Convolution Kernels on Constituent, Dependency and Sequential Structures for Relation Extraction

Truc-Vien T. Nguyen and **Alessandro Moschitti** and **Giuseppe Riccardi**
nguyenthi,moschitti,riccardi@disi.unitn.it
Department of Information Engineering and Computer Science
University of Trento
38050 Povo (TN), Italy

Abstract

This paper explores the use of innovative kernels based on syntactic and semantic structures for a target relation extraction task. Syntax is derived from constituent and dependency parse trees whereas semantics concerns to entity types and lexical sequences. We investigate the effectiveness of such representations in the automated relation extraction from text. We process the above data by means of Support Vector Machines along with the syntactic tree, the partial tree and the word sequence kernels. Our study on the ACE 2004 corpus illustrates that the combination of the above kernels achieves high effectiveness and significantly improves the current state-of-the-art.

> Jeff Zucker, the longtime executive producer of NBC's "Today" program, will be named Friday as the new **president** of **NBC's entertainment division**, replacing Garth Ancier, NBC executives said.

Figure 1: A document from ACE 2004 with all entity mentions in bold.

1 Introduction

Relation Extraction (RE) is defined in ACE as the task of finding relevant semantic relations between pairs of entities in texts. Figure 1 shows part of a document from ACE 2004 corpus, a collection of news articles. In the text, the relation between *president* and *NBC's entertainment division* describes the relationship between the first entity (person) and the second (organization) where the person holds a managerial position.

Several approaches have been proposed for automatically learning semantic relations from texts. Among others, there has been increased interest in the application of kernel methods (Zelenko et al., 2002; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005a; Bunescu and Mooney, 2005b; Zhang et al., 2005; Wang, 2008). Their main property is the ability of exploiting a huge amount of features without an explicit feature representation. This can be done by computing a kernel function between a pair of linguistic objects, where such function is a kind of similarity measure satisfying certain properties. An example is the sequence kernels (Lodhi et al., 2002), where the objects are strings of characters and the kernel function computes the number of common subsequences of characters in the two strings. Such substrings are then weighted according to a decaying factor penalizing longer ones. In the same line, Tree Kernels count the number of *subtree* shared by two input trees. An example is that of syntactic (or subset) tree kernel (SST) (Collins and Duffy, 2001), where trees encode grammatical derivations.

Previous work on the use of kernels for RE has exploited some similarity measures over diverse features (Zelenko et al., 2002; Culotta and Sorensen, 2004; Zhang et al., 2005) or subsequence kernels over dependency graphs (Bunescu and Mooney, 2005a; Wang, 2008). More specifically, (Bunescu and Mooney, 2005a; Culotta and Sorensen, 2004) use kernels over dependency trees, which showed much lower accuracy than feature-based methods (Zhao and Grishman, 2005). One problem of the dependency kernels above is that they do not exploit the overall structural aspects of dependency trees. A more effective solution is the application of convolution kernels to constituent parse trees (Zhang et al., 2006) but this is not satisfactory from a general per-

This work has been partially funded by the LiveMemories project (http://www.livememories.org/) and Expert System (http://www.expertsystem.net/) research grant.

spective since dependency structures offer some unique advantages, which should be exploited by an appropriate kernel.

Therefore, studying convolution tree kernels for dependency trees is worthwhile also considering that, to the best of our knowledge, these models have not been previously used for relation extraction[1] task. Additionally, sequence kernels should be included in such global study since some of their forms have not been applied to RE.

In this paper, we study and evaluate diverse convolution and sequence kernels for the RE problem by providing several kernel combinations on constituent and dependency trees and sequential structures. To fully exploit the potential of dependency trees, in addition to the SST kernels, we applied the partial tree (PTK) kernel proposed in (Moschitti, 2006), which is a general convolution tree kernel adaptable for dependency structures. We also investigate various sequence kernels (e.g. the word sequence kernel (WSK) (Cancedda et al., 2003)) by incorporating dependency structures into word sequences. These are also enriched by including information from constituent parse trees.

We conduct experiments on the standard ACE 2004 newswire and broadcast news domain. The results show that although some kernels are less effective than others, they exhibit properties that are complementary to each other. In particular, we found that relation extraction can benefit from increasing the feature space by combining kernels (with a simple summation) exploiting the two different parsing paradigms. Our experiments on RE show that the current composite kernel, which is constituent-based is more effective than those based on dependency trees and individual sequence kernel but at the same time their combinations, i.e. dependency plus constituent trees, improve the state-of-the-art in RE. More interestingly, also the combinations of various sequence kernels gain significant better performance than the current state-of-the-art (Zhang et al., 2005).

Overall, these results are interesting for the computational linguistics research since they show that the above two parsing paradigms provide different and important information for a semantic task such as RE. Regarding sequence-based kernels, the WSK gains better performance than previous sequence and dependency models for RE.

A review of previous work on RE is described in Section 2. Section 3 introduces support vector machines and kernel methods whereas our specific kernels for RE are described is Section 4. The experiments and conclusions are presented in sections 5 and 6, respectively.

2 Related Work

To identify semantic relations using machine learning, three learning settings have mainly been applied, namely supervised methods (Miller et al., 2000; Zelenko et al., 2002; Culotta and Sorensen, 2004; Kambhatla, 2004; Zhou et al., 2005), semi supervised methods (Brin, 1998; Agichtein and Gravano, 2000), and unsupervised method (Hasegawa et al., 2004). In a supervised learning setting, representative related work can be classified into generative models (Miller et al., 2000), feature-based (Roth and tau Yih, 2002; Kambhatla, 2004; Zhao and Grishman, 2005; Zhou et al., 2005) or kernel-based methods (Zelenko et al., 2002; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005a; Zhang et al., 2005; Wang, 2008; Zhang et al., 2006).

The learning model employed in (Miller et al., 2000) used statistical parsing techniques to learn syntactic parse trees. It demonstrated that a lexicalized, probabilistic context-free parser with head rules can be used effectively for information extraction. Meanwhile, feature-based approaches often employ various kinds of linguistic, syntactic or contextual information and integrate into the feature space. (Roth and tau Yih, 2002) applied a probabilistic approach to solve the problems of named entity and relation extraction with the incorporation of various features such as word, part-of-speech, and semantic information from WordNet. (Kambhatla, 2004) employed maximum entropy models with diverse features including words, entity and mention types and the number of words (if any) separating the two entities.

Recent work on Relation Extraction has mostly employed kernel-based approaches over syntactic parse trees. Kernels on parse trees were pioneered by (Collins and Duffy, 2001). This kernel function counts the number of common subtrees, weighted appropriately, as the measure of similarity between two parse trees. (Culotta and Sorensen, 2004) extended this work to calculate kernels between augmented dependency

[1] The function defined on (Culotta and Sorensen, 2004), although on dependency trees, is not a convolution tree kernel.

trees. (Zelenko et al., 2002) proposed extracting relations by computing kernel functions between parse trees. (Bunescu and Mooney, 2005a) proposed a shortest path dependency kernel by stipulating that the information to model a relationship between two entities can be captured by the shortest path between them in the dependency graph.

Although approaches in RE have been dominated by kernel-based methods, until now, most of research in this line has used the kernel as some similarity measures over diverse features (Zelenko et al., 2002; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005a; Zhang et al., 2005; Wang, 2008). These are not convolution kernels and produce a much lower number of substructures than PTK. A recent approach successfully employs a convolution tree kernel (of type SSTs) over constituent syntactic parse tree (Zhang et al., 2006; Zhou et al., 2007), but it does not capture grammatical relations in dependency structure. We believe that an efficient and appropriate kernel can be used to solve the RE problem, exploiting the advantages of dependency structures, convolution tree kernels and sequence kernels.

3 Support Vector Machines and Kernel Methods

In this section we give a brief introduction to support vector machines, kernel methods, diverse tree and sequence kernel spaces, which can be applied to the RE task.

3.1 Support Vector Machines (SVMs)

Support Vector Machines refer to a supervised machine learning technique based on the latest results of the statistical learning theory (Vapnik, 1998). Given a vector space and a set of training points, i.e. positive and negative examples, SVMs find a separating hyperplane $H(\vec{x}) = \vec{\omega} \times \vec{x} + b = 0$ where $\omega \in R^n$ and $b \in R$ are learned by applying the Structural Risk Minimization principle (Vapnik, 1995). SVMs is a binary classifier, but it can be easily extended to multi-class classifier, e.g. by means of the *one-vs-all* method (Rifkin and Poggio, 2002).

One strong point of SVMs is the possibility to apply kernel methods (robert Mller et al., 2001) to implicitly map data in a new space where the examples are *more easily* separable as described in the next section.

3.2 Kernel Methods

Kernel methods (Schlkopf and Smola, 2001) are an attractive alternative to feature-based methods since the applied learning algorithm only needs to compute a product between a pair of objects (by means of kernel functions), avoiding the explicit feature representation. A kernel function is a scalar product in a possibly unknown feature space. More precisely, The object o is mapped in $\vec{x}$ with a feature function $\phi : \mathcal{O} \rightarrow \Re^n$, where $\mathcal{O}$ is the set of the objects.

The kernel trick allows us to rewrite the decision hyperplane as:

$$H(\vec{x}) = \Big(\sum_{i=1..l} y_i \alpha_i \vec{x}_i \Big) \cdot \vec{x} + b =$$

$$\sum_{i=1..l} y_i \alpha_i \vec{x}_i \cdot \vec{x} + b = \sum_{i=1..l} y_i \alpha_i \phi(o_i) \cdot \phi(o) + b,$$

where y_i is equal to 1 for positive and -1 for negative examples, $\alpha_i \in \Re$ with $\alpha_i \geq 0$, o_i $\forall i \in \{1,..,l\}$ are the training instances and the product $K(o_i, o) = \langle \phi(o_i) \cdot \phi(o) \rangle$ is the kernel function associated with the mapping ϕ.

Kernel engineering can be carried out by combining basic kernels with additive or multiplicative operators or by designing specific data objects (vectors, sequences and tree structures) for the target tasks.

Regarding NLP applications, kernel methods have attracted much interest due to their ability of implicitly exploring huge amounts of structural features automatically extracted from the original object representation. The kernels for structured natural language data, such as parse tree kernel (Collins and Duffy, 2001) and string kernel (Lodhi et al., 2002) are examples of the well-known convolution kernels used in many NLP applications.

Tree kernels represent trees in terms of their substructures (called tree fragments). Such fragments form a feature space which, in turn, is mapped into a vector space. Tree kernels measure the similarity between pair of trees by counting the number of fragments in common. There are three important characterizations of fragment type (Moschitti, 2006): the SubTrees (STs), the SubSet Trees (SSTs) and the Partial Trees (PTs). For sake of space, we do not report the mathematical description of them, which is available in (Vishwanathan and Smola, 2002), (Collins and Duffy,

2001) and (Moschitti, 2006), respectively. In contrast, we report some descriptions in terms of feature space that may be useful to understand the new engineered kernels.

In principle, a SubTree (ST) is defined by taking any node along with its descendants. A SubSet Tree (SST) is a more general structure which does not necessarily include all the descendants. It must be generated by applying the same grammatical rule set, which generated the original tree. A Partial Tree (PT) is a more general form of substructures obtained by relaxing constraints over the SSTs.

4 Kernels for Relation Extraction

In this section we describe the previous kernels based on constituent trees as well as new kernels based on diverse types of trees and sequences for relation extraction. As mentioned in the previous section, we can engineer kernels by combining tree and sequence kernels. Thus we focus on the problem to define structure embedding the desired syntactic relational information between two named entities (NEs).

4.1 Constituent and Dependency Structures

Syntactic parsing (or syntactic analysis) aims at identifying grammatical structures in a text. A parser thus captures the hidden hierarchy of the input text and processes it into a form suitable for further processing. There are two main paradigms for representing syntactic information: constituent and dependency parsing, which produces two different tree structures.

Constituent tree encodes structural properties of a sentence. The parse tree contains constituents, such as noun phrases (NP) and verb phrases (VP), as well as terminals/part-of-speech tags, such as determiners (DT) or nouns (NN). Figure 2.a shows the constituent tree of the sentence: *In Washington, U.S. officials are working overtime*.

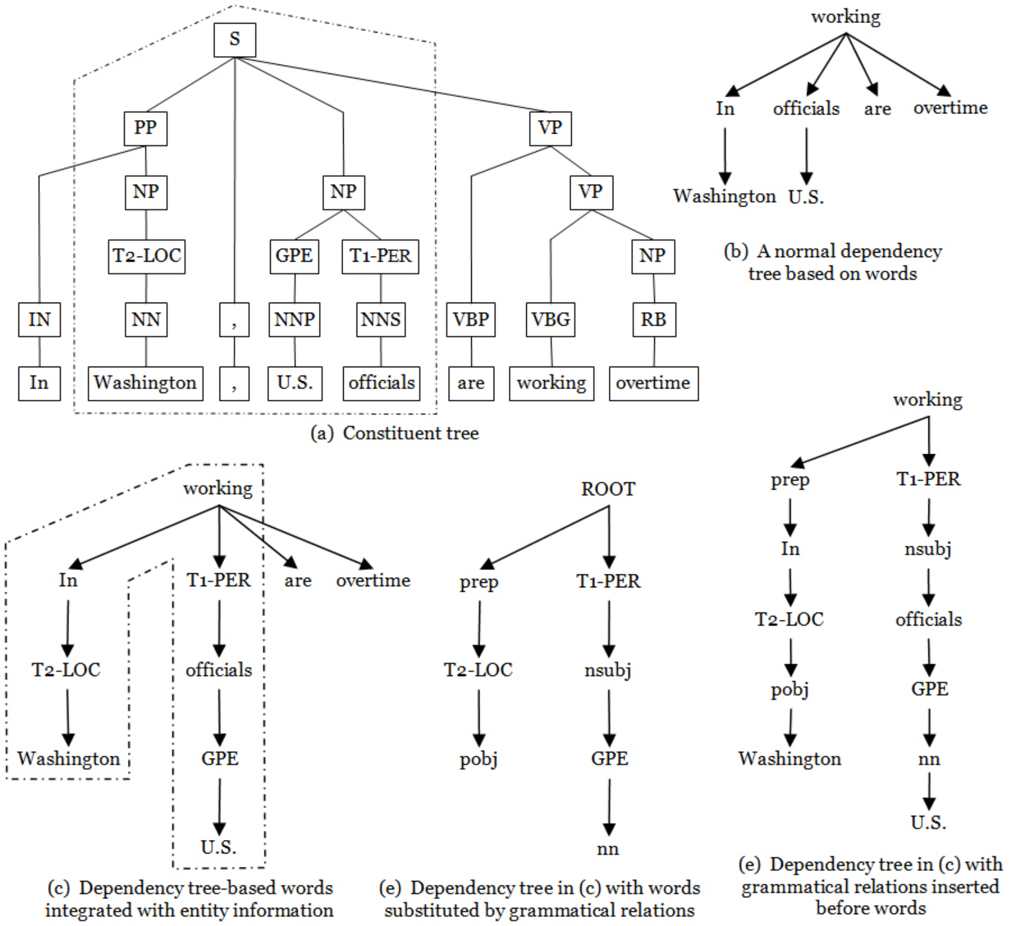
Dependency tree encodes grammatical relations between words in a sentence with the words as nodes and dependency types as edges. An edge from a word to another represents a grammatical relation between these two. Every word in a dependency tree has exactly one parent except the root. Figure 2.b shows and example of the dependency tree of the previous sentence.

Given two NEs, such as *Washington* and *officials*, both the above trees can encode the syntactic dependencies between them. However, since each parse tree corresponds to a sentence, there may be more than two NEs and many relations expressed in a sentence. Thus, the use of the entire parse tree of the whole sentence holds two major drawbacks: first, it may be too computationally expensive for kernel calculation since the size of a complete parse tree may be very large (up to 300 nodes in the Penn Treebank (Marcus et al., 1993)); second, there is ambiguity on the target pairs of NEs, i.e. different NEs associated with different relations are described by the same parse tree. Therefore, it is necessary to identify the portion of the parse tree that best represent the useful syntactic information.

Let e_1 and e_2 be two entity mentions in the same sentence such that they are in a relationship R. For the constituent parse tree, we used the path-enclosed tree (PET), which was firstly proposed in (Moschitti, 2004) for Semantic Role Labeling and then adapted by (Zhang et al., 2005) for relation extraction. It is the smallest common subtree including the two entities of a relation. The dashed frame in Figure 2.a surrounds PET associated with the two mentions, *officials* and *Washington*. Moreover, to improve the representation, two extra nodes T1-PER, denoting the type PERSON, and T2-LOC, denoting the type LOCATION, are added to the parse tree, above the two target NEs, respectively. In this example, the above PET is designed to capture the relation *Located-in* between the entities ”officials” and ”Washington” from the ACE corpus. Note that, a third NE, *U.S.*, is characterized by the node GPE (GeoPolitical Entity), where the absence of the prefix T1 or T2 before the NE type (i.e. GPE), denotes that the NE does not take part in the target relation.

In previous work, some dependency trees have been used (Bunescu and Mooney, 2005a; Wang, 2008) but the employed kernel just exploited the syntactic information concentrated in the path between e_1 and e_2. In contrast, we defined and studied three different dependency structures whose potential can be fully exploited by our convolution partial tree kernel:

- Dependency Words (DW) tree is similar to PET adapted for dependency tree constituted by simple words. We select the minimal subtree which includes e_1 and e_2, and we insert an extra node as father of the NEs, labeled with the NE category. For example, given



(a) Constituent tree

(b) A normal dependency tree based on words

(c) Dependency tree-based words integrated with entity information

(e) Dependency tree in (c) with words substituted by grammatical relations

(e) Dependency tree in (c) with grammatical relations inserted before words

Figure 2: The constituent and dependency parse trees integrated with entity information

the tree in Figure 2.b, we design the tree in Figure 2.c surrounded by the dashed frames, where T1-PER, T2-LOC and GPE are the extra nodes inserted as fathers of *Washington*, *soldier* and *U.S.*.

- Grammatical Relation (GR) tree, i.e. the DW tree in which words are replaced by their grammatical functions, e.g. *prep*, *pobj* and *nsubj*. For example, Figure 2.d, shows the GR tree for the previous relation: *In* is replaced by *prep* , *U.S.* by *nsubj* and so on.

- Grammatical Relation and Words (GRW) tree, words and grammatical functions are both used in the tree, where the latter are inserted as a father node of the former. For example, Figure 2.e, shows such tree for the previous relation.

4.2 Sequential Structures

Some sequence kernels have been used on dependency structures (Bunescu and Mooney, 2005b; Wang, 2008). These kernels just used lexical words with some syntactic information. To fully exploit syntactic and semantic information, we defined and studied six different sequences (in a style similar to what proposed in (Moschitti, 2008)), which include features from constituent and dependency parse trees and NEs:

1. Sequence of terminals (lexical words) in the PET (SK_1), e.g.:
T2-LOC Washington , U.S. T1-PER officials.

2. Sequence of part-of-speech (POS) tags in the PET (SK_2), i.e. the SK_1 in which words are replaced by their POS tags, e.g.:
T2-LOC NN , NNP T1-PER NNS.

3. Sequence of grammatical relations in the PET (SK_3), i.e. the SK_1 in which words are

replaced by their grammatical functions, e.g.: *T2-LOC pobj , nn T1-PER nsubj.*

4. Sequence of words in the DW (SK_4), e.g.: *Washington T2-LOC In working T1-PER officials GPE U.S.*.

5. Sequence of grammatical relations in the GR (SK_5), i.e. the SK_4 in which words are replaced by their grammatical functions, e.g.: *pobj T2-LOC prep ROOT T1-PER nsubj GPE nn.*

6. Sequence of POS tags in the DW (SK_6), i.e. the SK_4 in which words are replaced by their POS tags, e.g.: *NN T2-LOC IN VBP T1-PER NNS GPE NNP.*

It is worth noting that the potential information contained in such sequences can be fully exploited by the word sequence kernel.

4.3 Combining Kernels

Given that syntactic information from different parse trees may have different impact on relation extraction (RE), the viable approach to study the role of dependency and constituent parsing is to experiment with different syntactic models and measuring the impact in terms of RE accuracy. For this purpose we compared the composite kernel described in (Zhang et al., 2006) with the partial tree kernels applied to DW, GR, and GRW and sequence kernels based on six sequences described above. The composite kernels include polynomial kernel applied to entity-related feature vector. The word sequence kernel (WSK) is always applied to sequential structures. The used kernels are described in more detail below.

4.3.1 Polynomial Kernel

The basic kernel between two named entities of the ACE documents is defined as:

$$K_P(R_1, R_2) = \sum_{i=1,2} K_E(R_1.E_i, R_2.E_i),$$

where R_1 and R_2 are two relation instances, E_i is the i^{th} entity of a relation instance. $K_E(\cdot, \cdot)$ is a kernel over entity features, i.e.:

$$K_E(E_1, E_2) = (1 + \vec{x}_1 \cdot \vec{x}_2)^2,$$

where $\vec{x}_1$ and $\vec{x}_2$ are two feature vectors extracted from the two NEs.

For the ACE 2004, the features used include: entity headword, entity type, entity subtype, mention type, and LDC[2] mention type. The last four attributes are taken from the ACE corpus 2004. In ACE, each mention has a head annotation and an extent annotation.

4.3.2 Kernel Combinations

1. Polynomial kernel plus a tree kernel:

$$CK_1 = \alpha \cdot K_P + (1 - \alpha) \cdot K_x,$$

where α is a coefficient to give more impact to K_P and K_x is either the partial tree kernel applied to one the possible dependency structures, DW, GR or GRW or the SST kernel applied to PET, described in the previous section.

2. Polynomial kernel plus constituent plus dependency tree kernels:

$$CK_2 = \alpha \cdot K_P + (1 - \alpha) \cdot (K_{SST} + K_{PTK})$$

where K_{SST} is the SST kernel and K_{PTK} is the partial tree kernel (applied to the related structures as in point 1).

3. Constituent tree plus square of polynomial kernel and dependency tree kernel:

$$CK_3 = \alpha \cdot K_{SST} + (1 - \alpha) \cdot (K_P + K_{PTK})^2$$

4. Dependency word tree plus grammatical relation tree kernels:

$$CK_4 = K_{PTK-DW} + K_{PTK-GR}$$

where K_{PTK-DW} and K_{PTK-GR} are the partial tree kernels applied to dependency structures DW and GR.

5. Polynomial kernel plus dependency word plus grammatical relation tree kernels:

$$CK_5 = \alpha \cdot K_P + (1 - \alpha) \cdot (K_{PTK-DW} + K_{PTK-GR})$$

Some preliminary experiments on a validation set showed that the second, the fourth and the fifth combinations yield the best performance with $\alpha = 0.4$ while the first and the third combinations yield the best performance with $\alpha = 0.23$.

Regarding WSK, the following combinations are applied:

[2] Linguistic Data Consortium (LDC): http://www.ldc.upenn.edu/Projects/ACE/

1. $SK_3 + SK_4$
2. $SK_3 + SK_6$
3. $SSK = \sum_{i=1,..,6} SK_i$
4. $K_{SST} + SSK$
5. $CSK = \alpha \cdot K_P + (1 - \alpha) \cdot (K_{SST} + SSK)$

Preliminary experiments showed that the last combination yields the best performance with $\alpha = 0.23$.

We used a polynomial expansion to explore the bi-gram features of i) the first and the second entity participating in the relation, ii) grammatical relations which replace words in the dependency tree. Since the kernel function set is closed under normalization, polynomial expansion and linear combination (Schlkopf and Smola, 2001), all the illustrated composite kernels are also proper kernels.

5 Experiments

Our experiments aim at investigating the effectiveness of convolution kernels adapted to syntactic parse trees and various sequence kernels for the RE task. For this purpose, we use the subset and partial tree kernel over different kinds of trees, namely constituent and dependency syntactic parse trees. Diverse sequences are applied individually and in combination together. We consider our task of relation extraction as a classification problem where categories are relation types. All pairs of entity mentions in the same sentence are taken to generate potential relations, which will be processed as positive and negative examples.

5.1 Experimental setup

We use the newswire and broadcast news domain in the English portion of the ACE 2004 corpus provided by LDC. This data portion includes 348 documents and 4400 relation instances. It defines seven entity types and seven relation types. Every relation is assigned one of the seven types: Physical, Person/Social, Employment/Membership/-Subsidiary, Agent-Artifact, PER/ORG Affiliation, GPE Affiliation, and Discourse. For sake of space, we do not explain these relationships here, nevertheless, they are explicitly described in the ACE document guidelines. There are 4400 positive and 38,696 negative examples when generating pairs of entity mentions as potential relations.

Documents are parsed using Stanford Parser (Klein and Manning, 2003) to produce parse trees. Potential relations are generated by iterating all pairs of entity mentions in the same sentence. Entity information, namely entity type, is integrated into parse trees. To train and test our binary relation classifier, we used SVMs. Here, relation detection is formulated as a multiclass classification problem. The *one vs. rest* strategy is employed by selecting the instance with largest margin as the final answer. For experimentation, we use 5-fold cross-validation with the Tree Kernel Tools (Moschitti, 2004) (available at).

5.2 Results

In this section, we report the results of different kernels setup over constituent (CT) and dependency (DP) parse trees and sequences taken from these parse trees. The tree kernel (TK), composite kernel (CK_1, CK_2, CK_3, CK_4, and CK_5 corresponding to five combination types in Section 4.3.2) were employed over these two syntactic trees. For the tree kernel, we apply the SST kernel for the path-enclosed tree (PET) of the constituent tree and the PTK for three kinds of dependency tree DW, GR, and GRW, described in the previous section. The two composite kernels CK_2 and CK_3 are applied over both two parse trees. The word sequence kernels are applied over six sequences $SK_1, SK_2, SK_3, SK_4, SK_5$, and SK_6 (described in Section 4.3).

The results are shown in Table 1 and Table 2. In the first table, the first column indicates the structure used in the combination shown in the second column, e.g. PET associated with CK_1 means that the SST kernel is applied on PET (a portion of the constituent tree) and combined with the CK_1 schema whereas PET and GR associated with CK_5 means that SST kernel is applied to PET and PTK is applied to GR in CK_5. The remaining three columns report Precision, Recall and F1 measure. The interpretation of the second table is more immediate since the only tree kernel involved is the SST applied to PET and combined by means of CK_1.

We note that: first, the dependency kernels, i.e. the results on the rows from 3 to 6 are below the composite kernel CK_1, i.e. 68.9. This is the state-of-the-art in RE, designed by (Zhang et al., 2006), where our implementation provides a slightly smaller result than the original version

Parse Tree	Kernel	P	R	F
PET	$\mathbf{CK_1}$	**69.5**	**68.3**	**68.9**
DW	CK_1	53.2	59.7	56.3
GR	CK_1	58.8	61.7	60.2
GRW	CK_1	56.1	61.2	58.5
DW and GR	CK_5	59.7	64.1	61.8
PET and GR	$\mathbf{CK_2}$	**70.7**	**69.0**	**69.8**
	$\mathbf{CK_3}$	**70.8**	**70.2**	**70.5**

Table 1: Results on the ACE 2004 evaluation test set. Six structures were experimented over the constituent and dependency trees.

Kernel	P	R	F
$\mathbf{CK_1}$	**69.5**	**68.3**	**68.9**
SK_1	72.0	52.8	61.0
SK_2	61.7	60.0	60.8
SK_3	62.6	60.7	61.6
SK_4	73.1	50.3	59.7
SK_5	59.0	60.7	59.8
SK_6	57.7	61.8	59.7
$\mathbf{SK_3 + SK_4}$	**75.0**	**63.4**	**68.8**
$SK_3 + SK_6$	66.8	65.1	65.9
$\mathbf{SSK = \sum_i SK_i}$	**73.8**	**66.2**	**69.8**
$\mathbf{SST\ Kernel + SSK}$	**75.6**	**66.6**	**70.8**
$\mathbf{CK_1 + SSK}$	**76.6**	**67.0**	**71.5**
(Zhou et al., 2007) CK_1 *with Heuristics*	*82.2*	*70.2*	*75.8*

Table 2: Performance comparison on the ACE 2004 data with different kernel setups.

(i.e. an F1 of about 72 using a different syntactic parser).

Second, CK_1 improves to 70.5, when the contribution of PTK applied to GR (dependency tree built using grammatical relations) is added. This suggests that dependency structures are effectively exploited by PTK and that such information is somewhat complementary to constituent trees.

Third, in the second table, the model $CK_1 + SSK$, which adds to CK_1 the contribution of diverse sequence kernels, outperforms the state-of-the-art by 2.6%. This suggests that the sequential information encoded by several sequence kernels can better represents the dependency information.

Finally, we also report in the last row (in italic) the superior RE result by (Zhou et al., 2007). However, to achieve this outcome the authors used the composite kernel CK_1 with several heuristics to define an effective portion of constituent trees. Such heuristics expand the tree and remove unnecessary information allowing a higher improvement on RE. They are tuned on the target RE task so although the result is impressive, we cannot use it to compare with pure automatic learning approaches, such us our models.

6 Conclusion and Future Work

In this paper, we study the use of several types of syntactic information: constituent and dependency syntactic parse trees. A relation is represented by taking the path-enclosed tree (PET) of the constituent tree or of the path linking two entities of the dependency tree. For the design of automatic relation classifiers, we have investigated the impact of dependency structures to the RE task. Our novel composite kernels, which account for the two syntactic structures, are experimented with the appropriate convolution kernels and show significant improvement with respect to the state-of-the-art in RE.

Regarding future work, there are many research line that may be followed:

i) Capturing more features by employing external knowledge such as ontological, lexical resource or WordNet-based features (Basili et al., 2005a; Basili et al., 2005b; Bloehdorn et al., 2006; Bloehdorn and Moschitti, 2007) or shallow semantic trees, (Giuglea and Moschitti, 2004; Giuglea and Moschitti, 2006; Moschitti and Bejan, 2004; Moschitti et al., 2007; Moschitti, 2008; Moschitti et al., 2008).

ii) Design a new tree-based structures, which combines the information of both constituent and dependency parses. From dependency trees we can extract more precise but also more sparse relationships (which may cause overfit). From constituent trees, we can extract subtrees constituted by non-terminal symbols (grammar symbols), which provide a better generalization (with a risk of underfitting).

iii) Design a new kernel which can integrate the advantages of the constituent and dependency tree. The new tree kernel should inherit the benefits of the three available tree kernels: STs, SSTs or PTs.

References

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the 5th ACM International Conference on Digital Libraries*.

Roberto Basili, Marco Cammisa, and Alessandro Moschitti. 2005a. Effective use of WordNet semantics via kernel-based learning. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 1–8, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Roberto Basili, Marco Cammisa, and Alessandro Moschitti. 2005b. A semantic kernel to classify texts with very few training examples. In *In Proceedings of the Workshop on Learning in Web Search, at the*.

Stephan Bloehdorn and Alessandro Moschitti. 2007. Structure and semantics for expressive text kernels. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 861–864, New York, NY, USA. ACM.

Stephan Bloehdorn, Roberto Basili, Marco Cammisa, and Alessandro Moschitti. 2006. Semantic kernels for text classification based on topological measures of feature similarity. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 06), Hong Kong, 18-22 December 2006*, DEC.

Sergey Brin. 1998. Extracting patterns and relations from world wide web. In *Proceeding of WebDB Workshop at 6th International Conference on Extending Database Technology*, pages 172–183.

Razvan C. Bunescu and Raymond J. Mooney. 2005a. A shortest path dependency kernel for relation extraction. In *Proceedings of EMNLP*, pages 724–731.

Razvan C. Bunescu and Raymond J. Mooney. 2005b. Subsequence kernels for relation extraction. In *Proceedings of EMNLP*.

Nicola Cancedda, Eric Gaussier, Cyril Goutte, and Jean Michel Renders. 2003. Word sequence kernels. *Journal of Machine Learning Research*, pages 1059–1082.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Proceedings of Neural Information Processing Systems (NIPS'2001)*.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on ACL*, Barcelona, Spain.

Ana-Maria Giuglea and Alessandro Moschitti. 2004. Knowledge discovery using framenet, verbnet and propbank. In A. Meyers, editor, *Workshop on Ontology and Knowledge Discovering at ECML 2004*, Pisa, Italy.

Ana-Maria Giuglea and Alessandro Moschitti. 2006. Semantic Role Labeling via Framenet, Verbnet and Propbank. In *Proceedings of ACL 2006*, Sydney, Australia.

Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. 2004. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting on ACL*, Barcelona, Spain.

Nanda Kambhatla. 2004. Combining lexical, syntactic and semantic features with maximum entropy models for extracting relations. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, Barcelona, Spain.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Meeting of the ACL*, pages 423–430.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, , and Chris Watkins. 2002. Text classification using string kernels. *Journal of Machine Learning Research*, pages 419–444.

Mitchell P. Marcus, Beatrice Santorini, , and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330.

Scott Miller, Heidi Fox, Lance Ramshaw, , and Ralph Weischedel. 2000. A novel use of statistical parsing to extract information from text. In *Proceedings of the 1st conference on North American chapter of the ACL*, pages 226–233, Seattle, USA.

Alessandro Moschitti and Cosmin Bejan. 2004. A semantic kernel for predicate argument classification. In *CoNLL-2004*, Boston, MA, USA.

Alessandro Moschitti, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. 2007. Exploiting syntactic and shallow semantic kernels for question/answer classification. In *Proceedings of ACL'07*, Prague, Czech Republic.

Alessandro Moschitti, Daniele Pighin, and Roberto Basili. 2008. Tree kernels for semantic role labeling. *Computational Linguistics*, 34(2):193–224.

Alessandro Moschitti. 2004. A study on convolution kernels for shallow semantic parsing. In *Proceedings of the 42nd Meeting of the ACL*, Barcelona, Spain.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *Proceedings of the 17th European Conference on Machine Learning*, Berlin, Germany.

Alessandro Moschitti. 2008. Kernel methods, syntax and semantics for relational text categorization. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 253–262, New York, NY, USA. ACM.

Ryan Michael Rifkin and Tomaso Poggio. 2002. *Everything old is new again: a fresh look at historical approaches in machine learning*. PhD thesis, Massachusetts Institute of Technology.

Klaus robert Mller, Sebastian Mika, Gunnar Rtsch, Koji Tsuda, , and Bernhard Schlkopf. 2001. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201.

Dan Roth and Wen tau Yih. 2002. Probabilistic reasoning for entity and relation recognition. In *Proceedings of the COLING-2002*, Taipei, Taiwan.

Bernhard Schlkopf and Alexander J. Smola. 2001. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

Vladimir N. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer–Verlag, New York.

Vladimir N. Vapnik. 1998. *Statistical Learning Theory*. John Wiley and Sons, New York.

S.V.N. Vishwanathan and Alexander J. Smola. 2002. Fast kernels on strings and trees. In *Proceedings of Neural Information Processing Systems*.

Mengqiu Wang. 2008. A re-examination of dependency path kernels for relation extraction. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing-IJCNLP*.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2002. Kernel methods for relation extraction. In *Proceedings of EMNLP-ACL*, pages 181–201.

Min Zhang, Jian Su, Danmei Wang, Guodong Zhou, and Chew Lim Tan. 2005. Discovering relations between named entities from a large raw corpus using tree similarity-based clustering. In *Proceedings of IJCNLP'2005, Lecture Notes in Computer Science (LNCS 3651)*, pages 378–389, Jeju Island, South Korea.

Min Zhang, Jie Zhang, Jian Su, , and Guodong Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of COLING-ACL 2006*, pages 825–832.

Shubin Zhao and Ralph Grishman. 2005. Extracting relations with integrated information using kernel methods. In *Proceedings of the 43rd Meeting of the ACL*, pages 419–426, Ann Arbor, Michigan, USA.

GuoDong Zhou, Jian Su, Jie Zhang, , and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Meeting of the ACL*, pages 427–434, Ann Arbor, USA, June.

GuoDong Zhou, Min Zhang, DongHong Ji, and QiaoMing Zhu. 2007. Tree kernel-based relation extraction with context-sensitive structured parse tree information. In *Proceedings of EMNLP-CoNLL 2007*, pages 728–736.