The Triplex Approach for Recognizing Semantic Relations from Noun Phrases, Appositions, and Adjectives

Seyed Iman Mirrezaei[1], Bruno Martins[2], and Isabel F. Cruz[1]

[1] ADVIS Lab, Department of Computer Science, University of Illinois at Chicago, USA
smirre2@uic.edu, ifc@cs.uic.edu

[2] Instituto Superior Técnico, Universidade de Lisboa, Portugal
bruno.g.martins@ist.ul.pt

Abstract. Discovering knowledge from textual sources and subsequently expanding the coverage of knowledge bases like DBPedia or Freebase currently requires either extensive manual work or carefully designed information extractors. Information extractors capture triples from textual sentences. Each triple consists of a subject, a predicate/property, and an object. Triples can be mediated via verbs, nouns, adjectives, and appositions. We propose TRIPLEX, an information extractor that complements previous efforts, concentrating on triples related to nouns, adjectives, and appositions. TRIPLEX automatically constructs templates expressing noun-mediated triples from a bootstrapping set. The bootstrapping set is constructed without manual intervention by creating templates that include syntactic, semantic, and lexical constraints. We report on an automatic evaluation method to examine the output of information extractors both with and without the TRIPLEX approach. Our experimental study indicates that TRIPLEX is a promising approach for extracting noun-mediated triples.

Keywords: Open information extraction, relation extraction, noun-mediated relation triples, compound nouns, appositions.

1 Introduction

Deriving useful knowledge from unstructured text is a challenging task. Nowadays, knowledge needs to be extracted almost instantaneously and automatically from continuous streams of information such as those generated by news agencies or published by individuals on the social web to enrich properties related to people, places, and organizations in existing large-scale knowledge bases, such as Freebase [3], DBPedia [1], or Google's Knowledge Graph [10]. Subsequently these values can be used by search engines to provide answers for user queries (e.g., the resignation date of a given politician, or the ownership after a company acquisition). For many natural language processing (NLP) applications, including question answering, information retrieval, machine translation, and information extraction it is important to extract facts from text. For example, a question answering system may need to find the location of *Microsoft Visitor Center* in the sentence *The video features the Microsoft Visitor Center, located in Redmond.* Open Information Extractors (OIE) aim to extract triples from text, with each triple consisting of a subject, a predicate/property, and an object. These triples can be expressed via verbs, nouns, adjectives, and appositions. Most OIE systems described in the literature, such as TextRunner [2], WOE [11], or ReVerb [7] focus on the extraction of verb-mediated triples. Other OIE systems, such as OLLIE [9], ClauseIE [6], Xavier and Lima's system [13], or ReNoun [14], may also, or only, extract noun-mediated triples from text. OLLIE was the first approach for simultaneously extracting verb-mediated and noun-mediated triples, although it can only capture noun-meditated triples that are expressed in verb-mediated formats. For example, OLLIE can extract a triple `<Bill Gates; be co-founder of; Microsoft>` from *Microsoft co-founder Bill Gates spoke at a conference* but cannot extract a triple `<Microsoft; headquarter; Redmond>` from *Microsoft is an American corporation headquartered in Redmond.* ClauseIE extracts noun-mediated triples from appositions and possessives based upon a predefined set of rules. ReNoun uses seeds (i.e., examples gathered through manually crafted rules) and an ontology to learn patterns for extracting noun-mediated triples.

The OIE system that we have built is named TRIPLEX. It is designed specifically to extract triples from noun phrases, adjectives, and appositions. Systems like OLLIE, which can only extract triples corresponding to relations expressed through verb phrases, can be assisted by TRIPLEX, which extracts triples from grammatical dependency relations, involving noun phrases and modifiers that correspond to adjectives and appositions. TRIPLEX recognizes patterns that express noun-mediated triples during its automatic bootstrapping process. The bootstrapping process uses Wikipedia pages to find sentences that express triples. Then, it constructs templates from dependency paths in sentences from the bootstrapping set. The templates express how noun-mediated triples occur in sentences and allow information to be extracted relating to different levels of text analysis, from lexical (i.e., word tokens) and shallow syntactic features (i.e., parts of speech tags), to features resulting from a deeper syntactic analysis (i.e., features derived from dependency parsing). In addition, semantic constraints may be included in some templates to obtain more precise extractions. Templates are then generalized to broaden their coverage (i.e., those with similar constraints are merged together). Finally, the templates can be used to extract triples from previously unseen text. We evaluated TRIPLEX according to the automated framework suggested by Bronzi et al. [4], extending it to assess noun-mediated triples.

The remainder of this paper is organized as follows: In Section 2, we briefly summarize related work in the area of open-domain information extraction. The TRIPLEX pipeline is presented in Section 3. Section 4 describes our experiments, ending with a discussion on the obtained results. Finally, Section 5 concludes the paper, summarizing the main aspects and presenting possible directions for future work.

2 Related Work

OIE systems are used to extract triples from text and they can be classified into two major groups. The first group includes systems that consider verb-mediated triples (TextRunner [2], WOE [11], ReVerb [7], and OLLIE [9]). The second group includes systems that consider noun-mediated triples (OLLIE [9], ClauseIE [6], Xavier and Lima's system [13], and ReNoun [14]).

In the first group, the earliest proposed OIE system was TextRunner. This system first detects pairs of noun phrases; then it finds a sequence of words as a potential relation (i.e., predicate) between each pair of noun phrases. In a similar way, WOE uses a dependency parser to find the shortest dependency path between two noun phrases. All of the approaches in the first group assume that the object occurs after the subject.

OLLIE, which is a member of the second group, was the first approach to extract both noun-mediated and verb-mediated triples. It uses high confidence triples extracted by ReVerb as a bootstrapping set to learn patterns. These patterns, mostly based on dependency parse trees, indicate different ways of expressing triples in textual sources. It is important to note that OLLIE only extracts noun-mediated triples that can be expressed via verb-mediated formats. Therefore, it only covers a limited group of noun-mediated triples. In comparison, TRIPLEX only extracts triples from compound nouns, adjectives, and appositions. ClauseIE uses knowledge about English grammar to detect clauses based on the dependency parse trees of sentences [6]. Subsequently, triples are generated depending on the type of those clauses. ClauseIE has predefined rules to extract triples from dependency parse trees and it is able to generate both verb-mediated triples from clauses and noun-mediated triples from possessives and appositions. In contrast, TRIPLEX automatically learns rules that extract triples during its bootstrapping process.

Xavier and Lima use a boosting approach to expand the training set for information extractors so as to cover an increased variety of noun-mediated triples [13]. They find verb interpretations for noun and adjective based phrases, and transform these into verb-mediated triples to enrich the training set. Still, these verb interpretations can create long and ambiguous sentences. This makes filtering unrelated interpretations an essential step before adding the inferred verb interpretations to the training set of information extractors. TRIPLEX does not depend on verb patterns to extract noun-mediated triples, thereby making such filtering unnecessary.

Closest to our work is ReNoun, a system that uses an ontology of noun attributes and a manually crafted set of extraction rules, to extract seeds [14]. The seeds are then used to learn dependency parse patterns for extracting triples. In contrast, TRIPLEX uses data from Wikipedia, specifically infobox properties, that is, DBPedia properties extracted from infoboxes, and infobox values during its bootstrapping process, without requiring manual intervention.

3 TRIPLEX

OIE systems extract triples from an input sentence according to format `<subject; relation; object>`. In these triples, a relation phrase (i.e., a predicate or property) expresses a semantic relation between the subject and the object. The subject and the object are noun phrases and the relation phrase is a textual fragment that indicates a semantic relation between two noun phrases. The semantic relation can be verb-mediated or noun-mediated. For example, an extractor may find triples `<Kevin Systrom; profession; cofounder>` and `<Kevin Systrom; appears on; NBC News>` in the sentence *Instagram cofounder Kevin Systrom appears on NBC News*. The first triple is noun-mediated and the second one is verb-mediated.

The TRIPLEX approach focuses on noun-mediated triples from noun phrases, adjectives, and appositions. First, it finds sentences that express noun-mediated triples. These sentences are detected by using a dependency parser to find grammatical relations between nouns, adjectives, appositions, and conjunctions in sentences. Second, it automatically extracts templates from the sentences. Finally, these templates are used to extract noun-mediated triples from previously unseen text.

The TRIPLEX pipeline uses the Stanford NLP toolkit[3] to parse sentences, extract dependencies, label tokens with named entity (NE) and with part-of-speech (POS) information, and perform co-reference resolution.

The co-reference resolution module is used to replace in all the sentences pronouns and other co-referential mentions with the corresponding entity spans prior to subsequent processing. The dependency parser discovers the syntactic structure of input sentences. A dependency parse of a sentence is a directed graph whose vertices are words and whose edges are syntactic relations between the words. Each dependency corresponds to a binary grammatical relation between a governor and a dependent [5]. For example, the dependency relations `root<ROOT,went>`, `nsubj<went,Obama>`, and `prep-to<went,Denver>` can be found in the sentence *Obama went to Denver*. In `root<ROOT,went>`, `ROOT` is the governor and `went` is the dependent. The part-of-speech tagger assigns a morpho-syntactic class to each word, such as noun, verb, or adjective. The Named Entity Recognition (NER) model[4] labels sequences of words according to the pre-defined categories: *Person*, *Organization*, *Location*, and *Date*.

The other components of the pipeline are a noun phrase chunker, which complements the POS, NER, and dependency parsing modules from the Standard NLP toolkit, WordNet synsets, and Wikipedia synsets. The noun phrase chunker extracts noun phrases from sentences. WordNet is a lexical database that categorizes English words into sets of synonyms called synsets. WordNet synsets are used to recognize entities of a sentence according to the pre-defined categories, complementing the Stanford NER system. Several synsets are also built for each Wikipedia concept. There are different mentions for a Wikipedia page (e.g., redirects and alternative names) and also in the hypertext anchors that point to a Wikipedia page. For example, in the Wikipedia page for the *University of Illinois at Chicago*, the word *UIC* is extensively used to refer to the university. Synsets of Wikipedia pages are constructed automatically by using redirection page links, backward links, and hypertext anchors. These links are retrieved using the JWPL library.[5] TRIPLEX uses infobox properties and infobox values of Wikipedia during its bootstrapping process. We use a Wikipedia English dump[6] to extract all Wikipedia pages and query Freebase and DBPedia according to the Wikipedia page ID, to determine the type of the page. Wikipedia pages are categorized under the following types: *Person*, *Organization*, or *Location*. Additionally, we perform co-reference resolution on extracted Wikipedia pages to identify words that refer to the same Wikipedia page subject. We then use these words to enrich synsets of the respective Wikipedia page. We now de-

[3] http://nlp.stanford.edu/software/corenlp.shtml
[4] http://nlp.stanford.edu/software/CRF-NER.shtml
[5] https://code.google.com/p/jwpl/
[6] http://dumps.wikimedia.org/backup-index.html

scribe the TRIPLEX approach for extracting templates, starting with the generation of the bootstrapping set of sentences.

3.1 Bootstrapping Set Creation

Following ideas from OLLIE that leverage bootstrapping [9], our first goal is to construct automatically a bootstrapping set that expresses multiple ways in which information in noun phrases, adjectives, and appositions is encapsulated. The bootstrapping set is created by processing extracted Wikipedia pages and their corresponding infoboxes.

Extracted Wikipedia pages without infobox templates are ignored during sentence extraction, while the other pages are converted into sets of sentences. Finally, we perform preprocessing on sentences from extracted Wikipedia pages and use custom templates (i.e., regular expressions) to identify infobox values from the text. We also convert dates to strings. For instance, the infobox with value 1961|8|4 is translated to August 4, 1961. We begin template extraction by processing 3,061,956 sentences in extracted Wikipedia pages that are matched with infobox values.

The sentence extractor automatically constructs a bootstrapping set by matching infobox values of the extracted Wikipedia pages with sentences in the corresponding Wikipedia pages. The sentence extractor searches for sentences that are matched with infobox values in a given Wikipedia page. If in a sentence there exists a dependency path between the current infobox value and the synset of the page name, and if this dependency path only contains nouns, adjectives, and appositions, then the sentence is extracted. For instance, given the page for *Barack Obama*, the extractor matches the infobox value *August 4, 1961* with the sentence *Barack Hussein Obama II (born August 4, 1961)*. This process is repeated for all infobox values of a Wikipedia page. In order to match complete names with abbreviations such as *UIC*, the extractor uses a set of heuristics that was originally proposed in WOE [11], named *full match*, *synonym set match*, and *partial match*. Full match is performed when the page name is found within a sentence of the page. Synonym set match occurs when one member of the synonym set for the page name is discovered within a sentence. Partial match is performed when a prefix or suffix of a member of the synonym set is used in a sentence. Finally, a template is created by marking an infobox value and a synset member in the dependency path of a selected sentence. We apply a constraint on the length of the dependency path between a synset member and an infobox value to reduce bootstrapping errors. This constraint sets the maximum length of the dependency path to 6, which was determined experimentally by checking the quality of our bootstrapping set. To check its quality, we randomly selected 100 sentences for manual examination and found that approximately 90% of the extracted sentences satisfied the dependency path length.

After creating the bootstrapping set, the next step is to automatically create templates from dependency paths that express noun-mediated triples. Templates describe how noun-mediated triples can occur in textual sentences. Each template results from a dependency path between a synset member (a subject) and an infobox value (an object). We annotate these paths with POS tags, named entities, and WordNet synsets. In the template, to each infobox value we add the name of the infobox. In addition, a template includes a template type, based on the types of the Wikipedia page where the sentence occurred. The types of dependencies between synset members and infobox values are also attached to the template. If there is a copular verb or a verbal modi-

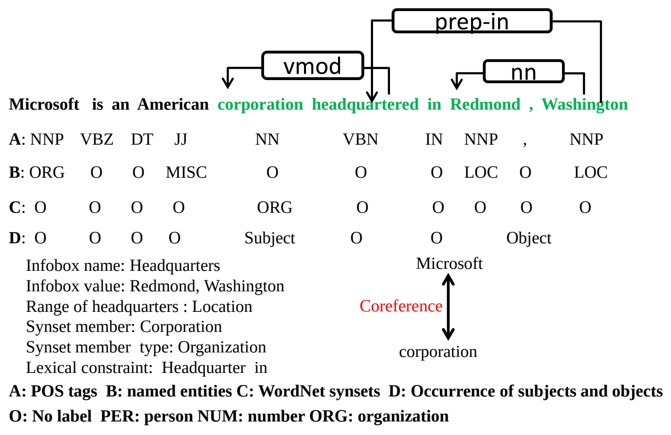
prep-in
vmod
nn
Microsoft is an American corporation headquartered in Redmond , Washington
A: NNP VBZ DT JJ NN VBN IN NNP , NNP
B: ORG O O MISC O O O LOC O LOC
C: O O O O ORG O O O O O
D: O O O O Subject O O Object
Infobox name: Headquarters
Infobox value: Redmond, Washington
Range of headquarters : Location
Synset member: Corporation
Synset member type: Organization
Lexical constraint: Headquarter in
Microsoft
Coreference
corporation
A: POS tags B: named entities C: WordNet synsets D: Occurrence of subjects and objects
O: No label PER: person NUM: number ORG: organization

Fig. 1. An example sentence annotated with the corresponding dependency parse, related POS tags, named entities, WordNet synsets, and occurrence of the synset member of the Wikipedia page (subjects) and the infobox values (objects).

fier in the dependency path, we will add them as a lexical constraint to the template. For example, *headquartered* is a verbal modifier added as a lexical constraint to the corresponding template for the sentence: *Microsoft is an American corporation headquartered in Redmond* (see Figure 1). *Born* is another lexical constraint for templates related to nationality, as in the sentence *The Italian-born Antonio Verrio was frequently commissioned*. We merge templates if exists the only differences among them relate to lexical constraints. We keep one template and a list of lexical constraints for the merged templates. Finally, we process all templates and remove redundant ones.

Infobox values may occur before or after synset members of the page name in sentences. If there exists a dependency path between these values without considering their position, the related template is extracted. For example, the infobox value occurs before the synset member in the sentence *Instagram co-founder Kevin Systrom announced a hiring spree.* In this example, *co-founder* is the infobox value and *Steve Hafner* is the synset member of the Wikipedia page. The infobox value may also occur after the synset member, as shown in the sentence *Microsoft is an American corporation headquartered in Redmond*. In this case, *corporation* is the synset member and *Redmond* is the infobox value (see Figure 1). We also considered some additional heuristics in the template generation process. When there is a conjunction dependency between nouns in a sentence, if one of the nouns has a specific template, the template will be expanded to include all the noun relations joined by conjunctions in the sentence.

The noun phrase chunker is finally used to search dependency paths and merge words that are part of the same noun phrase chunk. In addition, we do not apply the noun phrase chunker if a synset member and the infobox value occur in the same chunk.

3.2 Template Matching

This section describes how we use the dependency paths of a sentence together with the extracted templates to detect noun-mediated triples. First, named entities and WordNet synsets are used to recognize the candidate subjects of a sentence together with their types. Then, dependency paths between candidate subjects and all potential objects are identified and annotated by the NLP pipeline. Finally, candidate infobox names (which are properties in DBPedia) are assigned to a candidate subject and a candidate object, derived from matching templates with subject types, dependency types, WordNet synsets, POS tags, and named entity annotations. If there are lexical constraints in a template, the words in the dependency path between a subject and an object must be matched with one of the phrases in the lexical constraint list. We also consider the semantic similarity between the the words and the member of lexical constraint list, using Jiang and Conrath's approach to calculate the semantic similarity between words [8].

When there is a specific range (*Person*, *Organization*, *Location*, or *Date*) for an infobox name (property) of a triple and when the object type of a triple is unknown, a previously trained confidence function is used to decrease the confidence score of the triple. A logistic regression classifier is used in this confidence function after it is trained using 500 triples extracted from Wikipedia pages. Our confidence function is an extension of the confidence function proposed for OLLIE [9] and for ReVerb [7]. A set of features (i.e., frequency of the extraction template, existence of lexical words in templates, range of properties, and semantic object type) are computed for each extracted triple, and the classifier is used to predict the confidence score for the extracted triples.

Finally, each candidate triple has an infobox name that is mapped to a DBPedia property and the object type of a candidate triple should be matched with the range of a property in DBPedia. When the range of a property is a literal, all possible values of the property are retrieved from DBPedia and compared with the candidate object. If their values are not matched, the candidate triple is discarded.

4 Evaluation

We conducted a comprehensive set of experiments to compare the outputs of TRIPLEX, OLLIE, and ReVerb based upon the approach suggested by Bronzi et al. [4]. These authors introduced an approach to evaluate verb-mediated information extractors automatically. We improve on their approach by expanding it to the evaluation of noun-mediated triples. Additionally, we compare TRIPLEX, OLLIE, and ReVerb using a manually constructed gold standard. Finally, we compare information extractors according to the quality of their extracted triples.

We first created a dataset by taking 1000 random sentences from Wikipedia that have not been used during the bootstrapping process. Each sentence in the test dataset has a corresponding Wikipedia page ID. All extracted facts gathered by information extractors from these sentences needed to be verified. A fact is a triple `<subject; predicate; object>` that indicates a relation between a subject and an object. A fact is correct if its corresponding triple has been found in the Freebase or DBPedia knowledge bases or if there is a significant association between the entities (subjects and objects) and the predicate on the web [4]. In order to estimate the precision of an information extractor [4], we use the following formula:

$$Precision = \frac{|a| + |b|}{|S|} \tag{1}$$

In Equation 1, $|b|$ is the number of extracted facts from Freebase and DBPedia. $|S|$ is the total number of extracted facts by the system. Finally, $|a|$ indicates the number of correct facts returned by the information extractor, which have been validated by using pointwise mutual information (PMI) as defined in Equation 2 from occurrences in the web. Since values of properties in Freebase and DBPedia are not completely filled, Bronzi et al. [4] suggest computing PMI to verify a fact. The likelihood of observing a fact is computed based on its subject ($subj$), object (obj) and predicate ($pred$):

$$\mathrm{PMI}(subj, pred, obj) = \frac{\mathrm{Count}(subj\ \wedge\ pred\ \wedge\ obj)}{\mathrm{Count}(subj\ \wedge\ obj)} \tag{2}$$

When verifying the extracted facts, we use the corresponding Wikipedia ID of each sentence to retrieve all possible properties and their values from Freebase or DBPedia. These values will then be used to verify extracted facts from sentences. The semantic similarity between the properties of those knowledge bases and the predicate of a fact are calculated [8]. This similarity measure uses WordNet together with corpus statistics to calculate semantic similarity between words. If the semantic similarity is above a predetermined threshold and both entities are also matched together, the fact is deemed correct [4].

The function $Count(q)$ indicates the number of documents that contain the query elements in its proximity. The proximity value shows the the maximum number of words between the query elements. The proximity value is at most 4 in our experiments. We estimate query q by using the Google search engine. The range of the PMI function is between 0 and 1. The higher the PMI value, the more likely that the fact is correct. In particular, a fact is deemed correct if its PMI value is above a given threshold. The value for this threshold, determined experimentally, is 10^{-3} in our experiments. We also use the method in Equation 3 to estimate recall, as suggested by Bronzi at al.:

$$Recall = \frac{|a| + |b|}{|a| + |b| + |c| + |d|} \tag{3}$$

The parameters $|a|$ and $|b|$ are computed as we mentioned in Equation 1. We now describe how we estimate $|c|$ and $|d|$. First, all correct facts within sentences should be identified. Each fact contains two entities and a relation. All possible entities of a sentence are detected by the Stanford named entity recognizer and WordNet synsets. Furthermore, we use the Stanford CoreNLP toolkit to detect all verbs (predicates) in a sentence. Finally, we expand the set of predicates from sentences by adding DBPedia and Freebase properties.

We use three sets S, P, and O to create all the possible facts, which are respectively the set of recognized subjects, predicates, and objects in the sentences. All possible facts are produced by the Cartesian product of these three sets, $G = (S \times P \times O)$. Then, $|c|$ in Equation 3 is estimated by subtracting $|b|$ from the intersection of Freebase and DBPedia with G. D denotes the set of all the facts in Freebase and in DBPedia. Then, $G \setminus D$ is computed by applying PMI to the facts that are not in D . Finally, $|d|$ is determined by subtracting $|a|$ from $G \setminus D$.

We further select 50 sentences from the dataset of 1000 sentences, and a human judge (H) extracts all of the correct facts. Then, we use the suggested method by Bronzi et al. [4], to compute the agreement between the automatic evaluation and manual evaluation. The agreement between the automatic evaluation and human evaluation is found

	Automatic evaluation			**Manual evaluation**		
	Prec	Rec	F1	Prec	Rec	F1
ReVerb	0.61	0.15	0.24	0.55	0.11	0.18
OLLIE	0.64	0.30	0.40	0.65	0.32	0.42
OLLIE*	0.62	0.10	0.17	0.63	0.11	0.18
TRIPLEX	0.55	0.17	0.25	0.6 2	0.22	0.32
TRIPLEX + OLLIE	0.57	0.40	0.47	0.63	0.44	0.51
TRIPLEX + ReVerb	0.58	0.32	0.41	0.55	0.35	0.42

Table 1. Automatic and manual evaluation of information extractors. OLLIE* only generates noun-mediated triples. The confidence scores of all extracted triples are above 0.2.

to be 0.71. Now, we are able to determine the precision and recall for information extractors, based upon the automatic and manual evaluations.

We ran OLLIE, ReVerb, and TRIPLEX both individually and in combination and then computed precision and recall. Table 1 shows results for the automatic and manual evaluation of information extractors.

ReVerb only generates verb-mediated triples and OLLIE extracts verb-mediated triples and also noun-mediated triples, if they are expressed in verb-mediated styles. TRIPLEX generates noun-mediated triples and it can complement the results of OLLIE and ReVerb. OLLIE, ReVerb, and TRIPLEX all assign a confidence score to each extracted triple. In these experiments, the extracted triples are only considered if their confidence scores are above a threshold of 0.2. TRIPLEX shows an improvement in Table 1 in comparison with the automatic evaluation because extracted facts with very low PMI are considered as false in automatic evaluation. However, these facts were often evaluated as true positive by the human judge. We analyzed the errors made by TRIPLEX in the gold standard dataset that was manually annotated. TRIPLEX's errors can be classified into two groups: errors in precision (false positive) and errors in recall (false negative). In the gold standard 65% of the triples are related to verb-mediated triples, which are not considered by TRIPLEX.

Table 2 shows triples in the gold standard that are not extracted by TRIPLEX. Of those, 10% obtained low confidence scores (false negatives) because the NER module and WordNet could not find the semantic type for the objects. We penalize the confidence score of a candidate triple if its predicate has one particular property type and if no type is detected for the triple's object. For example, the range of the *nationality* property in DBPedia is a *Location* constraint but neither the NER module nor WordNet can recognize a type in the phrase *Swedish writer* or *Polish-American scientist*. Also, 12% of errors are related to the dependency parser because the parser could not detect a correct grammatical relation between the words in a sentence. Another 7% of the errors occurred when the coreferencing module did not properly resolve coreferential expressions during template extraction. This problem is alleviated by assigning low confidence scores to this group of templates. Finally, 6% of errors were caused by overgeneralized templates. During template generalization, POS tags where substituted by universal POS tags. Since some templates only extract triples for proper nouns, nouns or

	Missed extractions
10%	No semantic types
12%	Dependency parser problems
7%	Coreferencing errors
6%	Over generalized templates
65%	Verb-mediated triples (outside the of scope for TRIPLEX)

Table 2. Percentage of the gold standard triples missed by TRIPLEX.

	Distribution	**Triple category**
Noun-mediated	12%	Conjunctions, adjectives and noun phrases
	9%	Apposition and parenthetical phrases
	6%	Titles or professions
	8%	Templates with lexicon
Verb-mediated	65%	Verb-mediated triples

Table 3. Distribution of correct extracted triples for TRIPLEX + OLLIE based on their categories. The confidence score of extracted triples by TRIPLEX and OLLIE are above 0.2.

personal pronouns, generalizing and merging these templates together did not produce correct triples.

We also saw that 20% of the errors (false positive) are correct triples. Subjects for these triples were not popular on the web, therefore there were few documents about these subjects. Thus, applying the same PMI threshold as that used for prominent subjects proved to be ineffective. For example, triples extracted by TRIPLEX are judged as incorrect in the sentence, *Alexey Arkhipovich Leonov (born 30 May 1934 in Listvyanka, Kemerovo Oblast, Soviet Union) is a retired Soviet/Russian cosmonaut*. These triples include information about birth date, birth place, origin, and profession, but are not available in the gold standard. Many other false positive errors were due to dependency parser problems, named entity problems, chunker problems, and over generalized templates. OIE systems such as ReVerb and OLLIE usually fail to extract triples from compound nouns, adjectives, conjunctions, reduced clauses, parenthetical phrases, and appositions. TRIPLEX only covers noun-mediated triples in sentences and we can examine TRIPLEX's output with respect to the gold standard, as shown in the Table 3. The table shows that 12% of noun-mediated triples are related to conjunctions, adjectives, and noun phrases, meaning that TRIPLEX is also able to extract noun-mediated triples from noun conjunctions. For example, TRIPLEX extracts triples about Rye Barcott's professions from the sentences *Rye Barcott is author of It Happened on the Way to War. He is a former U.S. Marine and cofounder of Carolina for Kibera*. Moreover, TRIPLEX is able to extract triples from appositions and parenthetical phrases, containing 9% of extracted triples within this category. For example, extracted triples from the following sentence indicate Michelle Obama's two professions, birth date, and nationality; *Michelle LaVaughn Robinson Obama (born January 17, 1964), an American lawyer and writer, is the wife of the current president of the United States*. We see that 6%

of triples are related to titles or professions, such as *Sir Herbert Lethington Maitland*, *Film director Ingmar Bergman*, and *Microsoft co-founder Bill Gates*. OLLIE is similarly able to capture this kind of triples because they are expressed in a verb-mediated style. However, TRIPLEX does so without using a verb-mediated format. The final fraction of 8% is for noun-mediated triples that rely on the lexicon of noun-mediated templates. For example, the *headquarters* of Microsoft is extracted from the sentence, *Microsoft is an American multinational corporation headquartered in Redmond, Washington*. Finally, 65% of the extracted triples are verb-mediated triples. Both ReVerb and OLLIE generate verb-mediated triples from sentences. The majority of errors produced by OLLIE and ReVerb are due to incorrectly identifying subjects or objects. ReVerb first locates verbs in a sentence and then looks for noun phrases to the left and right of the verbs. ReVerb's heuristics sometimes fail to find correct subjects and objects because of compound nouns, appositions, reduced clauses, or conjunctions. OLLIE relies on extracted triples from ReVerb for its bootstrapping process and learning patterns. Therefore, OLLIE's patterns are limited to verb-meditated triples. However, OLLIE also produces noun-meditated triples if they can be expressed via verb-mediated formats. Thus, OLLIE's coverage of noun-mediated triples is limited.

We also analyzed some sentences to figure out why different information extractors were not able to produce all of the triples in the gold standard. The first reason is that there may not be sufficient information in a sentence to extract triples. For example, TRIPLEX can find the triple `<Antonio; nationality; Italian>` but it would not find the triple `<Antonio; nationality; England>` in the sentence *The Italian-born Antonio Verrio was responsible for introducing Baroque mural painting into England.* Second, OLLIE and ReVerb cannot successfully extract verb-mediated triples from sentences that contain compound nouns, appositions, parentheses, conjunctions, or reduced clauses. When OLLIE and ReVerb cannot yield verb-mediated triples, recall will be affected because verb-mediated triples are outside of the scope of TRIPLEX. Improvements to OLLIE and ReVerb could substantially lead to better precision and recall. Finally, improvements on the different NLP components can also lead to better precision and recall for information extractors that rely heavily on them.

5 Conclusions and Future Work

This paper presented TRIPLEX, an information extractor to generate triples from noun phrases, adjectives, and appositions. First, a bootstrapping set is automatically constructed from infoboxes in Wikipedia pages. Then, templates with semantic, syntactic, and lexical constraints are constructed automatically to capture triples. Our experiments found that TRIPLEX complements the output of verb-mediated information extractors by capturing more noun-mediated triples. The extracted triples can for instance be used to populate Wikipedia pages with missing infobox attribute values or to assist authors in the task of annotating Wikipedia pages. We also improved an automated method to evaluate triples by expanding it to include noun-mediated triples.

In future work, we plan to improve results for triples involving numerical values with different units (i.e., square meter, meter) in generating extraction templates. We would also like to enrich the bootstrapping set process by using a probabilistic knowledge base (e.g., Probase [12]), as it may broaden the coverage of the bootstrapping set and support the construction of more templates.

Acknowledgments
We would like to thank Matteo Palmonari for useful discussions. Cruz and Mirrezaei were partially supported by NSF Awards CCF-1331800, IIS-1213013, and IIS-1143926. Cruz was also supported by a Great Cities Institute scholarship. Martins was supported by the Portuguese FCT through the project grants EXCL/EEI-ESS/0257/2012 (DataStorm Research Line of Excellence) and PEst-OE/EEI/LA0021/2013 (INESC-ID's Associate Laboratory multi-annual funding).

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: A Nucleus for a Web of Open Data. In: International Semantic Web Conference (ISWC). pp. 722–735 (2007)
2. Banko, M., Cafarella, M.J., Soderland, S., Broadhead, M., Etzioni, O.: Open Information Extraction for the Web. In: International Joint Conferences on Artificial Intelligence (IJCAI). pp. 2670–2676 (2007)
3. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In: ACM SIGMOD International Conference on Management of Data. pp. 1247–1250. ACM (2008)
4. Bronzi, M., Guo, Z., Mesquita, F., Barbosa, D., Merialdo, P.: Automatic Evaluation of Relation Extraction Systems on Large-scale. In: Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction. pp. 19–24 (2012)
5. De Marneffe, M.C., Manning, C.D.: Stanford Typed Dependencies Manual (2008), `http://nlp.stanford.edu/software/dependencies_manual.pdf`
6. Del Corro, L., Gemulla, R.: ClausIE: Clause-based Open Information Extraction. In: International World Wide Web Conference (WWW). pp. 355–366 (2013)
7. Fader, A., Soderland, S., Etzioni, O.: Identifying Relations for Open Information Extraction. In: Conference on Empirical Methods in Natural Language Processing. pp. 1535–1545 (2011)
8. Jiang, J., Conrath, D.: Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In: International Conference on Research in Computational Linguistics. pp. 19–33 (1997)
9. Mausam, Schmitz, M., Bart, R., Soderland, S., Etzioni, O.: Open Language Learning for Information Extraction. In: Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. pp. 523–534 (2012)
10. Singhal, A.: Introducing the Knowledge Graph: Things, Not Strings. Official Google Blog, May (2012)
11. Wu, F., Weld, D.S.: Open Information Extraction Using Wikipedia. In: Annual Meeting of the Association for Computational Linguistics. pp. 118–127 (2010)
12. Wu, W., Li, H., Wang, H., Zhu, K.Q.: Probase: A Probabilistic Taxonomy for Text Understanding. In: ACM SIGMOD International Conference on Management of Data. pp. 481–492 (2012)
13. Xavier, C., Lima, V.: Boosting Open Information Extraction with Noun-Based Relations. In: International Conference on Language Resources and Evaluation. pp. 96–100 (2014)
14. Yahya, M., Whang, S.E., Gupta, R., Halevy, A.: ReNoun: Fact Extraction for Nominal Attributes. In: International Conference on Empirical Methods in Natural Language Processing. pp. 325–335 (2014)