Open Information Extraction with Tree Kernels

Ying Xu, Mi-Young Kim, Kevin Quinn, Randy Goebel and **Denilson Barbosa**
Department of Computing Science
University of Alberta
{yx2,miyoung2,kfjquinn,goebel,denilson}@cs.ualberta.ca

Abstract

Traditional relation extraction seeks to identify pre-specified semantic relations within natural language text, while open Information Extraction (Open IE) takes a more general approach, and looks for a variety of relations without restriction to a fixed relation set. With this generalization comes the question, what is a relation? For example, should the more general task be restricted to relations mediated by verbs, nouns, or both? To help answer this question, we propose two levels of subtasks for Open IE. One task is to determine if a sentence potentially contains a relation between two entities? The other task looks to confirm explicit relation words for two entities. We propose multiple SVM models with dependency tree kernels for both tasks. For explicit relation extraction, our system can extract both noun and verb relations. Our results on three datasets show that our system is superior when compared to state-of-the-art systems like REVERB and OLLIE for both tasks. For example, in some experiments our system achieves 33% improvement on nominal relation extraction over OLLIE. In addition we propose an unsupervised rule-based approach which can serve as a strong baseline for Open IE systems.

1 Introduction

Relation Extraction (RE) systems are designed to discover various semantic relations (e.g. <Obama, president, the United States>) from natural language text. Traditional RE systems extract specific relations for prespecified name-entity types (Bunescu and Mooney, 2005; Chan and Dan, 2011; Zhou and Zhu, 2011). To train such systems, every relation needs manually annotated training examples, which supports limited scope and is difficult to extend. For this reason, Banko et al. (2007) proposed Open Information Extraction (Open IE), whose goal is to extract general relations for two entities. The idea is to avoid the need for specific training examples, and to extract a diverse range of relations. This generalized form has received significant attention, e.g., (Banko et al., 2007; Akbik, 2009; Wu and Weld, 2010; Fader et al., 2011; Mausam et al., 2012).

Because Open IE is not guided by or not restricted to a prespecified list of relations, the immediate challenge is determining about what counts as a relation? Most recent Open IE systems have targeted verbal relations (Banko et al., 2007; Mausam et al., 2012), claiming that these are the majority. However, Chan and Dan (2011) show that only 20% of relations in the ACE programs Relation Detection and Characterization (RDC) are verbal. Our manually extracted relation triple set from the Penn Treebank shows that there are more nominal relations than verbal ones, 3 to 2. This difference arises because of the ambiguity of what constitutes a relation in Open IE. It is often difficult even for humans to agree on what constitutes a relation, and which words in the sentence establish a relation between a pair of entities. For example, in the sentence "Olivetti broke Cocom rules" is there a relation between *Olivetti* and *Cocom*? This ambiguity in the problem definition leads to significant challenges and confusion when evaluating and comparing the performance of different methods and systems. An example are the results in Fader et al. (2011) and Mausam et al. (2012). In Fader et al. (2011), REVERB "is reported" as su-

perior to WOE^{parse}, a system proposed in Wu and Weld (2010); while in Mausam et al. (2012), it is reported the opposite.

To better answer the question, what counts as a relation? we propose two tasks for Open IE. The first task seeks to determine whether there is a relation between two entities (called "Binary task"). The other is to confirm whether the relation words extracted for the two entities are appropriate (the "Triple task"). The Binary task does not restrict relation word forms, whether they are mediated by nouns, verbs, prepositions, or even implicit relations. The Triple task requires an abstract representation of relation word forms, which we develop here. We assume that relation words are nouns or verbs; in our data, these two types comprise 71% of explicit relations.

We adapt an SVM dependency tree kernel model (Moschitti, 2006) for both tasks. The input to our tasks is a dependency parse, created by Stanford Parser. Selecting relevant features from a parse tree for semantic tasks is difficult. SVM tree kernels avoid extracting explicit features from parse trees by calculating the inner product of the two trees. For the Binary task, our dependency path is the path between two entities. For the Triple task, the path is among entities and relation words (i.e. relation triples). Tree kernels have been used in traditional RE and have helped achieve state of the art performance (Culotta and Sorensen, 2004; Bunescu and Mooney, 2005; Wang, 2008; Nguyen et al., 2009; Zhou and Zhu, 2011). But one challenge of using tree kernels on Open IE is that the lexicon of relations is much larger than those of traditional RE, making it difficult to include the lexical information as features. Here we proposed an unlexicalized tree structure for Open IE. As far as we know, this is the first time an SVM tree kernel has been applied in Open IE. Experimental results on multiple datasets show our system outperforms state-of-the-art systems REVERB and OLLIE. Typically an Open IE system is tested on one dataset. However, because the definition of relation is ambiguous, we believe that is necessary to test with multiple datasets.

In addition to the supervised model, we also propose an unsupervised model which relies on several heuristic rules. Results with this approach show that this simple unsupervised model provides a robust strong baseline for other approaches.

In summary, our main contributions are:

- Use SVM tree kernels for Open IE. Our system is robust comparing with other Open IE systems, achieving superior scores in two test sets and comparative scores in another set.
- Extend beyond verbal relations, which are prevalent in current systems. Analyze implicit relation problem in Open IE, which is ignored by other work.
- Propose an unsupervised model for Open IE, which can be a strong baseline for other approaches.

The rest of this paper is organized as follows. Section 2 provides the problem description and system structure, before summarizing previous work in Section 3. Section 4 defines our representation of relation word patterns crucial to our task two, and Section 5 describes tree kernels for SVM. Section 6 describes the unsupervised model, and Section 7 explains our experiment design and results. Section 8 concludes with a summary, and anticipation of future work.

2 Problem Definition and System Structure

The common definition of the Open IE task is a function from a sentence, s, to a set of triples, $\{< E1, R, E2 >\}$, where $E1$ and $E2$ are entities (noun phrases) and R is a textual fragment indicating a semantic relation between the two entities. Our "Triple task" is within this definition. However it is often difficult to determine which textual fragments to extract. In addition, semantic relations can be implicit, e.g., consider the *located in* relation in the sentence fragment "Washington, US." To illustrate how much information is lost when restricting the relation forms, we add another task (the "Binary task"), determining if there is a relation between the two entities. It is a function from s, to a set of binary relations over entities, $\{< E1, E2 >\}$. This binary task is designed to overcome the disadvantage of current Open IE systems, which suffer because of restricting the relation form, e.g., to only verbs, or only nouns. The two tasks are independent to each other.

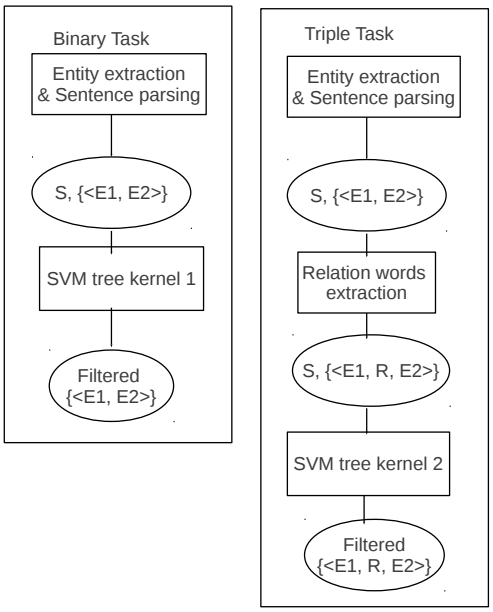


Figure 1: Our Open IE system structure.

Figure 1 presents our Open IE system structure. Both tasks need pre-processing with the Stanford NLP tools [1]. Entities and pairs within a certain distance are extracted[2], and sentences are parsed. We employ the typed collapsed dependency parse (De Marneffe et al., 2006), which is computed from the constituent parsing and has proved to be useful for semantic tasks (MacCartney et al., 2006). For the Binary task, an SVM model is employed to filter out the extracted entity pair candidates, and output pairs which have certain relations. For the Triple task, we identify relation word candidates of the pairs, based on regular expression patterns. Then another SVM model is employed to decide if the relation triples are correct or not.

3 Related Work

In traditional relation extraction, SVM tree kernel models are the basis for the current state of the art (Culotta and Sorensen, 2004; Bunescu and Mooney, 2005; Wang, 2008; Nguyen et al., 2009; Zhou and Zhu, 2011). But there is more recent work on Open IE (Banko et al., 2007; Akbik, 2009; Wu and Weld, 2010; Christensen et al., 2011; Fader et al., 2011; Mausam et al., 2012).

[1]Other equivalent tools such as Open NLP could be used.

[2]Here distance means number of tokens in between

Pattern = *RelW* E1 *RelW* E2 *RelW*
st. at least one *RelW* has *N* or *V*

RelW= *V* | *V P* | *V W* P* | *N* | *N P* | *P*
V = verb particle? adv?
W = (noun | adj | adv | pron | det)
P = (prep | particle | inf.marker)
N = (adj | noun) * noun

Figure 2: Relation Pattern Form (*RelW* represents *relation words*, E1 and E2 are two entities.)

Fader et al. (2011) have developed REVERB, which solves the problem of incoherent extractions and uninformative extractions of two previous systems. Instead of extracting entities first, they extract verbal relation sequences based on a set of POS patterns. Then entities are identified around the relation sequence, so their system only extracts relation tokens between two entities tokens, e.g. relations such as <he, live in, city> in "Living in this city, he loves the city." are ignored. Finally, relation triple candidate noise is filtered by a supervised model which is based on lexical and POS features.

Mausam et al. (2012) present an improved system called OLLIE, which relaxes the previous systems' constraints that relation words are mediated by verbs, or relation words that appear between two entities. OLLIE creates a training set which includes millions of relations extracted by REVERB with high confidence. Then OLLIE learns relation patterns composed of dependency path and lexicon information. Relations matching the patterns can then be extracted.

Both REVERB and OLLIE output a confidence value for every relation triples, instead of classifying them as true or false.

4 Relation Candidate Extraction

For the Triple task, we extract textual fragments which matches certain POS patterns in an entity pair's context as relation candidates for that pair. In our experiments, the fragments are n-grams with $n < 5$ and between the pairs or in a window size of 10 before the first entity or after the second entity, which is experimentally a good choice to minimize noise while attaining maximum number of relations.

Our representation of POS regular expression pat-

tern sets expands that of Fader et al. (2011). The patterns are composed of verb and noun phrases (see Figure 2). A relation candidate can consist of words before, between, or after the pair, or the combination of two consecutive positions. Instead of extracting only verbal relations (e.g. *give birth to*), our patterns also extract relations specified through noun phrases. In the sentence "Obama, the president of the United States, made a speech" the relation "president" matches the relational form "RelW=N, N=noun". Our method can also extract relation words interspersed between the two entities: e.g., ORG *has* NUM *employees*, which matches the pattern "E1 RelW E2 RelW"; the first RelW matches V, with V=verb, and the second RelW matches N, with N=noun. We choose not to use the dependency path for relation word extraction because of the reason mentioned in (Fader et al., 2011). The dependency method will create incoherent relations. For example, in the sentence "They recalled that Nungesser began his career as a precinct leader." *recall began* will be extracted as a relation because the two words are linked. Although this pattern based method has limitations, finding further improvements remains future work.

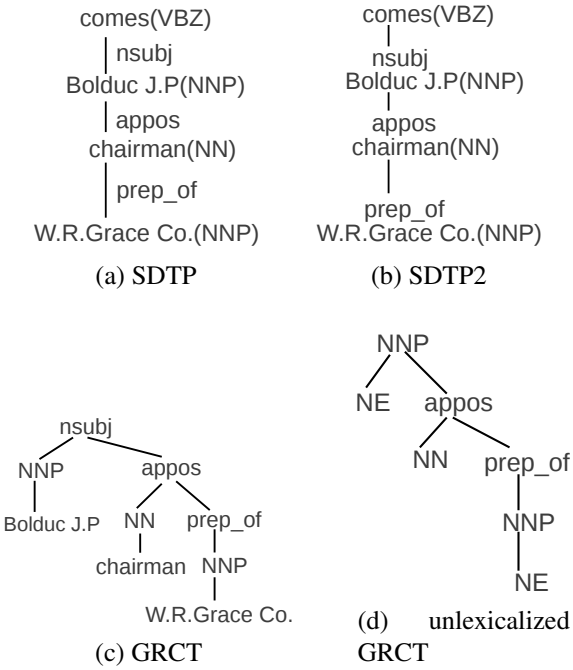


Figure 3: Example trees for shortest dependency path between *J.P. Bolduc* and *W.R.Grace Co.* in sentence "J.P. Bolduc, vice chairman of W.R.Grace Co., comes here." Figure (a) is the shortest dependency tree path (SDTP), (b) is the collapsed form, (c) is the GRCT, (d) is an unlexicalized GRCT with "NE".

5 Tree Kernels

Many methods recognize the value of leveraging parsing information in support of semantic tasks. But selecting relevant features from a parse tree is a difficult task. With kernel-based SVMs, both learning and classification relies on the inner-product between instances. SVM tree kernels avoid extracting explicit features from parse trees by calculating the inner product of the two trees, so the tree kernel value depends on the common substructure of two trees. A tree kernel function over Tree T_1 and T_1 is

$K(T_1, T_2) = \sum_{n1 \in N_{T_1}} \sum_{n2 \in N_{T_2}} \Delta(n_1, n_2)$,

N_{T_1} and N_{T_2} are the set of trees' nodes (Collins and Duffy, 2001). The Δ function provides the basis for identifying subtrees of nodes, which is the essential distinction between different tree kernel functions. Here we adapt the partial tree kernel (PTK) proposed by Moschitti (2006)[3], which can be used with both constituent and dependency parse trees. The computation of Δ function of PTK is

$$(\sum_{J_1, J_2, l(J_1)=l(J_2)} \lambda^{d(J_1)+d(J_2)} \prod_{i=1}^{l(J_1)} \Delta(c_{n_1}(J_{1i}), c_{n_2}(J_{2i})) + \lambda^2)\mu \quad (1)$$

when the node labels of n_1 and n_2 are the same, $\Delta = 0$ when they are different. c_{n_1} and c_{n_2} are child sequences of nodes n_1 and n_2 respectively, $J_1 =< J_{11}, J_{12}, J_{13}... >$ and $J_2 =< J_{21}, J_{22}, J_{23}... >$ are index sequences of the two child sequences, J_{1i} and J_{2i} are the i-th children of the two sequences. $l()$ means the sequence length, $d(J_1) = J_{1l(J_1)} - J_{11}$ and $d(J_2) = J_{2l(J_2)} - J_{21}$. μ and λ are two decay factors for the height of the tree and the length of the child sequences respectively, which we choose the default setting in the experiments. For a more detailed description of PTK, please refer to (Moschitti, 2006).

Now we present our unlexicalized dependency

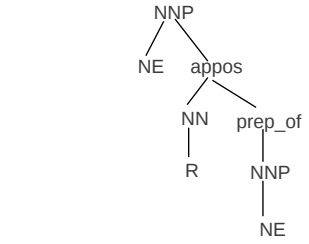
[3]Thanks to Prof. Moschitti for his PTK package.

tree structures for the tree kernel. One question arising in the conversion dependency structures (e.g., Figure 3a) for the tree kernel is how should we add POS tags and dependency link labels? The kernel cannot process labels on the arcs; they must be associated with tree nodes. Our conversion is similar to the idea of a Grammatical Relation Centered Tree (GRCT) of Croce et al. (2011). First we order the nodes of dependency trees so that the *dominant*, i.e. the parent of the dependency link is on the top, the *dependent*, i.e. the child at the bottom. At this stage, the link label is with the corresponding *dependent* POS-tag and the word (Figure 3b). If a dominant has more than one child, the children will be ordered according to their position in the sentence, from left to right. Next, every node is expanded such that the *dependent* POS-tags are the children of the link labels and parent of their words. For example, in Figure 3c, *NN* is the child of *appos*, parent of *chairman*. It is on the left of *prep_of* because *chairman* is on the left of *W.R.Grace Co.* in the sentence. As customary in Open IE, we do not add content words, while function words are optional. The unlexicalized GRCT is shown in Figure 3d. Note that for the root node, the link label is replaced by the POS-tag of the fist node in the path.

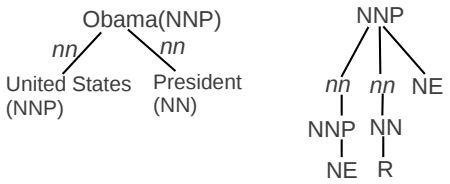
Recall that we have two tasks: detecting whether there is a relation between two entities (the Binary task), and whether the relation triple <*E1, relation, E2*> is correct (the Triplet task). We define two expanded versions of unlexicalized GRCT for the two tasks. The two versions contain different fragments of a dependency tree of a sentence.

For the Binary task, the shortest path between two entities' heads[4] is extracted and represented as a GRCT. The root node is the POS-tag of the fist node in the path. "NE" is used to represent the position of two entities while relation words are not specified. Figure 3d shows the example final outcome of our tree structure. It is used to decide if there is a relation between the entities *Bolduc J.P.* and *W.R.Grace Co.*

For the Triple task, we first extract relation words based on regular expression patterns as indicated in Section 4. If any relation word is between the shortest path of the two entities, the path is chosen as the input for SVM. Otherwise, two shortest paths between two entities and relation words will be extracted separately. The shortest one will be attached to the path between two entities. In our representation, relation words are tagged by having "R" as the child. Figure 4a shows the path form of the previous example. Figure 4b shows another example where "R" is not in the shortest path of the pair. The triple is <United States, president, Obama> for the sentence "United States President Barack Obama says so." The figure on the left is the dependency path. The figure on the right is the final tree for the triple task. The root is the POS-tag for *Obama*.



(a) Example 1.



(b) Example 2.

Figure 4: Tree structure with "R" added. Figure (a) is the example 1, which has R in the SDTP of the entity pair. Figure (b) is the example 2, with R not in the SDTP of the entity pair.

For the Triple task we combine the tree kernel with a polynomial kernel (Moschitti, 2005) applied to a feature vector. The feature set is in Table 1. F3 tries to preserve the semantic link between two discontinuous relation word segments. F6 constrains relation words to include only necessary prepositions. For verbal relations, if there is a preposition at the end of the relation word sequence, then there must be a preposition link between the relation and any of the two entities, and vice versa. For instance, in the sentence "Bob teaches at the Univer-

[4]The head words of phrases are words which do not depend on any words in the phrases.

sity" <Bob, *teach at*, University> is correct while <Bob, *teach*, University> is wrong. For nominal relations, inclusion of the head word is necessary. Prepositions can be ignored, but if they exist, they must match with the dependency link. We concentrate on verb prepositions because prepositions are more attached to noun phrases than verb phrases. Verb relations have more preposition choices, and different choices have different semantic impact, for example, the subject or object. But noun relations' preposition are more fixed, such as "president of". The last two features F7 and F8 are added according to the observation of experiment results in a development set: we note that one problem is the apposition or conjunction structure between entities [5].

6 Unsupervised Method

We also propose the use of an unsupervised method based on heuristic rules to produce a relation word noise filter, as an alternative to using SVM in the Triple task. The heuristic rules are also based on the Stanford collapsed dependency parsing. There are two parts in the noise filter: one is that the relation words should have necessary links with two entities and the other is that relation words should be consistent.

We first mention the heuristic rules for necessary dependency links. The intuition is from Chan and Dan (2011), they classified relations into 5 different syntactic structures; premodifier, possessive, preposition, formulaic, and verbal. They proposed heuristic POS patterns covering the first four patterns with the exception of the verbal structure.

We present heuristic rules based on dependency paths instead of POS for the structures, except the category formulaic, which are implicit relations. In a premodifier structure one entity and the relation are modifiers of the other entity, (e.g., US. *President* Obama). In a possessive structure one entity is in a possessive case (e.g., Microsoft's *CEO* Steve Ballmer). In a preposition structure, relation words are related with one entity by a preposition (e.g., Steve Ballmer, *CEO* of Microsoft). In a verbal structure relations are verb phrases.

The heuristic rules are presented in Figure 5. The

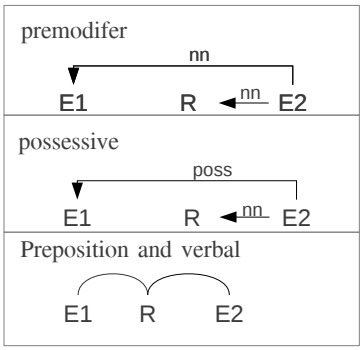


Figure 5: Dependent link heuristics for relation detection.

premodifier and possessive relation words are not in the Stanford collapsed form of the dependency path between two entities. When there is a direct dependency link between two entities that is labelled *nn* or *poss*, there should be an *nn* link between the second entity and the relation candidate (in Figure 5's top two rows). Otherwise, there should be links between the two entities and the relation, respectively (in Figure 5's last row). In this case, link types and directions are not constrained. For example, both *E1* ←(nsubj) *R* →(dobj) *E2* for the triple <Obama, visit, Canada> in "*Obama* visited *Canada*." and *E1* →(appos) *R* →(prep_of) *E2* for the triple <Obama, president, United States> in "*Obama*, the president of the *United States*, visited Canada." belong to that structure. To refine the verbal pattern, the link between the relation words and entities cannot be a conjunction.

Next, we need to check the consistency of relation words. Two separated sequences of relation words should have a dependency link between each other to confirm that they are semantically related. Relation sequences should include only necessary prepositions.

7 Experiments

We compared the unsupervised heuristic rule method and the supervised SVM method discussed above against REVERB (Fader et al., 2011) and OLLIE (Mausam et al., 2012), using three datasets. One dataset consists of sentences from the Penn Treebank, and the other two are the experiment datasets of each of the two systems being compared.

[5] But adding the two features seems does not solve the problem.

E feature	F1	the dependency link label between two entities, *null* if none.
R features	F2	whether relation is a noun phrase or a verb phrase
	F3	whether there is a link between the two segments (if there are two discontinuous segments)
between E and R	F4	whether there is a link between entities and the relation
	F5	the shortest dependency path distance between entities and the relation (1,2,3,4, or >4)
	F6	the preposition link and the last preposition word of relation (if there is such a link or word)
	F7	whether there is a conjunction link in the shortest path between entities and the relation
	F8	whether there is a apposition link in the shortest path between entities and the relation

Table 1: Noise filter feature vector.

7.1 Treebank Set

7.1.1 Preparing Data

Within the research community, it is difficult to find Open IE test data which includes all kinds of relations. So we have created our own data from the Penn Treebank for evaluation[6]. We assess the drop in performance introduced by using a tool to parse sentences compared to using "ideal" parse trees provided in the Penn Treebank. Named entities are tagged for every sentence using the Stanford NLP tool. Candidate NE pairs are extracted within a certain distance[7]. We randomly selected 756 sentences from WSJ Sections 2-21 as our training set, 100 each from Section 22 and Section 23-24 as the development and the test set, respectively. This is also the setting for most parsers.

We manually annotated whether there is a relation between two entities in a sentence (for evaluation of the Binary task). If there is a relation between two entities, the annotator needs to indicate which words are relation words (for evaluation of the Triple task). There is no restriction of relation forms for the annotator in this task.

We manually analyzed 417 relation instances from our training set. 28% are implicit relations, i.e., relations without words or with prepositions. Less than 1% are with adjectives, while 71% are noun or verb phrases. In the 71%, 60% are noun relations and 40% are verbal. The relation pattern in Section 4 can extract 80% of them. Our data contains more verbal relations than the ACE's RDC, less than corpora in other Open IE papers.

We compare every system by recall, precision, and F-score. The evaluation of the Binary task is based on entity pairs and is straightforward. The evaluation of the Triple task is based on relation triples. We need to manually compare the triples extracted by each system and the gold standard to avoid double-counting. For instance, if both *vice president* and *president* are extracted, it is counted as one[8]. Several entity pairs have multiple relations, such as "A is CEO and founder of B." Any relation which can not be represented by a verb or noun is counted as one miss in the Triple task.

To compare with the REVERB system, NE pairs are labelled as two noun phrase chunks for the system input. It is difficult to compare with OLLIE, as the system is a black box with integrated entity extraction and parsing. We compared manually the pairs extracted by OLLIE and the tagged data. Only results of intersection entity pairs are considered. The threshold of OLLIE and REVERB confidence is set to achieve the best F-score in the development set.

7.1.2 Results

The Binary task results on the test set are shown in Table 2. Each system decides whether there is a relation between two entities. The heuristic rule (DP rules) method, REVERB, and OLLIE each tag pairs containing a relation if any relation candidates are identified. As indicated, the SVM method performs the best with DP rules ranking second. Note that OLLIE uses MaltParser, so it's better to compare with the coupling of SVM with Stanford Parser, but that comparison doesn't change the result.

The Triple task results are shown in Table 3. Each system extracts relation triples from sentences. The SVM features include both tree (Figure 4) and vector features (Table 1). *All relations* in the table include nominal, verbal, and implicit relations. To scrutinize

[6]The data can be downloaded from http://cs.ualberta.ca/~yx2/

[7]Here we set the distance as 20, determined by empirical evidence, a majority of the relations are within this distance.

[8]It is difficult to decide if *president* in this case is wrong. This is related to multi-word expression and will be future work.

	P	R	F-score
Treebank parsing + DP rules	0.833	0.549	0.662
Treebank parsing + SVM	0.896	0.767	0.826
Stanford parsing + DP rules	0.783	0.522	0.627
Stanford parsing + SVM	0.744	0.711	0.727
REVERB (no parsing)	0.333	0.1	0.153
OLLIE (MaltParser)	0.583	0.389	0.467

Table 2: Relation extraction results on Treebank set (Binary)

All relations	P	R	F-score
Treebank parsing + DP rules	0.741	0.467	0.573
Treebank parsing + SVM	0.824	0.462	**0.592**
Stanford parsing + SVM	0.75	0.433	**0.549**
Stanford parsing(also for pattern)+SVM	0.7	0.43	**0.53**
OLLIE (MaltParser)	0.583	0.389	0.467
Noun relations	P	R	F-score
Treebank parsing + DP rules	0.75	0.735	0.742
Treebank parsing + SVM	0.829	0.708	**0.764**
Stanford parsing + SVM	0.756	0. 689	**0.721**
OLLIE (MaltParser)	0.8	0.408	0.54
Verb relations	P	R	F-score
Treebank parsing + DP rules	0.7	0.368	0.483
Treebank parsing + SVM	0.727	0.381	0.5
Stanford parsing + SVM	0.727	0.32	0.444
REVERB (no parsing)	0.286	0.381	0.327
OLLIE (MaltParser)	0.429	0.714	**0.536**

Table 3: Relation extraction results on Treebank set (Triple)

the result, we also show the results on noun and verb relations separately. The SVM model achieves best performance, 33% improvement on nominal relation extractions over OLLIE. (Revised: Later we found that the result 0.549 is when relation word extraction is based on the Treebank gold standard POS tagging. We reran our system with relation word extraction based on the Stanford POS tagging. The result is similar, with F-score as 0.53.)

The loss of recall for systems (except SVM) in the Binary task can be explained by the fact that nearly 20% of relations are implicit.

In both the Binary and Triple tasks, one source of failure arose from conjunction and apposition structures. For example, in the sentence "...industry executives analyzed the appointment of the new chief executive, Robert Louis-Dreyfus, who joins Saatchi ..." the method can detect the relation <*chief executive*, *joins*, *Saatchi*>, but not <*Robert Louis-Dreyfus*, *joins*, *Saatchi*>. We attempted to address this problem by adding features into SVM linear kernel (Table 1), but this has not worked in our tests.

	P	R	F-score
Stanford parsing + DP rules	0.711	0.811	0.756
Stanford parsing + SVM	0.718	0.859	**0.781**
REVERB	0.577	0.95	0.716

Table 4: Relation extraction results on REVERB set (Triple).

One cause of recall loss in the Triple task for REVERB and our two approaches is that verbal relation words can be non-consecutive. For instance, the preposition might be far away from the related verb in one sentence, in which case both our methods and REVERB can not confirm that extraction. OLLIE has better results on verb relations mainly because they use dependency link patterns to extract relation words, which alleviate the problem. On the other side, one drawback of OLLIE is that it failed to extract a few premodifer structure relations, e.g. "U.S. President Obama." That may happen because they do not have an independent step for named entity extraction, which is crucial for that type of relations.

7.2 REVERB Set

The authors of the REVERB method provide 1000 tagged training sentences and 500 test sentences. They also provide REVERB's extracted relations and instances' confidence for the 500 test sentences. The 500 test sentences are segmented into 5 folds for a significance t-test. At each iteration, the remaining 400 sentences are used as a development set to set the threshold of REVERB confidence.

To compare with REVERB, we use as input the sentences parsed by the Stanford parser and relation triples extracted by REVERB for both training and testing. The output of our system is true or false for every triple by using the tree kernel[9]. The SVM system is trained on the 1000 training sentences. The results are shown in Table 4. Only SVM is statistically significant better than REVERB (with $\alpha = 0.05$)[10].

[9]The polynomial kernel is not used for REVERB and OLLIE data as the their relation word form is simpler than ours.

[10]Note that the results here seem better than the results shown on (Fader et al., 2011). It is because our evaluation is based on the set REVERB extracted, as we only want to compare noise filters not with entity extraction, while the results in (Fader et al., 2011) is based on the union relation set of several systems.

	P	R	F-score
Stanford parsing + SVM	0.685	0.941	**0.793**
OLLIE	0.667	0.961	0.787

Table 5: Relation extraction results on OLLIE set (Triple).

7.3 OLLIE set

The authors of the OLLIE system provide a test set which has 300 sentences and OLLIE extracted 900 triples. Experiment setting is similar to that of REVERB set. The SVM tree kernel model is trained on OLLIE's leave one out dataset. The results in Table 5 show our method achieves slightly better performance, although not statistically significant.

Besides errors caused by parsing, one main cause of loss of precision is that our system is unable to detect entities that are wrong as we only concern the head of the entity. For instance, "Bogan 's Birmingham Busters , before moving to Los Angeles , California" is one entity in one OLLIE relation, where only "Bogan 's Birmingham Busters" is the correct entity.

8 Conclusion

We have described some of the limits of current Open IE systems, which concentrate on identifying explicit relations, i.e., relations which are mediated by open class words. This strategy ignores what we describe as implicit relations, e.g., *locate* relations in "Washington, U.S." We propose two subtasks for Open IE: first confirming whether there is a relation between two entities, and then whether a relation thus extracted is correct. The first task include both implicit and explicit relations; the second task is common in the previous Open IE which deals with explicit relations. In our case we have developed an Open IE system which uses SVM tree kernels applied to dependency parses for both tasks. Our system achieves superior results on several datasets. We also propose an unsupervised method which is based on heuristic rules from dependency parse links, and compared that with our SVM tree kernel methods. Our experiments show it is a strong baseline for Open IE.

For further work, we intend to improve Open IE by tackling the conjunction and apposition structure problem. Another direction will be to extract relation words for implicit relations. Relation words such as *locate* for "Washington, U.S." will be considered.

Acknowledgments

We would like to acknowledge Prof. Grzegorz Kondrak 's valuable advice. We also want to thank the anonymous reviewers for their helpful suggestions. This work was funded in part by the NSERC Business Intelligence Network, Alberta Innovates Center for Machine Learning (AICML) and Alberta Innovates Technology Futures (AITF).

References

Alan Akbik. 2009. Wanderlust : Extracting semantic relations from natural language text using dependency grammar patterns. In *WWW 2009 Workshop on Semantic Search*, volume 137, pages 279–290.

Michele Banko, Michael J Cafarella, Stephen Soderl, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *In International Joint Conference on Artificial Intelligence*, pages 2670–2676.

Razvan C. Bunescu and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 724–731, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yee Seng Chan and Roth Dan. 2011. Exploiting syntactico-semantic structures for relation extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 551–560, Stroudsburg, PA, USA. Association for Computational Linguistics.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2011. An analysis of open information extraction based on semantic role labeling. In *Proceedings of the sixth international conference on Knowledge capture*, K-CAP '11, pages 113–120, New York, NY, USA. ACM.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems 14*, pages 625–632. MIT Press.

Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2011. Structured lexical similarity via convolution kernels on dependency trees. In *Proceedings of the*

Conference on Empirical Methods in Natural Language Processing, EMNLP '11, pages 1034–1046, Stroudsburg, PA, USA. Association for Computational Linguistics.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

M.C. De Marneffe, B. MacCartney, and C.D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1535–1545, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bill MacCartney, Trond Grenager, Marie-Catherine de Marneffe, Daniel Cer, and Christopher D. Manning. 2006. Learning to recognize features of valid textual entailments. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 41–48. Association for Computational Linguistics.

Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534. Association for Computational Linguistics.

Alessandro Moschitti. 2005. *Automatic text categorization: from information retrieval to support vector learning*. Aracne.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *Proceedings of the 17th European conference on Machine Learning*, ECML'06, pages 318–329, Berlin, Heidelberg. Springer-Verlag.

Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2009. Convolution kernels on constituent, dependency and sequential structures for relation extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1378–1387, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mengqui Wang. 2008. A re-examination of dependency path kernels for relation extraction. In *Proceedings of the Third International Joint Conference on Natural Language Processing*, pages 841–846, Hyderabad, India. Asian Federation of Natural Language Processing, Association for Computational Linguistics.

Fei Wu and Daniel S. Weld. 2010. Open information extraction using wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 118–127, Stroudsburg, PA, USA. Association for Computational Linguistics.

Guo-Dong Zhou and Qiao-Ming Zhu. 2011. Kernel-based semantic relation detection and classification via enriched parse tree structure. *J. Comput. Sci. Technol.*, 26(1):45–56, January.