulate the real network structure, and the connection relationship of the neural network is generated according to the following model. The n nodes of the network are uniformly distributed in the ring with a radius of $R = \frac{n}{2\pi}$. Each node in the network has a parameter which follows that $\kappa \sim c\kappa^{(-\gamma)}$. The connection probability between two nodes follows that

$$p(x) = p(\kappa, \kappa', d) = \frac{1}{1 + \frac{d}{\mu\kappa\kappa'}^{\beta}} \tag{1}$$

$$x = \frac{d}{\mu\kappa\kappa'} \tag{2}$$

where $\mu = \frac{1}{2I\langle\kappa\rangle}$; $I = \int p(x)dx$; $\langle\kappa\rangle$ represents node variable κ Expectation of distribution. The expectations for generating node degrees in the network are $\bar{k}(\kappa) = \kappa$. $\beta(>1)$ is the parameter that controls the network aggregation coefficient.

Function of nodes and edges In this paper, network edge connection describes the data flow in the network, and the edge connection transfers the output characteristics of the node to its neighbors. The edge connection direction reflects the direction of data transmission; The network node realizes the feature processing function of data, specifically including three stages of feature aggregation, feature transformation and feature distribution [26].

3.2 Network Nodes model

Based on the neural network technologies such as batch normalization [27], deeply separable convolution [28], inverse residual structure [16] and channel attention mechanism [29], this paper considers the following node feature transformation model, as depicted in Figure 2.

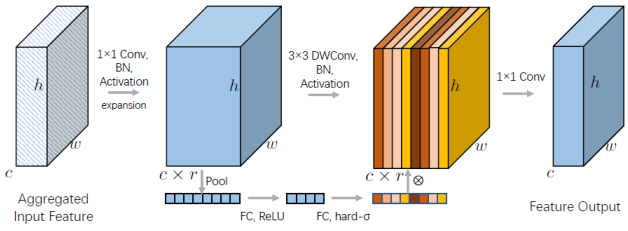


Fig. 2. Characteristic transformation of nodes.

The feature transformation of nodes includes three convolution layers. Aggregation characteristics of nodes are used in the first convolution layer. 1×1 convolution is used for improving the number of channels of the feature. The second convolution layer is 3×3 depth convolution, which is used to extract the spatial features of the image and reduce the number of parameters of the model. The third layer is a 1×1 convolution for restoring the channel dimension of the original feature. After the first convolution layer, this paper uses a lightweight channel attention mechanism to weigh the importance of different channels to improve the network performance. The first 1×1 convolution layer and depth convolution adopt batch normalization and nonlinear activation to improve the training speed of the network and the nonlinearity of feature transformation; In the node model, the last 1×1 convolution layer is used to realize linear weighting between characteristic channels [30].

3.3 Geometric neural network architecture

According to the connection structure and node model of the neural network, this paper constructs the image classification convolution neural network GeoNet model with the network structure shown in Table 1.

GeoNet model is mainly divided into three parts. The first part consists of two 3×3 Convolution layer conv1 and conv2, which is used to extract low-level features of images. The second part is composed of random geometric networks conv3 and conv4. In order to realize feature reuse in neural networks, this paper uses two 32-node random geometric networks stacked to replace a single 64-node random geometric network. For the input nodes of conv3 and conv4, the stripe of the depth convolution is set to 2 to reduce the size of the feature. Finally, the GeoNet uses 1280 dimensional 1×1 convolution layer conv5 to realize the

Table 1. GeoNet network architecture.

Stage	Output	Layer	Output Channel
input	32×32	-	3
conv1	32×32	3×3 conv	36
conv2	32×32	3×3 conv	36
conv3	16×16	geonet, n=32 expansion ratio=4	72
conv4	8×8	geonet, n=32 expansion ratio=3	144
conv5	8×8	1×1 conv	1280
gp	1×1	global pooling	1280
fc		fully connected layer	#classes

linear combination of feature channels and uses global pooling to form the global features of the image. The logit vector of global features that distinguishes the category is output through the full connection layer. In order to reduce the risk of overfitting, the dropout layer [31] with a probability of 0.5 is used to regularize the network after the full connection layer.

4 performance evaluation

4.1 Experimental data set

The experiments in this paper use the public CIFAR-10 and CIFAR-100 datasets. Because of their rich categories and appropriate image size, CIFAR-10 and CIFAR-100 datasets are suitable for model verification under complex connection structures.

4.2 model training

For image classification, this paper uses the cross-entropy loss function and the Minibatch Gradient Descent method for training. The initial learning rate is set to $\eta = \frac{B}{256}\eta_{base}$, where B is batch size set to 64; basic learning rate is set as $\eta_{base} = 0.1$. The momentum parameter is set to 0.9.

This paper adopts the half cycle cosine attenuation adjustment strategy [32] and the linear warm-up strategy [33] to gradually adjust the learning rate in the training process. For the linear warm-up strategy, the learning rate is linearly adjusted from 0 to the initial learning rate in the first five epochs.

In this paper, label smoothing [34, 35], regularization and data enhancement techniques are used to avoid the overfitting problem of the network. In which, the label smoothing parameter is set to 0.1. The regularization method of weight attenuation is used for the training of weight parameters in the network, and the weight attenuation parameter is set to 5e−5. In this paper, data enhancement techniques such as whitening, random translation (the translation distance is less than or equal to 4 pixels), and horizontal random flipping are used to process the training image and serve as the input of the neural network.

4.3 Image classification performance

In this paper, the convolution step of the benchmark model is adjusted to adapt to the image input size of CIFAR-10 and CIFAR-100 datasets, while the network connection mode remains unchanged. In order to achieve the comparison of network structures, this paper constructs the Resnetlike-GeoNet network, which uses the node model proposed in this paper, but uses the connection mode of the residual network to compare the performance of natural network structure and residual network structure. Through experiments, Table 2 shows the classification performance of this model and the benchmark model.

Table 2. Image classification performance

Model	Number of parameters	accuracy(%)	
		CIFAR-10	CIFAR-100
Resnet-50 [1]	23.74M	93.36	75.74
Resnet-50 v2 [13]	23.72M	94.31	77.21
Densenet-121 [2]	7.14M	94.33	75.85
Mobilenet [15]	3.33M	92.43	72
Mobilenet v2 [16]	2.38M	93.19	73.22
GeoNet	9.39M	95.59	79.73
GeoNet-Reslike	9.39M	95.20	78.39

The parameters of GeoNet: $\gamma = 9$. $\beta = 5.5$, $\langle\kappa\rangle = 3$.

The experimental results show that the GeoNet model in this paper achieves good classification accuracy in CIFAR-10 and CIFAR-100 datasets, with accuracy rates of 95.59% and 79.73%. Compared with other models, this paper achieves the best classification performance; Compared with the results of other models, the accuracy of the GeoNet model in this paper has been greatly improved. The results show that the network structure based on the geometric network model in this paper is superior to the classical model with residual structure in classification performance. To verify the advantages of network structure, the GeoNet model in this paper is superior to the Resnetlike-GeoNet model in classification performance. The network structure with natural characteristics is conducive to the improvement of neural network performance. Moreover, we draw the accuracy change curves of different models in the training process, as shown in Figure 3. Compared with other models, GeoNet network has a faster convergence speed.

5 Conclusion

The realistic biological neural network is distributed in a certain geometric space. This paper combines the geometric complex network model with the existing neural network model to build a neural network with geometric space

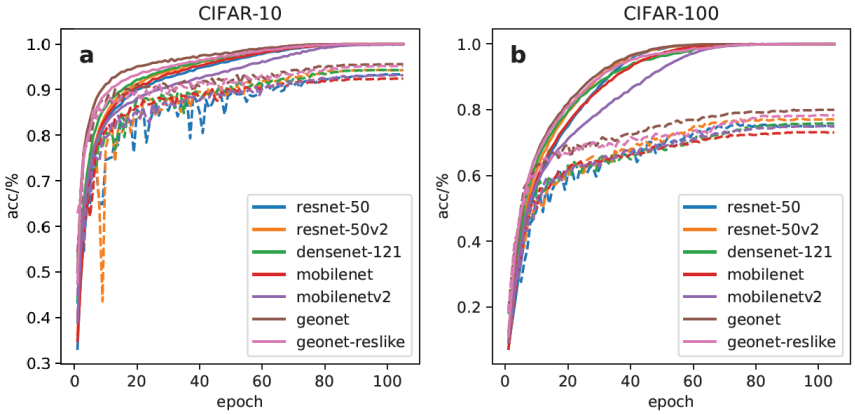


Fig. 3. Accuracy curve. The solid line in the figure represents the accuracy of the training set, and the dotted line represents the accuracy of the test set. (a) CIFAR-10 data set training curve; (b) CIFAR-100 dataset training curve.

structure characteristics. This chapter clarifies that due to the high aggregation characteristics and small world effect of the real network, the real network structure has some similarities with the current artificially designed residual network. The shortcut connection naturally exists in the real network structure, meeting the characteristics of efficient training of the backpropagation algorithm. With the help of current neural network components, this paper designs GeoNet neural network and tests the classification performance of the neural network with geometric space structure. Experiments show that GeoNet network has good classification performance, and its performance exceeds the current commonly used network structure.

Acknowledgement

This work was supported by the Open Funding Projects of the State Key Laboratory of Communication Content Cognition (No. 20K05 and No. A02107).

References

1. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
2. Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

3. András Gulyás, József J Bíró, Attila Kőrösi, Gábor Rétvári, and Dmitri Krioukov. Navigable networks as nash equilibria of navigation games. *Nature communications*, 6(1):1–10, 2015.
4. Vito Latora and Massimo Marchiori. Efficient behavior of small-world networks. *Physical review letters*, 87(19):198701, 2001.
5. Vito Latora and Massimo Marchiori. Economic small-world behavior in weighted networks. *The European Physical Journal B-Condensed Matter and Complex Systems*, 32(2):249–263, 2003.
6. Réka Albert, Hawoong Jeong, and Albert-László Barabási. Error and attack tolerance of complex networks. *nature*, 406(6794):378–382, 2000.
7. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.
8. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
9. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.
10. Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.
11. Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.
12. Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.
13. Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.
14. Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5927–5935, 2017.
15. Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.
16. Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.
17. Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019.
18. Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.
19. Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.

20. Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.
21. Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *International conference on machine learning*, pages 4095–4104. PMLR, 2018.
22. Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *Proceedings of the European conference on computer vision (ECCV)*, pages 19–34, 2018.
23. Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.
24. Danielle S Bassett and Edward T Bullmore. Small-world brain networks revisited. *The Neuroscientist*, 23(5):499–516, 2017.
25. Dante R Chialvo. Critical brain networks. *Physica A: Statistical Mechanics and its Applications*, 340(4):756–765, 2004.
26. Saining Xie, Alexander Kirillov, Ross Girshick, and Kaiming He. Exploring randomly wired neural networks for image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1284–1293, 2019.
27. Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.
28. Laurent Sifre and Stéphane Mallat. Rigid-motion scattering for texture classification. *arXiv preprint arXiv:1403.1687*, 2014.
29. Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.
30. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.
31. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.
32. Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 558–567, 2019.
33. Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.
34. Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.
35. Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? *Advances in neural information processing systems*, 32, 2019.

Research on Blockchain-based Smart Contract Technology

WANG Hongze[1,*] and ZHANG Qinying[2]

Department of Information Engineering, Wuhan Institute of City, Wuhan, China
1540509443@qq.com
Department of Information Engineering, Wuhan Institute of City, Wuhan, China
446618463@qq.com

Abstract. With the continuous development of blockchain technology, smart contract has become an important research object among the achievable technologies on blockchain technology. Based on the characteristics of decentralization, tamper-proof and transparency of blockchain, it provides a reliable technical support for the implementation of smart contract. Based on blockchain smart contract technology, this paper aims to design a smart contract management engine with higher versatility, security, and feasibility to develop a smart contract from the joint participation of multiple users. The smart contract is proliferated through the P2P network and deposited into the blockchain. And the blockchain is designed to automatically execute the smart contract, providing a new solution to the problems of opacity, easy tampering, and low efficiency of the traditional contract. It is safer and more reliable. By specifying the treaty and trigger conditions through the program code, once the conditions are met, the contract will be automatically executed, which greatly reduces the time and space costs. Using Ether and smart contracts to develop distributed applications to realize this technology, the feasibility of applying blockchain technology in this field is explored, and a new technical implementation is provided for traditional contract signing.

Keywords: Blockchain, Smart contracts, Decentralization, Byzantine fault-tolerant algorithm

1 Introduction

As digital assets have become more and more important in our lives, the protection of digital assets, transactions, etc. have also become more and more important in our lives. At present, our control of digital assets is not secure enough, our operation of digital assets are dependent on the third party, our digital assets are stored in the database of the third party, the security and privacy of digital assets are realized through the third party, that is to say, the preservation of our digital assets need to rely on the trustworthiness of the third party.

Since 2016, smart contract technology, represented by Ethernet, has suddenly become a hot topic in all walks of life. Smart contracts are essentially programs that digitize contracts and then run them on a computer. The concept was

first proposed by Nick Szabo back in 1994, but because there was no suitable platform for the development of smart contracts, they remained at a conceptual stage until the rise of blockchain technology, which provided a suitable platform for smart contracts.

Based on further optimization of the Byzantine fault-tolerant algorithm, this paper analyzes and designs an improved S-PBFT based on this algorithm to achieve a more stable smart contract system by optimizing the algorithmic mechanism in it. The experimental data in this paper illustrates that the performance metrics of the smart contract system in terms of latency and resource utilization are significantly improved after the optimization of S-PBFT.

The main contents of this paper are as follows:

- Introduce smart contract technology from the traditional contracting system for smart deployment and execution of contracts.
- Further modularize and analyze the body of smart contracts, loading methods, consensus algorithms used, and deployment environment.
- Combine the optimized Byzantine fault-tolerant algorithm S-PBFT with the practical application of the system, and verify the optimized algorithm in the application. The results show that the method of this paper largely improves the efficiency of system processing in practical applications.

2 Background

Blockchain technology is a new application model of computer technology such as distributed architecture data storage, P2P real-time transmission, and secrecy algorithm, which solves the core problem of Decentralization through distributed architecture database, digital encryption technology and unique consensus algorithm without the premise of third-party credit institutions, and realizes a decentralized and credible system without third parties. The theoretical basis of the consensus algorithm is Byzantine fault tolerance algorithm, common consensus algorithms such as proof of stake (PoS)[16].

And smart contract is based on the digital asset control program in blockchain 2.0. In layman's terms, smart contract is to code the relevant business logic and algorithm involved, and program the complex relationship between the legal agreement, business agreement and network in reality, which contains four core parts: one is digital asset and smart property; two is digital identity, building digital identity authentication service; three is digital asset custody, relying on blockchain technology to keep all kinds of property; fourth is contract arbitration, the arbitration platform to execute delivery contracts[3].

In a broad sense this technology is not only applicable to business, but also has great application prospects in distributed computing, Internet of Things, artificial intelligence and other fields.

3 Related work of the Smart Contract Operation Mechanism

3.1 Contract subject

The operation mechanism of a smart contract is shown in Fig.1. The smart contract, which has both value and state attributes, is pre-programmed with What-If and If-Then statements in the code to provide trigger scenarios and response rules for the contract terms. The user will be provided with the returned contract address and contract interface information and can invoke the contract by initiating a transaction[18]. When miners receive a contract to create or invoke a transaction, they create or execute the contract code in a sandbox execution environment. The contract code automatically determines whether the current scenario meets the contract trigger conditions based on trusted external data sources and world state checks to strictly enforce the response rules and update the world state[11]. After the transaction is validated and packaged into a new data block, the new block will be certified by consensus algorithm and linked to the main chain of the blockchain, and all updates will take effect[21].

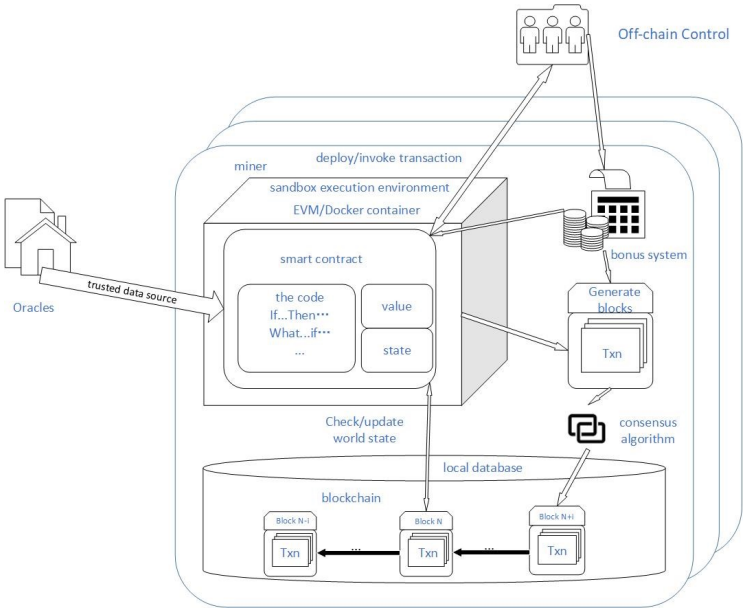


Fig. 1. The operation mechanism of smart contracts

3.2 Data Loading Method

The data layer includes state data, transaction data, application data, contract code, etc. State data and transaction data are generally stored on-chain for

verifiable and observable purposes[25]. The majority of blockchains adopt the on-chain method to publish the application data and code to the chain, and then load the data and code from the chain and execute them, but there are disadvantages. The storage burden will make the efficiency low[7].

3.3 Consensus mechanism

The consensus mechanisms currently used are Proof of Stake (PoS), Proof of WorkProof of workload mechanism (PoW), (DPoS), and Proxy proof-of-stake mechanism (DPoS),[24] and the state machine Byzantine system requires all nodes to perform a uniform operation to maintain the consistency of the system state[15]. Most of the existing state machine Byzantine systems are based on the PBFT algorithm, whose mechanism increases exponentially with the number of system nodes, thus lacking practical application value, which requires optimization of the Byzantine protocol in conjunction with practical application scenarios. An improved Byzantine algorithm, S-PBFT, is designed for the possible communication instability in distributed networks[12]. With the help of its algorithm, it can The request and reply phases in the PBFT algorithm are the interactions between the system nodes and the clients, while in the Blockchain consensus process the data blocks are generated directly by the master node without the participation of a dedicated client, and the system is able to maintain a certain level of functionality in the event of a network outage in up to 1/3 of the nodes, and after the network is restored, all nodes can quickly reach Consistent state[5].

The set of consensus nodes is G, denoted by $1, 2, ..., g$, and in order for the algorithm to reach an effective consensus, the number of consensus nodes g in the S-PBFT algorithm must not be less than $2f + 1$, combined with the consensus principle of Byzantine protocol; the set of candidate nodes is H, denoted by $1, 2, ..., h$; the set of reserve nodes is Y, denoted by $1, 2, ..., y$. The sum of the number of candidate nodes h and the number of reserve nodes y is f. Fig.2 shows the results of consensus delay comparison[1].

4 Related work to Smart Contract Implementation Technology

4.1 Ether

Ethernet is used to build applications through a complete set of Ethereum Virtual Machinecode scripting language, which is similar to assembly language[2]. We know that programming directly in assembly language is very painful, but programming in Etherium does not require the direct use of the EVM language, but rather something likeC language It is similar to a high-level language like Python, Python, etc. Lisp and other high-level languages, and then converted to EVM language by compiler[20].

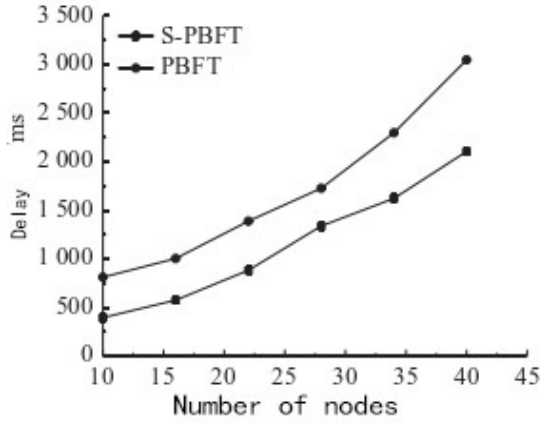


Fig. 2. Consensus Latency Comparison Results

4.2 Core Technologies

The general model of blockchain is shown in Fig.3. It usually has six layers, including the application layer, contract layer, incentive layer, consensus layer, network layer, and data layer. Among them, consensus layer and incentive layer are usually bound together, the reason is that in a large P2P network, a certain incentive is needed to reach most consensus[8], so the design of consensus algorithm also needs to take the rules of incentive distribution into account. The data layer is the most critical part of the blockchain, it is the root of the blockchain and will determine the organization based on the connectivity of individual blocks[19]. It can be considered as a composite data structure. The

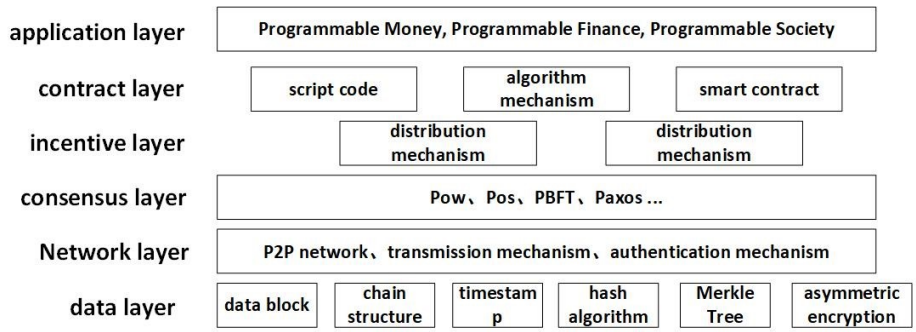


Fig. 3. Blockchain general model

network layer is mainly responsible for the validation and distribution of data

```
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\admin>java -version
java version "1.8.0_141"
Java(TM) SE Runtime Environment (build 1.8.0_141-b15)
Java HotSpot(TM) 64-Bit Server VM (build 25.141-b15, mixed mode)

C:\Users\admin>mvn -version
Apache Maven 3.3.9 (bb52d8502b132ec0a5a3f4c09453c07478323dc5; 2015-11-11T00:41:47+08:00)
Maven home: C:\Program Files\Java\apache-maven-3.3.9-bin\apache-maven-3.3.9\bin\..
Java version: 1.8.0_141, vendor: Oracle Corporation
Java home: C:\Program Files\Java\jdk1.8.0_141\jre
Default locale: zh_CN, platform encoding: GBK
OS name: "windows 7", version: "6.1", arch: "amd64", family: "dos"

C:\Users\admin>
```

Fig. 4. Experimental environment version information

throughout the distributed network. Smart contracts can be seen as a blockchain-based specificity application[14]. The contract layer consists mainly of a number of scripts, called smart contracts, which are prerequisites for the programmable functionality of the blockchain[23]. And the application layer is the layer where the blockchain development applications are located. In the blockchain model, each layer has a variety of implementations, and complex combinations produce a wide variety of blockchain types for different scenarios. The development of smart contracts has also liberated the use of blockchain scenarios, making it more flexible and diverse[22, 9].

5 Smart Signing System for Smart Contracts

5.1 System Function Implementation

The IDE used is STS (Spring Tool Suite). Part of the experimental environment is shown in Fig.4.

```
18  mapping(address => SmartLeaseContract[]) public userSmartLeaseContract;
19  function createRentContract(uint _waterMeter, uint _waterFee, uint _electricityMeter,
20      uint _electricityFee, uint _rentFee, uint _rentTerm, uint _pledgeTotal,
21      uint256 _startTime, uint256 _endTime, address _tenantPublicKey,
22      bytes _baseInfoJson) payable public {
23      address currentUser = msg.sender;
24
25      if (!(_waterMeter > 0 && _waterFee > 0 && _electricityMeter > 0 && _electricityFee > 0
26      && _rentFee > 0 && _rentTerm > 0 && _pledgeTotal > 0
27      && _startTime > 0 && _endTime > 0 && _tenantPublicKey != address(0)
28      && _baseInfoJson.length > 0)) {
29          emit CreateRentContractEvent(currentUser, false, address(0), "参数存储在空值");
30      }
31
32      SmartLeaseContract smartLeaseContract = new SmartLeaseContract(_waterMeter, _waterFee,
33          _electricityMeter, _electricityFee, _rentFee, _rentTerm, _pledgeTotal,
34          _startTime, _endTime, _tenantPublicKey, currentUser, _baseInfoJson);
35
36      userRentContract[currentUser].push(smartLeaseContract);
37      userRentContract[_tenantPublicKey].push(smartLeaseContract);
38      emit CreateRentContractEvent(currentUser, true, address(smartLeaseContract), "创建成功");
```

Fig. 5. Create contract source code

A contract system based on smart contracts and blockchain is proposed for software implementation . 1) This system digitizes traditional contracts and then deploys them to the blockchain environment, and utilizes the enforceability of programmed contracts and the decentralized features of blockchain to achieve a reliable smart contract behavior that is de-trusted between users[4]; 2) The Truffle framework, Testrpc test environment, and MetaMask wallet are organically combined together to form a complete system development and testing environment[17]. After rigorous testing, the intelligent distributed application of the contract system designed in this paper has the characteristics of robustness, simplicity of operation and practicality, and the system part of the implementation code is shown in Fig.5,

Fig. 6. System function diagram

5.2 System functional framework diagram

Unlike the traditional application architecture, this system adopts the Ethernet decentralized application architecture. Smart contracts are executed on the Ethernet virtual machine EVM, and the client only needs to make RPC calls through the web browser to interact with the instance, and the client does not need to request access to the centralized server[6].

The framework of each function of the system is shown in Fig.6.

By simulating and maintaining the operation of the whole Blockchain network through two computers together, each computer acts as a node, and these participating nodes prove the information and reach consensus through the improved S-PBFT mechanism based on the one proposed in Section 3.3. After consensus is reached, the management node broadcasts and stores the transaction information in the supply chain, and the other consensus nodes make relevant queries and traceability of the transaction information. Through a specific case implementation, the technical feasibility and the actual demand in the society, once again proves that the potential of smart contract technology in the future is not limited to this, and its technical maturity will be slowly shown in the future[13, 10].

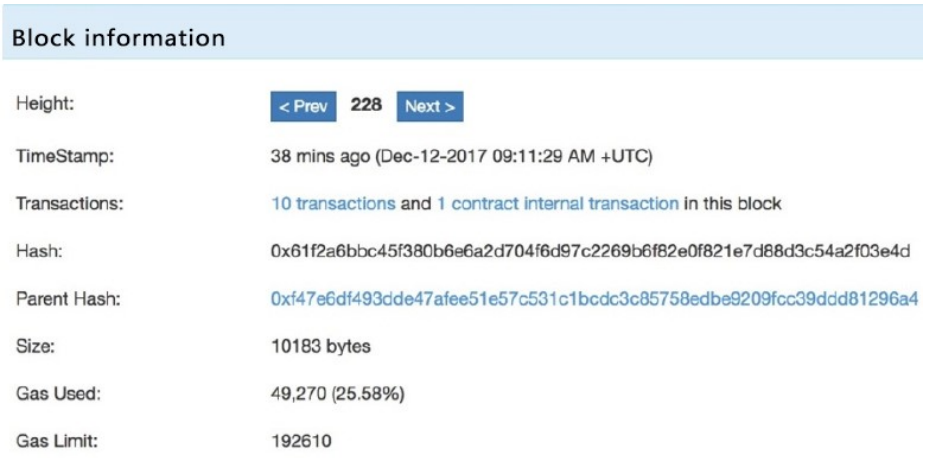


Fig. 7. Block information in the application backend

6 Conclusion

With the development of the information age, intelligent and digital related technologies are realized, and Smart contracts based on Blockchain are one of the most promising technologies among the currently achievable technologies with the help of this development. Smart contracts can not only be applied to more complex application scenarios and more advanced functional requirements, but also Smart contracts can provide an effective way for disciplines such as big data.

Smart contracts are still in a preliminary stage, and there is a lack of further exploration of some key technologies and theoretical knowledge, as well as some social and legal challenges brought about by them in reality are not taken into account, this paper only introduces and summarizes the smart contract technology

itself, and verifies the actual effect of smart contract technology in application through a specific case, aiming to provide some reference for scholars in related fields. This paper only introduces and summarizes the smart contract technology itself, and verifies the practical effect of smart contract technology in application through a specific case, aiming to provide some reference and reference for scholars in related fields.

References

1. B.J., Zhang, Y., Guo, K.W., Hu, K.: Research on data sharing incentive mechanism based on smart contract. Computer Engineering (048-008) (2022)
2. Das, D., Banerjee, S., Chatterjee, P., Biswas, M., Biswas, U., Alnumay, W.: Design and development of an intelligent transportation management system using blockchain and smart contracts. Cluster Computing 25(3), 1899–1913 (2022)
3. Fang, L.: A new interpretation of smart contracts for contracts under blockchain technology. Journal of Chongqing University (Social Science Edition) 27(5), 14 (2021)
4. Gai, K., Wu, Y., Zhu, L., Zhang, Z., Qiu, M.: Differential privacy-based blockchain for industrial internet-of-things. IEEE Transactions on Industrial Informatics 16(6), 4156–4165 (2019)
5. Gurgun, A.P., Koc, K.: Administrative risks challenging the adoption of smart contracts in construction projects. Engineering, Construction and Architectural Management 29(2), 989–1015 (2021)
6. Hauck, R.: Blockchain, smart contracts and intellectual property. using distributed ledger technology to protect, license and enforce intellectual property rights. Legal Issues in the Digital Age 1(1), 17–41 (2021)
7. He, H.W., Yan, A., Chen, H., Z.: A review of blockchain-based smart contract technologies and applications. Computer Research and Development 55(11), 15 (2018)
8. Hewa, T.M., Hu, Y., Liyanage, M., Kanhare, S.S., Ylianttila, M.: Survey on blockchain-based smart contracts: Technical aspects and future research. IEEE Access 9, 87643–87662 (2021)
9. Huanhuan, T.: Research on smart contract technology and application based on blockchain. Computer Programming Skills and Maintenance (003), 000 (2022)
10. Hunhevicz, J.J., Motie, M., Hall, D.M.: Digital building twins and blockchain for performance-based (smart) contracts. Automation in Construction 133, 103981 (2022)
11. K, H., XM, B., LC, G., AQ, D.: A formal verification method for smart contracts. Information Security Research 2(12), 10 (2016)
12. Kirli, D., Couraud, B., Robu, V., Salgado-Bravo, M., Norbu, S., Andoni, M., Antonopoulos, I., Negrete-Pincetic, M., Flynn, D., Kiprakis, A.: Smart contracts in energy systems: A systematic review of fundamental approaches and implementations. Renewable and Sustainable Energy Reviews 158, 112013 (2022)
13. Kolvart, M., Poola, M., Rull, A.: Smart contracts. In: The Future of Law and etechnologies, pp. 133–147. Springer (2016)
14. Liu, Y., Zhou, Z., Yang, Y., Ma, Y.: Verifying the smart contracts of the port supply chain system based on probabilistic model checking. Systems 10(1), 19 (2022)

15. Qiu, H., Qiu, M., Memmi, G., Ming, Z., Liu, M.: A dynamic scalable blockchain based communication architecture for iot. In: International Conference on Smart Blockchain. pp. 159–166. Springer (2018)
16. Shiyi, L., Lei, Z., Desheng, L.: A review of blockchain-based smart contract application research. Computer Application Research 38(9), 12 (2021)
17. Sigalov, K., Ye, X., König, M., Hagedorn, P., Blum, F., Severin, B., Hettmer, M., Hückinghaus, P., Wölkerling, J., Groß, D.: Automated payment and contract management in the construction industry by integrating building information modeling and blockchain-based smart contracts. Applied sciences 11(16), 7653 (2021)
18. Tian, Z., Li, M., Qiu, M., Sun, Y., Su, S.: Block-def: A secure digital evidence framework using blockchain. Information Sciences 491, 151–165 (2019)
19. Wu, Y., Li, J., Zhou, J., Luo, S., Song, L.: Evolution process and supply chain adaptation of smart contracts in blockchain. Journal of Mathematics 2022 (2022)
20. X., Y.: Research and implementation of blockchain-based smart contracts. Ph.D. thesis, Southwest University of Science and Technology
21. Xin Yuhong, Ran Cheng, L.D.: A review of blockchain smart contract technology applications. Modern Information Technology
22. Xiong, X.: Research on blockchain smart contract technology. Ph.D. thesis, University of Electronic Science and Technology
23. Y, Y., Feiyue, W.: Status and outlook of blockchain technology development. Journal of Automation 42(4), 14 (2016)
24. Yuan, X.: Blockchain smart contract technology application. China Finance (6), 2 (2018)
25. Zhao F, L.X., Tan JC, G.K.: Smart contract security issues and research status. Information Technology and Network Security 40(5), 7 (2021)

Research on Power Border Firewall Policy Import and Optimization Tool

Chen Zhang[1]*, Dong Mao[1], Lin Cui[2], Jiasai Sun[1], Fan Yang, [1] Cong Cao[3]*

[1]State Grid Zhejiang Electric Power Corporation Information & Telecommunication Branch, Hangzhou, China
[2]Beijing Guowang Xintong Accenture Information Technology Company, Beijing, China
[3] Hangzhou Innovative Institute, Beihang University, Hangzhou, China
chzhangxd@163.com,641743819@qq.com,
magicmaster356a@gmail.com, sunjiasai@126.com,
yang_fantin@foxmail.com, caocong0419@163.com

Abstract. At present, the security strategy of the lower boundary protection equipment of the power system can no longer meet the needs of the current business growth. A large number of redundant strategies cause the protection performance of the boundary firewall to decline. At the same time, the large number of business growth causes the network boundary order of the power grid system to be blurred. In order to prevent the paralysis and partial collapse of the network and ensure the reliability and integrity of the power business data and enterprise information, this paper develops a smart border firewall optimization tool. This tool can not only integrate the security device policies of different manufacturers through *Simple Policy Specification Description Language* (SPSDL), but also prioritize security rules according to the frequency of use through keyword filtering algorithms and rule optimization decision trees, then realize the classification, streamlining, optimization and upgrading of firewall security rules. The research results show that the power system firewall can achieve an accuracy rate of more than 90% when the strategy is imported. The rule optimization part can reduce the unique correlation addition index of this paper to about 0.2, which solves the problem of firewall security strategy import language diversification. It further eases the pressure of firewall policy redundancy under the power system.

Keywords: Blockchain, Food Safety, Traceability technology, Information dentification, Consensus mechanism.

1 Introduction

With the upgrading and the improvement of the software [1-2], hardware [3-4], and network [5-7], the security demand for power grid business is diversified and explosive, and the relevant security hardware equipment, software systems, and alarm emergency protection measures are constantly upgraded and improved. The traditional network architecture mainly considers the communication requirements of business applications [8], simply implements static security policies [9] in the aggregation layer and access layer and does not consider the security requirements of terminal devices

for multi-source requests in different operating environments. To solve this problem, the security protection of firewall boundary devices uses an ECA (*Enforcement Object/Controlled Object/Action*) based policy description language to uniformly normalize the policies of various types of firewall network security devices into a standard format, and realizes real-time control of terminal device traffic through policy account building, grouping classification, log analysis, etc. It has become an effective security policy management mode [10].

At present, the research on firewall boundary devices mainly includes filtering optimization of security policies, quantification of firewall performance indicators and application of distributed firewalls. As early as 1997, Lupu and Sloman published a conflict optimization analysis on firewall security policies at the IFIP/IEEE International Symposium. Based on this research [11], Bartal et al. developed the Firmato Firewall Management Toolkit [12] in 1999, realized the ternary separation management of security policies, network topology, and firewall hardware devices, and automatically generated firewall configuration files for multiple gateways and implemented advanced debugging. However, on the rule set with more than 20 host groups, firewall security policy rules are as dense as "spaghetti"; Moreover, Firmato does not allow users to control the order of rules, which makes it impossible for users to achieve finer control. To enable users to sort out firewall security rules, Hu et al. proposed a firewall framework based on rule segmentation, which effectively improved nearly 70% of rule conflicts through the network packet space segmentation and redundancy elimination mechanism defined by the firewall [13].

In addition, Han Guolong, Wang Wei and Sheng Honglei took the firewall of Tianjin Electric Power Company as an example in 2018 to propose an optimization scheme for expired rules, duplication and unreasonable configuration, and introduced practical application scenarios to expand the way of strategy optimization [14]. In 2021, Michigan State University proposed a cooperative firewall security policy framework VGuard based on *Virtual Private Networks* (VPNs), which processes data packets through Xhash (XOR and Secure Hash functions are parallel) and decision graph, and the processing efficiency is much higher than the linear search mode. The privacy and security of data packets are also guaranteed through VGuard. However, VGuard is a product based on VPC and lacks certain adaptability [15]. With the development of artificial intelligence and deep learning, in 2022, Journal of Shaanxi University of Science and Technology proposed the application of a security policy tool based on *Improved Mayfly Algorithm* (IMA) in the firewall [16] and optimized the parameters in the *Support Vector Machine* (SVM) firewall configuration model by training and testing the firewall dataset.

2 Power System Boundary Firewall Security Architecture

In order to realize the multidimensional analysis of the security strategy of the multi-node boundary firewall under the distributed power network information system and to clearly and accurately grasp the real-time state of the boundary equipment, this paper constructs the security boundary firewall model according to Figure 1.

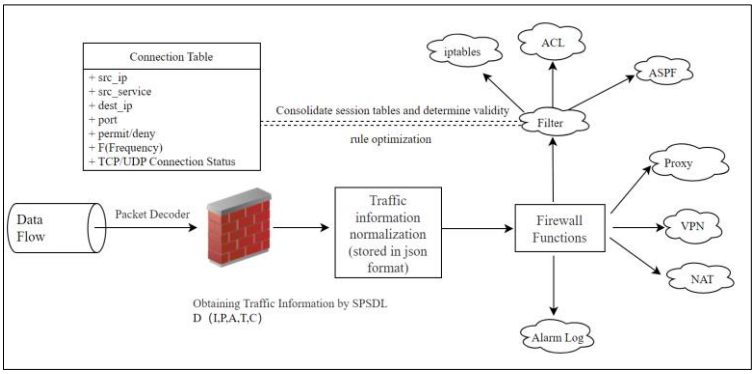


Figure 1: Border Security Firewall Architecture

Considering the intelligent filtering and standardized parsing of external network traffic information and the merging and de-duplication of border device security policies [17], this paper uses Simple Policy Specification Description Language (Simple Policy Specification Description Language, SPSDL and DFA algorithms filter key information, and combine with decision tree algorithm to merge, de-duplicate and classify firewall security policies. Finally, key information in the security policy (SRC_IP, DEST_IP, PORT, and ACTION) is selected to determine the functional dimensions of the firewall security policy architecture, including the import accuracy, correlation addition coefficient, rule scanning time, and coverage rate of key information.

Before the policy management of the firewall at the security boundary of the power network information system, it is necessary to interpret the extensibility and formalize the traffic data of the A-node and use the multivariate group theory to reify the traffic data as the set D(I,P,A,T,C), where *I* represents the source IP address set of the traffic data, *P* indicates the source port set of the traffic data, *A* indicates the protocol (TCP/UDP) carried by the traffic data, *T* indicates the sending time of the traffic data, and *C* indicates the specific content of the traffic data. Based on this, the multiple sets of traffic data are transferred to the firewall of each security boundary node under the power network system. The boundary firewall combines the existing policy library to compile the traffic set into an extensible language for formal processing. Different from the traditional firewall protection system, the intelligent boundary firewall under the power network system can monitor the network traffic in real time through the log storage detection system (Splunk, ELK, etc.) during the use of the application platform [18]. Combined with the intelligent security policy differentiation mode under the boundary firewall, the data packet can be filtered more efficiently. The boundary firewall architecture adopted in this paper will regularly apply:

$$BOOL(E,T,L) \tag{1}$$

Verify the effectiveness of the firewall policy. Where, E indicates whether the policy belongs to the policy library, T indicates the policy life cycle, and L indicates whether the policy log exists. Any Boolean value in the three rules is false, and the overall Boolean value is false. The firewall policy will no longer be effective. Aiming at the classification of intelligent border firewall strategies in power network system, this paper

uses key information retrieval and matching algorithm to filter data packets and extract key information. Then match the firewall policies and remove the duplication.

3 Method

3.1 Design of Security Policy Import Module for Border Firewall

The Overall Architecture of the Border Firewall Import Module. The main content of the key information filtering algorithm constructed in this paper is to build a firewall security policy library, and combine the original data flow to match and retrieve the security policy information. Finally, the key information is filtered out and finally output to the rule optimization function module in the format of Json data flow. The specific construction mode is shown in Figure 2:

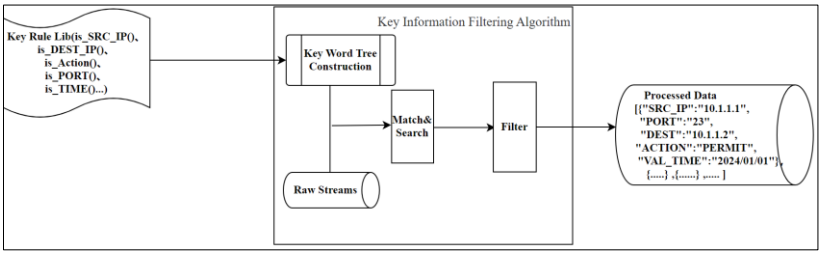


Figure 2: Security Boundary Firewall Import Module

As there are many types of firewall network security devices, the current policy description language cannot meet the description of the policy. If a new firewall device type is added, its policy cannot be quickly imported and analyzed, resulting in poor flexibility and scalability [19-21]. Therefore, through the simple policy description language SPSDL and the key information matching algorithm, a Json format based formalization mechanism is proposed, which can realize the unified formalization of various types of firewall network security device policies into the standard format under the condition of the complex and changeable firewall network devices, so as to be imported into the system, laying a foundation for policy analysis.

Policy Standardization of Border Firewall. In order to solve the problem of policy description, this paper designs a Simple Policy Specification Description Language (SPSDL) based on ECA (*Enforcement Object/Controlled Object/Action*).

SPSDL morphology. Morphology is the basic grammatical unit of language, which has definite meaning and plays various roles in policy compilation. How the words of a language are classified, divided into several categories, depends mainly on the convenience of processing. In SPSDL language will be divided into four categories:

1) key words: also called the reserved word. These words have a fixed meaning in the SPSDL.

2) operator: including logical operators, assignment operator, etc.

3) constants: such as digital constant, Boolean constants, character constants, etc.

4) delimiter: such as ";" , "{","} ", etc.

Application of key information matching algorithm in security policy import module of border firewall. Many accurate key information retrieval and matching algorithms are based on deterministic finite automation (DFA). However, the key information of the security policy is changeable, and there are many key information dimensions to be retrieved, and the algorithm and time complexity are high, which cannot be completed through the retrieval sensitive word matching algorithm. This article is based on the following strategies:

$$S:\left\{\left\{S1:r1,r2..,r_i\right\},\left\{S2:r_{i+1},r_{i+2}..,r_{i+j}\right\}...\right\} \tag{2}$$

Key information such as IP address, port, action and time is found by combining lexical analysis, syntax analysis and function judgment such as IP and PORT, and finally exported to the policy optimization module in JSON format. The algorithm simplifies the complexity of the model and improves the running speed.

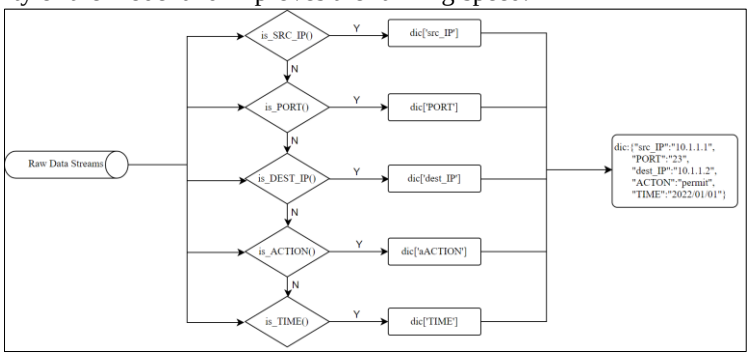


Figure 3: Key information matching algorithm implementation model

3.2 Rule optimization of firewall strategy in power system

The security policy of the boundary firewall under the power system integrates a series of rules and information, which define the operations to be performed on the specified data packets and are finally presented in the form of<"Condition", "Action">. The conditions in the rule consist of a set of fields used to identify the matching of specific types of packets (such as src_ip, dest_ip, src_port, dest_port, protocol, TCP/UDP, Expiration, Explanation, Log, etc.).

Judgment on the validity of rules for flow control. In the process of matching and optimizing traffic rules of firewall security policy, the unilateral relationship of rules is analyzed first. You only need to scan the rule column once, and then scan the rule's time parameter, comment parameter, generation log flag, and effective flag in turn. For example, check whether the comment is empty, and whether the calculation time parameter is expired. For each rule, the corresponding analysis results can be obtained without relying on the information of other rules. Therefore, context related parameters such as priority need not be considered. The specific process is shown in Figure 4.

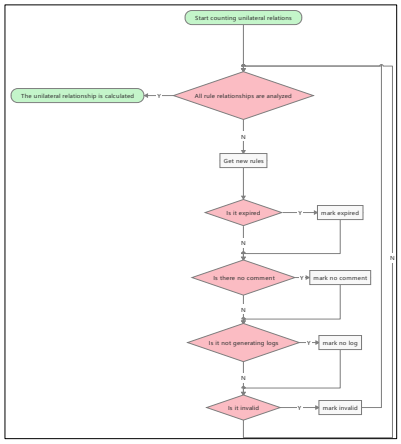


Figure 4: Rule Valid Judgement

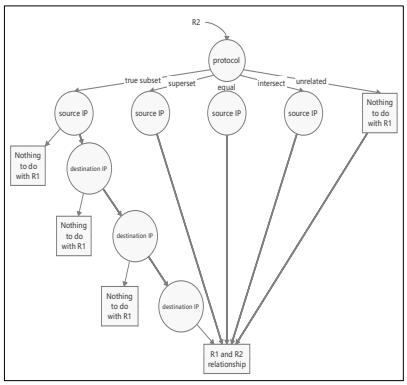


Figure 5: Rule Optimization Decision Tree

Judge the pairwise relationship between security policy rules in real-time traffic according to the decision tree. Next, the correlation analysis of firewall security policy traffic rules is carried out. The identified key information is shown in Table 1.

Id	*Src_IP*	*Dest_IP*	*Service*	*Action*	*Frequency*
1	*A*	*B*	*C*	permit	f_1
2	any	any	any	permit	f_2

Table 1: Key Info Query Table

The relationship analysis of rules is generally divided into irrelevant, intersecting, subset or exclusive relationships. If there is no association between the two rules, the next step will not be processed; If the two rules are intersected or included, the union is used; If the relationship between the two rules is mutually exclusive, select the one with higher priority for the next action after judging the priority. Due to the existence of priority, if the actual matching traffic of a high priority rule is small, it will result in a large number of matching attempts. Therefore, moving the rules with high matching frequency to the high priority direction as far as possible can reduce the cumulative matching frequency. Obviously, this also requires the matching information of the traffic and rules previously analyzed. Due to the existence of the rule security zone, in order to ensure that moving the rule location will not cause changes in the actions of the security device when filtering packets, the movable location of the rule will be limited within its security zone. The rule optimization decision tree is shown in Figure 5.

Regularly Identify and Update Flow Control Rules. For the formed policy rule base, it is necessary to regularly check the relationship and effectiveness of rules. At first, judge the validity of all rules in the firewall policy rule library, delete invalid rules, and repeat the previous operations for valid policy rules, as shown in the following figure:

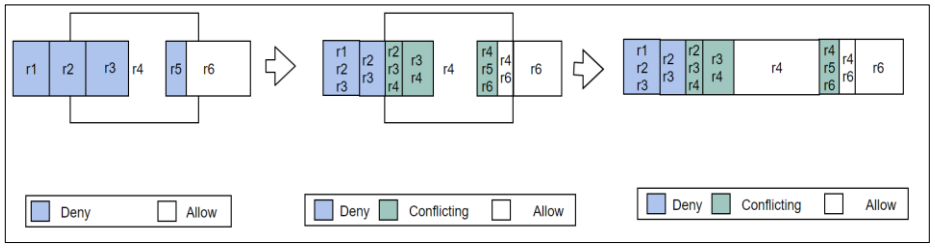


Figure 6: Rule Optimization Flow Chart

Finally, the correlation coefficient *Cor ()* between two pairs of rules is used to determine whether all rules have been processed. If all correlation coefficients are zero, the processing is completed; otherwise, continue to process the rules with non-zero correlation coefficients until the correlation coefficients cannot be reduced. The processing table is shown below. If the following conditions are achieved, the scanning and updating will be completed.

Regular_update_time_T	*Rule_Validation_BOOL(E,T,L)*	*Is_Cor(r1&r2,r1&r3…ri&rj)*
Date	True	False

Table 2: Rule Validation Judgement Table

4 Calculation

Throughput, number of concurrent connections, new connection speed, delay, packet loss rate, data backup speed, system recovery time, etc. can all be used as the main performance test indicators of the security boundary firewall. This paper focuses on the normalization import of firewall security boundary policy information and the optimization of security policy rules. Therefore, the accuracy rate of key information imported by security boundary firewall policy, the running time of policy import, the coverage rate of policy key information, the rule scanning time, the rule priority and correlation addition indicators are selected as the experimental evaluation results.

Border Firewall Security Policy Import Module. The import effect of key information is affected in many ways. This paper mainly evaluates the import effect of the security policy import module through the import time of key information, the coverage of key information in all policies, and the accuracy of relevant information when importing into the rule base.

1) Key Information Import Time

The import duration of key information is mainly aimed at the time when the firewall boundary filters out IP, port number, action and other related information in all policy databases. After verification, Figure 7 shows the time required to filter key information such as IP, port and action from 8-1000 firewall security policies.

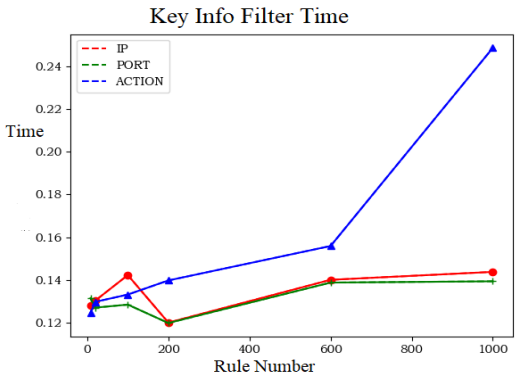


Figure 7: Filter time of IP、PORT、ACTION

2) Key Information Coverage

To ensure the efficiency of the border firewall security policy import module, this paper also counts the frequency of key information, that is, key information coverage. The calculation formula and process are as follows

$$F = \frac{1}{N}\sum_{i=1}^{N}\frac{k}{n_i} \tag{3}$$

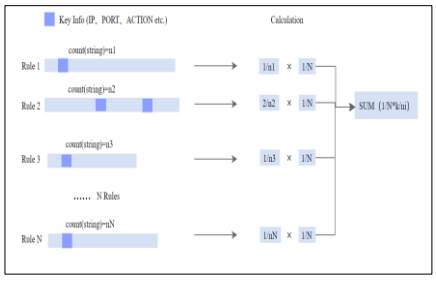


Figure 8: Key info. Coverage Rate

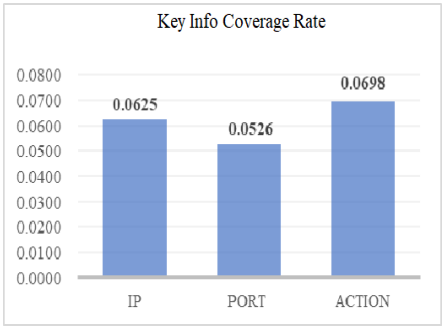


Figure 9: Key Info. Coverage Rate Result

Through the calculation of the above process, the frequency of key information such as IP, port and action of the security boundary firewall is calculated as shown in Figure 9. It can be seen that the frequency of the ACTION field is relatively high, and the error of the ACTION field in the statistical process is minimal.

3) Key Information Import Accuracy

In the border firewall based security policy import module, the key information matching algorithm also counts the matching accuracy of key information. The experimental results show that more than 90% of the key information can be matched successfully, and the identification error is mainly due to the confusion of subnet mask and IP address. In addition, according to the statistical results, the source IP address and actions appear frequently in the security policy, so they have high priority. The specific results are shown in Table 3:

	SRC_IP	*DEST_IP*	*PORT*	*ACTION*	*TOTAL*
Match_Accuracy	0.9091	0.9091	0.9231	1.00	0.9353
Error_Rate	0.0909	0.0909	0.0769	0.00	0.0647
Frequency_f	0.7273	0.2727	0.5455	0.8182	

Table 3: Key Info. Match Accuracy

Evaluation Index of Policy Rule Optimization Module of Boundary Firewall.

1) Correlation Bonus Indicator

To judge the performance of firewall policy rules based on traffic control, it is necessary to consider whether the correlation indicators in the firewall policy rule library meet the standards. Therefore, the correlation plus indicator S is introduced:

$$S = Cor1(r_1, r_2) + Cor2(r_1, r_3) + ... + Cor_{\frac{n\times(n-1)}{2}}(r_n, r_{n-1}) \quad (4)$$

When S is close to 0, it indicates that the rules in the firewall policy rule base are uncorrelated. At this time, the simplification of the firewall policy rule base reaches the optimal state.

2) Rule Priority Indicator

The priority of rules in the rejected domain is often higher than that of rules in the accepted domain. Based on this, the rule priority needs to be determined according to frequency f and action A. The specific calculation is as follows:

$$P = k_1 \times f + k_2 \times A + \varepsilon \quad (5)$$

3) Performance Analysis of Rule Optimization

In this paper, 55 rules in the security policy library are randomly selected for performance optimization analysis and the final results are shown in Table 4:

	Priority_P	*Correlation_S*
R1-R11	R11	0.2310
R12-R22	R17	0.3001
R23-R33	R31	0.2945
R34-R44	R39	0.1453
R45-R55	R53	0.2247

Table 4: Rule Scan Condition Table

The results show although the rules are deduplicated and optimized, some rules cannot be completely irrelevant, so the optimization algorithm needs to be further improved

5 Conclusion

In order to solve the problem of identification and standardization of diversified security policy description languages, this paper adds a security policy description language based on policy knowledge base and completes the policy normalization import. In addition, aiming at the redundancy, invalidity and all pass problems of firewall security

policies, a group of firewall policy analysis and optimization algorithms are designed to help administrators automatically analyze redundant, conflicting, overlapping, loose and other policy problems, solve the boundary leak caused by firewall configuration problems, reduce the security operation and maintenance costs, and improve the border protection capability.

References

1. L. Tao, S. Golikov, et al., "A reusable software component for integrated syntax and semantic validation for services computing", IEEE Sym. SOSE, 127-132, 2015
2. J. Li, Z. Ming, et al., "Resource allocation robustness in multi-core embedded systems with inaccurate information", JSA, 57 (9), 840-849, 2011
3. M. Qiu, Z. Ming, et al., "Three-phase time-aware energy minimization with DVFS and unrolling for chip multiprocessors", Journal of Systems Arch., 58 (10), 439-445,2012
4. M. Qiu, H. Li, E. Sha, "Heterogeneous real-time embedded software optimization considering hardware platform", ACM sym. on Applied Comp., 1637-1641, 2009
5. M. Qiu, C Xue, Z Shao, et al., "Efficient algorithm of energy minimization for heterogeneous wireless sensor network", IEEE EUC, 25-34, 2006
6. J. Niu, Y. Gao, M. Qiu, Z. Ming, "Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability", JPDC, 72(12), 1565-1575, 2012
7. F. Hu, S. Lakdawala, et al., Low-power, intelligent sensor hardware interface for medical data preprocessing, IEEE Trans. on Infor. Tech. in Biomedicine 13 (4), 656-663, 2009
8. LIU Jian-bing, MA Xu-yan, WANG Xiao-hong, WANG Zhen-xin. Security policy of active security network architecture[J]. *Information Security Research*, 2021,7(11):998-1006.
9. HAN Ming. Internal Network Security Policy, *Info. and Comp.* 2018(04):157-158.
10. YAN Zhao-zhi. Optimization of communication network firewall strategy [J]. *Computer Knowledge and Technology* ,2021,17(07):46-47+53.DOI:10.14004/j.cnki.ckt. 2021.0724.
11. Lupu E, Sloman M. Conflict analysis for management policies, *Int'l Symp. on Integrated Network Management*. Springer, Boston, MA, 1997: 430-443.
12. Bartal Y, Mayer A, Nissim K, et al. Firmato: A novel firewall management toolkit[J]. *ACM Transactions on Computer Systems (TOCS)*, 2004, 22(4): 381-420.
13. Hu H, Ahn G J, Kulkarni K. Detecting and resolving firewall policy anomalies[J]. *IEEE Transactions on dependable and secure computing*, 2012, 9(3): 318-331.
14. HAN Guo-long, WANG Wei, SHENG Hong-lei. Research on Firewall Strategy Sorting and Optimization Method[J]. *Electric Power Infor. and Comm. Tech.*, 2018,16(06):31-35.
15. Liu A X, Li R. Collaborative enforcement of firewall policies in virtual private networks[M]//*Algorithms for Data and Computation Privacy*. Springer, 2021: 139-170.
16. GAO Z., ZHANG Y., et al. Improved mayfly algorithm and its application in firewall policy configuration. *J. of Shaanxi U. of Sci. and Tech (Natural Sci. E.)* ,2022,38(02):41-48.
17. LIU Kun-can. Research and Implementation of Firewall Deep Packet Detection Technology[D]. Beijing University of Posts and Telecommunications,2013.
18. REN Zhan-rui. Research on key technologies of firewall security policy configuration[D]. National University of Defense Science and Technology,2011.
19. CHEN Xin-hao. Analysis of security policy conflicts in multi device firewalls[J]. *Computer CD Software and Application*,2012(02):104+102.
20. DENG Wen-jun, LIANG Yi-wen. Semantic analysis method of firewall security policy[J]. *Computer Engineering and Application*,2007(26):135-137.
21. WANG Bo. Active Security Policy Firewall Based on Honeynet[D]. Beijing University of Posts and Telecommunications,2010.

Cross-chain-based Distributed Digital Identity: A Survey

Tianxiu Xie[1,3], Hong Zhang[2,3], Yiwei Feng[2,3], Jing Qi[1], Chennan Guo[4], Gangqiang Yang[5], and Keke Gai[1]

[1] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China.
{3120215672, 3120221282, gaikeke}@bit.edu.cn
[2] School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China.
{3220211142, 3220211128}@bit.edu.cn
[3] Yangtze Delta Region Academy, Beijing Institute of Technology, Jiaxing 314000, China
[4] School of Softare, Dalian University of Technology, Dalian 116081, China.
guochennan@dlut.edu.cn
[5] School of Information Science and Engineering, Shandong University, Shandong 266237, China.
g37yang@sdu.edu.cn

Abstract. Due to the high degree of privacy and sensitivity, it is difficult to share distributed digital identity with multiple parties. Blockchain-based distributed digital identities could address the issues of data sharing. However, the blockchain systems contain a wide variety of heterogeneous autonomous systems, and each blockchain system performs independent identity authentication. To be specific, it is difficult to realize mutual recognition and trust in digital identity. In this survey, we introduce the technical architecture of distributed digital identity and research the technical principle and development application of cross-chain technology. In addition, we expound the research status of cross-chain technology and the cross-chain-based distributed digital identity mechanism to realize unified and trusted cross-chain identity mutual recognition.

Keywords: Distributed digital identity · Cross-chain technology · Blockchain · Relay chain.

1 Introduction

With the continuous enrichment and complexity of the application scenarios of the Internet, activities in the physical world are increasingly dependent on digital identities. Digital identity can support users to obtain different network services. However, in the traditional digital identity situation, a user has different digital identity accounts in multiple application systems, which will lead to decentralized information management and complicated identity authentication. In order to make digital identity more secure and complete, the *World Wide Web* (W3C)

proposes *Distributed Identities* (DID). Due to the characteristics of security, credibility and decentralization [1] [2], the blockchain-based DID can adapt to the new Internet system Web3.0 [3]. In addition, DID fundamentally addresses the issues of large-scale and messy user identities and limited data sharing in the Internet. As the key to trusted interaction, the blockchain-based DID is of great significance in interconnection and privacy identity management.

Distributed digital identity not only facilitates digital services, but also makes the demand for data sharing and application collaboration among blockchains increasingly apparent. The interoperability and scalability have always limited the large-scale application of blockchain [4]. Specifically, cross-chain technology facilitates the wider application of DID. At present, due to the independence of blockchains in different application fields [5] [6], distributed identity information is in an isolated state, which violates the original intention of distribution. Therefore, cross-chain technology is introduced into distributed identity. Cross-chain-based DID can realize mutual identity recognition, identity transfer and multi-party authentication, thereby promoting mutual trust and efficient collaboration of different blockchain systems.

Currently, the cross-chain-based DID faces an important challenge of ensuring the authenticity, validity, and legitimacy of the information source. Cross-chain identity authentication requires secure access to identity data and user permissions, making the process highly secure and reliable. Specifically, distributed digital authentication requires identity authentication and two-way verification. Therefore, the cross-chain-based DID technology needs to provide authentication services for different certificate domains, so as to realize the authentication of various digital certificates. The application of cross-chain technology in DID will bring about the security and credibility of identity data cross-chain.

In order to deal with these challenges, we researched a relay chain-based distributed digital identity. We proposed a *Cross-chain Channel-based DID* (C3-DID) model that enabled distributed storage of DIDs and was resistant to multiple threats [7]. In addition, we proposed a relay chain-based cross-chain system that achieved asynchronous consensus and improved the defensiveness and scalability of the cross-chain system [8].

The rest of the paper consists of the following sections. Section 2 gives the architecture and functionality of distributed digital identities and shows related research and landing projects. Section 3 introduces cross-chain related technologies. In addition, in Section 4, we present the existing research and project of the cross-chain-based distributed digital identity. Finally, Section 5 gives a summary and outlook.

2 Distributed Digital Identity

Distributed Identity (DID), as defined by the W3C distributed digital identity specification, is a new globally unique identifier. This identifier can be used not only for people, but also for everything, including a car, an animal or even a machine [9]. The core components of DID technology consist of three parts: the

DID, the DID Document, the Verifiable Credential and the Verifiable Expression. Figure 1 shows the architecture of DID. A DID identification is a string of a specific format used to represent the numerical identity of an entity, which in this case can be a person, machine or object [10] [11].

The *DID Document* (DID Doc) contains all the information related to the DID identity and is linked to the DID by a generic data structure *Uniform Resource Locator* (URL). Moreover, the DID controller is used to write or change data in the DID Doc. The DID document itself cannot represent the real identities of the users, so we provide the *Verifiable Credentials* (VCs) to support identity authentication. The binding of a user-centric identity to other identifiers issued by the *Certificate Authority* (CA) is known as the VC. It also provides a system similar to a PKI [12] [13].

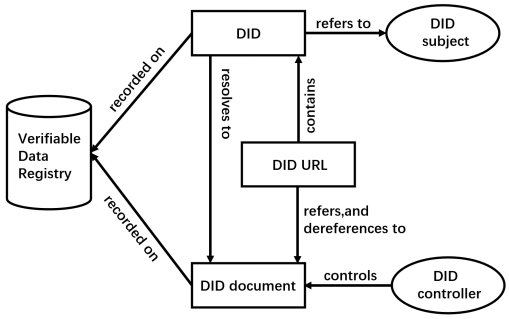


Fig. 1. Distributed Digital Identity Architecture

The distributed identity system has attracted extensive research from scholars. Zhou et al. [14] propose the EverSSDI framework, a distributed identity management system that relies on smart contracts, blockchain and the *Interplanetary File System* (IFS). In this framework, users first encrypt personal information and store them in the IPFS system, then use smart contracts for data verification. Another blockchain-based distributed identity management scheme, DNS-Idm, is proposed in [15]. In DNS-Idm, users and service providers can authenticate and declare identity attributes using real identities, while smart contracts protect the relevant operations among the authenticators and the users in this scheme.

Efat et al. [16] have proposed DT-SSIM, a distributed and trusted framework for self-sovereign identity management. DT-SSIM combines blockchain-based smart contract technology with a secret sharing scheme that can provide a reliable self-sovereign identity-based service for the IoT and can prevent tampering of IoT identity credentials. DT-SSIM uses the secret sharing method and smart contract technology to manage the generated identity shares and verify the user's identity declaration. In addition, DT-SSIM has proposed a series of

novel algorithms to ensure the information security of externally stored identity credentials. It has realized identity verification without disclosing the original identity data. For this framework, future improvements could be directed towards replacing dynamic active secret sharing with a moving target defence mechanism, in which the external sharing of identities could be updated periodically and the storage location dynamically refreshed, a modification and integration that could significantly improve the stability of the framework and increase the cost of attack for attackers.

Deepak et al. [17] have proposed CanDID, a user-friendly and usable distributed identity implementation platform in which users can manage their own certificates. CanDID uses a decentralized committee of nodes to provide a robust solution for users' real identities, while preventing users from generating multiple identities and allowing for the identification of sanctioned users.

One effective technique to address distributed authentication is *Distributed Hash Tables* (DHT) [18], which can perform fast lookups and load identifiers on key-value pairs. Blockchain can be combined with DHT technology for distributed authentication and is used as a verifiable data registry that can store public key DID. With the signature of the issuers, the DID stored on the blockchain can support the authenticity and validity of the credentials.

Digital credentials and digital identities have evolved, and the latest evolution of digital credentials is the VC, which is a digital representation of real-world physical credentials. The validity and portability of physical credentials is transferred to digital devices through encryption algorithms and digital signatures, and the declared content, signature and metadata can be digitally verified in seconds. It is a standardised representation of digital credentials and is designed to bring the benefits of physical credentials to the digital world by referring to real-world physical credentials in terms of usage scenarios and core model design. Typical features of VCs are cryptographic security, privacy protection and machine readability.

Figure 2 shows the VC model proposed by the W3C. There are three main user roles in this model. Issuer is the certificate issuer who can be an individual or an organization. The certificate issuer actually issues a certificate to the public key of the person who will hold the certificate and digitally signs it with his or her private key. In this way, when the certificate holder presents the certificate, it also needs to use a digital signature method to prove that it is indeed the owner of the public key of the person to whom the certificate is issued (a very common method of digital signature verification). The certificate is digitally signed by the certificate issuer, so using the digitally signed verification method, anyone can instantly verify that the certificate is valid, and any tampering with the certificate will result in the invalidation of the digital signature. As the certificate issuer needs to know the holder's public key, it is generally the case that the certificate holder first applies to the issuer, providing his or her public key and proof of digital signature with this public key, as well as other evidence and data required by the issuer when applying. The issuer will verify these data

and after confirming that they are correct, the certificate can be generated and sent to the certificate holder.

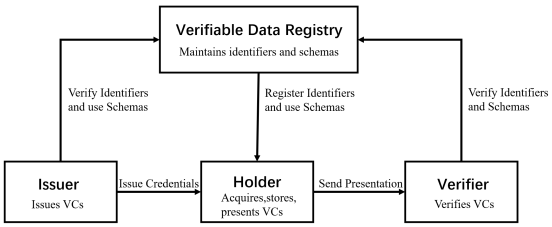


Fig. 2. Validatable Credential Data Model

The certificate holder is responsible for keeping his or her own certificate after the above process has been applied for. Usually these certificates are in digital form and can be downloaded and saved, and some advanced digital wallet systems have full digital certificate storage and management capabilities. In most cases, the responsibility for keeping the certificate rests with the certificate holder and the certificate issuer can decide for itself whether or not to keep a copy of the issued certificate. If the certificate holder loses the certificate, or the private key used to prove their identity, they will have to reapply for a new certificate. When a certificate needs to be verified, the certificate holder is responsible for providing a digital certificate and providing a digital signature to prove that it has the public key of the person to whom the certificate was issued.

The certificate verifier only needs to use the digital signature algorithm to verify the identity of the certificate holder and verify the correctness of the digital certificate to basically determine the authenticity of the certificate and the legitimacy of the certificate holder. Usually the certificate also contains some meaningful information inside, such as the expiry date, which is also used as a basis for determining the authenticity of the certificate.

3 Cross-chain Technology

The interoperability of blockchains limits the large-scale application of blockchains. The cross chain technology is a key technology to achieve the Internet of Value and a bridge for blockchain expansion. Three cross chain technologies were mentioned, notary mechanism, side chain and relay chain and hash locking. In this section, We briefly introduce and compare these cross chain technologies, as shown in Table 1.

Notary mechanism is essentially a way of mediation by introducing a third-party intermediary to verify and transmit cross chain messages between two chains that cannot directly interoperate. By introducing one or more trusted

Cross Chain	Advantage	Disadvantage
Notary mechanism	It is the simplest and most effective form of cross chain technology.	It needs to introduce notaries, which is equivalent to adding a centralized intermediary, and is contradictory to its own concept of decentralization.
Side chain and relay chain	It can complete cross chain asset transfer, exchange, cross contract, mortgage and other applications.	It is difficult from technology.
Hash locking	It can also be used to exchange cross chain assets without mutual trust.	It does not realize cross chain transfer of assets, nor so-called cross chain contracts, but only cross chain exchange.
Distributed private key technology	Users always have control over their assets.	The contract is incomplete and needs further optimization.

Table 1. Comparison of Different Cross-chain Technologies

third parties as credit endorsements, notary mechanism continuously monitors events on the chain, and is responsible for verifying and forwarding cross chain messages on other chains according to the obtained event information. It is a simple solution to achieve blockchain interoperability. It does not require complex proof of workload or proof of equity, and is easy to interface with existing heterogeneous chains. However, its disadvantage is that it can cause asset exchange and has the problem of over concentration.

As for side chain and relay chain technology, the side chain is referred to the blockchain which is parallel to the main blockchain. The side chain implementation is to lock the temporary digital currency on the main chain and release the equivalent digital assets on the side chain through the two-way pegging technology. Relay chain is the role of "intermediary", which is the integration and expansion of notary mechanism and side chain mechanism [19]. From the perspective of form, relay chain is a way, and side chain is a result. Side chain expresses the relationship between two chains, not a cross chain technology or scheme.

Due to the unidirectional and low collision of hash function, Hash locking is a mechanism that takes advantage of the delayed execution of transactions in the blockchain. In order to realize the asset security on each blockchain, for the asset transaction process, hash locking needs to ensure the transaction atomicity, that is, asset transactions are either completed or not completed [20]. A contract with a hash locking mechanism is used to lock assets to achieve pledge effect, providing a trust basis for transactions between different assets. The advantage of hash lock is that if the transaction fails due to various reasons, the time lock can enable all parties involved in the transaction to recover their own funds to avoid losses caused by fraud or transaction failure.

Distributed private key technology [21] is a technology based on multi-party computation and threshold keys in cryptography. It separates the use right and ownership of digital assets, so as to realize the decentralization of assets control rights. The assets on the original blockchain are safely mapped to each blockchain system, thus realizing the cross chain circulation and value transfer of assets among multiple blockchain systems.

Early cross chain technologies such as BTCRelay [22], they focus more on asset transfer. Interleader protocol first proposed by Ripple Lab [23] allows two different accounting systems to freely transmit currency to each other through a third-party "connector" or "verifier". It is applicable to the accounting systems of various blockchains and can accommodate their differences, so it can be used as a unified payment standard. BTC Relay connects Ethereum network and Bitcoin network through Ethereum smart contract, enabling users to verify Bitcoin transactions on Ethereum [24]. However, the existing cross chain technologies, represented by Polkadot and Cosmos, pay more attention to cross chain infrastructure. Polkadot integrates a variety of blockchain technologies of different public blockchains, and various public blockchains can communicate with each other by any message and exchange tokens. Cosmos provides convenience for application developers to use their own public blockchains by providing modular blockchains. This mode is very suitable for public blockchains focusing on vertical fields. The emerging FUSION realizes multi currency smart contracts, which is a public blockchain with great application value and can generate rich cross chain financial applications.

A large number of blockchain systems with different characteristics have formed a large number of value islands, and direct value circulation cannot be carried out among blockchains. It limits the functional expansion and development space of the blockchain. The security concerns observed on centralized cryptocurrency exchanges motivated the design of atomic swaps, which apply to the exchange of money between any two users. However, there is currently no atomic swap protocol that is both universal and multi-asset, that is, compatible with all cryptocurrencies and supports the exchange of multiple coins in a single atomic swap. [25] introduced a general currency exchange protocol for securely exchanging multiple coins from any currency to multiple currencies of any other currency, for any kind of currency, in addition to the ability to verify signatures, the protocol no need to rely on any special scripts in the blockchain.

4 Relay chain-based Identity Cross-chain Technology

The relay chain is to establish an additional chain between the source chain and the target chain to complete the verification and execution of cross-chain transactions. The nodes in the relay chain are deployed in various blockchain networks, and the information of initiating cross-chain transactions is synchronized at all times. After the user initiates a cross-chain transaction in the source chain, the relay node will forward the information to the relay chain and verify the transaction data. After the verification is completed, the corresponding

transaction will be constructed through the consensus node in the relay chain, and complete the signature, and finally transport it to the target chain through the relay node.

BitXHub adopts the relay chain to provide safe and efficient cross-chain services, and solves the core problems of capturing, transmitting and verifying cross-chain transactions. It focuses on the interoperability of ledgers between heterogeneous and homogeneous consortium chains, and supports asset exchange, data interoperability and service complementarity. XuperChain has implemented a set of trusted digital identity solutions, which aims to connect multiple industries and promote cross-institution and multi-scenario identity authentication and data cooperation. *TencentCloud Decentralized Identity* (TDID) provides infrastructure for trusted digital identity and data exchange services across systems and institutions. Based on the blockchain, TDID provides a mechanism for distributed generation, holding and verification of identity identifiers and VC that carry identity data to encrypt security, protect data privacy. It can reliably express various types of identities and credentials in the real world on the Internet in a way that can be machine-verified by a third party.

The use process of digital identity in BitXHub in practical application scenarios is as follows.

1. Apply for a DID: Initiate a request to the management relay chain and send the DID name;
2. Approval the DID: The administrator approves the user's application request, and the result is "approved" or "rejected";
3. Register the DID: After approval, the user can register the DID. First, organize the relevant information of the chain identity into DID Doc for storage to obtain the storage address docAddr and content hash docHash. Then initiate a registration request to the highest relay chain with docAddr and internal docHash as parameters;
4. Parsing the DID: The information of theDID is synchronized on all relay chains, so DID parsing can be initiated to any relay chain. First, the client initiates a resolution request to any relay chain, and obtains the relevant information of the identity. Then obtain the actual DID Doc through the storage address, perform hash verification, and if it passes, it proves that the DID Doc is credible.

5 Conclusion

The scalability of blockchain has always been a research focus. The early blockchain technology was developed with an independent single chain, but due to the limitations of the efficiency and performance of the single chain, it could not support the application requirements of Web3.0. Therefore, the single blockchain technology was gradually extended to multi-chain collaborative development. Many cross-chain approaches have emerged. As a solution to maximize the scalability and interoperability of blockchains, cross-chain technology has received widespread attention since the birth of Bitcoin. In this survey, we researched

the recent project on DID and cross-chain technology and analyzed cross-chain-based DID methods and business processes. In the future, we will conduct more in-depth research on the interoperability and robustness of cross-chain technology, the secure transmission of cross-chain data, and cross-chain efficiency.

Acknowledgements

This work is partially supported by the National Key Research and Development Program of China (Grant No.2021YFB2701300), Natural Science Foundation of Shandong Province (Grant No. ZR2020ZD01), and Graduate Interdisciplinary Innovation Project of Yangtze Delta Region Academy of Beijing Institute of Technology (Jiaxing) (No. GIIP2022-019).

References

1. K. Gai, Y. Wu, L. Zhu, Z. Zhang, and M. Qiu. Differential privacy-based blockchain for industrial internet-of-things. *IEEE Trans. on Industrial Informatics*, 16(6):4156–4165, 2019.
2. K. Gai, Y. Wu, L. Zhu, L. Xu, and Y. Zhang. Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks. *IEEE Internet of Things J.*, 6(5):7992–8004, 2019.
3. K. Gai, Y. Zhang, M. Qiu, and B. Thuraisingham. Blockchain-enabled service optimizations in supply chain digital twin. *IEEE Trans. on Services Computing*, 2022.
4. K. Gai, Y. Wu, L. Zhu, K. Choo, and B. Xiao. Blockchain-enabled trustworthy group communications in uav networks. *IEEE Trans. on Intelligent Transportation Systems*, 22(7):4118–4130, 2020.
5. K. Gai, Y. Wu, L. Zhu, M. Qiu, and M. Shen. Privacy-preserving energy trading using consortium blockchain in smart grid. *IEEE Trans. on Industrial Informatics*, 15(6):3548–3558, 2019.
6. K. Gai, H. Tang, G. Li, T. Xie, S. Wang, et al. Blockchain-based privacy-preserving positioning data sharing for iot-enabled maritime transportation systems. *IEEE Trans. on Intelligent Transportation Systems*, 2022.
7. T. Xie, Y. Zhang, K. Gai, and L. Xu. Cross-chain-based decentralized identity for mortgage loans. In *International Conference on Knowledge Science, Engineering and Management*, pages 619–633. Springer, 2021.
8. S. Zhang, T. Xie, K. Gai, and L. Xu. Arc: An asynchronous consensus and relay chain-based cross-chain solution to consortium blockchain. In *2022 IEEE 9th International Conference on CSCloud*, pages 86–92. IEEE, 2022.
9. N. Naik and P. Jenkins. Governing principles of self-sovereign identity applied to blockchain enabled privacy preserving identity management systems. In *2020 IEEE ISSE*, pages 1–6. IEEE, 2020.
10. K. Li, A. Ren, Y. Ding, Y. Shi, and X. Wang. Research on decentralized identity and access management model based on the oidc protocol. pages 252–255, 2020.
11. C. Rupa, R. Patan, F. Al-Turjman, and L. Mostarda. Enhancing the access privacy of idaas system using saml protocol in fog computing. *IEEE Access*, 8:168793–168801, 2020.

12. H. Zhou and L. Zhu. Research and design of cas protocol identity authentication. In *2020 International Conference on CVIDL*, pages 384–387. IEEE, 2020.
13. X. Hu, W. Tan, and C. Ma. Comment and improvement on two aggregate signature schemes for smart grid and vanet in the learning of network security. In *2020 ICISE-IE*, pages 338–341. IEEE, 2020.
14. T.Zhou, X.Li, and H.Zhao. Everssdi: blockchain-based framework for verification, authorisation and recovery of self-sovereign identity using smart contracts. *International Journal of Computer Applications in Technology*, 60(3):281–295, 2019.
15. A. Jamila, S. Sarwar, M. Hector, P. Zeeshan, and D. Keshav. Dns-idm: A blockchain identity management system to secure personal data sharing in a network. *Applied Sciences*, 9(15):2953, 2019.
16. E. Samir, H. Wu, M. Azab, C. Xin, , and Q. Zhang. Dt-ssim: A decentralized trustworthy self-sovereign identity management framework. *IEEE Internet Things J.*, 9(11):7972–7988, 2022.
17. M. Deepak, M. Harjasleen, F. Zhang, N. Jean-Louis, F. Alexander, et al. Candid: Can-do decentralized identity with legacy compatibility, sybil-resistance, and accountability. In *2021 IEEE SP*, pages 1348–1366. IEEE, 2021.
18. M. Alizadeh, K. Andersson, and O. Schelén. Comparative analysis of decentralized identity approaches. *IEEE Access*, 10:92273–92283, 2022.
19. N. Kannengießer, M. Pfister, M. Greulich, S. Lins, and A. Sunyaev. Bridges between islands: Cross-chain technology for distributed ledger technology. 2020.
20. L. Deng, H. Chen, J. Zeng, and L. Zhang. Research on cross-chain technology based on sidechain and hash-locking. In *International conference on edge computing*, pages 144–151, Seattle, WA, USA, 2018. Springer.
21. D.W. Li, J. Yu, X. Gao, and N. Al-Nabhan. Research on multidomain authentication of iot based on cross-chain technology. *Security and Communication Networks*, 2020.
22. P. Frauenthaler, M. Sigwart, C. Spanring, M. Sober, and S. Schulte. Eth relay: A cost-efficient relay for ethereum-based blockchains. In *2020 IEEE International Conference on Blockchain*, pages 204–213, Rhodes, Greece, 2020. IEEE.
23. F. Armknecht, G. Karame, A. Mandal, F. Youssef, and E. Zenner. Ripple: Overview and outlook. In *International Conference on Trust and Trustworthy Computing*, pages 163–180, Crete, Greece, 2015. Springer.
24. J.B.Zhang, Y.H.Liu, and Z.Q.Zhang. Research on cross-chain technology architecture system based on blockchain. In *International Conference in Communications, Signal Processing, and Systems*, pages 2609–2617, Urumqi, China, 2019. Springer.
25. S.A. Thyagarajan, G. Malavolta, and P. Moreno-Sanchez. Universal atomic swaps: Secure exchange of coins across all blockchains. In *2022 IEEE Symposium on SP*, pages 1299–1316. IEEE, 2022.

Block-gram: Mining Knowledgeable Features for Smart Contract Vulnerability Detection

Tao Li[1,2,3,4], Haolong Wang[2], Yaozheng Fang[2], Zhaolong Jian[2], Zichun Wang[2], and Xueshuo Xie[⋆2,3,4]

[1] Tianjin Key Laboratory of Network and Data Security Technology, Tianjin, China
[2] College of Computer Science, Nankai University, Tianjin, China
{whlong,fyz,jianzhaolong}@mail.nankai.edu.cn
{litao,wzc,xueshuoxie}@nankai.edu.cn
[3] Key Laboratory of Blockchain and Cyberspace Governance of Zhejiang Province
[4] State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences

Abstract. Effective vulnerability detection of large-scale smart contracts is critical because smart contract attacks frequently bring about tremendous economic loss. However, code analysis requiring traversal paths and learning methods requiring many features training is too time-consuming to detect large-scale on-chain contracts. This paper focuses on improving detection efficiency by reducing the dimension of the features, combined with expert knowledge. We propose a feature extraction method *Block-gram* to form low-dimensional knowledgeable features from the bytecode. We first separate the metadata and convert the runtime code to opcode sequence, dividing the opcode sequence into segments according to some instructions (*jump*, etc.). Then, we mine extensible *Block-gram* features for learning-based model training, consisting of 4-dimensional block features and 8-dimensional attribute features. We evaluate these knowledge-based features using seven state-of-the-art learning algorithms to show that the average detection latency speeds up 25 to 650 times, compared with the features extracted by *N-gram*.

Keywords: Smart Contract, Bytecode, Opcode, Knowledgeable Features, Vulnerability Detection

1 Introduction

Smart contracts are widely deployed on blockchain to implement complex transactions, such as decentralized applications on Ethereum [1]. As of August 9, 2022, the number of smart contracts exceeded 51.1 million on Ethereum[1]. These smart contracts are written in a domain specific language (e.g., *Solidity*), compiled into bytecodes, executed as opcodes in EVM after being deployed on-chain by the

⋆ Corresponding author: xueshuoxie@nankai.edu.cn
[1] https://explore.duneanalytics.com/

consensus mechanism [2]. Due to running on distributed nodes of blockchain, once vulnerabilities are found, they are difficult to upgrade and repair [3,4]. For example, hackers exploited a re-entrancy vulnerability in the DAO contract to steal 3.6 million ETH[2]. According to SlowMist Hacked, a huge economic loss of more than $10 billion has been caused due to the security issues of smart contracts[3]. With the rapid increase in smart contracts, an efficient smart contract vulnerability detection method is particularly important for the blockchain [5].

Nowadays, the mainstream smart contract vulnerability detection methods include code analysis and machine learning. For code analysis, we can analyze the types or causes of vulnerabilities with expert knowledge through formal verification, symbolic execution, fuzz testing, etc. [6–10]. But these methods need to traverse more paths of code or complexity mathematical proofs, the detection is time-consuming and labor-intensive. For machine learning, they primarily capture code features by training machine learning models to infer whether it is vulnerable. Some smart contract vulnerability detection algorithms are based on the combination of text information and neural network [5, 11, 12] or based on the combination of smart contract graph information and neural network [13,14]. But all of them need a large feature space for training, and the dimensional of features will influence the model performance and detection latency.

In this paper, we focus on detection efficiency of large-scale smart contracts and face the following challenges: (1) how to improve detection performance combined with vulnerability features through expert knowledge; (2) how to reduce the dimension of feature space for optimizing the detection latency without influencing the model performance. To tackle the first challenge, we preprocess the bytecode to opcode sequences according to the disassembling rules of Ethereum and divide the opcode sequences into flow graphs through some instructions (*jump*, etc.) for extracting 4-dimensional block features. For the second challenge, we mine other 8-dimensional attribute features of vulnerabilities through expert knowledge, to construct extensible 12-dimensional *Block-gram* features. The *Block-gram* features have lower dimensional than thousand dimensions features by *N-gram*, and will significantly reduce the detection latency without influencing the model performance. In summary, the contributions:

- We mine the extensible low-dimensional *Block-gram* features from bytecode, including 4-dimensional block features, and 8-dimensional attribute features by smart contract vulnerabilities analysis.
- We introduce expert knowledge when constructing opcode sequence flow graphs and extracting attribute features, and combine vulnerability analysis to improve the performance of the above low-dimensional features.
- We validate the efficiency of *Block-gram* features on seven state-of-the-art machine learning algorithms. The evaluation shows that the above low-dimensional features can flexibly support multiple detection algorithms and significantly reduce detection latency.

[2] http://www.coindesk.com/daoattacked-code-issue-leads-60-million-ether-theft, 2016.

[3] https://hacked.slowmist.io/en/

2 Preliminaries

2.1 Ethereum Smart Contract

Ethereum Virtual Machine. EVM provides 142 opcodes or bytecodes with 10 functions, such as *STOP* and *PUSH* operations, etc. There are only 256 opcodes at maximum, but some instructions are not defined now, only for future expansion. EVM has a simple stack structure with a maximum stack size of 1024. Each opcode is allocated one byte (for example, *STOP* is *0x00*), pushes or pops a certain number of elements from the stack, and can obtain information about the execution environment, or interact with other blockchains smart contracts [15]. During execution, the bytecode is split into bytes (1 byte equals 2 hex characters). Bytes in region *0x60-0x7f* (*PUSH1-PUSH32*) are treated differently because they contain data that needs to be pushed into the stack. If the call count is over 1024, the call-stack attack may take place. EVM explains how to change the system state given a series of the above instructions and a small part of environmental data.

Smart Contract Compilation. As shown in Fig. 1, the developers write the source code in a high-level language (e.g., *Solidity*). The source codes are compiled into byte arrays encoded by hexadecimal digits with a compiler as bytecodes [16]. Then, the bytecodes are uploaded to EVM with an Ethereum client and can be translated into EVM instructions or opcodes [17]. After the source code of the contract is compiled into EVM bytecode and ABI, it can be deployed using the *Web3.js* interface. For contract deployment, it is essential to execute a transaction, which has no destination address but the data field is EVM bytecode [18]. When processing this transaction, the EVM executes the input data as code. The bytecode is divided into deployment code, runtime code, and metadata. After the contract deployment, EVM will store the runtime code and metadata on the blockchain, and then match their storage addresses to the contract account to complete the deployment of the contract. It would be easier to analyze smart contracts with bytecodes or opcodes, because: (1) bytecodes or opcodes are not had man-made variables that are defined in source codes; (2) bytecodes or opcodes are easy to collect from the public blockchain.

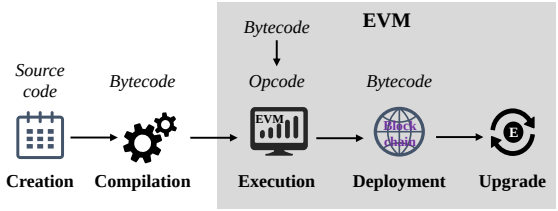


Fig. 1: Smart Contract Compilation Process.

Table 1: Comparisons among smart contract vulnerability detection methods.

Type	Name	Base Model	Detection Source	Platform
Code analysis	KEVM	Formal Verification	bytecode	EVM
	Oyente	Symbolic Execution	bytecode & ETH Condition	EVM
	Contractfuzzer	Fuzz Testing	EVM ABI & EVM Log	EVM
	Ethir	Intermediate Representation	bytecode	EVM
Learning methods	DR-GCN	GCN	source code	ETH & VNT
	TMP	TGNN	source code	ETH & VNT
	Deescvhunter	DNN	source code	ETH
	Eth2vec	DNN	bytecode	ETH
	Escort	DNN	bytecode	ETH
	Rechecker	BLSTM	sourcecode	ETH
	ContractWard	XGBoost	bytecode	ETH
	Vscl	DNN	bytecode	ETH
	EtherGIS	GNN	bytecode	ETH
	SafeSC	LSTM	opcode	ETH

2.2 Smart Contract Security Analysis

Code Analysis. As shown in Table 1, most of the traditional smart contract vulnerability detection methods are based on program code analysis and program path analysis. Hildenbrandt et al. [6] present KEVM based on formal verification and provide an executable formal specification for EVM's program language using the K framework. Luu et al. [8] use symbolic execution to implement Oyente which traverses smart contract execution paths based on control flow graphs to detect vulnerability. Jiang et al. [9] propose Contractfuzzer that sets up test cases and analyzes smart contract behavior logs to detect vulnerabilities based on fuzz testing. Albert et al. [10] implement Ethir based on intermediate representation and analyze the security properties of bytecode by converting Oyente's control flow graph into a rule-based representation. Due to the need to traverse most paths of the code, the detection is time-consuming and labor-intensive, and difficult to use for large-scale contract detection.

Learning method. As shown in Table 1, Wang et al. [5] propose Contract-Ward that can extract bigram features from smart contract opcodes and use multiple machine learning algorithms for vulnerability detection. Mi et al. [12] apply novel feature vector generation techniques from bytecode and metric learning-based deep neural network to detect vulnerability. Zhuang et al. [13] use DR-GCN to convert source code into contract graph and use graph convolutional neural network to build a vulnerability detection model. Based on DR-GCN, TMP considered the time sequence information in the contract graph and used the time sequence graph neural network to build a vulnerability detection model. Zeng et al. [14] use graph neural network and expert knowledge to build the control flow graph with attribute and input graph attribute features into graph neural networks to detect vulnerabilities. However, due to the lack of expert knowledge, most learning methods use high-dimensional feature spaces for training, such as *N-gram* extracting thousands of dimensional features. The detection is still time-consuming and may not be suitable for batch vulnerability detection. Therefore, combining the expert knowledge used in code analysis with the learning methods is precisely the problem that our work mainly solves.

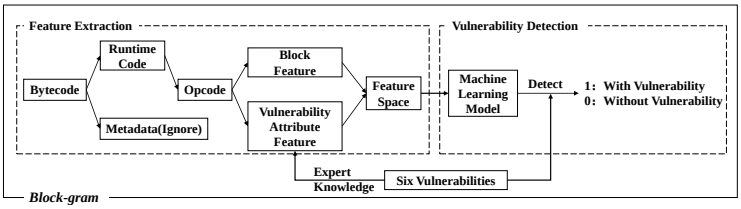


Fig. 2: The *Block-gram* feature extraction method overview.

3 Detailed Design

3.1 Overview

As shown in Fig.2, we present a detailed description of the features extraction method *Block-gram* using smart contract bytecode. We first extract the 4-dimensional bytecode block features from the opcode sequence flow graph according to EVM disassembling rules. Then, we divide the opcodes into eight categories and count their ratios as 8-dimensional attribute features through six vulnerabilities analysis by expert knowledge. Finally, we use these 12-dimensional *Block-gram* features by seven state-of-the-art machine learning algorithms to detect six vulnerabilities. The *Block-gram* features include in:

- **Rule-based bytecode block features.** We first collect the bytecode of the metadata header of various solidity versions. The metadata in the bytecode is separated from the runtime code by string matching, and convert the runtime code into opcode according to Ethereum disassembling rules. Then we divide the opcode sequence into blocks through `JUMP` opcode, and generate the opcode sequence flow graph, denoted by the adjacency matrix. We extract 4-dimensional bytecode block features from the adjacency matrix. These features can preserve the relationship between different bytecode blocks.
- **Attribute Features with Expert Knowledge.** We first divide the opcodes of Ethereum into eight categories, such as block opcodes, system opcodes, stack opcodes, etc. Each type of opcode corresponds to several smart contract vulnerabilities according to the vulnerability analysis with expert knowledge. We count their ratios as 8-dimensional extensional attribute features. These features can preserve the expert knowledge on vulnerability, and extend with the new vulnerabilities.

3.2 Rule-based Bytecode Block Features

The key to bytecode preprocessing are metadata separation and bytecode conversion. In the metadata separation module, we separate metadata and runtimecode according to metadata header(*e.g.,0x65 'bzzr0' 0x58 0x20 <32 bytes swarm hash>0x00 0x29*). Then we convert the runtimecode to the opcode according to Ethereum conversion rules. The bytecode of each non-PUSH class opcodes is

converted into the corresponding opcode(for example, *STOP* is *0x00*, *ADD* is *0x01*). But for the PUSH class opcodes, we get the length of their parameters according to the type of the PUSH opcode, so as to convert the bytecode.

The existing processing methods of opcode sequence are divided into block-sequence method and natural language processing method. Block-sequence method divides the opcode sequence into blocks and edges. Natural language processing method uses *N-gram* method to process the opcode sequence. We choose block-sequence method to process opcode sequence and build adjacency matrix. First, we divide the opcode into blocks according to the jump class instruction, and then determine the edges between blocks according to the jump type at the end of each block. Each blocks has the following boundary.

- **Starting point:** `JUMPDEST`...
- **End point:** `JUMP, JUMP I, STOP, REVERT, RETURN, SELFDESTRUCT, INVALID`

In order to construct the edges between blocks, we divide edges into three categories according to the end point of blocks:

- `JUMP`**:** If there is `PUSHn` before `JUMP`, the parameter of `PUSHn` is the destination address of `JUMP`. If there is not `PUSHn` before `JUMP`, we using stack execution algorithm provided by EtherSlove [19] to calculate the destination address.
- `JUMP I`**:** `JUMP I` is a conditional jump. True edge's target is the parameter of the PUSH opcode; false edge's target is the offset of the following block. This means that if a basic block ends with JUMP I, then there will be two edges starting from this block.
- `REVERT, SELFDESTRUCT, RETURN, INVALID, STOP`**:** These opcodes mean the interruption of control flow, so they have no subsequent basic blocks.

When we get the blocks and edges, we can build an opcode sequence flow graph. Because the opcode sequence flow graph is a directed graph, we use the adjacency matrix to represent the opcode sequence flow graph and extract sequence features from the adjacency matrix. Then, we extract four-dimensional smart contract block features according to the adjacency matrix. These features represent the sequence attribute of the opcode.

- **Number of nodes.** The number of rows or columns of the adjacency matrix is the number of nodes. The number of nodes represents how many basic blocks are in the opcode sequence flow graph.
- **Number of edges.** The sum of the elements in the adjacency matrix is the number of edges. The number of edges represent how many basic edges are in the opcode sequence flow graph.
- **Maximum out-degree.** The maximum value of the sum of the elements in row i is the maximum out-degree. Out-degree is the sum of the times when a block of the opcode sequence flow graph is used as the starting point.
- **Maximum in-degree.** The maximum value of the sum of the elements in row j is the maximum in-degree. In-degree is the sum of the times when a block of the opcode sequence flow graph is used as the end point.

Table 2: *Block-gram* Knowledgeable Features.

Feature		Opcode	Value	Knowledge
Block Feature	node	None	>0	feature of block
	edge		>0	
	maxout		>0	
	maxin		>0	
Attribute Feature	unary-arithmetic ratio	`ISZERO, NOT...`	(0,1)	feature of integer overflow
	binary-arithmetic ratio	`ADD, AND, SHA3...`	(0,1)	
	block ratio	`NUMBER, BLOCKHASH, COINBASE...`	(0,1)	feature of timestamp dependency
	control-flow ratio	`JUMP, JUMP I, JUMPDEST...`	(0,1)	feature of re-entrancy
	environment ratio	`CALLER, CALLDATASIZE...`	(0,1)	feature of TOD
	system ratio	`CALL, RETUREN, REVERT...`	(0,1)	feature of callstack depth attack
	stack ratio	`POP, PUSH, SWAP...`	(0,1)	
	invalid ratio	Others	(0,1)	feature of invalid opcodes

3.3 Attribute Features with Expert Knowledge

There are some common vulnerabilities in Ethereum, such as integer overflow vulnerabilities, integer underflow vulnerabilities, callstack depth attack vulnerability, transaction-ordering dependence vulnerability, timestamp dependency vulnerability, and re-entrancy vulnerability. We analyze above vulnerabilities and opcodes related to them and extract attribute features. We divide the opcodes into eight categories, such as unary arithmetic opcodes, binary arithmetic opcodes, block opcodes, control-flow opcodes, environmental opcodes, system opcodes, stack opcodes, and invalid opcodes. As shown in Table 2, each type of opcode corresponds to several smart contract vulnerabilities according to the vulnerability analysis with expert knowledge. These features can preserve the expert knowledge on vulnerability, and extend with the new vulnerabilities. We define *count (opcode, i)* as the number of all opcodes of the smart contract i and *count (j, i)* as the number of opcodes corresponding to feature j in the smart contract i. For example, *count (1, 1)* is the number of unary arithmetic opcodes in smart contract 1. $count(1, i)/count(opcode, i)$ is ratio of feature 1.

- **Unary and Binary arithmetic opcodes ratio:** In Ethereum, some opcodes are responsible for arithmetic operations, including unary, binary, and ternary arithmetic opcodes. Only these arithmetic opcodes cause integer overflow. In our research on smart contract, we find that almost no contract contain ternary arithmetic codes, so we only consider unary(*e.g.,* `ISZERO, NOT`) and binary(*e.g.,* `ADD, AND, SHA3`) arithmetic opcodes.
- **Block opcodes ratio:** In Ethereum, some opcodes are related to the block information. `BLOCKHASH` shows the hash value of the block, `COINBASE` is the address of the miner. At Ethereum system layer, block information is often used as a seed for generating random numbers. However, the block timestamp, number, and other information are often used by attackers. Block information opcodes are related to timestamp dependency vulnerabilities.
- **Control-flow opcodes ratio:** Some opcodes are used to change the control flow. `JUMP` is an unconditional jump that takes the top element of the stack as the destination address. `JUMP I` is a conditional jump that has the same

destination address with `JUMP` if the top element of the stack is not zero; otherwise, EVM will execute the opcodes following `JUMP I`. The re-entrancy vulnerability stems from the attacker's cyclic call changing the control flow. Control flow opcodes are related to this vulnerability.

- **Environmental opcodes ratio:** Some opcodes are responsible for interacting with contracts and message calls and transactions. `ADDRESS` is the address of the currently executed contract. `ORIGIN` is the address of the initiator of the transaction. `CALLER` is the address of the caller of the message. Since the environmental opcodes can obtain transaction information, they are related to Transaction-Ordering Dependence vulnerabilities that also rely on transaction information.
- **System opcodes ratio:** The system opcodes are responsible for calling between smart contracts. `CALL` calls the function in other contracts. `RETURN` returns from the contract calls. `REVERT` reverts transaction and return data.
- **Stack opcodes ratio:** EVM is a stack machine, and all calculations are performed on a data area called the stack. Some opcodes deal with the elements of the stack. `POP` pops the top element of the stack and discards it. `SWAP1` exchanges the top two members of the stack. They are stack opcodes.System opcodes and stack opcodes act on calls and stack operations, and these opcodes may cause callstack depth attack vulnerabilities.
- **Invalid opcodes ratio.** Invalid opcodes refer to the opcodes irrelevant to the six vulnerabilities detected.

3.4 Low-dimensional Knowledgeable Features

During feature extraction, to make the trained model suitable for all smart contracts, we first select 4-dimensional block features to highlight the relationship between different opcode blocks. The 4-dimensional block features are the number of nodes, the number of edges, the maximum out-degree, and the maximum in-degree of the control flow graph. These 4-dimensional features represent the complexity of the smart contract, emphasizing the role of the features of the smart contract itself in vulnerability detection. Then, we investigate the causes of vulnerabilities in smart contracts and mine 8-dimensional attribute features for opcodes associated with them. For example, unary-arithmetic and binary-arithmetic opcodes modify integers in Ethereum, and improper arithmetic operations can lead to integer overflow vulnerabilities; block information opcodes are closely related to timestamp vulnerabilities and block parameter dependency vulnerabilities; control flow opcodes are related to the re-entrancy vulnerability stems from the attacker's cyclic call changing the control flow. As shown in Table 2, we combined the 4-dimensional block features and 8-dimensional attribute features together for efficient vulnerability detection. When constructing the opcode sequence flow graph, we used depth-first-search algorithm, and the time complexity is $O(N)$. When constructing the adjacency matrix, the time complexity is $O(N^2)$. Therefore, the time complexity of feature extraction is $O(N + N^2)$, where N represents the nodes amount of the above graph.

Features normalization. Due to the difference in feature extraction, the first four-dimensional features are large integers, and the last eight-dimensional features are decimals between 0 and 1. Therefore, if these 12-dimensional features are directly used as the input of machine learning models, some machine learning models (such as K-Nearest Neighbors) will only focus on the first four-dimensional features while ignoring the last eight-dimensional features during training. In addition, some machine learning models also have requirements for the format of input data. To make *Block-gram* features suitable for most mainstream machine learning models, we use linear normalization to process these features. The normalization method is defined in Equation (1).

$$x^*_{(n,f)} = \frac{x_{(n,f)} - \min\limits_{0<i<r} x_{(i,f)}}{\max\limits_{0<i<r} x_{(i,f)} - \min\limits_{0<i<r} x_{(i,f)}} \tag{1}$$

where $x_{(n,f)}$ represents the value of the feature f in the n row.

4 Evaluation

4.1 Experimental Setup

Configuration. We perform experiments on a Windows 10 machine with 12th Gen Intel Core 2.10 GHz CPUs and 32GB RAM and use the GPU of a 1060ti graphics card to train the model and predict the results. To verify the efficiency and the validity of the above 12-dimensional feature, we choose seven state-of-the-art machine learning algorithms, such as eXtreme Gradient Boosting (XGBoost), Random Forest(RF), K-Nearest Neighbors(KNN), Logistic Regression(LR), Decision Tree(DT), Naive Bayes, and Long short-term memory(LSTM), as the training and detection model. We use the sklearn library in python3.6.8 to build the machine learning algorithms. We also select accuracy, recall, F1-score, and latency as the measured metrics of the model.

Datasets. We select 3000 smart contracts as the dataset for performance analysis from Contractward [5], 70% for training, and 30% for testing. The size of the dataset is 62.7MB. There are 871 contracts with vulnerabilities and 2179 contracts without vulnerabilities. The vulnerabilities of the dataset include integer overflow and integer underflow vulnerabilities, callstack depth attack vulnerability, transaction-order dependence vulnerability, timestamp dependency vulnerability, and re-entrancy vulnerability.

4.2 Performance Analysis

Detection performance. As shown in Table 3, in terms of accuracy, the maximum value of the seven models is 82.22% and the minimum value of the seven models is 73.88% when they use *Block-gram* features. When they use features extracted from opcodes by *N-gram*, almost all models' accuracy drops. *Block-gram* features perform better than features extracted by *N-gram* in terms of

Table 3: Model Performance by *Block-gram* Features.

Model	Feature	Accuracy(%)	Recall(%)	F1-score(%)	Latency(ms)
XGBoost	*Block-gram*	**82.22**	93.27	88.16	**0.2**
	N-gram	80.4	95.3	87.33	15
Random Forest	*Block-gram*	**81.77**	95.00	88.10	**0.3**
	N-gram	70.55	98.27	82.58	12
K-Nearest Neighbors	*Block-gram*	**75.44**	**84.51**	**83.01**	**0.004**
	N-gram	51.55	36.62	51.77	0.1
Logistic Regression	*Block-gram*	75.44	98.28	85.04	**0.01**
	N-gram	75.88	87.17	83.70	0.4
Decision Tree	*Block-gram*	77.44	84.66	84.20	**0.004**
	N-gram	76.11	83.72	83.27	2.6
Naive Bayes	*Block-gram*	**76.66**	**87.17**	**84.13**	**0.0003**
	N-gram	60.33	58.37	67.63	0.09
LSTM	*Block-gram*	**73.88**	66.28	59.55	**1**
	N-gram	69	75.10	58.42	90

accuracy. In terms of recall and F1-score, the performance of K-Nearest Neighbors and Naive Bayes drops significantly when they use features extracted by *N-gram*. And other measured metrics of K-Nearest Neighbors and Naive Bayes also dropped significantly. The two models poor perform when dealing with *N-gram* features. The reason is that we have considered the real jump relationship when the smart contract runs and the expert knowledge of six vulnerabilities but the *N-gram* method only extracts the combination of the opcode sequence.

Detection latency. As shown in Fig.3, the detection latency of all models is greatly improved when they use *Block-gram* features. Among the seven models, the detection latency of the decision tree model has reduced 650 times when using *Block-gram* features compared with using *N-gram*. In addition, compared with the traditional method Oyente and other machine learning methods (for example, VSCL and EtherGIS), the latency of using *Block-gram* features

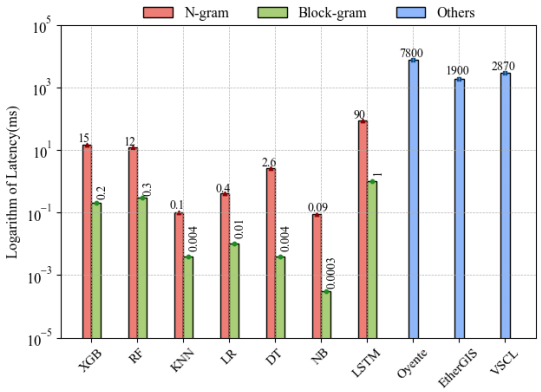


Fig. 3: Comparisons among smart contract detection latency.

is also significantly reduced. Since VSCL and EtherGIS did not publish open-source code, we directly cited the detection delay published in the paper. Their experimental environment is far better than our work. The reason is that the dimension of the feature space extracted by *N-gram* is too high, as high as tens of hundreds of dimensions during the initial extraction. In training, the features will be expanded up to tens of thousands of dimensions. When using *Block-gram* features, the initially extracted feature space is only 12-dimensional. Even if it needs to expand during the training process, it is only 15-dimensional at most. Significant differences in feature space dimensions lead to differences in latency. The experimental results demonstrate that *Block-gram* features are efficient and can be used by mainstream machine learning models.

5 Conclusion

This paper addressed improving the detection efficiency for the quick detection requirements of large-scale smart contracts. We only used extensible 12-dimensional features mining from bytecode and opcode. The low-dimensional features will speed up the detection time 25 to 650 times without influencing the model performance (*accuracy* etc.). We can also extend these features to support more vulnerability detection and security analysis in the future. Compared with the existing thousand-dimensional feature space, the features improve the detection efficiency and extend the detection range. The evaluation based on seven state-of-the-art learning-based methods has shown the effectiveness of *Block-gram* features and can significantly improve detection efficiency.

Acknowledgment

This work is partially supported by the National Natural Science Foundation (62272248), the Open Project Fund of State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences (CARCH201905, CARCHA202108), the Natural Science Foundation of Tianjin (20JCZDJC00610) and Sponsored by Zhejiang Lab (2021KF0AB04).

References

1. S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han, and F.-Y. Wang, "Blockchain-enabled smart contracts: architecture, applications, and future trends," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 11, pp. 2266–2277, 2019.
2. F. Ma, Y. Fu, M. Ren, M. Wang, Y. Jiang, K. Zhang, H. Li, and X. Shi, "Evm*: from offline detection to online reinforcement for ethereum virtual machine," in *2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 2019, pp. 554–558.
3. A. R. Sai, C. Holmes, J. Buckley, and A. L. Gear, "Inheritance software metrics on smart contracts," in *Proceedings of the 28th International Conference on Program Comprehension*, 2020, pp. 381–385.

4. T. Durieux, J. F. Ferreira, R. Abreu, and P. Cruz, "Empirical review of automated analysis tools on 47,587 ethereum smart contracts," in *Proceedings of the ACM/IEEE 42nd International conference on software engineering*, 2020, pp. 530–541.
5. W. Wang, J. Song, G. Xu, Y. Li, H. Wang, and C. Su, "Contractward: Automated vulnerability detection models for ethereum smart contracts," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1133–1144, 2020.
6. E. Hildenbrandt, M. Saxena, N. Rodrigues, X. Zhu, P. Daian, D. Guth, B. Moore, D. Park, Y. Zhang, A. Stefanescu *et al.*, "Kevm: A complete formal semantics of the ethereum virtual machine," in *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*. IEEE, 2018, pp. 204–217.
7. J. Krupp and C. Rossow, "{teEther}: Gnawing at ethereum to automatically exploit smart contracts," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 1317–1333.
8. L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 254–269.
9. B. Jiang, Y. Liu, and W. K. Chan, "Contractfuzzer: Fuzzing smart contracts for vulnerability detection," in *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2018, pp. 259–269.
10. E. Albert, P. Gordillo, B. Livshits, A. Rubio, and I. Sergey, "Ethir: A framework for high-level analysis of ethereum bytecode," in *International symposium on automated technology for verification and analysis*. Springer, 2018, pp. 513–520.
11. K. Gai and M. Qiu, "Reinforcement learning-based content-centric services in mobile sensing," *IEEE Network*, vol. 32, no. 4, pp. 34–39, 2018.
12. F. Mi, Z. Wang, C. Zhao, J. Guo, F. Ahmed, and L. Khan, "Vscl: Automating vulnerability detection in smart contracts with deep learning," in *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2021, pp. 1–9.
13. Y. Zhuang, Z. Liu, P. Qian, Q. Liu, X. Wang, and Q. He, "Smart contract vulnerability detection using graph neural network." in *IJCAI*, 2020, pp. 3283–3290.
14. Q. Zeng, J. He, G. Zhao, S. Li, J. Yang, H. Tang, and H. Luo, "Ethergis: A vulnerability detection framework for ethereum smart contracts based on graph learning features," in *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 2022, pp. 1742–1749.
15. T. Li, Y. Fang, Y. Lu, J. Yang, Z. Jian, Z. Wan, and Y. Li, "Smartvm: A smart contract virtual machine for fast on-chain dnn computations," *IEEE Transactions on Parallel and Distributed Systems*, 2022.
16. H. Qiu, M. Qiu, G. Memmi, Z. Ming, and M. Liu, "A dynamic scalable blockchain based communication architecture for iot," in *International Conference on Smart Blockchain*. Springer, 2018, pp. 159–166.
17. K. Gai, Y. Wu, L. Zhu, Z. Zhang, and M. Qiu, "Differential privacy-based blockchain for industrial internet-of-things," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4156–4165, 2019.
18. Z. Tian, M. Li, M. Qiu, Y. Sun, and S. Su, "Block-def: A secure digital evidence framework using blockchain," *Information Sciences*, vol. 491, pp. 151–165, 2019.
19. F. Contro, M. Crosara, M. Ceccato, and M. Dalla Preda, "Ethersolve: Computing an accurate control-flow graph from ethereum bytecode," in *2021 IEEE/ACM 29th International Conference on Program Comprehension (ICPC)*. IEEE, 2021, pp. 127–137.

Enhanced 4A Identity Authentication Center Based on Super SIM Technology

Renjie Niu[1], Zixiao Jia[2], Yizhong Liu[2⋆], Jianhong Lin[1], Xiaoli Huang[1], and Min Sun[1]

[1] China Mobile Information Technology Co., Ltd, Shenzhen, China
[2] School of Cyber Science and Technology, Beihang University, Beijing, China
niurenjie@chinamobile.com, jiazixiao@buaa.edu.cn, liuyizhong@buaa.edu.cn, linjianhong@chinamobile.com, huangxiaoliit@chinamobile.com, sunmin@chinamobile.com

Abstract. This paper explains how large-scale IT systems balance security and enabling under the pressure of supporting complex business models, constantly introducing new technologies, maintaining unified identity management, and meeting the business diversification of digital transformation. The paper introduces the necessity and basic idea of using China Mobile Super SIM technology to realize the enhanced Authentication, Authorization, Accounting, and Audit (4A) identity authentication center. We then describe the system's design principles, references, and architecture and detail several key processes and capabilities. Finally, the implementation and evolution of the enhanced identity authentication system are described.

Keywords: Super SIM card, Identity authentication, 4A, Treasury mode, Operation review

1 Introduction

With the continuous deepening of the development of the digital economy around the world, thanks to the rapid progress in cloud system [1, 2] and computer capability [3, 4], various companies have accelerated the pace of digital transformation [5, 6]. In this context, it is a considerable challenge for all communication industry businesses and security personnel to manage the identity, authentication, authorization, and access control of various operation and maintenance activities so as to ensure the security of crucial systems and sensitive information [7]. On November 19, 2020, China Mobile released the Super SIM card at the Global Partner Conference. It is an integrated security product based on the SIM card security chip and uses different functions to achieve different levels of authentication [8, 9]. It contains an encryption chip and NFC function, which can be used as a meal card, access control card, transportation card, and key to a car offline, or can perform financial security authentication, 5G electronic signature [10], and large-value transfers online.

⋆ Corresponding author.

This paper combines the super SIM authentication technology with the 4A system, adopts the concept of zero trust and multi-factor authentication, takes the 4A system as the core, and uses the super SIM to achieve the security capability of EAL5+, so as to build a convenient authentication process that supports hundreds of millions of times a day. If an identity authentication center is to be implemented based on super SIM technology, the following basic requirements must be met:

Independent computing requirements. Use the independent and removable key computing entity to achieve chip-level security assurance and achieve the self-security capability of EAL5+; Implement centralized remote management of the chip to avoid the risk of man-in-the-middle attacks and data replay attacks, and increase the complexity and time of cracking.

No perception/less perception requirement. Perform multiple automatic authentications in scenarios such as user login, resource login, multi-factor authentication, secondary authentication, and vault mode, and combine zero-trust technology to achieve a non-perceptual, non-disturbing automated authentication process [11], which improves security and reduces users' number of passive authentications;

Centralized management requirements. Realize the management of subject, object, authorization relationship, trust degree, and operation review; Allow different business operation and maintenance modes to adopt different authentication strengths and authorization relationships and simultaneously perform remote manual intervention control on abnormal subjects, access relationships, devices, and SIM cards.

In summary, the contributions of this paper include the following points:

- We put forward some design principles based on the super SIM technology, thus completing the design of the enhanced 4A identity authentication center so that it can well meet the three basic requirements mentioned above.
- Based on the design architecture of the enhanced 4A identity authentication center, we have realized the four capabilities of the identity authentication center, namely, no perception/low perception authentication capability, multi-factor authentication capability, secondary authentication capability, and operation review (vault mode) capability.
- Considered and designed the security of the enhanced 4A certification center, improved its security performance in the process of authentication and transmission, and enhanced the robustness of the certification center so that it can automatically detect and process abnormal behaviors in operation.

The remainder of the work is organized as follows. Section 2 introduces the design principles of the enhanced 4A identity authentication center and its overall process. Section 3 analyzes the unique capabilities of the enhanced 4A identity authentication center. The security of the system and its ability to handle abnormal behavior are discussed in Section 4. Finally, concluded remarks of the conducted work are discussed in Section 5.

2 Design of Enhanced 4A Identity Authentication Center

By designing an identity authentication center that combines super SIM and 4A, system security can be further improved, user perception can be optimized, information systems and physical systems can be connected, and refined management can be achieved. The core design principles are as follows:

The principle of dual-core certification. Two cores are formed by super SIM and 4A, combined through a secure and open channel. The core of the super SIM is the authentication capability of the EAL5+ level to realize the security root on the user side. The mobile phone and the super SIM integrate biometrics, geographical location, behavioral characteristics, and zero-trust capabilities of the mobile terminal to achieve financial-level authentication. It has become a universal ID card and pass card for users in the physical and information worlds. As the core capability of identity management and control of various business systems, the 4A system has gradually evolved into an identity middle platform to integrate current new, and future advanced identity-related capabilities.

User-perceived optimization principle. Achieving "safety" and "convenience" has always been challenging. It is necessary to use intelligent analysis, environmental perception, and chip-level security technologies to judge the security certification status through technical means. Only when certification is required are users required to authenticate, so that to improve security capabilities. At the same time, it optimizes user perception and improves refined identity management and control, reducing the "repetitive" and "tedious" experience of users in the authentication process.

The principle of advanced security management and control. In the face of critical systems and sensitive data operations, security controls must be improved to achieve real-time authentication and real-time audit of people, data, and behavior.

The principle of minimum authorization and convenient permission application. Due to the complex business scale of operators and the complex roles and permissions of operation and maintenance and operation personnel, in order to ensure access to the gateway system and sensitive data, it is necessary to adopt an on-demand, and per-event authorization mode, which is intuitively reflected in that users can see the function entry. However, it is necessary to apply for permission before use. The approval action is automatically performed by the work order or authorized by the superior. The user obtains the temporary permission and withdraws it immediately after the work is completed or after the timeout. In order to simplify the authorization work of the superior supervisor, it is necessary to combine the super SIM computing power of the supervisor to complete the authorization through a simple click operation on the mobile phone.

2.1 Identity Authentication Logic Combining Super SIM and 4A

In order to realize the integration of super SIM and 4A, it is necessary to incorporate the "5G super SIM card authentication open platform" into the identity security infrastructure connected with 4A to implement the overall integrated

identity authentication logic. The enhanced 4A identity authentication center logic based on super SIM technology is shown in Figure 1.

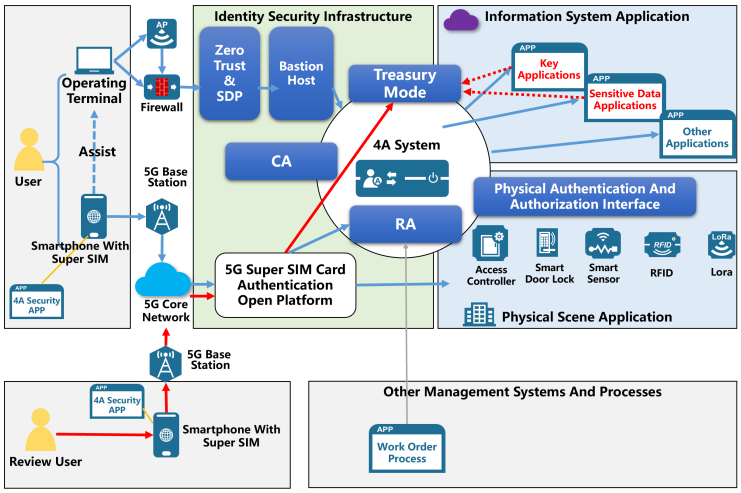


Fig. 1. Logic diagram of enhanced 4A identity authentication center based on super SIM technology

2.2 The Main Functions of 4A Security APP

Due to super SIM technology's emergence, smartphones' security has been dramatically improved, and mobile phone authentication, operation, and approval have been greatly improved. It is necessary to redesign a 4A security APP based on super SIM cards and smartphones. Its main functional structure is shown in Figure 2.

4A security APP is to transform the 4A capabilities from the traditional PC field to the mobile Internet field. It utilizes the security features of current smartphones and super SIMs. Based on the characteristics of mobile phones being carried around and closely related to natural persons, the APP program is used to convert the identity of natural persons into the information world. Its main features are divided into two fields, one is the functional field, including security authentication function, operation review function, and zero trust function; the other is its own security field, including software-based security capabilities, security capabilities based on mobile phone chips, and security capabilities based on super SIM card.

2.3 Certification Process of Enhanced 4A Authentication Center

Authentication Process of Enterprise-level Application System The main principle of enterprise-level application system authentication is based on the 4A system to achieve "one point authentication, everywhere authentication;

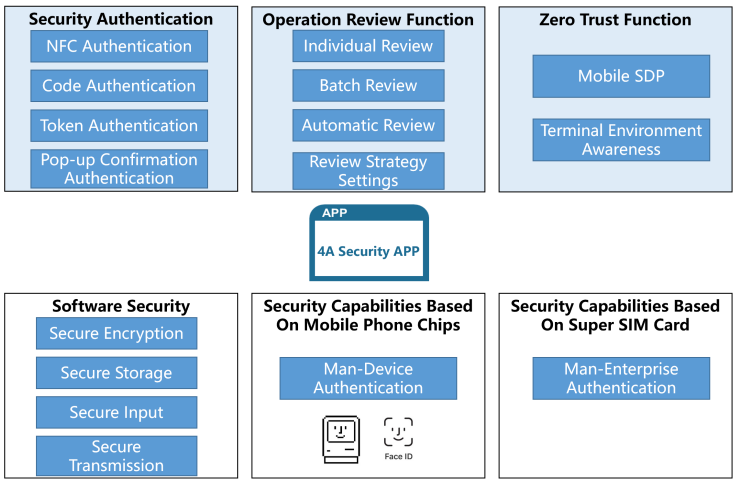


Fig. 2. The main function of the 4A security APP

one point exit, everywhere exit", that is, after the user completes the "man-device" verification and "man-identity authentication" in combination with the Super SIM and the 4A system, the authentication of the system, operation and data within its authority is also completed at the same time. It is only necessary to realize the simplified single sign-on process from the 4A system to other application systems. Once a login terminal is withdrawn from the 4A system, other login rights on the login terminal will also be withdrawn.

As shown in Figure 3, the user authentication process in the information system is as follows:

1. The user first logs in to the 4A portal and sends an authentication request to the 4A system;
2. 4A system sends an authentication challenge to the login portal, such as a username and password or a QR code that needs to be scanned with a mobile phone;
3. In the process of combining the super SIM, the user needs to use the 4A login APP on the mobile phone to scan the code;
4. While using the mobile phone, user must use fingerprint, face, encryption, and other technologies to pass the "man-device" authentication. Most of these authentication processes are completed by mobile phone software or exclusive security chips (such as Apple's T2 chip);
5. After the "man-device" authentication is completed, the "man-enterprise identity" authentication needs to be completed again. At this time, it needs to be completed by the authentication computing power of the super SIM;
6. The 5G-based super SIM security application will be connected to 4A through the base station, core network, and super SIM card authentication open platform to complete the authentication of "man-enterprise identity";

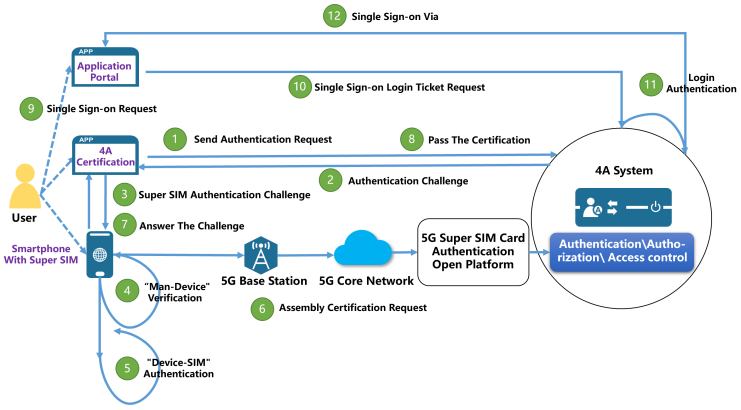


Fig. 3. Authentication application of super SIM and 4A technology in information system

7. After the certification is completed, the mobile phone super SIM application will complete the challenge response required for 4A certification, manifesting as passing the code on the 4A certification portal.
8. The 4A system judges the authentication process to be passed, and a complete set of enterprise identity authentication from natural persons, mobile devices, and PC terminals has been completed at this time;
9. In the second stage, the user requests to log in to a specific application at this time, which can be jumped through 4A or directly logged in to the application;
10. The modified applications will use the 4A single sign-on function to initiate a ticket login verification request to 4A;
11. 4 System A will re-verify factors such as people, PC terminals, mobile devices, enterprise applications, and related policies to determine whether the single sign-on process is released;
12. After the application login determination is completed, 4A will notify the application to release the user, thus completing the authentication login of the user to the application system.

In the above process, the user's recognition perception is the process of logging in to the 4A portal. In the subsequent use process, unless the security policy, timeout, user offline, secondary authentication, vault mode, and other processes are triggered, the user can use it within the adequate time. Log in to use authorized applications.

3 Enhanced Capability of 4A Certification Center

3.1 No perception/low perception Authentication Capability

In the identity authentication logic, a smartphone with a super SIM card acts as a convenient and credible man-machine trust consistency system, in which the

mobile phone itself and its security chip (such as Apple's T2 security chip) can realize "man-device." The super SIM card realizes the unique authentication of "device-enterprise identity." After combining the two, a smartphone with a super SIM card can realize the authentication of "natural man-enterprise identity." This authentication process is obtained by convenient means such as the face, fingerprint, forged voiceprint. It is enhanced by enterprise-level security technologies such as zero trust, realizing the coexistence of security and convenience.

Taking the steps of logging in to the application system through the 4A system as an example, only 3 of the 11 steps in the entire login process require user responses, and after successfully logging in to 4A, logging in to each application again only requires clicking on the relevant entry.

3.2 Multi-factor Authentication Capability

Multi-factor authentication is a method of computer access control. Users must pass two or more authentication mechanisms before being authorized to use computer resources. For example, the user wants to enter a PIN, insert a bank card, and finally, through fingerprint comparison, these three authentication methods can obtain authorization. This authentication method can improve security.

Smartphones with a Super SIM card are equipped with a security token function, which can form two-factor authentication with a username and password in certain situations. This specific situation is based on the evaluation of the zero-trust mechanism. When the security score of the user's device, network, behavior, and other factors is too low, the two-factor authentication will be turned on. The security score will be restored and returned under single-factor authentication if the two-factor authentication is passed.

3.3 Secondary Authentication Capability

The secondary authentication capability will be activated when the user's operation behavior triggers the policy. These behaviors include dangerous operations, access to sensitive data, behavior deviations from the baseline, and changes in the security status of the registrant's terminal. The action of the secondary authentication is to change the implicit authentication judgment to the explicit one and require the user to re-enter the user name, password, or super SIM card token code.

After the secondary authentication is passed, there will be no secondary authentication until the following secondary authentication action is triggered.

3.4 Operation Review (Vault Mode) Capability

Vault mode will be activated when user actions trigger policies. These actions include pre-determined actions involving dangerous operations and accessing sensitive data. The action of operation review is to suspend the operation process first and introduce a reviewer into the review process. The reviewer is usually the superior leader of the user or a post with management responsibility in the relevant operation field.

The operation reviewer will use a smartphone with a super SIM card to complete most of the operation review actions. The reviewer first needs to log in

to the 4A security APP on the mobile phone and complete the "man-device" authentication and "man-enterprise identity." Due to the high frequency of participation in operation review, the 4A security APP will support interface click and batch confirmation methods, allowing reviewers to submit review results to ensure safety quickly.

4 The Security of the Enhanced 4A Certification Center

A smartphone with a super SIM card is a security-increasing computer with independent computing and storage capabilities. Combined with some smartphones' existing independent security chips, the architecture of dual independent security computers is formed. The independent secure computer architecture formed by the super SIM card is shown in Figure 5

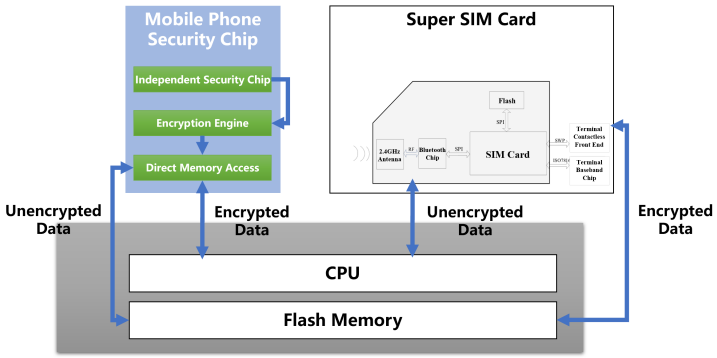


Fig. 4. Independent secure computer architecture formed by super SIM cards

4.1 Security Enhancement During Authentication and Transmission

Because the super SIM card has storage and computing capabilities, it supports domestic SM2, SM3, SM4, and SM9 security algorithms and supports RSA 2048, SHA 256, TDES, AES, ECC (Secp256r1), X 25519, ED 25519 algorithms. It can realize the encryption ability of the root key without the SIM so that the mobile phone carrying the SIM card, related applications, and network links cannot unilaterally steal or tamper with the transmission and storage containers. The transmitted data can be decrypted only through the internal program of the super SIM card and the super SIM authentication center. Code protection is enabled on the super SIM card, the long-term key K stored in the SIM card can only be accessed by the SIM card itself and the card-issuing operator, while most smartphones are used in network access and data transmission. The keys are all derived based on the key K, which shows that the communication process of the smartphone has exceptionally high security.

Authenticity of Data Transmission: In the communication of the authentication request, it is indispensable to ensure the authenticity of the identity of

the person transmitting the data to prevent the user authentication data from being counterfeited during the transmission process. A smartphone with a Super SIM card first digitally signs the basic security messages (BSM) it advertises using the private key corresponding to the pseudonymous certificate. Then the smartphone sends that signed message along with the pseudonymous certificate or the digest value of the pseudonymous certificate. The application system that receives the message first uses the certificate of the certificate authority (CA) that issued the pseudonym certificate to verify whether the signed certificate in the message is valid and then uses the public key in the pseudonym certificate to verify whether the signature in the signed message is correct. Finally, the receiving initiator uses the verified BSM content to determine the sent data content.

Data Transmission Integrity: In authentication communication, it is necessary to ensure the integrity of the transmitted data and prevent the message from being tampered with during the transmission process. The integrity of the transmitted data can be guaranteed by using cryptographic hash algorithms (such as SM3) and digital signature algorithms (such as SM2). The smartphone equipped with the super SIM card uses the hash cipher algorithm to calculate the digest value of the BSM and sends the BSM together with the signature of the digest value; the application system that receives the BSM re-uses the hash cipher algorithm to calculate the digest value of the BSM, and use the signature public key of the authentication initiator to verify the signature of the digest value to ensure the integrity of the transmitted data.

Data Transmission Confidentiality: In authentication communication, it is necessary to ensure the confidentiality of critical data to prevent the essential data in the message from being leaked during transmission. Confidentiality of transmitted data can be guaranteed by using a symmetric cryptographic algorithm such as SM4. The authentication initiator, 4A system, and application system can obtain the symmetric encryption key through the key distribution platform or preset methods, use the symmetric key to encrypt the important information in the BSM, and then the main certificate initiator sends the BSM out. The application or 4A system that receives the message uses the symmetric key to decrypt the important information encrypted in the BSM.

Anti-replay Multiple parties in the authentication process can use timestamps or cache queues to prevent anti-replay attacks. The authentication initiator, application, and 4A should all judge whether the timestamp in the received message has expired. Both communication parties need accurate time synchronization and guarantee the time source from a legitimate authority, such as GNSS (Global Navigation Satellite System) time.

Behavioral Non-repudiation The authentication initiator uses the private key to sign the BSM digitally, and the applications and 4A systems that receive the data use the sender's public key to verify the signature data, preventing the sender from denying its behavior of sending the BSM.

Discovery and Disposal of Abnormal Behavior The enhanced 4A identity authentication capability based on super SIM technology also has security risks,

mainly from the following directions: 1) On the mobile terminal side, the user's mobile phone and the super SIM card may be lost, fraudulently used, or replaced with the super SIM card. 2) On the communication network side, there may be a network configuration error, which may cause it to fail to connect with the right SIM card authentication center or fail to connect with the 4A system. 3) On the application side, there may be security issues in the 4A system itself, leading to risks such as leakage, tampering, and unavailability of user identity data.

Therefore, the project must have various abnormal behavior detection mechanisms and handling plans during the operation process. For security issues on the terminal side, it is necessary to conduct data analysis on the behavior of terminals, SIM cards, and users and analyze possible physical intrusion, network hijacking, and user impersonation. If the secondary authentication fails, the device and SIM card will be temporarily locked and confirmed by the security operator through dialogue with the direct user. When addressing network-side security issues, the 5G network evaluation process should be combined to minimize network infrastructure's hidden dangers in advance. When addressing the security risks on the application side, it is indispensable to improve its security and simultaneously design a bypass plan for the system's unavailable risks.

The super SIM could be used in 5G, which is a research hotspot. 5G could be combined with the emerging blockchain technology [12] to realize better performance [13], scalability [14], and higher lever of security [15, 16]. Blockchain consensus [17,18] is the most important fundamental, which decides the the system performance and security [19]. Sharding blockchain [20, 21] can be used in 5G authentication protocol, combined with the super SIM as a trusted hardware, to process large-scale user requests. Moreover, the 5G and blockchain technology can be used to process the cloud storage, access control problems, etc. [22–24].

5 Conclusion

The enhanced 4A identity authentication center based on Super SIM technology was an important innovation that combined the 4A system covering the entire network with Super SIM technology. The system provided a new method of identity control for the problems of expanding business scale and complex team composition faced by enterprises in digital transformation. In the future, when identity control needs are more diversified, the identity control system must achieve higher security, smarter perception, and finer management and control. At the same time, the system will also need to balance the contradiction between security and convenience, and unify the contradiction between identity and business complexity. The main evolution direction of this project is to create a safe, convenient, highly compatible identity authentication system. This system will use the capabilities of Super SIM fully and current smartphones, making mobile phones a powerful identification tool for natural persons, thereby creating a modern identity control solution.

Acknowledgment

This paper is supported by the National Natural Science Foundation of China (U21B2021, 62202027, 61972018, 61932014), Yunnan Key Laboratory of Blockchain Application Technology (202105AG070005-YNB202206).

References

1. M. Qiu, Z. Chen, et al., Energy-aware data allocation with hybrid memory for mobile cloud systems, IEEE Systems J. 11 (2) (2014) 813–822.
2. K. Gai, M. Qiu, S. Elnagdy, A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance, in: IEEE BigDataSecurity, 2016.
3. M. Qiu, C. Xue, et al., Efficient algorithm of energy minimization for heterogeneous wireless sensor network, in: IEEE EUC Conf., 2006, pp. 25–34.
4. J. Niu, Y. Gao, et al., Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability, JPDC 72 (12) (2012) 1565–1575.
5. K. Gai, Y. Zhang, et al., Blockchain-enabled service optimizations in supply chain digital twin, IEEE TSC (2022).
6. M. Qiu, H. Qiu, et al., Secure data sharing through untrusted clouds with blockchain-enabled key management, in: 3rd SmartBlock conf., 2020, pp. 11–16.
7. S. Huang, B. Guo, Y. Liu, 5g-oriented optical underlay network slicing technology and challenges, IEEE Commun. Mag. 58 (2) (2020) 13–19.
8. Y. Li, K. Gai, et al., Intercrossed access controls for secure financial services on multimedia big data in cloud systems, ACM TMCCA (2016).
9. X. Gao, M. Qiu, Energy-based learning for preventing backdoor attack, in: KSEM (3), 2022, pp. 706–721.
10. A. V. Vinel, J. Breu, T. H. Luan, H. Hu, Emerging technology for 5g-enabled vehicular networks, IEEE Wirel. Commun. 24 (6) (2017) 12.
11. T. Wang, Y. Zhang, et al., An effective edge-intelligent service placement technology for 5g-and-beyond industrial iot, IEEE TII 18 (6) (2022) 4148–4157.
12. Y. Liu, J. Liu, Z. Zhang, H. Yu, A fair selection protocol for committee-based permissionless blockchains, Computers & Security (2020) 101718.
13. Y. Liu, J. Liu, J. Yin, G. Li, H. Yu, Q. Wu, Cross-shard transaction processing in sharding blockchains, in: ICA3PP 2020, 2020, pp. 324–339.
14. Y. Liu, J. Liu, et al., SSHC: A secure and scalable hybrid consensus protocol for sharding blockchains with a formal security framework, IEEE TDSC 19 (3) (2022) 2070–2088.
15. Y. Liu, J. Liu, et al., A secure shard reconfiguration protocol for sharding blockchains without a randomness, in: IEEE TrustCom, 2020, pp. 1012–1019.
16. Y. Liu, J. Liu, Y. Hei, Y. Xia, Q. Wu, A secure cross-shard view-change protocol for sharding blockchains, in: ACISP 2021, Vol. 13083, Springer, 2021, pp. 372–390.
17. Y. Liu, J. Liu, Z. Zhang, T. Xu, H. Yu, Overview on consensus mechanism of blockchain technology, Journal of Cryptologic Research 6 (4) (2019) 395–432.
18. B. Hu, Z. Zhang, et al., A comprehensive survey on smart contract construction and execution: paradigms, tools, and systems, Patterns 2 (2) (2021) 100179.
19. Y. Liu, Y. Xia, J. Liu, Y. Hei, A secure and decentralized reconfiguration protocol for sharding blockchains, in: IEEE HPSC 2021, IEEE, 2021, pp. 111–116.
20. Y. Liu, J. Liu, et al., Building blocks of sharding blockchain systems: Concepts, approaches, and open problems, Comp. Sci. Review 46 (2022) 100513.
21. E. Kokoris-Kogias, P. Jovanovic, et al., Omniledger: A secure, scale-out, decentralized ledger via sharding, in: IEEE SP 2018, 2018, pp. 583–598.
22. Y. Hei, J. Liu, et al., Making MA-ABE fully accountable: A blockchain-based approach for secure digital right management, Comput. Networks 191 (2021) 108029.
23. Y. Hei, D. Li, C. Zhang, J. Liu, Y. Liu, Q. Wu, Practical agentchain: A compatible cross-chain exchange system, Future Gener. Comput. Syst. 130 (2022) 207–218.
24. Y. Hei, Y. Liu, D. Li, J. Liu, Q. Wu, Themis: An accountable blockchain-based P2P cloud storage scheme, Peer-to-Peer Netw. Appl. 14 (1) (2021) 225–239.

Verifiable, Fair and Privacy-Preserving Outsourced Computation Based on Blockchain and PUF

Jiayi Li[1], Xinsheng Lei[2(✉)], Jieyu Su[3], Hui Zhao[4], Zhenyu Guan[3], and Dawei Li[3]

[1] Datang Microelectronics Technology Co., LTD, Beijing, China joelyb@163.com
[2] China Information and Communication Technology Group Co., LTD, Beijing, China leixsh@cict.com
[3] School of Cyber Science and Technology, Beihang University, Beijing 100191, China sujieyu@buaa.edu.cn, guanzhenyu@buaa.edu.cn, lidawei@buaa.edu.cn
[4] Henan University, Henan, China zhh@henu.edu.cn

Abstract. With the increasing maturity of *Industrial Internet of Things* (IIoT) technology, resource-constrained devices are widely applied to segments of the factory, which puts significant pressure on their computational capacity. In order to address this issue securely, we propose a verifiable and privacy-preserving outsourced computation system that employs SRAM PUF to safeguard the hardware security of devices and blockchain to achieve public verifiability and data privacy, thereby greatly guaranteeing the security of outsourced computation in the IIoT environment. Additionally, we protect the rights of calculators using a mechanism that identifies malicious calculators. Finally, compared with other existing schemes, the experimental results demonstrate that our scheme provides more efficient and secure outsourced computation services for IIoT devices.

Keywords: IIoT · Blockchain · Outsourced computation · PUF · Verifiability · Privacy.

1 Introduction

The *Industrial Internet of Things* (IIoT) is the extension of the *Internet of Things* (IoT) in industry [15,17,19]. At the advent of the fourth industrial revolution, applying IIoT technology will become comprehensive, eventually realizing the transformation of manufacturing processes. In the IIoT environment, massive data [3,8,13] makes it impracticable for resource-constrained devices [20–22] to process it locally. This practical necessity is effectively addressed by the emergence of outsourced computation.

As shown in Fig. 1, Edge outsourced computation is a new computing paradigm [32] of providing real-time computational services, which originates from outsourcing customers' computing overhead to the cloud [7]. The advancement of

edge outsourced computation benefits several applications in the IIoT environment, including data processing in supply chain optimization, smart grids and intelligent industrial parks. However, due to the distributed network environment, security and accuracy of outsourced computation cannot be protected [6]. Further, hardware security of IIoT devices is not fully guaranteed, which gives more weight to device authentication. PUF is a novel hardware primitive utilized in authentication, whose unclonability and low-power dissipation meet the needs of authentication in the IIoT system.

To address the aforementioned problems, we present a blockchain-based outsourced computation system with PUF and implement outsourced linear regression computation. *Linear regression* (LR) is a simple but efficient machine learning algorithm utilized in the IIoT environment [2, 23]. A standard model of linear regression is: $y = X\beta$, where y and β are $m \times 1$ and $n \times 1$ vectors in $\mathbb{R}^m$ respectively and X is an $m \times n$ matrix in $\mathbb{R}^{m \times n}$, for $m > n$ [1]. We apply the least squared error method to find an approximation of β using the formula $\beta = (X^T X)^{-1} X^T y$ [1]. Then, in order to verify the results, we introduce blockchain [4, 5] into our system, which is a decentralized ledger using cryptography to host applications, store data, and share information securely [18, 25].

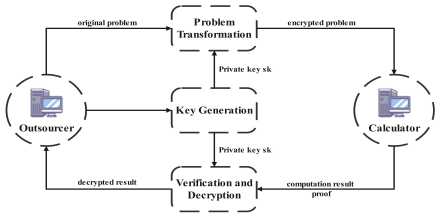


Fig. 1. The model of edge outsourced computation

The main contributions of our paper can be concluded as follows: (1) We propose a verifiable and privacy-preserving outsourced computation system using PUF and blockchain, which achieves data privacy and public verifiability and allows IIoT devices to access computing resources securely. (2) Our proposed scheme guarantees the hardware security of IIoT devices with low security levels utilizing SRAM-PUF-based authentication scheme, which only allows PUF-authenticated outsourcers to initiate outsourced tasks. (3) We introduce a mechanism to identify malicious calculators, allowing the selection of calculators on the blockchain. The experimental results indicate that our scheme not only is in milliseconds level with high verification success rate, which is more efficient than other related schemes, but also achieves privacy, verifiability, and fairness.

In the rest of this paper, Sect. 2 reviews the related work. Fundamental knowledge is interpreted in Sect. 3. Sect. 4 presents the system model and the threat model. We detail our scheme in Sect. 5. We implement our proposal and analyze the experimental results in Sect. 6. Sect. 7 concludes our work.

2 Related Work

Researches in outsourced computation cover diverse directions, the majority of which focus on data privacy and verifiability. In the study of data privacy, Liu et al. [16] presented a privacy-preserving outsourced computing scheme that employs Intel SGX as a trusted execution environment to guarantee data privacy. Compared with them, Zhang et al. [30] avoided introducing a trusted third party and proposed an IoT RSA outsourced computation scheme based on the Chinese residual theorem that simultaneously assures the privacy and security of the scheme. In the study of verifiability, Li et al. [12] devised a secure distributed outsourced computation system for modulo power operations, which combines logical partitioning and segmentation methods to verify the results. However, the verification can only be performed by users. Further, Li et al. [14] and Zhang et al. [29] achieved public verifiability. Li et al. [14] validated the results based on the Elgamal algorithm and protected data privacy. Zhang et al. [29] created a safe, equitable, and verifiable method to outsource linear regression computation to unreliable cloud, which enables public verifiability.

3 Preliminaries

3.1 Physical Unclonable Function

Physical Unclonable Function (PUF) is a hardware primitive employing intrinsic random deviations introduced during the manufacture, which provides a unique fingerprint for a physical entity. It inputs a challenge C and outputs an unpredictable and unique response R using its inherent unclonability [11]. Particularly, SRAM PUF generates response bits based on unpredictable initial values of SRAM cells that are determined by manufacturing process, which makes it convenient to be integrated into IIoT devices without additional changes [10]. Therefore, we employ SRAM PUF in the authentication phase of our system.

3.2 Sparse Matrix Masking

The core concept of *Sparse matrix masking* (SMM) technique is to mask sensitive data using a randomly selected sparse matrix, which can be utilized in matrix-related calculations to avoid expensive encryption operations and secure data privacy effectively [31]. In order to reduce computational consumption while protecting information related to zero elements, we introduce the definitions of $\chi-$sparse matrix and adaptive adjustment function in [31], which adjusts placements of non-zero elements according to non-zero elements of the input matrix, thereby concealing zero-element-related information of the original matrix.

4 Problem Formulation

4.1 System Model

Five entities compose our system as illustrated in Fig. 2: task outsourcers, task calculators, certificate authority(CA), authentication committee, and blockchain.

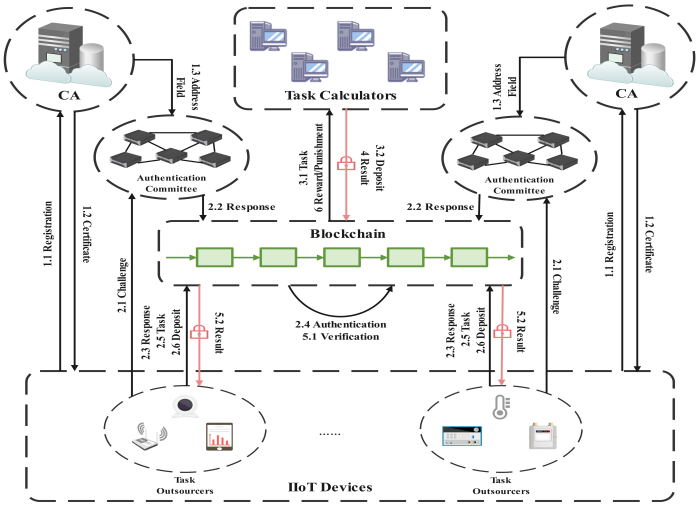


Fig. 2. System model

- Task Outsourcers and Task Calculators. The task outsourcers are resource-constrained devices initiating the outsourced tasks and the task calculators are resource-rich devices accomplishing the outsourced tasks.
- CA and Authentication Committee. CA is responsible for issuing certificates for task outsourcers and distributing the information of CRPs to nodes in the P2P network, which comprise the authentication committee. In our system, each participant in authentication committee holds a part of CRPs.
- Blockchain. Blockchain is a recorder of all transactions in the system that shows the outcomes of authentication and outsourced tasks.

4.2 Threat Model

We present our threat model in this part. The primary security threats originate from calculators and adversaries in the P2P network, therefore, we define identity unforgeability, privacy-preserving, and verifiable.

- Identity unforgeability. The adversary cannot forge the identity of low-security-level devices from the physical layer to steal their assets, and the forged authentication information of the adversary cannot be authenticated.
- Privacy-preserving. The outsourced computation should protect the privacy of the outsourced data that belongs to outsourcers, even if the openness and transparency of the blockchain-based system introduce the issue of privacy disclosure, which makes private data susceptible to eavesdropping.
- Verifiable. Under the premise of guaranteeing data privacy, the verification algorithm does not validate a calculator if the calculator submits inaccurate results in an attempt to obtain rewards.

5 Concrete Scheme

In this section, we provide the details of our scheme that consists of six phases: Setup, Task announcement, Task selection, Result calculation, Verification and Payment. We denote the linear regression calculation $\overrightarrow{y} = X \cdot \beta$ as $\Phi_{LR} = (X, \overrightarrow{y})$, where X is a $n_1 \times n_2$ matrix and $\overrightarrow{y}$, $\beta = (X^t \cdot X)^{-1} \cdot X^t \cdot \overrightarrow{y}$ are vectors with respective sizes of $n_1 \times 1$ and $n_2 \times 1$, where $n_1 > n_2$ [1].

5.1 Setup

KeyGen On inputting the security parameter λ and the size of matrix (n_1, n_2), the outsourcer O_i randomly creates a diagonal matrix A and a χ-Sparse matrix M^{χ}_{post} with respective size of $n_1 \times n_1$ and $n_2 \times n_2$ to secure data privacy. Specifically, A satisfies $A^t \cdot A = \varphi^2 \cdot I$, where φ is a random value chosen by O_i, I is an identity matrix of size $n_1 \times n_1$ and M^{χ}_{post} is constructed according to [31]. Set A and M^{χ}_{post} as the private key k, which is indicated as $k = (A, M^{\chi}_{post})$.

ProbGen On inputting the private key k, O_i produces an encrypted LR problem $\Phi^{'}{}_{LR} = (X^{'}, \overrightarrow{y}^{'})$. Firstly, O_i executes the function $M^{\chi}_{adj-post} \leftarrow adj(X, M^{\chi}_{post})$ in [31] to obtain matrix $M^{\chi}_{adj-post}$. Then, O_i calculates $X^{'} = A \cdot X \cdot M^{\chi}_{adj-post}$ and $\overrightarrow{y}^{'} = A \cdot \overrightarrow{y}$, which are outputted as encrypted inputs of the LR problem.

Registration This phase is performed jointly by CA, authentication committee and outsourcer which is embedded in a SRAM PUF. Firstly, O_i sends its identity to CA, who issues a certificate to O_i in response. Then CA randomly assigns the memory address fields of SRAM PUF to nodes in authentication committee [10].

5.2 Task Announcement

Authentication According to [10], first, O_i randomly chooses CRPs on the basis of the distribution of memory address fields and submits the challenges to authentication committee. Given the challenges, each node delivers the hash value of the response $H(R_i)$ to the smart contract based on its possession of the partial CRPs. After powering on the SRAM PUF, O_i concatenates the responses $R_1^{'}, R_2^{'}, \ldots, R_n^{'}$ into a string and signs it as $\sigma = Sig_{sk}(H(R_1^{'} \ldots R_n^{'}, r))$, where r is randomly chosen. Subsequently, O_i calculates the value of $R_{e_1}, R_{e_2}, \ldots, R_{e_n}$, where $R_{e_i} = g^{H(R_{i'})} (1 \leq i \leq n)$. Then, O_i sends σ and R_{e_i} to the smart contract, which checks the validity of the signature and the equation $g^{H(R_i)} \stackrel{?}{=} R_{e_i} (1 \leq i \leq n)$. The authentication of O_i will be successful if the equation and the signature are validated, otherwise, it fails.

Initialization On the premise that authentication is successful, O_i initiates the outsourced task $T_i = (ID, input(X^{'}, \overrightarrow{y}^{'}), \Phi^{'}_{LR} = (X^{'}, \overrightarrow{y}^{'}), reward, deposit, Time)$. Six parameters characterize T_i: ID denotes the task identifier; $input(X^{'}, \overrightarrow{y}^{'})$ is the input data; ${\Phi^{'}}_{LR} = (X^{'}, \overrightarrow{y}^{'})$ represents the description of the problem; $reward$ indicates the task reward; $deposit$ denotes the deposit required to be submitted; $Time$ is the deadline for the task. After initialization, O_i submits T_i to the smart contract.

5.3 Task Selection

In this phase, calculators select public tasks on the blockchain according to their computational resources by signing tasks and submitting the deposit. The selections will be confirmed if calculators pass the related validation.

5.4 Result Calculation

First, C_j calculates of $\beta^{'} = ({X^{'}}^{t} \cdot X^{'})^{-1} \cdot {X^{'}}^{t} \cdot \overrightarrow{y}^{'}$ locally, which guarantees data privacy. After that, C_j calculates the hash value $H = H(\beta^{'}||address)$ and submits it to the smart contract. Eventually, C_j submits $\beta^{'}$ and its public key address after the confirmation of H.

5.5 Verification

First, the smart contract validates the identity of C_j by comparing the hash value $H^{'} = H(\beta^{'}||address)$ with H. Then, it checks whether the equation $\Delta\overrightarrow{y}^{'} = \overrightarrow{y}^{'} - X^{'} \cdot \beta^{'}$ holds. If $\forall i \in \{1, \ldots, n\}$, $\Delta\overrightarrow{y}^{'}(i, 1)$ is within the standard error interval, the result will be accepted, otherwise, rejected. After validation, the result $\beta^{'}$ is recorded on the blockchain, which can be decrypted using the private key k of O_i according to the formula $\beta = M^{\chi}_{adj-post} \cdot \beta^{'}$.

5.6 Payment

In this phase, all honest calculators that submit correct results on time are rewarded, while the ones that submit inaccurate results or fail to complete the task before deadline are deemed malicious. Calculators marked as malicious are forced to submit more deposits than honest ones and their deposits are fully deducted for the purpose of protecting the rights of all calculators. As all nodes are considered rational, we believe that this strategy can effectively decrease malicious calculators in the system.

6 Performance Evaluation

In this section, We first compare our scheme with other existing works, and then we evaluate the time cost and the predictive performance of our scheme. The scheme is performed using the Combined Cycle Power Plant Data Set from the UCI Machine Learning Repository [9,26]. Finally, we prove that our scheme achieves the security requirements.

6.1 Performance Comparison

We compare the performance of our scheme with some existing works, including verifiability, fairness, privacy, authentication, and security. Based on the comparison results displayed in Table 1 using "Yes" and "No", we can observe that our scheme is more advanced than other related works.

Table 1. Performance comparison with existing schemes.

Ref.	Verifiable	Fairness	Privacy	Authentication	Security
[27]	No	Yes	Yes	No	Yes
[24]	No	Yes	Yes	No	Yes
[28]	Yes	Yes	Yes	No	Yes
[14]	Yes	Yes	Yes	No	Yes
Our scheme	Yes	Yes	Yes	Yes	Yes

6.2 Prototype Evaluation

Our scheme is implemented on a consortium blockchain using Hyperledger Fabric v2.3. We deploy Go-language-based smart contracts on a desktop with a quad-core processor, 4GB RAM and 20GB memory, which is running Ubuntu 22.04.1. The implementation can be separated into three parts. In the first part, we evaluate the time cost of each stage multiple times and average the results. From Fig.3, we can clearly see that the overhead of all stages is milliseconds, which is efficient and applicable in the IIoT environment. Compared with the scheme mentioned above, our scheme is more advanced in that it achieves identity unforgeability, data privacy and public verifiability with acceptable time consumption. Then, we analyze the time complexities of our algorithms as shown in Table 2.

Table 2. Time complexities of our proposed algorithms.

Setup	Task announcement	Task selection	Result calculation	Verification	Payment
$O(n_1^2)$	$O(n)$	$O(1)$	$O(n_2^3)$	$O(n_2^2)$	$O(1)$

In the second part, we evaluate the ratio of verification that successfully identifies malicious calculators and show the resistance of our scheme to malicious calculators in Fig.4. With the percentage of malicious calculators varying from 10% to 50%, a graceful increase in the ratio of verification that identifies incorrect results can be observed, which proves the effectiveness of the verification.

In the third part, we evaluate the predictive performance of the linear regression model. The comparison between measured and predicted data is presented in Fig.5. As we can see, the majority of the predicted results fall in the standard

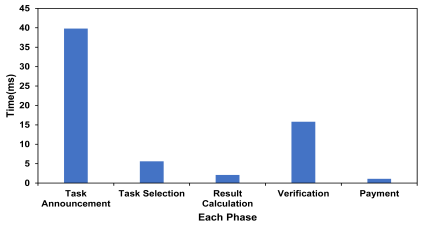


Fig. 3. The time consumption of each stage

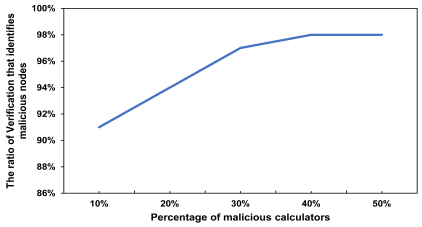


Fig. 4. The success ratio of verification that identifies malicious calculators

error interval indicating that the predictive performance satisfies the requirements of the IIoT environment, which demonstrates the practical application value of our proposal.

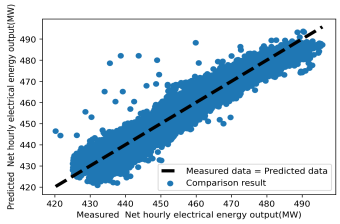


Fig. 5. Comparison between measured and predicted data

6.3 Security Analysis

We analyze the security requirements in this section, including identity unforgeability, privacy-preserving, and verifiable.

- Identity unforgeability. We assume that adversaries attack the hardware of devices in the system from the physical layer to forge their identity, which can be detected through authentication. Due to the unclonability of PUF, this attack changes the CRPs of PUF and consequently leads to the failure of authentication. Devices that fail to pass the authentication can not publish outsourced tasks in our system, as a result of which the assets of devices are protected.
- Privacy-preserving. In our system, a malicious entity can eavesdrop private and sensitive data contained in the outsourced tasks from the transmission channel and blockchain. Therefore, we use the SMM technique to encrypt the original input data so that malicious eavesdroppers can not decrypt private information from public task-related data in the P2P network and the privacy of data is protected.

- Verifiable. We assume that malicious calculators submit inaccurate results in order to obtain rewards, which can be detected with the verification algorithm in our scheme. If a calculator tries to submit results without accomplishing the calculation correctly, the submission shall not pass the verification phase as the smart contract fails to verify the equation mentioned in Sect. 5. The malicious calculators will be punished to guarantee the rights of both outsourcers and calculators.

7 Conclusion

In this paper, we proposed a new edge outsourced computation scheme for the IIoT environment. Our scheme adopted SRAM PUF to accomplish device authentication and guarantee their hardware security. To the best of our knowledge, our scheme was the first outsourced computation scheme that achieved hardware security of IIoT devices. In addition, to verify the accuracy of results and protect sensitive data, the scheme utilized blockchain and SMM technologies to achieve public verifiability and data privacy. Furthermore, the scheme introduced a mechanism for identifying malicious calculators. Our scheme provided a verifiable, fair and privacy-preserving outsourced computation method for resource-constrained IIoT devices with low security level. Eventually, the experimental results illustrated that our scheme was at the millisecond level with high verification success rate and achieved data privacy, public verifiability and identity unforgeability, proving our scheme to be feasible and fair.

References

1. Chen, F., Xiang, T., Lei, X., Chen, J.: Highly efficient linear regression outsourcing to a cloud. IEEE transactions on cloud computing **2**(4), 499–508 (2014)
2. Ciulla, G., D'Amico, A.: Building energy performance forecasting: A multiple linear regression approach. Applied Energy **253**, 113500 (2019)
3. Gai, K., et al.: A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance. In: IEEE BigDataSecurity (2016)
4. Gai, K., Fang, Z., et al.: Edge computing and lightning network empowered secure food supply management. IEEE IoT J. **9**(16), 14247–14259 (2022)
5. Gai, K., Hu, Z., et al.: Blockchain meets dag: A blockdag consensus mechanism. In: Algor. and Arch. for Parallel Proc. pp. 110–125 (2020)
6. Gai, K., Wu, Y., et al.: Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks. IEEE IoT J. **6**(5), 7992–8004 (2019)
7. Gai, K., Wu, Y., et al.: Differential privacy-based blockchain for industrial internet-of-things. IEEE TII **16**(6), 4156–4165 (2020)
8. Hu, F., Lakdawala, S., et al.: Low-power, intelligent sensor hardware interface for medical data preprocessing. IEEE TITB **13**(4), 656–663 (2009)
9. Kaya, H., Tüfekci, P., Gürgen, F.S.: Local and global learning methods for predicting power of a combined gas & steam turbine. In: conf. on emerging trends in comp. and elec. eng. (ICETCEE). pp. 13–18 (2012)
10. Li, D., Chen, R., et al.: Blockchain-based authentication for iiot devices with puf. Journal of Systems Arch. **130**, 102638 (2022)

11. Li, D., Ren, Y., Liu, D., et al.: Puf-based intellectual property protection for cnn model. In: Knowledge Science, Engineering and Management. pp. 722–733 (2022)
12. Li, H., Yu, J., Zhang, H., Yang, M., Wang, H.: Privacy-preserving and distributed algorithms for modular exponentiation in iot with edge computing assistance. IEEE Internet of Things Journal **7**(9), 8769–8779 (2020)
13. Li, J., Ming, Z., et al.: Resource allocation robustness in multi-core embedded systems with inaccurate information. J. of Systems Arch. **57**(9), 840–849 (2011)
14. Li, T., Tian, Y., Xiong, J., Bhuiyan, M.Z.A.: Fvp-eoc: Fair, verifiable, and privacy-preserving edge outsourcing computing in 5g-enabled iiot. IEEE Transactions on Industrial Informatics **19**(1), 940–950 (2023)
15. Li, Y., Gai, K., et al.: Intercrossed access controls for secure financial services on multimedia big data in cloud systems. ACM TMCCA (2016)
16. Liu, Z., Hu, C., Li, R., Xiang, T., Li, X., Yu, J., Xia, H.: A privacy-preserving outsourcing computing scheme based on secure trusted environment. IEEE Transactions on Cloud Computing (2022)
17. Qiu, H., Dong, T., et al.: Adversarial attacks against network intrusion detection in IoT systems. IEEE Internet of Things J. **8**(13), 10327–10335 (2020)
18. Qiu, H., Qiu, M., et al.: A dynamic scalable blockchain based communication architecture for IoT. In: Smart Blockchain. pp. 159–166 (2018)
19. Qiu, M., Chen, Z., et al.: Energy-aware data allocation with hybrid memory for mobile cloud systems. IEEE Syst. J. **11**(2), 813–822 (2014)
20. Qiu, M., Jia, Z., et al.: Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP. JSPS (2007)
21. Qiu, M., et al.: Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems. In: IEEE DATE. pp. 1–6 (2007)
22. Qiu, M., et al.: Dynamic and leakage energy minimization with soft real-time loop scheduling and voltage assignment. IEEE TVLSI **18**(3), 501–504 (2009)
23. Qiu, M., Thuraisingham, B., et al.: Special issue on robustness and efficiency in the convergence of artificial intelligence and IoT. IEEE IoT J. **8**(12), 9460–9462 (2021)
24. Shao, J., Wei, G.: Secure outsourced computation in connected vehicular cloud computing. IEEE Network **32**(3), 36–41 (2018)
25. Tian, Z., Li, M., Qiu, M., Sun, Y., Su, S.: Block-def: A secure digital evidence framework using blockchain. Information Sciences **491**, 151–165 (2019)
26. Tüfekci, P.: Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods. International Journal of Electrical Power & Energy Systems **60**, 126–140 (2014)
27. Yang, H., Su, Y., et al.: Privacy-preserving outsourced inner product computation on encrypted database. IEEE TDSC **19**(2), 1320–1337 (2022)
28. Yu, X., Yan, Z., Zhang, R.: Verifiable outsourced computation over encrypted data. Information Sciences **479**, 372–385 (2019)
29. Zhang, H., Gao, P., et al.: Machine learning on cloud with blockchain: a secure, verifiable and fair approach to outsource the linear regression. arXiv preprint arXiv:2101.02334 (2021)
30. Zhang, H., Yu, J., et al.: Efficient and secure outsourcing scheme for rsa decryption in internet of things. IEEE IoT J. **7**(8), 6868–6881 (2020)
31. Zhao, L., Chen, L.: Sparse matrix masking-based non-interactive verifiable (outsourced) computation, revisited. IEEE TDSC **17**(6), 1188–1206 (2018)
32. Zhao, M., Hu, C., et al.: Towards dependable and trustworthy outsourced computing: A comprehensive survey and tutorial. JNCA **131**, 55–65 (2019)

Architecture Search for Deep Neural Network

Xiangyu Gao[1], Meikang Qiu[2] ✉*, and Hui Zhao[3]

[1] New York University, New York City, New York, USA xg673@nyu.edu
[2] Dakota State University, Madison, SD 57042, USA Meikang.Qiu@dsu.edu
[3] Henan University, China, zhh@henu.edu.cn

Abstract. Deep learning has become a popularly used tool in large amount of applications. Given its ability to explore the input and output relationship, deep learning can perform well in terms of prediction. However, one important drawback in this framework is that the model cannot be easily trained due to huge search space and large number of parameters. In response to such problem, this paper proposes a novel way to build a deep neural network. Specifically, it tries to consider an unbalanced structure of deep neural network by expanding the number of nodes in the beginning to extract as much information as possible and then shrinking quickly to converge to the final result. The experiment results show that our proposed structure can output equally good output with much faster time compared with traditional methods.

Keywords: Deep Learning · Machine Learning · Neural Architecture Search · Deep Neural Network · Information Set · Back Propagation.

1 Introduction

Deep learning has become an essential tool across different fields including cyber security [32, 31, 37], stock trading [49] and Go playing [46]. Specifically, as for complex problems, deep learning tries to build a layer-by-layer structure so as to precisely capture the relationship between the input features and output results. Most of these relationships are nonlinear, so cutting it into different layers and learning the relationship through deep neural network model is a good option. For example, Gao and Qiu [10] consider the energy-based learning as the tool to prevent backdoor in various situations.

Due to the good performance of deep learning techniques, people are always trying to explain the reason why deep learning can do better than traditional machine learning methods. Artzai et al. [28] attribute this to the fact that the multi-level structure can provide the system a way to learn complex functions to map input to the output. This also reflects the motivation of the invention of deep learning, which tries to simulate the human brain structure and behave like the cell-to-cell connection. Therefore, with the development of computation technology, one famous deep learning machine AlphaGo [46] can leverage deep neural network models to easily beat top-level human being Go players.

* Corresponding author

However, there are still several problems existing in the deep learning framework [23]. First of all, it is usually hard to explain the result of deep learning because the complex structure, including many layers and a lot of nodes, makes itself impossible to clearly show the correlation between the input and the output. In addition, even with Hecht-Nielsen's back propagation [14] method, it still takes too long to train the deep learning model.

Recently, Kung [20] proposed an effective neural architecture search algorithm to find useful DNN structures for training. Specifically, it comes up with an *X-learning Neural Architecture Search* (XNAS) to automatically train the network's structure and parameters, which seems to be more competitive than state-of-the-art approaches. However, different from the previous approaches, we think if it is possible for the user decouple structure exploration and model training, this can save a lot of time to do deep learning model training because we only need to focus on one concrete model to train the deep neural network.

In order to speed up the training process of deep learning model, we claim that setting up the concrete and effective structure of *Deep neural network* (DNN) beforehand should be helpful for the future model training process. Accordingly, in this paper, we come up with a novel technique to build the structure of DNN in order to speed up the training speed. To be specific, we consider the DNN as a structure to extract useful information from the input features so as to quickly build the connection between input and output. Therefore, we want to include as much information as possible in the initial several layers.

However, too many nodes within the DNN will increase the training difficulties a lot. Our algorithm aims at reducing the number of nodes dramatically layer by layer until the final one. To be specific, we will set the number of nodes in the first layer to be equal to the number of input features. Then the number of nodes in the next layer would be half of the number of nodes in the previous layer. We will also guarantee that each layer (except the output layer) should have more than one node.

There are a lot of benefits of such DNN architecture design. In the beginning, this design can help the structure avoid losing any important information. As the data goes through layers, more important information has been extracted, which means there is no need to remain the number of nodes the same as layers before. In order to reduce the training complexity, it is reasonable to reduce the number of nodes in the following layers. We believe this should be an effective way to find a trade-off between fitting accuracy and training speed in DNN. There are three main contributions of this paper:

- We give a detailed analysis of deep learning to find some possible reason of its low training speed.
- We propose an deep neural network structure that can quickly capture the input and output relationship.
- We compare our proposed structure with several existing structures to illsutrate the benefits of our proposal.

The remainder of this paper has the following structure. Section 2 summarizes the background of deep neural network and overviews several up-to-date

structure learning techniques in the domain of DNN. Then, Section 3 starts from a simple example and formally presents the algorithm design of our proposed architecture search algorithm for DNN. Furthermore, Section 4 gives the implementation details together with several experiment results. Finally, conclusions are shown in Section 5.

2 Background

In this section, we will provide a brief description of the deep neural network and explain the pros and cons of DNN. Then, several related structure learning techniques will also be mentioned to show some of the current trials to find the close-to-optimal structure of the DNN.

2.1 Deep neural network

With the rapid development of computer capability [43, 39, 44], new algorithm design [41, 42, 26], and cloud computing [22, 36, 30], large amounts of data [9, 21, 16] can be generated in a very short time period. Hence, this demands fast processing capability of big data. Machine learning [35, 33, 34] and artificial intelligence are the results of this big demand. There are many different learning models. DNN [4] is one important part of a broader family of machine learning methods based on artificial intelligence. It has become a widely used technique across several fields. The general structure of DNN is shown in the figure 1.

At a really high level, one DNN framework consists of two main components: layer and node. The first layer is called the input layer while the last layer is called the output layer. All layers between input and output layer are all regarded as hidden layers. The data would come from the input layer and then pass through all layers before arriving at the last layer, which stores the output information. Each node will represent a value and be used as the inputs to all nodes in the next layer. Usually, the nodes' values in the next layer will be a linear combination of all nodes in the layer before and their output will be determined by the activation functions chosen by the users. Popularly used activation functions include linear combination, sigmoid, tanh and relu. Selecting the best activation function [13] in different scenarios remains an interesting research topic in DNN fields.

In fact, DNN achieve good performance in many areas, such as various software [50, 52, 45], complicated systems [29, 40], and parallel structures [38, 17, 53]. Even though people do not have official proof of why DNN can work well under a lot of scenarios, there are several intuitive explanation of such phenomenon. After all, the motivation to develop the DNN structure originates from people's idea to simulate the function in human's brain. They want to find a way to let the machine understand new knowledge similar to how people learn new stuff.

In reality, most of the input/output relationship cannot be easily captured by a simple function. Therefore, the DNN is able to divide such complex relationship into multiple stages, each of which can be determined by one simple linear relationship. Then, the combination of multiple layers' simple function can

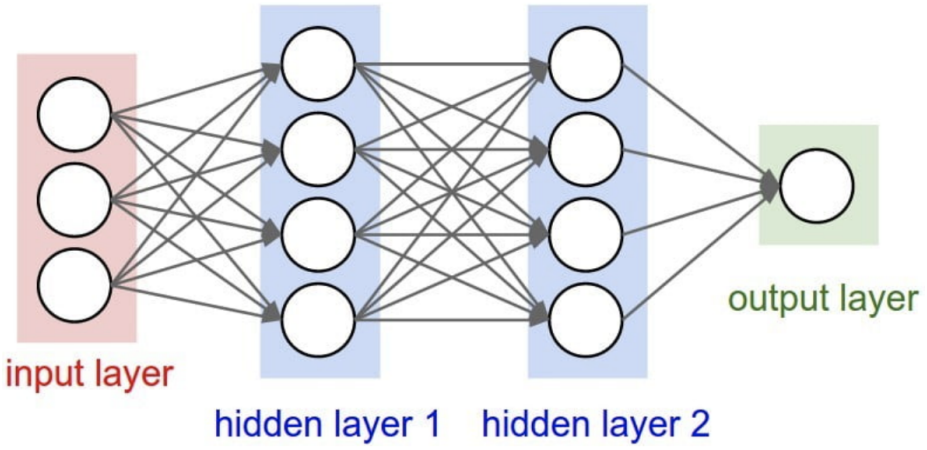


Fig. 1. The visualization of deep neural network.

precisely understand the complex input/output relationship if sufficient data is provided to train the model.

Therefore, as is shown in Figure 2, when the amount of training data is small, deep learning and old machine learning techniques will generate quite similar outcomes. However, as the amount of data increases, the deep learning will significantly out-perform the old machine learning techniques. Hence, We can draw the conclusion that in the current era of big data, deep learning will be a better choice.

However, DNN has several drawbacks [1]. Here, We will list two most important ones. First of all, the training time of DNN structure would be quite long due to a large number of nodes and complex layer-to-layer connection relationship. Some techniques [19], including back propagation [14], have been proposed to speed up the process, but they are still not sufficient. In addition, DNN, similar to many other machine learning techniques, is hard to explain. This brings problems that people sometimes cannot differentiate between scenarios where DNN is a suitable model or scenarios that overfitting [6] happens. Therefore, even if the fitting accuracy over the training set is good for DNN, people are not willing to totally rely on DNN for new datasets, which strictly restrict the number of application fields of DNN.

2.2 Structure searching techniques

An important improvement in DNN is related to the structure learning techniques development. To be specific, when selecting DNN as our target model structure, people usually want to do the model selection together with model training. In other words, they can only provide a general model, but more details (e.g., number of layers, number of nodes per layer) will be finalized during the training process. This increase the difficulty-level of the training process.

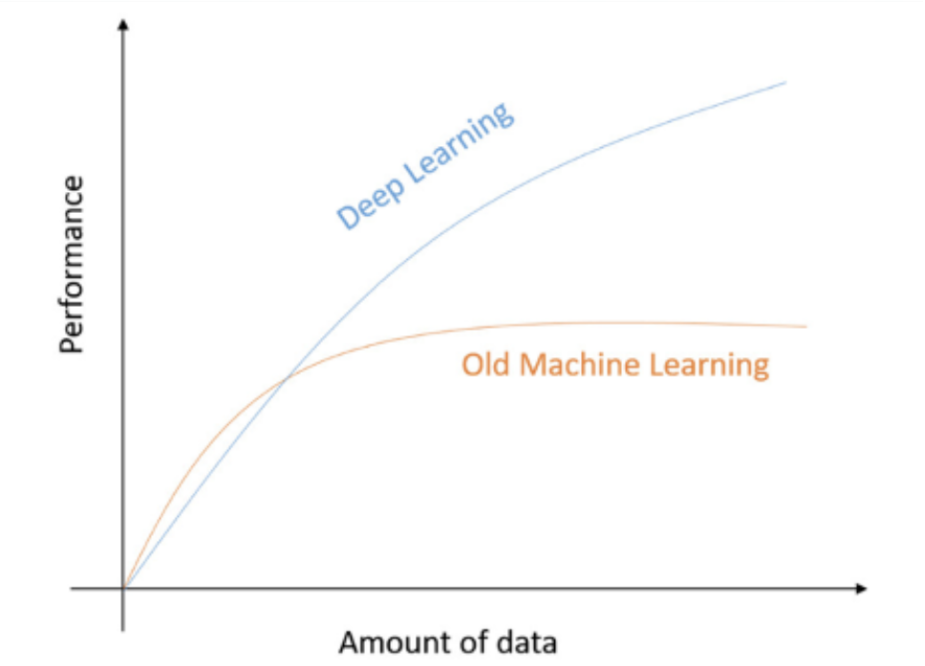


Fig. 2. deep learning vs traditional machine learning.

Just as shown in Figure 3, *Neural Architecture Search* (NAS) is a technique for automating the design of *Artificial Neural Networks* (ANN), a widely used model in the field of machine learning. It has three main components: the search space defines the type(s) of neural networks that can be designed and optimized; the search strategy defines the approach used to explore the search space; the performance estimation strategy evaluates the performance of a possible neural network from its design. By now, NAS methods have outperformed several manually designed architectures on some tasks including image classification [54] and semantic segmentation [5].

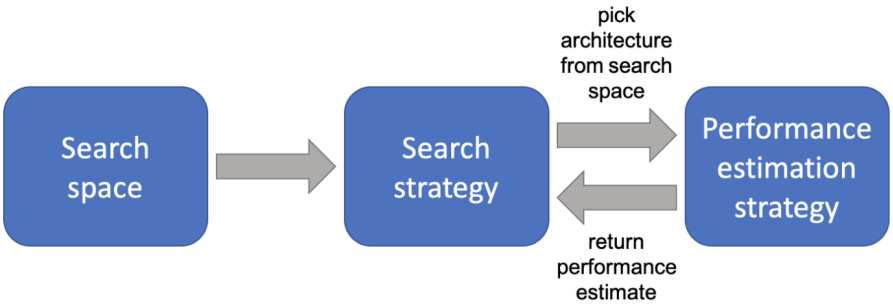


Fig. 3. Steps to do neural architecture search.

There have already been many related works [7] in this field. For instance, Kung [20] proposed X-learning NAS to automatically train network's structure and parameters, which is built based on the subspace and correlation analyses between the input and output layers. Cai et al. [7] framed NAS as a sequential decision process: regarding the state as the current (partially trained) architecture, the reward as an estimate of the architecture's performance, and the action as an application of function-preserving mutations.

Although they are effective neural architecture search algorithm, sometimes the training process is still not short. In addition, given the fact that it is sometimes hard for everyone to get access to powerful computation machines, people sometimes are not willing to consider DNN as the best choice for them to find correlation between input variables and output results.

3 Our Approach

In response to the problems shown in the previous section, we will present our approach in this section. To be specific, in order to deal with the problem that it takes too long to train a DNN model, we start from one motivation example which offers a modified DNN model with smaller search space. Then, a formalized algorithm will be provided in the following subsection together with our analysis of such algorithm

3.1 A small example

General DNN structure is illustrated in the LHS of Figure 4. Assume we have four layers in total, the number of nodes in each layer is the same across layers. In addition, all nodes of two adjacent layers are fully connected by linear functions. The assumption behind is that it will collect all information from the input set and then find the correlation between input and output nodes.

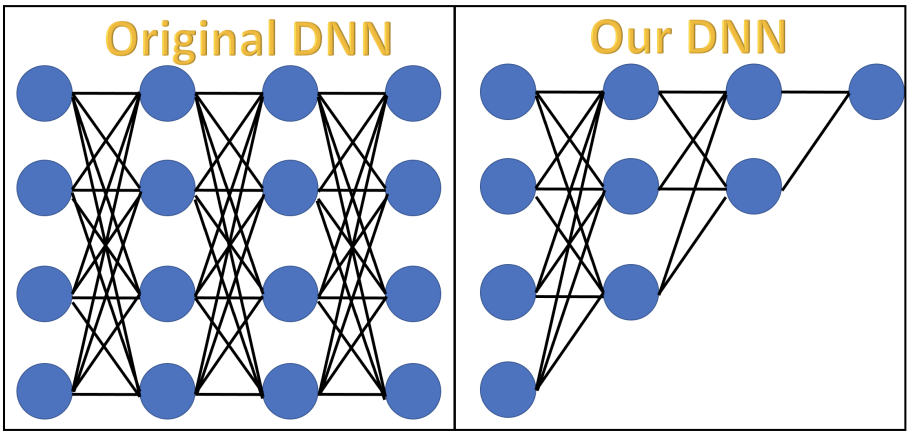


Fig. 4. Comparison of our approach and original DNN (suppose the total number of layers is four).

However, this might easily cause a problem: when we need to train the DNN, it might require too many computation resources and long training time. Specifically, in some scenarios where multiple layers are needed, the total number of nodes will increase linearly, causing the training complexity [25] to increase exponentially.

We believe it is easy to reduce the training complexity by considering another better DNN structure. If we retrospect the DNN structure shown on the LHS of Figure 4, it is usually unnecessary for us to put so many nodes in the last layer. Usually, if people use DNN to do classification problem, the output is just one node with binary values. Giving too many nodes to the last layer is redundant. In addition, it is highly possible that most of the useful information can be stored with only a few number of nodes, making it unnecessary to keep the number of nodes the same across layers.

Therefore, we give a new DNN structure shown on the RHS of Figure 4. In other words, given the situation that there are four layers in total, it keeps the number of nodes equal to the number of input features in the first layer. Afterwards, we reduce the number of nodes in each layer and leave with only one node in the last layer. Given this is just a motivation example, there are

some slight difference between this example and the proposed algorithm. But the main idea behind is that we want to remove the redundant nodes in DNN structure so as to speed up the training process. Since we also want to keep the fitting accuracy of our framework, we believe the structure with decreasing number of nodes as the layer increase will be the good solution.

Compared with original DNN, our proposed structure reduces the number of nodes by almost a half (16 vs 10) and simplify the training process. We claim that this structure can generate equally good training result with much faster speed especially for classification problems.

3.2 Algorithm design

We are now presenting the concrete algorithm. The general idea is presented in Figure 4 but concrete steps of our algorithm will present more details to show how to build a new structure for DNN.

Our algorithm will take as input the number of input features, the activation function, the threshold of nodes in each layer and the total number of layers. Then, it will set the number of nodes in the input layer to be equal to the number of input features so that we can map each feature into one node, which can guarantee that there is no information loss in the beginning. Then, as the data goes through the layer, it will cut the number of nodes by half because most of the nodes in the future layers would be redundant. Cutting the number of nodes can save a lot of search space. However, in order to prevent removing too many nodes, this algorithm will force the number of nodes in each layer to be at least as large as the threshold given by the users. This can guarantee that we do not over-reducing necessary information.

3.3 Algorithm analysis

The intuition behind our proposed algorithm can be explained as follows. In the beginning, it wants to collect as much information as possible, which forces its number of nodes in the input layer to be equal to the total number of input features. However, it would be a waste if we keep the number of nodes the same across layers because the goal of DNN is to collect useful information but drop useless one. Therefore, in order to strike a balance between fitting effect and the complexity of DNN model, we reduce the number of nodes by half layer by layer so as to reduce the number of necessary nodes exponentially. If we compare our proposed DNN structure against the original DNN structure, our search space is much smaller, leading to the fact that our training time will be much shorter than original DNN model.

We believe this algorithm can have good performance in scenarios where the final output is only a binary variable (e.g., whether the price is over or below the average). As for the classification problem, we only need one nodes in the output layer to determine the final output, reducing the nodes in each layer will not have too many negative effects on the fitting results. Extending such structure design

Algorithm 1 ***DNN structure generation*** Algorithm

Require: M input features, activation function of DNN F, total number of layers of DNN N, accuracy level L, and threshold of nodes K per layer.
Ensure: The threshold K must be smaller than the dimension M of input features.
Ensure: The threshold K must be greater than 2.

1: Set up a Deep Neural Network structure with only N layers (one input layer, one output layer and N - 2 hidden layer).
2: Set the number of node in the input layer equal to M.
3: Set the number of node in the output layer equal to 1.
4: Set the number of nodes in each hidden layer
for i = 1; i < M - 1; i++:
Set the number of node in the layer i to be max(number of nodes in previous layer/2, K).
5: Train the DNN on the input data set for several epochs until the accuracy is greater than or equal to accuracy level L.
6: Output the DNN training result together with the training time (number of epochs) required to generate such result.
Output results: The final results include all parameters of the trained DNN framework together with the fitting score.

from classification problem to non-classification problem will be an interesting direction for future work.

4 Implementation and Experiments

This section is mainly about the implementation details and the experiment results of our proposed algorithm through showing its comparison against other model benchmarks. In the traditional DNN training process, people would directly feed the data into the neural network pipeline and train such model using feedforward [47] and backpropagation methodology. We want to see how much improvement our algorithm can provide. We are concerned of three main metrics: accuracy [18], loss [48] and training time [3].

Accuracy is an important determinant since it directly decides how good the correlation between input variables and output variables could be learnt by DNN. Besides, we also want to get the reasonably good result as soon as possible; hence, the training time is another dimension people pay attention to. In our experiment setup, we regard the number of epochs required to reach some specific level as the metric to measure the length of training time. Since usually the fitting result of DNN will keep improving as the number of epochs increase, we want to see whether it is possible for our proposed structure to quickly learn the correlation between input features and output results.

4.1 Data description

In our experiment, we leverage our model to fit the open source data of house price [2]. This is because the data set is popularly used by researchers to test

DNN model. Usually the price of a house depends on many features, so within the data set, the output Y variable is binary, where 1 means the price is above the median and 0 means the price is below the median. Ten features (e.g., overall quality, overall condition, distance to surrounding utilities) are considered as important determinants to the final price. There are 1,460 data points in total within the data set.

We utilize related packages in python (e.g., sklearn [27], keras [12]) to implement our algorithm. Concretely speaking, house price data is stored in the csv format. We read the data from csv file and store it into the pandas data frame. Then, we implement several preprocessing techniques into the data with inserted packages such as standardization. The standardized data is split into training set and testing set. In our setting, we put the proportion of training data to be 70% of the whole data set while the remaining 30% are used as the testing set [11]. The DNN model from keras package will be trained in the training data and we test its performance over testing data set.

4.2 Benchmark selection

As for benchmarks, we mainly consider the original DNN with the number of nodes the same across layers as the candidates. Since our approach is an improvement based on the original DNN model, this should be the best candidate we want to compare against. To be specific, we let the total number of layers of DNN to be four. As for the number of nodes, we set it to be ten because of ten features in total for the input data. The number of nodes is the same across all layers for the original DNN model while as for our proposed DNN model, it has five, two and one node for the second, third and last layer.

There are several dimensions we can do to increase the number of benchmarks. For instance, we can set the number of layers to be a variable and generate a bunch of different DNN models. In addition, we can also randomize the number of nodes per stage. In this paper, we mainly focus on the comparison of our deterministic algorithm against the original DNN structure.

In the future, we also want to extend our approach to other deep learning models [8] [24] to show the benefits of such proposal.

4.3 Results

When it comes to the results comparison, we compare our method against original DNN model because we want to see how much improvement such proposed algorithm can bring. As for the metrics, we take into consideration accuracy rate [18], loss value [48] together with required time spent training the model.

The comparison results are illustrated in figure 5, figure 6, figure 7, and figure 8. In general, given the high accuracy and low loss value, we can see that DNN should be a suitable candidate model for house price prediction. It means that DNN is a suitable model for people to use to predict the house price over this given data set. In reality, given the fact that whether the house price is

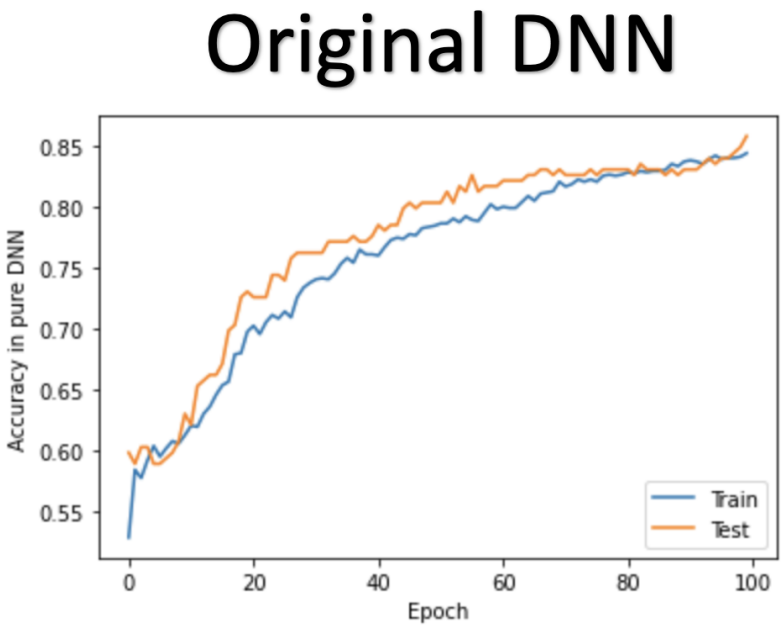


Fig. 5. Accuracy value of original DNN.

above or below the median should be determined by a series of factors, DNN should be better than other simple models.

We can also find other interesting phenomena from the figures. First of all, it is easy to witness that the big trend is that the fitting results get better and better as the number of epochs increases, which conforms to the common sense that longer training time will lead to better fitting result [15]. In addition, we can find that the improvement speed of the accuracy decreases as the training time increases. It means that in the beginning, a fixed number of training epochs can improve the accuracy a lot but further improvement will take much longer training time.

Therefore, when using DNN as a candidate structure, people do not want to reach 100% accuracy but will stop whenever the accuracy reaches a threshold. Otherwise, they have to wait too long for the training process to stop. Spending too much time increasing the accuracy might have some side effects such as overfitting [51].

Compared with the original DNN, our DNN model achieve the similar (even better) fitting effect from Figure 5 and Figure 6. Specifically, the accuracy of the testing set is always higher than the training set and after 100 epochs, the accuracy increases to a level greater than 85 percent. In comparison, the accuracy of traditional accuracy is always lower than 85 percent. When it comes to the loss value in Figure 7 and Figure 8, our approach is slightly lower than original DNN

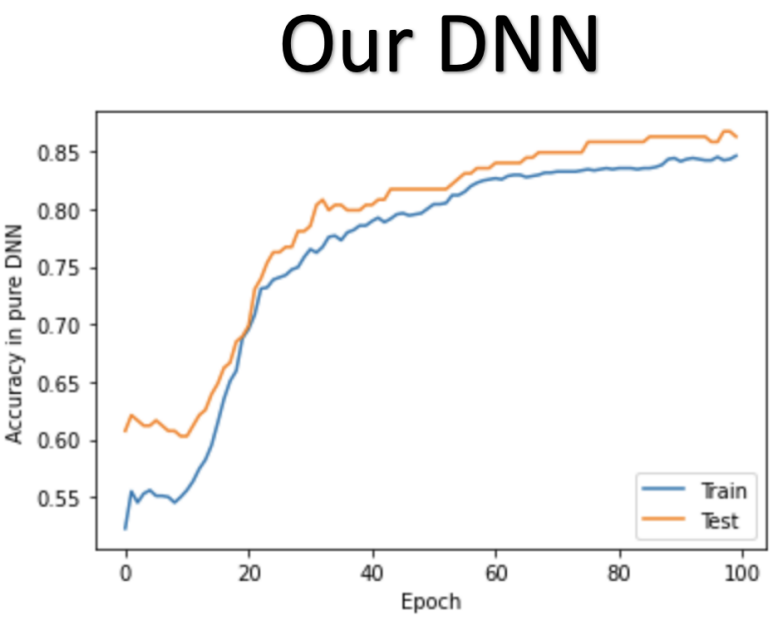


Fig. 6. Accuracy value of our DNN structure.

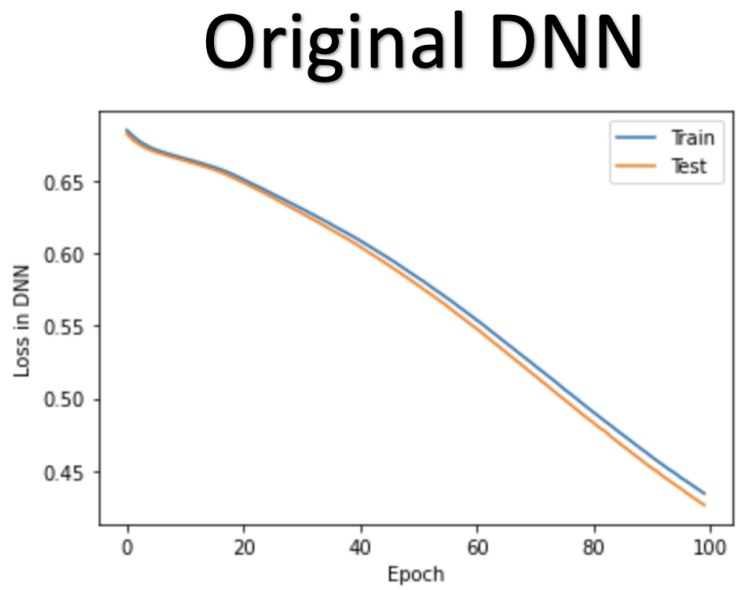


Fig. 7. Loss value of original DNN.

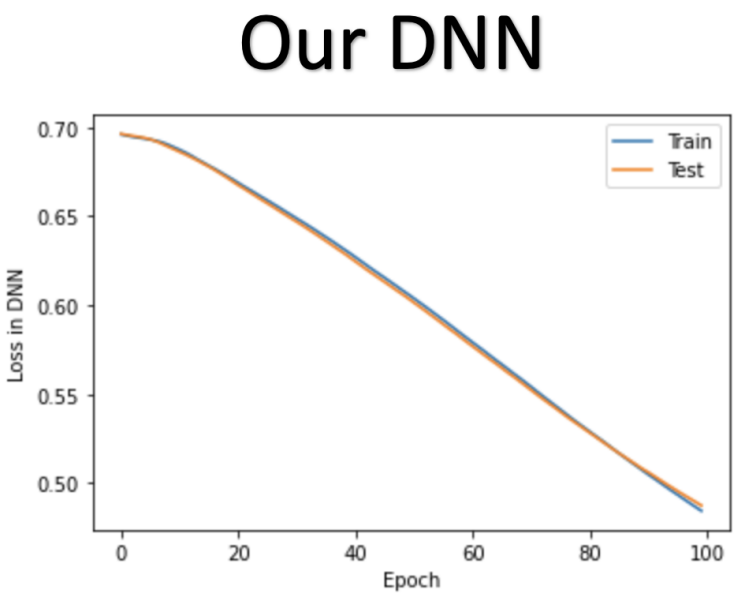


Fig. 8. Loss value of our DNN structure.

structure but they are very close to each other (45% vs 50%). How to reduce the loss rate for our proposed DNN to a level that is equivalent to original DNN should be a next step worthwhile to explore.

In addition, we also measure the training time of both original DNN and our DNN, which is the main part to show the benefits of our proposal. It is easy to see that our approach converges much faster than original DNN when setting the accuracy threshold to be 80 percent. For instance, it only takes around 40 epochs for our DNN model to reach accuracy level of over 80% while original DNN requires more than 60 epochs. We can see that our proposed DNN can be trained more quickly to reach good accuracy result. This means that our model is simpler to train and faster to reach a level with relatively high accuracy.

5 Conclusions

In this paper, we presented a new structure for deep neural network, which decreases the number of nodes in each layer exponentially. Our approach is an extension of existing DNN structure but aims at speeding up the training time of the model. The experiment results shown that in terms of house price prediction, our model's fitting result is similar to the currently popularly used one but with faster training time. We hope this proposal can motivate more deep learning users to implement in more application fields.

References

1. Advantages and Disadvantages of Deep Learning. https://www.analyticssteps.com/blogs/advantages-and-disadvantages-deep-learning.
2. Zillow's home value prediction kaggle competition data. https://www.kaggle.com/c/zillow-prize-1/data.
3. K. Bennett and O. Mangasarian. Neural network training via linear programming. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 1990.
4. P. Brahma, D. Wu, and Y. She. Why deep learning works: A manifold disentanglement perspective. *IEEE transactions on neural networks and learning systems*, 27(10):1997–2008, 2015.
5. L. Chen, M. Collins, Y. Zhu, et al. Searching for efficient multi-scale architectures for dense image prediction. *Advances in neural information processing systems*, 31, 2018.
6. T. Dietterich. Overfitting and undercomputing in machine learning. *ACM computing surveys (CSUR)*, 27(3):326–327, 1995.
7. T. Elsken, J. Metzen, and F. Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019.
8. K. Ferentinos. Deep learning models for plant disease detection and diagnosis. *Computers and electronics in agriculture*, 145:311–318, 2018.
9. K. Gai, M. Qiu, and S. Elnagdy. A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance. In *IEEE BigDataSecurity conf.*, 2016.
10. X. Gao and M. Qiu. Energy-based learning for preventing backdoor attack. In *International Conference on Knowledge Science, Engineering and Management*, pages 706–721, 2022.
11. A. Gholamy, V. Kreinovich, and O. Kosheleva. Why 70/30 or 80/20 relation between training and testing sets: a pedagogical explanation. *Technical Report: UTEP-CS-18-09*, 2018.
12. A. Gulli and S. Pal. *Deep learning with Keras*. Packt Publishing Ltd, 2017.
13. S. Hayou, A. Doucet, and J. Rousseau. On the selection of initialization and activation function for deep neural networks. *arXiv preprint arXiv:1805.08266*, 2018.
14. R. Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural networks for perception*, pages 65–93. Elsevier, 1992.
15. E. Hoffer, I. Hubara, and D. Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *Advances in neural information processing systems*, 30, 2017.
16. F. Hu, S. Lakdawala, et al. Low-power, intelligent sensor hardware interface for medical data preprocessing. *IEEE TITB*, 13(4):656–663, 2009.
17. H. Huang, V. Chaturvedi, et al. Throughput maximization for periodic real-time systems under the maximal temperature constraint. *ACM TECS*, 13(2s):1–22, 2014.
18. V. Johnson and L. Rogers. Accuracy of neural network approximators in simulation-optimization. *Journal of Water Resources Planning and Management*, 126(2):48–56, 2000.
19. S. Kamarthi and S. Pittner. Accelerating neural network training using weight extrapolations. *Neural networks*, 12(9):1285–1299, 1999.

20. S. Kung. Xnas: A regressive/progressive nas for deep learning. *ACM Transactions on Sensor Networks (TOSN)*, 2022.
21. J. Li, Z. Ming, et al. Resource allocation robustness in multi-core embedded systems with inaccurate information. *J. of Systems Arch.*, 57(9):840–849, 2011.
22. Y. Li, K. Gai, et al. Intercrossed access controls for secure financial services on multimedia big data in cloud systems. *ACM TMCCA*, 2016.
23. G. Marcus. Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*, 2018.
24. A. Miglani and N. Kumar. Deep learning models for traffic flow prediction in autonomous vehicles: A review, solutions, and challenges. *Vehicular Communications*, 20:100184, 2019.
25. T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Černockỳ. Strategies for training large scale neural network language models. In *2011 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 196–201. IEEE, 2011.
26. J. Niu, Y. Gao, et al. Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability. *JPDC*, 72(12):1565–1575, 2012.
27. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.
28. ARTZAI PICON RUIZ, AITOR ALVAREZ GILA, Unai Irusta, JONE ECHAZARRA HUGUET, et al. Why deep learning performs better than classical machine learning? *Dyna Ingenieria E Industria*, 2020.
29. D. Qi, M. Liu, et al. Exponential synchronization of general discrete-time chaotic neural networks with or without time delays. *IEEE Trans. on Neural networks*, 21(8):1358–1365, 2010.
30. H. Qiu, T. Dong, et al. Adversarial attacks against network intrusion detection in IoT systems. *IEEE Internet of Things J.*, 8(13):10327–10335, 2020.
31. H. Qiu, M. Qiu, M. Liu, and G. Memmi. Secure health data sharing for medical cyber-physical systems for the healthcare 4.0. *IEEE J. of biomedical and health infor.*, 24(9):2499–2505, 2020.
32. H. Qiu, M. Qiu, and R. Lu. Secure V2X communication network based on intelligent PKI and edge computing. *IEEE Network*, 34(2):172–178, 2019.
33. H. Qiu, Y. Zeng, et al. Deepsweep: An evaluation framework for mitigating DNN backdoor attacks using data augmentation. In *ACM AsiaCCS*, 2021.
34. H. Qiu, Q. Zheng, et al. Deep residual learning-based enhanced JPEG compression in the internet of things. *IEEE Trans. on Industrial Infor.*, 17(3):2124–2133, 2020.
35. H. Qiu, Q. Zheng, et al. Topological graph convolutional network-based urban traffic flow and density prediction. *IEEE Trans. on ITS*, 2020.
36. M. Qiu, Z. Chen, et al. Energy-aware data allocation with hybrid memory for mobile cloud systems. *IEEE Syst. J.*, 11(2):813–822, 2014.
37. M. Qiu, K. Gai, and Z. Xiong. Privacy-preserving wireless communications using bipartite matching in social big data. *FGCS*, 87:772–781, 2018.
38. M. Qiu, M. Guo, et al. Loop scheduling and bank type assignment for heterogeneous multi-bank memory. *JPDC*, 69(6):546–558, 2009.
39. M. Qiu, Z. Jia, et al. Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP. *JSPS*, 2007.
40. M. Qiu, E. Khisamutdinov, et al. Rna nanotechnology for computer design and in vivo computation. *Philosophical Transactions of the Royal Society A*, 2013.
41. M. Qiu, H. Li, and E. Sha. Heterogeneous real-time embedded software optimization considering hardware platform. In *ACM SAC*, pages 1637–1641, 2009.

42. M. Qiu, C Xue, et al. Efficient algorithm of energy minimization for heterogeneous wireless sensor network. In *IEEE EUC*, pages 25–34, 2006.
43. M. Qiu, C. Xue, Z. Shao, and E. Sha. Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems. In *IEEE DATE Conf.*, pages 1–6, 2007.
44. M. Qiu, L. Yang, et al. Dynamic and leakage energy minimization with soft real-time loop scheduling and voltage assignment. *IEEE TVLSI*, 18(3):501–504, 2009.
45. M. Qiu, K. Zhang, and M. Huang. Usability in mobile interface browsing. *Web Intelligence and Agent Systems*, 4(1):43–59, 2006.
46. D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, and A. Bolton. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.
47. D. Svozil, V. Kvasnicka, and J. Pospichal. Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems*, 39(1):43–62, 1997.
48. T. Takase, S. Oyama, and M. Kurihara. Effective neural network training with adaptive learning rate based on training loss. *Neural Networks*, 101:68–78, 2018.
49. L. Takeuchi and Y. Lee. Applying deep learning to enhance momentum trading strategies in stocks. In *Technical Report*. Stanford University Stanford, CA, USA, 2013.
50. L. Tao, S. Golikov, K. Gai, and M. Qiu. A reusable software component for integrated syntax and semantic validation for services computing. In *IEEE Sym. on Service-Oriented System Eng.*, pages 127–132, 2015.
51. X. Ying. An overview of overfitting and its solutions. *Journal of physics: Conference series*, 1168(2), 2019.
52. K. Zhang, J. Kong, et al. Multimedia layout adaptation through grammatical specifications. *Multimedia Systems*, 10(3):245–260, 2005.
53. L Zhang, M Qiu, et al. Variable partitioning and scheduling for mpsoc with virtually shared scratch pad memory. *J. of Signal Proc. Sys.*, 58(2):247–265, 2018.
54. B. Zoph, V. Vasudevan, J. Shlens, and Q. Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.

Blockchain Development

Siqi Xie[1,2], Jiahong Cai[1,2,*], Hangyu Zhu[1,2,3], Ce Yang[1,2], Lin Chen[1,2], and Weidong Xiao[4]

[1] School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China
[2] Hunan Key Laboratory for Service computing and Novel Software Technology, Xiangtan 411201, China
[3] Guangdong Financial High-tech Zone "Blockchain +" Fintech Research Institute, Foshan 528253, China
[4] School of Software Engineering, Xiamen University of Technology, Xiamen 361024, China
1224551752@qq.com,jiahongcai@hnust.edu.cn,2770848109@qq.com,yangce@hnust.edu.cn,lynn070021@163.com,xiaoweidong@xmut.edu.cn

Abstract. In recent years, blockchain research has set off an upsurge in academia, and it is called the next generation of value Internet. Because of its decentralization, anonymity, security, immutability, traceability and other characteristics, blockchain is gradually accepted and developed by people. With the deepening of research and the integration of technologies such as deep learning, blockchain has gradually been applied to various fields such as credit reporting, government, medical care, and industrial Internet of Things, not just the initial virtual currency field. This article mainly discusses the three important stages of blockchain public chain development, namely Bitcoin, Ethereum, and metaverse, and introduces some basic supporting technologies of blockchain, as well as the research status and future trends of blockchain. Simple Analysis. By vertically introducing the development history of the blockchain, researchers can have a more concrete understanding of the status quo of the blockchain, and provide ideas for blockchain-related research.

Keywords: Blockchain, Bitcoin, Ethereum, Metaverse, Smart contracts.

1 Introduction

Since the birth of blockchain, most countries had a positive attitude towards its research and industry development, with the United States legislating support, Canada having a positive view, and the Asia-Pacific region watching to explore the development of blockchain technology. In fact, blockchain is a database placed in a non-secure environment, which uses cryptography [1] to ensure that existing data cannot be changed, and consensus algorithms [2] to reach a consensus on new data. Only after Satoshi Nakamoto published "Bitcoin: A Peer-to-Peer Electronic Cash System" [3] blockchain received widespread attention because the core idea of implementing Bitcoin is the blockchain, so Bitcoin is also called the blockchain 1.0 era by scholars. With the continuous research and development of applications, some problems gradu-

ally appeared in the Bitcoin system, and to solve these problems, Ethereum was created, and this means that the blockchain officially entered the 2.0 era. The metaverse as the latest concept [4] has caused another lively discussion in the academic world [5] [6], and blockchain technology is one of the basic technologies to realize the metaverse, and people's vision of blockchain 3.0 is that it can realize assets on the chain and construct various applications based on blockchain as the underlying framework to promote the large-scale creation in the fields of science, health [7], education, and images [8], so the metaverse whether the development of blockchain can bring it into the 3.0 era, everything is possible.

The remainder of this paper is organized as follows. Section II focuses on the birth of the blockchain and the basic components of the Bitcoin blockchain. Section III focuses on the features of the Ethereum blockchain and its improvement on the Bitcoin blockchain. Section IV introduces the metaverse concept and the application of blockchain in the metaverse. Finally, Section V concludes the paper.

2 Blockchain 1.0 - Bitcoin

Scholars in different fields give different definitions of blockchain, which according to the literature [9] is a decentralized technology for transactions and data management. For users of the technology, blockchain represents a great improvement in the field of information collection, distribution, and governance [10]. The literature [11] compares blockchain to a tamper-proof digital ledger, implemented in a distributed manner. The literature [12] considers blockchain as a technology that makes the concept of a shared registry in distributed systems a reality in many application domains.

The concept of blockchain can be traced back to "How to Time-Stamp a Digital Document" written by Stuart Haber and W. Scott Stornetta in 1991 [13]. However, it was not until 2008 that the concept of blockchain appeared in the form of "block" and "chain" in Satoshi Nakamoto's article, attracting the attention of scholars from a new perspective of the application. This article focuses on Bitcoin, a virtual currency that he proposed. Bitcoin is not the earliest attempt at currency in the digital world. Virtual currencies such as Goofycoin and Zainucoin emerged before this, but they all ultimately failed. Until the emergence of Bitcoin, the problem of value and reliability was balanced with simple logic. A synthesis of the failures of previous virtual currencies sums up the following ideas.

1. Original currency transactions are simple and unlimited because they are made directly person-to-person, without the need for a bank-like third party. In fact, virtual currencies can also take advantage of this feature to simplify transactions, so the idea of decentralization emerged, directly turning the currency payment process into a "transaction" to a "transaction" form.
2. Digital products are replicable in nature and if they appear in monetary transactions, they are prone to double-spending problems, so to prevent "double-spending", it is better for everyone to witness it.
3. Since there is no third-party intervention and no concept of a balance wallet, a distributed consensus system is necessary to reach a consensus on the transaction wit-

nessed by all. In order to prevent someone from being evil in the consensus process, a penalty and reward mechanism was introduced to implement a consensus mechanism for many people---Proof of Work.

It was these seemingly simple, yet logical structures that led others to agree that the mechanism by which this currency operated was valid and reliable. Since the basis for Bitcoin's implementation of distributed consensus logic is the blockchain, there are great similarities between the two structures. Zhang et al. [14] summarize the structure of Bitcoin, dividing it into a data layer, a network layer, a consensus layer, a contract layer, and an application layer.

2.1 Data Layer

Block. A blockchain can be viewed as a distributed database that consists of individual blocks linked to each other. There are two main parts of the block, which are the block header and the block body. The block header stores the hash of the previous block, the hash of the content of the block body of the current block (the root of the Merkle tree), and the padding data Nonce. the block body contains the branch nodes of the Merkle tree, i.e., the specific transactions encapsulated in the block. As miners continue to experiment and over time, a continuously growing chain of blocks is created, and the constant updating of the chain represents the updating of the state of the Bitcoin ledger.

Timestamp. Blockchain is a chain structure. The process of chain formation is related to time stamps. The timing of each transaction is in order to prevent some illegal transactions. The timestamp identifies the time of each transaction and forms a chain relationship, its structure in Bitcoin is shown in Fig. 1.

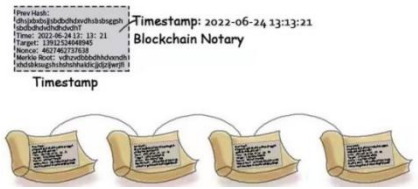


Fig. 1. Blockchain timestamp structure.

Hash Algorithm. The hashing algorithm used in the blockchain is SHA256. there is a probability of hash collision in the hashing algorithm, i.e., it is possible that there are two different numbers that have the same value obtained after calculation by the hash function [15]. However, since the output space of SHA256 is 2^{256}, with the existing computer computing power to forge a number and artificially create a hash collision unlikely, SHA256 can be used to verify whether a block in the blockchain has been tampered with. Hashing the transaction information in the block can form a summary of the block information, and when you need to verify the authenticity of the data, you

can hash the transaction information again to determine whether there is a problem with the data.

2.2 Consensus layer

Proof-of-Work. Bitcoin uses a Proof-of-Work (PoW) mechanism. Proof-of-Work is the mining process, which is simply the search for a number (Nonce) that will satisfy the target value by increasing the number of random bits in the block so that the zero bits when SHA256 is performed. The amount of work required is exponentially related to the number of required zero bits. The process of searching for Nonce is a continuous trial-and-error process, which requires CPU and power consumption. Finding a Nonce that meets the requirements represents the creation of a valid block. The target value is typically reset every 2016 blocks to ensure a frequency of one block generated every ten minutes.

Consensus mechanism. In the traditional BFT consensus algorithm, the identity of each node must be known, but the nature of blockchain is fully public, everyone can participate in the system and can't reveal their identity, so BFT is no longer applicable. Blockchain is a timestamped server on a peer-to-peer basis, and proof-of-work solves the problem of identifying representatives in consensus decisions, in addition to preventing witch attacks and malicious chain generation. Since proof of work is essentially CPU arithmetic, a CPU represents one vote in a consensus decision. Most decisions are represented by the longest chain since a longer chain represents more workload proofs invested. As long as most of the CPU arithmetic is controlled by honest nodes, then honest chains will grow faster and outpace other competing chains.

2.3 Network layer

Broadcast Mechanism. To give a macro summary of the Bitcoin transaction process. When a transaction is initiated, the initiator broadcasts it to all nodes, which receive the transaction in a block and perform a proof of work, and do not stop until one node finds the proof and broadcasts the block it is into all nodes. When the transactions in the block are verified by the nodes as all valid and unspent, the node accepts the block, and creates the next chain of acceptances to the block and uses the hash of the accepted block as the previous hash stored in the block header.

Validation Mechanism. In Bitcoin, since there is no concept of a wallet, this type of cryptocurrency ledger can also be seen as a state transition system, and the "state" refers to the current unused currency (UTXO), each UTXO contains a denomination and an owner, and the UTXO is updated after each transaction. UTXO is also one of the steps to verify the validity of the transaction. If two nodes include a block at the same time in this process, the remaining nodes must make a choice and continue mining. If the chain where the block is located is no longer the longest, the mined block will be invalidated.

2.4 Contract layer

Incentives. All nodes are profit-driven, and the honesty of most nodes is critical to blockchain growth, so there must be incentives to ensure that nodes remain honest and gain more than they would from a malicious attack. Node mining consumes CPU time and power, while the revenue is mainly through transaction fees. If the input value of a transaction is less than the output value, the difference is the transaction fee, which is included in the block containing the transaction. Also, every miner who successfully generates a block has the ability to include in the block a fee issued to itself, worth 12.5 BTC. if a malicious node, wants to get a fee by forging a transaction, then it will re-mine more blocks to make it more than the longest chain, thus making the forged transaction legitimate, which will undoubtedly consume more resources. Therefore, most nodes will still choose to be honest in order to gain revenue. The final summary of the bitcoin transactions and the blockchain generation process is shown in Fig. 2.

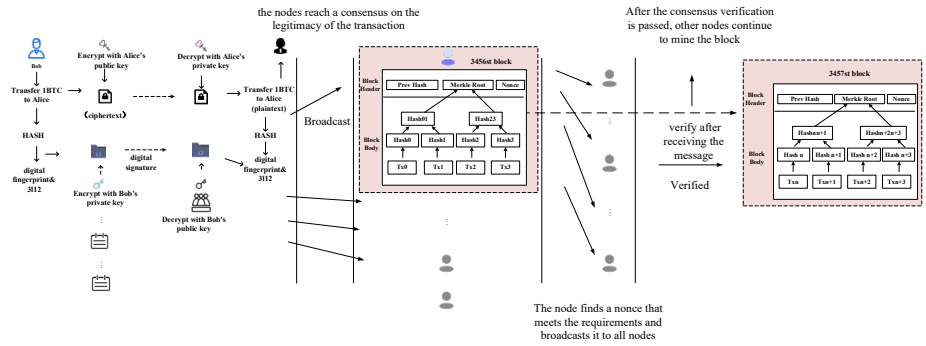


Fig. 2. Blockchain generation process.

3 Blockchain 2.0 - Ethereum

3.1 Ethereum's improvements to Bitcoin's logic

Bitcoin, as a virtual currency, is based on the core idea of a numerical exchange system. And while decentralization, consensus, hashing, and proof-of-work are all joint technologies adopted to allow this currency to be recognized and circulated. After the emergence of Bitcoin's decentralized consensus system, many currencies developed using this model have emerged, such as domain coins, colored coins, and meta-coins. However, Butlerin, the founder of Ethereum, argues that there are some problems with the scripts that implement this process of UTXO for verifying transactions in Bitcoin [16]. For example, lack of Turing completeness, lack of state, etc., so he proposed the Ethereum platform to address these problems and make some improvements.

Account. The "state" in Bitcoin consists of UTXO, while the "state" in Ethereum consists of accounts. It is divided into a contract account and an external account. Both of them contain transaction counters, Ethereum balances, storage for the account, and contract accounts additionally contain a contract code for the account. The external account has no code, the account holder can send messages to the contract account by creating and signing transactions, and when the contract account receives the message, its code will be activated to perform the corresponding operation.

Trading. Since Ethereum is based on an account model, if a cryptocurrency transaction is initiated, then only the sufficiency of the balance on the Ethereum blockchain account needs to be verified, and the source of the Ethereum on the Ethereum blockchain does not need to be stated. This is also an improvement to the transaction-based model in Bitcoin. It is suitable for cases where transactions are large and complex. A transaction in Ethereum needs to contain the message recipient, a signature identifying the sender, the amount of Ethereum transferred, an optional data field, the maximum number of steps allowed for the transaction to execute, and a fee paid by the sender for each computational step. All but the last two concepts are fields that must be included in a cryptocurrency. The last two concepts are used to prevent certain nodes from maliciously provoking circular transactions or other arithmetic waste, and using them, a limit can be placed on the number of steps required for each transaction. Table 1 compares Bitcoin transactions with Ethereum transactions.

Table 1. Differences between Bitcoin and Ethereum.

Blockchain 1.0 Bitcoin (P2P)	Blockchain 2.0 Ethereum (end-to-end)
Decentralized Currency	Decentralized Contract Support
Based on the transaction model No concept of balance	Based on the account model Concept of balance
Go and check if the currently used bitcoin has already been used.	Natural protection against a double-spend attack.

3.2 Other Improvements in Ethereum

Ethereum not only improves on the Bitcoin blockchain but also establishes a protocol that can create decentralized applications that can be effectively used for small and micro applications and improve the interaction between programs. The proposal of smart contracts has brought the application of blockchain technology to a new level. Based on it, Ethereum is more like a development platform where users can write different applications through different smart contracts and allows everyone to write smart contracts through built-in Turing completeness, so Ethereum is a black box close to a Turing machine.

Smart Contracts. Smart contracts appeared before the birth of Bitcoin and almost simultaneously with the birth of the Internet. It was conceptualized by SZABO in 1995 [9], published on the website of the Extropy Institute. But the Internet environment at the time was not suitable for the development of smart contracts, they were never well used. Until the successful emergence of Bitcoin and the continuous development of blockchain technology provided good underlying support for the development of smart contracts. Smart contracts applied on the blockchain are defined: smart contracts are event-driven, stateful programs that run on a replicable and shared ledger and are capable of holding the capital on the ledger [17]. In blockchain systems, smart contracts can run in three environments, namely embedded, virtual machine-based, and container-based [18].

Ethereum smart contracts are executed inside the Ethereum Virtual Machine EVM. The creation and use of a contract can be seen as a transaction process. Among them, creating a contract can be seen as a special kind of transaction process, where the creation function implements the creation of a new contract using a set of fixed parameters that produce a new set of states. The process is as follows.

$$(\sigma^{'},\mathrm{g}^{'},A) \equiv \Lambda(\sigma,s,o,g,p,v,i,e) \tag{1}$$

Where $\sigma^{'}$ is the latest status, $g^{'}$ is the available gas value, A is the sub-state , σ is the system status, s is the transaction sender, o is the source account body, g is the available gas value, p is the gas price, v is the account balance, i is the initialization EVM code, e is the depth of the created contract stack.

Message. Another special feature of Ethereum is messages, which are executed in a similar but different way to "transactions". A transaction is initiated by a message sent from an external account and is real-name. A message is initiated by a contract to other contracts and is anonymous. It contains the sender of the message, the receiver of the message, the amount of Ethereum to be transferred with the message, an optional data field, and the maximum number of computation steps allowed for the message. The emergence of blockchain 2.0 - Ethereum has expanded the application of blockchain technology from cryptocurrency transactions to savings wallets [19], crop insurance [20], intelligent multi-signature escrow [21], cloud computing [22], and many more areas [23][24].

4 Blockchain 3.0 - Metaverse

4.1 Introduction to the concept of metaverse

The metaverse as the latest buzzword has attracted a wide range of attention from industry and academia. The metaverse seamlessly merges the real and virtual worlds while allowing computers to perform many complex activities, including creation, presentation, entertainment, social networking, and trade, thus promising an exciting digital world. In exploring the metaverse, a better real world can also be transformed

[25]. The concept of metaverse was first mentioned in a science fiction novel called Snow Crash [26], and the development of blockchain has made it possible to realize this world that exists only in science fiction. Some technology giants, such as Facebook [27], Epic [28], and Jingteng Tech, are working on the integration of the metaverse into their lives.

The image of users in the metaverse is a projection of humans in the real world, and as the metaverse develops, the image, creation, and consumption of humans in it will refine and influence the real world. In an idealized metaverse, transactions between virtual goods, such as clothes, cars, and real estate, can be realized, as well as the exchange of virtual goods for real substances, in either form, it will affect the regular economy in the real world. The economic system in the metaverse can be divided into four parts: digital creation, digital assets, digital market, and digital currency. The article [25] summarizes them and compares their differences with the conventional economic system, as shown in Fig. 3.

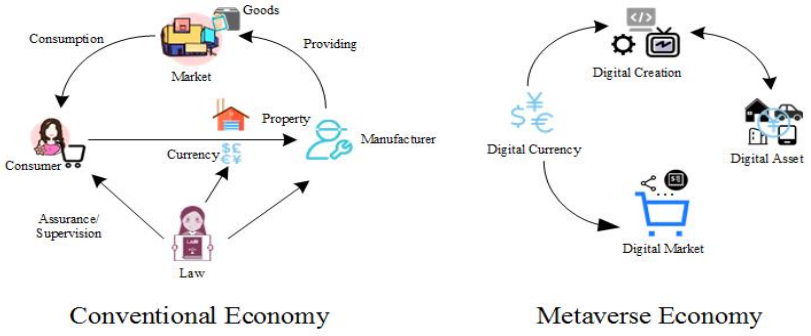


Fig. 3. Comparison of Traditional Economy and Metaverse Economy.

4.2 The role of blockchain in the metaverse

If Ethereum is a black box close to a Turing machine, then by the 3.0 era blockchain will evolve into a Turing machine. Blockchain does three main things in this Turing machine. One is to ensure that most nodes are good [29]. Two is to maintain data security from being tampered with [30-32]. The third is to ensure that the transactions work correctly. The metaverse is a virtual world with an operation mechanism similar to the real world [33], a complete and self-consistent economic system, a complete industrial chain for producing and consuming digital products, and various transactions linking this virtual world together. The metaverse is positioned to be fair, notarized, and self-organizing, then the centralized economic system in the real world cannot function well in the metaverse due to the high transaction volume involved and the complexity of transactions. Thanks to the aid of big data techniques [34-36], the emergence of blockchain has broken the transaction barriers between regions in the circulation of money and opened the barriers in production, life, learning, and work so that the metaverse economic system is decentralized and the transactions using virtual images and virtual assets in the metaverse are legal and effective.

5 Conclusion

The development of blockchain technology from 1.0 to 3.0 is both the continuous improvement of technology and the continuous expansion of application scenarios, but there are also some problems during the development process, such as how to effectively communicate between different chains, the emergence of mining machines leading to the concentration of arithmetic power, how to establish a perfect cross-chain protocol when one chain represents one currency, and how to use blockchain to handle massive transactions. For any technology, it will go through the process of gradually increasing the heat, reaching the peak, and then gradually decreasing and finally leveling off. At present, the research on selfish mining, fragmentation technology, and micropayment channel in blockchain had gradually matured, while the research on blockchain economy, decentralized finance, the integration of blockchain and 5G/6G, blockchain and edge computing were gradually rising in popularity. This also shows that blockchain technology is and will be changing various fields such as finance and communication. In addition to research on blockchain application areas, many scholars are also exploring the improvement aspects of blockchain based technologies, such as side-chain, lightning network, cross-chain, etc. are still in the popular stage of research.

References

1. Huang Y, Liang W, Long J, et al. A Novel Identity Authentication for FPGA Based IP Designs, 17th IEEE TrustCom/BigDataSE, 2018: 1531-1536.
2. Bach L M, Mihaljevic B, Zagar M. Comparative analysis of blockchain consensus algorithms, 41st IEEE MIPRO, 2018: 1545-1550.
3. Nakamoto S. Bitcoin: A peer-to-peer electronic cash system[J]. Decentralized Business Review, 2008: 21260.
4. Wang Y, Su Z, Zhang N, et al. A survey on metaverse: Fundamentals, security, and privacy[J]. IEEE Communications Surveys & Tutorials, 2022.
5. Liang W, Tang M, et al. SIRSE: a secure identity recognition scheme based on electroencephalogram data with multi-factor feature. Com. & Electrical Eng., 2018, 65: 310-321.
6. Xu Z, Liang W, Li K C, et al. A Time-sensitive Token-Based Anonymous Authentication and Dynamic Group Key Agreement Scheme for Industry 5.0[J]. IEEE TII, 2021.
7. Li Y, Liang W, Peng L, et al. Predicting Drug-Target Interactions via Dual-Stream Graph Neural Network[J]. IEEE/ACM Trans. on Comp. Biology and Bioinformatics, 2022.
8. Liang W, Li Y, Xie K, et al. Spatial-temporal aware inductive graph neural network for C-ITS data recovery[J]. IEEE Transactions on Intelligent Transportation Systems, 2022.
9. Szabo N. Smart contracts: building blocks for digital markets[J]. EXTROPY: The Journal of Transhumanist Thought, (16), 1996, 18(2): 28.
10. Bambara J J, Allen P R. Blockchain[J]. A practical guide to developing business, law and technology solutions. New York City: McGraw-Hill Professional, 2018.
11. Yaga D, Mell P, Roby N, et al. Blockchain technology overview[J]. arXiv preprint arXiv:1906.11078, 2019.
12. Belotti M, Božić N, Pujolle G, et al. A vademecum on blockchain technologies: When, which, and how[J]. IEEE Communications Surveys & Tutorials, 2019, 21(4): 3796-3838.

13. Haber S, Stornetta W S. How to time-stamp a digital document[C]//Conference on the Theory and Application of Cryptography. Springer, Berlin, Heidelberg, 1990: 437-455.
14. Zhang Feng, Shi Boxuan, Jiang Wenbao. A review of key technologies and applications of blockchain [J]. Journal of Network and Information Security, 2018, 4(4): 22-29.
15. Rachmawati D, Tarigan J T, Ginting A B C. A comparative study of Message Digest 5 (MD5) and SHA256 algorithm[C]//Journal of Physics: Conference Series. IOP Publishing, 2018, 978(1): 012116.
16. Buterin V. A next-generation smart contract and decentralized application platform[J]. white paper, 2014, 3(37): 2-1.
17. Osterland T, Rose T. Model checking smart contracts for ethereum[J]. Pervasive and Mobile Computing, 2020, 63: 101129.
18. Fan Jili, Li Xiaohua, Nie Tiezheng, et al. Overview of Smart Contract Technology in Blockchain System [J]. Computer Science, 2019, 46(11): 1-10.
19. Praitheeshan P, Pan L, Doss R. Security evaluation of smart contract-based on-chain ethereum wallets, Int'l Conf. on Netw. and System Secu.. Springer, Cham, 2020: 22-41.
20. Jha N, Prashar D, Khalaf O I, et al. Blockchain based crop insurance: a decentralized insurance system for modernization of Indian farmers[J]. Sustainability, 2021, 13(16): 8921.
21. Yang X, Liu M, Au M H, et al. Efficient Verifiably Encrypted ECDSA-Like Signatures and Their Applications[J]. IEEE TDSC, 2022, 17: 1573-1582.
22. Gai K, Guo J, Zhu L, et al. Blockchain meets cloud computing: a survey[J]. IEEE Communications Surveys & Tutorials, 2020, 22(3): 2009-2030.
23. Liang W, Yang Y, Yang C, et al. PDPChain: A consortium blockchain-based privacy protection scheme for personal data[J]. IEEE Transactions on Reliability, 2022.
24. Liang W, Xiao L, Zhang K, et al. Data fusion approach for collaborative anomaly intrusion detection in blockchain-based systems[J]. IEEE Internet of Things Journal, 2021.
25. Yang Q, Zhao Y, Huang H, et al. Fusing blockchain and AI with metaverse: A survey[J]. IEEE Open Journal of the Computer Society, 2022, 3: 122-136.
26. Joshua J. Information Bodies: Computational Anxiety in Neal Stephenson's Snow Crash[J]. Interdisciplinary Literary Studies, 2017, 19(1): 17-47.
27. Meta, Introducing. "A Social Technology Company." Meta 12.11 (2021): 2021.
28. Games E. Fortnite[J]. Epic Games, 2017.
29. Liang W, Tang M, Long J, et al. A secure fabric blockchain-based data transmission technique for industrial Internet-of-Things[J]. IEEE TII, 2019, 15(6): 3582-3592.
30. Y. Li, K. Gai, et al., "Intercrossed access controls for secure financial services on multimedia big data in cloud systems", ACM Trans. on Mult. Comp., Comm., and App., 2016
31. K. Gai, M. Qiu, S. Elnagdy, "A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance," IEEE BigDataSecurity 2016
32. H. Qiu, T. Dong, T. Zhang, J. Lu, G. Memmi, M. Qiu, "Adversarial attacks against network intrusion detection in IoT systems," IEEE IoT J., 8(13), 10327-10335, 2020
33. Kumar P, Kumar R, et al. PPSF: a privacy-preserving and secure framework using blockchain-based machine-learning for IoT-driven smart cities[J]. IEEE Transactions on Network Science and Engineering, 2021, 8(3): 2326-2341.
34. F. Hu, S. Lakdawala, et al., Low-power, intelligent sensor hardware interface for medical data preprocessing, IEEE Trans. on Info. Tech. in Biomedicine 13 (4), 656-663, 2009
35. J. Niu, Y. Gao, et al., "Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability", JPDC, 72(12), 1565-1575, 2012
36. M. Qiu, C Xue, Z Shao, et al., "Efficient algorithm of energy minimization for heterogeneous wireless sensor network", IEEE EUC, 25-34, 2006

Research on Blockchain-based Food Safety Traceability Technology

ZHANG Qinying[1,⋆] and WANG Hongze[2]

Department of Information Engineering, Wuhan Institute of City, Wuhan, China
446618463@qq.com
Department of Information Engineering, Wuhan Institute of City, Wuhan, China
1540509443@qq.com

Abstract. Food traceability can be used to quickly pinpoint problematic links and minimise the risks associated with food safety incidents by viewing the trajectory of food circulation after a food safety incident has occurred. Blockchain, as an emerging technology with characteristics such as decentralisation, asymmetric encryption, tamper-proof and traceability, can be used to generate link traceability technology, which provides great convenience for us to effectively regulate food safety. In response to the need for multiple parties to share information in the traditional food safety traceability system there are many problems such as non-uniformity of the traceability chain, user cultivation and high traceability costs. This paper introduces Blockchain technology to improve the food safety traceability system, using the decentralised and fully distributed DNS service provided by Blockchain to achieve domain name query and resolution through peer-to-peer data transfer services between various nodes in the network, which can be used to ensure that the operating system and firmware of the infrastructure of the food production process have not been tampered with, introducing QR code technology, RFID, ZigBee.Web Web server and other key IoT technologies to achieve information identification and coding for identifying production objects, and identify the bottlenecks that limit the performance of the system by analysing and studying the consensus mechanism of the super ledger Fabric, and use smart contracts and consensus mechanisms as support to build food safety data assurance and food supervision methods to improve overall traceability efficiency.

Keywords: Blockchain, Food safety, Traceability technology, Information dentification, Consensus mechanism

1 Introduction

As a country with a large population, Food Safety is a matter of great importance to people's livelihood. The recurring Food Safety problems in recent years warrant our consideration, and how to make consumers eat with greater peace of mind is an issue that we urgently need to address. In fact, any issue concerning Food Safety can be solved by discussing Traceability systems. With globalisation, the food supply chain is becoming more and more complex. Food Safety

Traceability systems mainly address safety issues in the information field, reducing the risk of hazards occurring and giving consumers a real and tangible sense of security. Blockchain technology is a new technology in the Internet era, which has features such as decentralisation and tamper-proof, and the whole process can achieve anonymous machine autonomy and procedural auditing, which can be applied to Food Safety Traceability systems to address the trust crisis in a targeted manner and alleviate resource wastage.

Based on the idea of consensus mechanism optimization, this paper analyzes and studies the existing consensus framework and transaction consensus process of Blockchain Fabric, and makes Fabric support Byzantine fault tolerance by adopting the parallel plus cache scheme strategy for sorting services, so as to realize the optimization of Blockchain Fabric consensus mechanism. The experimental data in this paper shows that the optimized Fabric system makes the system Byzantine fault-tolerant while the system processing efficiency is also significantly improved.

The main points of this paper are as follows.

- Starting from the traditional traceability system itself, blockchain technology is introduced for the traceability of the information base and the decentralisation of the information processing.
- Use RFID, QR code, Zigbee and Web server technologies in different fields to realize the information identification function of food products.
- Combining the idea of consensus algorithm optimization with the practical application of the system, the method of this paper is tested and analyzed in the Hyperledger Caliper program, and the experimental results show that the method of this paper makes the Fabric system fault-tolerant and greatly improves the system processing efficiency and increases the system throughput to a certain extent.

2 Background

The continuous high rate of economic and social development in China is accompanied by a continuous improvement in people's living standards. The constant Food Safety accidents and other incidents in the market have caused a great impact on government departments, the stability of enterprises and the safety of people's lives. By describing the current situation in the field of Food Safety and analysing the shortcomings of existing Food Safety Traceability systems, this paper attempts to use Blockchain technology as a core technology and carry out the design of Food Safety Traceability methods by utilising the decentralised advantages of this technology.

Blockchain technology is essentially a decentralised distributed ledger concave, which uses cryptography to make the data on the ledger tamper-proof and uses Consensus mechanisms to ensure data consistency between distributed nodes. The decentralised storage mechanism, data immutability and data Traceability are well suited to the food Traceability application scenario.

3 Related work on Food Safety Analysis

In practice, the current food traceability system adopts a segmented supervision system, and the traceability system is obviously "fragmented", with a wide range of participation and huge coverage[11]. The existing food quality traceability system itself has the following main problems. Poor data security, the central database can be tampered with, it is difficult to identify the responsible parties for food quality problems[17], the cost of traceability is high, consumer participation is low, the information acquisition and sharing capability of the traceability system is insufficient, and the management of the quality and safety traceability system is difficult and inefficient[6].

4 The Framework of Food Safety Traceability System

Blockchain technology has been introduced to ensure the traceability of food safety production information by decentralising the information through the traceability of the food information database, thus ensuring the traceability of all aspects of production[23]. In this process, food safety production information is entered into a database that integrates blockchain technology, which actively scans and decomposes the data after entry, dividing a complete base data into data consisting of "data headers + data blocks". This process is illustrated in Fig. 1.

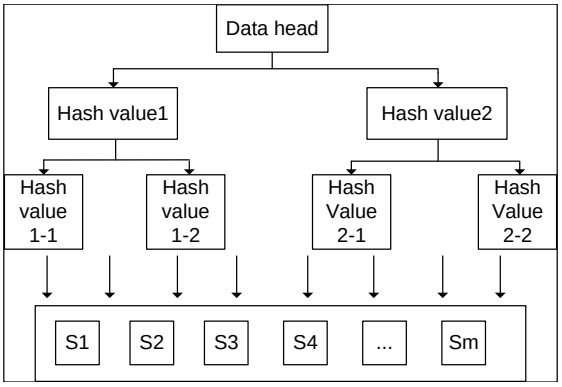


Fig. 1. Blockchain technology-based decomposition of entry data processing diagram

5 Related work of Food Safety Traceability System

5.1 RFID technology

The use of RFID radio frequency identification technology can be equipped with electronic tags from the raw material production, processing and purchase of food

to the product sales stage, while using the global unified article coding technology to determine a unique ID number for each food product, and then using radio frequency identification technology to read or write the food process information, while using wireless transmission technology to release all the information in the food chain to the network management platform in a timely manner[20].

5.2 Two-dimensional code technology

The introduction of two-dimensional code technology allows for the marking of food safety production information[22].

PDF417 barcode This code has a high error correction capability, with each line of the code involving both basic information about the line and also recording some information reflecting the characters used for error correction at the location[25].

Maxi Code The main purpose is to enable the tracking and searching of packages[18].

5.3 ZigBee technology

The intelligent sensors such as temperature, humidity, acidity and alkalinity distributed in the processing environment are usually composed of three parts: acquisition circuit, control transmission circuit with ZigBee chip CC2530 as the core, and antenna, etc[2]. The CC2530 can transmit the collected data to the aggregation node, also called the gateway, through the ZigBee network, which is the management platform mentioned earlier. Finally, the client program of the management platform is saved to the background database[9].

5.4 Web server technology

Each food processing enterprise has a unique fixed network IP address, through the network address translation can make Internet users and other ordinary users through the public network into the local area network of the food processing enterprise[3]. The data centre uses a browser/server (B/S) working mode, which allows users to view data in real time through a browser without the need to install any client software[19].

6 Related work to research on Consensus Algorithms for Hyperledger

6.1 Overview of the consensus algorithm

Consensus ensures that the nodes in a distributed system can reach a final agreement even if a part of the nodes fails arbitrarily. In order to address the problem

of non-determinism in blockchain transactions where there is a certain probability that the consensus outcome will be different, many blockchain platforms address the problem of non-determinism in blockchain transactions by writing smart contracts in specific languages[1].

6.2 Blockchain Fabric Consensus Mechanism

In order to improve the operational efficiency of the system for large-scale application scenarios, Fabric adopts the transaction model of executing first and then sequencing and verifying last (as shown in Fig. 2[21]). Compared with sequential execution, executing transactions first avoids the problems of low performance, poor scalability and flexibility. The transaction steps: initially verifying the transaction and endorsing it[14], the consensus sorting service sorts the endorsed transactions and generates blocks, verifies the endorsement strategy of the transaction and submits it to the ledger[13].

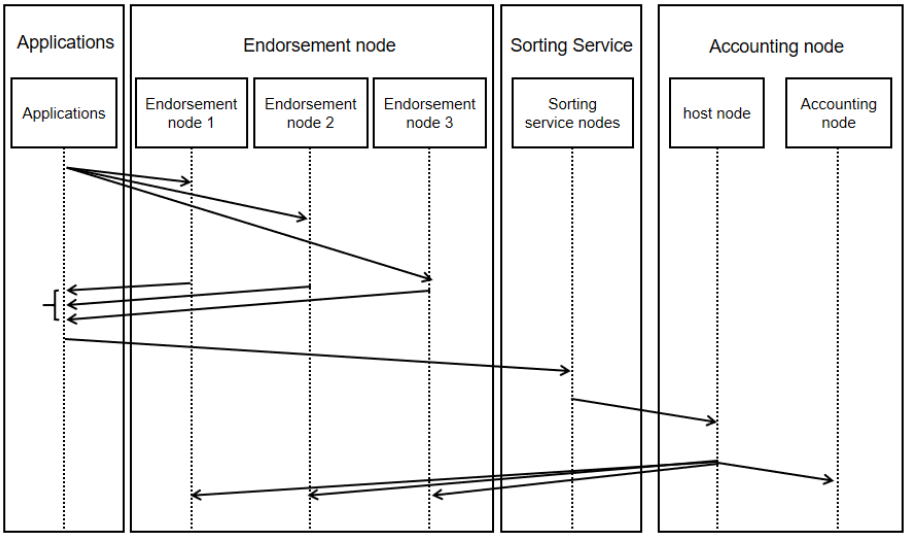


Fig. 2. Hyperledger transaction flow

6.3 Sorting service

The following improvements are made to the sorting service.

Divide the sorting nodes into groups, taking into account the complexity of PBFT consensus and the fact that PBFT consensus efficiency decreases rapidly as the number of consensus nodes increases[7]. Continue to combine the ideas of POS and VRF. The nodes involved in the sorting service are divided into

multiple parallel groups or organizations by means of a random lottery, but in order to increase the execution efficiency and throughput of the system[10]. A Java-based implementation of PBFT, such as Fig.3, shows the PBFT node state code and tests its performance under different node states[15].

```
public enum VoteEnum {
  PREPREPARE("节点生成区块", 10),
  PREPARE("节点收到区块，进入准备状态，并对外广播", 20),
  COMMIT("节点收到超过2f+1个不同节点的commit消息后，" +
          "进入committed状态，并将其持久化到区块链数据库中", 40);

  // 投票情况描述
  private String msg;
  // 投票情况状态码
   private int code;

   // 根据状态码返回对应的Enum
   public static VoteEnum find(int code) {
     for (VoteEnum ve : VoteEnum.values()) {
       if (ve.code == code) {
         return ve;
       }
     }
     return null;
   }
```

Fig. 3. PBFT node state code

In the case where $2f + 1$ is less than or equal to n, it can be agreed that the algorithm complexity is $O(n^{f+1})$, with a higher exponential level of time complexity. Kasloot and Liskov proposed PBFT to optimize its time complexity from the exponential level to the polynomial level, with $O(n^2)$.

6.4 Experimental test analysis

The Hyperledger Caliper program was chosen as the performance benchmarking framework[12]. Hyperledger Caliper allows users to measure the performance of the blockchain platform through a customised test environment and corresponding parameters for the test tool that needs to measure the performance of the blockchain platform[8]. Caliper's simplified architecture is shown in Fig.4.

6.5 Analysis of experimental results

Optimized throughput of the Fabric system can reach 6000tps with a single block transaction count of 200 at a node thread count of 30, with latency below 500ms.

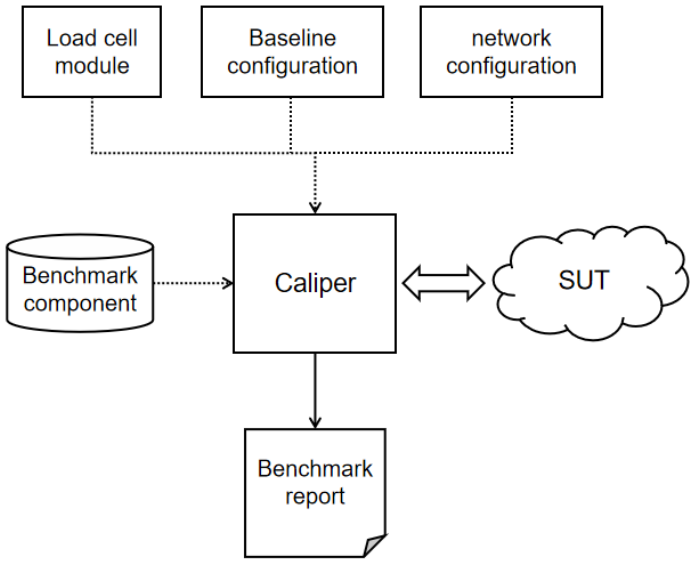


Fig. 4. Sketch of Caliper test architecture

this is a significant improvement in throughput over the Fabric system and reduces transaction latency at high performance[24]. This performance meets the performance requirements of food traceability system scenarios.

Table 1. Comparison of Fabric performance before and after optimization

Arithmetic	Handing capacity/tps	Delay/ms	CPU(avg)/%
Fabric v1.4	2876	680	50.43
Optimized Fabric	5962	496	78.19

7 Blockchain-based Food Traceability System Implementation

7.1 Functional implementation of the system

The implementation of the food traceability system is divided into two parts, namely the web application and the functional implementation of the chain code[18]. The web application is further divided into the back-end business based on the java SDK of the Super Ledger Fabric and the front-end page based on

the front-end architecture[4]. The front-end web page application interacts with the back-end business through a RESTful interface, and the back-end business interacts with the blockchain platform through a gRPC interface, as shown in Fig. 5.

Fig. 5. Relationship between the different parts of the system

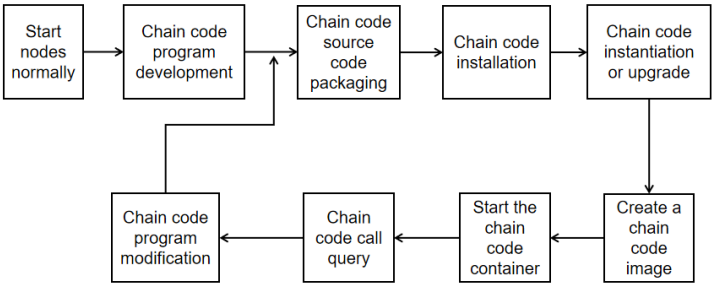


Fig. 6. Flow chart for chain code debugging

7.2 Chain Code Development

Smart contracts, the core of the blockchain, define the executable logic that can be generated to add to the distributed ledger data[16]. Smart contracts act as a set or series of common contracts defined by the system business prior to the transaction, including common terms, data, rules, concept definitions and processes. The Superledger Fabric currently supports chain code development in Go, Java and Node.js development languages[5]. The specific debugging process for chain code is shown in Fig. 6.

A script to quickly start the blockchain network and initialize it is available from the Fabric's project on GitHub. First start the blockchain network and front-end services with the script, then create the channel and install the instantiated chaincode. The instantiation is completed as shown in Fig. 7.

Fig. 7. Instantiation results diagram

Once the chain code has been deployed, the system can be accessed via a browser to verify the relevant functionality.

8 Conclusion

The food safety traceability system has been deeply integrated with blockchain technology. Blockchain ensures the data and transparency of food-related information with its decentralized service, and provides a reliable traceability system for consumers. At the same time, the framework of food traceability system based on blockchain technology as the core and Fabric's optimized consensus algorithm made relative improvements to initially realize a demonstration system of blockchain food traceability.

References

1. Bao, S.: Research on quality and safety assurance system of food supply chain. Ph.D. thesis, Chang'an University (2015)
2. Fei, C., Chunming, Y., Tao, C.: Blockchain-based food traceability system design. Computer Engineering and Applications 57(2), 10 (2021)
3. Furong, L., Sha, C.: Research on building china's agricultural products quality and safety traceability system based on blockchain technology. Rural Finance Research (12), 5 (2016)
4. Gai, K., Wu, Y., Zhu, L., Zhang, Z., Qiu, M.: Differential privacy-based blockchain for industrial internet-of-things. IEEE Transactions on Industrial Informatics 16(6), 4156–4165 (2019)
5. Gajo, M.: Valereum blockchain übernimmt mehrheit an börse von gibraltar. Die Aktiengesellschaft 67(1-2), r8–r8 (2022)
6. He, W., H., Yan: A review of blockchain-based smart contract technologies and applications. Computer Research and Development 55(11), 15 (2018)
7. Leibfried, P., Petry, H.: Blockchain in der finanzberichterstattung. Controlling & Management Review 66(1), 54–59 (2022)
8. LIU Qiqi, X.C.: Construction of agricultural products quality traceability system based on blockchain technology. High Technology Communication 29(3), 9 (2019)
9. M., L.T., X., Y., N., Z.Z., Y., T., X., W., Chaoqun., L.: Research on the application of blockchain+internet of things in agricultural products traceability. Computer Engineering and Applications 57(23), 11 (2021)
10. Müller, M.: Unternehmen halten blockchain für wichtig. Die Aktiengesellschaft 66(17), r253–r254 (2021)
11. Qifeng, S., Chulqing, J., Zao, Z.: Blockchain technology:architecture and progress. Journal of Computer Science 41(5), 20 (2018)

12. Qiu, H., Qiu, M., Memmi, G., Ming, Z., Liu, M.: A dynamic scalable blockchain based communication architecture for iot. In: International Conference on Smart Blockchain. pp. 159–166. Springer (2018)
13. Shengyan, L.: Research on blockchain-based food safety traceability technology. Modern Food 28(17), 3 (2022)
14. Sun, Z., Xu, Q., Baoli, S.: Price and product quality decisions for a two-echelon supply chain in the blockchain. Asia-Pacific Journal of Operational Research 39 (03 2021)
15. Sun, Z., Xu, Q., Shi, B.: Price and product quality decisions for a two-echelon supply chain in the blockchain era. Asia-Pacific Journal of Operational Research 39(01), 2140016 (2022)
16. Tian, Z., Li, M., Qiu, M., Sun, Y., Su, S.: Block-def: A secure digital evidence framework using blockchain. Information Sciences 491, 151–165 (2019)
17. Wang Hongmei, Y.Y.: Research on blockchain-based food safety traceability technology. Electronic Design Engineering 27(13), 6 (2019)
18. Xiong, L.B., Na, Z.X., Xin, X.: Supply chain management in the era of e-commerce. China Management Science (03), 1–7 (2000)
19. Yang, D.R., Wenhui, Z., Snap, W., Long., W.: Design and implementation of hyperledger fabric-based food traceability system. Electronic Technology Applications 47(3), 6 (2021)
20. YG, Y., SHuxin, Z.: A review of blockchain consensus mechanisms. Information Security Research 4(4), 11 (2018)
21. Ying, L.: Research on the mechanism of realizing the key links of sustainable green food supply chain. Ph.D. thesis, Dalian University of Technology (2014)
22. Yong, Y., Xiao-Chun, N.I., Zeng, S., Fei-Yue, W.: The development status and outlook of blockchain consensus algorithm. Zidonghua Xuebao/Acta Automatica Sinica 44(11), 2011–2022 (2018)
23. Youlin, Z.: Exploration of food safety in china. North China Trade and Economics (10), 4 (2009)
24. Zhao, Z., Ma, J.: Application of blockchain in trusted digital vaccination certificates. China CDC Weekly 4(6), 106 (2022)
25. Zhihong, D., Jinwang, L.: Exploring the privacy protection mechanism of super ledger. Science and Technology Innovation (17), 2 (2021)

Few-Shot Learning for Medical Numerical Understanding Based on Machine Reading Comprehension

Xiaodong Zeng[1] and Wenhui Hu [2] and Xueyang Liu [3] and Yuhang Chen [4] and Wenyu Shao [5] and Lizhuang Sun [6]

[123456] Peking University, No.5 YiHeYuan Road, Haidian District, Beijing, China
[1] zengxiaodong@stu.pku.edu.cn, [2] huwenhui@pku.edu.cn
[3] liuxueyang@pku.edu.cn,[4] 2101210553@pku.edu.cn
[5] swy1798@stu.pku.edu.cn, [6] sunlizhuang@stu.pku.edu.cn

Abstract. Numerical understanding relies on some content understanding techniques, which can be based on rules, entity extraction, and machine reading comprehension. Traditional methods often require a large number of regular expressions or a large number of data annotations, and often do not have a deep understanding of numerical values, lacking the ability to distinguish similar numerical values. In this paper, we propose a few-shot learning framework for numerical understanding tasks in Chinese medical texts, and through dynamic negative sampling of the training data, the model's ability to discriminate similar numerical values is enhanced. We use patient text data provided by 13 hospitals in Beijing to conduct experiments. The results show that our newly proposed method is superior to training the baseline pretrained language model directly, the EM increases by 38% and the F1 increases by 27.59%.

Keywords: Numerical Understanding, Few-shot Learning, Negative Sampling.

1 Introduction

With the development of computer [1-3] and network [4-6], hospitals have increasingly collected and stored text data [7-9] related to the patient's condition through the electronic medical record technology [10] to better help analyze and predict the patient's condition and better serve the patient health management work in recent years. Among these text data, there are relatively regular structured texts, such as physical examination reports, inspection reports, etc. However, there are also a large number of unstructured data produced by doctors and nurses through hand-writing and keyboard typing, such as ward round records, patient complaints, admission records, discharge records, etc. These unstructured data often contain very important numerical information and disease entity information. Numerical information is helpful for diagnosis. Considering the diagnosis of diarrhea, 3 times a day and 10 times a day are different.

1.1 Numerical Understanding

The numerical understanding task was first systematically proposed by Corey A Harper et al. [11,12], which mainly proposed five dimensions for numerical understanding, including numerical unit, modifier, measured entity, measured property, and qualifier. The English training data provided by it includes fields of agriculture, biology, chemistry, computer science, earth science, engineering, materials science, mathematics, and medicine, with a total of 448 documents, and the average number of numerical values in per document is about 5. Early numerical understanding systems were often constructed by rule-based methods. By defining a series of complex grammar rules, the numerical value and its corresponding five understanding dimensions were extracted according to lexical analysis tree, syntax analysis tree, and regular expressions [13]. Later, some scholars adopted NER (*Named Entity Recognition*) methods to define numerical entities and related entities through BIO or BIOES tags, which used LSTM, Bi-LSTM, CRF, ELMo, BERT, GPT-3 to train an entity extraction model, then extracted the entities and matched pairs based on a set of predefined distance rules [14,15]. Some scholars used the methods based on machine reading comprehension for numerical understanding [16,17]. The above-mentioned NER method first extracts numerical entities, and then performs machine reading comprehension on the extracted numerical entities to find their corresponding references. The question-answering method has gradually become a mainstream numerical understanding technology. However, these methods either rely on complex rules design or require a relatively large training data set to achieve better results.

1.2 Formula Definition

We analyze Chinese medical texts, especially patient discharge summaries and ward round records. A large number of meaningful values consist of a number and a unit. We focus on finding the reference, which is the combination of measured entity and measured property, as shown in the following table:

Table 1. Examples of Chinese Medical Numerical Understanding.

Quantity	Digit	Unit	Reference
36.5℃	36.5	℃	T
83次/分 (83 times/min)	83	次/分 (times/min)	P
20次/分 (20 times/min)	20	次/分 (times/min)	R
11月 (11 months)	11	月 (months)	反复便血 (Recurrent blood in the stool)
3月 (3 months)	3	月 (months)	乏力 (Weak)
4天 (4 days)	4	天 (days)	腹胀 (Abdominal distension)

The numerical extraction part can complete high-accuracy extraction through a simple rule-based method, such as regular expressions. The output is one of the inputs of our entire system.

In the numerical understanding part, the mainstream and effective method is to ask questions about the extracted numerical values, fill in the numerical values into a slot

to get the question sentence: "{quantity}prompt?", and then ask the model with the context and questions to get the answers as the numerical references.

Specifically, we can define the problem as the following mathematical formula:

Suppose the question is Q:

$$Q = \{q_1, q_2, q_3, q_4 \dots q_m\}$$

Suppose the text is C:

$$C = \{c_1, c_2, c_3, c_4 \dots c_n\}$$

Then the input is I:

$$I = \{[cls], q_1, q_2, \dots q_m, [sep], c_1, c_2, \dots c_n, [sep]\}$$

After going through the deep transformer encoder network, we get the vector of the last hidden layer as H:

$$H = \{h_0, h_1, h_2, \dots h_m, h_{m+1}, h_{m+2}, h_{m+3}, \dots h_{m+n+1}, h_{m+n+2}\}$$

The output of the model is:

$$P_{start(i)} = softmax_i(W_{start} \mid H)$$

$$P_{end(i)} = softmax_i(W_{end} \mid H)$$

Finally, the loss of the model is:

$$loss = -log\, p_{start}(s^*) - log\, p_{end}(e^*)$$

1.3 Overview of Our Work

We propose a few-shot learning framework and a dynamic negative sampling method for numerical understanding tasks to address the above two problems. We assume that there is a special coreference relationship between numerical value and reference, and use the data of antecedent and anapnoric in the ontoNotes_Release_5.0 [18] coreference resolution task to pretrain the current numerical understanding as a large-scale middle task. We suppose the language model to be knowledgeable [19,20,21], let the language model do the fill-mask task to generate the most suitable question for the current scene, and select the most effective question from a large number of generated questions. We use negative sampling to generate (context, quantity', no answer) as one unanswerable triple for each positive (context, quantity, reference) triple. We use a training method similar to SQuAD2.0 [22]. After comparing and assuming the experimental results, we design a dynamic negative sampling method that increases the difficulty of negative samples as the number of the training epoch increases.

In summary, the main contributions of this paper are as follows:

1. A few-shot learning framework for Chinese medical numerical understanding is proposed, and the evaluation indicators are greatly improved.
2. A dynamic negative sampling method is designed to generate unanswerable training samples, thereby enhancing the model's ability to discriminate similar values.

The results show that our newly proposed method is superior to training the baseline pretrained language model directly, the EM increases by 38% and the F1 increases by 27.59%.

2 Related Work

2.1 Regular Expressions

We compare our method with “Microsoft Recognizers Text”. This is a very simple method, through the enumeration of the rules of expression for templated extraction, this method has advantages in time efficiency, but cannot meet the richness of natural language expression in the extraction effect.

2.2 BERT + CRF

Some scholars adopted NER (Named Entity Recognition) methods to define numerical entities and related entities through BIO or BIOES tags, which used LSTM, Bi-LSTM, CRF, ELMo, BERT, GPT-3 to train an entity extraction model, then extracted the entities and matched pairs based on a set of predefined distance rules [14,15].

2.3 QuAnt

Some scholars used the methods based on machine reading comprehension for numerical understanding [16,17]. The above-mentioned NER method first extracts numerical entities, and then performs machine reading comprehension on the extracted numerical entities to find their corresponding references. The question-answering method has gradually become a mainstream numerical understanding technology. However, these methods either rely on complex rules design or require a relatively large training data set to achieve better results.

Due to the lack of numerical understanding datasets in the Chinese medical field, we found that training with only a small amount of annotated data could not achieve the desired accuracy, however, a large number of annotations brought a large cost. On the other hand, when we used mainstream numerical understanding techniques for numerical understanding in the Chinese medical field, the model is not able to discriminate similar numerical values. Consider the quantity "38℃", which is similar to the quantity "37℃" extracted from "the temperature of patient is 37℃", when we ask questions about 38℃, the model still gives "temperature".

3 Our Approach

We present a few-shot learning framework for numerical understanding, and a dynamic negative sampling method for unanswerable question for numerical understanding (FSLUA). We will describe this algorithm in detail in this section.

Our FSLUA designed an intermediate task for numerical understanding to improve with only a small number of downstream samples. We use the knowledge characteristics of the language model to generate questions. We use the idea of negative sampling to generate unanswerable samples, improving the model's ability to understand similar values.

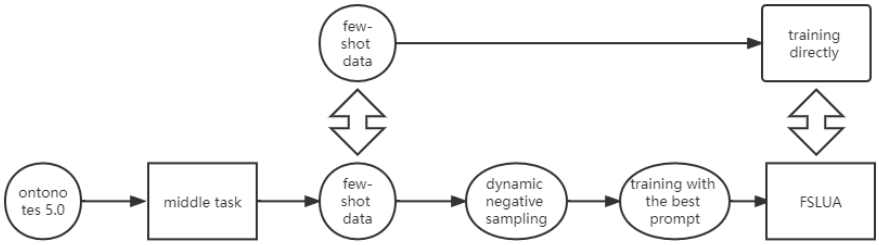


Fig. 1. The framework of our FSLUA.

3.1 Design of Middle Task

Since both a value and its reference describe the same thing, we hypothesize that a value and a reference can be viewed as a special coreference relationship. Specifically, it can correspond to two types of coreference. The QR value is in front of the reference, which can be regarded as the antecedent. The RQ value is after the reference, which can be regarded as the anapnoric.

Table 2. Basic types divided by relative position.

Type	Context	Quantity	Reference
QR	8%的病人不容乐观 (8% of patients are not optimistic)	8%	病人(patients)
RQ	病人的体温为37度 (The patient's temperature is 37 degrees)	37度 (37 degrees)	体温 (temperature)

We use the coreference resolution dataset of ontonotes 5.0 [18] and train it under our existing machine reading comprehension framework. This process can be regarded as a middle task for numerical understanding tasks or large-scale pre-training to improve the model in the current framework.

3.2 Searching the Best Question Template

We use the same BERT [23,24,25] model as the subsequent machine reading comprehension for the fill mask task. Based on some inspirations for using BERT to do question and answer tasks [19,20,21], the language model is knowledgeable after pre-training. Since the language model can give answers based on its own knowledge, we assume that the language model can also use its own knowledge to give a question that is most suitable for the current scene. The sentence we design for the input model is "original text + numerical value + [MASK]*n + reference to numerical value", and the [MASK] part is filled in by the BERT model.

3.3 Negative Sampling and Unanswerable Data

We use the idea of SQuAD2.0 to mix negative samples into the training data for training. We assume that when the model can give no answer to the quantity that does not appear in the original text, it can “identify” the quantity in questions more carefully.

Unlike SQuAD2.0 which uses crowdworkers to generate negative samples, we mainly hope to use some strategies to automatically generate a large number of high-quality negative samples. When constructing negative samples, we mainly consider how to design similar values that do not appear in the context as unanswerable negative samples. Based on numerical analysis, we consider the modification strategy in terms of two independent parts, including digit and unit.

For digit type, we replace the original n bits with the n bits random digits in replace.

Table 3. Numeric Types and Modification Strategies.

Example	Quantity type	Modify	Replace
37	Int	n bits	n bits random digit
15.3	Float	n bits, except "."	n bits random digit
8-10, 159/87	With special token	n bits	n bits random digit

For the unit type, we consider two negative sampling methods. The first one is random sampling. For a certain unit, one negative unit is randomly sampled from all the unit library, except for the original unit. The other is sampling based on edit distance. We calculate the edit distance between a certain unit and all units in the unit library, and sort according to the edit distance, then randomly select a unit in the top k as a negative sample each time.

Table 4. Unit Negative Sampling Strategies and Examples.

Unit negative sampling method	Original	Candidate	Negative
Random sampling	mg	元, L, mol, …… (Yuan-RMB, L, mol)	元 (Yuan-RMB)
Top k sampling based on edit distance	mg	kg, g, m, ml……	kg

4 Experiment

In this section, we evaluate our FSLUA on one dataset collected from 13 hospitals in Beijing and compare it with some previous baselines [13,14,16,17].

4.1 Dataset

We collected data on 200,000 patients from 13 hospitals in Beijing, with a total of 19,730,000 medical records. We randomly selected 100 discharge summary texts for annotation, and selected 145 <context, quantity, reference> as the training set, the other 559 <context, quantity, reference> as the validation set.

4.2 Evaluation Metric

Following previous work, we mainly evaluate the numerical understanding performance on our dataset with EM (exact match) and F1 score. For Chinese, the F1 score is calculated based on the character level split.

4.3 Implementation

Pre-trained language models based on continuous word masking such as SpanBERT, WWM (whole word masking) [24,25] have been shown to have good results on the machine reading comprehension task and other natural language understanding tasks. According to the characteristics of Chinese text, we choose chinese-roberta-wwm-ext-large as our baseline model. For a fair comparison with our model, the baseline model is also used in the NER tasks in the experiments.

We train all the models on 8 NVIDIA SXM4 A100 GPUs with 40GB memory. The max sequence length is 512, and we will do padding to the max length. We set the batch size to 64. We use AdamW optimizer with the learning rate of 3e-5, linear decay with 5% warm up steps.

4.4 Results and Analysis

Table 5 illustrates the evaluation result of our model and the previous baselines.

Table 5. Evaluation result.

Method	EM (%)	F1 (%)
Regular Expressions	14.67	21.88
SciBERT + CRF	36.12	44.26
QuAnt	48.20	61.63
FSLUA(Ours)	86.20	89.22

It can be seen that the accuracy of the rule-based method is not good, largely because we did not manually design all the relevant rules. What's more, complex rules are also difficult to deal with casually styled natural language. SciBERT + CRF reduces the enumeration of a large number of rules, but is not completely rule-free, so the accuracy is also not ideal. The mainstream QA method is better than the ruled-related methods. However, it does not meet business needs due to lacking enough annotation. The EM of our FSLUA increases by 38% and the F1 of our FSLUA increases by 27.59%, compared with QuAnt.

4.5 Ablation Study

In this section, we evaluate the EM and F1 score of our FSLUA, along with analysis on how its output differs from training directly, the effects of different training schemes.

Table 6. Table captions should be placed above the tables.

The amount of data of coreference elimination pre-training	EM (%)	F1 (%)
0	48.20	61.63
1000	52.34	69.34
2000	66.73	76.23
3000	72.41	84.70
4000	72.62	83.22
5000	72.51	83.17
6000	70.12	81.23

We first use different amounts of coreference resolution data for pre-training, then use the training set for finetune, and perform evaluation on the validation set, finally get the following results. After experimental verification, we find that with the increase of coreference resolution data, the final accuracy of the model gradually increases. When the coreference resolution data reaches 3000, the model accuracy basically reaches its peak value. Adding the data continuously will not improve the accuracy of the model, however, there will be a certain decline, as shown in the table above. In this process, the model learns to find the related pairs according to coreference resolution, which helps the model to find numerical values and their corresponding references to a certain extent. However, the two tasks are not exactly the same after all. Excessive data can easily cause the model to over-fit the coreference resolution task and fail to migrate to the numerical understanding task after training with small samples. Therefore, adding an appropriate amount of coreference resolution data as an intermediate task before the training of the numerical understanding task is helpful with only a small number of samples. Compared with training directly, the EM increases by 24.42% and the F1 increases by 23.07%.

We search the question and get 191 templates in total. We have selected some templates with better accuracy and put them in the paper for comparison.

Table 7. Experiment of searching templates.

Templates: {quantity}prompt	EM (%)	F1 (%)
是什么？ (shi shen me)	78.11	84.63
是甚么？ (shi shen me)	85.01	87.84
是什事？ (shi shen shi)	85.01	88.02
都是？ (dou shi)	81.56	87.02
就是？ (jiu shi)	78.11	80.18
是什点？ (shi shen dian)	85.01	85.01
份是？ (fen shi)	85.01	85.91
钟是？ (zhong shi)	81.56	84.72

Note that the meanings of the templates in the table are all “what is it” with different special characters, so we just indicate the Chinese pinyin in the table.

Overall, language expressions that are more human-like will have higher EM and F1 scores, for example, "是什么？(shi shen me)", "是甚么？(shi shen me)". However, we also found that the best template "是什事？(shi shen shi)" is not absolutely coherent in terms of language, which is also contrary to our initial assumption. But we believe that these templates are completed by the model itself, so these prompts are the most suitable ways for the model to do this process, which will be slightly different from the paradigm learned by human. The best EM value is up to 85.01%, and the F1 value is up to 88.02%.

We compared the effects of different negative sampling methods on the experimental results. Training with 1 or 2 or 3 or 4 directly, the negative samples are very similar to the positive quantities from start to end, which will have an undesirable impact on the positive samples. The final result is worse than not using the strategy, because the

model will also give no answer to some positive data. However, training with 5 or 6 or 7 or 8 directly, the negative samples are relatively simple for the model, thus the training for the model is insufficient.

Table 8. Table captions should be placed above the tables.

id	Strategy	EM (%)	F1 (%)
1	Only quantity, n=2	81.51	83.26
2	Only quantity, n=1	79.23	82.13
3	Only unit, top k=20	84.54	87.31
4	Only unit, top k=10	82.31	85,12
5	Both quantity and unit, n=2 top k=20	85.05	88.06
6	Both quantity and unit, n=1 top k=20	85,82	88.93
7	Both quantity and unit, n=2 top k=10	85.45	88.57
8	Both quantity and unit, n=1 top k=10	85.23	88.38
9	Dynamic Negative Sampling	86.2	89.22

After some experiments, we try a training order “5-6-7-8-3-4-1-2”, whose degree of confusing the model increases as the epoch increases. And we call this method dynamic negative sampling. After experimental comparison, it is found that the accuracy is better than directly using 1, 2, 3, 4, 5, 6, 7, and 8. By increasing the noise gradually, the model can learn from simple negative samples to difficult negative samples, which results in a large boosting.

5 Conclusion

We present a few-shot learning framework for numerical understanding, and a negative sampling method for unanswerable question for numerical understanding. The middle task part of few-shot learning framework boosts the EM from 48.20% to 72.62%, F1 from 61.63% to 84.70%, compared with training directly. Then the question searching part of few-shot learning framework boosts the EM to 85.10%, F1 to 88.02%. Although there is not much confusing data for similarity numerical understanding in the validation dataset, the confusing questions get the right answer after our dynamic negative sampling. In order to illustrate our conclusion, we do some test case, and model can distinguish the difference between the similar quantities. Finally, the negative sampling boosts the EM to 86.2%, F1 to 89.22%.

References

1. M. Qiu, H. Li, E. Sha, "Heterogeneous real-time embedded software optimization considering hardware platform", ACM sym. on Applied Comp., 1637-1641, 2009
2. J. Li, Z. Ming, et al., "Resource allocation robustness in multi-core embedded systems with inaccurate information", J. of Systems Arch. 57 (9), 840-849, 2011
3. M. Qiu, Z. Chen, Z. Ming, X. Qin, J. Niu, "Energy-aware data allocation with hybrid memory for mobile cloud systems", IEEE Systems J., 11 (2), 813-822, 2014

4. J. Niu, Y. Gao, et al., "Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability", JPDC, 72(12), 1565-1575, 2012
5. M. Qiu, C Xue, Z Shao, et al., "Efficient algorithm of energy minimization for heterogeneous wireless sensor network", IEEE EUC, 25-34, 2006
6. H. Qiu, T. Dong, et al., "Adversarial attacks against network intrusion detection in IoT systems," IEEE Internet of Things Journal 8(13), 10327-10335, 2020
7. H. Qiu, Q. Zheng, et al., "Topological graph convolutional network-based urban traffic flow and density prediction", IEEE Trans. on ITS, 2020
8. Y. Li, K. Gai, et al., "Intercrossed access controls for secure financial services on multimedia big data in cloud systems", ACM Trans. on Multimedia Comp., Comm., and App., 2016
9. K. Gai, M. Qiu, S. Elnagdy, "A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance," IEEE BigDataSecurity 2016
10. F. Hu, S. Lakdawala, et al., Low-power, intelligent sensor hardware interface for medical data preprocessing, IEEE Trans. on Info. Tech. in Biomed. 13 (4), 656-663, 2009
11. Harper, C., Cox, J., Kohler, C., et al. (2021, August). SemEval-2021 Task 8: MeasEval–Extracting Counts and Measurements and their Related Contexts. 15th Int'l Workshop on Semantic Evaluation (SemEval-2021) (pp. 306-316).
12. Therien, B., Bagherzadeh, P., & Bergler, S. (2021, August). CLaC-BP at SemEval-2021 Task 8: SciBERT Plus Rules for MeasEval. SemEval-2021, pp. 410-415.
13. Foppiano, L., Romary, L. et al, (2019). Automatic identification and normalization of physical measurements in scientific literature. ACM Sym. on Doc. Eng. 2019 pp. 1-4.
14. Gangwar, A., Jain, S., Sourav, S., & Modi, A. (2021). Counts@ iitk at semeval-2021 task 8: Scibert based entity and semantic relation extraction for scientific data. arXiv preprint arXiv:2104.01364.
15. Kohler, C., & Daniel Jr, R. (2021). What's in a Measurement? Using GPT-3 on SemEval 2021 Task 8--MeasEval. arXiv preprint arXiv:2106.14720.
16. Davletov, A., Gordeev, D., Arefyev, N., & Davletov, E. (2021, August). LIORI at SemEval-2021 task 8: Ask transformer for measurements. 15th SemEval-2021, pp. 1249-1254.
17. Avram, A. M., Zaharia, G. E., Cercel, D. C., & Dascalu, M. (2021). Upb at semeval-2021 task 8: Extracting semantic information on measurements as multi-turn question answering. arXiv preprint arXiv:2104.04549.
18. Hovy, E., Marcus, M., Palmer, et al., (2006, June). OntoNotes: the 90% solution. Human language tech. conf. of the NAACL, Companion Volume: Short Papers (pp. 57-60).
19. Jiang, Z., Xu, F. F., Araki, J., & Neubig, G. (2020). How can we know what language models know? Transactions of the Association for Computational Linguistics, 8, 423-438.
20. Roberts, A., Raffel, C., & Shazeer, N. (2020). How much knowledge can you pack into the parameters of a language model?. arXiv preprint arXiv:2002.08910.
21. Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M., & Liu, Q. (2019). ERNIE: Enhanced language representation with informative entities. arXiv preprint arXiv:1905.07129.
22. Rajpurkar, P., Jia, R., & Liang, P. (2018). Know what you don't know: Unanswerable questions for SQuAD. arXiv preprint arXiv:1806.03822.
23. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
24. Joshi, M., Chen, D., Liu, Y., et al. (2020). Spanbert: Improving pre-training by representing and predicting spans. Trans. of the Association for Computational Linguistics, 8, 64-77.
25. Cui, Y., Che, W., Liu, T., Qin, B., & Yang, Z. (2021). Pre-training with whole word masking for Chinese bert. IEEE/ACM Trans. on Audio, Speech, and Lang. Proc., 29, 3504-3514.

A Feature Extraction Algorithm Based on Blockchain Storage that Combines ORB Feature Points and Quadtree Division

Yawei Li[1], Yanli Liu[1,*], Heng Zhang[1], and Neal Xiong[2]

[1] Shanghai Dianji University,P.O. Box 201306, Shanghai, China
216003010223@st.sdju.edu.cn, liuyl@sdju.edu.cn, zhangheng@sdju.edu.cn
[2] Department of Mathematics and Computer Science, Sul Ross State University, Alpine, TX 79830, USA
xiongnaixue@gmail.com

Abstract. In the process of 5G power grid inspection robot moving for a long time, the sensor constantly collects the feature information of the substation. Due to the limited memory capacity, this feature information must be stored on the network, which requires the storage network must have strong security. The storage network based on blockchain can better solve the problem of feature data encryption, and a good feature extraction method can also relieve the pressure of network storage. As the input information of the whole SLAM, feature points play a crucial role in the detection performance and accuracy of the whole SLAM. When the extracted feature points are few or evenly distributed, they cannot express the information of the whole environment, which will make the mapping and localization error of SLAM system larger, and seriously lead to the loss of tracking. In this paper, we first analyze the standard ORB algorithm and the Qtree_ORB algorithm. Aiming at the problems existing in the two algorithms, an improved ORB feature extraction algorithm is developed. For the problem that the feature points extracted by the Qtree_ORB algorithm are too uniform, the maximum division depth of the quadtree is limited according to the number of feature points required for each layer of the image pyramid, which improves the problem that the feature points are too uniform. Finally, we evaluate the performance of the improved algorithm, and analyze the uniformity of feature points to verify the performance and robustness of the improved algorithm.

Keywords: Visual SLAM · Blockchain · ORB Feature Points · Quadtree · Feature extraction.

1 Introduction

Feature point extraction and matching is the most basic front-end step of visual SLAM (*Simultaneous Localization and Mapping*), which directly determines the effect of SLAM localization and mapping [1,2]. Feature points can express the essence of the image and analyze the information in the environment. They play

an important role in the algorithm based on feature matching [3,4]. High-quality feature points will have a better contribution to the system performance and robustness of SLAM, and it is not easy to lose environmental information and tracking loss. In addition, image feature points are also an important representation of scene information. If the extracted image feature points are not uniform, they cannot fully reflect the information in the environment and also affect the system accuracy of SLAM. For mobile robots, robotic SLAM is generally required to have low cost [5], low power consumption [6], and high accuracy [7]. Connecting the robot to the web and cloud would expose the robot SLAM process to possible hacking, which could lead to loss of map data and or tampering with the SLAM process [8,9,10]. Blockchain is an alternative to build a transparent and secure environment for data storage [11,12,13], which can be well protected against data tampering in transmission [14,15,16]. The storage network based on blockchain can better solve the problem of feature data encryption [17,18,19,20], and a good feature extraction method can also relieve the pressure of network storage [21,22].

There are many algorithms for the extraction of environmental feature points, each with its own characteristics. The standard ORB(Oriented FAST and Rotated BRIEF) [23,24] algorithm is widely used because of its high real-time performance [25]. However, the standard ORB feature point extraction algorithm [26] also has some unavoidable problems. For example, most of the feature points will be concentrated in the regions with strong texture information, while the number of feature points extracted in some regions with weak texture information is relatively small [27]. Yang et al. [28] proposed a 3D reconstruction method for weakly textured scenes. They use calibrated cameras to take photos from multiple views, and then use these images to complete high-precision reconstruction through SFM (*Structure from Motion*) and MVS (*Multiple View Stereovision*) algorithms. Zhou et al. [29] add line features on the basis of ORB-SLAM, and construct an improved point-line integrated monocular VSLAM(Visual SLAM) algorithm to solve the problem of missing feature points in regions with weak textures. An algorithm for extracting ORB feature points based on quadtree structure (Qtree_ORB) is proposed in ORB-SLAM2 [30]. The Qtree_ORB algorithm divides the quadtree structure for each layer of the image pyramid, and retains the point with the largest Harris response value in each node. Although the algorithm can evenly extract the feature points in the environment, the feature points are too uniform in this algorithm, and the operation efficiency is slow due to too many quadtree nodes, and the algorithm retains some low-quality features point. Therefore, it is necessary to study a feature point extraction algorithm with feature point homogenization, better real-time performance and higher efficiency.

This paper improves the feature point extraction algorithm in ORB-SLAM based on the above problem, reduces the impact on real-time performance due to too deep division of quadtrees, and limits the division depth of quadtrees to reduce the impact of division depth on feature point extraction when processing feature point homogenization, which effectively solves the above problem. The

main contributions of this paper are as follows: (1) This paper improves the feature point extraction algorithm based on quadtree in ORB-SLAM2 algorithm, limits the division depth of quadtree based on the number of feature points required by each layer of the image pyramid, reduces the number of iterations of quadtree partition and the extraction of low-quality feature points. (2) In this paper, the meshing of each image layer is performed adaptively according to the number of feature points required. The number of divisions is then reduced by limiting the quadtree segmentation depth. The problem of uniform extraction of feature points is improved, and the feature matching time is reduced.

The rest of this paper is organized as follows. The standard ORB algorithm and the Qtree_ORB algorithm are described in Section 2. The improved Qtree_ORB feature point algorithm is described in Section 3. Some experiment results are shown in Section 4, and Section 5 sets out the conclusions.

2 Standard ORB and Qtree_ORB Feature Point Algorithm

2.1 Introduction to Standard ORB Algorithms

In the research process, it has been found that the corner and edge points in the image are more representative of the information in the scene than the grayscale values of the image pixels directly. Researchers define these points as key points, which describe information such as the position, direction and scale of feature points in the scene. But how to describe a feature point is also a problem that needs to be solved, so the concept of the descriptor is proposed to describe that feature point. ORB algorithm combines the two concepts and proposes an ORB feature point extraction algorithm. In the feature matching of the algorithm, descriptors are used to judge whether two feature points are on the same point.

2.2 Standard Qtree_ORB Feature Point Algorithm

In order to extract feature points in the ORB-SLAM2 algorithm, the ORB algorithm is used for extraction, and the quadtree structure is used to scree the extracted feature points. It mainly includes the following processes: (1) The process of feature point extraction at different scales is solved by constructing image pyramid; (2) Mesh the images constructed in the first step at different scales. (3)Limit the extraction of feature points, set a minimum threshold, extract feature points from the grid set in the second step. If feature points are not enough to extract, lower the threshold and continue to extract feature points until enough feature points are extracted. (4) Based on the quadtree, the obtained FAST corner points are selected uniformly.

3 Improved Qtree_ORB Feature Point Algorithm

3.1 Construction of Image Pyramids

By constructing an n-layer image pyramid to increase the scale information of the image, the problem that ORB feature points do not have scale invariance is solved. Marking the initial image as I_0, the latter I_1~I_{n-1} layers can be obtained by down sampling the image from the previous layer. The scaling scale of each layer is:

$$S_i = ScaleFactor^i (i = 0, 1, ..., n-1) \tag{1}$$

Where n is the number of layers of the pyramid (n=8), and $ScaleFactor$ is the scale reversal of each layer of the image pyramid. S_i is the scaling scale of the image pyramid for each layer set. Define $Invs = 1/ScaleFactor$, the width and height of the original image are W and H, and the corresponding relationship between the width and height of images in each layer of the image pyramid is:

$$w_i = w \times InvS^i \tag{2}$$

$$h_i = h \times InvS^i \tag{3}$$

Where W_i is the width of the image at layer i, h_i is the height of the image at layer i, $i = 0, 1, \ldots, 7$.

Through the construction of image pyramids, different scale information is added to the image to satisfy the problem of scaling in the image scene, which solves the problem that ORB feature extraction algorithm does not have scale invariance.

3.2 Division of The Grid

After obtaining the image pyramid of each layer, we divide the image pyramid of each layer into grids. The original ORB-SLAM2 algorithm uses a fixed aspect ratio to mesh the image, but this does not adapt well to the environment in the scene. In this paper, we improve it by using an adaptive form of meshing, which adaptively meshes each layer of the image according to the number of feature points needed, and determines the meshing according to the area.

$$\begin{cases} width = maxBordeX - minBordeX \\ height = maxBordeY - minBordeY \\ W = \sqrt{width \times height \times \epsilon / N} \end{cases} \tag{4}$$

Where $width$ is the width of each pyramid image, $height$ is the height of each pyramid image, ϵ is the scale factor, $maxBordeX$, $minBordeX$, $maxBordeY$, $minBordeY$ is the boundary coordinates of each image layer, and the experimentally verified ϵ value is 1.8. N is the required number of feature points. W is the width and height of the divided grid, which is rounded if it is not an integer.

The actual width and height of the grid are: where $wCell$ is the actual width of the divided grid, $hCell$ is the actual height of the divided grid, and $round$ is the rounding function.

$$\begin{cases} wCell = round(width/W) \\ hCell = round(height/W) \end{cases} \tag{5}$$

The actual divided rows and columns are calculated by the width and height of the image pyramid of each layer. In the case that they are not integers, they are rounded upward, as shown in the following formula:

$$\begin{cases} nCols = width/W \\ nRows = height/W \end{cases} \tag{6}$$

After meshing the images of each layer, feature points are extracted from the images in the mesh, and the feature points of each layer are calculated according to the following criteria:

$$N_i = N \times \frac{1 - InvS^i}{1 - InvS} \tag{7}$$

Define N_i as the number of feature points per layer, and $InvS$ is set as each layer's reciprocal of the image pyramid scale factor.

After the grid division in the feature point extraction process, a minimum threshold is set and the feature points are extracted in the grid. After feature extraction, if enough feature points cannot be extracted from the grid, the threshold is lowered until enough feature points are finally extracted. Finally the non-maximal value suppression algorithm is used to eliminate the overlapping feature points.

3.3 Division of Quadtrees

After the information extraction of the scene by the ORB algorithm extraction process, there are many redundant and duplicate feature points, which are eliminated and selected by the quadtree in this paper. Extracted feature points are divided into four nodes using a quadtree structure, and feature points in the nodes are filtered according to the Harris response values.

The standard quadtree filters feature points by splitting nodes at each level. However, in some scenarios, the division depth of the quadtree is too deep, which has an impact on the efficiency of the algorithm and thus on the real-time performance of SLAM, and extracted feature points are too uniform when the depth of the quadtree is too deep. To address these problems, this paper limits the division depth of quadtrees. Firstly, the maximum partition depth D_{max} of the pyramid is set according to the number of feature points required by each layer of image pyramid. The relationship between the division depths in a quadtree can be expressed as follows:

$$d \leqslant D_{max} \tag{8}$$

Where d is the current division depth of the quadtree, D_{max} is the maximum division depth. The expression of the relationship between the number of nodes per layer and the depth of division is:

$$Nodes_i = 4^d \leqslant 4^{D_{max}} \tag{9}$$

Where $Nodes_i$ is the number of nodes at the current division depth.

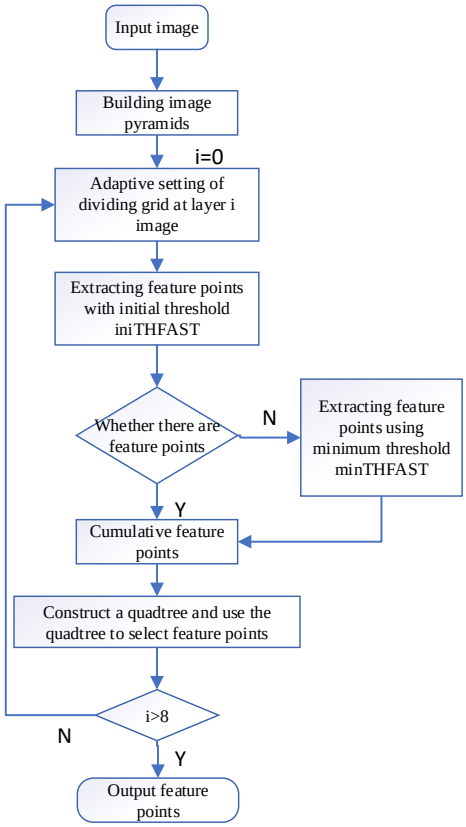


Fig. 1. Flowchart of improved Qtree_ORB feature point algorithm

Figure 1 shows the overall flow chart of the improved algorithm in this section. Firstly, feature points are extracted through adaptive thresholds in each layer of image pyramid, then feature points are screened according to the improved quadtree structure, and finally feature points are retained according to Harris response values [31].

4 Experimental Verification

4.1 Experimental Evaluation Index

The improved feature point extraction algorithm in this paper uses the uniformity of distribution of image feature points for evaluation[32].

$$\sigma = \sqrt{(\frac{(m_1 - \overline{m})^2 + (m_2 - \overline{m})^2 + ... + (m_{10} - \overline{m})^2}{10}} \tag{10}$$

Where σ denotes the uniformity of the feature point distribution, $m_1, m_2, .., m_{10}$ indicates the ten divided regions, $\overline{m}$ is the average of ten areas. The time taken to extract feature points in the image is measured between the improved algorithm and the original algorithm, and the cost time T is compared to reflect the operation efficiency of the algorithm.

4.2 Comparison of Feature Point Uniformity

In this experiment, the images in the Mikolajczyk [33] dataset are used to compare ORB-SLAM2 with the improved feature point extraction algorithm in this paper. Each group of images in the Mikolajczyk dataset contains six images, which are divided into one standard image and five images in different changing environments. In this experiment, eight datasets are selected for testing, where boat is the rotation change dataset, bark is the scale change dataset, leuven is the illumination change dataset, bikes and trees are the blur change dataset, wall and graf are the viewpoint change dataset, and ubc is the image compression change dataset. In this paper, the first two images in the dataset are selected as samples for the experiments of feature point extraction by different algorithms in the above six variations, respectively. The hardware configuration of all experiments in this paper is as follows: GPU NVIDIA 2080Ti, CPU: Inter I5-10300H and Ubuntu18.04 operating system.

All the data in the experiments in this paper have different errors in different hardware devices, so the data measured in the experiments are taken 10 times to avoid random mistakes. After experimentally determining the depth of the quadtree division in the image pyramid the maximum depth of the first to the fourth level is taken as 3, the maximum depth of the fifth to the sixth is 2, and the maximum depth of the seventh to the eighth level is taken as 1. In order to balance the algorithm between the calculation speed and the uniformity of feature points, the minimum threshold of the Harris response value is 15. In this paper, the first benchmark image and the second matching image in 8 sets of datasets are selected. According to the feature point evaluation criteria set in this paper, the uniformity of image feature points is tested in different scenes in the dataset. In order to quantify the distribution of feature points extracted from images by different algorithms, the evaluation criteria in this paper are selected to calculate the uniformity of feature points in images, as shown in Table 1.

Table 1. Comparison of the uniformity of image feature point distribution.

Algorithm	boat	bark	leuven	tress	wall	ubc	graf	bikes	Uniformity
Improved_Qtree_ORB	85.27	17.85	35.62	18.39	52.68	37.04	25.36	38.26	38.81
Qtree_ORB	90.49	54.72	43.65	88.74	97.85	78.91	48.64	25.75	66.09
ORB	179.72	267.74	174.58	179.72	134.72	183.55	186.12	176.23	185.30

In Table 1, the improved Qtree_ORB algorithm extracts feature points with less uniformity compared to the standard ORB and Qtree_ORB. Since the standard deviation of the number of feature points is used as the uniformity standard in the standard in this section, it can be seen that the feature points extracted by the improved Qtree_ORB algorithm in this paper are more uniform and have a great improvement compared to the other two. This is due to the restriction on the division depth of the quadtree in the Improved_Qtree_ORB algorithm so that it extracts essentially the same number of feature points in each quadtree node. The uniformity of Improved_Qtree_ORB is 38.81, which is 79.1% higher than ORB algorithm and 41.2% higher than Qtree_ORB's 66.09.

Table 2 shows the time spent by the three algorithms in extracting feature points as a reflection of the operational efficiency of the three algorithms. In this table, ORB algorithm consumes the shortest time and runs with the highest efficiency. The least efficient operation is the Qtree_ORB algorithm, mainly because its algorithm needs to add a quadtree to manage the feature points when extracting them. Then an image pyramidal grid division is used for each layer to extract feature points, which is not limited by the depth of its division in the quadtree division process, resulting in a longer time to extract feature points. Compared to Qtree_ORB, the average running time of our method tested with multiple datasets is improved by 5.27%. The Improved_Qtree_ORB algorithm running efficiency is 8.93% higher than the original Qtree_ORB algorithm on the bot dataset. Our method saves running time while ensuring good uniformity of feature points. In this paper, the efficiency of the algorithm is improved by limiting the depth of the quadtree division.

Table 2. Feature point extraction time for Mikolajczyk dataset.

Algorithm	bark	wall	graf	bikes	leuven	boat	ubc	trees	Average
ORB	130.27	195.85	152.62	190.39	160.68	164.04	188.52	215.82	174.77
Improved_Qtree_ORB	168.49	278.72	196.65	225.74	214.85	240.91	235.73	316.62	234.71
Qtree_ORB	178.72	284.74	206.58	231.72	236.72	264.55	242.36	336.74	247.77

5 Conclusion

The standard visual SLAM algorithm uses a quadtree form for feature point extraction in the scene. However, when the depth of the quadtree is too deep,

the problem of extracting feature points too uniformly is observed when extracting feature points. In this paper, the algorithm for extracting feature points was improved, and the division principle of each layer of the quadtree was connected with the image pyramid, which solves the problem of low quality and too uniform feature points. It can better extract environmental information, reduce the redundancy of extracting environmental information, and relieve the storage pressure on the blockchain network. Finally, the uniformity of the extracted feature points was experimentally compared to analyze the advantages and disadvantages of the improved algorithm. The experimental results showed that our method can meet the real-time requirements of the system.

Acknowledgements This work was supported in part by the National Natural Science Foundation of China under Grant 61963017; in part by Shanghai Educational Science Research Project, China, under Grant C2022056; in part by the Outstanding Youth Planning Project of Jiangxi Province, China, under Grant 20192BCBL23004.

References

1. Yujiao Jia, Xinying Yan, and Yihan Xu. A survey of simultaneous localization and mapping for robot. In IEEE 4th Advanced Info. Tech., Elec. and Auto. Cont. Conf. (IAEAC), volume 1, pages 857–861, 2019.
2. Jiandong Guo, Rongrong Ni, and Yao Zhao. Deblurslam: A novel visual slam system robust in blurring scene. In IEEE 7th ICVR, pages 62–68, 2021.
3. Yulin Li, Wenfeng Zheng, Xiangjun Liu, Yuanyuan Mou, Lirong Yin, and Bo Yang. Research and improvement of feature detection algorithm based on fast. Rendiconti Lincei. Scienze Fisiche e Naturali, 32(4):775–789, 2021.
4. Richard Szeliski. Feature detection and matching. In Computer Vision, pages 333–399. Springer, 2022.
5. M. Qiu et al. Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems. In IEEE DATE, pages 1–6, 2007.
6. M. Qiu et al. Dynamic and leakage energy minimization with soft real-time loop scheduling and voltage assignment. IEEE TVLSI, 18(3):501–504, 2009.
7. J. Li, Z. Ming, et al. Resource allocation robustness in multi-core embedded systems with inaccurate information. J. of Systems Arch., 57(9):840–849, 2011.
8. Han Qiu, Meikang Qiu, et al. A dynamic scalable blockchain based communication architecture for iot. In SmartBlock, pages 159–166. Springer, 2018.
9. Zhihui Lu, Keke Gai, et al. Machine learning empowered content delivery: Status, challenges, and opportunities. IEEE Netw., 34(6):228–234, 2020.
10. Yue Zhang, Keke Gai, et al. Blockchain-empowered efficient data sharing in internet of things settings. IEEE J. on Selected Areas in Comm., 2022.
11. H. Qiu, M. Qiu, and R. Lu. Secure V2X communication network based on intelligent PKI and edge computing. IEEE Network, 34(2):172–178, 2019.
12. H. Qiu, Y. Zeng, et al. Deepsweep: An evaluation framework for mitigating DNN backdoor attacks using data augmentation. In ACM AsiaCCS, 2021.
13. K. Gai et al. A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance. In IEEE BigDataSecurity, 2016.

14. Keke Gai, Yulu Wu, Liehuang Zhu, Zijian Zhang, and Meikang Qiu. Differential privacy-based blockchain for industrial internet-of-things. IEEE Transactions on Industrial Informatics, 16(6):4156–4165, 2019.
15. Zhihong Tian, Mohan Li, et al. Block-def: A secure digital evidence framework using blockchain. Info. Sciences, 491:151–165, 2019.
16. Yue Zhang, Keke Gai, Yihang Wei, and Liehuang Zhu. Bs-kgs: Blockchain sharding empowered knowledge graph storage. In International Conference on Knowledge Science, Engineering and Management, pages 451–462. Springer, 2021.
17. Prabhat Kumar, Randhir Kumar, et al. Ppsf: A privacy-preserving and secure framework using blockchain-based machine-learning for iot-driven smart cities. IEEE Trans. on Netw. Sci. and Eng., 8(3):2326–2341, 2021.
18. Hongju Cheng, Zhe Xie, Yushi Shi, and Naixue Xiong. Multi-step data prediction in wireless sensor networks based on one-dimensional cnn and bidirectional lstm. IEEE Access, 7:117883–117896, 2019.
19. Anmin Fu, Xianglong Zhang, et al. Vfl: a verifiable federated learning with privacy-preserving for big data in industrial iot. IEEE TII, 2020.
20. Jin Zhao, Jifeng Huang, and Naixue Xiong. An effective exponential-based trust and reputation evaluation system in wireless sensor networks. IEEE Access, 7:33859–33869, 2019.
21. Shaobo Huang, Zhiwen Zeng, et al. An intelligent collaboration trust interconnections system for mobile information control in ubiquitous 5g networks. IEEE TNSE, 8(1):347–365, 2020.
22. Yonglei Yao, Naixue Xiong, Jong Hyuk Park, Li Ma, and Jingfa Liu. Privacy-preserving max/min query in two-tiered wireless sensor networks. Computers & Mathematics with Applications, 65(9):1318–1325, 2013.
23. Zou Bin, Zhao Xiaohu, and Yin Zhishuai. image feature matching algorithm based on improved orb. Laser & Optoelectronics Progress, 58(2):0210006, 2021.
24. Ethan Rublee, Vincent Rabaud, et al. Orb: An efficient alternative to sift or surf. In IEEE ICCV, pages 2564–2571, 2011.
25. Feng Xia, Ruonan Hao, Jie Li, Naixue Xiong, Laurence T Yang, and Yan Zhang. Adaptive gts allocation in ieee 802.15. 4 for real-time wireless sensor networks. Journal of Systems Architecture, 59(10):1231–1242, 2013.
26. Hongwei Dong, Wei Song, et al. Autonomous recognition technology of carrier robot on various terrain environment. Institu. of Mechanical Engineers, Part D: J. of Auto. Eng., 235(9):2568–2584, 2021.
27. Shiqiang Yang, Guohao Fan, Lele Bai, Rui Li, and Dexin Li. Mgc-vslam: A meshing-based and geometric constraint vslam for dynamic indoor environments. IEEE Access, 8:81007–81021, 2020.
28. Xuyuan Yang and Guang Jiang. A practical 3d reconstruction method for weak texture scenes. Remote Sensing, 13(16):3103, 2021.
29. Fei Zhou, Limin Zhang, et al. Improved point-line feature based visual slam method for complex environments. Sensors, 21(13):4604, 2021.
30. R. Mur-Artal and J. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. IEEE T. on robotics, 33(5):1255–1262, 2017.
31. M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Comm. of the ACM, 24(6):381–395, 1981.
32. Jinjin Yao, Pengchao Zhang, et al. An adaptive uniform distribution orb based on improved quadtree. IEEE Access, 7:143471–143478, 2019.
33. Krystian Mikolajczyk, Tinne Tuytelaars, et al. A comparison of affine region detectors. Int'l J of comp. vision, 65(1):43–72, 2005.

Smart contract vulnerability detection model based on Siamese network

Weijie Chen[1], Ran Guo[2*], Guopeng Wang[3*], Lejun Zhang[1,2*], Jing Qiu[2], Shen Su[2], Yuan Liu[2], Guangxia Xu[2], Huiling Chen[4]

[1] College of Information Engineering, Yangzhou University, Yangzhou, 225127, China
[2] Cyberspace Institute Advanced Technology, Guangzhou University, Guangzhou, 510006, China
[3] Engineering Research Center of Integration and Application of Digital Learning Technology, Ministry of Education, Beijing, 100039, China
[4] Department of Computer Science and Artificial Intelligence, Wenzhou University, Wenzhou 325035, China
MZ120200850@yzu.edu.cn; guoran@gzhu.edu.cn; wangguopeng@sohu.com; zhanglejun@yzu.edu.cn; qiujing@gzhu.edu.cn; johnsuhit@gmail.com; yuanliu@gzhu.edu.cn; xugx@gzhu.edu.cn; chenhuiling.jlu@gmail.com

Abstract. Blockchain is experiencing the transition from the first generation to the second generation, and smart contract is the symbol of the second generation blockchain. Under the background of the explosive growth of the second-generation blockchain platform and applications represented by smart contracts, frequent smart contract vulnerability events seriously threaten the ecological security of the blockchain, reflecting the importance and urgency of smart contract vulnerability detection. In this paper, we proposed a smart contract vulnerability detection method based on a Siamese network. We combined the Siamese network with Long Short-Term Memory (LSTM) Network neural network to complete the task of smart contract vulnerability detection. The Siamese network used in this paper consists of two subnetworks that share the same parameters onto a low dimension and easily separable feature space. Siamese network is now widely used in the field of image similarity and target tracking. In this paper, we improve the Siamese network so that it can be used for smart contract vulnerability detection. By comparing with previous research results, the model has better vulnerability detection performance and a lower false-positive rate.

Keywords: Smart Contract, Deep learning, Siamese Network

1 Introduction

The idea of smart contract was proposed by Nick Szabo in the 1990s [1], but due to the lack of a trusted execution environment at the time, smart contracts were not used and developed. Blockchain has the characteristics of openness, transparency, decentralization and invariance. Due to its characteristics, blockchain is now used in the Internet of Things [2], finance and other fields. So blockchain can be used as a vehicle for smart

contracts [3]. Compared to traditional contracts, smart contracts are themselves participants and executors of the contract, so the process of contract execution does not require third-party involvement.

Due to the convenience, smart contracts are beginning to be widely used in finance, energy, smart cities and other fields. Ethereum [4] is one of the most popular blockchain platforms and has deployed tens of thousands of smart contracts controlling billions of dollars' worth Ether (Cryptocurrency of Ethereum). In 2016, hackers exploited the reentrancy vulnerability to attack The Dao, resulting in the loss of over 60 million USD [5]. Security incidents like this can create a serious trust crisis in the blockchain, so we need to build an efficient and smart contract vulnerability detection tool. In this paper, we propose a smart contract vulnerability detection model based on Siamese networks named SCVSN model, which performs vulnerability detection by calculating similarity and has the advantages of simple structure, high correct rate and low false positive rate. Through experiments, we demonstrate that our proposed SCVSN model has better performance in vulnerability detection compared to previous smart contract vulnerability detection tools. The main contributions of this paper are described as follows:

1) Applying the natural language processing side to smart contract processing, and proposing a smart contract embedding method that reduces the impact of irrelevant content on smart contract vulnerability detection.
2) This is the first model that applies Siamese network to smart contract vulnerability detection at the source code level. SCVSN improves the accuracy of smart contract vulnerability detection and reduces the false positive rate of smart contract vulnerability detection by calculating the similarity. And to ensure that the number of positive and negative sample pairs is balanced during training, we also propose a classification algorithm for positive and negative sample pairs.
3) Many experiments are performed to confirm the performance of SCVSN model, and this is compared with the recently proposed deep learning based smart contract vulnerability detection method.

The paper is organized as follows: We discuss our methodology for our study in Section II. We present the experimental procedure which includes dataset processing and performs performance comparisons in Section III. Finally, we conclude the whole paper and future work in Section IV.

2 Smart Contract Vulnerability Detection Model Based on Siamese Network

From the perspective of model structure, the SCVSN model has the following main steps: (1) The first step is to input the sol files. (2) The second step is to process the sol files and use word2vec for word embedding. (3) The third step SCVSN model performs feature extraction on the word vectors, calculates the Euclidean distance, and obtains the final result by comparing it with the boundary value(m). The architecture of the SCVSN model is shown in Figure 2.

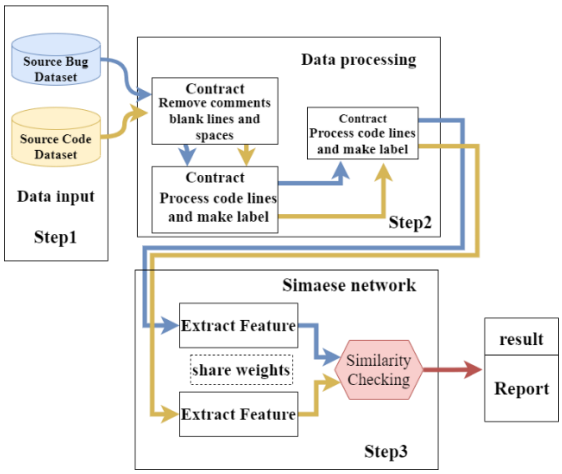


Fig. 2. SCVSN model structure

2.1 Siamese Network Structure in SCVSN Model

In the third step of our proposed model, feature extraction using the SCVSN model is required to calculate the Euclidean distance d. Then the Euclidean distance d is compared with the edge value margin to finally derive the result. The Siamese network structure used in our study is the combination of the Siamese network and LSTM neural network and the structure of the Siamese network used in this paper is shown in Figure 3. It can be seen that two subnetworks have the same structure and the weights are shared between the two subnetworks. The subnetworks in the Siamese network used in SCVSN model consisted of only the input layer, LSTM neural network layer, Dropout layer, ReLU layer and fully connected layer, the structure of the subnetwork is simple. The input data goes through the LSTM neural network layer to extract the features and then passes through the Dense layer, which nonlinearized the previously extracted features and extracts the association between them, and then passes through the Dropout layer to avoid overfitting, the data are then processed using the nonlinear ReLU activation function. For large-scale data, ReLU has a better fitting ability compared to tanh activation function and sigmoid activation function, and also better enhances the nonlinearity of the model, making the neural network more discriminative. The data goes through the Dropout layer and ReLU layer once more to further increase the stability of the model, followed by the vector transformation through the Dense layer, and finally the feature extraction results are output. After the two subnetwork models extract the feature values, the Siamese network determines whether the two smart contracts are similar by calculating the Euclidean distance between the output values of the two subnetworks. The processing of the Siamese network structure used in this paper can be represented by the following equation.

$$word2vec(C_1) \Rightarrow C_1 \tag{1}$$

$$word2vec(C_2) \Rightarrow C_2 \tag{2}$$

$$LSTM(C_1) \Rightarrow X_1 \tag{3}$$

$$LSTM(C_2) \Rightarrow X_2 \tag{4}$$

$$|Distance(X_1, X_2)| \Rightarrow d \tag{5}$$

Where $C_1$and C_2 denote sample smart contracts, C_1 and C_2 denote the input matrices obtained after word embedding, X_1 and X_2 denote the feature values obtained after processing by the LSTM network in the Siamese network, *d* denotes the distance between two smart contracts.

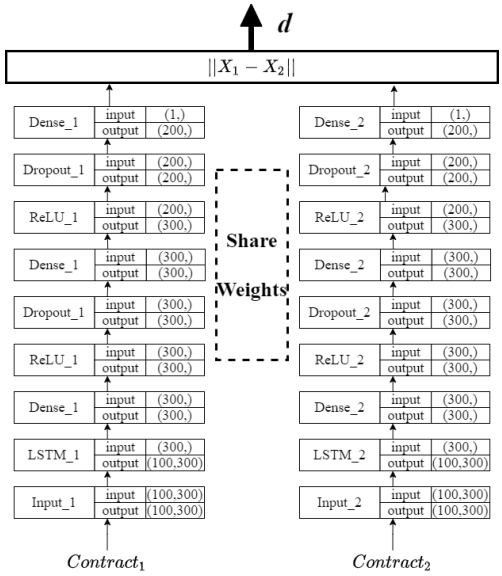


Fig. 3. Siamese network structure

LSTM can make full use of temporal relationship and semantic information among data. The benefits of using LSTM for our subnetwork are that LSTM is able to establish connections between units in a loop and remember previous inputs through their internal states, LSTM neural networks have specialized gate control structures capable of controlling the memory and forgetting of data [6], and LSTM neural networks do not suffer from gradient disappearance and gradient explosion. So, the LSTM neural network is used as the subnetwork of the Siamese network structure in the research.

2.2 The Core Idea of SCVSN Model

Our proposed approach uses Siamese network model to compute the similarity of two smart contracts, and based on comparing the similarity, thus performing vulnerability detection of smart contracts. The core idea of our proposed model can be briefly described as two subnetworks of SCVSN model, one of which extracts the feature of a

smart contract *A* and other extracts the feature of a smart contract *B*. Smart contract *A* contains the vulnerability and is used as a reference sample, smart contract *B* may or may not contain vulnerability. First of all, the features extracted from the two subnetworks are subjected to Euclidean distance calculation, and then the Euclidean distance is compared with the threshold value. If the value of Euclidean distance is lower than the threshold, it means that the two smart contracts are similar and smart contract *B* contains vulnerability, if the Euclidean distance is higher than the threshold, it means that the two contracts are not similar and thus it is judged that smart contract *B* does not contain the vulnerability.

In order to implement the idea, we proposed, the following issues need to be addressed:1). How to implement smart contract vulnerability detection via SCVSN model. 2). How to train SCVSN model to improve their feature extraction capabilities.3). How to reduce rate of false positives.

First question. Our proposed SCVSN model performs features extraction through subnetworks, After the training of the SCVSN model is completed (the training process described in detail in second question), the subnetwork that specifically extracts features of the smart contract containing the vulnerability saves the trained parameters. In testing or practical use, the sample to be tested is passed through another subnetwork of SCVSN model to extract features, then the Euclidean distance between the two-feature value is calculated, followed by a comparison of the Euclidean distance with a threshold value to determine whether the sample to be tested contains vulnerabilities. The structure of our proposed SCVSN model is simple, using only 2 LSTM neural network layers and 4 ReLU layers, but achieves good detection results and small computational effort. After the network is trained, the features of the reference samples (smart contracts containing vulnerabilities) can be extracted in advance, and the samples to be tested only need to extract their features and then calculate the Euclidean distance and compare it with the threshold value. The vulnerability detection process is shown in Figure 4.

Second question. Our approach is to first use the 50 smart contract samples that have been processed in the Source Bug Dataset as the reference samples, since these 50 smart contract samples are certain to contain vulnerabilities, then 50 smart contract reference samples containing vulnerabilities are combined with 2918 smart contract samples already processed in Source Code Dataset to form positive and negative sample pairs respectively. The positive sample is composed of a sample of smart contracts containing vulnerabilities in the Source Bug Dataset and a sample of smart contracts containing vulnerabilities in the Source Code Dataset, and this sample pair corresponds to a label of 1, negative sample is a sample pair consisting of a sample of smart contracts with vulnerabilities in the Source Bug Dataset and a sample of smart contracts without vulnerabilities in the Source Code Dataset, and this sample pair corresponds to a label of 0. After constructing a large number of positive and negative samples, the samples are input into the network model, which is used to train the network model and improve the ability of the SCVSN model to extract features.

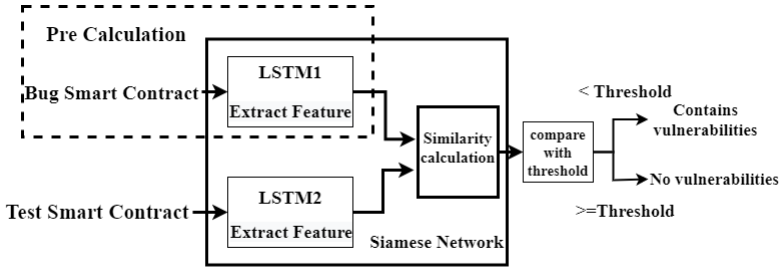


Fig. 4. Vulnerability detection process

Third question. Since smart contract vulnerability can appear anywhere in a smart contract, the fact that two smart contracts are similar does not necessarily mean that the tested smart contract contains the smart contract vulnerabilities. As shown in Figure 4, where Contract1 is the one containing reentrancy vulnerability When the attacker uses reentrancy vulnerability to attack, Contract1 will not execute according to the logic intended by the contract creator, where the vulnerability appears in line 12. Contract2 does not contain reentrancy vulnerability. These two smart contracts are similar, but one contains reentrancy vulnerability and the other does not. If Contract1 is used as the reference sample, the result must be that the two smart contracts are similar by the usual method of calculating similarity, thus proving that Contract2 contains reentrancy vulnerability, but this is not the actual case. By training the SCVSN model, the false positive rate of smart contract vulnerabilities in this case can be reduced. The previous subsection illustrates that SCVSN model uses two subnetworks together to extract features, and the parameters are shared between the subnetworks, providing a stronger feature extraction capability than a single network, so the loss function used in SCVSN model needs to be able to handle the relationship among sample pair labels, Euclidean distance and threshold. Therefore, the loss function used in our proposed SCVSN model is Contrastive Loss which shown in Eq. (6).

$$loss = \frac{1}{2N}\sum_{n=1}^{N} yd^2 + (1-y)\max(m-d,0)^2 \tag{6}$$

In the formula, d denotes the Euclidean distance between the features of two smart contract samples, which can also indicate the degree of similarity between two smart contract samples, with the expression $d = \left|\left|a_n - b_n\right|\right|^2$, where a and b denote two different smart contracts. y indicates the label of weather two smart contract samples match, y=1 means they are similar, y=0 means they are not similar, and this labelling rule is also used when construct the labels of positive and negative sample pairs, m indicates threshold. The advantage of our choice to use Contrastive Loss as the loss function is that the overall loss function ensures that sample pairs that are already similar remain similar when SCVSN model performs feature extraction, while sample pairs that are not similar remain dissimilar after feature extraction. When the sample pair label y=1, the loss function is shown in Eq. (7).

$$loss = \frac{1}{2N}\sum_{n=1}^{N} yd^2 \tag{7}$$

The sample pair label y=1 indicates that both smart contracts composing the sample pair contain vulnerabilities, and if the Euclidean distance between the two smart sample features is too large, it indicates that SCVSN model is not good at extracting critical information and needs to increase the loss. When the sample pair label y=0, the loss function is shown in Eq. (8).

$$loss = \sum (1-y)\max(margin - d, 0)^2 \tag{8}$$

The sample pair label y=0 indicates that one of the two smart contract samples is not containing the smart contract vulnerability. If the Euclidean distance between the two smart contract sample pairs is too small, the same indicates that SCVSN model is not good at extracting critical information and needs to increase the loss. If the Euclidean distance between sample pairs is too large over the threshold margin, it is a normal sample pair, but it does not help the model's ability to extract features, so we choose to ignore these sample pairs, which can greatly reduce the amount of computation and increase the training speed.

3 Experiment

3.1 Experimental Procedure

During the experiments, the evaluation criteria for model performance is the accuracy of the model on the test set. We recorded the training data for each epoch and calculated the average of the accuracy and loss values based on the recorded data, as shown in Figure5.

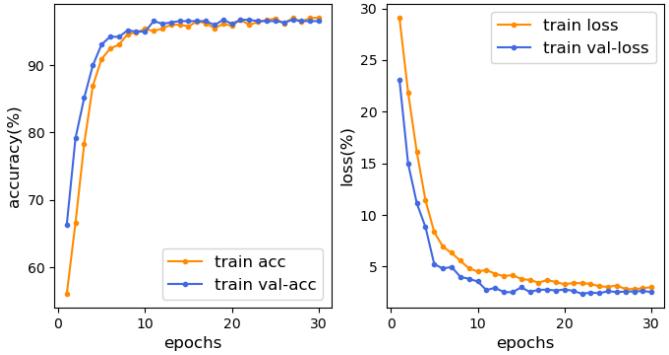


Fig. 5. Training results

In the research, SCVSN model uses two datasets, the first one is a recently released large-scale dataset named Smart Bugs Dataset-wild [7]. It contains 47398 real and unique sol files, which together have about 203,716 smart contracts in total, this is because a sol file may contain more than one smart contract. The Source Code Dataset used for training is extracted from this dataset. In addition, the SCVSN model used another small dataset SolidiFI Benchmark [8], which contains 7 different vulnerabilities, each vulnerability type contains 50-sol files, and this dataset serves as Source Bug Dataset.

The plot on the right shows the relationship between epoch and loss function, and the plot on the left shows the relationship between epoch and accuracy. In this paper, the data of each epoch is recorded during the training to facilitate the observation and optimization of the model. From the change of the curve, it can be seen that the loss value and the correct rate of the model on both the training and test sets are changing rapidly in the first 10 epochs. After 10 epochs, the loss values and accuracy rates of the model on the training and test sets fluctuate up and down regularly, and the overall trend tends to level off, indicating that the model starts to converge gradually at this point. In the right plot, it can be seen that the curve of loss declines in the test set and training set is gradually easing, and there is no violent jitter, indicating that the value of batch size in the experiment is correct. The loss curves of the training and test sets crossed during the training process, and the distance between the two curves was moderate, without overlap or excessive distance, indicating that there was no overfitting or underfitting in the training process.

In order to correctly represent the data of the model, we generate a graph of the data from multiple training of the model, and the specific results are shown in Figure 6. The straight line in the figure indicates the average accuracy calculated from the test results after several training times. The results of 40 times of the model on the test set are recorded in Figure 6, with the worst accuracy of 94.35% and the best accuracy of 97.14%.

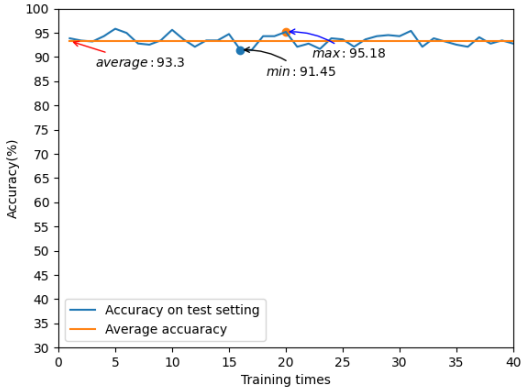


Fig. 6. Multi-training results

By averaging the 40 results, the average correct rate was calculated to be 96.01%, and it can also be seen that the results of all 40 tests fluctuate around 96.01%. So, we conclude that our proposed SCVSN model has a correct detection rate of 96.01% for smart contract reentrancy vulnerability.

3.2 Experimental Performance Comparison

The performance metrics are those introduced above, including ACC, REC, FPR and F1. In the same experimental setting, the DeeSCV Hunter, the Peculiar, the BLSTM-ATT, and the TMP were selected for comparison. SCVSN data is the average of multiple test results. The results of the comparison are shown in the table2. The measurements we chose were accuracy (ACC), precision (PRE), F1-score (F1), and recall rate (REC). They can be calculated by the following formulas:

$$\mathrm{ACC} = \frac{TP+TN}{TP+TN+FP+FN} \tag{10}$$

$$\mathrm{TPR} = \frac{TP}{TP+FP} \tag{11}$$

$$\mathrm{Recall} = \frac{TP}{TP+FN} \tag{12}$$

$$\mathrm{FPR} = \frac{FP}{FP+TN} \tag{13}$$

$$\mathrm{F1} = \frac{2*PRE*TPR}{PRE+TPR} \tag{14}$$

TP denotes positive samples predicted by the model as positive class, TN denotes negative samples predicted by the model as negative class, FP denotes negative samples predicted by the model as positive class, and FN denotes positive samples predicted by the model as negative class.

It can be seen from Table 1 that our proposed SCVSN model has an excellent performance in smart contract vulnerability detection. The SCVSN model has relatively stable values in all four-performance metrics, where the SCVSN model has a correct rate of 96.01%, which is better than the models involved in the comparison. The Recall of the SCVSN model proposed in this paper is 96.64%, which exceeds the rest of the models, this shows that the SCVSN approach to detecting smart contract vulnerabilities can reduce the false positive rate in vulnerability detection.

Table1. Comparison of experimental results.

Model	ACC (%)	REC (%)	PRE (%)	F1(%)
SCVSN	96.01	96.04	94.25	94.78
DeeSCVHunter [9]	93.02	83.46	90.70	86.87
Peculiar [7]	92.37	92.4	91.80	92.10
BLSTM-ATT [10]	88.47	88.48	88.5	88.26
TMP [11]	84.48	82.63	74.06	83.82
DR-GCN [11]	81.47	80.89	72.36	76.39
LSTM	81.91	91.43	76.80	83.48

4 Conclusions

We apply Siamese networks to smart contract vulnerability detection for the first time, and we perform smart contract vulnerability detection by comparing similarity. Our proposed SCVSN model has an excellent performance in smart contract vulnerability

detection due to the simple structure of Siamese networks, powerful similarity computation capability and feature extraction ability. In the work of this paper, we apply a natural language processing approach to the processing of smart contracts for two realistic datasets and apply the processed dataset to the training and testing of the model. We demonstrate through extensive experiments that our proposed SCVSN model can achieve good results in smart contract reentrancy vulnerability detection tasks. Our research also provides a new way of thinking about vulnerability detection, specifically by calculating the similarity to detect code vulnerabilities, and our code will be open-sourced afterwards.

SCVSN can detect whether the smart contract contains vulnerabilities, but cannot detect what kinds of vulnerabilities, so our future work is to study how to make the SCVSN model detect the types of vulnerabilities contained in the smart contract.

References

1. Giancaspro, M. Is a ‘smart contract’ really a smart idea? Insights from a legal perspective. Computer law & security review 33(6). 825-835(2017).
2. Qiu, H., Qiu, M., Memmi, G. A dynamic scalable blockchain based communication architecture for IoT. International Conference on Smart Blockchain. 159-166(2018).
3. Khan, S.N., Loukil F., Ghedira-Guegan, C., Benkhelifa, E., Bani-Hani, A. Blockchain smart contracts: Applications, challenges, and future trends. Peer-to-peer Networking and Applications 14(5). 2901-2925(2021).
4. Oliva, G. A., Hassan, A. E., Jiang, Z. M. An exploratory study of smart contracts in the Ethereum blockchain platform. Empirical Software Engineering 25(3). 1864-1904(2020).
5. Samreen, N. F., Alalfi, M. H. Reentrancy vulnerability identification in ethereum smart contracts. In: 2020 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE) 2020, pp. 22-29.
6. Karevan, Z., Suykens, A. Transductive lstm for time-series prediction: An application to weather forecasting. Neural Networks 125. 1–9(2020).
7. Wu, H., Zhang, Z. Wang, S., Lei, Y., Lin, B., Qin, Y., Zhang, H., Mao, X. Peculiar: Smart contract vulnerability detection based on crucial data flow graph and pre-training techniques. In: 2021 IEEE 32nd International Symposium on Software Reliability Engineering (ISSRE) 2021. pp. 378-389.
8. Ghaleb, A., Pattabiraman, K. How effective are smart contract analysis tools? evaluating smart contract static analysis tools using bug injection. In: Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis 2020. pp. 415-427.
9. Yu, X., Zhao, H. Hou, B., Ying, Z., Wu, B. Deescvhunter: A deep learning-based framework for smart contract vulnerability detection. In: 2021 International Joint Conference on Neural Networks (IJCNN) 2021. pp. 1-8.
10. Qian, P., Liu, Z., He, Q., Zimmermann, R., Wang X. Towards automated reentrancy detection for smart contracts based on sequential models. IEEE Access 8. 119685–19695 (2020).
11. Zhuang, Y., Liu, Z., Qian, P., Liu, Q., Wang, X., He, Q. Smart contract vulnerability detection using graph neural network. In: IJCAI 2020. pp. 3283–3290.

Context-User Dependent Model for Cascade Retweeter Prediction

Tong Chunyan[1], Zhang Kai[2], Yang Song[3], Zhang Zheng[4], Zhang Hongfeng[5], Wang Hao[6], Shuang Xinzhuo[7], and Liu Yerui[8⋆]

[1] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
tongchunyan@people.cn
[2] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
zhangkai@people.cn
[3] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
yangsong@people.cn
[4] State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing 100733, China,
zhangzheng@people.cn
[5] Wuhan Second Ship Design and Research Institute, Wuhan 430064, China,
2654565030@qq.com
[6] Wuhan Second Ship Design and Research Institute, Wuhan 430064, China
whuwanghao@163.com
[7] Fuxin Higher Vocational College, Liaoning 123000, China.
382010@qq.com
[8] Electronic Information School, Wuhan University, Wuhan, 430072, China
1093252514@qq.com

Abstract. This paper proposes a retweeter predction model based on attention model and Tranformer encoding–Forwader Prediction Model based on User Preference (FPM-UP). Considering the impact of release time, content, and current external context information in the forwarding process, FPM-UP integrates user attribute embedding and context-user dependency into a temporal and text attention model for the prediction of the next forwarding user. Compared with the existing methods, FPM-UP significantly improves the prediction accuracy.…

Keywords: Social Network, Forwarding Prediction, Attention Mechanism, Graph Neural Network, Artificial Neural Network.

1 Introduction

In major online social networking platforms such as Twitter or Weibo, users are connected by their following status and the information can diffuse via users'

⋆ Corresponding author

reposting behavior from their followees. Such a retweeting behavior pushs the original information to more users thus accelerating the diffusion of the original information. Such diffusion paths are also refered to as cascade structures, which has been widely studied [1–6].

Generally, retweeter prediction models can be divided into two categories: social network prediction meth-ods and diffusion path prediction methods.

The social network prediction method describes users and their relationships as a complex network, in which retweeting behavior is carried on the edges. This method originally originated from two theories: Independent Cascade (IC) and Linear Threshold (LT) [7]. Recently, many papers have revised the theory [8, 9]. The basic problem of this method is that the assumptions are usually difficult to meet in realworld data.

The diffusion path prediction method expresses the retweeting process as a user sequence. The goal of the model is to predict the next or all future users participating in the retweeting of the information given the histor-ical diffusion path of a piece of information.

RNN model is used in the SNIDSA model proposed by Wang [10] to extract the currently observed retweeting information. Attention mechanism is used to learn the relationship between users before RNN. CYAN-RNN model uses a two-layer RNN structure to learn global information and local information separately. Also, attention mechanism [11] is used to correct the impact of current global information on local information. In the DCE model pro-posed by Zhao [12], the joint embedding method is used to learn the influence of the diffusion sequence to learn the user embedding vector. These models have had some success. However, due to the inherent defect that RNN cannot learn long-term feature when extracting sequence features, none of these models can achieve better results.

Many scholars have relied on the attention mechanism in diffusion sequence prediction to achieve good results. In the HiDAN model proposed by Wang [13], the traditional RNN model architecture was abandoned, and a redesigned non-sequential model based on the attention mechanism was used to predict the diffusion sequence. Ma [14] discussed the influence of followers' hot topic discussion on users with self-attention model.

In addition, some scholars try to describe the attractiveness of other information to users. In the HMCF model proposed by Yang [15], text and visual information are used to describe the attractiveness of graphic information in different places of brochures to users. Wu [16] used user learning model and object learning mod-el in its model to learn user preference characteristics and information content characteristics. Wang [17] pro-posed to design corresponding description vectors for different roles of users, discussing their different characteristics as publishers and receivers.

In summary, current works ignore the underlying users graph and text information in the information. However, it is possible to extract enough features to pre-dict both popularity and retweeters.

In this paper, a model is proposed for different features based on the results of the analysis of forwarding behavior. For user relationship network, this

paper embeds user attributes through Graph Attention Network. For the user's forwarding process, this paper designs an attention model to extract the dependency information between context and users. Subsequently, the propaga-tion process relies on the Transformer encoder to further extract user relationship information. Finally, this paper designs a temporal and text attention model to model the influence of release time, the content, and the current external context information on the forwarding process.

The rest of this paper is organized as follows. Data analysis is illustrated in Section 2. Models are defined in Section 3. In Section 4, we analyze the experimental results. Finally, we briefly review the work in Section 5.

2 Data Analysis And Discoveries

2.1 Data Analysis and Discoveries

Many studies have shown that, time, user, structure, and content characteristics are effective for information popularity prediction. We analyze the effectiveness of different features in predicting information popularity and retweeters from the perspective of information entropy.

2.2 Text Content And Publish Time and Delay Time

Text content is the main information described by microblog, which is the content that users read before retweeting.

For each user, the retweeting frequency of different content is analyzed. The counters are normalized for eliminate the unbalance of categories' counter and user's retweeting times. In order to investigate whether users' preferences are stable, we calculated the information entropy of all users' retweeting probablities, which is shown in Equation (1).

$$E_u = -\sum_{c \in C} P_u(c) \log P_u(c) \tag{1}$$

Where C means all possible text content categories, and $P_u(c)$ means the probability that user u retweets a microblog of category c.

Time mainly affects the activeness of users, which indirectly affects the probability of retweeting. Time features can be divided into two categories: release time features and delay time features.

In order to quantitatively describe the impact of the release time feature on the user's willingness to retweet, we use Shannon entropy of information published by the user when the release time of the information is not given.

$$H_{pt}^{sh} = -\sum_{u \in G, p \in P} P(u,p) \log P(u,p) \tag{2}$$

Where P represents the set of all possible publish times. H_{pt}^{sh} means the information entropy of users' retweeting. P(u,p) is the retweeting probability of user u at release time p.

To distinguish the influence of publish time, the Condition entropy of user retweeting with the same publish time was calculated as in Equation (3).

$$H_{pt}^{con} = -\sum_{p \in P} P(p) \sum_{u \in G} P(u,p) \log P(u,p) \tag{3}$$

The second type of time feature is the interval between the retweeting behavior and the release time of the content which mainliy affects the popularity of the content. With the extension of time, the popularity decay is approximately exponential.

3 Model

3.1 Framework

To solve the above problems, we propose a Forwader Prediction Model based on User Preference (FPM-UP). In FPM-UP, in order to deal with the observed retweeting information, we use Transformer encoders as sequence feature extractors. Transformer encoders are suitable for dealing with such problems where the relationship between nodes appears unclear. At the same time, there is a delay in each step of the retweeting process. This pa-per uses the attention mechanism to consider the influence of the delay time on each step of the retweeting process. For the release time and text content of the mi-croblog, we embedded them into vectors respectively, and again used the attention mechanism to consider their influence on the retweeting process, so as to get the final retweeting cascade embedded vector. Finally, we consider the similarity between the obtained sequence expression vector and the user expression vector, and select the user with the highest similarity as the prediction result. The overall frame of the model is shown in Figure 1.

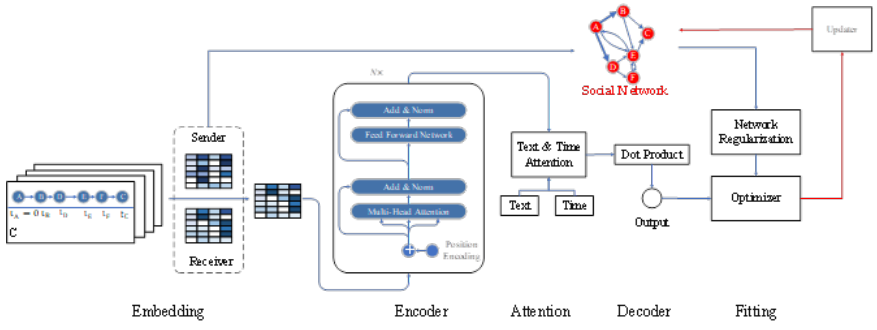


Fig. 1. FPM-UP model framework

3.2 Tweet Embedding

Content and release time belong to the information of microblogs themselves, which mainly affect users' pref-erence for different types of microblogs.

For text information, FPM-UP uses the standard Bert model to embed text content as vectors. In order to include the influence of external information on content preference, we extracted the content of "Hot Search List" from Weibo, converted it into vector expression, and calculated its similarity with the content currently pro-cessed.

$$\beta_i = \frac{exp(f_t(W_t x_t + b_t) \cdot f_h(W_h x_h + b_h))}{\sum_{i=0}^{N_h} exp(f_t(W_t x_t + b_t) \cdot f_h(W_h x_h + b_h))} \tag{4}$$

The external influence vector finally obtained as Equation (5):

$$e_h = \sum_{i=0}^{N_h} \beta_i x_h \tag{5}$$

Where, x_t represents the embedded information of the text content, and x_h represents the embedded information of the top search list content.

In addition, the release time of a microblog will affect the user's activity, thus affecting the user's retweeting intention. Cycle information is used to describe the publishing time characteristics of Weibo. The cycle information represents the time period of the day when the release time is located. One-hot coding is converted to vector representation using the full connection layer.

$$e_p^t = f_d(W_{td} x_w + b_w) \tag{6}$$

The $W_{td} \in R^{d_t \in N_d}$, $b_w \in R^{d_t}$ is the model parameters, N_d is the number of segments of one day, whose time length is $24/N_d$ hours. The final embedded vector is

$$e^w = f_x(W_w[x_t, e_h, e_p^t] + b_w) \tag{7}$$

in which $[\cdot]$ is concatenation operation.

3.3 Retweeting Sequence Embedding

To embed the retweeting sequence accurately, each user is transformed into a user expression vector using two full connection layers, one vector representing the extent to which the user influences other users as a pub-lisher and the other vector representing the extent to which the user prefers different content as a receiver.

$$e_i^{up} = f_x(W_u p x_i + b_u p) \tag{8}$$

$$e_i^{ur} = f_x(W_u r x_i + b_u r) \tag{9}$$

where $W_{up} \in R^{d_{up} \times N}$,$W_{ur} \in R^{d_{ur} \times N}$,$b_p \in R^{d_{up}}$,$b_r \in R^{d_u r}$is model parameters, d_{up} and d_{ur} represent publisher embedded dimension and receiver embedded dimension respectively, and f_x represents nonlinear activation function.

The embedding of retweeting sequence mainly uses the encoder module in Transformer module. In each layer of Transformer, the input data is an expression of the user's influence, arranged in the order in which it is retweeted, denoted as $C_o \in R^{d_{up} \times l}$, where l represents the sequence length. The embedding results are mainly achieved through the self-attention mechanism.

$$SA\left(C_o\right) = f_x\left(\frac{\left(C_o \cdot W^Q\right) \cdot \left(C_o \cdot W^K\right)^T}{\sqrt{d_{up}}} \cdot M\right) \cdot \left(C_o \cdot W^V\right) \tag{10}$$

In order to analyze the influence of delay information on retweeting intention, the corresponding vector representation $\lambda_i \in R^{d_d}$ is determined according to different delay times. Then, the vector representation is spliced with the vector representation of microblog content and publishing time to obtain $e^f = [\lambda_i, e^w]$. The coefficient of attention mechanism was calculated as Equation (11):

$$\alpha_i = \frac{\exp\left(w^f \cdot \left(e^f \odot e_i^c\right)\right)}{\exp\left(w^f \cdot \left(e^f \odot e_i^c\right)\right)} \tag{11}$$

where $\omega^f \in R^d$ is model parameter. Finally, the overall embedding result is shown in Equation (12):

$$e^c = \sum_{i=1}^{n} \alpha_i \cdot e_i^c \tag{12}$$

3.4 Prediction

Given the cascade embedding result e^c at the time of t_i, the estimated probability of user u_i retweeting the Weibo is shown in Equation (13):

$$\hat{p}(u_{i+1}|c_i) = softmax(e_c \cdot e_{ur} \tag{13}$$

The objective of model optimization is to maximize the retweeting probability of real retweeters, and the objective function is shown in Equation (14):

$$\mathcal{L} = -\sum_{m=1}^{M} \sum_{i=1}^{n_m - 1} \log \hat{p}\left(u_{i+1} \mid c_i\right) \tag{14}$$

4 Experiments

4.1 Dataset

We used Douban, China's largest online platform for book and movie sharing, where users can share their current reading status.

In this dataset, each book is regarded as a piece of content. This dataset contains the encrypted user ID and reading time, as well as the relationship network between users.

Finally, we extract dataset for retweeter predition through sampling. The statistical data of the dataset we use in this paper is shown in Table 1.

Table 1. Douban Dataset for retweeter prediction

Item	Information
Number of Users	3796
Number of Tweets	3349
Number of Edges	79356
Average Retweeting Times	21.66
Average Occurrences of Users	19.99

4.2 Experimental Result

For retweeter prediction, the performance of the model refers to the success rate of predicting the next retweeting user. Two methods are usually used to quantify the model performance: MRR index (Mean Reciprocal Rank) and A@k accuracy.

MRR uses probability ranking to measure model performance, and the specific calculation method is as Equation (15):

$$MRR = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{rank_i} \tag{15}$$

Where $rank_i$ represents the probability ranking of the correct result in the ith prediction result. The larger the MRR value, the better the model performance.

A@k represents the probability of real retweeting users ranking top k in probability.

$$A_k = \frac{\sum_{i \in U} |\{u_i \mid u_i \in U_{ik}\}|}{|U|} \tag{16}$$

The higher A@k is, the better the model performance is, and the smaller k is, the more important the negative index is.

Here we use map@k, which comprehensively considers the effect of A@k and MRR. It considers the probability ranking of the model and the ranking of the model itself by applying an average operation on the ranking result.

$$map_k = \frac{1}{N} \sum_{i=1}^{N} s\left(r_i, k\right) \tag{17}$$

$$s\left(r_i, k\right) = \begin{cases} \frac{1}{N}, r_i \leq k \\ 0, r_i > k \end{cases} \tag{18}$$

The experimental results are shown in Table 3 and Table 4.

Table 2. Experimental Result

Model	MRR	map@1	map@10	map@20	map@50	map@100
FOREST	0.03400	1.74	3.01	3.15	3.26	3.30
HiDAN	0.04783	1.87	3.97	4.27	4.51	4.62
DeepDiffuse	0.04283	1.90	3.66	3.87	4.04	4.13
TopoLSTM	0.03751	1.34	3.03	3.29	3.49	3.59
SNIDSA	0.04621	1.93	4.03	4.23	4.41	4.49
FPM-UP	0.05210	2.05	4.22	4.60	4.90	5.04

By observing the data in the experimental results, the following rules can be found.

- The FPM-UP model significantly improves the pre-diction accuracy on the same dataset. For instance, The FPM-UP model's accuracy rate is about 9.1% higher in comparison with HiDAN in terms of map@100, proving that using FPM-UP on cascade modeling can improve the prediction accuracy.
- The nonsequential models are more suitable for the modeling of forwarding cascades. In the com-parison methods, HiDAN, SNIDSA and the FPM-UP method proposed in this paper are all non-sequential models.

5 Conclusion

This article summarizes the current processing se-quence prediction problem of the main difficulties and verifies the impact of a variety of characteristics for re-tweeting process. A diffusion sequence prediction model based on Transformer encoder is proposed. This model uses relevant methods of representation learning to learn the influence of upstream users on subsequent retweeting. In the end, this paper verifies the performance of the model on the real data set. The results show that the accuracy of the proposed model has about 10% higher accuracy than that of the current models.

Acknowledgement

This work was supported by the Open Funding Projects of the State Key Laboratory of Communication Content Cognition (No. 20K05 and No. A02107).

References

1. Peng Bao, Hua-Wei Shen, Junming Huang, and Xue-Qi Cheng. Popularity prediction in microblogging network: a case study on sina weibo. In *Proceedings of the 22nd international conference on world wide web*, pages 177–178, 2013.
2. Justin Cheng, Lada Adamic, P Alex Dow, Jon Michael Kleinberg, and Jure Leskovec. Can cascades be predicted? In *Proceedings of the 23rd international conference on World wide web*, pages 925–936, 2014.
3. Shuai Gao, Jun Ma, and Zhumin Chen. Effective and effortless features for popularity prediction in microblogging network. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 269–270, 2014.
4. Bolei Zhang, Zhuzhong Qian, and Sanglu Lu. Structure pattern analysis and cascade prediction in social networks. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 524–539. Springer, 2016.
5. Keke Gai, Meikang Qiu, Zhong Ming, Hui Zhao, and Longfei Qiu. Spoofing-jamming attack strategy using optimal power distributions in wireless smart grid networks. *IEEE Transactions on Smart Grid*, 8(5):2431–2439, 2017.
6. Keke Gai, Yulu Wu, Liehuang Zhu, Lei Xu, and Yan Zhang. Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks. *IEEE Internet of Things Journal*, 6(5):7992–8004, 2019.
7. David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146, 2003.
8. Sheng Gao, Huacan Pang, Patrick Gallinari, Jun Guo, and Nei Kato. A novel embedding method for information diffusion prediction in social network big data. *IEEE Transactions on Industrial Informatics*, 13(4):2097–2105, 2017.
9. Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. Deepinf: Social influence prediction with deep learning. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2110–2119, 2018.
10. Zhitao Wang, Chengyao Chen, and Wenjie Li. A sequential neural information diffusion model with structure attention. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 1795–1798, 2018.
11. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
12. Yuhui Zhao, Ning Yang, Tao Lin, and S Yu Philip. Deep collaborative embedding for information cascade prediction. *Knowledge-Based Systems*, 193:105502, 2020.
13. Zhitao Wang and Wenjie Li. Hierarchical diffusion attention network. In *IJCAI*, pages 3828–3834, 2019.
14. Renfeng Ma, Xiangkun Hu, Qi Zhang, Xuanjing Huang, and Yu-Gang Jiang. Hot topic-aware retweet prediction with masked self-attentive model. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 525–534, 2019.
15. Yang Yang, Yaqian Duan, Xinze Wang, Zi Huang, Ning Xie, and Heng Tao Shen. Hierarchical multi-clue modelling for poi popularity prediction with heterogeneous tourist information. *IEEE Transactions on Knowledge and Data Engineering*, 31(4):757–768, 2018.

16. Qitian Wu, Yirui Gao, Xiaofeng Gao, Paul Weng, and Guihai Chen. Dual sequential prediction models linking sequential recommendation and information dissemination. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 447–457, 2019.
17. Zhitao Wang, Chengyao Chen, and Wenjie Li. Information diffusion prediction with network regularized role-based user representation learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(3):1–23, 2019.

FedGraph-KD: An Effective Federated Graph Learning Scheme Based on Knowledge Distillation

Shiyu Wang[1,⋆], Neal N. Xiong[2], Jiahao Xie[1], and Mingming Lu[1]

[1] School of Computer Science and Engineering, Central South University, ChangSha, 410083, HuNan, China
[2] Department of Computer, Mathematical and Physical Sciences, Sul Ross State University, Alpine, TX 79830, USA

Abstract. Graph Neural Networks (GNNs) have achieved success in a variety of domains due to their potent graph-data processing skills. However, it is highly challenging to gather graph structure data from various universities and apply GNNs for centralized training due to privacy concerns and regulatory limitations. As a solution, Federated Graph Neural Networks (Fed-GNN) do not need to share data but support collaborative training of public models by sharing parameters or features between multiple parties. Thus, Fed-GNN has gained more attention recently. However, existing Fed-GNN schemes do not consider the problem that the participants of the public model often have different private GNN models, i.e., the model heterogeneity problem, which will lead to failure in the model heterogeneity scenario. To address the above issue, this paper explored an effective novel Federated Graph Learning scheme Based on the Knowledge Distillation models (FedGraph-KD). On the one hand, each client trains its local models through knowledge distillation. On the other hand, this paper uses a federated learning framework to update the shared model parameters. Extensive experiments and analyses on several different graph classification datasets demonstrate the effectiveness of our approach.

Keywords: Federated graph neural network · Model heterogeneity · Knowledge distillation.

1 Introduction

With the generation and use of enormous volumes of data, data privacy and security issues are gaining attention. Various fields have proposed different approaches to protect data privacy and network security [4–6,8,12,14,22,27,29,31]. Also, various approaches have emerged for efficiently transferring and processing big data on existing resources [2,3,7,9,18,23–25]. To address isolated data island problem in graph data field, Federated Graph Neural Networks(Fed-GNN) [16], which is the conbination of federated learning and GNN [1], has proposed as

⋆ Corresponding author: 224711075@csu.edu.cn

a novel distributed machine learning paradigm and gained rapidly development in recent years. The current research on Fed-GNN mainly focuses on aggregation algorithms and applications, however, Fed-GNN has a problem with model heterogeneity. Due to the differences in the local hardware, each client has a unique local GNN model, which is challenging for aggregating the parameters of the various client models in federated learning. Thus, how to make heterogeneous client models collaborate with federated learning to complete the training while satisfying local application requirements is an urgent problem to be solved. Additionally, GNN produces high costs when training larger graphs. For example, the Graph Convolutional Neural Network(GCN) [13], a typical GNN model, stacks numerous similar graph-structure network layers, resulting in too many model parameters. Therefore, it is meaningful work to transfer the knowledge obtained by GNN models to clients who cannot apply GNN models when there is a hardware bottleneck restriction. To address the above issue, this paper proposes federated graph learning based on a knowledge distillation model (FedGraph-KD). It has the following two main contributions:

- To address the computational differences in the client models, Knowledge distillation is used to transfer the knowledge learned by the deep GNN model to the shallow GNN model on the client. Specifically, we take a single-layer GNN model and an MLP model as the local model to simulate clients with feeble computational performance.
- Federated learning is introduced to protect the clients' privacy. We firstly use the public dataset to train a deep GNN model and then use local private datasets to fine-tune the client models.

The experiments are performed on four different graph classification datasets and the experimental results demonstrate the effectiveness of our approach.

2 Related work

In traditional federated learning, to address the model heterogeneity problem, Shen et al. present FML [21], which trains a homogeneous and a heterogeneous model locally by deep mutual learning [30]. The homogeneous model has the same architecture for all clients, and the heterogeneous model is used as the customized local model. Li et al. combine knowledge distillation to propose heterogenous Federated Learning via Model Distillation(FedMD) [15]. FedMD allows the client to create uniquely designed models. To avoid the risk of privacy leakage, FedMD combines all client models' logical layer outputs on the public dataset and their private dataset to train client models. However, it does not perform well in extreme cases of model heterogeneity, i.e., when some clients' models are particularly weak (e.g., only 2-layer MLP). The knowledge extracted by MLP on the public dataset is usually inferior to the deep CNN [20] models of other clients, which affects the accuracy of knowledge extraction on the public dataset by all clients. For different clients, the above methods only consider the number of layers and the number of neurons per layer. In contrast, in the GNN field, GNN models can be classified into five types [33] based on

their different aggregation methods. The GNN models of different clients will not only have variability in depth but also have noticeable differences in their network design, i.e., their network layers also have variability in computation and inference. Hence, the heterogeneity based on Fed-GNN is more challenging and meaningful to study.

3 Method

3.1 Overview

To solve the model heterogeneity problem in Fed-GNN, this paper proposes a new Fed-GNN model framework: Federated Graph Learning Based on Knowledge Distillation (FedGraph-KD). On the one hand, FedGraph-KD uses FedMD for reference. On the other hand, this paper uses the GLNN [28] as a local model for each client. In addition, FedGraph-KD combines knowledge distillation to transfer the knowledge to each client. Overall, FedGraph-KD consists of two main phases as follows:

(1) **Knowledge distillation.** The server trains a deep GNN model (D-GNN) on the public dataset and distributes it to each client. Each client trains its local model on the private dataset by combining D-GNN and knowledge distillation. We take a single-layer GNN model and an MLP model as the local model to simulate clients with feeble computational performance.
(2) **Federated learning.** Based on transfer learning [34], the clients fine-tune the partial network of D-GNN using local datasets. The server aggregates only the last part of the logical layers' parameters of all the client D-GNN and redistributes the aggregated model to the clients. Further, the client only uses D-GNN to do inference and prediction to obtain soft labels, which do not need back-propagation to train the model parameters. This reduces the computation and memory costs significantly and effectively improves the training speed and accuracy of the local model.

3.2 Network Architecture

As shown in Figure 1, FedGraph-KD consists of five main processes:

(1) **Pre-training.** The server trains a deep GNN on a public dataset until the model converges.
(2) **Clients download pre-trained models.** Each client downloads the server D-GNN model as the T-Net of knowledge distillation in local training.
(3) **Local training.** Use knowledge distillation to train S-Net, i.e., local shallow GNN or MLP.
(4) **Client transfer learning.** Fine-tune local D-GNN models using private datasets.
(5) **Federated learning.** The server aggregates the parameters of the last layers of D-GNN from each client and updates D-GNN with the updated parameters.

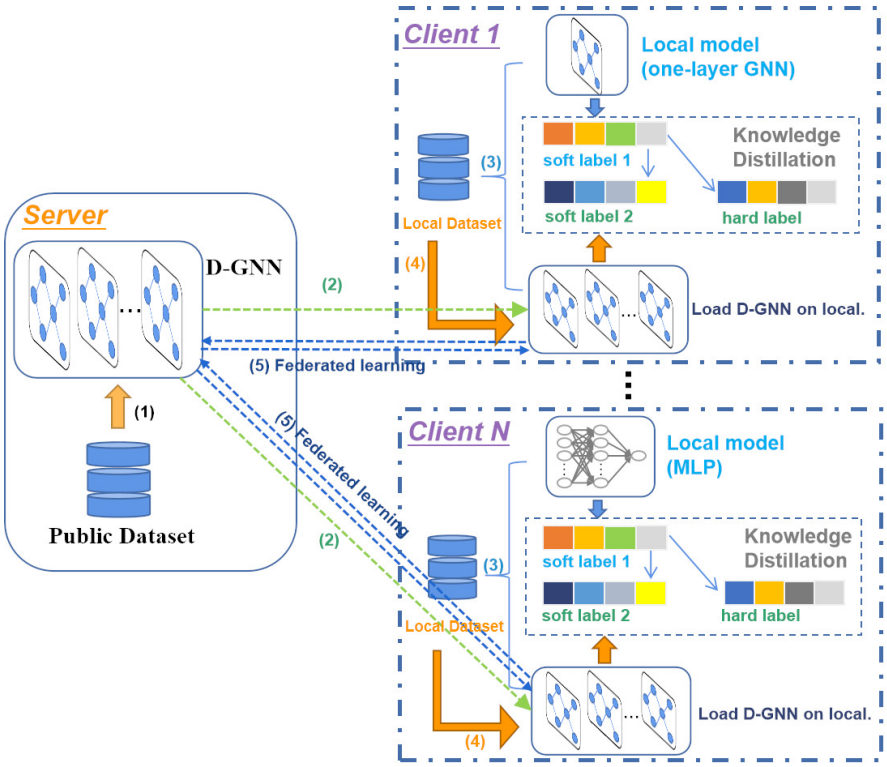


Fig. 1. The workflow of FedGraph-KD.

Not all processes are always required in the actual training. Step (1) performs only once at the initial training to obtain the D-GNN model for subsequent processes. Then repeat steps (2)-(5) until reaching the maximum training epoch.

Pre-training the D-GNN model This section introduces the specific architecture of the T-Net model, i.e., D-GNN. We select GIN [26] as the GNN model. Given the input graph $G = (V, A, X)$, the forward propagation of D-GNN is computed as follows:

$$h_v^{(l+1)} = \sigma\left(GNN\left(h_v^{(l)}, A\right)\right), \forall l \in [L] \tag{1}$$

$$h_G = READOUT(h_v, v \in V) \tag{2}$$

$$\hat{y} = Linear(h_G) \tag{3}$$

where V, A, and X denote the set of nodes, adjacency matrix, and node features, respectively. l is the number of GNN layers, σ is an activation function. GNN takes different approaches depending on different operators. $h_v^{(l)}$ is the graph node embedding vector of layer I. The initial value $h_v^{(0)} = X$, h_G is the whole graph embedding vector. *Linear* denotes the linear layer of D-GNN, i.e., MLP. $\hat{y}$ is the predicted value of D-GNN for the input graph G.

Knowledge distillation module The knowledge distillation module in FedGraph-KD takes D-GNN as T-Net and uses GNN or MLP as S-Net. It allows the local model to combine the advantages of D-GNN and MLP to accelerate the model while maintaining efficient graph representation capabilities. The overall loss L_total of knowledge distillation is calculated as follows:

$$L_total = \lambda L_D(G)\hat{,} z) + (1-\lambda)L_S(\hat{G}, y) \tag{4}$$

$$L_D(\hat{G}, z) = \sum_{i=0}^{k} -p_i(\hat{G}_i, 1)\log\left(p_i(z_i, 1)\right) \tag{5}$$

$$L_S(\hat{G}, y) = \sum_{i=0}^{k} -y_i \log\left(p_i(\hat{G}_i, 1)\right) \tag{6}$$

$$p_i(z_i, T) = \frac{\exp(z_i/T)}{\sum_j^k \exp(z_j/T)} \tag{7}$$

Where L_D, L_S are distillation loss and student loss, respectively. λ is an hyperparameter. We refer to the GLNN loss function and set the temperature in knowledge distillation $T = 1$. z is the soft label generated by T-Net for the whole input graph, and y is the ground-truth label. $\hat{G}$ is the soft label of the S-Net output.

Federated learning Module The federated learning module contains local transfer learning and global parameter aggregation. Since the datasets in pre-training do not cover the client's private data, it may lead to bias in the prediction of D-GNN on the client dataset, i.e., generalization error. To address this problem, we adopt the fine-tuning framework to fine-tune the D-GNN model using local private datasets while local client distillation training. The model fine-tuning increases the generalization capability of D-GNN to the client's private dataset, and the fine-tuned model can better guide the training of the local model during knowledge distillation.

Transfer learning allows D-GNN to understand the knowledge of each client dataset. To allow the server to collect global knowledge updates, we adopt FedAvg [17] to aggregate the parameters of each client's fine-tuned D-GNN.

4 Experiment

This section focus on validating the effectiveness of FedGraph-KD on graph classification tasks.

4.1 Implementation details

Datasets We use four real-world datasets from two domains: bioinformatics and social networks. Table 1 provides statistical information about each dataset [19].

Table 1. Statistical information about Datasets.

Dataset	Graph	Feature	Class	Average Node	Average Edge
NCI1 [19]	4110	37	2	29.87	32.30
PROTEINS [19]	1113	3	2	39.06	72.82
IMDB-BIN [19]	1000	135	2	19.77	96.53
IMDB-MUL [19]	1500	88	3	13	65.94

NCI1 and PROTEINS are bioinformatics-related datasets. 1) The NCI1 dataset is a sample of 4100 plots provided by the National Cancer Institute. Each figure represents a compound and is divided into two categories based on whether the compound prevents cancer cell growth. 2) The PROTEINS dataset contains 1113 protein samples, classified into 2 categories based on whether the protein is an enzyme or a non-enzyme.

IMDB-BIN and IMDB-MUL are social network-related datasets. 1) IMDB-BIN collects information about actor types for different movies on IMDB. This data collects movies in two categories, action genre and romance genre. 2) IMDB-MUL is a multi-category version of IMDB-BIN, containing three different types of movie datasets, comedy, romance, and sci-fi.

In our experiments, we divide datasets into a public dataset for the server and a private dataset for the client in the ratio of 3:7. The public dataset is used for pre-training the server model, and the private dataset is randomly distributed to different clients. The number of graphs for each client is set to 100-200 by dataset size. Both the pre-training and client local training datasets are divided into a training set, test set and validation set, with a ratio of 8:1:1.

Training In the pre-training stage, we use GIN to construct the D-GNN model and leverage PairNorm [32] after each layer of GNN output to prevent over-smoothing. We set hyper-parameters: $random_seed = 25$, $learning_rate = 0.01$, $activation_function = ReLu$, $weight_initialization = Xavier$,$epoch = 200$,$Dropout = 0.5$.

Table 2. Accuracy of D-GNN after pre-training. The number in front of GNN represents the number of GIN network layers.

Model	D-3GNN	D-4GNN	D-5GNN
NCI1 [19]	80.6	79.2	82.6
PROTEINS [19]	76.2	78.6	80.1
IMDB-BIN [19]	78.8	70.1	73.3
IMDB-MUL [19]	43.3	52.2	51.1

As shown in Table 2, on the bioinformatics datasets NCI1 and PROTEINS, the model accuracy gradually increases as GIN network layers increases. In contrast, on the social network datasets IMDB-BIN and IMDB-MUL, the deepest D-5GNN does not perform as well as D-3GNN and D-4GNN. This may be due

to the different strengths of association between node features and graph categories in different types of datasets. Compared with the bioinformatics dataset, the node features on the social network dataset are less correlated with the categories. Therefore, further deepening the network hierarchy beyond a certain level does not improve the performance even if the nodes can capture multi-order neighbour node information. And it performs worse in the multi-category classification task, i.e., on the IMDB-MUL dataset.

4.2 Results

FedGraph-KD Classification Performance Evaluation We analyze the performance of our approach on different graph classification datasets, and we select the following models for comparison:

(1) **No knowledge distillation local training (Local).** Each client only uses private datasets to train local models without communication with the server, e.g., parameter exchange.
(2) **No federal local knowledge distillation training (KD-Local).** Use the pre-trained model to guide the client to perform knowledge distillation.
(3) **Isomorphic model comparison with the same client model.** The GraphSAGE [10] model with the best performance in FedGraphNN [11] is taken as Fed-SAGE. Meanwhile, based on the FedGraphNN, a GIN model with the same network layers is set up as another set of control experiments for the isomorphic model, as Fed-GIN, and uses FedAvg as the aggregation algorithm.
(4) **FedMD.**

The settings of client models and the division of datasets are shown in Table 3.

Table 3. Comparison with the different models.

Dataset	T-Net	Data volume of the client	Client model
NCI1 [19]	D-5GNN	250-300	3GCN+3SAGE+4MLP
PROTEINS [19]	D-5GNN	110-150	2GCN+2SAGE+2MLP
IMDB-BIN [19]	D-3GNN	90-120	2GCN+2SAGE+2MLP
IMDB-MUL [19]	D-4GNN	120-150	3GCN+2SAGE+3MLP

As shown in Table 4, the following conclusions can be drawn:

(1) FedGraph-KD achieves optimal results on all four datasets.
(2) The client training using only local data is not practical because the client models are single-layer GNNs and MLPs, which have a poor ability for graph embedding and cannot represent the graph structure information well.
(3) In isomorphic model benchmark experiments, FedGraph-KD shows a considerable performance improvement. KD-Local performs comparably to Fed-GIN on PROTEINS and IMDB-BIN and outperforms Fed-GIN on NCI1. The performance of Fed-SAGE indicates that transfer learning performs poorly on datasets we used, while Fed-GIN can achieve good performance.

(4) FedGraph-KD has a slight improvement over FedMD because the consensus of MLP models in FedMD on the public dataset affects the global training process, while D-GNN in FedGraph-KD guides the training for each local model individually.

Table 4. Comparison with the different models.

Model	NCI1	PROTEINS	IMDB-BIN	IMDB-MUL
Local [26]	58.2±1.90	68.1±2.41	65.7±1.90	35.4±2.25
KD-Local	70.2±0.74	71.9±1.09	72.3±0.97	41.1±2.41
Fed-SAGE [10]	55.7±2.34	51.4±0.65	61.0±1.39	32.6±0.12
Fed-GIN	63.3±0.22	69.4±1.18	71.0±0.47	44.3±1.09
FedMD [15]	71.46±1.32	70.9±2.24	73.68±0.39	45.88±1.25
FedGraph-KD	**81.3±0.76**	**77.1±0.6**	**78.6±0.9**	**50.4±0.96**

Knowledge distillation effectiveness analysis In this section, we explore the role of knowledge distillation in FedGraph-KD through separate experiments.

We verify the effectiveness from two perspectives: graph-less knowledge distillation and graph knowledge distillation. Federated learning is not set up. We train the T-Net (D-GNN) with the dataset first and then use T-Net to instruct training of the S-Net (single-layer GNN or MLP). We obtain the classification accuracy of the S-Net.

Table 5. Results for graph classification. We show classification accuracies on the NCI1 and IMDB-BIN datasets. Results are averaged over 10 experiments. KD indicates that the model with knowledge distillation. GNNs are single-layer, and MLPs are two-layer.

Dataset	SAGE	GCN	MLP	KD-SAGE	KD-GCN	KD-MLP	D-GNN
NCI1 [19]	73.2	71.3	64.2	78.5	79.1	74.2	82.1
IMDB-BIN [19]	67.2	70.9	59.5	75.3	74.9	71.1	77.3

Table 5 shows that knowledge distillation can effectively improve the model's classification accuracy. Compared to those without knowledge distillation, MLP, GCN and GASE with knowledge distillation improve by 15.6%, 10.9% and 7%, respectively, and the improvement is most apparent in MLP.

5 Conclusion

To address the heterogeneity of each client's model in graph classification scenarios, we proposed FedGraph-KD. Specifically, it used a pre-trained model as T-Net, one complex GNN model, and transfered its knowledge with a distributed federated learning paradigm to simple models(S-Net). Experiments demonstrated the effectiveness of our approach. In the future, we can use feature-based knowledge distillation to improve FedGraph-KD to solve the generation gap in feature representation capability between T-Net and S-Net.

Acknowledgements This work was partially supported by the National Natural Science Foundation of China under Grant No. U20A20182 and 62177019.

References

1. Battaglia, P.W., Hamrick, J.B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al.: Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261 (2018)
2. Chen, Y., Zhou, L., Pei, S., Yu, Z., Chen, Y., Liu, X., Du, J., Xiong, N.: Knn-block dbscan: Fast clustering for large-scale data. IEEE transactions on systems, man, and cybernetics: systems **51**(6), 3939–3953 (2019)
3. Cheng, H., Xie, Z., Shi, Y., Xiong, N.: Multi-step data prediction in wireless sensor networks based on one-dimensional cnn and bidirectional lstm. IEEE Access **7**, 117883–117896 (2019)
4. Fu, A., Zhang, X., Xiong, N., Gao, Y., Wang, H., Zhang, J.: Vfl: a verifiable federated learning with privacy-preserving for big data in industrial iot. IEEE Transactions on Industrial Informatics (2020)
5. Gai, K., Qiu, M., Ming, Z., Zhao, H., Qiu, L.: Spoofing-jamming attack strategy using optimal power distributions in wireless smart grid networks. IEEE Transactions on Smart Grid **8**(5), 2431–2439 (2017)
6. Gai, K., Qiu, M., Xiong, Z., Liu, M.: Privacy-preserving multi-channel communication in edge-of-things. Future Generation Computer Systems **85**, 190–200 (2018)
7. Gai, K., Qiu, M., Zhao, H., Tao, L., Zong, Z.: Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing. Journal of network and computer applications **59**, 46–54 (2016)
8. Gai, K., Wu, Y., Zhu, L., Zhang, Z., Qiu, M.: Differential privacy-based blockchain for industrial internet-of-things. IEEE Transactions on Industrial Informatics **16**(6), 4156–4165 (2019)
9. Gao, Y., Xiang, X., Xiong, N., Huang, B., Lee, H.J., Alrifai, R., Jiang, X., Fang, Z.: Human action monitoring for healthcare based on deep learning. Ieee Access **6**, 52277–52285 (2018)
10. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Advances in neural information processing systems **30** (2017)
11. He, C., Balasubramanian, K., Ceyani, E., Yang, C., Xie, H., Sun, L., He, L., Yang, L., Yu, P.S., Rong, Y., et al.: Fedgraphnn: A federated learning system and benchmark for graph neural networks. arXiv preprint arXiv:2104.07145 (2021)
12. Huang, S., Zeng, Z., Ota, K., Dong, M., Wang, T., Xiong, N.N.: An intelligent collaboration trust interconnections system for mobile information control in ubiquitous 5g networks. IEEE transactions on network science and engineering **8**(1), 347–365 (2020)
13. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
14. Kumar, P., Kumar, R., Srivastava, G., Gupta, G.P., Tripathi, R., Gadekallu, T.R., Xiong, N.N.: Ppsf: a privacy-preserving and secure framework using blockchain-based machine-learning for iot-driven smart cities. IEEE Transactions on Network Science and Engineering **8**(3), 2326–2341 (2021)
15. Li, D., Wang, J.: Fedmd: Heterogenous federated learning via model distillation. arXiv preprint arXiv:1910.03581 (2019)

16. Liu, R., Yu, H.: Federated graph neural networks: Overview, techniques and challenges. arXiv preprint arXiv:2202.07256 (2022)
17. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. PMLR (2017)
18. Qiu, H., Qiu, M., Memmi, G., Ming, Z., Liu, M.: A dynamic scalable blockchain based communication architecture for iot. In: International Conference on Smart Blockchain. pp. 159–166. Springer (2018)
19. Rossi, R.A., Ahmed, N.K.: The network data repository with interactive graph analytics and visualization. In: AAAI (2015), https://networkrepository.com
20. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015)
21. Shen, T., Zhang, J., Jia, X., Zhang, F., Huang, G., Zhou, P., Kuang, K., Wu, F., Wu, C.: Federated mutual learning. arXiv preprint arXiv:2006.16765 (2020)
22. Tian, Z., Li, M., Qiu, M., Sun, Y., Su, S.: Block-def: A secure digital evidence framework using blockchain. Information Sciences **491**, 151–165 (2019)
23. Wu, C., Ju, B., Wu, Y., Lin, X., Xiong, N., Xu, G., Li, H., Liang, X.: Uav autonomous target search based on deep reinforcement learning in complex disaster scene. IEEE Access **7**, 117227–117245 (2019)
24. Wu, C., Luo, C., Xiong, N., Zhang, W., Kim, T.H.: A greedy deep learning method for medical disease analysis. IEEE Access **6**, 20021–20030 (2018)
25. Xia, F., Hao, R., Li, J., Xiong, N., Yang, L.T., Zhang, Y.: Adaptive gts allocation in ieee 802.15. 4 for real-time wireless sensor networks. Journal of Systems Architecture **59**(10), 1231–1242 (2013)
26. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)
27. Yao, Y., Xiong, N., Park, J.H., Ma, L., Liu, J.: Privacy-preserving max/min query in two-tiered wireless sensor networks. Computers & Mathematics with Applications **65**(9), 1318–1325 (2013)
28. Zhang, S., Liu, Y., Sun, Y., Shah, N.: Graph-less neural networks: Teaching old mlps new tricks via distillation. arXiv preprint arXiv:2110.08727 (2021)
29. Zhang, W., Zhu, S., Tang, J., Xiong, N.: A novel trust management scheme based on dempster–shafer evidence theory for malicious nodes detection in wireless sensor networks. The Journal of Supercomputing **74**(4), 1779–1801 (2018)
30. Zhang, Y., Xiang, T., Hospedales, T.M., Lu, H.: Deep mutual learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4320–4328 (2018)
31. Zhao, J., Huang, J., Xiong, N.: An effective exponential-based trust and reputation evaluation system in wireless sensor networks. IEEE Access **7**, 33859–33869 (2019)
32. Zhao, L., Akoglu, L.: Pairnorm: Tackling oversmoothing in gnns. arXiv preprint arXiv:1909.12223 (2019)
33. Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M.: Graph neural networks: A review of methods and applications. AI Open **1**, 57–81 (2020)
34. Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., He, Q.: A comprehensive survey on transfer learning. Proceedings of the IEEE **109**(1), 43–76 (2020)

Heterogeneous System Data Storage and Retrieval Scheme Based on Blockchain

Ni Zhang[1], BaoQuan Ma[1(✉)], Peng Wang[1], XuHua Lei[1,2], YeJian Cheng[1,2],

JiaXin Li[1], XiaoYong Huai[1], ZhiWei Shen[1], NingNing Song[1] and Long Wang[1]

[1] National Computer System Engineering Research Institute of China, Beijing 102200, China
15501210877@163.com
[2] School of Computer Science and Technology, Xidian University, Xi'an 710071, China

Abstract. In the field of information interaction, when a project involves a large amount of heterogeneous information, it is difficult to transmit and update the required information timely, accurately, reliably and securely in such a complex environment to maintain synchronization. At present, blockchain itself has problems such as high storage pressure of nodes, low access efficiency, and simple query. Therefore, this paper takes this as a starting point and proposes a data mapping method of physical resources based on node attributes and heterogeneous nodes, which provides a general method for the data mapping from actual physical resources to information domain. This method can define the attributes of heterogeneous physical information nodes and support unified expression of various physical information resources on the same platform in the real physical world, then connect heterogeneous physical information nodes and improve the sharing ability of data resources between nodes. When accessing data, the corresponding content can be found in the off-chain database by obtaining the index information of the off-chain location stored on the chain. This method takes advantage of the large space and high access efficiency of the off-chain storage system to share the pressure of on-chain data storage. This paper expounds the design idea of the system, introduces the design objective and method in detail, gives the flow chart of the system operation, and carries out a simple software test and verification. Finally, this paper summarizes the work and prospects the development direction of future work, hoping to provide inspiration for solving such problems.

Keywords: Blockchain, Data Query, Heterogeneous Information，Data Storage, Retrieval Scheme

1 Introduction

Cyber-Physical Systems (CPS) is a complex system with new capabilities, where computing [1-3], physical elements [4-6] and network environment [7-9]are tightly coupled. With the integration of CPS and technologies such as 5G and blockchain, the security problems [10-12] of CPS have been exposed. In 2017, Ouaddahet proposed a

perceptive security framework for CPS, which laid the foundation for the security mechanism of CPS [10-12], but the different security requirements of each layer brought complexity to the solution. Meanwhile, the growing scale of CPS also brought hidden dangers to the confidentiality and integrity [13]. System framework as Fig.1 shown.

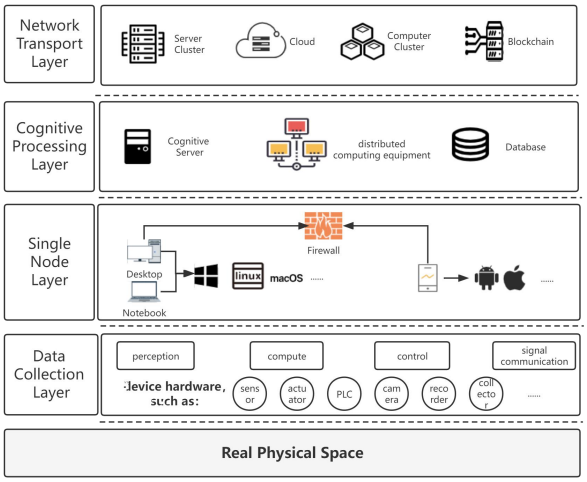


Fig. 1. System Framework

Blockchain-related technologies have brought new solutions to this problem. Blockchain has been widely applied in many fields due to its characteristics of decentralization, immutable, whole-process retention, traceable, collective maintenance, openness and transparency [14], from the initial Bitcoin project to the present application in how to provide abundant query functions to massive applications. At present, many scholars have proved through research and testing that blockchain technologies can make the system have a higher level of security and better privacy protection ability [15].

Due to the strong real-time, dynamic and massive nature of these resources [16-17], it brings a huge challenge to the query [18]. How to effectively integrate heterogeneous platforms and heterogeneous data, complete the association between human society and the physical resource information world, build a unified information system, manage data, build a storage system, and set up an efficient query mechanism [19] are important issues that need to be solved at present. Therefore, this paper proposes a method of resource index construction and query in CPS to solve the query problem in resource management under CPS environment [20].

The structure of the paper is as follows: Sect. 2 the Blockchain Architecture Design for Task Management Business; Sect. 3 the Software Design; Sect. 4 Work Summarization.

2 Related Work

Jinshan Shi et al. [21] proposed an IoT access control framework based on blockchain. Siyuan Wang et al. [22] proposed the power token ring network based on blockchain to solve the unauthorized access of the Internet of Things [23,24]. Guanjie Cheng et al. [25] proposed a data management architecture of IoT based on blockchain and edge computing to achieve data security management. However, they assume that the edge nodes are safe and reliable, ignore that the edge nodes should be confirmed by the trusted mechanism when accessing the IoT under the expansion of the scale of the IoT, and also ignore the computing resources brought by the edge nodes. That is, the role of the edge nodes in the process of access control and storage query is not considered under the current new paradigm of the Internet of Things system integrating edge computing.

3 Preliminaries

3.1 Design goals

1. Aiming at the problem of heterogeneous physical resource mapping, this paper proposes a mapping method based on node attributes and node data, which provides a general method for actual physical resources to be mapped into the information domain. This method can achieve the definition of attributes and characteristics of heterogeneous physical information nodes and nodes access, support the unified representation of various physical information resources in the real physical world on the same platform, and further improve the ability of data resource sharing among nodes.

2. This paper proposes a storage scheme that combines the information of the on-chain index table with the off-chain database. The blockchain data storage problem is solved by the sharing mode combining the on-chain and off-chain storage of data blockchain, where the original data is stored off-chain and the data description and data sharing log are stored on-chain. The data attributes are graded according to different sensitivities to meet the requirement of flexible data sharing.

3. Aiming at the complex data information generated by various heterogeneous resources, this paper proposes an index construction method based on abstract. This method supports fast query based on attributes through index query processing method of classification. It uses the on-chain and off-chain collaborative way to share data to release a large amount of space on the blockchain, ensure the efficient transmission of data on the chain, and improve the efficiency of information query and sharing.

3.2 Blockchain technology

Blockchain provides a decentralized distributed data system [26]. The trusted or semi-trusted nodes of the participating system jointly maintain a growing chain through a consensus mechanism, which eliminates the need for centralized control

and uses cryptography mechanisms such as Hash, digital certificates, and signatures to ensure that records cannot be forged or destroyed. In a distributed storage management system based on blockchain, data requesters must obtain data access permission from the blockchain before accessing specific data. The system uses the aggregation of data access rights as an incentive mechanism in the blockchain to encourage institutions to participate in building the blockchain.

Blockchain technology can improve the trust between the sharing parties, which has gradually become a consensus in the current data sharing mode. Leveraging blockchain's de-neutralization and immutable nature, data providers can use it as a tool for logging data usage. At present, most applications based on blockchain use the characteristics of it to achieve functions such as trusted depository and traceability query. In the above applications, blockchain system can be regarded as a new type of secure and trusted distributed database system. However, blockchain itself has shortcomings in storage and query, such as low storage efficiency, huge storage cost, slow query speed and single query function on the chain, etc. These problems have been restricting the development of blockchain, and become the bottleneck of blockchain application landing. Although scholars at home and abroad have done a lot of research to solve above questions, most of them only analyze the similarities and differences of data management technology between traditional database and blockchain database [27].

4 Heterogeneous Data Storage and Retrieval Method Based on Blockchain

4.1 Procedure of Operation

1) Data generation: a large amount of data will be generated in production activities. Upon the consent of the organization, the institutions will collect, clean and process these data.

2) Data storage: The institutions grades the data according to data grading criteria, encrypts and stores the data using the algorithm proposed in this paper [28].

3) Data registration: Institutions will upload data information (including data description, encryption key, institution information, relevant individuals, access control policies and data addresses, etc.) to the data sharing platform and store the data information on the blockchain [29].

4) Key application: Users can apply for keys to the data sharing platform according to their own attributes. After authenticating their identity, attribute authorization agencies can generate corresponding keys using the key generation method and send them to users.

5) Data request: the users retrieve data on the shared platform and send a data request to the platform. After receiving the request, the platform returns data information to users.

6) Query: Participants can query data usage by sending requests to the platform, as Fig. 2 shown.

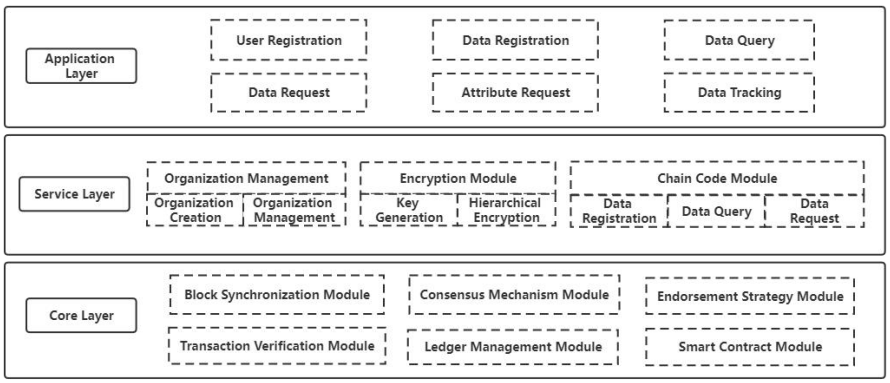


Fig. 2. System Framework

4.2 Storage methods

This scheme combines the on-chain index table information with the off-chain database for storage. On the one hand, it can release a large amount of space on the blockchain. On the other hand, it can improve the efficiency of information sharing. The index table on the chain stores the index information (index category and the address of the encrypted file) and forms the index block and stores it on the blockchain. The index table corresponds the information category queried by the queriers to the storage address value and occupies a small portion of memory on the chain. The off-chain database stores the encrypted data files uploaded by the data owner to ensure the security of the data. The data storage process as fig3 shown.

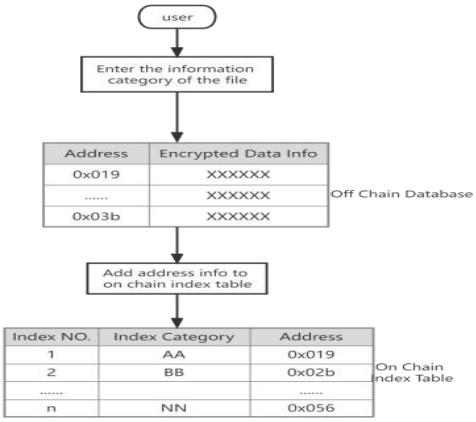


Fig. 3. Storage process

4.3 Index build

The composition structure of data attributes is complex and diverse, and the access control policies should be different for data from different sources and with different levels of sensitivity. Aiming at the high flexibility privacy protection of data, the data

is divided into the following three levels: high sensitivity, medium sensitivity and low sensitivity. Different access control policies can be implemented for different levels of data. The specific data grading criteria are as follows: (1) Highly sensitive data: information that can be identified or that will cause significant impact if exposed. (2) Medium sensitive data: the information that cannot identify personal identity and still has important significance after being blurred can retain the fuzzy result. (3) Low sensitive data: other non-important information. At the same time, in order to realize data access control, data users need to be divided according to their identity information, including user type and user permission level. The attribute and relational data structures are shown in Fig. 4 and Fig. 5.

Filed	Description
key	Index Key，whose value is the full hash value of the data digest.
preVersion	Pointer Field，whose value is the address of data node corresponding to the previous version of this data digest.
UserType	Value，whose value is the data user type.
UserLevel	Value，whose value is the professional level of the data user.
AbAddr	Value，whose value is the data digest address of the data digest hash.

Fig. 4 Attribute data structure definition

Field	Description
key	Index Key，whose value is the public prefix of the data digest hash.
next	Pointer Array

Fig. 5 Relational data structure definition

Construct data abstract, index. According to the established field extraction rules, the data abstract is constructed for the original data on the chain, and the abstract index is constructed for each data abstract to be put on the chain, such as the abstract dictionary tree index. Abstract dictionary tree is a resource level node structure definition which is constructed by using the hash values of the on-chain data abstracts distributed in different blocks as the key values. The hash values of the on-chain data abstracts are calculated, and the hash values of all data abstracts are used as key values to build a dictionary tree, so as to build a centralized index for all data abstracts in the blockchain, thus accelerating the query efficiency of data abstracts..

Hybrid index building method. In the process of tracing query based on blockchain, a hybrid index structure based on Merkle tree is proposed for specific multiple information query. The hybrid index structure adopts the basic structure of Merkle tree, combined with Merkle index structure, and introduces data structures such as hash table into Merkle tree to improve the efficiency of transaction query. It supports for member existence queries, that is, whether the collection contains the element. During the construction of Merkle tree nodes, the corresponding transaction keyword information of the node is stored in each node. When querying the transaction information corresponding to the keyword key, this paper will first calculate from the root node of the Merkle tree. If the key exists in Bloom Filter (BF) node, it will successively judge whether the queried keyword exists in BF of the subtrees around the node. If so, the query will continue; otherwise, *none* will be

returned. Multiple pruning operations can be performed to improve the search efficiency during searching and traversing the Merkle tree. For the data that does not exist in the blockchain, the previous block is directly searched because there is no index stored on the key in the root node Bloom filter. In the block header, this paper introduces a Hash Table. The hybrid index structure records the transaction information corresponding to the ID in the Hash Table and stores it in the leaf node of the Merkle tree, so that the leaf node index where the transaction information is located can be quickly located according to the ID in the tracing process.

Based on the hybrid index structure of Merkle tree, the algorithm is constructed as follows: ① Input transaction set is sorted according to the numerical attributes; ② After the maximum and minimum attributes are obtained, they are placed in the block header; Traverse the transactions, Hash the transaction information and place it on the leaf node of the Merkle tree. Take out the ID of the transaction information and the corresponding location index stored in the leaf node to construct the Hash Table and store it in the block header. ③ Hash the Hash value of leaf nodes in pairs, and calculate the keywords in the transaction information of corresponding nodes in the calculation process. The Bloom filter (BF) is stored in the node, and then the Merkle tree is constructed. The Merkle root Hash is placed in the block header, and the hybrid index structure based on the Merkle tree is constructed. The output hMerkleTree is the blockchain structure, and the hybrid index structure is shown in Fig 6.

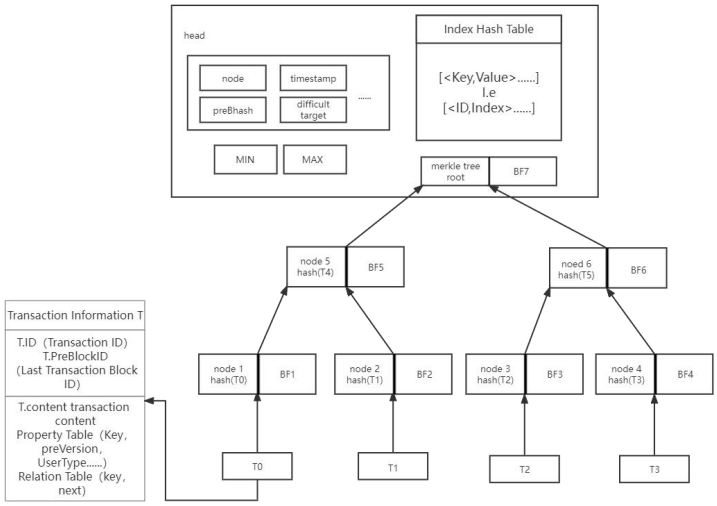


Fig. 6 Relational data structure definition

4.4 Query methods

In the algorithm, as long as the input tracing source code can be in the alliance chain to take out the corresponding transaction address, and then take out the specific value from the database according to the address. In line 1, from the newly generated block to the Genesis block, the information of the products stored in the block from production to sales can be obtained, so there may be multiple transactions in one

tracing source. In line 2, judge whether proCode is in the hash value of Merkle root. If not, directly search the next block; if so, enter the Merkle tree for search. In lines 2 to 12, the hash values of the left and right child nodes are used to determine which subtree the transaction exists in, and then several pruning operations are carried out until back to the source code to find the corresponding designated trading under the chain in the address database, and then root address queries specific transaction from the database under the chain, and adds the transaction to the list. After traversing over all blocks, return the transaction list.

Algorithm : On-chain and Off-chain Collaborative Query Algorithm
Input: Data index identification : proCode
Output: A collection of transactions that meet the requirements:$T^p=\{t_1^p.data, t_2^p.data, ..., t_m^p.data\}$

```
1:  for block in blockchains do
2:      if proCode in block.Merkle.root.bloom then
3:          while root.leftchild  root.rightchild  do
4:              if proCode in root.leftchild.bloom then
5:                  root = root.leftchild;
6:              elseif proCode in root.rightchild.bloom
7:                  root = root.rightchild;
8:              end if
9:          end while
10:         Address=root.transaction;
11:     end if
12:     if Address in CouchDB then
13:         T^p.add.(data);
14:     else
15:         Address is err!
16:     end if
17: end for
18: return T^p={t_1^p.data, t_2^p.data, ..., t_m^p.data}
```

Algorithm. 1. Query methods

4.5 System environment

This scheme uses GO SDK to invoke Hyperledger Fabric framework, and the database is based on CouchDB library. All nodes run on the same physical server and use Docker virtualization technology during deployment. The experimental environment is configured as follows:

1) Operating system: Ubuntu18.04, 64-bit;

2) CPU and memory: 4 cores (vCPU) 16 GiB.

In this paper, Hyperledger Fabric V2.3 licensed blockchain is used to implement data storage and retrieval testing. Network nodes include each node of blockchain and each node of attribute authority. Among them, the blockchain node, as the underlying core of the system, provides support for storage and retrieval testing; Attribute authority node provides encryption and decryption support for access control algorithm. There are two organizations in the blockchain network, namely, Institution Org and Requester Org, where the institution organization corresponds to the data

collector in the data sharing, and the requester organization corresponds to the individual and the consumer in the data sharing.

5 Conclusion

In this paper, we propose a mapping method based on node attributes and node data to solve the heterogeneous physical resource mapping problem, which provides a general method for mapping actual physical resources to the information domain. On this basis, a blockchain-oriented storage, index construction and efficient query method is proposed. This method can realize the definition of attributes and characteristics of heterogeneous physical information nodes and node access, and support the unified expression of various physical information resources in the real physical world on the same platform. A storage scheme combining the on-chain index table information with the off-chain database is adopted, in which the original data is stored under the chain, and the data description and data sharing log are stored on the chain. The attributes of data are graded according to different sensitivities to meet the flexibility of data sharing. Aiming at the complex data information generated by various heterogeneous resources, an index query processing method based on attribute classification is proposed, which supports the fast query based on attribute. This scheme adopts the on-chain and off-chain collaborative method to share data, which can release a large amount of space on the blockchain and ensure the efficient transmission of data on the chain. While improving data security and removing information barriers, it also greatly improves the query efficiency. However, the method proposed in this paper has slightly increased spatial complexity, and the current society has increasingly higher requirements for data privacy sharing. Therefore, in the future, on the basis of ensuring storage and query efficiency, the storage burden of the system will be reduced as much as possible, and further in-depth research will be made on how to share traceable data securely on the blockchain.

References

1. M. Qiu, Z. Jia, et al., "Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP", J. of Signal Proc. Systems, 2007
2. M. Qiu, L. Yang, et al., "Dynamic and leakage energy minimization with soft real-time loop scheduling and voltage assignment", IEEE TVLSI, 18 (3), 501-504, 2009
3. M. Qiu, C. Xue, Z. Shao, E. Sha, “Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems,” IEEE DATE Conf., 1-6, 2007
4. F. Hu, S. Lakdawala, et al., Low-power, intelligent sensor hardware interface for medical data preprocessing, IEEE Trans. on Info. Tech. in Biomedicine 13 (4), 656-663, 2009
5. M. Qiu, H. Li, E. Sha, "Heterogeneous real-time embedded software optimization considering hardware platform", ACM sym. on Applied Comp., 1637-1641, 2009
6. H. Qiu, Q. Zheng, et al., "Topological graph convolutional network-based urban traffic flow and density prediction", IEEE Trans. on ITS, 2020
7. M. Qiu, Z. Chen, et al., "Energy-aware data allocation with hybrid memory for mobile cloud systems", IEEE Systems J., 11 (2), 813-822, 2014

8. J. Niu, Y. Gao, M. Qiu, Z. Ming, "Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability", JPDC, 72(12), 1565-1575, 2012
9. M. Qiu, C Xue, Z Shao, et al., "Efficient algorithm of energy minimization for heterogeneous wireless sensor network", IEEE EUC, 25-34, 2006
10. Y. Li, K. Gai, et al., "Intercrossed access controls for secure financial services on multimedia big data in cloud systems", ACM TMCCA, 2016
11. H. Qiu, T. Dong, et al. "Adversarial attacks against network intrusion detection in IoT systems," IEEE Internet of Things Journal 8(13), 10327-10335, 2020
12. K. Gai, M. Qiu, S. Elnagdy, "A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance," IEEE BigDataSecurity 2016
13. Han Qiu, Meikang Qiu, Gerard Memmi, Zhong Ming, Meiqin Liu, A Dynamic Scalable Blockchain Based Communication Architecture for IoT[C]//Smart blockchain.2018
14. Gai K., Wu Y., Zhu L., et al., Differential Privacy-Based Blockchain for Industrial Internet-of-Things[J]. IEEE Transactions on Industrial Informatics, 2020, 16(6):4156-4165.
15. Li Y., Gai K., Qiu L., et al. Intelligent cryptography approach for secure distributed big data storage in cloud computing[J]. Information Sciences, 2017, 387: 103-115.
16. J. Li, Z. Ming, et al., "Resource allocation robustness in multi-core embedded systems with inaccurate information", Journal of Systems Arch., 57 (9), 840-849, 2011
17. M. Qiu and H. Qiu, "Review on Image Processing Based Adversarial Example Defenses in Computer Vision," IEEE 6th BigDataSecurity, pp. 94-99, Baltimore, MD, USA, 2020.
18. Tian Z. , Li M. , Qiu M., et al. Block-DEF: A Secure Digital Evidence Framework using Blockchain[J]. Information Sciences, 2019.
19. Tang S, Du X, Lu Z, et al. Coordinate-based efficient indexing mechanism for intelligent IoT systems in heterogeneous edge computing[J]. JPDC, 2022, 166: 45-56.
20. Du X, Tang S, et al. A novel data placement strategy for data-sharing scientific workflows in heterogeneous edge-cloud computing environments, IEEE ICWS, 2020: 498-507.
21. Shi Jinshan, Li Ru, Song Tingting, Internet of Things access control framework based on blockchain [J]. Journal of Computer Applications,2020,40(04):931-941.
22. Wang Siyuan, Zou Shihong. Access control mechanism based on block chain and power in Multi-domain Internet of Things [J]. Chinese J. of Applied Science,2021,39(01):55-69.
23. M. Qiu, H. Qiu, et al., "Secure Data Sharing Through Untrusted Clouds with Blockchain-enabled Key Management", the 3rd SmartBlock, pp. 11-16, Oct. 2020, Zhengzhou, China.
24. K. Gai, Y. Zhang, et al. "Blockchain-enabled Service Optimizations in Supply Chain Digital Twin", IEEE Transactions on Service Computing, 2022
25. Cheng Guanjie, Huang Zhengjie, Deng Shuiguang. Internet of Things Data Management based on blockchain and Edge computing [J]. Journal of Internet of Things,2020,4(02):1-9.
26. XinG X, ChEn Y, Li T, et al. A blockchain index structure based on subchain query [J]. Journal of Cloud Computing, 2021, 10(1): 1-11.
27. Hegde P, Streit R, et al. Achieving Almost All Blockchain Functionalities with Polylogarithmic Storage. Int'l Conf. on Financial Crypto. and Data Secu., Springer, 2022: 642-660.
28. Validi A, Kashansky V, Khiari J, et al. Hybrid On/Off Blockchain Approach for Vehicle Data Management, Processing and Visualization Exemplified by the ADAPT Platform[J]. arXiv preprint arXiv:2208.06432, 2022.
29. Sagirlar G, Sheehan J D, Ragnoli E. On the design of co-operating blockchains for IoT[C]//3rd IEEE Int'l Conf. on Info. and Computer Tech. (ICICT), 2020: 548-552.

Fibereum: A Novel Distributed Ledger Technology System

Dylan Yu[1] Ethan Yang[2] Alissa Shen[3]

Dan Tamir[4] Naphtali Rishe[5]

[1]Dulles High School, Sugar Land, Texas, USA, dylanyu66@gmail.com

[2]Westwood High School, Austin, Texas, USA, kempwood88@gmail.com

[3]St. Stephen's Episcopal School, Austin, Texas, USA, shenalissa@gmail.com

[4]Texas State University, San Marcos, Texas, USA, dt19@txstate.edu

[5]Florida International University, Miami, FL, USA, ndr@acm.org

Abstract. Over the past several years, due to the progression toward data-driven scientific disciplines, the field of Big Data has gained significant importance. These developments pose certain challenges in the area of efficient, effective, and secure management and transmission of digital information. This paper presents and evaluates a novel Distributed Ledger Technology (DLT) system, Fibereum, in a variety of use-cases, including a DLT-based system for Big Data exchange, as well as the fungible and non-fungible exchange of artwork, goods, commodities, and digital currency. Fibereum's innovations include the application of non-linear data structures and a new concept of Lazy Verification. We demonstrate the benefits of these novel features for DLT system applications' cost performance and their added resilience towards cyber-attacks via the consideration of several use cases.

Keywords: Distributed Ledger Technology, Blockchain, Bitcoin, Ethereum, Hyper Ledger, Consensus Verification, Cybersecurity, Proof of Work, Proof of Elapsed Time

1 Introduction

This paper presents Fibereum, a Distributed Ledger Technology (DLT) system that applies novel methods to address several issues with current Blockchain [1-5] data structures and storage mechanisms, including deficiencies with respect to defending against blockchain attacks. A typical implementation of conventional DLT systems is via Blockchain. That is, conventional DLT systems, such as the permissionless Bitcoin [3] and Ethereum [4], as well as the permission-based Hyper Ledger Technology [6], use

a linear data structure to manage blocks of data. Furthermore, conventional DLT systems often require a complex time- and energy-consuming process for the verification of the DLT system integrity. The DLT system presented here, Fibereum, utilizes novel methods for Big Data exchange, as well as for the fungible and non-fungible exchange of artwork, goods, commodities, and digital currency.

The novel features of the proposed DLT system include: **(i)** the enablement of the use of non-linear data structures-based systems; **(ii)** employing a procedure of lazy verification, where the verification of the DLT system integrity is delayed indefinitely and applied only on a need-to-do basis; and **(iii)** the enablement of permissionless as well as permission-based implementations. The use of non-linear data structures for the storage of transactions in blocks and storage of blocks within the DLT system enhances the efficiency of the overall system. For example, in one implementation, the proposed DLT system can use cryptographic trees for intra-block and inter-block management in conjunction with lazy verification. This approach significantly improves the management and security of Big Data and other types of digital data-driven systems. Furthermore, the use of lazy verification along with non-linear data structures, as well as the utilization of a time/energy consumption-efficient consensus mechanism, can provide a significant saving in energy consumption.

Fibereum introduces several innovative modifications to Blockchain technology, providing a more general framework for DLT systems. The modifications extend the utility of DLT systems to several new applications, including business-to-business data governance and data exchange. The Fibereum DLT system has various applications in the fields of Big Data, including Healthcare, Transportation, Smart Cities, the Internet of Things, and process control. The Fibereum DLT system is also suitable for applications such as digital currency, smart contracts, licensing [7], inventory management [8], supply chain management [9], counterfeit detection [10], and the exchange of copyrighted material, e.g., Non-Fungible Tokens [11].

We have developed and implemented an event-based simulation for Fibereum use cases. The simulation and theoretical analysis show that, due to the option to use nonliner data structures and the concept of lazy verification, in most cases Fibereum computational complexity is lower than other DLT systems such as the Bitcoin Blockchain.

The rest of this paper is organized as follows. Section 2 provides background information and definitions. Section 3 includes a literature review. Section 4 presents the main features of the Fibereum DLT system. Section 5 presents several Fibereum use-cases, and Section 6 includes a conclusion and directions for further research.

2 Background and Definitions

2.1 Definitions

A blockchain is a peer-to-peer network that stores transactions between multiple parties, organized as a cryptographic linked list – i.e., a chain of nodes. Blockchains attempt to guarantee decentralization, transparency, and immutability [1, 2].

A **Merkle tree** is a tree in which all the leaves contain the cryptographic hash of a block of data, potentially along with the data, and every non-leaf node contains the hash of its child nodes' data [12]. The notion of the basic Merkle tree can be extended to Merkle Heaps (Min and Max heaps) [5], Merkle binary search trees [5], Merkle Hash tables [5], and Merkle cyclic and acyclic graphs [5].

A **permissionless** DLT system is open to the public. Any user can create or access data or smart contracts in the DLT system, and all the transactions made on the DLT system are displayed to all the users, making the permissionless DLT system completely transparent [1-3]. In general, blocks are mined by users, referred to as miners, onto the ledger, in exchange for incentives for the miners [1-3]. Furthermore, users may be engaged in establishing the DLT system's integrity (potentially with verification incentives) [1-3]. For specific use-cases, Fibereum offers a permissionless version of a DLT system, where the system construction is extremely simple and does not require significant incentives for miners. Additionally, Fibereum offers an alternative approach for verifying the integrity and incentivizing the verification process. This approach is referred to as 'Lazy Verification.' In this case, initially, the DLT system is in a "verifiable" state. The verification and its incentives are enacted only on a "need to do" basis.

Lazy verification is a form of consensus term-setting, first introduced in the context of Fibereum, where the consensus verification process (as well as verification incentives) is/are delayed as much as possible and only performed when an immediate urgent need, e.g., taking care of an exception, arises. While continuous and prompt verification, which may require ample incentives, computational resources, and a high amount of energy consumption, is mandatory in certain use-cases and applications (e.g., digital currency), lazy verification allows for an efficient method of verifying transactions in a DLT system as it removes the unnecessary steps of verifying every block before an exception has occurred. The verification algorithm may apply the same concept as the standard blockchains' verification procedures, such as the Proof-of-Work (PoW)-based Byzantine consensus [3, 13], but with more scalability for Big Data when there are large amounts of data entering the DLT system at a high rate.

A **permission-based** DLT system is a private network where only certain users are authorized to access the DLT system. The network users are identifiable and complete anonymity is not possible [14, 15]. Hence, access control and encryption may be implemented as a part of the permission mechanism [14]. **Hyperledger Fabric Technology** (HFT) is the most commonly used framework for permission-based blockchains [6]. Fibereum offers a permission-based version of a DLT system.

A **Consensus Algorithm** replaces a centralized authority to preserve the security and fault tolerance of a DLT system. Two consensus algorithms, Proof-of-Work (PoW) [3, 14] and Proof-of-Elapsed-Time (PoET) [15] are most relevant for the Fibereum use-cases. Other commonly used consensus algorithms include Proof-of-Inclusion [16] and Proof-of-Stake [17]. The Practical Byzantine Fault Tolerance is often used as a part of consensus algorithms [13]. We elaborate on the PoET mechanism, which is less known to many DLT system practitioners and is advantageous in terms of cost performance over other consensus mechanisms, especially with respect to some Fibereum use-cases.

PoET is a consensus algorithm in which all nodes "sleep" for an arbitrary amount of time, with the first node to wake up receiving authorization/rewards for verification

and mining. PoET is more energy efficient and less resource costly than PoW. Nevertheless, this algorithm must resolve "collisions" in a way that may be similar to the collision detection, avoidance, and resolution of the Carrier Sense, Multiple Access, with Collision Detection (CSMACD) procedures that govern many of the commonly used communication protocols [18, 19].

Notably, Fibereum's use of lazy verification, along with providing the option to use non-linear data structures as well as time/energy consumption efficient DLT system construction and consensus verification mechanisms, can provide improved resilience against attacks, as well as a significant saving in operational costs.

2.2 Resilience and Security

In this sub-section, we list several of the common attacks applied to existing blockchain DLT systems and related security concerns. In Section 4, we will refer to these items in the context of Fibereum.

Some of the common attacks on the Bitcoin Blockchain are Eclipse/Sybil attacks [20] and double-spending attacks [21]. Among other attack types are the Vector76 attack [22], the Blockchain reorganization attack [3, 23], and Denial of Service (DoS) attacks [24]. Additionally, careless management of passwords and security measures might jeopardize the anonymity of the DLT system users. The **Majority Attack / 51% attack,** is one of the most commonly discussed attacks in the context of digital coins [3, 23]. In this type of attack, the attacker controls more than 50% of the network's computation power and thus is able to successfully perform bogus blockchain modifications and reorganizations and obtain [temporary] consensus for the bogus blocks [3, 23].

Many of the attacks on the Bitcoin blockchain listed above are applicable to the Ethereum blockchain. An additional set of attacks on Ethereum exploits its smart contract functionality, specifically the computational complexity of the embedded "Turing Complete" [25] functions, which is quantified in terms of "gas," reflecting the cost of computation [5]. The major Ethereum attacks include Reentrancy [26], Front running [27], Integer Overflow and Underflow attacks [28], Unexpected Revert attacks [28], Gas Limit attacks [29], Block Stuffing attacks [29], and Multi-Signature attacks [30].

Common Attacks on HFT Blockchain deal with the centralized and permission-based aspects of the HFT DLT system, particularly attacking the membership service provider that authorizes and provides permissions for entrance into the blockchain and blockchain transactions [31]. The major HFT attacks are the Insider Threat attacks [32] and the Certificate of Authority attack [33].

3 Literature Review

DLT systems provide a decentralized platform. Hence, its potential usage in the field of big data exchange may have significant benefits. Nevertheless, to the best of our knowledge, Fibereum is the first DLT system that provides an optimal solution for that purpose while offering significant benefits in other use-cases. Since Fibereum is a unique DLT system using lazy verification and non-linear data structures, its

composition and computation processes are especially efficient. An extensive literature review performed resulted in very few publications that specifically address the issues that Fibereum addresses. Three papers are listed below.

Cäsar et al. have developed a DLT system named Cerberus that focuses on the particular ordering of State Machine Replication (SMR) across a network of unreliable machines [34]. This DLT's consensus mechanism is based on a leader-based Byzantine fault-tolerant consensus approach [13]. In contrast, we propose consensus mechanisms such as the lazy verification mechanism, which minimizes the need for prompt and incentivized consensus, and enhance fault tolerance at lower computational resources.

Snow has developed Factom, a general-purpose data layer that creates a consensus system to ensure that entries are quickly recorded [35]. Comparatively, Fibereum offers the lazy verification approach, which is more suitable for numerous use-cases (see Subsection 4.2 and Section 5). Additionally, Fibereum offers the use of other non-linear data structures, such as Merkle-heaps, for the Inter-block DLT system's construction and maintenance. Thus, Fibereum enables a more efficient method of verification and DLT operation, especially for Big Data exchange.

Parachain [36] uses chains that are processed in parallel, thereby has the potential to improve throughput. Parachain DLT has not addressed certain issues related to overseeing blockchain creation. Fibereum provides better support of "safe" parallelism via the mechanism of using Merkle Heaps for inter-block construction; at the same time, the mechanism is highly efficient and provides $O(log(n))$ [5] complexity for DLT consensus and construction.

Other relevant papers that are not completely overlapping Fibereum concepts and targeted use cases. Examples include work Gay et al. [37], and by Zhu et al. [38].

4 The Fibereum DLT

The Fibereum DLT introduces the following novel features: (i) Options for storing information/transactions blocks in data structures, including Merkle trees, Merkle heaps, or Merkle hash tables, rather than as a linear list in the form of a blockchain; (ii) Enablement of low complexity algorithms and parallel processing; (iii) Lazy Verification – minimizing the need for incentivized DLT systems' construction and consensus verification; (iv) Enablement of permission-based and permissionless modes of access and operations; (v) Enablement of encryption and compression of the data; (vi) Enabling improved cyber security, protection against attacks, and fault tolerance; (vii) Providing additional layers of encryption and digital signatures (in addition to cryptographic hash functions referred to as the digests [1, 2]); (viii) Enablement of Embedded Turing Complete [25], static and/or dynamic, code, which provides efficient management of smart contracts [4] and End User License Agreements (EULA) [37, 38]; (ix) Enablement of efficient management of static, dynamic, and ad-hoc federated data, including terms and policy management for monetization (please see the section on use-cases); (x) Enablement of systems for data governance and currency exchange; (xi) Providing an option for using more than one DLT system in tandem. The latter is referred to as multi-plan implementations. For example, in the exception maintenance use-cases (4.2 and

5.1), we introduce three DLT system plans: one for data transactions governance, one for data exchange, and one for smart contracts – defining data ownership, usage policies, rights, management, and governance

4.1 A Merkle-based Verification System

Implementations of Merkle trees-based non-linear DLT systems offer several key advantages over linear blockchains. These advantages include: (i) Merkle trees-based implementations maintain the integrity by cascading any change to the cryptographic hash. Pointing from the previous node in the tree back to the Merkle root would invalidate the changed block. (ii) Merkle trees-based implementations are typically efficient for the construction and verification of DLT systems, offering the complexity of $O(log(n))$ (or, in some cases, $O(n \times log(n))$) rather than $O(n)$ (or, in some cases, $O(n^2)$). Hence, Merkle tree-based implementations can reduce the temporal and spatial computational complexity. Moreover, without a Merkle tree, the data would need to be sent across the network for verification. Hence, Merkle trees reduce the data transfer delay. (iii) The Merkle tree structure of the local blockchain can use Proof of Inclusion [16], a method of verifying the validity of data without needing to move data across all parts of the network. This algorithm can work in conjunction with consensus mechanisms, such as PoW and PoET. The joint Merkle heap, proposed in some of the Fibereum use-cases, is beneficial for efficient traversal without revealing all portions of the structure. To further demonstrate the principles of Fibereum's operation and its novelty, we present an important use-case here, and the rest of the use-cases are presented in Section 5.

4.2 Exception Maintenance 1

This use-case considers the situation that an airplane manufacturing company X buys an engine from an engine manufacturing company Z, and the engine is installed on an airplane of an airline company Y. In other words, this is a business-to-business-to-business (B2B2B) scenario. To simplify the example, we assume that the engine is in X's possession, and the use-case is a typical B2B use-case between X and Y. Generally, Y owns the data. However, in some scenarios, the ownership of the data might be shared between X and Y. It is assumed that according to a licensing agreement between X and Y, Y collects and owns the engine's sensor data. The proposed Fibereum DLT system is designed to be used for managing data usage, ownership, and storage used by the companies and their affiliates in a way that enables dealing with exceptions in the regular operation of the airplane.

DLT system operation procedures:
(i) Verifiable sensor data is collected in a joint heap, accessible by both X and Y. The data is "verifiable" in the sense that it includes means for verifying its authenticity, e.g., cryptographic hashes of time stamps and sensor IDs. (ii) Each heap node stores specific components of the sensor data (e.g., temperature, pressure, and position) in the storage

area. In some implementations, the heap storage is based on a string pool [40]. (iii)Similarly to blockchains, such as the Bitcoin blockchain, the data is stored in blocks and organized in blocks via an internal Merkle tree (the Intra-Merkle Tree). These blocks are maintained by an external Merkle heap (the Inter-Merkle heap). (iv) Inter-block and Intra-block storage via Merkle heaps or trees simply imply that both parties can traverse each piece of sensor data as well as the entire collection of sensor data efficiently and "quickly." (v) If legitimately requested, timely verification is used to ensure data integrity. The fact that the verification is done only on a need-to-do basis and at the time of the need-to-do verification is the origin of the name "lazy verification." The lazy verification process can reduce the cost of operations.

Lazy verification

The lazy verification process is activated when the need arises (e.g., dealing with an exception in the engine's operation). At this time, a new DLT system based on a data structure such as a sorted Merkle (tree in this example) or a Blockchain (in use-case 5.1) may be constructed. If there is no malicious activity, valid blocks are fetched from the joint heap of the first DLT, and each valid block is appended to the second DLT system in the order of the recency of activities.

Our current implementation of the system has the following components: permissionless or permission-based application of the Fibereum DLT system used for validated exchanges, min-heap DLT system for exchanges that still have to be validated, and license agreement programmed into smart contracts.

To elaborate: data continuously flows from Company X's engine to the joint Merkle heap-based DLT system (DLT System-1) shared by both X and Y. On exception (e.g., overheating, mis-assembly, or exhaustion), X or Y can choose to request lazy verification regarding the exception. In this case, X and Y can nominate one or more proxies and assign the task of verification to the proxies. At this point, the process might resemble verification on blockchains, such as the Bitcoin blockchain. The proxies act like miners and assemble a second DLT system (DLT System-2), which is a Merkle Tree-based DLT or a blockchain-based DLT. DLT System-2 contains only blocks that have been verified by the proxies. Following the request, Companies X and Y (or their proxies) check the sensor data stored in DLT System-2 to determine whether there is a legitimate error, such as engine exhaustion. If there is a nonfunctional component, the lazy verification tags the exception as valid, and Company Y can take steps to fix the issue with the engine. Otherwise, the lazy verification deems the exception invalid, implying that there has been a human error in maintaining the engine and that the engine is functioning properly.

Figure 1 depicts the process and the related scenario of lazy verification. As shown in the figure, sensor data from Company Y's engine is transported from X's plane to the shared heap, demonstrating Fibereum's ability to function in B2B relationships.

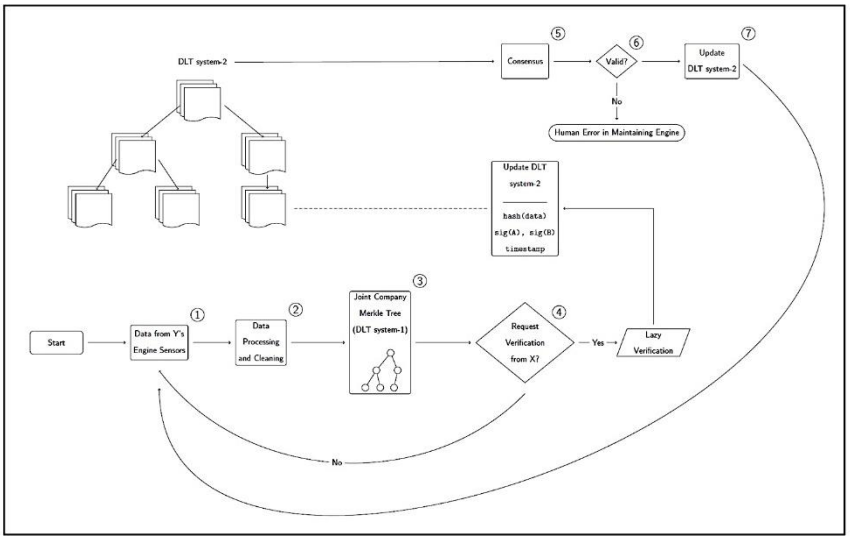


Fig. 1: Big Data Exchange Use-Case

As Figure 1 shows, in a regular mode of operation: (i) Company Y collects data from the engine. (ii) Y might perform data pre-processing and conditioning. (iii) Y places the data in a verifiable form into DLT System-1. This process (i, ii, and iii) continues as long as there is no valid request for verification, e.g., Company X wishing to terminate the contract with Company Y and replace the engine with a new engine from Company W.) (iv) In case that X, Y, or an authorized third party initiates a legitimate request for verification, X and Y nominate miners (e.g., proxies). (v) The proxies generate local copies of DLT System-1. (vi) A consensus algorithm, such as PoET, is applied and used to construct and/or extend a second DLT system (DLT System-2). For clarity, DLT system-2 is referred to as the global DLT system – that is, a DLT system that has been verified or extended according to a consensus algorithm. If the data is valid, DLT System-2 is updated. Otherwise, an exception is raised. (vii) DLT System-2, the global DLT system, is used to address the exception.

If some blocks do not pass consensus verification due to a malicious act or negligence by one of the parties breaching the agreement, then the matter may be further pursued, e.g., brought to courts or arbitration. Note that in order to save storage space, DLT System-2 might only contain cryptographic pointers to nodes of DLT System-1, thereby serving as a transaction management DLT system.

4.3 Fibereum Permissionless and Permission-based DLT System Applications

The general mode of operation of Fibereum is permissionless. Nevertheless, in some implementations and use-cases, Fibereum enables permission-based operations via options for access control, compression, and encryption, as well as compression and encryption in tandem [41]. Access control can include password protection access for

administrators, system utilities, users, user groups, and the general public. Additional layers of access control can include encryption. Access to utilities for the DLT system construction (e.g., adding blocks) and consensus verification can be subjected to access verification protocols.

The proposed new schemas enable the construction of DLT systems' implementations that are permissionless, permission-based or mixed permission-based/permissionless mode DLT systems. This is enabled via one or more of the following mechanisms: (i) Post verification, some of the generated DLT system blocks (e.g., data, transaction, and contract blocks) might be completely open, i.e., permissionless. Other blocks might be fully protected via access and encryption mechanisms, enabling a mixture of permission-based and permissionless DLT systems. (ii) Parts of plain and encrypted blocks might be available to different entities like a puzzle, where the mechanism to assemble the puzzle pieces is controlled via cryptographic functions. (iii) Multiple copies of the DLT system might increase fault tolerance. Finally, (iv) it should be noted that the level of protection can be related to the sensitivity of the data, where sensitive data might be encrypted and subject to access control.

4.4 Cyber Security and Fault Tolerance Enablement

The proposed Fibereum DLT system schema provides additional security layers for permission-based and permissionless DLT systems. The concept of lazy verification enables storing the data in local storage, where access can be controlled and protected in several ways. The data can be digitally signed, compressed, and encrypted – potentially using methods for tandem compression and encryption. In some implementations, the digests, digital signatures, as well as public and private keys used for encryption can be associated with the physical devices used to generate, transmit, or process the data and with the time that the data was generated. Furthermore, the verification stage can be limited to "trusted parties" and proxies that have protected and potentially permission-based access to the data. Consequently, Fibereum provides better protection against commonly used Blockchain attacks listed in Section 3. The following subsection provides further details concerning potential attacks on Fibereum and the resilience of Fibereum implementations to such attacks. This resilience is referred to as "counterattacks."

4.5 Counterattacks by Fibereum

The main threats to Fibereum (and many other DLT systems) are Sybil [20], Majority (51%) attack [3, 23], Denial of Service (DoS) [24, 30], and insider attacks [32]. Several components of the Fibereum DLT system can reduce or completely eliminate the risk emanating from the above and other DLT system attacks. First, when applicable, the option for lazy verification provides ample time to detect and prevent those attacks. Second, some implementations may use the PoET verification protocol for lazy or prompt verification. This increases the resilience of Fibereum to DLT system attacks. Finally, the utilization of permission-based or mixed permission-based and permissionless systems' components can be a paramount counterattack method.

It has been established that without a centralized authority, a system might be susceptible to Sybil attacks [31]. Consensus algorithms (e.g., PoW and PoET) mitigate the effects of Sybil attacks, and permission-based implementations of Fibereum can completely prevent such attacks. Similarly, a Majority Attack would fail with permission-based Fibereum utilizing PoET. In particular, data passing the lazy verification process would enter the second Merkle-based DLT system (System-2), where it would then need to pass through PoET in order for the first and/or the second DLT system to be updated.

Many forms of DoS counterattack methods in general networks exist; some of these methods are applicable to Blockchain DoS [24, 30]. Fibereum can be more resilient to DoS since it is possible that for long periods of time, the only DLT system activity is updating DLT System-1 with new verifiable data, which provides ample time to detect and mitigate a DoS attack.

Insider attacks are the most difficult to prevent, but they often only affect permission-based DLT systems. Measures to reduce the threat of insider attacks have been proposed. Some of these measures are common to many other permission-based systems. Other measures use a blockchain traceability system with a differential traceability algorithm, both of which can be implemented into Fibereum [32].

4.6 Fibereum Smart Contracts

Fibereum Smart contracts are irreversible contracts enforced by the program code embedded in the Fibereum blocks, which are not controlled by users. These contracts are limited by their inability to send HTTP requests and access off-chain data directly. Ethereum can circumvent this limitation via oracles, but this leaves transactions susceptible to attacks that manipulate data and price values [28]. In some implementations, in order to resolve this issue, Fibereum maps the smart contract onto a specific version of an End User License Agreement (EULA) [39]. Hence, the EULA might guide the lazy verification procedure.

5 Additional Fibereum Use-Cases

In this section, several Fibereum use-case examples are presented, concentrating on B2B Applications of the Fibereum DLT system for [Big] Data Exchange use-cases. The concerns related to the data exchange use-cases are data ownership, data rights, data use agreements, managing survivability and termination clauses for contracts, data integrity, liability, monetary value, and responsibility for disclosing the data and its use to third parties, governments, and governing authorities. In these types of use-cases, our objectives include creating a framework for policies governing data exchange in a B2B environment, where data exchange transactions are bounded by a legal contract, potentially in the form of license agreements or subscriptions (signed or click-through), specifying certain terms, such as ownership, usage policies, rights, management, and governance. Often, these agreements take the form of End User License Agreements, Developer License Agreements, and Data License Agreements. Given a predetermined

legal contract, Fibereum aims to minimize the computational burden of consensus verification. It should be noted that Fibereum also provides efficient mechanisms for support of other use-cases, including digital currency exchange, B2C data exchange, as well as services related to data exchange. This section includes examples of these use-cases as well.

Some of the use-cases might include additional Fibereum-based DLT systems, e.g., a DLT system for transaction management that records the process of data exchange and a smart contract DLT system that is used to dictate data usage, rights, and termination. The use-case examples, however, do not elaborate on the internals of the smart contract DLT systems and their operation. Finally, all the use cases may deploy a permissionless version of Fibereum, a permission-based version, or a combination.

5.1 Exception Maintenance 2: Data Networks

In some Fibereum implementations, DLT System-2 is a Blockchain DLT. As an example of such an implementation, one can consider a use-case where company U is a process control firm that has sensors installed in an oil refinery that belongs to company V. In this implementation of the Fibereum DLT system, the data is stored locally by the data stakeholders (e.g., by companies U, V, and their affiliates/proxies) in Merkle heaps. Additionally, a Fibereum DLT system for transaction management may record the process of data exchange, and a Fibereum smart contract DLT system may be employed to dictate data usage, rights, and termination. When new sets of sensor data are available, they are aggregated into blocks, and the blocks are inserted into the Merkle heaps. At the same time, the transaction management DLT system is updated. At the time that consensus verification is mandated (e.g., a dispute between Company X and Company Z or a discovery subpoena by local authorities due to an accident), the integrity of the data stored in the heaps and proxies is assessed.

The following is a flowchart of the of DLT system-2 construction and lazy verification procedure applied in the Exception Handling use cases:

The Lazy Verification Procedure

Require: LazyVerification(*DLT system-1*, *DLT system-2*)
1: **while** *DLT system-1* is non-empty **do**
2: $d \leftarrow$ POP(*DLT system-1*)
3: **if** *d* is valid **then**
4: APPEND(*DLT system-2*, *d*)
5: **else if** *d* is invalid **then**
6: raise legal issue
7: **end if**
8: **end while**

5.2 Digital Currency

This use-case considers digital currency applications that are similar to Bitcoin and Ethereum digital coins exchange. In contrast to most other digital coin DLT systems, both the intra-block and the inter-block may be managed via Merkle trees. Due to the nature of the application and potential attacks, the verification may be prompt and incentivized using fees or digital coin mining rewards. A PoET consensus mechanism may be employed to reduce operational complexity and energy consumption and improve counterattack capabilities.

5.3 Targeted Advertising

This use-case may be a B2B or a B2C use-case. For example, assume that Company P manufactures autonomous vehicles and Company Q, or a consumer R, uses these vehicles, which collect federated data along with sensor data. Specifically, suppose that a consumer X buys a car manufactured by Company Y, and Company Z wishes to access parts of the sensor data from the car. The process is similar to the process described in the above exception maintenance use-cases. However, it might utilize two Merkle heaps or one heap with access control to heap elements. Both classified and unclassified information is accessible to X and Y via one heap, but, for the protection of X's privacy, the second heap contains only unclassified information for Z.

5.4 Patient medical history

This use-case considers situations where patient X wishes to switch from care provider Y to care provider Z and then transfer their medical history to Z. Due to stringent confidentiality requirements, it is most likely that the DLT system implementation would be permission-based, preventing unauthorized access to medical records. Any new medical data that goes into the records by the care provider must be verifiable and potentially include encryption, digests, and the digital signature of the patient before it is added as a block to the DLT. Provider Y uses its own encryption, digests, and digital signature to securely access medical records. If the patient wishes to share their information with other care providers, e.g., Z, then Provider Y might require a digital signature of Patient X and Provider Z for consent to release information. The DLT system includes a network of care providers, as medical records may need to be transferred from one care provider to another care provider. In this case, the medical records and other information are encrypted in a Merkle heap (DLT system-1) and, following verification, sent to other care providers through the Merkle tree of DLT System-2. Lazy verification, initiated on a "need to do" basis, e.g., switching a care provider, is used to check permissions and verify information correctness.

5.5 Digital Cartography

This use-case considers a situation where the system includes satellites, e.g., X1, X2, X3, and X4, a ground station Y, a user Z, and an object of interest W. The information

generated by the remote sensing satellites and gathered by the ground station (e.g., GPS locations of Object W) is stored in the Merkle heap-based DLT System-1 and is accessible to User Z. The system allows User Z to request a legitimate verification of certain parts of the information. This triggers lazy verification and the creation of DLT System-2. The verification may include proxies. Since storage is placed within a heap, the location may be constantly updated, and User Z can constantly update the positions of Object W by requesting lazy verification. If an exception occurs, the data stored in the heap can be used to track previous locations with precise timestamps from the satellites' atomic clock in order to help figure out what may have happened.

5.6 Non-Fungible Tokens

This use-case considers situations where a DLT system is used for the exchange and management of Non-Fungible Tokens (NFT) [11]. In contrast to most other NFT DLT systems, both the intra-block and the inter-block may be managed via Merkle trees. Due to the nature of this application and potential attacks, the verification may be prompt and incentivized using fees[1].

5.7 Smart Contracts and Licensing Agreements

This pertains to cases where a DLT system is used for smart contract management. In contrast to most other smart contract DLT systems (e.g., Ethereum-based smart contracts), both the intra-block and the inter-block may be managed via Merkle trees. Due to the nature of the application and potential attacks, the verification may be prompt and incentivized using fees[1].

USCG is interested in exploring, along with our team, the utility of DLT systems for these use-cases and may utilize the DLT systems for licensing, e.g., licensing of fishing companies and vessels. Given that the DLT users are not necessarily USCG staff members, the system may have to tighten security measures with respect to access to the DLT and the construction of DLT system blocks.

5.8 Copyrighted material

This use-case considers situations where Consumer X wishes to access copyrighted material produced by Company Y. In a possible DLT system implementation, the operation procedures are similar to the operating procedures of the DLT system described in Use-case 4.1. However, the amount of data stored in DLT System-1 is not as big as the amount of data expected in Use-case 4.1. Furthermore, the DLT system may be permission-based so that only Consumer X and Company Y can access the material. Release of the material requires the consent of both parties, i.e., Consumer X and

[1] A PoET consensus mechanism may be employed to reduce operation complexity and energy consumption and improve counterattack capabilities.

Company Y must both sign in order to sell content to a third party. Note that this use case has some overlap with NFT and can be used as a DLT system for NFT.

5.9 Commerce, Supply Chain Management, and Inventory Management Systems

This use-case considers situations where Company X wishes to transport goods to a warehouse owned by Company Y. In this case, a comprehensive database accessible by both X and Y can be used. This DLT system is likely to be permission-based so that only the two parties and their affiliates have access to it. The original owner of the DLT system, Company X, shares parts of the database of verifiable transactions with Company Y in a Merkle heap-based DLT System-1. Lazy verification may be used to generate DLT System-2 in order to mine data, verify transactions, manage inventory, and validate the integrity of the data and the underlining supply chain. Inventory management can be implemented in a similar way.

The US Coast Guard (USCG) is interested in exploring, along with our team, the utility of DLT systems for these use-cases and may utilize DLT systems under the assumption that the users are internal to the organization. Hence, they may have "some" level of trust by the system (e.g., after supplying credentials that associate them with the USCG).

5.10 Weather Broadcasting

This use-case considers a situation where two or more weather stations (e.g., X and Y) wish to share data regarding weather conditions in a certain region. The DLT system may be permission-based so that only Stations X and Y can access the data. Sensor data created by X and Y is verifiable and stored in DLT System-1. Lazy verification and the creation/update of DLT System-2 may take place when there are discrepancies between X and Y as to the data relied upon or in their weather prediction.

6 Conclusion and Future Research

The DLT systems and methods discussed above include novel modifications to blockchain technology, providing a superior framework for DLT systems. Those modifications can improve the cost/performance of DLT systems in current applications and extend the utility of current DLT systems to several new applications and use-cases. The Fibereum DLT system has various applications in the fields of Big Data, including Transportation, Smart Cities, Healthcare, Process Control, and Internet of Things. Fibereum DLT systems are also suitable for other applications, such as Digital Currency, Smart Contracts and licensing, and Supply Chain Management.

Future work can include: (i) Implementations for other use-cases and data exchange applications; (ii) Monetization and control of federated data; (iii) Enhancements to B2C applications where concerns may include tight privacy constraints, as well as consumer rights protection; (iv) Further exploration of the utility of additional cryptographic data

structures and other non-linear data structures, e.g., directed acyclic graphs for transactions and/or data storage; (v) Appending new data transaction information to a concurrent transaction data structure rather than directly appending it to the DLT; (vi) Appending new sensor data to a concurrent sensor data structure rather than directly appending it to the DLT; (vii) Further exploring the management of federated data where different parts of copies of the data reside in the DLT systems of individual parties; (viii) Exploring implementations where the data is compressed and encrypted, potentially for enabling permission-based access.

Acknowledgment

This material is based in part upon work supported by the Department of Homeland Security grants TXST83938 and E2055778 and by the National Science Foundation under Grant CNS-2018611 and CNS-1920182.

References

1. Arvind Narayanan. Bitcoin and cryptocurrency technologies: A comprehensive introduction. Princeton University Press, 2016.
2. Alexander Lipton and Adrien Treccani. Blockchain and distributed ledgers: Mathematics, technology, and economics. World Scientific, 2022.
3. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2009. URL: http://www.bitcoin.org/bitcoin.pdf.
4. Gavin Wood. "Ethereum: A secure decentralised generalised transaction ledger." In: Ethereum project yellow paper 151 (2014), pp. 1–32.
5. Ellis Horowitz and Sartaj Sahni. *Fundamentals of Data Structures*. Sung Kung Computer Book Co., 1987.
6. Elli Androulaki et al. "Hyperledger fabric." In: *Proceedings of the Thirteenth EuroSys Conference*. ACM, Apr. 2018.
7. *Introduction to smart contracts*. https://ethereum.org/en/developers/docs/.
8. *Max Muller. Essentials of inventory management. HarperCollins Leadership, 2019.*
9. Sunil Chopra and Peter Meindl. "Supply chain management. Strategy, planning & operation." In: What's New in Operations Management. Pearson, Jan. 2018.
10. Paula Fraga-Lamas and Tiago M. Fernández-Caramés. "Leveraging Distributed Ledger Technologies and Blockchain to Combat Fake News." In: *CoRR*abs/1904.05386 (2019). arXiv:1904.05386. http://arxiv.org/abs/1904.05386.
11. Qin Wang et al. Non-Fungible Token (NFT): Overview, Evaluation, Opportunities and Challenges. 2021. eprint: arXiv:2105.07447.
12. Ralph Merkle. "A Certified Digital Signature." In: Crypto 89. Aug. 1989, pp. 218–238.
13. Miguel Castro and Barbara Liskov. "Practical Byzantine Fault Tolerance and Proactive Recovery." In: ACM Trans. Comput. Syst. 20.4 (Nov. 2002), pp. 398–461
14. Mohammad Dabbagh et al. "A survey of empirical performance evaluation of per-missioned blockchain platforms." In: Computers &Security 100 (2021), p. 102078.
15. Amitangshu Pal and Krishna Kant. "DC-PoET: Proof-of-Elapsed-Time Consensus with Distributed Coordination for Blockchain Networks." In: 2021 IFIP Networking Conference (IFIP Networking). 2021, pp. 1–9
16. Kun Peng. *A general, flexible, and efficient proof of inclusion and exclusion*. Jan. 1970
17. Proof-of-stake (POS). https://ethereum.org/en/developers/docs
18. Yih-Chiao Liu. Performance of a CSMA/CD protocol for Local Area Networks. https://ieeexplore.ieee.org/document/1146621.

19. Andrew S. Tenenbaum. *Computer Networks*. Prentice Hall of India Pvt. Ltd., 2009.
20. Ethan Heilman, Alison Kendler, Aviv Zohar, Sharon Goldberg. *Eclipse Attacks on Bitcoin's Peer-to-Peer Network*. 2015. usenixsecurity15.
21. Mubashar Iqbal and Raimundas Matulevi˘cius. "Exploring Sybil and Double-Spending Risks in Blockchain Systems." In: IEEE Access 9 (2021), pp. 76153–76177.
22. *Janhvi Joshi and Rejo Mathew. "A Survey on Attacks of Bitcoin." In: Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI - 2018). 2020, pp. 953–959.*
23. Jake Frankenfield. *51% attack: Definition, who is at risk, example, and cost*. Oct. 2022. https://www.investopedia.com/
24. Mayank Raikwar and Danilo Gligoroski. *DoS Attacks on Blockchain Ecosystem*. 2022.
25. Marc Jansen. "Do Smart Contract Languages Need to Be Turing Complete?" In: Blockchain and Applications. 2020, pp. 19-26
26. Ethereum Smart Contracts." In: *2020 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*. IEEE, Feb. 2020.
27. Christof Ferreira Torres, Ramiro Camino, and Radu State. An Empirical Study of Frontrunning on the Ethereum Blockchain. Usenixsecurity21 2021.
28. Zulfiqar Ali Khan and Akbar Siami Namin. A Survey on Vulnerabilities of Ethereum Smart Contracts. 2020. ArXiv 2012.1448.
29. Huashan Chen et al. A Survey on Ethereum Systems Security: Vulnerabilities, Attacks and Defenses. 2019. ArXiv.1908.04507.
30. Noama Fatima Samreen and Manar H. Alalfi. "SmartScan: An approach to detecting Denial of Service Vulnerability in Ethereum Smart Contracts." In: *CoRR* abs/2105.02852 (2021). arXiv: 2105-02852. https://arxiv.org/abs/2105.02852.
31. Efthimios-Enias Gojka et al. "Security in Distributed Ledger Technology: An Analysis of Vulnerabilities and Attack Vectors." In: Intelligent Computing. 2021, pp. 722–742.
32. Benedikt Putz and Günther Pernul. Trust factors and insider threats in permissioned distributed ledgers - an analytical study and evaluation of popular DLT Frameworks. https://core.ac.uk/outputs/232204350.
33. Amanda Davenport, Sachin Shetty, and Xueping Liang. "Attack Surface Analysis of Permissioned Blockchain Platforms for Smart Cities." In: *2018 IEEE International Smart Cities Conference (ISC2)*. 2018, pp. 1–6.
34. Florian Cäsar. *Cerberus: A parallelized BFT consensus protocol for radix*. Jan. 1970. https://api.semanticscholar.org/CorpusID:221297416.
35. Paul Snow et al. *Factom Ledger by Consensus*. Jan. 2015. https://cryptochainuni.com/wp-content/uploads/Factom-Ledger-by-Consensus. pdf.
36. Gagandeep Kaur and Charu Gandhi. "Chapter 15 - Scalability in Blockchain: Challenges and Solutions". In: Handbook of Research on Blockchain Technology. 2020, pp. 373–406.
37. Gai, K. Wu, Y. Zhu, L. Qiu M., Shen, M. "Privacy-Preserving Energy Trading Using Consortium Blockchain in Smart Grid," in *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, 2019, pp. 3548-3558.
38. Zhu, L. Wu, Y. Gai, K., Kwang, K. sand Choo. R., "Controllable and trustworthy blockchain-based cloud data management." Future Generation. Computer. Systems. 91, C, 2019, pp. 527–535.
39. Hamish Cotton and Christopher Bolan. "User perceptions of end user license agreements in the smartphone environment." Australian Information Security Management Conference. 2018. PP 235-244
40. Richard Bubel, Reiner Hähnle, and Ulrich Geilmann. "A Formalisation of Java Strings for Program Specification and Verification." In: Software Engineering and Formal Methods.
41. Tamir, D. and Bruck D. *Compression and Decompression Engines and Compressed Domain Processors*, US Patent 10404277. 2019

Indistinguishable Obfuscated Encryption and Decryption Based on Transformer Model

Pengyong Ding[1], Zian Jin[2], Yizhong Liu[2⋆], Min Sun[1], Hong Liu[1], Li Li[1], and Xin Zhang[1]

[1] China Mobile Information Technology Company Limited, Shenzhen, China,
[2] Beihang University, Beijing, China
dingpengyong@chinamobile.com, JinZiAn@buaa.edu.cn, liuyizhong@buaa.edu.cn, sunmin@chinamobile.com, liuhong@chinamobile.com, liliit@chinamobile.com, zhangxinit@chinamobile.com

Abstract. To solve the problem in secure encryption in cryptography, *indistinguishability Obfuscation* (iO) was born. It is a crypto-complete idea, based on which we can build many cryptographic construction. The implementation of it can hide both the dataset and the program itself. In this paper, we use the idea of translation in the (*Natural Language Processing*) NLP-like language model to realize the conversion between plaintexts and ciphertexts with the help of hints. We trained a self-attention transformer model, successfully hiding the dataset as well as the encryption and decryption programs. The input of the encryption model is a plaintext prefixed with a hint and the output is the result of encryption using one of the specified algorithms. The input and output of the decryption model are the opposite of the encryption one.

Keywords: Information security, indistinguishable confusion, transformer, blockchain

1 Introduction

With the informationnization of the whole society, new information technologies such as cloud computing[1,2,3], new computer hardware [4,5,6], the Internet of Things [7,8,9], blockchain [10,11,12], etc., have been gradually applied to various industries in society, leading to the exponential growth of all kinds of data [13,14,15]. As the core asset of information systems, data are of great value, so data security[16] is becoming more and more important. In the information era, data assets have become one of the most important assets of each enterprise, thus data security [17,18,19] has become a key concern for enterprises. However, transmitting data in public is inevitable, so the security of data encryption and decryption is the top priority.

Secure encryption has a crypto-complete construction in cryptography, also known as indistinguishable Obfuscation(iO)[20], which could hide both the dataset

⋆ Corresponding author.

and the program itself. Based on iO, we could build an encryption algorithm that can implement almost all other encryption protocols, including but not limited to public key encryption, function encryption, digital watermarking system, etc. Computer scientists proved that indistinguishable obfuscation can be the basis for almost all cryptographic protocols (except black-box obfuscation), including classical cryptographic tasks (e.g., public-key encryption) and emerging tasks (e.g., fully homomorphic encryption). It also covers cryptographic protocols that no one knows how to construct, such as deniable encryption and function encryption. But until now there is no practical iO implementation. This paper provides a viable solution for implementing it.

Blockchain[21] is a distributed system with multiple participants, secret communication [22], transaction processing [23], and committee reconfiguration [24,25] among the participants often requires public-key cryptographic systems. Therefore, iO-based public-key cryptographic protocols can significantly improve the security of communication between the participants of a blockchain system [26]. The iO-based public-key cryptographic protocols require shorter keys to achieve the same security strength, which can effectively reduce the communication complexity of blockchain consensus protocols[27] and accelerate the agreement of the consensus [28]. It can be seen that iO could be well coupled with blockchain and has a wide application prospect.

This paper analyzed and discussed the combination of iO and blockchain[29,30]. Applying iO to blockchain can improve the security of the system[31,32] or reduce the time needed for consensus, which has a wide application prospect in various field [33,34,35,36]. The combination of blockchain and iO cryptography need to be studied further [37,38].

The rest of the paper is organized as follows. In Section 2, we introduce the basic ideas of the method, followed by the description of the algorithm. Then, we present the experiment and give the evaluation of the results in Section 3. Finally, we conclude the paper in Section 4.

2 Constructing indistinguishable obfuscated encryption and decryption based on transformer model

2.1 Basic ideas

In this paper, we implement the conversion of plaintext and ciphertext by combining hints through a translation behavior similar to that in the language model[39]. The ciphertext output from the model plus some noise (encrypted data from public transmissions, etc.) is then converted to plaintext as input to the model.

For the linguistic autoregressive model, we build the following conditional language model, which predicts a word y from the first n known words, or phrases.Known $X(x_1, x_2, ..., x_n)$, according to the conditional language model,

$$p(y_t \mid y_1, y_2, ..., y_{t-1}, X) \tag{1}$$

to output the corresponding $y = (y_1, y_2, ..., y_T)$.

The Transformer[40] self-attentive architecture, on the other hand, has a great improvement in the expressiveness of the model for computing these conditional probabilities. The above unconditional language model could be transformed into a conditional language model when the specific encryption algorithm is known. The model in this paper is controlled based on the following 2 methods:

1) When encrypting the plaintext or decrypting the ciphertext, we can add hints (numbers or other representations of encryption algorithm categories) in front of the input.

2) In the normalization of the Transformer model's translation behavior, the main method layer is normalized by the following formula:

$$\mu^l = \frac{l}{H}\sum_{i=1}^{H} a_i^l \quad \sigma^l = \sqrt{\frac{l}{H}\sum_{i=1}^{H}\left(a_i^l - \mu^l\right)^2} \tag{2}$$

l denotes the L^{th} hidden layer, H denotes the number of nodes in the layer, and a denotes the value of a node before activation. Suppose the current input is x^t and the previous hidden state is h^{t-1} , then the weighted input vector (input of the nonlinear cell) is

$$a^t = W_{hh}h^{t-1} + W_{\mathrm{xh}}x^t \tag{3}$$

Layer normalization for weighted input vectors, followed by scaling and translation (for recovering nonlinearity)

$$y = g \cdot \hat{a}^t + b \quad \hat{a}^t = \frac{a^t - u^t}{\sigma^t} \tag{4}$$

where g is denoted as the gain and b denotes the bias parameter. The above is the basic normalization, while the condition c (the representation of cryptographic algorithm classes) is turned into the same dimensions as g and b in equation (4) by different matrix transformations (i.e., linear mapping), respectively, and then the transformations are mapped to g and b.

$$g' = w_{\boldsymbol{g}} * c + g \quad b' = w_b^* c + b \tag{5}$$

The input is processed by the transformations mentioned above, and the translation behavior of the Transformer model is controlled by the method in 2), so that the conversion of plaintext and ciphertext are realized. At the same time, the ciphertext transmitted to the public is obfuscated and only the corresponding decoding module can convert it to plaintext, so that there is no fear of interception. The model can also generate an obfuscated ciphertext with a specified encryption algorithm on demand, which can only be decrypted into plaintext by the trained decryption model.

To be able to train more layers (e.g., 100, 1000 or even 10,000 layers), the scaling multiplier associated with the number of layers needs to be adjusted during layer normalization to prevent gradient explosion. The scaling multiplier is given by the following formula, where N is the number of layers:

$$\alpha = (2N)^{\mathbf{1/4}} \quad \beta = (8N)^{-1/4} \tag{6}$$

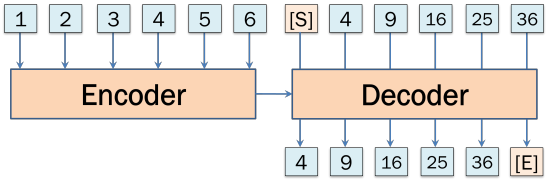


Fig. 1. Simple architecture for encryption and decryption model

2.2 Algorithm description

With the rapid development of multicore [41,42,43] and large memory [44,45,46], large amounts of data and big software [47,48,49] can be implemented quickly. Hence, privacy and security becomes a big concern in data storage, processing, and transmission [50,51,52]. The whole system architecture is divided into an encryption module and a decryption module, which form the transformer model [53,54,55]. Fig1 is a schematic diagram of the encryption model and the decryption training model.

Algorithm1 is the description of our specific algorithm (encryption and decryption processes are almost the same, only the source and the target are reversed).

Algorithm 1

Input: source data, src_seq; target with label, $target\_seq$; algorithm category, C; number of self-attentive computation block layers, num_layers; batch, $epoch$

1: $num\_layers = 200, \alpha = 1599.75,\ \beta = 4.472,\ initialweights(including\ w_g, w_b,$ etc.)
2: **for** each $i \in [0, epoch]$ **do**
3: **for** each $j \in [0, num\_layers]$ **do**
4: $new\_src\_seq \leftarrow LN(src\_seq_{i,j} + C_{i,j})$
5: $out\_projection \leftarrow Self\_Attention(new\_src\_seq)$
6: $loss \leftarrow loss + (out\_projection, target\_seq_{i,j})$
7: **end for**
8: **end for**
9: MIN($loss$)

As can be seen from Algorithm1, we treat plaintext and ciphertext as sequences that can be translated into each other, add hints and obfuscation before the input data, and make the algorithm or obfuscation a condition to be added during the layer normalization of the deep self-attentive computation. We control the translation behavior of the transformer by the method described above.

2.3 Algorithm implementation

The implementation steps of the indistinguishable obfuscation and decryption algorithm based on the transformer model are as follows.

1) Determine the number of model layers, calculate the parameters that can prevent gradient explosion for each layer normalized by the number of layers, i.e., equation (6), and initialize the individual weights.

2) Input source sample data and target data, the input data can be prefixed to indicate which encryption algorithm is used, and it can also be used as obfuscated data. The model is trained to predict the ciphertext after encryption.

3) At each layer of layer normalization, i.e., equation (4), algorithms or obfuscated data are added to the normalization as conditions in order to control the translation behavior, i.e., equation (5).

4) input the source sequence into the model, and then perform the self-attentive calculation after the above process, and the model outputs the predicted target sequence, which is subjected to cross-entropy loss calculation with the real target sequence, and the gradient is calculated according to the loss value to update each parameter.

5) Repeat steps 2 to 4 until the loss converges to 0, and the training of the encryption model is completed.

6) Train the corresponding decryption model and repeat the above steps 1 to 5, the difference is that the input of step 2 is the ciphertext predicted by the encryption model, i.e., the source sequence and the target sequence of the above encryption model are switched.

The entire data flow for model training can be briefly summarized as follows.

(i) Encryption model training process: input plaintext or plaintext with prefix added in front, label is the encrypted ciphertext, and the model training is mainly to predict the encrypted ciphertext.

(ii) Decryption model training process: the input is the ciphertext predicted by the above encryption model, the label is the decrypted plaintext, and the model training is mainly to predict the decrypted plaintext.

Fig2 describes the algorithm flow, in which we can find that there are hints (indicating the type of cryptographic algorithm) at the data input and at the normalization of each layer, similar to the "noise perturbation" added to the neural network, thus allowing the model to learn better.

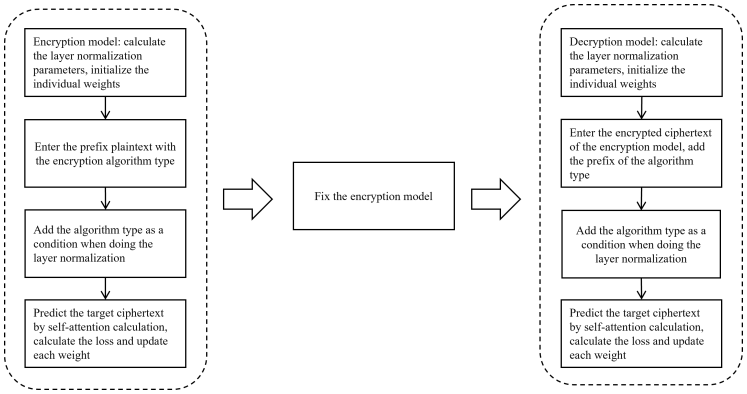


Fig. 2. Algorithm flow

3 Experiment and Analysis

3.1 Experimental environment

In order to verify and analyze the algorithm proposed in this paper, we conducted simulation experiments using the following hardware (arithmetic) and data.

Table 1. Hardware Configuration

Name	**Quantity**	**Configuration**
Calculation Cards	1	A100 80G
CPU	2	64 cores 128 threads intel Xeon
Memory	3	32*3 total 96G
Hard Disk	1	2T

3.2 Experiment

Randomly generate 600,000 pieces of plaintext, each of which is a number of length 11. The 600,000 plaintext are equally divided into 6 data sets, and the data in different data sets are encrypted differently. Then the transformed data sets are combined and a global random disruption is performed. The file is stored by rows, and each row represents a sample, what we want to do is to randomly disrupt the file by rows:

(a) Assuming that the file has a total of $m*n$ lines, the original file is divided equally into m files of n lines each.

(b) Randomly breaking up the file by line for each n lines, since n is arbitrarily specified, so this step can be done in memory.

(c) Read the first line of each file (to get m lines of data) and write these m lines of data randomly to an output file.

(d) Read the 2^{nd},..., n^{th} line of each file in turn, and repeat the operation in step 3 to generate m output files.

Simply, the data is written to m files at random, and the contents of each file are randomly scrambled.

3.3 Evaluation

Table 2 shows the time required to break RSA algorithms with different key lengths.

Table 2. Time required to break RSA

Break Time/MIPS Year	**Key Length/bits**	**Security Level**
10^4	512	Low
10^8	768	Middle
10^{11}	1024	High
10^{20}	2048	High

Table 3 shows the time required to break the iO algorithm proposed in this paper. Observing the data, we can see that the security level of iO algorithm is high when the encryption algorithm type is greater than 6 and the key length is greater than or equal to 512 bits.

Table 3. Security Analysis of iO

Num of encrytion algorithms	n=6	n=7
Break Time 512(Security Level)	10*7! = 10 (Middle)	10*8! = 10 (High)
Break Time 768(Security Level)	10*7! = 10 (High)	10*8! = 10 (High)
Break Time 1024(Security Level)	10*7! = 10 (High)	10*8! = 10 (High)

4 Conclusion

This paper introduced a feasible algorithm implementation for iO with information security protection as the core. By performing security encryption and decryption in a way similar to language translation, it provided a new scheme for data security transmission, and provides a research idea for the feasibility of iO implementation, which is of great significance to improve data security transmission and information protection level. This paper also analyzed and discussed the combination of iO and blockchain. Our study showed that applying iO to blockchain can improve the security of the system or reduce the time needed for consensus, which has a wide application prospect in various field. In future, we will continue to study the combination of blockchain and iO cryptography.

Acknowledgement

This paper is supported by the National Natural Science Foundation of China (U21B2021, 62202027, 61972018, 61932014).

References

1. Lori M Kaufman. Data security in the world of cloud computing. *IEEE Security & Privacy*, 7(4):61–64, 2009.
2. Y. Li, K. Gai, et al. Intercrossed access controls for secure financial services on multimedia big data in cloud systems. *ACM TMCCA*, 2016.
3. M. Qiu, Z. Chen, et al. Energy-aware data allocation with hybrid memory for mobile cloud systems. *IEEE Syst. J.*, 11(2):813–822, 2014.
4. M. Qiu et al. Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems. In *IEEE DATE*, pages 1–6, 2007.
5. M. Qiu, Z. Jia, et al. Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP. *JSPS*, 2007.
6. M. Qiu et al. Dynamic and leakage energy minimization with soft real-time loop scheduling and voltage assignment. *IEEE TVLSI*, 18(3):501–504, 2009.

7. M. Qiu, C Xue, et al. Efficient algorithm of energy minimization for heterogeneous wireless sensor network. In *IEEE EUC*, pages 25–34, 2006.
8. J. Niu, Y. Gao, et al. Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability. *JPDC*, 72(12):1565–1575, 2012.
9. H. Qiu, Q. Zheng, et al. Deep residual learning-based enhanced JPEG compression in the internet of things. *IEEE Trans. on Industrial Infor.*, 17(3):2124–2133, 2020.
10. M. Qiu, H. Qiu, et al. Secure data sharing through untrusted clouds with blockchain-enabled key management. In *3rd SmartBlock conf.*, pages 11–16, 2020.
11. K. Gai, Y. Zhang, et al. Blockchain-enabled service optimizations in supply chain digital twin. *IEEE TSC*, 2022.
12. M. Qiu and H. Qiu. Review on image processing based adversarial example defenses in computer vision. In *IEEE 6th Intl Conf. BigDataSecurity*, pages 94–99, 2020.
13. F. Hu, S. Lakdawala, et al. Low-power, intelligent sensor hardware interface for medical data preprocessing. *IEEE TITB*, 13(4):656–663, 2009.
14. J. Li, Z. Ming, et al. Resource allocation robustness in multi-core embedded systems with inaccurate information. *J. of Systems Arch.*, 57(9):840–849, 2011.
15. H. Qiu, Q. Zheng, et al. Topological graph convolutional network-based urban traffic flow and density prediction. *IEEE Trans. on ITS*, 2020.
16. Dorothy Elizabeth Robling Denning. *Cryptography and data security*, volume 112. Addison-Wesley Reading, 1982.
17. H. Qiu, M. Qiu, and R. Lu. Secure V2X communication network based on intelligent PKI and edge computing. *IEEE Network*, 34(2):172–178, 2019.
18. H. Qiu, Y. Zeng, et al. Deepsweep: An evaluation framework for mitigating DNN backdoor attacks using data augmentation. In *ACM AsiaCCS*, 2021.
19. K. Gai et al. A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance. In *IEEE BigDataSecurity*, 2016.
20. Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 60–73, 2021.
21. Yizhong Liu, Jianwei Liu, Zongyang Zhang, and Hui Yu. A fair selection protocol for committee-based permissionless blockchains. *Computers & Security*, page 101718, 2020.
22. Yizhong Liu, Jianwei Liu, Qianhong Wu, Hui Yu, Yiming Hei, and Ziyu Zhou. SSHC: A secure and scalable hybrid consensus protocol for sharding blockchains with a formal security framework. *IEEE Trans. Dependable Secur. Comput.*, 19(3):2070–2088, 2022.
23. Yizhong Liu, Jianwei Liu, Jiayuan Yin, Geng Li, Hui Yu, and Qianhong Wu. Cross-shard transaction processing in sharding blockchains. In *ICA3PP 2020*, pages 324–339, 2020.
24. Yizhong Liu, Jianwei Liu, Yiming Hei, Wei Tan, and Qianhong Wu. A secure shard reconfiguration protocol for sharding blockchains without a randomness. In *TrustCom 2020*, pages 1012–1019. IEEE, 2020.
25. Yizhong Liu, Yu Xia, Jianwei Liu, and Yiming Hei. A secure and decentralized reconfiguration protocol for sharding blockchains. In *IEEE HPSC 2021*, pages 111–116. IEEE, 2021.
26. Rafael Pass, Lior Seeman, and Abhi Shelat. Analysis of the blockchain protocol in asynchronous networks. In *EUROCRYPT 2017*, pages 643–673, 2017.
27. Yizhong Liu, Jianwei Liu, Zongyang Zhang, Tongge Xu, and Hui Yu. Overview on consensus mechanism of blockchain technology. *Journal of Cryptologic Research*, 6(4):395–432, 2019.

28. Eleftherios Kokoris-Kogias, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ewa Syta, and Bryan Ford. Omniledger: A secure, scale-out, decentralized ledger via sharding. In *IEEE SP 2018*, pages 583–598, 2018.
29. Bin Hu, Zongyang Zhang, Jianwei Liu, Yizhong Liu, Jiayuan Yin, Rongxing Lu, and Xiaodong Lin. A comprehensive survey on smart contract construction and execution: paradigms, tools, and systems. *Patterns*, 2(2):100179, 2021.
30. Yizhong Liu, Meikang Qiu, Jianwei Liu, and Meiqin Liu. Blockchain-based access control approaches. In *IEEE CSCloud 2021*, pages 127–132. IEEE, 2021.
31. Yizhong Liu, Jianwei Liu, Marcos Antonio Vaz Salles, Zongyang Zhang, Tong Li, Bin Hu, Fritz Henglein, and Rongxing Lu. Building blocks of sharding blockchain systems: Concepts, approaches, and open problems. *Computer Science Review*, 46:100513, 2022.
32. Yizhong Liu, Jianwei Liu, Yiming Hei, Yu Xia, and Qianhong Wu. A secure cross-shard view-change protocol for sharding blockchains. In *ACISP 2021*, volume 13083, pages 372–390. Springer, 2021.
33. Yiming Hei, Jianwei Liu, Hanwen Feng, Dawei Li, Yizhong Liu, and Qianhong Wu. Making MA-ABE fully accountable: A blockchain-based approach for secure digital right management. *Comput. Networks*, 191:108029, 2021.
34. Yiming Hei, Dawei Li, Chi Zhang, Jianwei Liu, Yizhong Liu, and Qianhong Wu. Practical agentchain: A compatible cross-chain exchange system. *Future Gener. Comput. Syst.*, 130:207–218, 2022.
35. Yiming Hei, Yizhong Liu, Dawei Li, Jianwei Liu, and Qianhong Wu. Themis: An accountable blockchain-based P2P cloud storage scheme. *Peer-to-Peer Netw. Appl.*, 14(1):225–239, 2021.
36. Jun-feng Xie, Helen Tang, Tao Huang, F. Richard Yu, Renchao Xie, Jiang Liu, and Yunjie Liu. A survey of blockchain technology applied to smart cities: Research issues and challenges. *IEEE Communications Surveys and Tutorials*, 21(3):2794–2830, 2019.
37. Keke Gai, Jinnan Guo, Liehuang Zhu, and Shui Yu. Blockchain meets cloud computing: A survey. *IEEE Communications Surveys and Tutorials*, 22(3):2009–2030, 2020.
38. Roman Beck, Michel Avital, Matti Rossi, and Jason Bennett Thatcher. Blockchain technology in business and information systems research. *Bus. Inf. Syst. Eng.*, 59(6):381–384, 2017.
39. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
40. Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.
41. M. Qiu, H. Li, and E. Sha. Heterogeneous real-time embedded software optimization considering hardware platform. In *ACM SAC*, pages 1637–1641, 2009.
42. M. Qiu, E. Sha, M. Liu, M. Lin, S. Hua, and L. Yang. Energy minimization with loop fusion and multi-functional-unit scheduling for multidimensional DSP. *JPDC*, 68(4):443–455, 2008.
43. D. Qi, M. Liu, M. Qiu, and S. Zhang. Exponential synchronization of general discrete-time chaotic neural networks with or without time delays. *IEEE Trans. on Neural networks*, 21(8):1358–1365, 2010.
44. J. Li, M. Qiu, et al. Thermal-aware task scheduling in 3D chip multiprocessor with real-time constrained workloads. *ACM Trans. on Embedded Comp. Sys. (TECS)*, 12(2):1–22, 2013.

45. Z. Shao, M. Wang, et al. Real-time dynamic voltage loop scheduling for multi-core embedded systems. *IEEE Trans. on Circuits and Systems II*, 54(5):445–449, 2007.
46. M. Qiu, Z. Ming, et al. Three-phase time-aware energy minimization with dvfs and unrolling for chip multiprocessors. *J. of Systems Architecture*, 58(10):439–445, 2012.
47. L. Tao, S. Golikov, K. Gai, and M. Qiu. A reusable software component for integrated syntax and semantic validation for services computing. In *IEEE Sym. on Service-Oriented System Eng.*, pages 127–132, 2015.
48. K. Zhang, J. Kong, M. Qiu, and G. Song. Multimedia layout adaptation through grammatical specifications. *Multimedia Systems*, 10(3):245–260, 2005.
49. M. Qiu, K. Zhang, and M. Huang. Usability in mobile interface browsing. *Web Intelligence and Agent Systems*, 4(1):43–59, 2006.
50. L Zhang, M Qiu, W. Tseng, and E. Sha. Variable partitioning and scheduling for mpsoc with virtually shared scratch pad memory. *J. of Signal Proc. Sys.*, 58(2):247–265, 2018.
51. M. Qiu, E. Khisamutdinov, et al. Rna nanotechnology for computer design and in vivo computation. *Philosophical Transactions of the Royal Society A*, 2013.
52. M. Qiu, K. Gai, and Z. Xiong. Privacy-preserving wireless communications using bipartite matching in social big data. *FGCS*, 87:772–781, 2018.
53. M. Qiu, H. Su, M. Chen, Z. Ming, and L. Yang. Balance of security strength and energy for a pmu monitoring system in smart grid. *IEEE Comm. Magazine*, 58(5):142–149, 2012.
54. H. Qiu, T. Dong, et al. Adversarial attacks against network intrusion detection in IoT systems. *IEEE Internet of Things J.*, 8(13):10327–10335, 2020.
55. X. Gao and M. Qiu. Energy-based learning for preventing backdoor attack. In *KSEM (3)*, pages 706–721, 2022.

An Investigation of Blockchain-based Sharding

Jiahong Xiao[1,2],Wei Liang[1,2], Jiahong Cai[1,2], Hangyu Zhu[1,2,3], Xiong Li[4], and Songyou Xie[5]

[1]School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China
[2]Hunan Key Laboratory for Service computing and Novel Software Technology, Xiangtan 411201, China
[3]Guangdong Financial High-tech Zone "Blockchain +" Fintech Research Institute, Foshan 528253, China
[4]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
[5]College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
xiaoiaong@163.com, wliang@hnust.edu.cn, cjphil@163.com, a2770848109@163.com, lixiongzhq@163.com, syxie@hnu.edu.cn

Abstract. Nowadays, blockchain distributed ledger technology is becoming more and more prominent, and its decentralization, anonymization, and tampering obvious features have been widely recognized. These excellent technical features of blockchain have also made it a hot issue for global research. With the wide application of blockchain technology in various industries, some defects are gradually exposed, and more prominently, the blockchain system is unable to meet the current demand of explosive growth of data volume and frequent data interaction. As one of the key technologies to solve this problem, sharding technology is gaining attention. This article introduces common blockchain scaling schemes and focuses on an overview of blockchain sharding. Sharding technology is introduced from two perspectives of intra-slice consensus and inter-slice consensus. The current mainstream slicing technology is summarized according to three different slicing methods: network sharding, transaction sharding, and state sharding. Finally, the challenges faced by current blockchain sharding technology are analyzed and the full text is summarized.

Keywords: Blockchain, Scaling, Sharding, Consensus Algorithm, Capacity Extension.

1 Introduction

The digital economy has become a major component of today's world, and blockchain as new technology is gaining increasingly widespread attention worldwide and has penetrated into various aspects of people's production life, such as the Internet of Things [1][2], finance [3-5], energy [6-8], health care [9][10], and many other fields [11]. Blockchain [12] is a new application technology with technical features such as distributed data storage [13-15], decentralization, peer-to-peer transactions, and dy-

namic encryption algorithms [16][17]. However, the performance [18-20] of current decentralized systems based on blockchain technology is low and cannot afford the data processing under big data. Blockchain sharding technology is proposed as an important solution for blockchain sharding.

In 2016, Luu et al. [21]proposed an Elastico algorithm to combine the sharding method of expansion in the database with blockchain technology to improve the efficiency of transaction processing and reduce the cost of transactions. Since then, researchers from all walks of life have conducted a lot of research on sharding technology in terms of consensus mechanisms, distributed ledgers, and sharding methods. What we call sharding is not an innovative concept, but the essence of the idea is to divide the data in the database into small data shards [22], and then store these shards in different locations. This will avoid generating a large number of data access requests in a short period of time, thus overprocessing the transactions of one server. In traditional blockchain networks, transactions must be confirmed by each node in the network so as to ensure the security of transactions, but this is one of the difficulties in improving the speed of transactions. In contrast, the application of sharding technology in blockchain systems is to divide the blockchain network into several subnetworks, each of which will contain some nodes, and new transactions will be randomly assigned to individual shards for processing. Based on the sharding problem in the blockchain sharding scheme, this article introduces the research progress of blockchain sharding technology and gives a comprehensive introduction to sharding-related technologies.

The rest of this paper is as follows: Chapter 2 introduces blockchain technology and common blockchain scaling schemes; Chapter 3 categorizes blockchain slicing schemes based on consensus algorithms; Chapter 4 introduces three current mainstream slicing approaches and provides theoretical references for further research and development of blockchain slicing technology; Chapter 5 outlines the current challenges faced by blockchain slicing technology; Chapter 6 summarizes the paper.

2 Related Technology Introduction

Blockchain technology is increasingly developing into a mature digital economy infrastructure [23], which is essentially a bookkeeping method, usually using a general ledger with peer-to-peer storage [24], and has a wide range of applications in many fields [25][26]. In this article, we present blockchain in a six-layer model, which are the application layer, contract layer, incentive layer, consensus layer, network layer, and data layer.

According to the six-layer architecture model of blockchain, its sharding schemes are divided into two main categories: on-chain scaling and off-chain scaling [27][28]. On-chain expansion is the optimization and improvement of the basic structure, model, and algorithm of the blockchain from the data layer, network layer, consensus layer, and incentive layer of the blockchain, including changing the size of the block, sharding, changing the consensus mechanism, isolating the witness, increasing the capacity of the blockchain, and other aspects. The off-chain expansion is to build a

transaction network on another layer other than the main chain, which is an adjustment of the contract and application layers of the blockchain. Placing some complex work under the chain and returning the result to the chain reduces the workload on the chain and improves the performance of the blockchain, including state channels [29], side chaining [30], cross-chain, and off-chain computing. Blockchain scaling techniques are shown in Table 1.

3 Consensus Algorithm-based Sharding Technology

The essence of a blockchain is a distributed application, and the first problem in solving a distributed system is to reach a consensus among multiple independent nodes, i.e., all of them have to be consistent. The most common consensus algorithms are the proof-of-work (PoW) mechanism [31], proof-of-stake (PoS) mechanism [32], and delegated proof-of-stake (DPoS) mechanism [33]. However, high-performance consensus protocols used in distributed databases cannot be applied to blockchains, and secondly, blockchains rely on BFFs, which have been shown to lead to low scalability performance [34].

Table 1. Blockchain System Architecture.

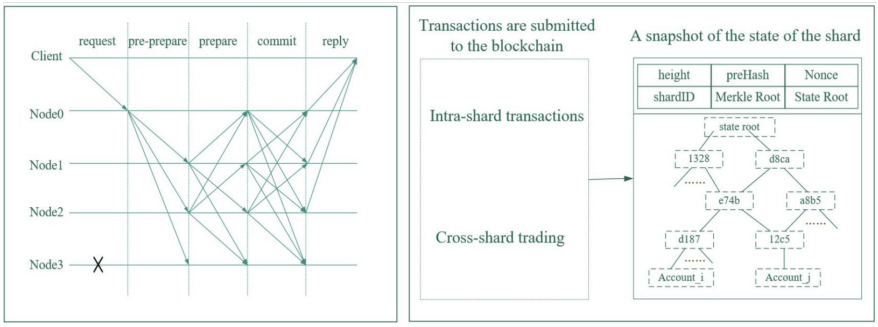
Heading level	Example	Font size and style
On-chain expansion	Data Layer	Change block size
		Isolation testimonial
	Network layer	Sharding
	Consensual level	Pos, PBFT
Off-chain expansion	Tier 2 improvements	Status channel
		Side chain
		Cross-chain
		Below the chain calculation

3.1 Intra-slice Consensus Protocol

The consensus protocol is divided into intra-slice consensus and cross-slice consensus. Intra-slice consensus requires each node in the same slice to agree and broadcast according to the slice's protocol and finally results in a consensus result for the whole slice. In 1999, The PBFT proposed by Castro et al. [35] is the earliest BFT algorithm, in which the whole system works if more than 2/3 of the nodes are normal. In 2016, Miller et al. [36] proposed the HoneyBadgerBFT algorithm, which is the first asynchronous BFT algorithm that can guarantee its effectiveness without time constraints.

Figure 1 presents two different on-slice consensus flows. The consensus protocol used for blockchain sharding is based on the Practical Byzantine Consensus Protocol, and Figure 1(a) depicts the flow of the intra-slice consensus mechanism based on PBFT, describing how a request is then passed between nodes in a distributed system

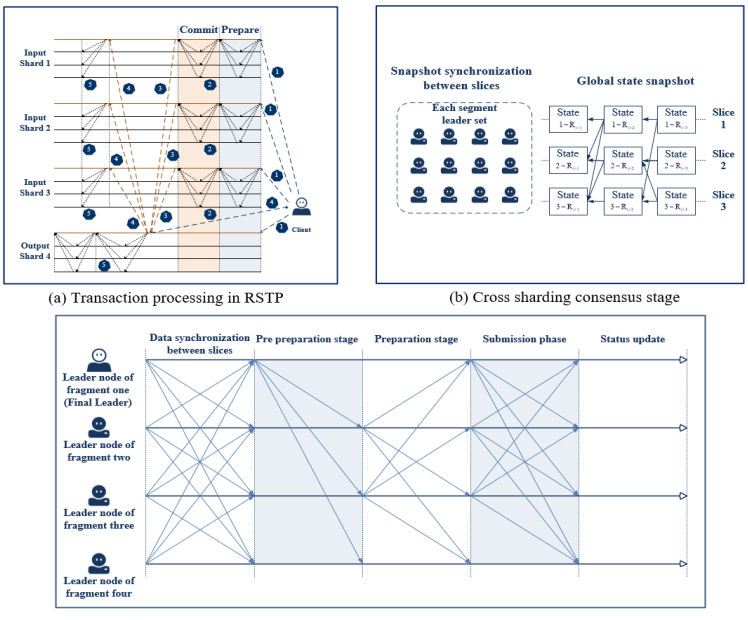
running the PBFT protocol. PBFT is widely used in many blockchain systems, such as Zilliqa, and Hyperledger, where each consensus of consensus nodes for a block requires receiving and forwarding multiple blocks. Figure 1(b) gives the intra-slice consensus phase under the two-layer slice consensus protocol under the federated chain, in which the task is to gather the intra-slice and cross-slice transactions forwarded to the slice into a single block and submit it to the blockchain after completing the consensus.



(a) Operation process of practical Byzantine fault tolerance protocol (b) Consensus stage in shards

Fig. 1. Intra-slice consensus.

3.2 Cross-Slice Consensus Protocol



(a) Transaction processing in RSTP (b) Cross sharding consensus stage

(c) Cross shard consensus process

Fig. 2. Cross-slice consensus.

In traditional blockchain systems, communication between all nodes is required to maintain identical copies of the blockchain, which leads to relatively low performance of the blockchain system. Now all the slices are facing the challenge of cross-slice transaction processing and need to handle data from multiple slices. To prevent the problem of double spending [37], all the slices that involve cross-slice transactions need to execute a multi-stage protocol to confirm the transactions.

There are three main approaches to cross-slice consensus, which are transaction atomization, transaction centralization, and the use of class routing protocols. Figure 2 describes three different cross-slice consensus processes. Figure 2(a) depicts the RSTP protocol, which efficiently handles intra-slice and cross-slice transactions by invoking FBFT [38]. In RSTP, the operations of input and output slices are separated. Figure 2(b) depicts the two-layer slice consensus protocol mechanism [39], which is based on the principle of committing transactions to the blockchain before updating them using state snapshots, which in turn enables the final atomicity of cross-slice transactions. In the intra-slice public phase, all transactions are submitted to the blockchain, and the intra-slice submitted blocks generate a new snapshot of the partition state. In the cross-slice consensus phase, the snapshots of the slice state generated by all the slices are synchronized to the DAG global state snapshot, and then the global state snapshot is updated for the cross-slice transactions. The consensus flow of the cross-slice consensus phase is shown in Figure 2(c).

4 Mainstream Sharding Technology

4.1 Network Sharding

The sharding technique was first proposed by Wang et al. [40]. The major challenge of the blockchain sharding protocol is that the vast majority of transactions in the blockchain system are cross-slicing, and these cross-slicing transactions not only reduce the throughput of the system but also increase the cost of transaction processing. Network sharding is the most basic sharding method, which divides the nodes in a blockchain network into individual shards, and each network independently performs consensus and processes transactions, thus improving the throughput of the blockchain. Network sharding involves the problem of how to ensure the security of on-chain transactions after slicing [41], and network slicing usually takes into account factors such as slicing size, slicing security, and slicing method, and the process of slicing is usually implemented using random functions. Sharding divides the entire blockchain network into several small subnetworks, and each slice can independently and simultaneously build consensus and process different transactions in the network in parallel to increase the throughput and reduce the latency of the blockchain.

Zhou [42] proposed an NRSS sharding scheme to reduce the difference in performance among slices through a node scoring strategy; Cai et al. [43] proposed a multi-objective optimization algorithm (MaOEA-DRP) based on a dynamic reward and penalty mechanism to optimize the slice validation effectiveness model and then obtain an optimal blockchain slice scheme; Since most shard systems use a Byzantine

Fault Tolerance (BFT) based consensus protocol as their intra-committee protocol, the communication overhead is $O(n^2)$. Zhang et al. [44] proposed Skychain, a new blockchain framework based on dynamic partitioning that achieves a good balance between performance and security in a dynamic environment without compromising scalability, and it has a communication overhead of $O(\log n)$; Kokoris-Kogias et al. [45] proposed OmniLedger, a novel horizontally scalable distributed ledger that maintains long-term security under permissionless operation; Zamani et al. [46] proposed RapidChain, the first sharding-based public blockchain protocol that is resistant to Byzantine errors, and RapidChain use efficient cross-slice transaction validation techniques to avoid passing transactions across the network; Liu et al. [47] proposed FleetChain, a secure and scalable fractional blockchain consisting of FBFT and RSTP that uses a star network for both intra- and cross-fractional communication. The computation, communication, and storage complexity of FleetChain are all $O(n/m)$, achieving an optimal scaling factor of $O(n/\log n)$; Manuskin et al. [48] proposed Ostraka, a blockchain node architecture that slices the nodes themselves, allowing the nodes in the network to scale and with minimal overhead. In this algorithm, block validation consists of block header validation, where most operations are performed in $O(1)$, and transaction validation, where operations are in $O(n)$.

4.2 Transaction Sharding

Transaction sharding is based on network sharding, where different transactions are divided into different slices based on certain rules to reach an agreement. All the sliced networks are able to perform consensus and process transactions at the same time and assign some highly related transactions to the same slice, thus reducing the transaction processing overhead. The idea is to allocate transactions in the slices according to specific rules, both to achieve parallel processing and to avoid double-spending attacks. Transaction sharding focuses on which transactions are to be assigned to which slices. Transaction sharding allows individual network slices to have greater processing power over transactions. Although transaction sharding can improve the operational efficiency of the network to a certain extent, it cannot fundamentally solve the resource deficiencies in the network.

Nguyen et al. [49] proposed a new sharding paradigm-Optchain based on OmniLedger, which utilizes a lightweight dynamic transaction placement method that assigns related transactions to the same slice, minimizing transactions across slices, significantly reducing the confirmation time required for transactions, and increasing throughput. Its average computation time takes only $O(n)$. Castro et al. [35] proposed Zilliqa, which runs consensus mechanisms at a higher frequency within each slice to significantly increase throughput by processing transactions in parallel, but the scheme does not slice the blockchain data and is vulnerable to attacks; Liu et al. [50] proposed a secure and scalable hybrid consensus (SSHC) and fair slice selection (FSS) with the strict proof scheme that designs a transaction batching mechanism for the corresponding slice to handle transactions across the slice, thus reducing the number of calls to the BFT algorithm.

4.3 State Sharding

In blockchain systems, state sharding is a common solution, also known as data sharding. It classifies the data of the whole network according to the state, and each slice keeps only part of the data, not the whole blockchain. State sharding is the most ideal sharding method, and only by implementing state sharding can we essentially solve the problem of public chain scalability. At the same time, state sharding is the most challenging sharding scheme so far, which stores completely different ledgers in each shard, and the whole sharding network forms a complete ledger.

Chen et al. [51] proposed SSChain, a new non-reconfiguration structure that supports transaction slicing and state slicing; Wang et al. [52] proposed Monoxide, which introduces a human-specific asynchronous consensus that enables full slicing; Huang et al. [53] proposed BrokerChain, a state slicing design based on the account \ balance of cross-slicing blockchain protocol. In the success case, the computing complexity of cross-shard verification is O(1); Zilliqa can realize transaction slicing and state slicing, based on Zilliqa, Harmony [54] can not only split transaction validation like Zilliqa but also split data state, and its slicing process ensures high security and makes up for some of the shortcomings of Zilliqa.

5 The Challenges of Sharding Technology

The current development of blockchain has entered a bottleneck period, unable to meet the needs of large-scale application scenarios in the era of big data, and a large part of the bottleneck faced by blockchain technology today is due to the need to go to distributed consensus, while resource scarcity also makes blockchain systems have significant performance defects. The introduction of sharding can fundamentally reduce the resources required by nodes and is the most effective on-chain scaling solution. However, although the sharding technology has solved the problem of blockchain scalability [55] and improved the performance of transaction processing, there are still many areas for improvement, and still face many challenges.

The main challenges within the sharding are as follows: the security threat is by far the most important issue facing the blockchain. When sharding divides the tasks of the whole network into n different shards, the arithmetic power is also allocated to the corresponding shards, so for a single shard, only 1/n of the original arithmetic power is available, and at this time the difficulty of launching a 51% attack on a single shard is reduced to 1/n of the original one. This leads to a significant decrease in the security of the system. Under the PoW consensus, the blockchain mainly faces the 51% attack problem. That is, the blockchain can be tampered with and forged as long as it has more than 51% of the computing power on the network.

PBFT allows less than (n-1)/3 number of failed nodes and does not require multiple blocks to determine, so there are multiple sharding projects that have chosen the PBFT consensus mechanism within sharding. However, because PBFT consensus has the fault tolerance of (n-1)/3 nodes, 100 nodes cannot fully secure the slice. A witch attack is when a node disguises itself as n nodes and falsely claims that it stores n

copies of data, thus weakening the redundant backup role of the data. The PBFT itself cannot prevent the witch attack, so it needs to be prevented with other means.

The challenges between slices are as follows: Because digital information is easily replicated, double-splash attacks must be taken into account regardless of decentralization. Double-splash attacks within slices can be prevented using traditional methods, but in the UTXO model, an attacker can create multiple transactions with the same input but different outputs to perform a double-splash attack. In order to avoid "double spend attacks", the system must communicate a lot across slices, which increases the complexity of the system and also degrades its performance of the system.

6 Conclusion

Blockchain scalability is a hot topic right now, and sharding technology is one of the effective ways to solve this problem. Without reducing decentralization, sharding is the most likely solution to achieve high-performance on-chain scalability. This article firstly gives an overview of blockchain slicing-related technologies based on blockchain sharding consensus protocol, and classifies blockchain consensus algorithms from two aspects: intra-chip consensus and cross-chip consensus; secondly, it summarizes and outlines the existing blockchain sharding technologies from three aspects: network sharding, transaction sharding, and state sharding, and to help readers quickly understand blockchain sharding. Finally, with the development of blockchain sharding technology, more efficient blockchain slicing schemes will emerge in the future, and these results will further improve the blockchain technology ecology based on the sharding mechanism.

References

1. Xu, Zisang, et al. "A Time-sensitive Token-Based Anonymous Authentication and Dynamic Group Key Agreement Scheme for Industry 5.0." IEEE TII (2021).
2. Xie Y, Liang W, Li RF, et al., An in-vehicle CAN signal packing algorithm for connected vehicle environment[J]. Journal of Software,2016,27(09):2365-2376.
3. Chen, Yan, and Cristiano Bellavitis. "Blockchain disruption and decentralized finance: The rise of decentralized business models." J. of Busi. Vent. Insights 13 (2020): e00151.
4. Y. Li, K. Gai, et al, "Intercrossed access controls for secure financial services on multimedia big data in cloud systems", ACM Trans. on Multi. Comp., Comm., and App., 2016
5. K. Gai, M. Qiu, S. Elnagdy, “A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance,” IEEE BigDataSecurity 2016
6. Chen, Zuo, et al. "Trust-aware and low energy consumption security topology protocol of wireless sensor network." Journal of Sensors 2015 (2015).
7. M. Qiu, C. Xue, Z. Shao, E. Sha, “Energy minimization with soft real-time and DVS for uniprocessor and multiprocessor embedded systems,” IEEE DATE Conf., 1-6, 2007
8. M. Qiu, C Xue, Z Shao, et al., "Efficient algorithm of energy minimization for heterogeneous wireless sensor network", IEEE EUC, 25-34, 2006
9. Peng, Li, et al. "Improved low-rank matrix recovery method for predicting miRNA-disease association." Scientific reports 7.1 (2017): 1-10.

10. F. Hu, S. Lakdawala, et al., Low-power, intelligent sensor hardware interface for medical data preprocessing, IEEE TITB, 13 (4), 656-663, 2009
11. H. Qiu, Q. Zheng, et al., "Topological graph convolutional network-based urban traffic flow and density prediction", IEEE Trans. on ITS, 2020
12. K. Gai, Y. Zhang, M. Qiu, B. Thuraisingham, “Blockchain-enabled Service Optimizations in Supply Chain Digital Twin”, IEEE Transactions on Service Computing, 2022
13. Wang Jing, Zhang Chong, Liang Wei, Liu Xiangyang. Localized restoration coding based on Pyramid codes in distributed storage systems[J]. Journal of Electronic Measurement and Instrumentation,2017,31(09):1481-1487.DOI:10.13382/j.jemi.2017.09.020.
14. M. Qiu, Z. Chen, Z. Ming, X. Qin, J. Niu, "Energy-aware data allocation with hybrid memory for mobile cloud systems", IEEE Systems J., 11 (2), 813-822, 2014
15. J. Li, Z. Ming, et al., "Resource allocation robustness in multi-core embedded systems with inaccurate information", Journal of Systems Arch. 57 (9), 840-849, 2011
16. Conoscenti, Marco, Antonio Vetro, and Juan Carlos De Martin. "Blockchain for the Internet of Things: A systematic literature review." IEEE/ACS AICCSA, 2016.
17. H. Qiu, T. Dong, et al., "Adversarial attacks against network intrusion detection in IoT systems," IEEE Internet of Things Journal 8(13), 10327-10335, 2020
18. J. Niu, Y. Gao, M. Qiu, Z. Ming, "Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability", JPDC, 72(12), 1565-1575, 2012
19. M. Qiu, H. Li, E. Sha, "Heterogeneous real-time embedded software optimization considering hardware platform", ACM sym. on Applied Comp., 1637-1641, 2009
20. M. Qiu, Z. Jia, et al., "Voltage assignment with guaranteed probability satisfying timing constraint for real-time multiproceesor DSP", J. of Signal Proc. Systems, 2007
21. Luu, Loi, et al. "A secure sharding protocol for open blockchains." Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. 2016.
22. Liu, Yimeng, Yizhi Wang, and Yi Jin. "Research on the improvement of MongoDB Auto-Sharding in cloud environment." 7th IEEE ICCSE, 2012.
23. Liang, Wei, et al. "Secure data storage and recovery in industrial blockchain network environments." IEEE Transactions on Industrial Informatics 16.10 (2020): 6543-6552.
24. Wang, Shuai, et al. "Blockchain-enabled smart contracts: architecture, applications, and future trends." IEEE Trans. on SMC: Systems 49.11 (2019): 2266-2277.
25. Gai, Keke, et al. "Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks." IEEE Internet of Things Journal 6.5 (2019): 7992-8004.
26. P Kumar, R Kumar, et al., “PPSF: a privacy-preserving and secure framework using blockchain-based machine-learning for IoT-driven smart cities”, IEEE Transactions on Network Science and Engineering 8 (3), 2326-2341, 2021.
27. Yu, Hui, Zhang, Zongyang,and Liu, Jianwei." Research on bitcoin blockchain scaling technology." Computer Research and Development 54.10 (2017): 2390-2403.
28. Zeng, Shuai, et al. "Blockchain scaling for bitcoin: key technologies, constraints and derived problems." J. of Auto. 45.06 (2019): 1015-1030. doi:10.16383/j.aas.c180100.
29. Zhang, Fan, et al. "Federated Learning Meets Blockchain: State Channel based Distributed Data Sharing Trust Supervision Mechanism." IEEE IoT Journal (2021).
30. Singh, Amritraj, et al. "Sidechain technologies in blockchain networks: An examination and state-of-the-art review." J. of Network and Computer App. 149 (2020): 102471.
31. Vukolić, Marko. "The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication." Int’l workshop on open problems in network security. Springer, Cham, 2015.
32. Kang, Jiawen, et al. "Incentivizing consensus propagation in proof-of-stake based consortium blockchain networks." IEEE Wireless Comm. Letters 8.1 (2018): 157-160.

33. Luo, Yinghui, et al. "A new election algorithm for DPos consensus mechanism in blockchain." 2018 7th international conference on digital home (ICDH). IEEE, 2018.
34. Dang, Hung, et al. "Towards scaling blockchain systems via sharding." Proceedings of the 2019 international conference on management of data. 2019.
35. Castro, Miguel, and Barbara Liskov. "Practical byzantine fault tolerance." OsDI. Vol. 99. No. 1999. 1999.
36. Miller, Andrew, et al. "The honey badger of BFT protocols." Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. 2016.
37. Karame, Ghassan O., et al. "Misbehavior in bitcoin: A study of double-spending and accountability." ACM Trans. on Info. and System Security (TISSEC) 18.1 (2015): 1-32.
38. Liu, Yizhong, et al. "Fleetchain: A secure scalable and responsive blockchain achieving optimal sharding." Int'l Conf. on Algo. and Arch. for Parallel Proc.. Springer, 2020.
39. Luo Nankun. Research and system implementation of blockchain sharding technology in federated chain scenario [D]. University of Electronic Science and Technology, 2021. doi:10.27005/d.cnki.gdzku.2021.005445.
40. Wang, Gang, et al. "Sok: Sharding on blockchain." Proceedings of the 1st ACM Conference on Advances in Financial Technologies. 2019.
41. Zhang, Shiwen, et al. "A novel blockchain-based privacy-preserving framework for online social networks." Connection Science 33.3 (2021): 555-575.
42. Yang Yifan Zhou. Research on the improvement of blockchain slicing strategy [D]. Tianjin University,2019.DOI:10.27356/d.cnki.gtjdu.2019.002400.
43. Cai, Xingjuan, et al. "A sharding scheme-based many-objective optimization algorithm for enhancing security in blockchain-enabled industrial internet of things." IEEE Transactions on Industrial Informatics 17.11 (2021): 7650-7658.
44. Zhang, Jianting, et al. "Skychain: A deep reinforcement learning-empowered dynamic blockchain sharding system." 49th IEEE ICPP. 2020.
45. Kokoris-Kogias, Eleftherios, et al. "Omniledger: A secure, scale-out, decentralized ledger via sharding." 2018 IEEE Symposium on Security and Privacy (SP). 2018.
46. Zamani, Mahdi, Mahnush Movahedi, and Mariana Raykova. "RapidChain: A Fast Blockchain Protocol via Full Sharding." IACR Cryptol. ePrint Arch. 2018 (2018): 460.
47. Liu, Yizhong, et al. "Fleetchain: A secure scalable and responsive blockchain achieving optimal sharding." Int'l Conf. on Algo. and Arch. for Parallel Proc., Springer, 2020.
48. Manuskin, Alex, Michael Mirkin, and Ittay Eyal. "Ostraka: Secure blockchain scaling by node sharding." IEEE Euro. Sym. on Security and Privacy (EuroS&PW), 2020.
49. Nguyen, Lan N., et al. "Optchain: optimal transactions placement for scalable blockchain sharding." IEEE 39th Int'l Conf. on Distributed Computing Systems (ICDCS), 2019.
50. Liu, Yizhong, et al. "SSHC: A secure and scalable hybrid consensus protocol for sharding blockchains with a formal security framework." IEEE Transactions on Dependable and Secure Computing (2020).
51. Chen, Huan, and Yijie Wang. "Sschain: A full sharding protocol for public blockchain without data migration overhead." Pervasive and Mobile Computing 59 (2019): 101055.
52. Wang, Jiaping, and Hao Wang. "Monoxide: Scale out blockchains with asynchronous consensus zones." 16th USENIX NSDI, 2019.
53. Huang, Huawei, et al., "BrokerChain: A Cross-Shard Blockchain Protocol for Account/Balance-based State Sharding." IEEE INFOCOM. 2022.
54. Harmony Team, Technical Whitepaper. Harmony. Acce-ssed: Feb. 11, 2020. [Online]. Available: https://harmony.one/whitepaper.pdf
55. Croman, Kyle, et al., "On scaling decentralized blockchains." International conference on financial cryptography and data security. Springer, Berlin, Heidelberg, 2016.