MRI20 Y3 Appendix2 amended 2023-10

1 A Latent Variable Based Approach for Exploring Geographic Datasets.8pp.pdf
2 Crime-avoiding Routing Navigation.11pp.pdf
3 Integrating Location Information as Geohash Codes in Convolutional Neural Network-Based Satellite Image Classification.7pp.pdf
4 Spatiotemporal Model of Real Estate Valuation Trends.15pp.pdf
5 Towards Real-time House Detection in Aerial Imagery Using Faster Region-based Convolutional Neural Network.9pp.pdf

1. A Latent Variable Based Approach for Exploring Geographic Datasets.8pp

LVRF: A Latent Variable Based Approach for Exploring Geographic Datasets

Deng, Liangdong; Mahara, Arpan; Adjouadi, Malek; and Rishe, Naphtali

Abstract: ***Geographic datasets are usually accompanied by spatial non-stationarity – a phenomenon that the relationship between features varies across space. Naturally, non-stationarity can be interpreted as the underlying rule that decides how data are generated and alters over space. Therefore, traditional machine learning algorithms are not suitable for handling non-stationary geographic datasets, as they only render a single global model. To solve this problem, researchers often adopt the multiple-local-model approach, which uses different models to account for different sub-regions of space. This approach has been proven efficient but not optimal, as it is inherently difficult to decide the size of sub-regions. Additionally, the fact that local models are only trained on a subset of data also limits their potential. This paper proposes an entirely different strategy that interprets non-stationarity as a lack of data and addresses it by introducing latent variables to the original dataset. Backpropagation is then used to find the best values for these latent variables. Experiments show that this method is at least as efficient as multiple-local-model-based approaches and has even greater potential.***

Index Terms: ***Back-propagation, Geographically Weighted Regression (GWR), Latent Variable, Machine Learning Algorithm, Non-stationary, Random Forest***

Manuscript received April 10, 2023.

This material is based in part upon work supported by the National Science Foundation under Grant Nos. MRI20 CNS-2018611, MRI CNS-1920182, by FDEP Grant C-2104, and DHS Grant E2055778.

Liangdong Deng, Arpan Mahara, and Naphtali Rishe are with the School of Computing and Information Sciences, Florida International University, USA. Correspondence email is liadeng@cs.fiu.edu.

Malek Adjouadi is with the Department Electrical and Computer Engineering, Florida International University, USA.

1. Introduction

Geographic data is defined as information that is implicitly or explicitly associated with a location on the surface of the Earth [1]. With advancements in remote sensing technologies and the widespread use of GPS-enabled devices, the number of available physical and human geography datasets has vastly increased in recent years [2]. These data are studied and utilized for social good, such as mitigating damages caused by natural disasters [3], discovering mineral resources [4], preventing crimes [5], improving traffic conditions [6], and many other scenarios.

However, when dealing with geographic datasets, researchers find that many traditional machine learning algorithms do not perform very well due to the presence of non-stationarity. In such data, the relationship between features does not necessarily remain the same everywhere, meaning the underlying model that governs the data changes over space. To address this issue, a natural solution is to replace the global model with many local models. Each local model is only responsible for describing a much smaller region within which the data is supposed to be relatively stationary. Most studies that have taken this approach (such as [7], [8] and [9]) have observed significantly better results compared to traditional algorithms, which are not specifically designed to handle non-stationarity.

These multiple-local-model based approaches all face similar challenges. First, the dataset used to train local models is only a subset of all available data. Previous research has shown that the accuracy of a model is strongly correlated with the amount of data used to train this model. There can be a

significant decrease in model performance if the training data size drops below a certain threshold [10]. Second, determining the size of sub-regions to which local models correspond is difficult. A larger size means more data can be used to train local models, but the region is more likely to exhibit non-stationary. Conversely, a smaller size implies the opposite. As a result, compromise is always necessary.

Our insight is that the source of non-stationarity can be explained as a lack of data, i.e., some dimensions of the data are not being collected. For example, a crime dataset could exhibit strong non-stationarity, as crime patterns in New York could be fundamentally different from those in Washington DC. Even within New York, it is hard to imagine that Brooklyn shares the same crime pattern as Manhattan. Ultimately, these differences are caused by various factors such as household income, population composition, culture, and the number of police officers per capita, among others. If one were able to collect data on every single aspect of an area, the dataset would ultimately become stationary. This theory is also in accordance with the fact that non-stationarity is quite often observed in human geography datasets but rarely found in physical geography data. Since physical geography data – which is generated by Earth's natural processes - has fewer determining factors and is usually simpler to collect, it is less prone to non-stationarity. In contrast, human geography data focuses on human activities and is much more complex. Even seemingly simple datasets can have countless deciding factors that are impossible to collect comprehensively. For example, house sale price data generally includes features of the house itself and its nearby areas, but other factors - such as school, traffic, population, and crime - are usually not included, even though they are important and would certainly affect the pricing model. The lack of these data would then be observed as non-stationarity in the dataset and would impact the final model in some way.

Based on this insight, we propose an entirely different strategy that addresses non-stationarity by introducing latent variables to the original dataset. These latent variables would account for all the missing factors that not collected by the original dataset but observable as non-stationarity. Theoretically, assuming we have unlimited calculating power, the optimal values of the latent variables could be easily found through a brute-force search of the entire vector space. However, this solution is obviously impossible due to the tremendous size of the vector space. Thus, inspired by neural networks, we use a back-propagation algorithm to find the optimal values of the latent variable. Experiments demonstrate that this new approach can build models as accurate as the state-of-the-art algorithms while offering the potential for further improvement.

2. Background and Study Area

2.1 Background

The first renowned method for exploring spatial non-stationarity, known as Geographically Weighted Regression (GWR), was proposed by Brunsdon, Fotheringham, and Charlton in 1996 [7]. The "main characteristic of GWR is that it allows regression coefficients to vary across space, and so the values of the parameters can vary between locations" [11]. The motivation for inventing GWR was that "a single global model cannot explain the relationship between some sets of variables" [7]. To address non-stationarity, GWR allows relationships between features and labels to differ across spaces. The basic idea of how GWR works is to learn a regression equation for every feature in the dataset, during which dependent and explanatory components are accounted for by examining neighboring data points. The neighbors contribute differently to this process according to their distance, which is why it is called a "weighted" regression. The closer a data point is, the more weight it is assigned. This design complies with Tobler's first law of geography, "everything is related to everything else, but near things are more related than distant things" [12]. Later, in 2002, Brunsdon further improved this algorithm to Semiparametric GWR (SGWR) [13], which allows some features to have fixed regression equations across space, while others can still be variable.

Due to the success of GWR, many later

studies followed this multiple-local-model design. One example is Multiscale GWR (MGWR), which was introduced in 2017 by Fotheringham, Yang, and Kang. This method "is similar in intent to Bayesian nonseparable spatially varying coefficients (SVC) models, although potentially providing a more flexible and scalable framework in which to examine multiscale processes" [9]. It improves upon GWR in a way that not only adapts to datasets on different levels of non-stationarity but also provides the necessary information to evaluate the scales of different processes. The latest research using this approach is Geographical Random Forest (GRF), proposed by Stefanos, Tais, et al. in 2019. It adopts Random Forests [14] as the base algorithm to create local models. The principle idea of this method is the "disaggregation of RF into geographical space in the form of local sub-models" [8], which is basically another version of the multiple-local-model approach.

In conclusion, all these methods are directly or indirectly based on the multiple-local-model approach and consequently suffer from the same problems mentioned in the previous section. In this work we propose a completely different approach with the goal of better understanding and accounting for the intrinsic nature of non-stationarity.

2.2 Study Area

We selected housing sales data from King County, US as the target study area (obtained from [15]). The dataset contains 21,613 records, with each record being a real estate transaction that occurred between May 2014 and May 2015, a period during which the housing market remained relatively stable in King County.

In this dataset, there are 20 features related to the house's location (latitude, longitude, zip code), its basic information (size, number of stories and rooms, garage, air conditioning), and transaction-related information (sale date and price). Some of the features have missing values. This is not a problem for our algorithm, which is based on the Random Forests algorithm and can handle missing values. However,

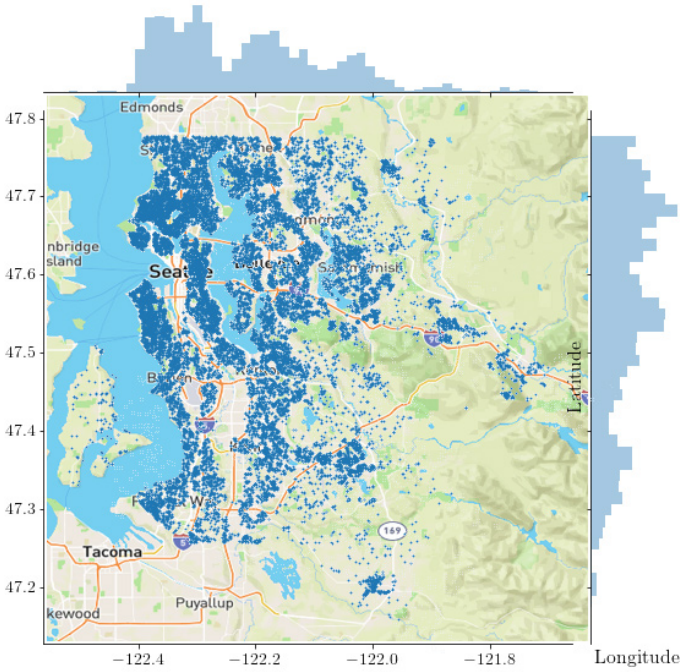


Figure 1: Distribution of the King County housing data.

some other algorithms we use for performance comparison are incapable of doing this. Therefore, during the data preparation stage, we fill in the missing values with the average value of that column.

The goal of this dataset is to build a predictive model that can estimate house sale prices, given the house's location and some of its basic information. It is a well-researched topic that has been studied for a long time. However, even state-of-the-art algorithms in this area still have ample room for improvement due to the complicated nature of this task. Additionally, it is a very typical human geography dataset in which data availability varies depending on the amount of human activity. Figure 1 shows the distribution of the dataset on the map. As depicted in the figure, the downtown area in Seattle is populated with data, with some areas left blank which are mostly parks or commercial zones. Rural regions have much less data scattered all over the place. The fact that this dataset is distributed extraordinarily unevenly across the space presents additional challenges when using the previously mentioned multiple-local-model approach, as local models which correspond to rural areas will have fewer train-

ing samples, leading to inaccurate results. In urban areas, overcrowded data points will only bring marginal improvement to models built for that area.

Another issue with this dataset is that the house sale price spans over a fairly large range with a long tail, as shown in figure 2, which is undesirable. To eliminate the tail, we convert Price to log(Price), which follows the normal distribution and is a much better target variable to deal with.

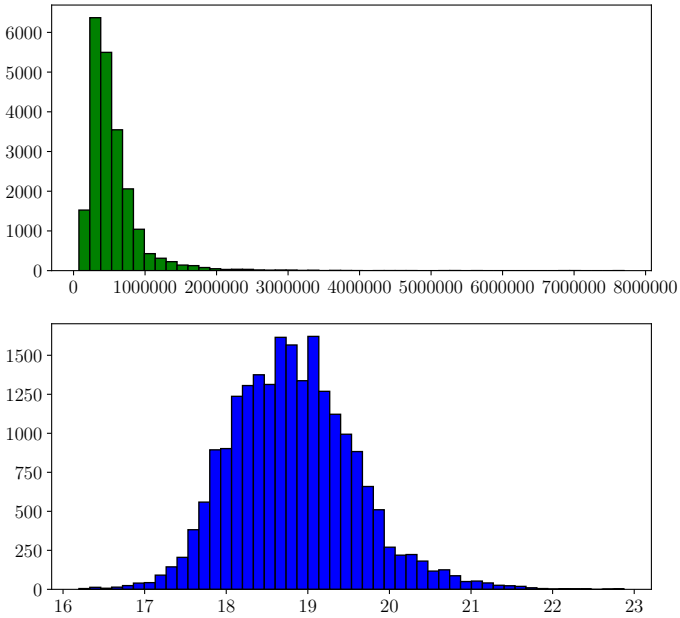


Figure 2: Distribution of Price vs. log(Price)

3. Latent Variable Random Forests

In this section, we provide a detailed description of the key designs of the Latent Variable Random Forests as follows.

3.1 Key Design of the Latent Variable

By introducing a new latent variable, we aim to use it to represent the hidden factors that cause non-stationarity. In our housing price model example, it would be a combination of various unknown factors that could affect how house prices should be modeled. For instance, the security level of a community obviously has an impact on house value. Although we don't have any information on which area is more secure and which is not, its influence on the sale price will be observable via non-stationarity. It is important to note that the target variable might be affected by multiple hidden factors such as security, traffic, nearby schools, and so on. But no matter how many hidden factors there are, they will influence the target variable together. It is impossible to know which factor has a larger impact. Fortunately, we don't need to care about that. Our primary focus is on how these hidden factors as a whole would affect the target variable we want to predict.

To better describe the problem, let $(f_1, f_2, ..., f_n)$ denote the features in the dataset and t denote the target variable to be modeled and predicted. After adding a latent variable lv, the feature vector becomes $F(lv) = (f_1, f_2, ..., f_n, lv)$. Thus, the task is converted to finding the best $\vec{lv}$ that makes the model trained from $F(lv)$ (using a predetermined regular machine learning algorithm) achieve the highest accuracy.

The vector space $\vec{lv}$ is obviously unlimited. Thus we introduce a value range of $[0, 1]$ to lv and define a minimum step interval of 0.01. The reason why we limit the value range to $[0, 1]$ is that the value range of lv actually doesn't play an important role in the final model. If lv is multiplied by 2, the resulting model will still be the same. So, only the relative value matters and is what we should care about. Also, during the machine learning stage, all the features of the original dataset need to be standardized and normalized anyway, thus a standardized lv will, in fact, benefit the entire procedure. For the minimum step, the smaller it is, the more fine-grained the final model would be. However, setting it too small will also considerably increase the calculation time and may not be worth the marginal return. So we recommend setting it to 0.01 as a balance between speed and accuracy.

Theoretically, the value array of latent variable $\vec{lv}$ can be inferred by an exhaustive brute-force search of the entire vector space. The time complexity of doing so is as follows:

$$O(n) = (\frac{R}{S})^n * (T_{train} + T_{test}) \qquad (1)$$

where n is the number of data points in the dataset, R is the value range, S is the step size,

T_{train} and T_{test} are the time needed for training and testing the model, respectively. Note that the value of n is usually very large. Even for a very small dataset, n will probably be greater than 1000. Thus, this brute-force method is completely impractical considering the amount of calculation needed.

3.2 Grid Based Latent Variable System

To solve the time complexity problem, we clearly need a smarter algorithm, for example, a heuristic search, which could greatly reduce the search space. But before that, let's examine the possibility of reducing the size of the potential vector space, which would greatly benefit the entire procedure even if a heuristic search is to be adopted.

Here we introduce a grid-based latent variable system. Let $(x_{min}, x_{max}, y_{min}, y_{max})$ denote the minimum bounding box that contains the entire dataset. A step size of s will evenly divide the space into this many grids:

$$G(s) = \lceil \frac{x_{max} - x_{min}}{s} \rceil * \lceil \frac{y_{max} - y_{min}}{s} \rceil \quad (2)$$

For each intersection of the grid system, we assign an *Influence Center* (abbreviated as IC) to it. For a data point with a coordinate of (x, y), we first determine which $grid$ it is located in. Then calculate its latent variable value from all the nearby ICs located at the four corners of $grid$. Here we use an inverse distance weighted method to combine the values from nearby ICs, in accordance with the idea that nearby ICs should have a stronger influence on the latent variable than remote ones. The detailed formula is as follows:

$$v(x, y) = \frac{\sum_{i=1}^{N} W(IC_i)V(IC_i)}{\sum_{i=1}^{N} W(IC_i)} \quad (3)$$

where $W(IC_i)$ is the weight for the ith influence center which equals the inverse of the Euclidean distance between the data point and the IC.

This design simulates how the hidden factors create non-stationarity in the dataset. No matter what hidden factors there are, as a general rule, they would affect nearby data points more than remote ones. Thus we simulate this procedure by introducing the concept of Influence Centers and making them impact nearby records in a similar way. Another benefit brought by this design is that now the search space is greatly reduced down to the number of ICs. Instead of finding the best values for all the records, we only need to optimize the values for ICs now, which is way less than the total number of records.

3.3 Random Forests as the Base Algorithm

Before proceeding, we still need to decide which base machine learning algorithm is to be used to train models. Here, our choice is the Random Forests [14] algorithm. As suggested in the name, Random Forests will create many randomly generated decision trees to perform the prediction task together. For classification tasks, the final result would be a majority vote of results from all the decision trees. For regression, this would be an average of all results. The core idea of RF is to create a bagging procedure where the variance of the model is decreased but the bias remains unchanged, thus generating a better result from sub-optimal models.

There are multiple reasons why we choose RF as our base algorithm. First, RF is based on decision trees which are naturally good at handling coordinates in geographic datasets. Then, Random Forests is among the top machine learning algorithms available and often shows exceedingly good results when handling spatial data, as proven by [16] and [17]. We will be able to inherit all of these advantages by using RF as the base algorithm.

3.4 Back Propagation

With a reduced search space, the time complexity is still massive as we are only replacing $(\frac{R}{S})^n$ in Formula 1 with X^n (X is the total number of influence centers) if a brute-force search is to be used. Thus we must find a way to further reduce the search space, i.e., a heuristic-search like method.

Here, inspired by the backpropagation algorithm in Neural Networks [18], we have de-

signed a backpropagation process to search for the best values for influence centers, as detailed in Algorithm $BackPropagation()$. In this function, a learning rate α is introduced, which determines how fast the backpropagation converges. A large value will cause $BackPropagation()$ to converge faster, but the generated result will be more likely to be coarse-grained and thus less than optimal. Conversely, a smaller value will converge slower but produce better results. Generally speaking, the best α value is recommended to be set to the smallest value within acceptable training time.

```
Function BackPropagation()
    Initialize IC_Array
    while IC_Array has not converged do
        foreach IC in IC_Array do
            foreach learn_rate in [α, -α] do
                IC_new = IC + learn_rate
                if Trained model sees improvement in accuracy then
                    IC = IC_new
                else
                    continue
                end
            end
        end
    end
    return IC_Array
end
```

The *converge* condition in the *BackPropagation()* algorithm is a bit tricky. Ideally, if *IC_Array* remains the same after an iteration, the algorithm is considered converged as future iterations will produce the same results. However, this does not necessarily happen as *IC_Array* may always change slightly with pretty much the same results. So, we insert a process at the end of each iteration, which will evaluate the test accuracy under the current *IC_Array*. If the test accuracy does not improve for more than 5 iterations, we consider the algorithm converged and stop the backpropagation iteration. Although this extra calculation slows down the entire algorithm, it is worth the cost.

3.5 Prediction

The prediction process is relatively simple. After the *IC_Array* is returned by *BackPropagation()*, the final *Model* is trained from the original dataset plus the latent vector generated from *IC_Array*. When predicting an unknown observation, the latent variable is first calculated by using the inverse distance weighted method from Formula 3. Then, *Mode* is applied to get the final prediction result.

3.6 Assessment Measurements and Results

One thing that wasn't mentioned in the previous sections is that a proper assessment measurement must be chosen. This actually plays an important role in the algorithm, as the evaluation result generated by the measurement will be used to determine how the backpropagation process runs and guide it to generate a better result for each iteration. Some of the most commonly used measurements are [19]: mean absolute error (MAE), mean squared error (MSE), and root mean squared error (RMSE). In our case, MAE is preferred as the other ones will penalize large errors and cause bias in our algorithm.

Now that the algorithm is complete, we have run LVRF on the King County housing dataset and achieved an MAE of 0.263. As a comparison, we also experimented with unmodified Random Forests on the same dataset and obtained a result of 0.289. This means that the learned latent variables were able to offset some of the non-stationarity and made it easier for the standard RF to generate a more accurate model. To compare with the others, we also evaluated the same dataset using two state-of-the-art algorithms, RFsp [20] and MGWR [9], which are specifically designed to handle geographic datasets and non-stationarity. The results for RFsp and MGWR were 0.261 and 0.272, respectively. These results suggest that the idea of using latent variables to capture hidden factors that cause non-stationarity is at

least as effective as the best results achieved using the multiple-local-model approach.

4. Conclusion

This paper presents LVRF, a machine-learning algorithm that can create predictive models for non-stationary geographic datasets. Unlike other algorithms, LVRF adopts a latent variable based approach, instead of the widely used multiple-local-model strategy. Experiments show that LVRF can build models as accurately as state-of-the-art algorithms while avoiding the common disadvantages of the multiple-local-model approach. First, LVRF establishes grid-based influence centers. The latent variable value of any data point is decided by the nearby influence centers using an inverse distance weighted method. Then it uses a backpropagation algorithm to train the values of the influence centers until they converge. To predict unknown observations, the data point's latent variable is calculated from the converged influence centers, and fed into the model with its other features.

The insight of LVRF is that the design of the influence center can mimic the hidden factors which affect nearby data points in different ways depending on the location. By learning these hidden factors with a backpropagation algorithm and then including them in the model creation stage, the impact brought by non-stationarity will be offset. This approach allows for a single global model to be used to describe the features plus the hidden factors.

It is also worth mentioning that, although Random Forests is selected as the base algorithm, LVRF is capable of using any other regular machine learning algorithm as the base algorithm. Doing so may bring advantages in certain scenarios when there is preknowledge regarding the dataset.

References

[1] ISO/TC 211 committee, "ISO/TC 211," https://www.iso.org/committee/54904.html, 2011.

[2] C. Beath, I. Becerra-Fernandez, J. Ross, and J. Short, "Finding value in the information explosion," *MIT Sloan Management Review*, vol. 53, pp. 18–20, 06 2012.

[3] E. Spyrou and Y. Avrithis, "A region thesaurus approach for high-level concept detection in the natural disaster domain," vol. 4816, 12 2007, pp. 74–77.

[4] P. Partsinevelos and Z. Mitraka, "Change detection of surface mining activity and reclamation based on a machine learning approach of multi-temporal landsat tm imagery," *Geocarto International*, vol. 28, pp. 1–20, 01 2012.

[5] A. Wheeler and W. Steenbeek, "Mapping the risk terrain for crime using machine learning," 01 2020.

[6] Y. Meidan, M. Bohadana, A. Shabtai, J. D. Guarnizo, M. Ochoa, N. O. Tippenhauer, and Y. Elovici, "Profiliot: A machine learning approach for iot device identification based on network traffic analysis," in *Proceedings of the Symposium on Applied Computing*, ser. SAC '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 506–509. [Online]. Available: https://doi.org/10.1145/3019612.3019878

[7] C. Brunsdon, A. S. Fotheringham, and M. E. Charlton, "Geographically weighted regression: A method for exploring spatial non-stationarity," *Geographical Analysis*, vol. 28, no. 4, pp. 281–298, 1996. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1538-4632.1996.tb00936.x

[8] S. Georganos, T. Grippa, A. N. Gadiaga, C. Linard, M. Lennert, S. Vanhuysse, N. Mboga, E. Wolff, and S. Kalogirou, "Geographical random forests: a spatial extension of the random forest algorithm to address spatial heterogeneity in remote sensing and population modelling," *Geocarto International*, pp. 1–16, 2019.

[9] A. S. Fotheringham, W. Yang, and W. Kang, "Multiscale geographically weighted regression (mgwr)," *Annals of the American Association of Geographers*, vol. 107, no. 6, pp. 1247–1265, 2017.

[10] J. Morgan, R. Dougherty, A. Hilchie, and B. Carey, "Sample size and modeling accuracy with decision tree based data mining tools," *Acad Inf Manag Sci J*, vol. 6, 01 2003.

[11] J. Mateu, "Comments on: A general science-based framework for dynamical spatio-temporal models," *Test*, vol. 19, pp. 452–455, 11 2010.

[12] W. R. Tobler, "A computer movie simulating urban growth in the detroit region," *Economic Geography*, vol. 46, pp. 234–240, 1970. [Online]. Available: http://www.jstor.org/stable/143141

[13] A. Fotheringham, C. Brunsdon, and M. Charlton, "Geographically weighted regression: The analysis of spatially varying relationships," *John Wiley and Sons*, vol. 13, 01 2002.

[14] L. Breiman, "Random forests," in *Machine Learning*, 2001, pp. 5–32.

[15] Kaggle, "King County Housing Market data," https://www.kaggle.com/harlfoxem/housesalesprediction, 2016.

[16] A.-L. Boulesteix, S. Janitza, J. Kruppa, and I. R. König, "Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics," *WIREs*

Data Mining and Knowledge Discovery, vol. 2, no. 6, pp. 493–507, 2012. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1072

[17] M. Nussbaum, K. Spiess, A. Baltensweiler, U. Grob, A. Keller, L. Greiner, M. E. Schaepman, and A. Papritz, "Evaluation of digital soil mapping approaches with large sets of environmental covariates," *SOIL*, vol. 4, no. 1, pp. 1–22, 2018. [Online]. Available: https://soil.copernicus.org/articles/4/1/2018/

[18] S. Putra and A. Wanto, "Analysis of artificial neural network accuracy using backpropagation algorithm in predicting process (forecasting)," *International Journal Of Information System & Technology (IJISTECH)*, vol. 1, pp. 34–42, 11 2017.

[19] J. Hernandez-Orallo, C. Ferri, N. Lachiche, A. Martínez-Usó, and M. Ramírez-Quintana, "Binarised regression tasks: methods and evaluation metrics," *Data Mining and Knowledge Discovery*, vol. 30, 11 2015.

[20] T. Hengl, M. Nussbaum, M. N. Wright, G. B. Heuvelink, and B. Gräler, "Random forest as a generic framework for predictive modeling of spatial and spatio-temporal variables," *PeerJ*, vol. 6, p. e5518, Aug. 2018. [Online]. Available: https://doi.org/10.7717/peerj.5518

2. Crime-avoiding Routing Navigation.11pp

Crime-Avoiding Routing Navigation

Rishe, Naphtali; Sadjadi, Masoud; and Adjouadi, Malek

Abstract: ***Extensive prior work has provided methods for the optimization of routing based on the criteria of travel time and/or on the cost of travel and/or the distance traveled. A typical method of routing involves building a graph comprised of street segments, assigning a normalized weighted value to each segment, and then applying the weighted-shorted path algorithm to the graph in order to find the best route. Some users desire that the routing suggestion include consideration pertaining to the reduction of risk of encountering violent crime. For example, a user desires a leisure walk via a safe route from her hotel in an unknown city. Here we present a method to quantify such user preferences and the risks of encountering crime and to augment the standard routing methods by giving weight to safety considerations. The proposed method's advantages, in comparison to other crime-avoidance routing algorithms, include weighing crime types with respect to their potential detrimental value to the user, with temporal qualification and quantification of crime and its statistical aggregation at the geographic resolution down to a city block.***

Index Terms: Crime-avoidance, Crime classification, Crime data, Crime impact weighting, Multi-parametric routing, Navigation, Routing, Spatiotemporal analysis of crime

1. *Introduction*

Previous research [1-9] has developed methods for the optimization of routing based on the criteria of travel time and/or on the cost of travel and/or the distance traveled. Routing can be in various modalities, such as by car, on foot, by bicycle, via public transit, or by boat. A typical method of routing involves building a graph comprised of street segments, assigning a normalized weighted value to each segment, and then applying the weighted-shorted path algorithm to the graph in order to find the best route.

Routing can take into account preference parameters in addition to time and distance. For example, routing suggestions can include c consideration pertaining to the reduction of the risk of encountering violent crime. For example, a user desires a leisure walk via a safe route from her hotel in an unknown city. Here we present a method to quantify such user preferences and the risks of encountering crime and to augment the standard routing methods by giving weight to said safety considerations.

Galburn *et al.*[4] have utilized crime data to optimize the safety aspect of navigation within a city. Their case study involved urban crime data from Illinois and Pennsylvania. Their proposed risk model for the street network within a city facilitated estimating probabilities of criminal incidents that the traveler may encounter on any road segment. In their approach, the same importance is assigned to the path traversal time and the crime incident risk. Their method solves a dual-objective shortest-path problem.

Here we presented an improved method to co-optimize crime avoidance with other criteria. The proposed method's advantages, in comparison to other crime-avoidance routing algorithms, include weighing crime types with respect to their potential detrimental value to the user, with temporal qualification and quantification of crime and its statistical aggregation at the geographic resolution down to a city block.

The following figure shows traditional routing optimizing the time and/or distance.

Manuscript received February 19, 2023.

Naphtali Rishe is at the Knight Foundation School of Computing and Information Sciences, Florida International University, Miami, FL, USA (e-mail: rishe@cs.fiu.edu)

Masoud Sadjadi is at the Knight Foundation School of Computing and Information Sciences, Florida International University, Miami, FL, USA (e-mail: sadjadi@cs.fiu.edu)

Malek Adjouadi is at the Department of Electrical and Computer Engineering, Florida International University, Miami, FL, USA (e-mail: adjouadi@fiu.edu).

Correspondence email is rishe@cs.fiu.edu

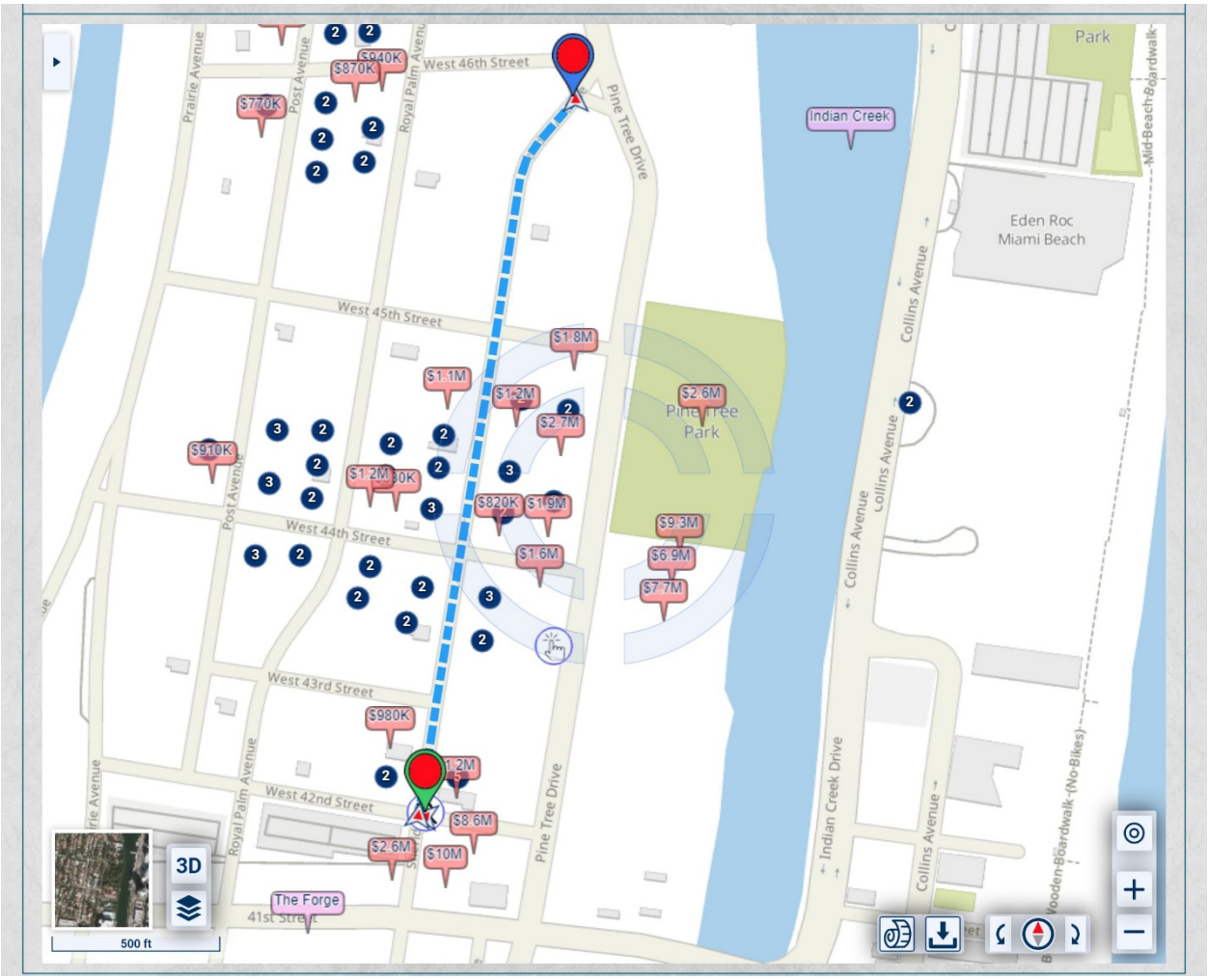


Figure 1: Routing that optimizes time and/or distance

Here we present an improved method to co-optimize crime avoidance with other criteria. The proposed method's advantages, in comparison to Galburn [4] and the other crime-avoidance routing algorithms, include:

(1) weighing crime types with respect to their potential detrimental value to the user,

(2) with temporal qualification,

(3) quantification of crime and its statistical aggregation at the geographic resolution down to a city block, and

(4) evaluation of the crime detriment to the user in each segment by considering the needs, exposure, and preferences of the user rather than merely considering the general crime incidence statistics. For example, violent crime committed outdoors have a higher impact, and severe violence, such as homicide in the street, have the highest impact. Crimes without a direct unrelated victim, such as code violations or embezzlement, have no impact on travelers. Pick-pockets have an impact on travelers in walking mode but minimal impact on travelers by car. Non-statutory rape may be of high concern to a woman walking alone. For each type of traveler and travel modality, the present method provides default formulas for the evaluation of crime detriment in each segment. Additionally, the user may modify the formula by assigning greater or lesser importance to various types of crimes.

2. Methodology

In order to quantify crime risks for each street segment, we count police reports that occurred close to that segment during a set period of time, e.g., a particular year of reference, counting only violent and property crimes of the type that would directly affect the traveler (e.g., exclude domestic violence, exclude insider trading, exclude code violations, exclude statutory rape) and can further assign weights to various crime crimes based on the impact it may have in the traveler. The following is an example of a query to a crime database for an area in mid-Miami Beach.

Criteria **Description offense=BATTERY, Date and time≥2018-01-01, Date and time≤2018-12-31**

Selection Criteria:		Try also:	Or fill in &
Description of offense	**=Battery**	any null non-null Accident Administrative Aggravated Alarm All Alrm Aoa Assault Atm Au Audible Battery Burglary Buy Card Cml Code Con Credit Criminal Death Del Detail Directed Dispute Distrb Disturbance Domestic Drug Equipment Events F False Florida Found Fraud From Hang Illegal Impersonation Import Incident Information Intimidation Larceny Lost Manuf Mnr Motor Narcotic Natural Non Offenses Open Or Order Other Others Over Person Poss Pret Property Rape Recovery Residence Residential Robbery Rsd Run S Sell Service Shoplifting Simple Stolen Stop Structure Susp Suspic Suspicious Swindle Theft To Traffic Trespassing Under Up Vandalism Veh Vehicle Viol Violations Warrant Watch Weapons	= BATTERY
Date and time the incident occured	**≥2018-01-01**	any null non-null ≥2021-06-01 ≥2021-12-01 ≥2022-01-01 ≥2022-06-01	≥ 2018-01-01
	≤2018-12-31		≤ 2018-12-31
Keywords in	**=**	any	=

Figure 2: A sample query to a crime database

The above query may result in a set of incidents shown in the following map.

Figure 3: Map of incidents

The following is a tabular output of the query:

Case number	Description of offense	Date and time the incident occured	Reported	Address where the incident occured	Police district	Clearance code description as reported by Officer	Business name involved in incident	Signal code description	Victim type description	Victim name(s)	Suspect name(s)	Arrestee name
MBc2018-00120592	Assault Or Battery OF LAW ENFOR-CEMENT OFFICERS, FIREF-IGHTERS, EMERGENCY MEDICAL CARE PROVIDERS, PUBLIC TRANSIT EMPLOYEES Or AGENTS, Or Other SPECIFIED OFFICERS; RECLA-SSIFI-CATION OF OFFENSES; MINIMUM SENTENCES	≤2018-12-19 18:40:57≥		1425 WASHI-NGTON AVE		APPROVED						
MBc2018-00054670	BATTERY-FELONY Battery	≤2018-05-18 06:48:59≥		600 ESPAN-OLA WAY		Closed						
MBc2018-00010991	BATTERY-FELONY Battery	≤2018-01-27 07:57:55≥		1409 WASHI-NGTON AVE		Closed No SOLVA-BILITY						
MBc2018-00061490	BATTERY-FELONY Battery	≤2018-06-08 02:39:47≥		1420 COLLINS AVE		1 - Closed N- A						

Figure 4: Report of incidents

The mid-Miami Beach area of the previous example did not have homicide reports during the sampling period. To see homicide reports, which should be considered with a higher weight than battery, we need to query an area further west:

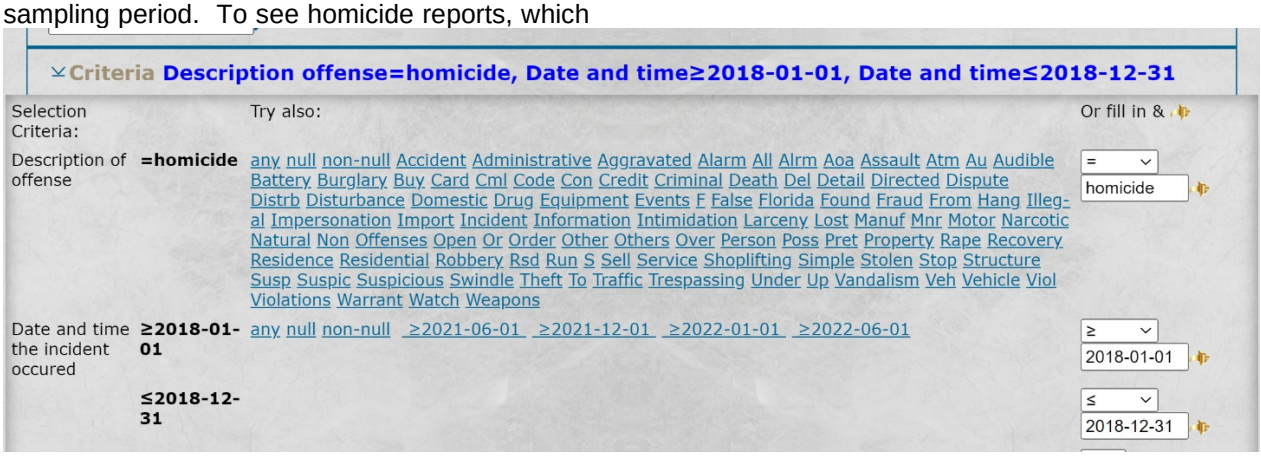


Figure 5: Homicide query

The results are shown in the following map and table.

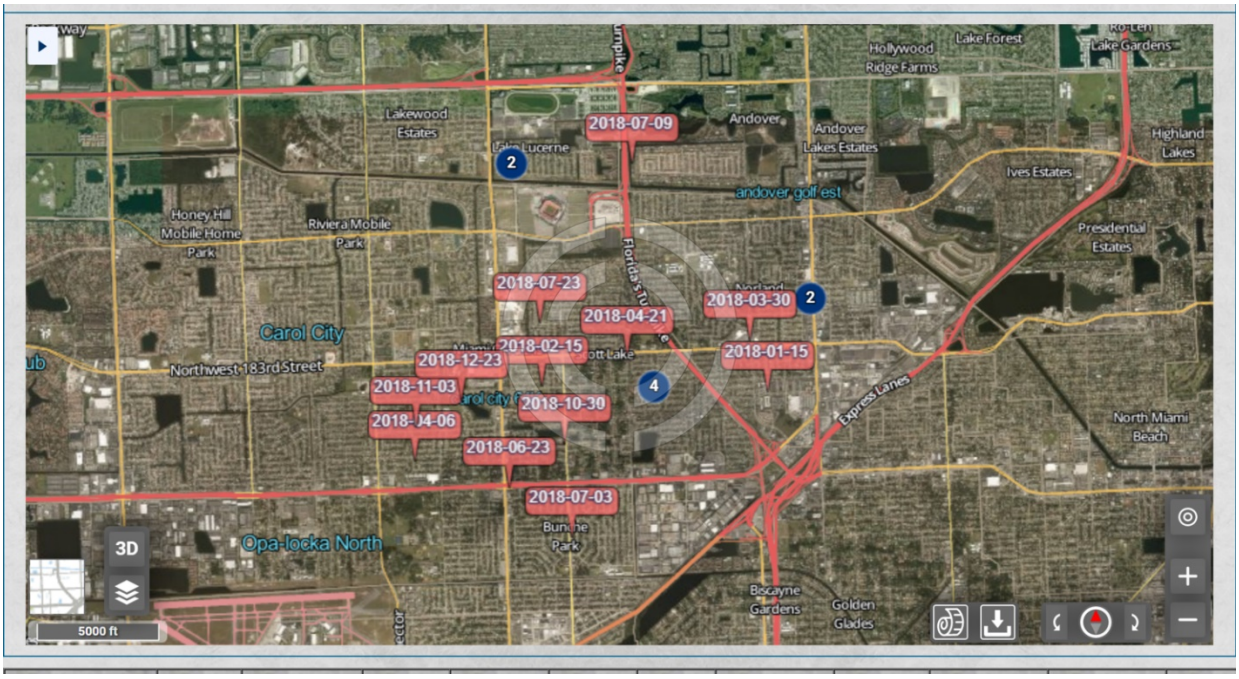


Figure 6: Map of homicide incidents

links to locations & details	Case number	Description of offense	Date and time the incident occured	Reported	Address where the incident occured	Police district	Clearance code description as reported by Officer	Business name involved in incident	Signal code description	Victim type description	Victim name(
1: 0.2±0.31 miles	MGc20-18007-150	Information - Accident Traffic HOMICIDE	≤2018-04-21≥		18249 NW 17TH AVE NW 183RD ST	Zone 11					
2: 2107'	MGc20-18009-665	Information - Accident Traffic HOMICIDE	≤2018-05-28≥		1621 NW 179TH ST	Zone 11					
3: 2883'	MGc20-18016-121	HOMICIDE-MURDER	≤2018-09-03≥		1300 NW 180TH TER	Zone 11					
4: 2890'	MGc20-18020-205	HOMICIDE-MURDER	≤2018-11-02≥		17701 NW 15TH CT	Zone 11					
5: 3137'	MGc20-18008-225	HOMICIDE-ATTEMPTED MURDER	≤2018-05-07≥		17730 NW 13TH CT	Zone 11					
6: 3383'	MGc20-18013-358	HOMICIDE-MURDER	≤2018-07-23≥		18700 NW 23RD AVE	Zone 31					
7:	MGc20-	HOMICIDE-	≤2018-		2335	Zone					

Figure 7: Table of homicide incidents

The importance of querying for only specific types of crime (and weighting them) is demonstrated by the following query, whose results are mostly crimes that have no bearing on the prospective traveler.

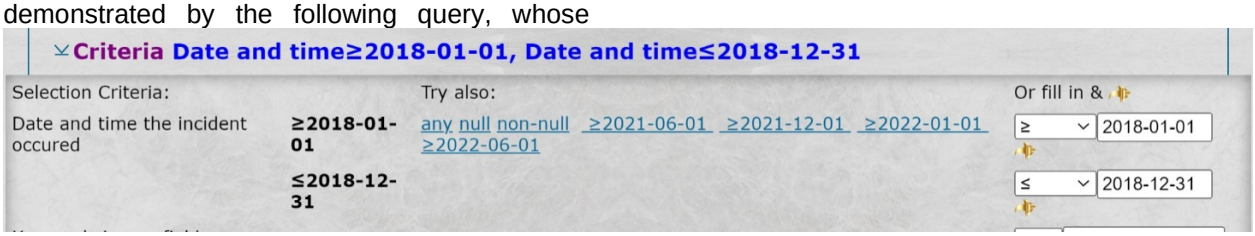


Figure 8: Query not restricting crime types

Figure 9: Map of the output of a query not restricting crime types

Case number	Description of offense	Date and time the incident occured	Reported	Address where the incident occured	Police district	Clearance code description as reported by Officer	Business name involved in incident	Signal code description	Victim type description	Victim name(s)	Suspect name(s)
MBc2018-00119260	PROHIBITED ACTS; PENALTIES	≤2018-12-15 01:13:19≥		200 30TH ST		Pending					
MBc2018-00116267	Larceny - Under $50.00 (+ ATT.)	≤2018-12-05 01:09:38≥		200 30TH ST		1 - Closed N- A					
MBc2018-00079230	Larceny - Under $50.00 (+ ATT.)	≤2018-08-01 10:52:39≥		200 30TH ST		Closed No SOLVA-BILITY					
MBc2018-00064680	FRAUD-ILLEG USE Credit CARDS	≤2018-06-18 10:56:58≥		2940 COLLINS AVE		Closed					
MBc2018-00115181	Larceny - $50 To $200	≤2018-12-01 08:53:53≥		2940 COLLINS AVE		Closed No SOLVA-BILITY					
MBc2018-00109397	MUNICIPAL ORDINANCE Viol	≤2018-11-10 12:32:44≥		2940 COLLINS AVE		Closed No SOLVA-BILITY					
MBc2018-00102292	FORCIBLE Rape COMMITTED	≤2018-10-18 14:46:55≥		2940 COLLINS AVE		APPROVED					
MBc2018-00118987	Assault AGG	≤2018-12-14 02:29:59≥		2940 COLLINS AVE		REVIEW					
MBc2018-00038646	Criminal MISCHIEF; PENALTIES; PENALTY FOR MINOR	≤2018-04-04 11:20:39≥		220 30TH ST		Closed					
MBc2018	BURGLARY	≤2018-04		2010		Open					

Figure 10: Tabular output of a query that does not restrict crime types, including crimes irrelevant for the traveler, e.g., credit card fraud

Turning back to routing, the following is a route optimizing travel time, which traverses segments where relevant crimes have occurred during the sampling period:

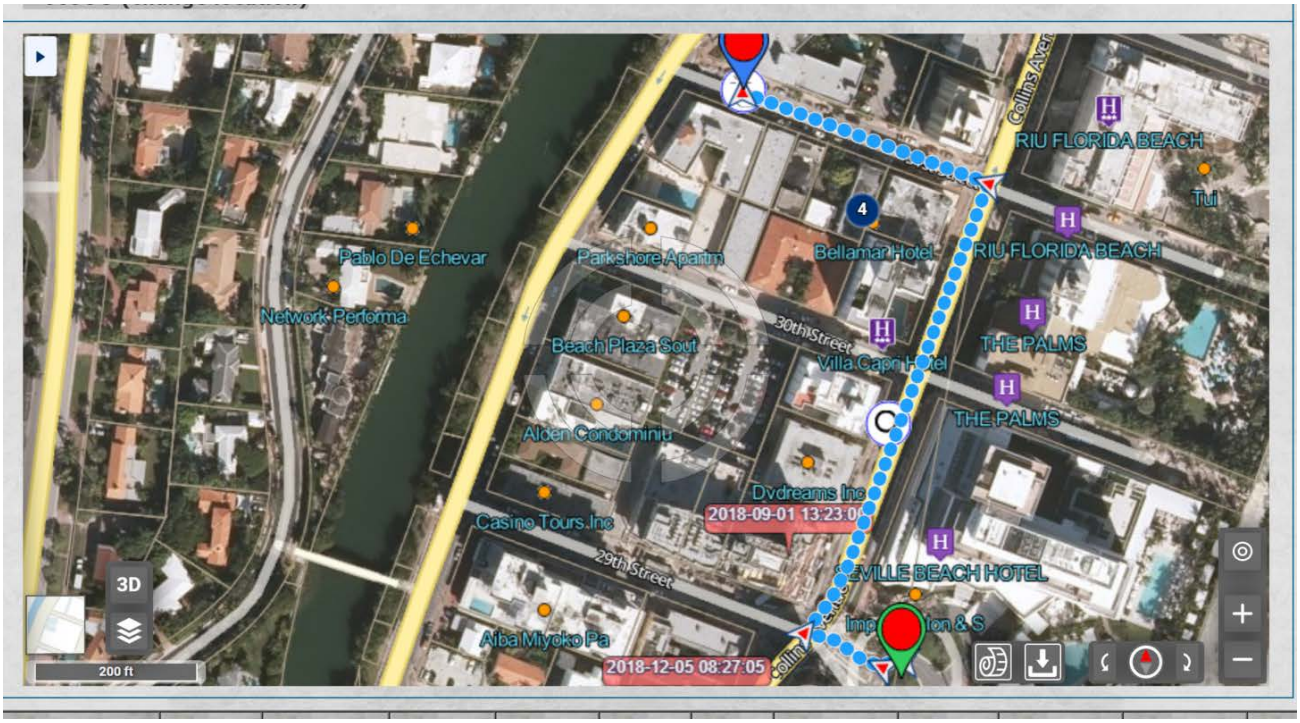


Figure 11: Time-optimized routing path, going through segments with higher crime potential

By co-optimizing the walk duration and crime encounter probability reduction, we get a slightly different route:

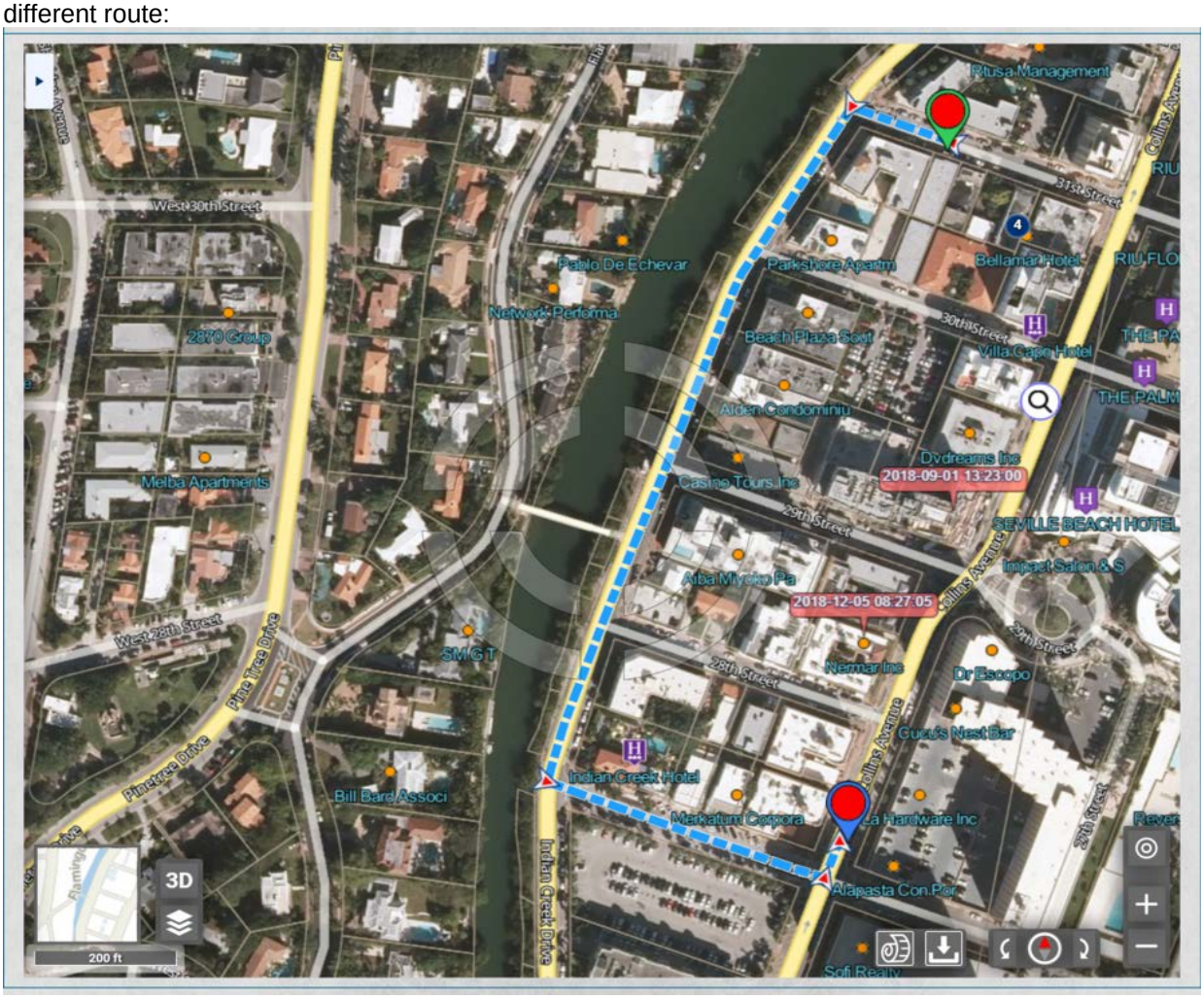


Figure 12: Routing co-optimizing time and crime avoidance

The routing may be different based on the mode of walking or transportation:

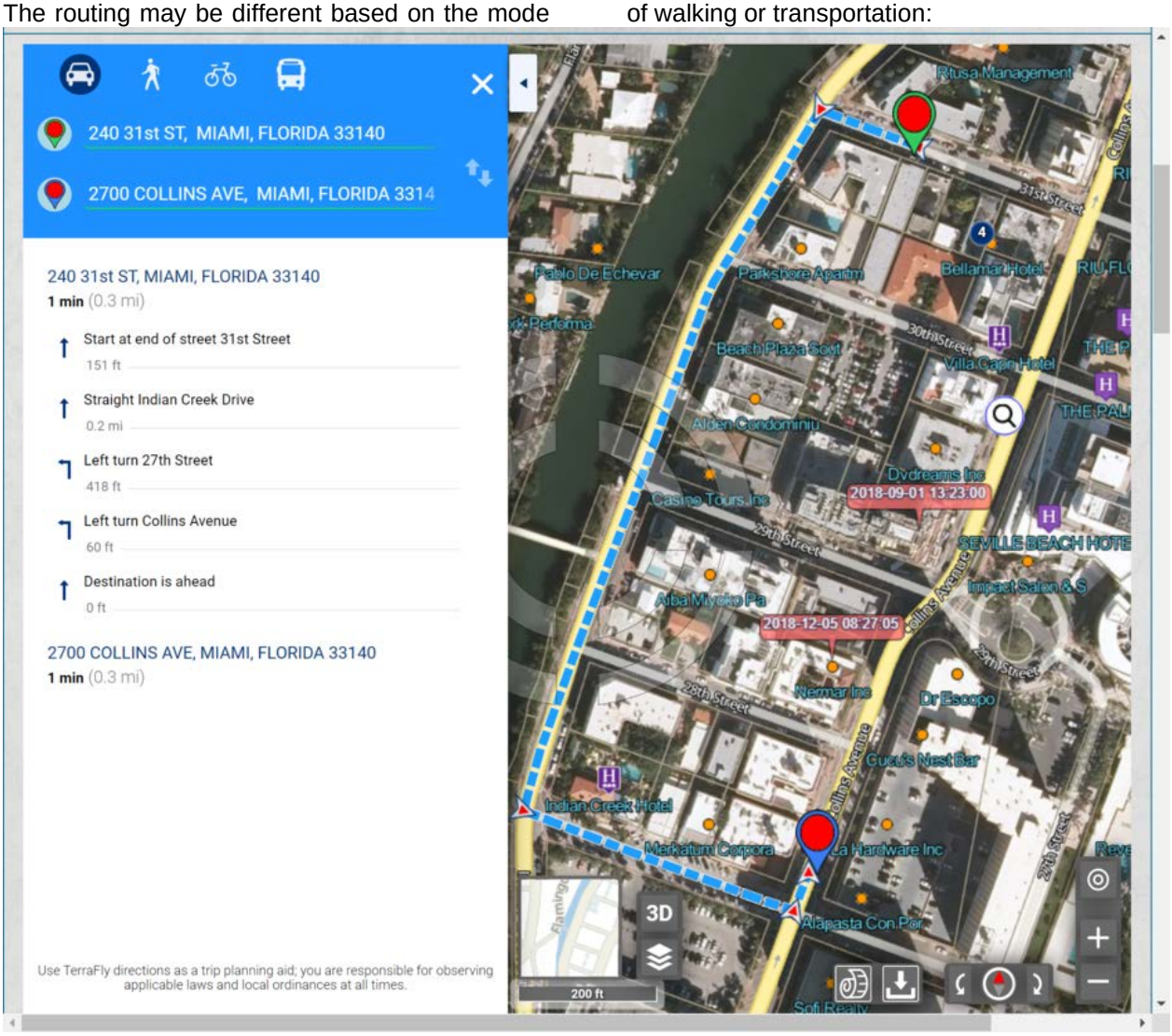


Figure 13: Time-optimized routing path, going through segments with higher crime potential, for various transportation modalities

The relative importance of time, cost of travel, and crime avoidance can be determined by the user utilizing a prior-art technology of weight selection triangle: a touchable triangle allows the user to assign importance weights to three interrelated decision optimization objectives using a single gesture [Oliver Ullrich, Naphtali Rishe, Daniel Luckerath. U.S. Patent US10061501B2 “User Interface for Co-Optimizing Weight Factors” issued on: August 28, 2018]:

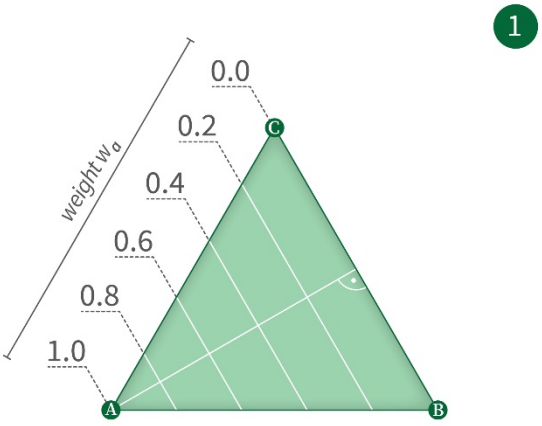


Figure 14: A weighting triangle with values along one side

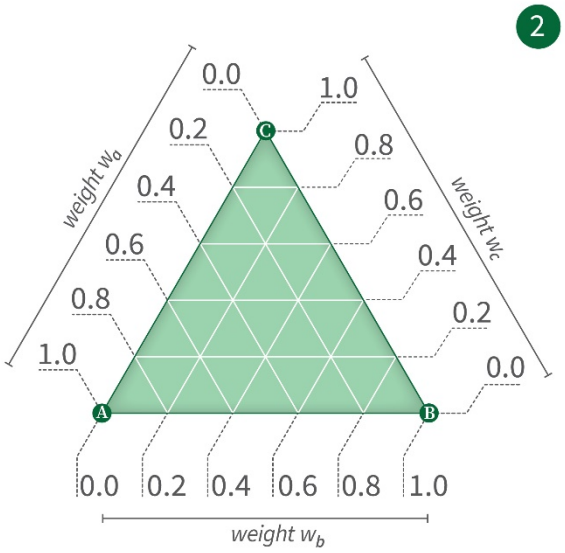


Figure 15: A weighting triangle with weighting values along all three sides

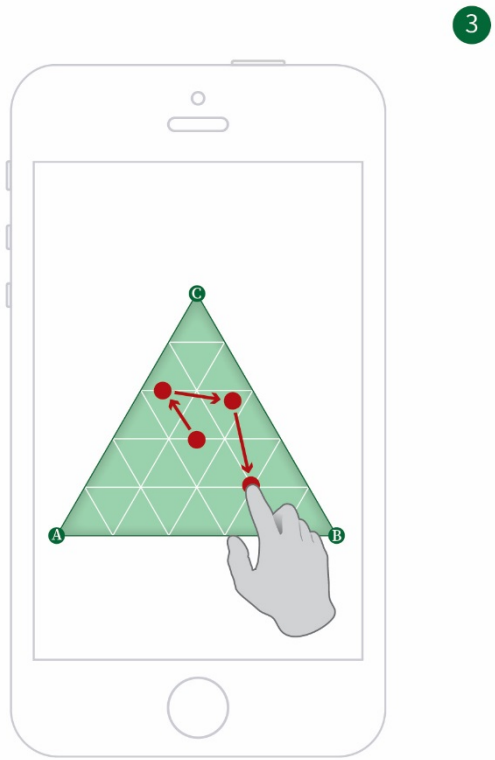


Figure 16: A smart device with the weighting triangle displayed thereon, showing a user selecting different weighting points

Applying said prior-art method to the herein proposed weighting selection problem, three objectives (*A=time*, *B=cost of travel*, and *C=crime avoidance*) are presented in a triangular fashion on a touch screen. Sub-figure 1 shows the underlying principle of the establishment of a single weight w_A for Objective *A*; Sub-figure 2 combines three objectives into a single triangle, allowing for the establishment of a tri-variable weight function *(w_A, w_B, w_C).* By applying a finger gesture, the user moves an indicator freely inside the triangle (see Sub-figure 3). The position of the indicator establishes a tri-variable weight function, which in further steps, is then used as input for a co-optimization algorithm. When the user is satisfied with the established weights, she indicates this, e.g., by pressing a touch screen button labeled “Go.”

Availability of Data and Materials

The data used in this work is available at http://terrafly.com. The geospatial data sets used in case studies to illustrate the method proposed herein can be provided by the corresponding author with appropriate arrangements.

Competing Interests

The authors declare that they have no competing interests.

FUNDING

The authors' effort resulting in this publication has been funded by the U.S. National Science Foundation (Grants CNS-2018611 and CNS-1920182), the Florida Department of Environmental Protection (Grant C-2104), and the U.S. Department of Homeland Security (Grant E2055778). The funding agencies had no role in the design of the study and collection, analysis, and interpretation of data, nor in writing the manuscript.

AUTHORS' CONTRIBUTIONS

Conceptualization: Rishe; Methodology: Rishe and Adjouadi; Investigation: Rishe, Sadjadi, and Adjouadi; Writing: Rishe and Sadjadi; Funding acquisition: Rishe, Sadjadi, and Adjouadi. All the authors of this paper concur with its content and consent to its publication.

ACKNOWLEDGMENT

Divya Saxena has helped in editing this manuscript. Liangdong Deng has helped with mapping technology.

REFERENCES

[1] E. Kanoulas, Yang Du, Tian Xia, and Donghui Zhang (2006). Finding Fastest Paths on A Road Network with Speed Patterns. In: 22nd International Conference on Data Engineering (ICDE'06), pp. 10-10, doi: 10.1109/ICDE.2006.71.

[2] Y.-J. Joo, S.-H. Kim (2011). A new route guidance method considering the pedestrian level of service using a multi-criteria decision-making technique. Journal of Korea Spatial Information Society, 19. pp. 83–91.

[3] Michael Shekelyan, Gregor Jossé, Matthias Schuber, Hans-Peter Kriegel (2014). Linear Path Skyline Computation in Bicriteria Networks. International Conference on Database Systems for Advanced Applications (DASFAA 2014). Lecture Notes in Computer Science, volume 8421, Springer, pp. 173-187.

[4] Esther Galbrun, Konstantinos Pelechrinis, Evimaria Terzi (2016). Urban navigation beyond shortest route: The case of safe paths. Information Systems, Volume 57, pp. 160-171.

[5] Hochmair, H. H. (2010). Spatial association of geotagged photos with scenic locations. In A. Car, G. Griesebner, & J. Strobl (Eds.), Geospatial Crossroads@GI_Forum '10: Proceedings of the geoinformatics forum Salzburg (pp. 91–100). Heidelberg: Wichmann.

[6] Lu, X., Wang, C., Yang, J. M., Pang, Y., & Zhang, L. (2010). Photo2Trip: Generating travel routes from geo-tagged photos for trip planning. In Proceedings of the international conference on Multimedia (pp. 143–152). New York City: ACM.

[7] Sun, Y., Fan, H., Bakillah, M., & Zipf, A. (2013). Road-based travel recommendation using geo-tagged images. Computers, Environment and Urban Systems.

[8] Alivand, M., Hochmair, H. and Srinivasan, S. (2015), "Analyzing how travelers choose scenic routes using route choice models," Computers, Environment and Urban Systems, Vol. 50, pp. 41–52.

[9] Mišković, S. and Stanimirović, Z. (2017), "Variable Neighborhood Search Based Heuristics for the Hard Capacitated k -facility Location Problem," IPSI Bgd Trans. Internet Res., pp. 1–8.

[10] Oliver Ullrich, Naphtali Rishe, Daniel Luckerath. U.S. Patent US10061501B2 "User Interface for Co-Optimizing Weight Factors" issued on: August 28, 2018.

Rishe [http://cake.fiu.edu/Rishe] is the inaugural outstanding university professor for Florida International University (FIU). He is a professor at FIU's Knight Foundation School of Computing and Information Sciences. Rishe is the Director of FIU's High-Performance Database Research Center. From 2013 to 2022, Rishe was the executive director of a multi-university Center for Advanced Knowledge Enablement sponsored by the National Science Foundation's industry-university cooperation program. Rishe's research has been funded by the U.S. Government and Industry at over $50 million. Rishe's inventions include 26 U.S. patents. Rishe is a Fellow medalist of the National Academy of Inventors. In 2021, Rishe received the IBM Global University Programs award. Rishe's publications include 440 papers and six books. Rishe's pioneering geospatial system TerraFly has been highlighted by the National Science Foundation in its reports to U.S. Congress.

Sadjadi received the B.S. degree in Hardware Engineering in 1995, the M.S. degree in Software Engineering in 1999, and the Ph.D. degree in Computer Science from Michigan State University in 2004. Dr. Sadjadi is currently an Associate Professor in the Knight Foundation School of Computing and Information Sciences at Florida International University, where he has been on the faculty since 2004. He is the Director of the Center of Partnership for International Research & Education (PIRE) funded by the National Science Foundation for $2.3 million. He is also the Director of the Autonomic Cloud Research Laboratory (ACRL) and leads several projects under the Latin American Grid initiative. He has extensive experience in software development and leading large scale software engineering projects both in industry and in academia. Currently, he is collaborating with top researchers in 8 countries and is leading several international collaborative research projects. He is serving as a General Chair of SEKE 2012 and has served as the Program Chair, Co-Chair, and Committee Member of several top-tier international conferences and workshops of his field. He has served as a referee for several IEEE and SP&E journals and as a referee and panelist for several funding agencies including National Science Foundation (NSF), Luxembourg National Research Fund (FNR), and Florida Sea Grant. His current research interests include Distributed Systems, Software Engineering, Autonomic Computing, High-Performance Computing, Grid Computing, Cloud Computing, Pervasive Systems, and Mobile Computing. He has more than 80 refereed publications and is PI or Co-PI of 17 grants from NSF, IBM, Kaseya, TeraGrid, and FIU for a total of about $6 million. He is a member of the IEEE and can be reached at sadjadi@cs.fiu.edu and http://www.cs.fiu.edu/~sadjadi/.

Adjouadi is the WARE Distinguished Professor at FIU's Department of Electrical and Computer Engineering. Since 1993, Adjouadi has directed the FIU Center for Advanced Technology and Education [http://cate.fiu.edu]. Adjouadi's doctorate is from the University of Florida (1985).

3. Integrating Location Information as Geohash Codes in Convolutional Neural Network-Based Satellite Image Classification.7pp

Follows

Integrating Location Information as Geohash Codes in Convolutional Neural Network-Based Satellite Image Classification

Mahara, Arpan; and Rishe, Naphtali

Abstract: ***In the past few years, there have been many research studies conducted in the field of Satellite Image Classification. The purposes of these studies included flood identification, forest fire monitoring, greenery land identification, and land-usage identification. In this field, finding suitable data is often considered problematic, and some research has also been done to identify and extract suitable datasets for classification. Although satellite data can be challenging to deal with, Convolutional Neural Networks (CNNs), which consist of multiple interconnected neurons, have shown promising results when applied to satellite imagery data. In the present work, first we have manually downloaded satellite images of four different classes in Florida locations using the TerraFly Mapping System, developed and managed by the High Performance Database Research Center at Florida International University. We then develop a CNN architecture suitable for extracting features and capable of multi-class classification in our dataset. We discuss the shortcomings in the classification due to the limited size of the dataset. To address this issue, we first employ data augmentation and then utilize transfer learning methodology for feature extraction with VGG16 and ResNet50 pretrained models. We use these features to classify satellite imagery of Florida. We analyze the misclassification in our model and, to address this issue, we introduce a location-based CNN model. We convert coordinates to geohash codes, use these codes as an additional feature vector and feed them into the CNN model. We believe that the new CNN model combined with geohash codes as location features provides a better accuracy for our dataset.***

Index Terms: ***CNN (Convolutional Neural Network), Data Augmentation, Geohash Code, Satellite Image, Transfer Learning***

1. INTRODUCTION

THE classification of remotely sensed data has numerous practical applications, including forest fire detection, landslide detection, and environmental monitoring. In recent years, several machine learning and deep learning algorithms, including but not limited to K-Nearest Neighbor (KNN), Random Forest (RF), Support Vector Machine (SVM), and Neural Networks (NNs), have been applied to the classification of remotely sensed data. In the Deep Learning field, CNNs have demonstrated the capability to learn complex models [1]. One of the key reasons for CNNs' success is their ability to extract features automatically, which greatly benefits researchers in achieving generalized and efficient classification. Comprehensive reviews of various models, architectures, and classifications related to CNNs can be found in references [1]–[3].

In general, image classification is performed based on pixel-wise feature extraction and assigning them to certain classes. Mnih proposed a CNN architecture for aerial image classification using a patch-based framework[4]. In that paper, the CNN network outputs a dense classification patch rather than a single categorical value. As a result, the patch-based CNN architecture increases the number of unproductive trainable parameters, potentially leading to inefficiencies in classification. To provide a solution to this issue, Maggiori *et al.* [5] proposed a fully convolutional architecture that only incorporates the convolution and deconvolution norms of CNN, producing classification maps that can be used for satellite image classification. In [5], the authors have created a more efficient CNN architecture, but their focus was on binary classification with only one class, i.e., buildings. The authors have not addressed the importance of using image location to enhance classification accuracy. The CNN architecture we use in this paper is based on the architectures described in [4] and [5], and we focus on multi-class image classification by integrating the location concept. In [6], coordinates were integrated into CNN to enhance remote sensing image classification. During the training phase, they directly fed spatial information, such as longitude and latitude, as an additional feature to the CNN for feature extraction. Similarly, Tang et al. [7] proposed a GPS encoding idea that incorporates location information into CNN for extracting features and improved image classification. They represented location as a

Manuscript received March 24, 2023.

Arpan Mahara is at the Knight Foundation School of Computing and Information Sciences, Florida International University, Miami, FL, USA (e-mail: amaha038@cs.fiu.edu)

Naphtali Rishe is at the Knight Foundation School of Computing and Information Sciences, Florida International University, Miami, FL, USA (e-mail: rishe@cs.fiu.edu)

Correspondence email is amaha038@cs.fiu.edu

binary code, with each bit corresponding to a specific geographic location. They devised a method for creating a set of grid cells covering the Earth's geographical area, primarily focusing on regions within the United States.

In the present work, we first downloaded satellite images of Florida using TerraFly Map's raster API, which incorporates a predefined tile system utilizing the Microsoft Bing projection. The images are all 256 * 256 pixels and have three color channels (Red, Green, and Blue). We have grouped the images into four different classes: Building, GreeneryLand, House, and WaterResource. Here, by a "house" we mean a structure of 1-2 stories, and by a "building" we mean a structure of 3 or more stories. We have developed our CNN architecture based on the idea mentioned in [4] and [5]. However, CNNs require large datasets to learn features and make efficient predictions, and our CNN may not be able to generalize efficient classification from manually-collected datasets due to the lack of a large number of images. To improve the efficiency of our classification, we first use data augmentation presented in [8]. Then, we adopt transfer learning strategies presented in [9] to extract features using pre-trained models, such as VGG16 and ResNet50. Finally, we enhance the CNN's feature set by converting each longitude and latitude to geohash codes and feeding them as extra features. Geohash is a process that converts coordinates into strings of data, which are easy to handle; more information on geohash codes can be found in [10]–[12]. We then evaluate the accuracy of our model with these additional features.

The paper is structured as follows. In Section 2, we describe the mechanism of CNN and how we have prepared the dataset. In Section 3, we propose a CNN architecture and analyze the shortcomings of the lack of a large dataset. In Subsection 3.1, we set up a transfer learning architecture to obtain an efficient classification model. In addition, we integrate coordinates as geohash codes into our model. Section 4 presents the computational results achieved from all the models, including the results obtained after the integration of location information. Finally, we summarize our findings and outline future research directions in Section 5.

2. CNN Introduction and Dataset

CNNs are a special type of neural networks that have been invented to mimic the mechanism of human brain for identifying or recognizing objects. They contain numerous interconnected neurons, each of which responds only to their own receptive field. The interesting part of the neurons in CNNs is that they possess the ability to automatically extract features from an image. In a CNN, each neuron undergoes input and output procedures to learn the pattern of the model. The common mathematical interpretation of the neural operation to obtain an output 'o' can be expressed as follows:

$$o = \sigma\left(\sum_{k=1}^{n} w_k \,.\, x_k + b\right) \quad (1)$$

where σ is an activation function that helps the CNN to learn an intricate pattern by encompassing non-linearity in the output. Similarly, x_k and w_k are k^{th} input and k^{th} weight, respectively, and b denotes a scalar parameter added to each output, which helps the CNN to extract complicated patterns from data. Biases should be carefully addressed; otherwise, they may lead to overfitting or underfitting in the model.

In general, the CNN architecture has three different layers: a convolutional layer, a pooling layer, and a connected layer. In the convolutional layer, the dot product between the kernel and the input image is calculated by sliding a filter over the image. This aids the architecture in extracting features from the images in the dataset. The sliding of the filter around the image can be controlled with a specific stride size. Let's say we have an image of dimension D*D with C channels. We define the size of the stride as S, the size of the kernel or filter as K, and X as the amount of padding to maintain the same size of images in both the input and output sectors. The output of the convolutional layer can be stated as follows:

$$C_{out} = \frac{D-K+2X}{S}+1 \quad (2)$$

Once the output is calculated, it is passed through an activation function. A pooling layer is applied in the CNN in order to deduct trainable parameters and balance the computation, which serves as an efficient feature extraction by reducing the size of the output map obtained from the convolutional layer. A fully connected layer simply flattens the output obtained from the previous layer, which helps to connect the obtained features to the labels in the given model.

As mentioned above, we used the TerraFly Map's raster API (which uses the Microsoft Bing projection) to download the images. We use the TerraFly Map to determine the XY tile coordinates for specific regions within Florida, keeping the zoom level constant at 19. After determining those coordinates, we pass the values to the Raster API's URL, and then we use web scraping to download the images. Since we aim to integrate the location feature into our CNN model, we need to prepare a dataset of satellite images that also have associated coordinates. To achieve this, we converted each XY tile coordinates obtained from the map to longitude and latitude by using the following procedure:

A = X tile, B = Y tile coordinates (i)

$$pixelA = A * 256 + 128 \quad \text{(ii)}$$

$$pixelB = B * 256 + 128 \quad \text{(iii)}$$

$$sizeofMap = 256 * 2^{zoom} \quad \text{(iv)}$$

$$normA = (pixelA / (sizeofMap)) - 0.5 \quad \text{(v)}$$

$$normB = 0.5 - (pixelB / (sizeofMap)) \quad \text{(vi)}$$

$$Latitude = 90 - \left(\frac{360}{\pi}\right) * \tan^{-1}(\exp(-2\pi * normB) \quad \text{(vii)}$$

$$Longitude = 360 * normA \quad \text{(viii)}$$

In our model, we use a specific zoom value of 19. To calculate the geohash code, we use the values of the latitude and longitude obtained from equations (vii) and (viii), as described in [12]. We use the Python Geohash Library to convert the latitude and longitude to geohash codes. In our final step of dataset preparation, we map each geohash code to the right images by using a Python dictionary. The keys of the dictionary are the filenames, and the values are the corresponding geohash codes.

3. *The Proposed Architecture*

Our CNN architecture utilizes ideas from [4] and [5]. We apply convolutional layers that incorporate both convolutional and deconvolution operations, as described in [5]. We flatten the multi-dimensional tensor into a single-dimensional tensor output and apply the dense layer principle the output, as suggested by Mnih [4]. We feed the fully connected layer of the images into the CNN to extract the feature map, which is used for classifying the images according to their given labels. Our CNN architecture differs from the one presented in [6], as we focus on extraction that is capable of detecting features in the images, rather than extracting the spatial features of pixels in the images. As we can observe in Figure 1, the CNN architecture has three convolutional layers and three max pooling layers. In each max pooling layer, we downsample the dimension of each input map by a factor of 2, resulting in a feature map of size 32*32. Downsampling is a common approach in neural networks to reduce memory usage during computation and to enable high-level feature extraction [13]. We flatten the resulting feature map by applying a flatten layer, which transforms it into a one-dimensional array of size 65,536. We then apply two separate dense layers followed by a Softmax activation function. The final dense layer has 4 units, as our model has 4 classes of satellite images and the probability distribution is over those 4 classes.

In the first stage of our image classification procedure, we use a satellite image dataset that excludes geohash codes. We split the dataset into a training set, a testing set, and a validation set, with 80%, 10%, and 10% of the full dataset, respectively. We have experimented with our model using various numbers of epochs and batch sizes, and have determined that using 60 epochs with a batch size of 32 produces the best results. In general, researchers tend to choose an optimal number of epochs to achieve good accuracy in complex models and prevent the model from overfitting. A lower accuracy in the testing set indicates that the model is overfitting. One reason for this overfitting is the lack of a large amount of data in our model, as we only had 300 images in each class, with a total of 1200 images. CNNs require a large dataset to extract complex features and provide better accuracy in image classification [13].

To address this problem, we have used data augmentation strategies of deep learning, as presented in [8], [14], and [15]. In terms of images, data augmentation involves increasing the size of the dataset by applying variations, such as rotating images, changing the visual effects, etc., to the existing images [14]. To increase the size of the dataset, we have applied random horizontal flipping, random rotation with approximately 8.62 degrees, and random zooming of 20% scale. The data augmentation has helped to address the problem of overfitting, but we have concluded that we can further increase the overall accuracy of our dataset by training our model using a pretrained model, such as VGG16 and ResNet50, with the concept of transfer learning. In the following Subsection 3.1, we provide details on how we use transfer learning in our model to improve overall accuracy.

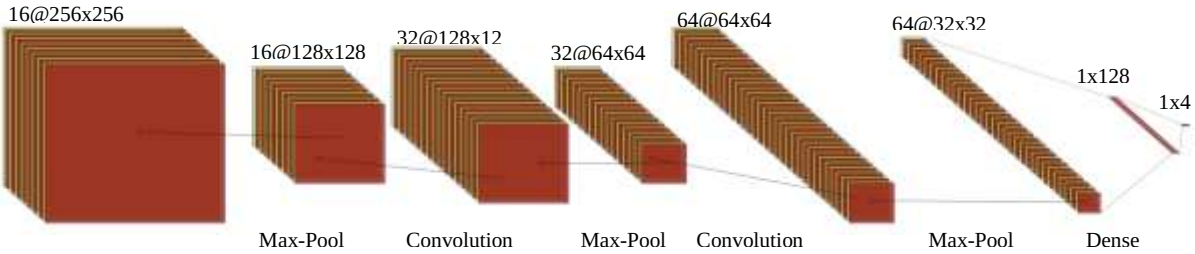


Figure 1: An illustration of the architecture of the CNN used. The template of the image has been obtained via https://alexlenail.me/NN-SVG/LeNet.html.

3.1 Applying Transfer Learning

Transfer learning is a way to use a pretrained model in a different but related model to solve the problem of the lack of abundant data to extract effective features and reduce the time required for training the dataset [16]. Our idea on integrating Transfer Learning is based on [9] and [17], and we have selected VGG16 and ResNet50 as the two pretrained models for our experiment. VGG16 is one of the most widely used deep neural network architectures; it consists of 13 convolutional layers and 3 dense layers, and the model has been trained on the ImageNet dataset [18]. Similarly, ResNet50 is another widely used deep neural network trained on the ImageNet dataset, consisting of 50 layers in total; it enables the network to assimilate residual functions rather than underlying mappings [19].

To set up the model using the transfer learning idea, we first remove the final connected layer of the VGG16 model. Then, we use the pretrained weights, and we set up the desired input shape to 256*256*3, the same shape that matches the input shape of the images in our original dataset. We freeze all the pre-trained layers and use only pre-trained weights to extract features, training the two new dense layers to predict new images in the dataset. The output of the flatten layer obtained from the pretrained model is passed to the first dense layer with 256 neurons, followed by the Relu activation function. In addition to this, we apply the Dropout function to the output from the dense layer to prevent overfitting in our model. Thereafter, we apply the final fully connected layer with 4 output nodes to obtain the probability distribution among 4 classes to predict the images followed by the Softmax activation function. We follow the same procedure when using the ResNet50 model. Once both models were ready, we experimented with them in our dataset. However, we have found misclassification in some of our data, which was further hindering the accuracy. To improve the accuracy, we integrate location coordinates in our image classification model in Subsection 3.2.

3.2 Integration of Location as Geohash Codes

Our goal is to increase the accuracy of satellite image classification in the downloaded dataset by integrating location information. We have decided to use geohash codes obtained from the conversion of latitude and longitude values. Geohash is a type of data structure used with spatial data that provide an encoding of latitude and longitude [20]. We are motivated to use geohash codes because locations with long common geohash prefixes are generally located nearby each other [20]. Our dataset contains satellite images downloaded within Florida, and there is a correlation between geographical location and image content. Images of houses and buildings are in two different classes, and some of these images might be misclassified if the model only considers visual characteristics because building images and house images captured from satellites have some visual similarity. In our dataset, two images of houses or buildings tend to be nearby each other as they have been downloaded by specifying the tiles coordinates. We believe that we can exploit this idea in our model by using geohash codes and prevent the misclassification of data.

We have experimented with the location concept by incorporating geohash codes into the VGG16 pretrained model. We have converted each geohash code into a floating-point value since neural networks typically deal with numerical values rather than strings. Next, we add a new input layer for the geohash code and concatenate the flattened layer containing the weight features of VGG16 with the geohash codes, as shown in Figure 2. We then apply the same dense layers noted in Subsection 3.1 to extract the features that assist in the prediction of new images. We follow the same procedure of concatenation geohash code with the output layer in ResNet50.

Finally, we integrate location information, i.e., geohash codes, into our CNN architecture. We concatenate the feature map induced by applying 3 convolutional and 3 pooling layers with geohash codes to obtain a combined feature vector. We then follow the same procedure as noted above and apply a flatten layer and a dense layer, respectively, to the combined feature vector. We have experimented with our models using 60 epochs and a batch size of 32 to obtain accuracy.

4. Computational Results

We have sequentially tested all the models, starting from the CNN architecture that only extracts features from the image without concatenating the geohash codes. Our intent is not just to check the accuracy in the dataset but also to analyze whether the model is overfitting by checking how well it performs on unseen image data. We use the Top-1 accuracy metric to check accuracies on all the models. Our CNN architecture yields an accuracy of 0.9244 on the training set but only 0.8842 on the testing set, indicating overfitting due to the limited size of the dataset. To address this issue, we apply data augmentation to the dataset, and our CNN architecture can generate approximately 0.9185 accuracy on the testing dataset and 0.9253 on the training set. Even though data augmentation helps to increase the dataset, it still lacks the power to generalize efficient feature extraction. So, we utilize transfer learning by using pretrained models, VGG16 and ResNet50, for efficient feature extraction that could be used to obtain better accuracy in our dataset. Having tested these models on all the datasets, we achieve an accuracy of approximately 0.9456 on the testing set and 0.9529 on the

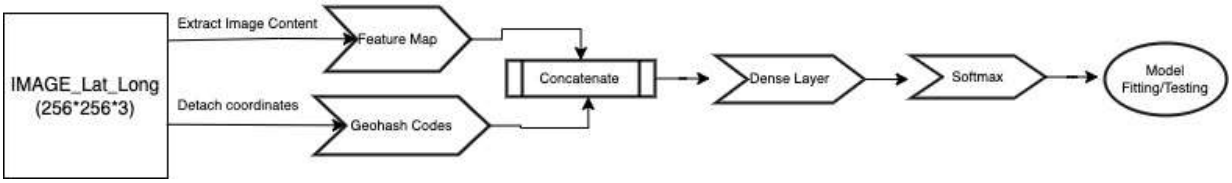


Figure 2: An illustration of the architecture obtained by integrating geohash codes into the CNN (including pretrained models) architecture.

training set for VGG16, as well as approximately 0.9516 on the testing set and 0.9576 on the training set for ResNet50, respectively.

Similarly, as mentioned in Subsection 3.2 above, to integrate location as an additional feature, we initially integrate the geohash codes with VGG16 and ResNet50 pretrained models. We achieve top-1 accuracies of 0.9789 on the testing set and 0.9769 on the training set using the combined VGG16 and geohash feature architecture, and top-1 accuracies of 0.9812 on the testing set and 0.9795 on the training set using the combined ResNet50 and geohash feature architecture. Finally, we experimented with integrating location information into our CNN architecture by concatenating the geohash codes with the image features. By doing so, we are able to increase the top-1 accuracy on the testing set from 0.9185 to 0.9542 and on the training set from 0.9253 to 0.9512 by incorporating location as a feature.

From the results mentioned above, we can see that incorporating the geohash coding feature has led to an improvement in our classification accuracy. In each model, after integrating geohash as a location feature, there is an increase of top-1 accuracy by 2 to 3 percentage points. The reason for the small increases in the accuracies is because of the small size of the dataset. We have observed misclassifications mainly among house and building images, as they have a resemblance, but the number of misclassifications is relatively small due to our small dataset size. However, we can mitigate these misclassifications by utilizing geohash codes to differentiate between these images with similar features.

We present the results and comparisons of all the models mentioned above in Table 1. The notations used in Table 1 are as follows:

- $Acc.$ – accuracy in the testing set (general accuracy of the model);
- Acc_T – accuracy in the training set;
- $Loss$ – categorical cross-entropy loss in our multi-class classification model;

Table 1. Results and comparisons among our models based on the accuracy

Method	Acc.	Acc_T	Loss
CNN (only)	0.8842	0.9244	0.8272
CNN + Data Augmentation	0.9185	0.9253	0.4380
VGG16 (CNN)	0.9456	0.9529	0.2549
RestNet50 (CNN)	0.9516	0.9576	0.1590
CNN + Data Augmentation + Geohash Code	0.9542	0.9512	0.0954
VGG16 (CNN) + Geohash Code	0.9789	0.9769	0.0443
ResNet50 (CNN) + Geohash Code	0.9812	0.9795	0.0394

As shown in Table 1, the proposed CNN model, as well as the VGG16 and ResNet50 models, show improved accuracy after the integration of geohash codes. As shown in the table, the categorical cross-entropy loss decreases after the geohash codes have been applied, indicating that the models are able to make predictions that are closer to the true class membership probabilities. The lower loss value and similar accuracy on both the training and testing datasets suggest that the model is not overfitting to the training data.

5. CONCLUSION

This paper analyzes the limitations of using only image features in multi-class satellite image classification using CNNs. In multi-class satellite image classification, CNN architectures tend to make false predictions when there is a high degree of visual similarity between images from different classes. This issue is addressed by integrating geohash codes as an additional feature in the CNN model. With the additional geohash code feature map, the CNN model is able to make more accurate predictions.

According to the results presented in this paper, we can deduce that geohash codes can be used as an additional feature vector in the CNN architecture to make correct predictions and increase accuracy in satellite image classification. However, this may not apply in scenarios where

there is no correlation between geographical location and image content. To build a robust model for satellite image classification, it is important to take into account a range of factors, such as the size and preprocessing of the dataset, integrating additional feature vectors, the risk of overfitting, and the CNN architecture itself. This is because even a well-designed architecture may produce poor results if there is insufficient data or if the data has not been efficiently preprocessed.

In the future, we plan to explore the use of hybrid models in satellite image classification. The K-NN machine learning algorithm will be one of our focuses to identify the K-number of images that are most similar to each other based on their geohash codes, and then to automatically classify them into their respective classes. This approach has the prospect of increasing the accuracy of classifying images that are difficult to distinguish based on visual features alone, and may enable real-time classification of satellite imagery for applications such as disaster management and environmental monitoring.

ACKNOWLEDGMENT

This material is based in part upon work supported by the National Science Foundation under Grant Nos. MRI20 CNS-2018611, MRI CNS-1920182, B-BROIPS FDEP C-2104, and DHS E2055778.

REFERENCES

[1] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects," *IEEE Transactions on Neural Network Learning Systems*, vol. 33, no. 12, pp. 6999–7019, Dec. 2022, doi: 10.1109/TNNLS.2021.3084827.

[2] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Computing*, vol. 29, no. 9, pp. 2352–2449, Sep. 2017, doi: 10.1162/NECO_A_00990.

[3] N. Aloysius and M. Geetha, "A review on deep convolutional neural networks," *Proceedings of the 2017 IEEE International Conference on Communication and Signal Processing, ICCSP 2017*, vol. 2018-January, pp. 588–592, Feb. 2018, doi: 10.1109/ICCSP.2017.8286426.

[4] V. Mnih, "Machine Learning for Aerial Image Labeling," 2013.

[5] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, "Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 2, pp. 645–657, Feb. 2017, doi: 10.1109/TGRS.2016.2612821.

[6] F. Zhang, M. Yan, C. Hu, J. Ni, and Y. Zhou, "Integrating Coordinate Features in CNN-Based Remote Sensing Imagery Classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, 2022, doi: 10.1109/LGRS.2020.3045744.

[7] K. Tang, M. Paluri, L. Fei-Fei, R. Fergus, and L. Bourdev, "Improving Image Classification With Location Context." pp. 1008–1016, 2015.

[8] J. Wang and L. Perez, "The Effectiveness of Data Augmentation in Image Classification using Deep Learning".

[9] R. P. de Lima and K. Marfurt, "Convolutional Neural Network for Remote-Sensing Scene Classification: Transfer Learning Analysis," *Remote Sensing 2020, Vol. 12, Page 86*, vol. 12, no. 1, p. 86, Dec. 2019, doi: 10.3390/RS12010086.

[10] C. Zhou, H. Lu, Y. Xiang, J. Wu, and F. Wang, "GeohashTile: Vector Geographic Data Display Method Based on Geohash," *ISPRS International Journal of Geo-Information 2020, Vol. 9, Page 418*, vol. 9, no. 7, p. 418, Jun. 2020, doi: 10.3390/IJGI9070418.

[11] K. Huang, G. Li, and J. Wang, "Rapid retrieval strategy for massive remote sensing metadata based on GeoHash coding," *https://doi.org/10.1080/2150704X.2018.1508907*, vol. 9, no. 11, pp. 1070–1078, Nov. 2018, doi: 10.1080/2150704X.2018.1508907.

[12] R. Moussalli, M. Srivatsa, and S. Asaad, "Fast and flexible conversion of geohash codes to and from latitude/longitude coordinates," *Proceedings - 2015 IEEE 23rd Annual International Symposium on Field-Programmable Custom Computing Machines, FCCM 2015*, pp. 179–186, Jul. 2015, doi: 10.1109/FCCM.2015.18.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[14] J. Shijie, W. Ping, J. Peiyi, and H. Siping, "Research on data augmentation for image classification based on convolution neural networks," *Proceedings - 2017 Chinese Automation Congress, CAC 2017*, vol. 2017-January, pp. 4165–4170, Dec. 2017, doi: 10.1109/CAC.2017.8243510.

[15] A. Fawzi, H. Samulowitz, D. Turaga, and P. Frossard, "Adaptive data augmentation for image classification," *Proceedings - International Conference on Image Processing, ICIP*, vol. 2016-August, pp. 3688–3692, Aug. 2016, doi: 10.1109/ICIP.2016.7533048.

[16] R. Ribani and M. Marengoni, "A Survey of Transfer Learning for Convolutional Neural Networks," *Proceedings - 32nd Conference on Graphics, Patterns and Images Tutorials, SIBGRAPI-T 2019*, pp. 47–57, Oct. 2019, doi: 10.1109/SIBGRAPI-T.2019.00010.

[17] D. Theckedath and R. R. Sedamkar, "Detecting Affect States Using VGG16, ResNet50 and SE-ResNet50 Networks," *SN Comput Sci*, vol. 1, no. 2, pp. 1–7, Mar. 2020, doi: 10.1007/S42979-020-0114-9/TABLES/5.

[18] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, Sep. 2014, Accessed: Mar. 20, 2023. [Online]. Available: https://arxiv.org/abs/1409.1556v6

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition." pp. 770–778, 2016. Accessed: Mar. 20, 2023. [Online]. Available: http://image-net.org/challenges/LSVRC/2015/

[20] K. Lee, R. K. Ganti, M. Srivatsa, and L. Liu, "Efficient Spatial Query Processing for Big Data," 2014, doi: 10.1145/2666310.2666481.

Arpan is a Ph.D. student at the Florida International University. He is currently working as a research assistant at the High Performance Database Research Center (HPDRC) lab. His primary research is focused on the domain of neural network and geospatial data. At present, he is doing research on advanced deep learning models like Recurrent Neural Networks (RNN) and Transformers, with the goal of applying these models to various unique domains.

Rishe [http://cake.fiu.edu/Rishe] is the inaugural outstanding university professor for Florida International University (FIU). He is a professor at FIU's Knight Foundation School of Computing and Information Sciences. Rishe is the Director of FIU's High-Performance Database Research Center. aFrom 2013 to 2022, Rishe was the executive director of a multi-

university Center for Advanced Knowledge Enablement sponsored by the National Science Foundation's industry-university cooperation program. Rishe's research has been funded by the U.S. Government and Industry at over $50 million. Rishe's inventions include 26 U.S. patents. Rishe is a Fellow medalist of the National Academy of Inventors. In 2021, Rishe received the IBM Global University Programs award. Rishe's publications include 440 papers and six books. Rishe's pioneering geospatial system TerraFly has been highlighted by the National Science Foundation in its reports to U.S. Congress.

4. Spatiotemporal Model of Real Estate Valuation Trends.15pp

Follows

Spatiotemporal Model of Real Estate Valuation Trend

Rishe, Naphtali; Tamir, Dan; and Adjouadi, Malek

Abstract: ***Presented here is a model objectivizing real estate prices so that prices across time could be compared to understand historical price trends and also to assist in a property evaluation or appraisal, as well as for the analysis of comparables in estimating a reasonable offer for a property on the market. Given a timespan of interest, a locale (e.g., a particular zipcode, a city, a county, a state), a category of properties of interest (e.g., condos), an objective historical trend in values can be computed by first evaluating the ratios between the transactions' realized prices and objective governmental assessment of the properties at some fixed point of time; then, for each period (a month) averaging the ratios of all transaction in that period; then, comparing said averages (or medians) between different periods.***

Index Terms: Automatic Valuation Model, Geospatial Data Trend Analysis, House Price Trend Analysis, Real Estate, Spatiotemporal Extrapolation, Spatiotemporal Interpolation, Spatiotemporal Summarization

1. Background

Various services and methods exist for the estimation of the change over time in real estate prices in any given locale. Said prior models typically compute the average or median sale price in the locale during each period and then compare said statistics between the various periods. Some of said prior models can also focus their comparison on specific property categories, e.g., single-family homes or condos, and may further narrow the categories down, e.g., 3-bedroom homes or houses of 2000-2500 interior square feet. Yet, in said prior models, there is, in fact, a comparison of apples to oranges. Even in a small locale, e.g., a zipcode, and even in a narrow category, there are vastly different properties being averaged. This creates a statistical bias when different periods are compared since in one period there could dominate sales of quality-built properties with a view, while in another period, lesser properties could dominate. This bias becomes even stronger when larger areas are analyzed, e.g., at the county or state level, because demographic changes can favor sale activity more in cheaper subareas in one period and in more exclusive subareas in another period.

Models exist comparing price per unit of size, e.g., price per interior square foot of a home. However, that too comingles residences with a view and residences without a view, well-built houses to poorly built; further accounting for one size metric, such as interior area, ignores other size metrics, such as the lot size.

A recent improvement to Automatic Valuation Models (AVM) [1-4] of properties includes the computation of ratios of actual sale prices to government-assessed values and the extrapolation of such ratios for the valuation of a specific property.

2. The Present Method

Presented here is a model objectivizing real estate prices so that prices across time can be compared to understand historical price trends and also to assist in property evaluation or appraisal, as well as for the analysis of comparables in estimating a reasonable offer for a property on the market.

In order to objectivize and normalize real estate transactions across a locale and a time period, we need to have a metric of valuation of properties that was consistent among all the properties in the locale at some point in time. Said point in time of the objective metric does not need to be within said period. Further, said metric does not have to represent the true value

Manuscript received April 11, 2023.

Naphtali Rishe is at the Knight Foundation School of Computing and Information Sciences, Florida International University, Miami, FL, USA (e-mail: rishe@cs.fiu.edu)

Dan Tamir is at the Texas State University, San Marcos, Texas, USA (e-mail: dt19@txstate.edu)

Malek Adjouadi is at the Department of Electrical and Computer Engineering, Florida International University, Miami, FL, USA (e-mail: adjouadi@fiu.edu).

Correspondence email is rishe@cs.fiu.edu

U.S. patent pending.

of each property at said point in time; rather, it has to be consistently related or proportional to

the true value. Said relationship does not have to be a precise linear proportion, nor does it have to be truly consistent in 100% of the cases since we only need that metric for a statistical aggregation of large numbers of cases. A good candidate for said metric is property valuation by local government tax assessors, particularly the tax appraisal offices in most counties in the United States. Said offices typically invest immense effort in the attempt of consistent valuation of all the properties under their jurisdiction, taking into account quantitative metrics (such as the size of the interior, the size of the lot, year built, year renovated, the ground elevation, the floor level elevation of a condo in a building, the costs of improvement made based on the permits filed, etc.) and qualitative metrics (location, exposure, view, special features, etc.). For example, in Florida, the county assessor offices determine what they call the “just value” of the properties as of January 1 of the assessment year. In order to minimize litigation, the assessor’s office typically sets the “just value” at 10-20% below the true value, which does not affect the algorithm presented here as long as said discount is reasonably consistent.

Folio	Use	Just Value	County-assessed value	Land-value ($Land)	Land area (sq ft)	Year improved	Sq.ft (sq ft)	Owner	Legal	Parcel address	Parcel city
232270040180	Residential--Multi-family (3)	$1.44M	$1.44M	1.06M	7080	1960	5484	T&T GLOBAL INVEST LLC	FAIRG-REEN RESUB BLKS C-D-E	2843 SHERI-DAN AVE	Miami Beach
232270040170	Residential--Multi-family (3)	$1.33M	$949K	1.06M	7080	1940	7408	STEVEN STARR	FAIRG-REEN RESUB BLKS C-D-E	2851 SHERI-DAN AVE	Miami Beach
232270040190	Residential--Multi-family (8)	$1.6M	$1.6M	1.28M	8538	1961	5918	TUDAN LLC	FAIRG-REEN R- S PB 4-154	2825 SHERI-DAN AVE	Miami Beach
232270040230	Residential--Multi-family (8)	$1.29M	$912K	1.29M	8580	1950	4832	PHOEN-ECIA REAL ESTATE INVEST L	FAIRG-REEN RESUB BLK C-D-E	2830 PINE TREE DR	Miami Beach

Figure 1: Annual property valuation by a county assessor

It should be noted that government offices sometimes provide multiple types of valuations for tax purposes. The following example shows the various official "valuations" available from Florida counties. Among these valuations, the only meaningful one for the present purposes is the "Just Value." The other valuations either reflect only a part of the property value (e.g., the Land Value and the Improvement Value) affected by the demographics of the property owner and, thus, are not meaningful for understanding the true value of the property.

Just Value: Market Value per County Appraiser, total of land and building. What the County Appraiser calls, but is typically less than, The most probable price in cash, terms equivalent to cash, or other precisely revealed terms, for which the appraised property will sell in competitive market under all conditions requisite to fair sale as of January 1 of the Roll Year. AKA Just Value (US$)

Land-value: The portion of the just value attributed to the land only, as determined by the County Property Appraiser. Most counties do not have a model to correctly divide the total value between Land and Building ($Land)

County-assessed value: Assessed value for taxation other than by school districts. For 2008, school and non school assessed value will differ in counties where the county or the city has adopted ordinances for assessing historic property used for commercial or non-profit purposes and high water recharge property based on character or use. Beginning in 2009, the 10% assessment increase limitation on non-homestead property will also apply only for non-school purposes and further cause the assessed values for school and non-school purposes to be different (US$)

Homestead assessed value per School District: The assessed value of only the portion of the property that is considered a homestead, for school district taxation. The difference between homestead just value and homestead assessed value is that the latter is net of the Save Our Homes assessment increase limitation. Blank for non-homesteaded properties (US$)

School Taxable Value: The taxable value for school district taxation. School taxable value is based only on school assessed value and does not include subtractions for the new additional homestead exemption or local option exemptions which are applicable only to the county or municipality adopting the exemption (US$)

Taxable value, Not for School district: The taxable value for general county and municipal taxation. County taxable value is the county assessed value minus all exemption (US$)

AV-RNR: Residential and non-residential property assessed value. The assessed value of only the portion of the property that is residential or non-residential property under s. 193.155, F.S (US$)

Building Value: The portion of the just value attributed to the improvements of the property, just value of new construction. Note: most counties do not have a model to correctly split value into land and building, so this data item might not be reliable (US$)

Figure 2: Meta-data of various property valuations by county assessor offices in Florida; the “Just Value” is an objective valuation.

The method proposed herein compares the transactional sale price of each property, no matter when, to one time-fixed metric of an objective valuation in order to evaluate the ratio by which the realized price is above (or below) said metric. That is, this ratio is the ratio between the realized price and said objective metric. In the example of this figure, sales at different times are compared to the county’s “Just Value” as of January 1, 2021, to compute the Ratio factor. Notice that Row 3 in the table contains an obvious data entry error. Therefore, there can be a data-cleansing process in order to disregard outliers that are outside a reasonable range. Data about the realized prices of each transaction can be obtained from proprietary databases, such as those provided by data consolidators, from county or state records, or from the Real Estate Multiple Listing Service (MLS), as in the following figure.

Area total (Sq Ft)	City	Address	List Price	Type of property	View	Year built	Status of the listing	Closing Date	Sale Price	$ / SqFt as sold	Interior size (sqft)	Ratio Sold to County Value
1110	Miami Beach	2850 Pine Tree Dr #8	$320K	Residential	Other View	1951	Closed	≤**2016-07-01**≥	293K	$264	1110	1.64
1088	Miami Beach	2850 Pine Tree Dr #7	$250K	Residential	Garden View	1951	Closed	≤**2018-05-21**≥	248K	$227	1088	1.41
1173	Miami Beach	2850 PINE TREE Dr #1	$295K	Residential	Other View	1951	Closed	≤**2018-10-18**≥	289	$0.246	1173	0.0016
1128	Miami Beach	2858 PINE TREE Dr #5	$258K	Residential	Garden View	1966	Closed	≤**2017-05-18**≥	253K	$224	1128	1.2
688	Miami Beach	2858 Pine Tree Dr #4	$179K	Residential	None	1966	Closed	≤**2021-01-08**≥	160K	$233	688	1.1

Figure 3: Ratios of the realized price, at various times, to the County "Just Value" of 2021-0-01. Row 3 is an outlier to be disregarded.

The Ratio thusly computed is an objective comparison metric between different sale transactions in a locale at close times or across long timespans.

To better compare sale transactions over time within a locale, we can subdivide properties into categories because it is possible that in different property categories, prices increased at different paces. For example, we can consider two

categories of properties: single-family homes vs. condominium apartments.

Next, we consider a locale of interest, e.g., Zipcode 33175; a category of interest, e.g., Houses (single-family homes); and a timespan of interest, e.g., from January 1, 2006, through December 31, 2007. We subdivide said timespan into periods, e.g., calendar months. In each period, for each sale transaction, we evaluate the Ratio of the price to the fixed objective metric, e.g., the 2021 County "Just Value." We can exclude outlier transactions based on any criteria of outlier exclusion. For each period, we evaluate a representative statistical aggregator of the ratios, e.g., the average of the ratios or the median of the ratios, of all the relevant sale transactions. We can further exclude months with a very low number of transactions, e.g., less than 6, to avoid the possibility of excess weight of any single transaction, which may cause bias in statistical analysis across time.

33175	Houses	2006-02	6	1.37
33175	Houses	2006-03	14	1.28
33175	Houses	2006-04	19	1.26
33175	Houses	2006-05	23	1.33
33175	Houses	2006-06	10	1.42
33175	Houses	2006-07	15	1.35
33175	Houses	2006-08	16	1.28
33175	Houses	2006-09	18	1.34
33175	Houses	2006-10	19	1.44
33175	Houses	2006-11	19	1.39
33175	Houses	2006-12	11	1.3
33175	Houses	2007-01	10	1.24
33175	Houses	2007-02	10	1.3
33175	Houses	2007-03	12	1.38
33175	Houses	2007-04	7	1.36
33175	Houses	2007-05	20	1.37
33175	Houses	2007-06	8	1.21
33175	Houses	2007-07	18	1.2
33175	Houses	2007-08	12	1.18
33175	Houses	2007-10	10	1.2
33175	Houses	2007-11	6	1.29
33175	Houses	2007-12	11	1.14

Figure 4: The number of sale transactions in each month in 2006-2007 in Zipcode 33175, excluding outliers, and the median of their ratios of the sale price to the fixed objective metric of the county valuation as of 1/1/2021; months with less than six transactions (September 2007) are excluded.

To facilitate human comprehension of said average (or median) ratios, we can normalize them to a specific period (month) as the base, e.g., the beginning month of said timespan, i.e., by computing the Factor as the median Ratio of any given month divided by the median Ratio of the base period. Thereby average (or median) prices can be expressed as the percentage increase (or decrease) since the base month, as in the following figures.

33175	Houses	2006-02	6	1	0% (since 2006-02)
33175	Houses	2006-03	14	0.934	-6.6% (since 2006-02)
33175	Houses	2006-04	19	0.92	-8% (since 2006-02)
33175	Houses	2006-05	23	0.971	-2.9% (since 2006-02)
33175	Houses	2006-06	10	1.04	3.6% (since 2006-02)
33175	Houses	2006-07	15	0.985	-1.5% (since 2006-02)
33175	Houses	2006-08	16	0.934	-6.6% (since 2006-02)
33175	Houses	2006-09	18	0.978	-2.2% (since 2006-02)
33175	Houses	2006-10	19	1.05	5.1% (since 2006-02)
33175	Houses	2006-11	19	1.01	1.5% (since 2006-02)
33175	Houses	2006-12	11	0.949	-5.1% (since 2006-02)
33175	Houses	2007-01	10	0.905	-9.5% (since 2006-02)
33175	Houses	2007-02	10	0.949	-5.1% (since 2006-02)
33175	Houses	2007-03	12	1.01	0.73% (since 2006-02)
33175	Houses	2007-04	7	0.993	-0.73% (since 2006-02)
33175	Houses	2007-05	20	1	0% (since 2006-02)
33175	Houses	2007-06	8	0.883	-12% (since 2006-02)
33175	Houses	2007-07	18	0.876	-12% (since 2006-02)
33175	Houses	2007-08	12	0.861	-14% (since 2006-02)
33175	Houses	2007-10	10	0.876	-12% (since 2006-02)
33175	Houses	2007-11	6	0.942	-5.8% (since 2006-02)
33175	Houses	2007-12	11	0.832	-17% (since 2006-02)

Figure 5: Normalization of the median ratios (the realized prices divided by the 2021 county valuation) to Month 2006-02, i.e., dividing by the median Ratio of 2006-02, whereby; the last column shows the percentage increase since 2006-02.

Zipcode	Property type	Month	Number of closings	Factor	Percentage increase since 02/2006
33175	Houses	2022-06	7	1.37	37% (since 2006-02)
33175	Houses	2022-05	24	1.35	35% (since 2006-02)
33175	Houses	2022-04	22	1.34	34% (since 2006-02)
33175	Houses	2022-03	25	1.27	27% (since 2006-02)
33175	Houses	2022-02	25	1.18	18% (since 2006-02)
33175	Houses	2022-01	17	1.29	29% (since 2006-02)

Figure 6: Normalization of the median ratios (the realized prices divided by the 2021 county valuation) of January-June 2022 to February 2006, i.e., dividing by the median Ratio of 2006-02; the last column shows the percentage increase since 2006-02.

For better understanding by users, said factors can be presented as a graph, as in the following figure.

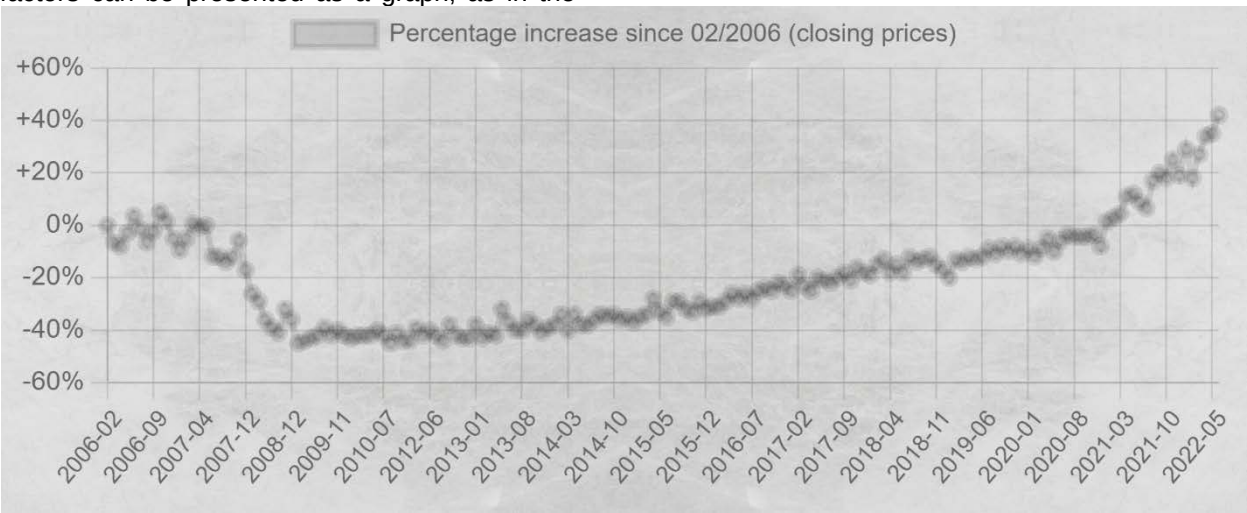


Figure 7: Chart of the change in the median ratios (the realized prices divided by the 2021 county valuation) in comparison to February 2006, for houses in Zipcode 33175.

Said chart informs how property values in the locale changed over time. The locale can be of any size as long as there are enough sale transactions therein to make a statistically significant analysis. The example in the following figure shows entire Southeast Florida as one locale and differentiates two property categories: condominium units and single-family homes.

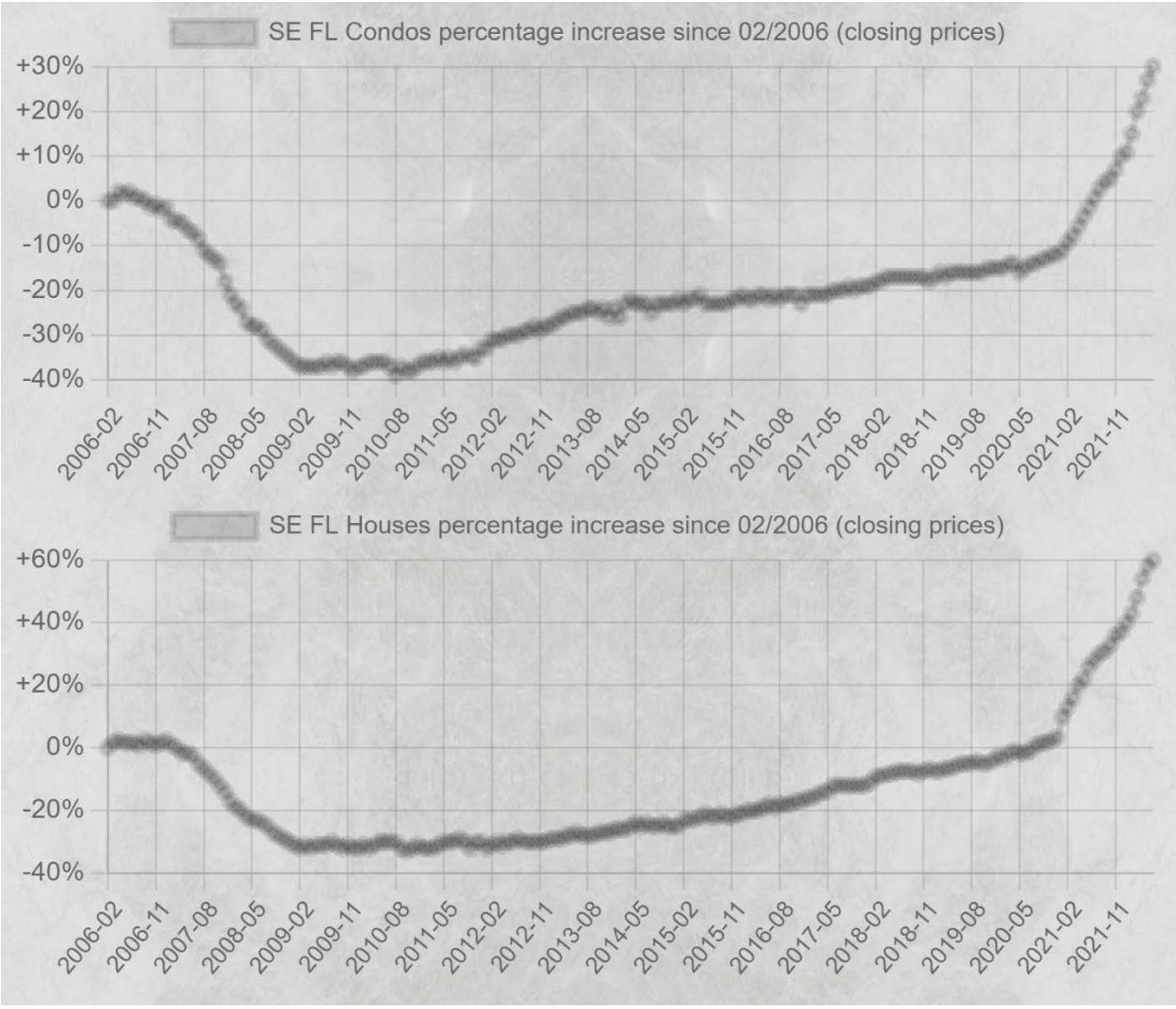


Figure 8: Charts of the change in the median ratios (the realized prices divided by the 2021 county valuation) in comparison to February 2006, for houses and condos in Southeast Florida.

3. *PSEUDO-CODE*

1. *MLS* := database of all multiple-listing service real estate transactions in SE Florida

2. *State_Parcels* := database of county valuations of all properties in Florida as of a fixed date, e.g., 1/1/2021

3. *Allreal* := inner **join** on the field of FOLIO_NUMBER of the MLS and Parcel databases:

MLS [FOLIO_NUMBER] *State_Parcels*;

and **projection** of said join to all the fields on of *MLS* plus the field Just_Value from *State_Parcels*, i.e.:

Allreal := **select** MLS.*, *State_Parcels*.Just_Value **from** *MLS*, *State_Parcels* **where** *MLS*.Folio_nbr=*State_Parcels*.Folio_nbr

4. Zipcodes := all the zip codes in Allreal, i.e.:

Zipcodes := **select unique** Zipcode **from** *Allreal*

5. **for every** *zipcode* in *Zipcodes* **do** {

5.1. *Sub_Allreal* := **select** * **from** *Allreal* **where** *Allreal*.Zipcode = *zipcode*

5.2. *Months* := **select unique** (Closing_Date as

yyyy-mm-dd).substring(1,7) from *Sub_Allreal*

5.3. **for each** *month* **in** *Months*
let *Factor*[*zipcode*,*month*] :=
select median (Closing_Price/Just_Value)
from *Sub_Allreal*
where Closing_Date **is within** *month*

5.4. *reference_month* := **minimum**(*Months*) (Any month can be chosen to serve as the reference, in particular, it could be the minimum (earliest) month or the maximum (latest) month.)

5.5. **Display** or **plot**
Factor[*zipcode*,*] / *Factor*[*zipcode*,*reference_month*]
}

4. *Alternative Model with Contract-Pending Dates*

The closing date of property sale transactions has an imperfection in its utility to assess the contemporary market sentiment. That is because the market sentiment is manifested at the time of the execution of a contract for purchase and sale between the buyer and the seller, while the closing of the transaction typically occurs a month or a couple of months later. To capture the timeliness of the market sentiment more precisely, we can look at transactions that have closed, but we date them at the purchase contract's effective date rather than at the closing date. Said purchase contract date can typically be obtained from MLS (multiple listing service) data sources (where it is often called the "Pending Date," i.e., the date the property went under a purchase contract and became pending closing), like in the following figure.

Bed rooms	Area total (Sq Ft)	City	Address	List Price	Type of property	View	Year built	Status of the listing	Closing Date	Pending Date	Sale Price	$ / SqFt as sold	Interior size (sqft)	Ratio Sold to County Value
2	1110	Miami Beach	2850 Pine Tree Dr #8	$320K	Residential	Other View	1951	Closed	≤**2016-07-01**≥	2016-05-05	293K	$264	1110	1.64
2	1088	Miami Beach	2850 Pine Tree Dr #7	$250K	Residential	Garden View	1951	Closed	≤**2018-05-21**≥	2018-04-23	248K	$227	1088	1.41
2	1173	Miami Beach	2850 PINE TREE Dr #1	$295K	Residential	Other View	1951	Closed	≤**2018-10-18**≥	2018-08-14	289	$0.246	1173	0.0016
2	1128	Miami Beach	2858 PINE TREE Dr #5	$258K	Residential	Garden View	1966	Closed	≤**2017-05-18**≥	2017-04-25	253K	$224	1128	1.2
1	688	Miami Beach	2858 Pine Tree Dr #4	$179K	Residential	None	1966	Closed	≤**2021-01-08**≥	2020-11-30	160K	$233	688	1.1

Figure 9: MLS data showing the Contract-Pending Date, in addition to the Closing Date, as well as the ratio of the closed sale price to the 2021 county valuation.

By reanalyzing the same data for sales closed between January 2006 and June 2012, we get a chart more accurately showing the timely market sentiment during most periods, as in the following figure.

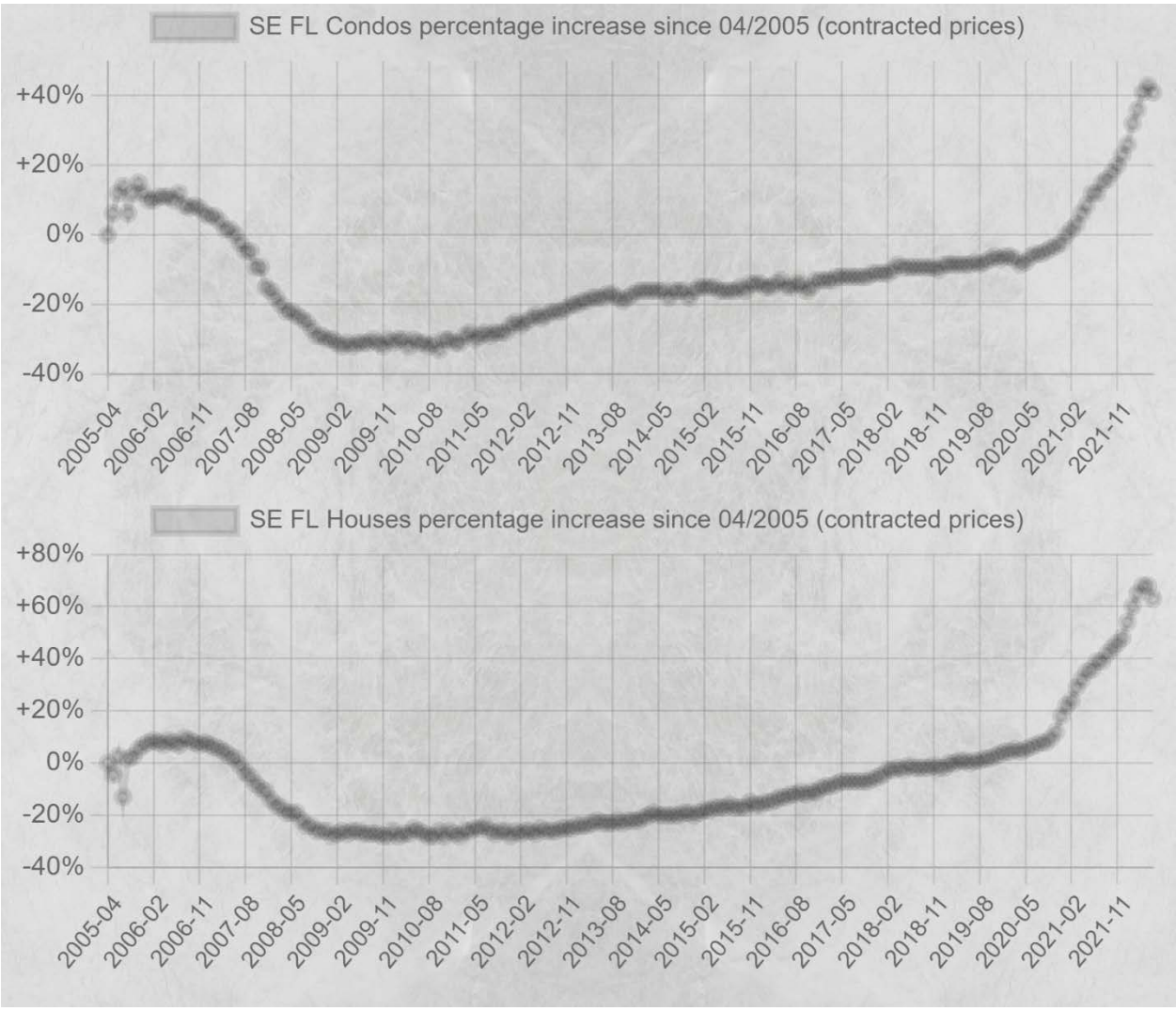


Figure 10: Charts of the change in the median ratios (the realized prices divided by the 2021 county valuation) in comparison to February 2006, for houses and condos in Southeast Florida, using the dates of purchase contracts rather than the closing dates.

Although in this model we have more accurate market sentiment analysis in most periods, we do have noise bias at the edges. The two rightmost data points in this example aggregate properties closed by June 2022 but contracted for purchase in May or June 2022 (because the chosen timespan in this example is user-defined as properties closed from 1/2006 to 6/2022). Because the time elapsing between the contract date and the closing date in said May and June's data is very short, these data points are biased towards cash sales (not contingent on mortgages), which often allow the buyer to negotiate lower prices. This bias can be excluded by disregarding the rightmost edge of the chart. There is also a bias noise at the left edge of the chart because the leftmost points include few but unusual transactions with contract dates as early as April 2005 that were closed in January 2006 or later. This bias can be excluded by disregarding the transactions where the purchase contract date is prior to the beginning of the user-chosen timespan (in this example, January 2006).

5. Hierarchy of Locales

Large locales, e.g., states and metropolitan areas, can be partitioned into smaller locales, e.g., townships and zipcodes, thus enabling the comparison of a locale to its neighbors as well as to its subsuming locales, as follows.

1 33---.Condos.SE Florida.htm
2 33---.Houses.SE Florida.htm
3 330--.Condos..htm
4 330--.Houses..htm
5 3300-.Condos..htm
6 3300-.Houses..htm
7 33004.Condos.Boulevard Gardens.htm
8 33004.Houses.Boulevard Gardens.htm
9 33009.Condos.Golden Isles.htm
10 33009.Houses.Golden Isles.htm
11 3301-.Condos..htm
12 3301-.Houses..htm
13 33010.Condos.Sun-Tan Village.htm
14 33010.Houses.Sun-Tan Village.htm
15 33012.Condos.Hialeah Estates.htm
16 33012.Houses.Hialeah Estates.htm
17 33013.Condos.Hialeah.htm
18 33013.Houses.Hialeah.htm
19 33014.Condos.Palm Springs.htm
20 33014.Houses.Palm Springs.htm
21 33015.Condos.Palm Springs Estates.htm
22 33015.Houses.Palm Springs Estates.htm
23 33016.Condos.Hialeah Estates.htm
24 33016.Houses.Hialeah Estates.htm
25 33018.Condos.Sun-Tan Village.htm
26 33018.Houses.Sun-Tan Village.htm
27 33019.Condos.Golden Isles.htm
28 33019.Houses.Golden Isles.htm
29 3302-.Condos..htm
30 3302-.Houses..htm

Figure 11: Partitioning Southeast Florida into a hierarchy of smaller locales

6. *Statistical Aggregators and Outliers*

A representative statistical aggregator function is a function that matches any set of numbers to a single number intended to be a typical representative of said set. Examples of representative statistical aggregator functions are:

- Median ("Pure Median")
- Average ("Pure Average")
- Average of the input set's elements excluding the lowest 10% and the highest 10% of said set
- 0.5*Median+0.5*Average
- 0.3*Median+0.7*(Average of the input set's elements excluding the lowest 5% and the highest 9% of said set)
- Average of the input set's elements, excluding those elements that are outside predefined outlier thresholds of minimum 0.5 and maximum 1.5.

The present method involves the computation of a representative statistical aggregator function of all the purchase transactions in a given locale during a given period.

The easiest such aggregator function to compute is Pure Average. Among various statistical concerns with the Pure Average function, it may deliver significantly misleading results if the input data is not pre-cleansed off outliers. The Pure Median aggregator is more resilient to outliers, yet it still can benefit from the pre-cleansing of outliers. Outliers can be the result of

(a) erroneous data entry or

(b) the inclusion of esoteric transactions.

From the data cleansing algorithms' point of view, there are several types of outlier cleansing that can be applied to a dataset of said ratios between transactional prices and the fixed-date objective valuation.

- Fixed threshold: disregard transactions with ratios outside of a given range, e.g., the range of 0.5 to 3.0.
- Percentage threshold: for a given category of properties, locale, and period, disregard certain percentages of the lowest and the highest ratios, e.g., the lowest 10% and the highest 5%.
- Statistically insignificant periods: for a given category of properties, locale, and period, if the number of the otherwise qualified transactions in the period is very small, e.g., less than 6, disregard all these transactions, i.e., skip this period for this locale (and for trend presentation purposes, interpolate this period form neighboring periods).
- Date-dependent threshold: for transaction dates far removed from the fixed year of the valuation, allow more liberal thresholds than those close to the valuation year. For example, if the objective valuation date is 1/1/2021, then for transactions in year y, where y<2021, e.g., y=2010, set the minimum threshold to 0.7-0.05*(2021-y).
- Semantic outliers that involve analysis of additional data fields, for example:
  - If there is a data field indicating that this is a foreclosure sale, disregard the transaction for being esoteric, with the expected price being too low.
  - Likewise, for short sales.
  - If there is a data field showing when the house was built (what in the governmental language is called "year of property improvement"), then disregard the transactions where said improvement date falls in between the transaction date and the fixed objective valuation date – this would prevent, e.g., the incorrect relating of the sale price of a building to the appraised value of bare land before the building was built).

7. *Prototype Deployment*

We have deployed a system for Southeast Florida based on the algorithms presented here. Using county and MLS data, the system computes the value trend using transaction closing dates [5] and contract dates [6]. The model is computed for nested areas down to a zipcode and the category of condos vs. single-family homes. For example, the price trend of condominium apartments in Zipcode 33140 is at [7], and for houses is at [8]. The contract-date model for houses and condos in Zipcode 33140 is at [9] and [10].

Authors' contributions

Conceptualization: Rishe; Methodology: Rishe and Adjouadi; Investigation: Rishe, Tamir, and Adjouadi; Writing: Rishe and Tamir; Statistics: Tamir; Funding acquisition: Rishe and Adjouadi. All the authors of this paper concur with its content and consent to its publication.

ACKNOWLEDGMENT

This material is based in part upon work supported by the National Science Foundation under Grant Nos. MRI20 CNS-2018611, MRI CNS-1920182, B-BROIPS FDEP C-2104, and DHS E2055778.

REFERENCES

[1] Lind, H.; Nordlund, B. *Advanced Issues in Property Valuation*; John Wiley & Sons: Hoboken, NJ, USA, 2021.

[2] Francois, P. Comment on Artificial Intelligence and Economic Growth. In *The Economics of Artificial Intelligence: An Agenda*; Agrawal, A., Gans, J., Goldfarb, A., Eds.; University of Chicago Press: Chicago, IL, USA, 2018; pp. 282–289.

[3] International Valuation Standards Council (IVSC). 2021. International Valuation Standards (IVS). Available online: https://www.ivsc.org/standards/

[4] The Appraisal Foundation. 2020-2021. The Uniform Standards of Professional Appraisal Practice (USPAP). Available online: https://www.appraisalfoundation.org/iMIS/TAF/USPAP.aspx

[5] Real Estate Price Trend in SE Florida, Closing Date Model. http://n6.cs.fiu.edu/scripts/Real_estate_trend/

[6] Real Estate Price Trend in SE Florida, Contract Date Model. https://n6.cs.fiu.edu/scripts/Real_estate_trend,_Contract_Date_Model/

[7] Real estate sale price trend in 33140 Miami Beach Condos. http://n6.cs.fiu.edu/scripts/Real_estate_trend/33140.Condos.Miami_Beach.htm

[8] Real estate sale price trend in 33140 Miami Beach Houses. http://n6.cs.fiu.edu/scripts/Real_estate_trend/33140.Houses.Miami_Beach.htm

[9] Real estate sale price trend in 33140 Miami Beach Houses, Contract Date Model. http://n6.cs.fiu.edu/scripts/Real_estate_trend,_Contract_Date_Model/33140.Houses.Miami_Beach.htm

[10] Real estate sale price trend in 33140 Miami Beach Condos, Contract Date Model. http://n6.cs.fiu.edu/scripts/Real_estate_trend,_Contract_Date_Model/33140.Condos.Miami_Beach.htm

Rishe [http://cake.fiu.edu/Rishe] is the inaugural outstanding university professor for Florida International University (FIU). He is a professor at FIU's Knight Foundation School of Computing and Information Sciences. Rishe is the Director of FIU's High-Performance Database Research Center. From 2013 to 2022, Rishe was the executive director of a multi-university Center for Advanced Knowledge Enablement sponsored by the National Science Foundation's industry-university cooperation program. Rishe's research has been funded by the U.S. Government and Industry at over $50 million. Rishe's inventions include 26 U.S. patents. Rishe is a Fellow medalist of the National Academy of Inventors. In 2021, Rishe received the IBM Global University Programs award. Rishe's publications include 440 papers and six books. Rishe's pioneering geospatial system TerraFly has been highlighted by the National Science Foundation in its reports to U.S. Congress.

Tamir [https://userweb.cs.txstate.edu/~dt19/] is an associate professor in the Department of Computer Science at the Texas State University. He obtained the PhD-CS from Florida State University and the MS/BS-EE from Ben-Gurion University, Israel. He has managed applied research and design in DSP Core technology in Motorola-SPS/Freescale. He has served as an assistant/associate professor at the CS Department at Florida Tech, has served as a member of the MPEG committee and has worked in the Applied Research Division, Tadiran, Israel.

Adjouadi is the WARE Distinguished Professor at FIU's Department of Electrical and Computer Engineering. Since 1993, Adjouadi has directed the FIU Center for Advanced Technology and Education [http://cate.fiu.edu]. Adjouadi's doctorate is from the University of Florida (1985).

5. Towards Real-time House Detection in Aerial Imagery Using Faster Region-based Convolutional Neural Network.9pp

Follows

Towards Real-time House Detection in Aerial Imagery Using Faster Region-based Convolutional Neural Network

Ahmed, Khandaker Mamun; Ghareh Mohammadi, Farid; Matus, Manuel; Shenavarmasouleh, Farzan; Manella Pereira, Luiz; Ioannis, Zisis; Amini, M. Hadi

Abstract: ***In the past few years, automatic building detection in aerial images has become an emerging field in computer vision. Detecting the specific types of houses will provide information in urbanization, change detection, and urban monitoring that play increasingly important roles in modern city planning and natural hazard preparedness. In this paper, we demonstrate the effectiveness of detecting various types of houses in aerial imagery using Faster Region-based Convolutional Neural Network (Faster-RCNN). After formulating the dataset and extracting bounding-box information, pre-trained ResNet50 is used to get the feature maps. The fully convolutional Region Proposal Network (RPN) first predicts the bounds and objectness score of objects (in this case house) from the feature maps. Then, the Region of Interest (RoI) pooling layer extracts interested regions to detect objects that are present in the images. To the best of our knowledge, this is the first attempt at detecting houses using Faster R-CNN that has achieved satisfactory results. This experiment opens a new path to conduct and extent the works not only for civil and environmental domain but also other applied science disciplines.***

Index Terms: ***RCNN, Neural Network, Deep Learning, Convolution, Mini batch***

1. Introduction

In this section, we present the motivation for the development of an application to detect houses in aerial images. Subsequently, we discuss the prior works that have recently been published and explain how our proposed framework can be beneficial in the modern urbanized world. We also show the novelty of this paper, which is followed by a brief description of the paper's organization.

Manuscript received February 13, 2023.

Corresponding Author: M. Hadi Amini, moamini@fiu.edu

Khandaker Mamun Ahmed, M. Hadi Amini and Luiz Manella Pereira are with the Knight Foundation School of Computing and Information Sciences, Florida International University (FIU), Miami, FL, USA; and the Sustainability, Optimization, and Learning for InterDependent networks laboratory (solid lab), FIU, Miami, FL, USA

Farid Ghareh Mohammadi is with the Department of Radiology, Center for Augmented Intelligence (CAI), Mayo Clinic, Jacksonville, FL, USA

Manuel Matus and Ioannis Zisis is with the Dept. of Civil & Environ. Engineering Florida International University, Miami, FL, USA

Farzan Shenavarmasouleh is with the R&D Department, MediaLab Inc., GA, USA

1.1 Motivation

House detection is an important problem in computer vision and pattern recognition which has gained considerable attention in the past few decades [1]–[3]. Due to rapid urbanization, detecting houses plays a salient role in modern city planning, urban monitoring, change detection, and population estimation. Moreover, building shape related information can provide valuable input in engineering and risk applications related to natural hazards (e.g. extreme wind events, flooding, etc.). Aerial imagery is one of the prominent data sources for urban monitoring because it extracts various information such as roads, trees, buildings, etc. Although aerial imagery provides valuable insights, extracting appropriate features from them is a challenging task.

On the other hand, in recent years, deep learning models, especially Convolutional Neural Network (CNN) based models, have become a popular choice among the researchers for its state-of-the-art success in image classification, object detection, and localization tasks [4]–[7]. Faster-RCNN is a recently proposed object detection algorithm that has achieved state of-the-art results in ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [8], [9]. In this work, we have utilized a faster-RCNN algorithm to detect buildings in aerial images.

1.2 Literature Review

In this section, we first talk about the history the algorithm applied in this work followed by a brief review of the prior works.

1.2.1 CNN and RCNN family of Algorithms:

Due to the rapid developments of science and technology (e.g., advancements in automated vehicles, robotic navigation, and object tracking), object detection has become a prominent field of study. The goal of object detection is to find the location of an object from a given image and mark the object in an appropriate category. However, object detection is a challenging task. The object's orientation, location, size, and

altitude can vary greatly in an image, making the task more difficult to solve. In the human visual system, we not only see and identify an object, we can identify multiple overlapping objects in diverse backgrounds. Moreover, we can classify these different objects and identify their boundaries, differences, and relationship to one another. However, in the field of computer vision, CNN-based architectures are applied successfully to solve various detection related tasks such as face detection, pedestrian detection and vehicle detection [10]–[14].

The first successful CNN architecture was developed by Yann Lecun in 1998 to recognize handwritten digits on checks [15]. In 2012, more than 12 years later, Alex Krizhevsky et al. followed his path and built the famous AlexNet algorithm that won the ImageNet challenge [16]. Since then, CNN architectures have become the gold standard for solving computer vision tasks and are now outperforming humans in some scenarios.

In 2014 Girshick et al. proposed the Regions with CNN features (R-CNN) algorithm for object detection, which is the first algorithm of the R-CNN family of algorithms [17]. RCNN achieved the mean average precision (mAP) result of 53.3% in PASCAL VOC dataset. To capture all possible objects' locations from a given image, authors applied the selective search algorithm [18]. The selective search algorithm proposes 2k regions for an image. In Figure 1, two examples of selective search are given where different sized scales are used to capture all possible objects. Each proposed region is warped to a compatible form of 227×227 pixels and forward propagated through the CNN architecture to compute feature maps. Next, the Support Vector Machine (SVM) algorithm is utilized to compute the classification score. In the RCNN architecture the workflow is like: an input image is given to detect possible objects; the selective search algorithm proposes ∼2k regions which are forwarded to the CNN layers, and the CNN architecture generates feature maps to detect which objects are present in the image. To compute the region proposal and features for images, R-CNN requires 13 s/image on a GPU integrated environment and 53 s/image on a CPU based environment, which is a significantly high computation time. Therefore, to minimize the computation time required by RCNN, an improved version of RCNN named Fast-RCNN was proposed by the same author Ross Girshick [19] in 2015.

The Fast-RCNN model requires an input image and a set of object proposals for its computation. Initially, it processes the whole image with several convolutional (conv) layers and max-pooling layers to produce the feature maps. Then, a fixed□length feature vector from the feature map is extracted by the RoI pooling layer to classify objects. Fast-RCNN is 25 times faster than R-CNN with the test time of 2 seconds per image. Even though Fast-RCNN significantly improved the processing time and model's performance, the selective search was still the bottleneck that slowed down the overall process. Region proposals are dependent on the feature maps and reusing the feature maps to generate region proposals will be cost-free. Taking this idea into consideration, Ren et al. developed the faster R-CNN that exceptionally improved the overall model performance [8]. In Figure 2, we show a faster R-CNN algorithm where conv layers compute the feature maps and RPN layer extracts region proposals from the feature maps for classification. The faster R-CNN algorithm can detect objects in real time with the computational time of 0.2 seconds per image.

Figure 3 demonstrates the performance comparison of the R-CNN architectures where we can see that faster R-CNN reduced processing time by 250x, whereas Fast-RCNN had a reduction of 25x against the base case processing time of x for R-CNN. Both faster and Fast-RCNN maintained the same mean average precision (mAP) score of 66.9%, where R-CNN architecture's mAP score was 66.0%. [1]

1.2.2 Recent Works on House Detection:

Buildings are the primary source of information for urban planners and, many governmental and non-governmental agencies as they provide the holistic overview of a geographical area. However, building detection is a challenging task because of its complex appearance, variant shapes, and surroundings. In the past few years, researchers have proposed several building extraction methods and followed various approaches [20]–[22]. Although building detection methods with good performance have evolved significantly over the years, there are still many aspects that have not been considered and need improvements.

Stankov et al. [23], [24] exploited the multispectral information and applied a grayscale hit-or-miss transform (HMT) method for building detection. In the paper, authors transformed the multispectral images to grayscale images in order to apply grayscale HMT. Sirmacek et al. [25] extracted shadow information and areas of interest using invariant color features and utilized edge information building detection. In [26], Ziaei et al. presented a comparison between three object-based models for urban feature classification from WorldView-2 images, where they have shown that rule-based classification outperformed support vector machines (SVM), and nearest neighbour (NN) algorithms. Building extraction from Quickbird images is presented by Lefevre et al. [27] by using an adaptive ` binary HMT method. Authors also proposed a

[1]Stanford lecture notes on CNN by Fei Fei Li and Andrej Karpathy

clustering-based approach to convert grayscale image to binary image and to determine operators parameters automatically. In [28], Grinias et al. presented a novel segmentation algorithm based on a Markov random field model for building and road detection. To detect changes of buildings from VHR imagery, Guo et al. [29] presented a parameter mining approach by introducing GIS data. For automatically extracting and recognizing 2- D building shape information, Sahar et al. [30] used vector parcel geometries and their attributes to simplify the building extraction task. Huang et al. [31] introduced a framework for building extraction from high-resolution imagery aiming to alleviate Morphological Building Index (MBI) algorithm's limitations. Benarchid et al. [32] used shadow information and object-based approach to extract buildings where they first used object-based classification to detect building and then the invariant color features to extract shadow information of the buildings. Based on shadow detection, Chen et al. [33] proposed a superpixel segmentation algorithm for splitting input image into patches, and the Level Set segmentation algorithms is leveraged to extract buildings for detection.

In this paper, we present a Faster RCNN based deep learning model that can detect different houses in aerial images.

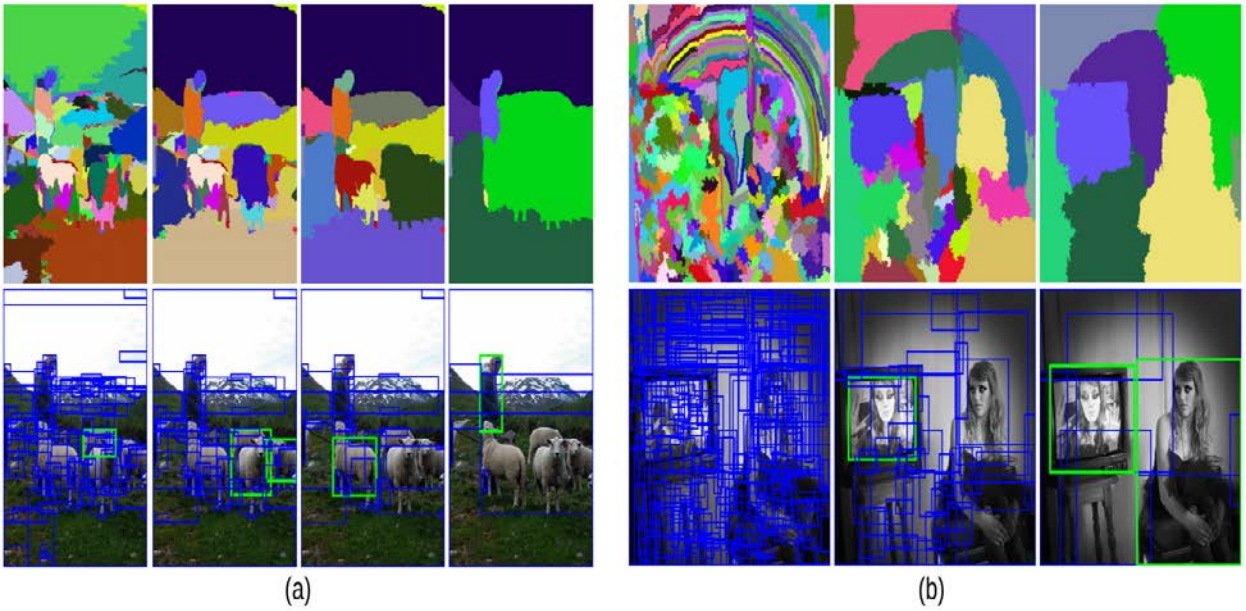


Figure 1: Two examples of selective search showing the necessity of different scales. On the left we find many objects at different scales. On the right we necessarily find the objects at different scales as the girl is contained by the tv [18].

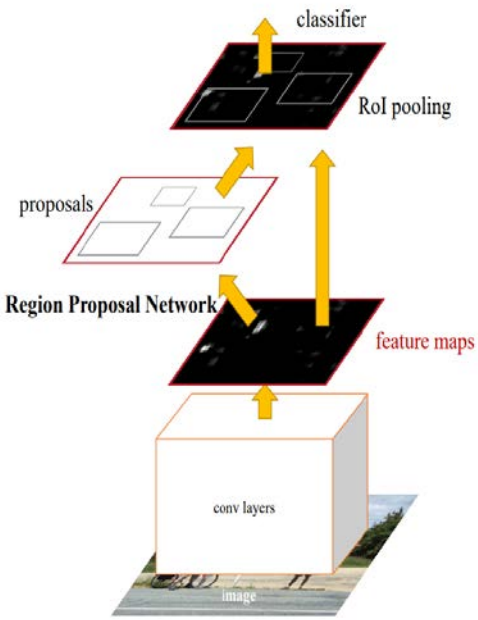
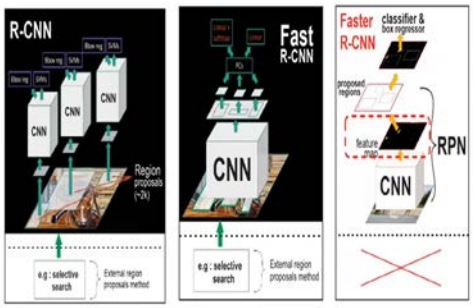


Figure 2: Faster-RCNN architecture.



	R-CNN	Fast R-CNN	Faster R-CNN
Test time per image	50 seconds	2 seconds	0.2 seconds
Speed-up	1x	25x	250x
mAP (VOC 2007)	66.0%	66.9%	66.9%

Figure 3: Performance comparison of R-CNN architectures: R CNN, Fast-RCNN, Faster R-CNN. [1]

1.3 Contribution

Faster-RCNN is one of the promising algorithms for object detection that has also opened up the area of real time object detection. In some situations, we need to extract the building's information in real time and our proposed method can be a good fit for such scenarios. It is our understanding that faster-RCNN based house detection technique, which paves the way for real time detection, has not been considered in previous works. The main contributions of this paper are listed as follows:

- House detection in aerial images leveraging faster R-CNN algorithm that paves the way for real time detection.
- Bounding-box information extraction and preprocessing of the dataset to remove inconsistent data that may hamper the overall performance of the model.
- Demonstrate the effectiveness of data augmentation such as random rotation, horizontal flip and shearing to im□prove performance and generalizability, and avoid over-fitting.
- Demonstrate our model's performance by considering average precision, loss function, prediction scores and image precision.

1.4 Organization

The paper is organized as follows: Section II presents the methodology of the work including data pre-processing, data augmentation and the house detection technique. Section III represents experimental setup. Section IV is dedicated for result analysis. Finally, Section V concludes the paper.

2. Proposed Method

This section discusses data pre-processing, and data augmentation techniques, and the methodology used to detect houses. In Figure 4, we show the overall architecture of our proposed model that includes dataset generation, data preprocessing, data augmentation, object detection, and results afterwards.

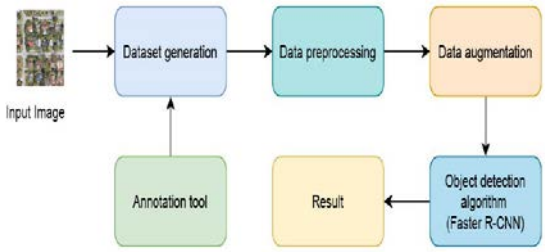


Figure 4: Overview of methodology adopted in this study

2.1 Data pre-processing

In our dataset, we have aerial images and XML files containing the annotation information of the images. XML file is an extensible markup language file where components of the file are described by tags, and texts in between the start tag and end tag are the contents of the component. From the XML files, we extract the associated bounding-box information (for our case its the aerial image file, xmin, ymin, xmax, ymax and label) of each image. In the generated dataset, we observed 37 different labels / categories of houses where most of them are redundant (e.g., typo and inconsistent labels). For example the category of T shaped houses were labelled as t shape, t-shaped, t type, type t and t-shape which is inconsistent and it can be minimized to one category. After analyzing 37 labels, we concluded that 37 different labels can be minimized to only 5 categories (T shaped, L shaped, C shaped, Rectangular shaped, and U shaped). Moreover, we had some anomalies in the extracted information such as xmin > xmax or ymin > ymax. In such cases, if possible, we exchanged min and max values without changing the bounding-box information of an object, otherwise we disregarded them due to incorrect bounding boxes.

2.2 Data augmentation

Data augmentation is a technique to artificially expand the dataset size by marginally modifying the original data. Data augmentation helps to avoid overfitting and improves model's performance. In images data augmentation technique is performed by flipping, random rotation, shifting, or shearing the original image. Deep learning is a data-hungry technique that yields better performance with larger dataset, avoids over-fitting, and improves the model's generalizability. Therefore to improve model performance and avoid overfitting, we augmented our dataset using horizontal flip, random rotation with the angle value of 10 degrees, shears with the value of 0.1, and

random rotation with randomly generated angle value. In Figure 5, we demonstrated the augmented results after applying the data augmentation techniques.

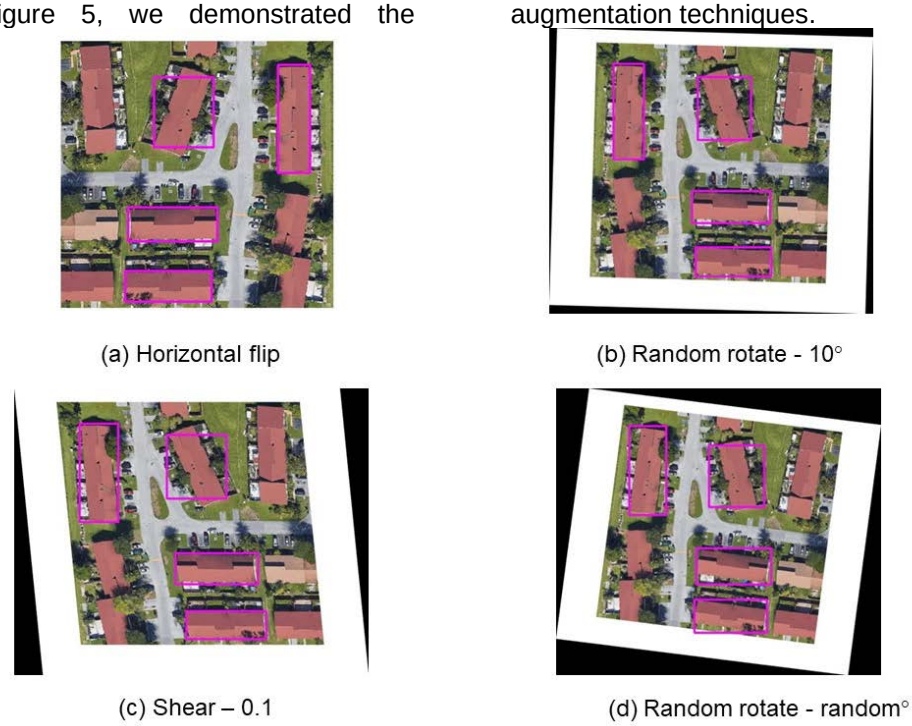


Figure 5: Data augmentation: 5a Horizontal flip; 5b Random rotation with 10°;5c Shear with 0.1;5d Random rotation with a random value.

2.3 House Detection using Faster-RCNN

The most widely used state-of-the-art object detection technique of the R-CNN family is Faster R-CNN that was first published in 2015 [8]. In the R-CNN family of papers, the evolution among versions is usually in terms of computational efficiency, processing time, and performance improvement (i.e. mAP). These networks usually consist of

1. A region proposal algorithm to generate “bounding boxes” or locations of possible objects in the image.
2. A feature generation stage to obtain features of these objects (usually using a CNN).
3. A classification layer to predict which class an object belongs to. 4) A regression layer to make the coordinates of the object bounding boxes more precise.

To generate feature maps (e.g., Figure 7), ResNet50 is utilized in the initial stage where the input image goes through a set of convolutional layers, pooling layers and fully connected layers. After generating feature maps, RPN layer which is a small network, takes the feature map as an input, slides over it, and outputs a set of rectangular object proposals. Nine region proposals (anchors) are predicted at each sliding window location with respect to the center (Figure 8) of the anchor associated with scales of (128 x 128, 256 x 256, 512 x 512) and aspect ratios of (1:1, 1:2 and 2:1) (Figure 6). A binary class label of being an object or not an object is assigned to each anchor for RPN training based on the Intersection-over-Union (IoU) overlap with the ground-truth box. An anchor is considered positive if it has the highest IoU with any ground truth box or is greater than 0.7. If the IoU is less than 0.3 it is labeled as negative. The anchors which are neither positive nor negative (greater than 0.3 and less than 0.7) are disregarded from the RPN training. The loss function of RPN is defined as:

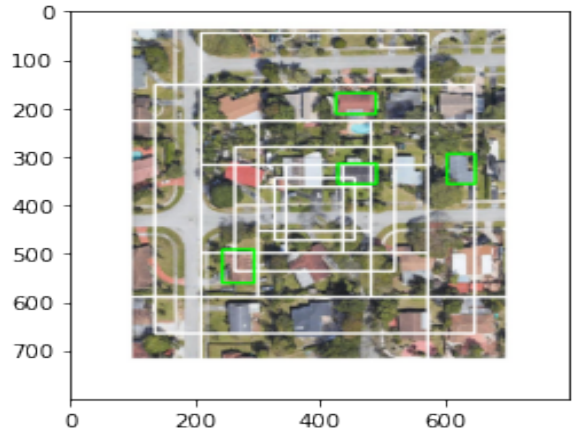


Figure 6: An example of generating 9 anchors from a single centroids with different scales and aspect ratios.

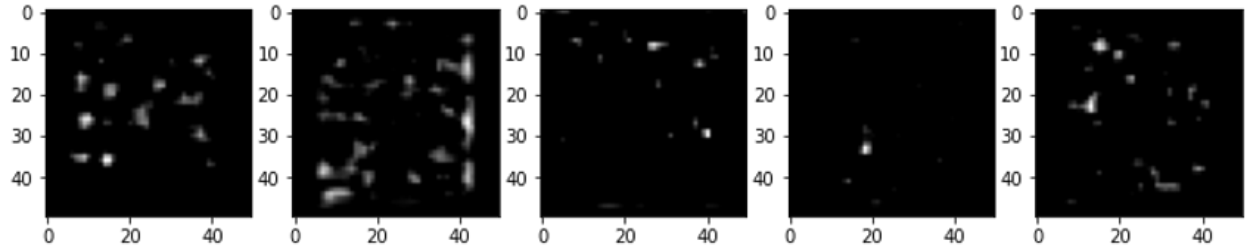


Figure 7: Sample feature map

$$L(p_i, t_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(P_i, P_i^*) + \lambda \frac{1}{N_{reg}} \sum_i P_i^* L_{reg}(t_i, t_i^*)$$

Here, i is the index of an anchor in a mini-batch and p_i is the predicted probability of anchor i being an object. The ground-truth label $P_i *$ is 1 if the anchor is positive and is 0 if the anchor is negative. t_i is a vector representing the 4 parameterized coordinates of the predicted bounding box and $t_i *$ is that of the ground-truth box associated with a positive anchor. The classification loss L_{cls} is log loss over two classes (object vs. not object). For the regression loss, we use $L_{reg}(t_i , t_i *) = R(t_i - t_i *)$ where R is the robust loss function (smooth L1). The term $P_i * L_{cls}$ means the regression loss is activated only for positive anchors $P_i * = 1$ and is disabled otherwise (i.e. $P_i * = 0$). The outputs of the cls and reg layers consist of p_i and t_i respectively. The two terms are normalized by N_{cls} and N_{reg} and weighted by a balancing parameter λ.

For the model training, the batch size is defined to 16 and stochastic gradient descent (SGD) optimizer is applied with the learning rate of 0.005, momentum of 0.9 and weight decay of 0.005.

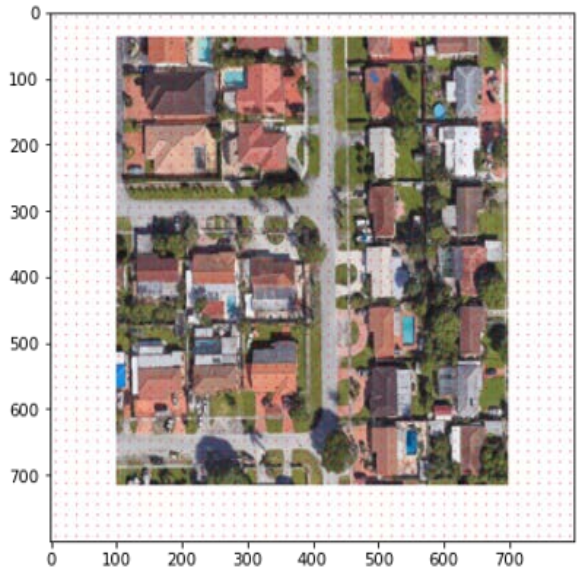


Figure 8: Centriods of RPN.

3. EXPERIMENTAL SETUP

The entire experiment is carried out in Google Colab environment developed by Google as a simulation environment. The experiment leverages Colab environment utilizing GPU runtime settings using python as the programming language. The deep learning object detection classifier has been implemented using python version 3.7.3 and the PyTorch framework.

4. EXPERIMENTAL EVALUATION

This section provides a brief description of the dataset we have used for our experiments followed by the performance evaluation of our proposed work.

4.1 Dataset Description

In this experiment, we explored google earth images to detect houses of different shapes. In Figure 9, we demonstrate the process of creating our dataset using LabelMe [34] annotation tool where house objects are manually annotated in each image. The annotation tool then generates an XML file containing the annotated information for each image. (Figure 11) shows the structure of a sample xml file after completing the annotation process and in Figure 10 we show a sample annotated image afterwards. Finally, the annotation files along with the associated aerial image dataset are downloaded from the LabelMe application for carrying out the experiment.

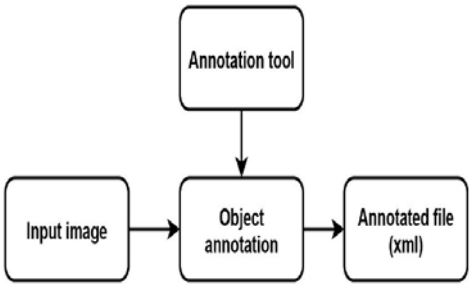


Figure 9: Flowchart for dataset annotation.

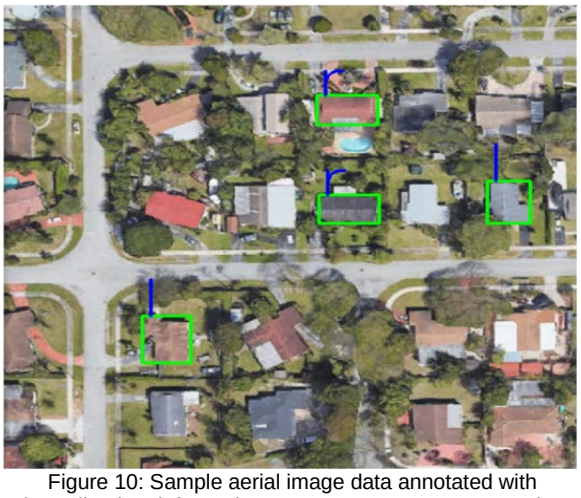


Figure 10: Sample aerial image data annotated with bounding box information. Here, r represents rectangular shaped houses and l represents l shaped houses


```
−<annotation>
  <filename>17.jpg</filename>
  −<folder>
    users/WindEngineering//topics_in_wind_engineering_final/sunil
  </folder>
  −<source>
    <submittedBy>Manuel Matus</submittedBy>
  </source>
  −<imagesize>
    <nrows>945</nrows>
    <ncols>1072</ncols>
  </imagesize>
  −<object>
    <name>building 1</name>
    <deleted>0</deleted>
    <verified>0</verified>
    <occluded>no</occluded>
    <attributes>c shaped</attributes>
   −<parts>
      <hasparts/>
      <ispartof/>
    </parts>
    <date>02-Dec-2019 03:32:24</date>
    <id>0</id>
    <type>bounding_box</type>
   −<polygon>
      <username>anonymous</username>
     −<pt>
        <x>169</x>
        <y>531</y>
      </pt>
     −<pt>
        <x>284</x>
        <y>531</y>
```


Figure 11: XML file: Annotation information of images such as shape, number, bounding-box information

4.2 Experimental results

Object detectors performance is measured by average precision (AP), image precision and loss functions. In our experiment, we evaluated our methods performance by average precision, image precision and loss function. We defined different number of epochs to observe the model's performance. In our observation, the simulation performs better with twenty epochs. In Figure 12, we demonstrate average precision in different IoU thresholds: 0.50, 0.55, 0.60, 0.65, 0.70, 0.75. As the IoU threshold increases the average precision decreases naturally. Moreover, in Figure 13, we show the average image precision by comparing all IoU thresholds. From Figure 13, we can see that image precision increases moderately for 20 epochs. In Figure 14, we show the loss function against the number of iterations where we observe that after 400 iterations with twenty epochs the loss function is converged. The equations for calculating precision, average precision are discussed in the followings where t_p = True positive; f_p = False positive; t_n = True negative; f_n = False negative.

$$Precision(P) = \frac{tp}{tp + fp} \tag{1}$$

$$Recall(R) = \frac{tp}{tp + fn} \tag{2}$$

4.2.1 Intersection over union (IoU)

IoU measures the overlap between 2 boundaries. We use that to measure how much our predicted boundary overlaps with the ground truth. In our dataset, we defined various IoU threshold r ∈ {0.5, ..., 0.75} in classifying whether the prediction is a true or a false positive. Intersection over Union (IoU) for comparing similarity between the ground-truth and predicted shapes A, B ⊆ S ∈ Rn is attained by equation 3.

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{3}$$

4.2.2 Interpolated precision

The interpolated precision, p_{interp}, is calculated at each recall level, r, by taking the maximum precision measured for that r. The formula is given as such:

$$p_{interp}(r) = \max_{r' \geq r} P(r') \tag{4}$$

In our experiment an average for the 6-point interpolated average precision (AP) is calculated. And the formula to calculate the AP is attained by:

$$AP = \frac{1}{6} \sum_{r \in \{0.5,\ldots,0.75\}} AP_r = \frac{1}{6} \sum_{r \in \{0.5,\ldots,0.75\}} p_{interp}(r) \tag{5}$$

where

$$p_{interp}(r) = \max_{r' \geq r} P(r')$$

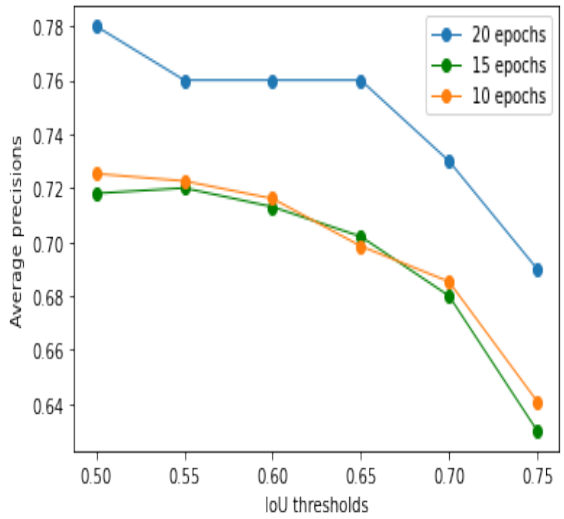


Figure 12: Average precision

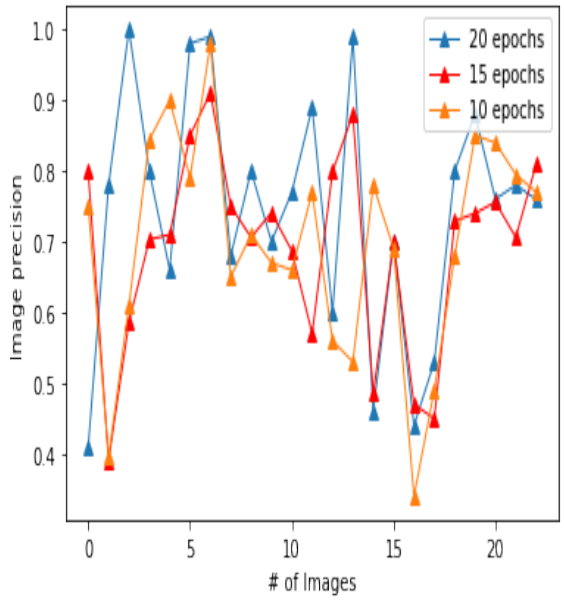


Figure 13: Image precision

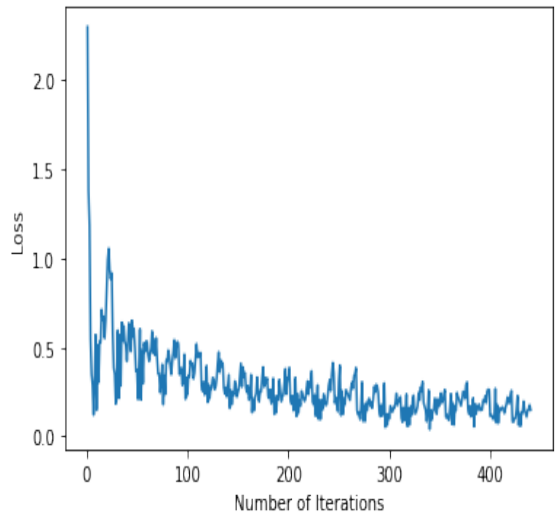


Figure 14: Loss function.

5. Conclusion And Future Works

House detection is a fundamental but challenging issue in the field of aerial and satellite image analysis. It provides valuable information in different domains including civil engineering, urbanization, and modern city planning. During the last few years, considerable efforts have been made to develop various methods for detecting houses in aerial images. In this paper, we present a Faster-RCNN based house detection method that achieved a satisfactory result. Our proposed method can be utilized in real time object/house detection scenarios. A wide range of ensembles of faster RCNN is being utilized in various contexts such as pedestrian detection, vehicle detection, and face detection. In this experiment, we have leveraged pretrained resnet-50 model to detect houses in aerial images. A performance comparison of various models, such as VGG19, SeNet, GoogleNet, MobileNetV2, DenseNet201, and InceptionResNetV2, is important for both application and academic purposes and thus remains an integral part of our future research.

Acknowledgment

In this work, we have leveraged faster RCNN algorithm and carried out the experiment in Google Colab environment. We are thankful to both communities. We also acknowledge the resources and support of Sustainability, Optimization, and Learning for InterDependent networks laboratory (solid lab) at Knight Foundation School of Computing and Information Sciences (KFSCIS), Florida International University (www.solidlab.network). We also acknowledge the effort of Divya Saxena from the KFSCIS, FIU, funded by NSF grant CNS-2018611 at the FIU High Performance Database Research Center.

This work was partially supported by the Graduate Assistantships in Areas of National Need (GAANN) fellowship from the Department of Education grant P200A210087.

Declaration Of Competing Interest

Authors declare no conflict of interest.

References

[1] S. Zou and L. Wang, “Detecting individual abandoned houses from google street view: A hierarchical deep learning approach,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 175, pp. 298– 310, 2021.

[2] S. Law, B. Paige, and C. Russell, “Take a look around: using street view and satellite images to estimate house prices,” ACM Transactions on Intelligent Systems and Technology (TIST), vol. 10, no. 5, pp. 1–19, 2019.

[3] J. A. Tullis and J. R. Jensen, “Expert system house detection in high spatial resolution imagery using size, shape, and context,” Geocarto International, vol. 18, no. 1, pp. 5–15, 2003.

[4] W. Rawat and Z. Wang, “Deep convolutional neural networks for image classification: A comprehensive review,” Neural computation, vol. 29, no. 9, pp. 2352–2449, 2017.
[5] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, “Object detection with deep learning: A review,” IEEE transactions on neural networks and learning systems, vol. 30, no. 11, pp. 3212–3232, 2019.
[6] X. Wu, D. Sahoo, and S. C. Hoi, “Recent advances in deep learning for object detection,” Neurocomputing, vol. 396, pp. 39–64, 2020.
[7] K. M. Ahmed, T. Eslami, F. Saeed, and M. H. Amini, “Deepcovidnet: Deep convolutional neural network for covid-19 detection from chest radiographic images,” in 2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). IEEE, 2021, pp. 1703–1710.
[8] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” arXiv preprint arXiv:1506.01497, 2015.
[9] “ImageNet Large Scale Visual Recognition Challenge 2016 Results (ILSVRC2016),” http://www.image-Net.org/challenges/LSVRC/2016/results, [Online; accessed 06-April-2021].
[10] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, “Unitbox: An advanced object detection network,” in Proceedings of the 24th ACM international conference on Multimedia, 2016, pp. 516–520.
[11] S. S. Farfade, M. J. Saberian, and L.-J. Li, “Multi-view face detection using deep convolutional neural networks,” in Proceedings of the 5th ACM on International Conference on Multimedia Retrieval, 2015, pp. 643–650.
[12] S. Yang, P. Luo, C.-C. Loy, and X. Tang, “From facial parts responses to face detection: A deep learning approach,” in Proceedings of the IEEE international conference on computer vision, 2015, pp. 3676–3684.
[13] Z. Jiang and D. Q. Huynh, “Multiple pedestrian tracking from monocular videos in an interacting multiple model framework,” IEEE transactions on image processing, vol. 27, no. 3, pp. 1361–1375, 2017.
[14] D. M. Gavrila and S. Munder, “Multi-cue pedestrian detection and tracking from a moving vehicle,” International journal of computer vision, vol. 73, no. 1, pp. 41–59, 2007.
[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.
[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” Advances in neural information processing systems, vol. 25, pp. 1097–1105, 2012.
[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580–587.
[18] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, “Selective search for object recognition,” International journal of computer vision, vol. 104, no. 2, pp. 154–171, 2013.
[19] R. Girshick, “Fast r-cnn,” in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440–1448.
[20] D. A. Yudin, V. Adeshkin, A. V. Dolzhenko, A. Polyakov, and A. E. Naumov, “Roof defect segmentation on aerial images using neural networks,” in International Conference on Neuroinformatics. Springer, 2020, pp. 175–183.
[21] H. Miura, T. Aridome, and M. Matsuoka, “Deep learning-based identification of collapsed, non-collapsed and blue tarp-covered buildings from post-disaster aerial images,” Remote Sensing, vol. 12, no. 12, p. 1924, 2020.
[22] A. D. Schlosser, G. Szabo, L. Bertalan, Z. Varga, P. Enyedi, and S. Szabo, “Building extraction using orthophotos and dense point cloud ´ derived from visual band aerial imagery based on machine learning and segmentation,” Remote Sensing, vol. 12, no. 15, p. 2397, 2020.
[23] K. Stankov and D.-C. He, “Building detection in very high spatial resolution multispectral images using the hit-or-miss transform,” IEEE Geoscience and Remote Sensing Letters, vol. 10, no. 1, pp. 86–90, 2012.
[24] ——, “Detection of buildings in multispectral very high spatial resolution images using the percentage occupancy hit-or-miss transform,” IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 7, no. 10, pp. 4069–4080, 2014.
[25] B. Sirmacek and C. Unsalan, “Building detection from aerial images using invariant color features and shadow information,” in 2008 23rd International Symposium on Computer and Information Sciences. IEEE, 2008, pp. 1–5.
[26] Z. Ziaei, B. Pradhan, and S. B. Mansor, “A rule-based parameter aided with object-based classification approach for extraction of building and roads from worldview-2 images,” Geocarto International, vol. 29, no. 5, pp. 554–569, 2014.
[27] S. Lefevre, J. Weber, and D. Sheeren, “Automatic building extraction ` in vhr images using advanced morphological operators,” in 2007 Urban Remote Sensing Joint Event. IEEE, 2007, pp. 1–5.
[28] I. Grinias, C. Panagiotakis, and G. Tziritas, “Mrf-based segmentation and unsupervised classification for building and road detection in peri□urban areas of high-resolution satellite images,” ISPRS journal of photogrammetry and remote sensing, vol. 122, pp. 145–166, 2016.
[29] Z. Guo and S. Du, “Mining parameter information for building extraction and change detection with very high-resolution imagery and gis data,” GIScience & Remote Sensing, vol. 54, no. 1, pp. 38–63, 2017.
[30] L. Sahar, S. Muthukumar, and S. P. French, “Using aerial imagery and gis in automated building footprint extraction and shape recognition for earthquake risk assessment of urban inventories,” IEEE Transactions on Geoscience and Remote Sensing, vol. 48, no. 9, pp. 3511–3520, 2010.
[31] X. Huang and L. Zhang, “Morphological building/shadow index for building extraction from high-resolution imagery over urban areas,” IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 5, no. 1, pp. 161–172, 2011.
[32] O. Benarchid, N. Raissouni, S. El Adib, A. Abbous, A. Azyat, N. B. Achhab, M. Lahraoua, and A. Chahboun, “Building extraction using object-based classification and shadow information in very high reso□lution multispectral images, a case study: Tetuan, morocco,” Canadian Journal on Image Processing and Computer Vision, vol. 4, no. 1, pp. 1–8, 2013.
[33] D. Chen, S. Shang, and C. Wu, “Shadow-based building detection and segmentation in high-resolution remote sensing image.” journal of multimedia, vol. 9, no. 1, pp. 181–188, 2014.
[34] “LabelMe, the open annotation tool,” http://labelme.csail.mit.edu/Release3.0/, accessed: 2021-11-10