Information Sciences 180 (2010) 1313–1325

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Tree kernel-based semantic relation extraction with rich syntactic and semantic information

Zhou Guodong *, Qian Longhua, Fan Jianxi

School of Computer Science and Technology, Soochow University, 1 Shizi Street, Suzhou 215006, China

ARTICLE INFO

Article history:
Received 28 March 2008
Received in revised form 5 September 2009
Accepted 11 December 2009

Keywords:
Semantic relation extraction
Tree kernel-based methods
Context-sensitive convolution tree kernel
Rich semantic relation tree structure
Semantic information
Syntactic information

ABSTRACT

This paper proposes a novel tree kernel-based method with rich syntactic and semantic information for the extraction of semantic relations between named entities. With a parse tree and an entity pair, we first construct a rich semantic relation tree structure to integrate both syntactic and semantic information. And then we propose a context-sensitive convolution tree kernel, which enumerates both context-free and context-sensitive sub-trees by considering the paths of their ancestor nodes as their contexts to capture structural information in the tree structure. An evaluation on the Automatic Content Extraction/Relation Detection and Characterization (ACE RDC) corpora shows that the proposed tree kernel-based method outperforms other state-of-the-art methods.

1. Introduction

Information extraction is an important research topic in natural language processing. It tries to find relevant information from the large amount of text documents available in digital archives and on the World Wide Web. Research on information extraction has been promoted by the Message Understanding Conferences (1987–1998) and the Automatic Content Extraction program [1]. According to the ACE program, information extraction subsumes a broad range of tasks, including entity detection and tracking, Relation Detection and Characterization (RDC), and event detection and characterization. This paper focuses on the extraction of semantic relations between named entities, as defined by the ACE RDC task, which detects and classifies semantic relationships (usually of predefined types) between pairs of entities. According to the ACE program, an entity is an object or a set of objects, while a relation is an explicitly or implicitly stated relationship between two entities. For example, the sentence "Bill Gates is the chairman and chief software architect of Microsoft Corporation." conveys the ACE-style relation "EMPLOYMENT.exec" between the entities "Bill Gates" (PER, person) and "Microsoft Corporation" (ORG, organization). The extraction of semantic relations between entities can be very useful in many applications such as answering questions (like "Who is the President of the United States?") and retrieving information,(by expanding the term "Barack Obama" to "the President of the United States" via his relationship with "the United States").

Much research has been performed on the extraction of semantic relations between named entities. Feature vector-based methods [8,10,24–28] recast the semantic relation extraction task as a classification problem first by transforming relation instances into multi-dimensional vectors with various features and then by applying machine learning approaches to detect and classify the semantic relationship between the named entities. These researchers have achieved certain success by employing diverse linguistic features, varying from lexical knowledge and entity-related information to syntactic parse trees

* Corresponding author. Tel.: +86 13402679766; fax: +86 512 65241071.
E-mail addresses: gdzhou@suda.edu.cn (G. Zhou), qianlonghua@suda.edu.cn (L. Qian), jxfan@suda.edu.cn (J. Fan).

0020-0255/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2009.12.006

(either constituent- or dependency-based) and semantic information. However, it is rather difficult for them to effectively capture structural parse tree information [23,27,8], which is critical to the improvement of the performance of semantic relation extraction.

As an alternative to feature vector-based methods, tree kernel-based methods provide a good solution to the implicit exploration of structural features by directly computing the similarity between two trees. Earlier researchers [20,6,2] have achieved success in simple tasks but failed in complex ones, such as the ACE RDC task. However, thanks to the pioneering work of Moschitti [13] which employed convolution kernels in a similar relation extraction task called semantic role labeling, tree kernel-based methods have recently progressed rapidly. This process determines the semantic relationship between a predicate and one of its arguments (instead of between named entities). Initially, Sonnenburg et al. [19] and Zhang et al. [22] applied the standard convolution tree kernel [5] and achieved comparable performance with a state-of-the-art linear kernel [25] on the ACE RDC 2003 corpus, using the Shortest Path-enclosed Tree (SPT) structure, that is, the sub-tree enclosed by the shortest path linking two involved entities in the parse tree. It should be noted that the SPT is actually an application of the Predicate Argument Feature (PAF) structure in extracting semantic relations between named entities (instead of between a predicate and one of its arguments), as proposed by Moschitti [13].

However, there are some deficiencies in the present tree kernel-based methods for semantic relation extraction. First, the manner in which they determine a proper tree structure for semantic relation extraction is still problematic. Zhang et al. [21,22] explored five tree structures in semantic relation extraction and were a bit surprised to find that the SPT performed best. However, *this phenomenon is contrary to our understanding of semantic relationship in some sentences.* For example, in the sentence "John and Mary got married", "got married" is critical when we determine the relationship between "John" and "Mary" as shown in Fig. 1e. It is obvious that the information contained in the SPT "John and Mary" is not enough to determine their relationship. Second, the current tree kernels may not be able to adequately capture the structural information in a tree structure. For example, the sub-trees enumerated in the present standard convolution tree kernel (CTK) are context-free. In other words, each sub-tree enumerated in the kernel computation does not consider the contextual information outside the sub-tree. Finally, while it is well known that semantic information plays a critical role in semantic relation extraction, it has not been well studied in the literature [25,27,21,22]. Therefore, this paper is aimed at systematically exploring syntactic and semantic information to determine a more effective method of extracting semantic relations between named entities.

To address the above-mentioned problems in tree kernel-based semantic relation extraction, this paper proposes a new tree kernel method with rich syntactic and semantic information. First, a rich semantic relation tree structure is

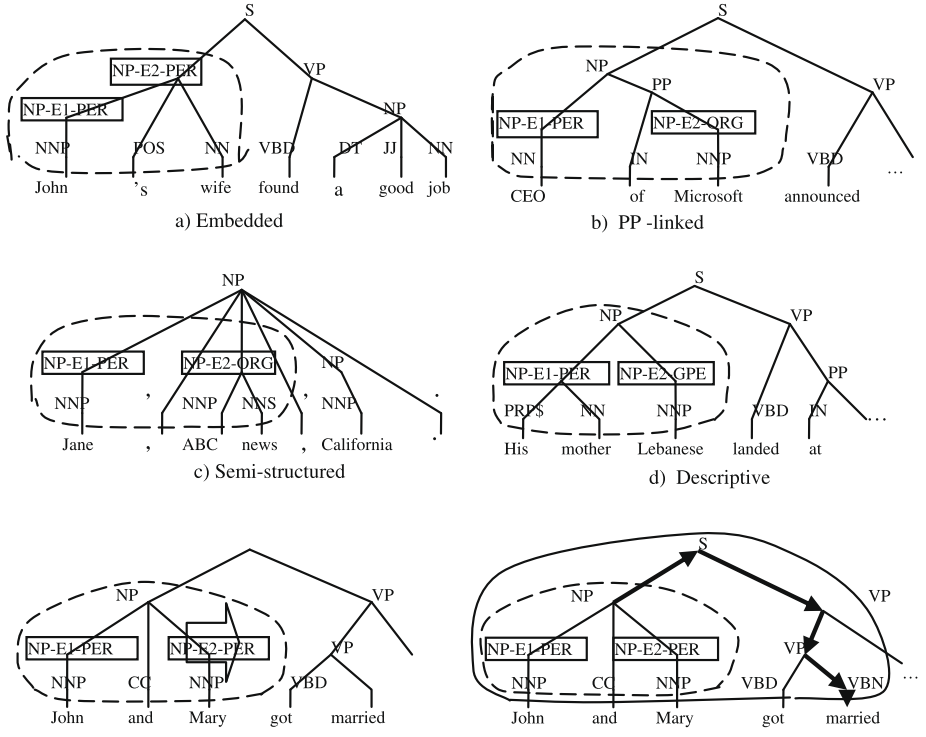

Fig. 1. Parse trees of different structural categories with the SPTs (contained within the dotted circle) and an example of the rich semantic relation tree structure (contained within the solid circle) for the "predicate-linked" category.

automatically determined to better include syntactic and semantic information. Moreover, a context-sensitive CTK, which enumerates both context-free and context-sensitive sub-trees by considering their ancestor node paths as their contexts, is proposed to better capture structural information in the semantic relation tree structure. Finally, our tree kernel and a state-of-the-art linear kernel are interpolated by using a composite kernel to evaluate their complementary nature.

The layout of this paper is as follows: First, related work is reviewed in more detail in Section 2. The rich semantic relation tree structure is then introduced in Section 3, while the context-sensitive CTK is proposed in Section 4. In Section 5, the tree kernel-based semantic relation extraction is systematically evaluated on the ACE RDC corpora. Finally, our work is concluded in Section 6.

2. Related work

Semantic relation extraction was first introduced as part of the Template Element task in the sixth Message Understanding Conference (MUC-6) and then formulated as the template relation task in the seventh Message Understanding Conference (MUC-7). With the introduction of the ACE program, it was further reformulated as the RDC task in the ACE program. Since then, many methods, such as feature vector-based methods [8,10,25–28], tree kernel-based methods [20,6,2,21,22], and composite kernel-based methods [24,21,22] have been proposed in the literature.

For the feature vector-based methods, Kambhatla [10] employed Maximum Entropy models to combine diverse lexical, syntactic and semantic features in semantic relation extraction, and achieved an *F*-measure of 52.8 on the 24 relation subtypes of the ACE RDC 2003 corpus. Zhou et al. [25,27] systematically explored diverse features through a linear kernel and with Support Vector Machines (SVM), and achieved *F*-measures of 68.0 and 55.5 on the five relation types and the 24 relation subtypes of the ACE RDC 2003 corpus, respectively. Zhou et al. [26,28] further improved the performance by exploring the commonality among related classes in a class hierarchy with a hierarchical learning strategy. Jiang and Zhai [8] also systematically evaluated the effectiveness of different feature subspaces with different complexities and obtained the best *F*-measure of 71.5 on the seven relation types of the ACE RDC 2004 corpus. One problem with feature vector-based methods is that they often require extensive feature engineering (e.g. feature design, implementation and selection). Another problem is that although they can explore some structural information in the parse tree, it is difficult to preserve the structural information in the parse trees with the feature vector-based methods (e.g., [10] used the non-terminal path connecting the two given entities in a parse tree, while Zhou et al. [23,27] introduced additional chunking features to enhance the performance).

As an alternative to the feature vector-based methods, the kernel-based methods [7] have been proposed to implicitly explore various features in a high dimensional space by employing a kernel to directly calculate the similarity between two objects. In particular, kernel-based methods can be effective in reducing the burden of feature engineering for structured objects in Natural Language Processing (NLP) research, such as the tree structure in semantic relation extraction.

Zelenko et al. [20] proposed a kernel between two parse trees, which recursively matches nodes from roots to leaves in a top-down manner. For each pair of matched nodes, a subsequent kernel on their child nodes is invoked. They achieved great success in two simple semantic relation extraction tasks. Culotta and Sorensen [6] extended their work to estimate the similarity between augmented dependency trees and achieved an *F*-measure of 45.8 on the five relation types of the ACE RDC 2003 corpus. One problem with the above two tree kernels is that two matched nodes must be at the same height and have the same path to their respective root nodes. Bunescu and Mooney [2] proposed the shortest path dependency tree kernel, which sums up the number of common word classes at each position in the two paths, and achieved an *F*-measure of 52.5 on the five relation types of the ACE RDC 2003 corpus. They argued that the information to model a relationship between two entities could be typically captured by the shortest path between them in the dependency graph. Their kernel is unable to fully preserve the structured dependency tree information, and it is also conditioned by the fact that the two matched paths should have the same length. This makes it suffer from behavior similar to that reported in the work of Culotta and Sorensen [6], that is, high precision but low recall.

To develop an effective tree kernel method, Zhang et al. [21,22] explored various semantic relation tree structures and used the standard CTK over semantic relation trees [5] to model structural information for semantic relation extraction. They achieved *F*-measures of 61.9 and 63.6 on the five relation types of the ACE RDC 2003 corpus and the seven relation types of the ACE RDC 2004 corpus, respectively, without entity-related information, while the *F*-measure on the five relation types of the ACE RDC 2003 corpus reached 68.7 when entity-related information was included in the parse tree. One problem with the standard CTK is that the sub-trees involved in the tree kernel computation are context-free, that is, they do not consider the information outside the sub-trees. This is different from the tree kernel in [6], where the sub-trees involved in the tree kernel computation are context-sensitive (i.e., they consider the path from the tree root node to the sub-tree root node). Zhang et al. [21,22] also showed that the widely-used SPT structure performed best. However, one problem with the SPT is that it fails to capture the contextual information outside the shortest path, yet such information is important for semantic relation extraction in many cases. Our random selection of 100 positive training instances from the ACE RDC 2003 training corpus shows that about 25% of the cases need contextual information outside the shortest path. Among others, Bunescu and Mooney [3] proposed a subsequence kernel and applied it in protein–protein interaction extraction and the ACE RDC tasks. Zhang et al. [23] employed a grammar-driven CTK in semantic role labeling and achieved certain success, following the work pioneered by Moschitti [13].

In order to exploit the advantages of both feature vector-based and tree kernel-based methods, some researchers have turned to composite kernel-based methods. Zhao and Grishman [24] defined several feature vector-based composite kernels to integrate diverse features for semantic relation extraction and achieved an *F*-measure of 70.4 on the seven relation types of the ACE RDC 2004 corpus. Zhang et al. [21] proposed two composite kernels to integrate a linear kernel and the standard CTK. They achieved *F*-measures of 70.9/57.2 on the five relation types/24 relation subtypes of the ACE RDC 2003 corpus and *F*-measures of 72.1/63.6 on the seven relation types/23 relation subtypes of the ACE RDC 2004 corpus.

The above discussion suggests that the parse tree may not have been fully utilized in the previous research, regardless of the approaches they adopted, such as the feature vector-based, tree kernel-based, and composite kernel-based ones. Compared with the previous works, this paper proposes a rich semantic relation tree structure to cover crucial syntactic and semantic information, and a context-sensitive CTK considering both context-free and context-sensitive sub-trees. Furthermore, a composite kernel is applied to combine our tree kernel and a state-of-the-art linear kernel for integrating both flat and structural features in semantic relation extraction, as well as for validating their complementary nature.

3. Rich semantic relation tree structure

The rich semantic relation tree structure is constructed on a default tree structure (i.e., SPT in this paper) in the following three ways:

(1) Contextual expansion: expanding the tree structure to include necessary contextual information.
(2) Structural refinements: Adjusting the tree structure to better preserve structural information.
(3) Semantic expansion: Incorporating various kinds of semantic information into the tree structure.

3.1. Contextual expansion

A semantic relation instance between two entities can often be represented as a syntactic parse tree. Thus, understanding which portion of a parse tree is important is critical in the tree kernel calculation. Zhang et al. [21,22] systematically explored five different semantic relation tree structures, including the Shortest Path-enclosed Tree (SPT) structure and the Context-Sensitive Path-enclosed Tree (CSPT) structure. Here, we borrow the term "context-sensitive" from formal grammar. In particular, the term "context-sensitive" in CSPT means that it is extended to the first left sibling of the node of entity 1 and the first right sibling of the node of entity 2, and that in the case when there is no sibling available, it moves to the parent of the current node and repeats the same process until a sibling is available or the root is reached. They found that the SPT performed best, that is, the SPT outperformed the CSPT. However, this is contrary to our understanding of semantic relationship in some sentences. For example, in the sentence "John and Mary got married...", "got married" is critical when determining the relationship between "John" and "Mary", as shown in Fig. 1e, and the information contained in the SPT "John and Mary" is not enough to determine their semantic relationship. The example shows that the CSPT should have the potential for better performance than the SPT. The reason for the failure of the CSPT may be that it considers only the availability of entities' siblings and fails to consider the following two critical factors:

(1) Is the information contained in the SPT always enough to determine the relationship between two entities occurring in different syntactic structures? In most cases, the SPT is sufficient for semantic relation extraction. For example, "John's wife" is enough to determine the relationship between "John" and "John's wife" in the sentence "John's wife got a good job...", as shown in Fig. 1a. However, the SPT is insufficient in other cases, such as the one in which the two involved entities appear in a coordinated noun phrase. This can be seen from the semantic relationship between "John" and "Mary" in the sentence "John and Mary got married..." (Fig. 1e).
(2) How can the SPT be extended to include necessary contextual information if there is not enough information in the SPT for semantic relation extraction?

To answer the above two questions, we randomly chose 100 positive instances from the ACE RDC 2004 training data and studied their necessary semantic relation trees. We observed that we could classify them into five structural categories: (1) Embedded (20 instances), where one entity is embedded in another entity, for example, "John" and "John's wife", as shown in Fig. 1a; (2) Prepositional Phrase (PP)-linked (28 instances), where one entity is linked to another entity via PP attachment, for example, "CEO" and "Microsoft" in the sentence "CEO of Microsoft announced ...", as shown in Fig. 1b; (3) semi-structured (10 instances), where the sentence consists of a sequence of noun phrases (including the two given entities), for example, "Jane" and "ABC news" in the sentence "Jane, ABC news, California", as shown in Fig. 1c; (4) descriptive (23 instances), for example, the citizenship between "his mother" and "Lebanese" in the sentence "his Lebanese mother landed at ...", as shown in Fig. 1d; and (5) predicate-linked and others (19 instances, including coordinated cases), where the predicate information is necessary to determine the relationship between two entities, for example, "John" and "Mary" in the sentence "John and Mary got married", as shown in Fig. 1e.

Based on the above observations, an algorithm is implemented to determine the necessary semantic relation tree structure for semantic relation extraction. The idea behind the algorithm is that the semantic relation tree structure for a semantic relation should be dynamically determined according to its structural category and context. Given a syntactic parsed tree

and the two considered entities, we first determine the structural category and then extend it accordingly. We then adopt the SPT as the default semantic relation tree structure and only expand the tree structure if it belongs to the "predicate-linked" category. This is based on the observation that semantic relation trees belonging to the "predicate-linked" category vary a great deal syntactically, and the majority (~70%) of them need information outside of the SPT, while it is quite safe (>90%) to use the SPT as the default semantic relation tree structure for the remaining categories. In our algorithm, the expansion is made first by moving up until a predicate-headed phrase is found and then by moving down along the predicate-headed path to the predicate terminal node. Fig. 1e shows an example for the "predicate-linked" category, where the lines with arrows indicate the expansion path.

One issue with the above algorithm is how to determine whether an entity pair belongs to the "predicate-linked" category. In this paper, a simple method is applied by considering the "predicate-linked" category as the default category. That is, the entity pairs that do not belong to the four well-defined and easily-detected categories (i.e., embedded, PP-linked, semi-structured, and descriptive), are classified into the "predicate-linked" category. "Predicate-linked" instances account for only about 20% of cases. This explains why the SPT performs better than the CSPT, as described by Zhang et al. [21]. Consistently adopting the CSPT may introduce unnecessary information in the semantic relation tree structure.

3.2. Structural refinements

After contextual expansion to the widely-used SPT structure, the semantic relation tree structure may contain some unnecessary information and still miss some important contextual information. Based on the observation that different semantic relation types may have different kinds of contextual information and syntactic structures, we explore further to refine the semantic relation tree structure, based on our understanding of semantic relationship between entities in English sentences, in the following ways:

(1) Removing unnecessary components from the tree structure using two heuristics: (a) DEL_ENT_PRE: Removing the constituents (except the headwords) of both entities. This is because the headword of an entity plays a key role in semantic relation extraction, and the other constituents may not be necessary. For example, in the sentence fragment "the families of *seven other former* hostages", "families" (PER) and "hostages" (PER) are the headwords for the two entities. Together with the preposition "of", they uniquely determine the relation type "PER-SOC. Family", while other constituents such as "*seven other former*", can be safely removed. (b) DEL_PATH_ADVP/PP: Removing adverbial/prepositional phrases along the path connecting the two entities. Normally, adverbial phrases along the path imply the way in which two entities relate to each other, while prepositional phrases indicate special relationships, such as location, time, and quantity. Therefore, these phrases are irrelevant to semantic relation extraction and can be safely removed. For example, in the sentence "Pepsi *also* owns Tropicana juices", "*also*" is an adverbial phrase and can be removed. Likewise, in the sentence "Pankhurst moved *with his family* to California", "*with his family*" is a prepositional phrase (PP) and can be removed without affecting the relationship between "Pankhurst" (PER) and "California" (GPE, geographic and political entity).

(2) Compressing coordinated conjunctions into a single one using three heuristics: (a) CMP_NP_CC_NP: Compressing noun phrase-coordinated conjunctions into a single noun phrase. Fig. 2 illustrates the parse trees before and after compression for the sentence fragment "governors from *Connecticut, South Dakota, and* Montana", where a relation "EMP-ORG.Employ-Executive" exists between "governors" (PER) and "Montana" (GPE). (b) CMP_VP_CC_VP: Compressing verb phrase-coordinated conjunctions into a single verb phrase. Similar to noun phrase-coordinated conjunctions, verb phrase-coordinated conjunctions can also be compressed into a single verb phrase. For example, "players had *come from and* gone to leagues" can be compressed into "players had gone to leagues" without

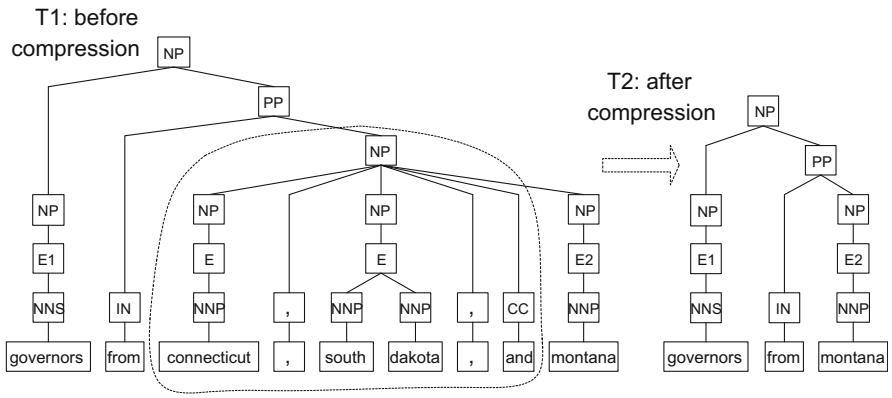

Fig. 2. Compression of noun phrase-coordinated conjunctions.

affecting the "EMP-ORG.Employ-Staff" relationship between "players" (PER) and "leagues" (ORG). (c) CMP_SINGLE_INOUT: Compressing single in-and-out nodes, that is, ignoring the "Y" node from "X → Y → Z" and transforming it into "X → Z."

(3) Recovering useful contextual information outside the two entities using three heuristics: (a) EXP_ENT2_POS: Expanding the possessive structure after the second entity. Normally, the possessive structure after the second entity is important for semantic relation extraction. For example, in the sentence segment "one of the town's two meat-packing plants", where a relation "DISC" exists between "one" (FAC, facility) and "plants" (FAC), the dropping of "'s two meat-packing plants" means that there is a relationship between "one" (FAC) and "town" (GPE), which actually does not exist. Therefore, the possessive structure "'s two meat-packing plants" should be recovered. (b) EXP_ENT2_COREF: Expanding the entity mention co-reference before the second entity. In the case of an appositive phrase "US Destroyer Cole", there is no relation between "US" and "Cole", although "Destroyer" has a co-reference relationship with "Cole" (VEH) and there exists a relation "ART.User-or-Owner" between "US" (GPE) and "Destroyer" (VEH). While the heuristic "DEL_ENT_PRE" may remove "Destroyer" from the appositive phrase "US Destroyer Cole", this rule recovers "Destroyer" to indicate this co-referential information. (c) EXP_ENT1_PrePREP: Expanding the preceding preposition. For example, in the sentence fragment "in Gaza today, Israeli soldiers opened fire..." where the location occurs at the beginning of the sentence as a prepositional phrase and a relation "PHSY.Located" exists between "Gaza" (GPE) and "soldiers" (PER), the preposition "in" is incorrectly omitted from the SPT and should be restored.

3.3. Semantic expansion

The intuition behind semantic expansion is that semantic information imposes strong constraints on semantic relation types. Thus, it should be included in semantic relation extraction. In this paper, we explore several ways to incorporate semantic information into the semantic relation tree structure: added as children of the entity nodes (Fig. 3-T3), embedded in the entity nodes (Fig. 3-T4), and added as children of the top node (Fig. 3-T5). For comparison, we also include the original parse tree (Fig. 3-T1) and the semantic relation tree structure without semantic expansion (Fig. 3-T2).

Moreover, we also systematically explore various kinds of semantic information in semantic relation extraction, such as entity type, entity subtype, entity mention type, entity class, GPE role, entity headword, and the base form of the predicate nearest to the second entity.

4. Context-sensitive convolution tree kernel

Given any form of semantic relation trees—like the one in the last section—we now discuss how to measure the similarity between them with a convolution kernel. A convolution kernel [7] aims to capture common structural information between

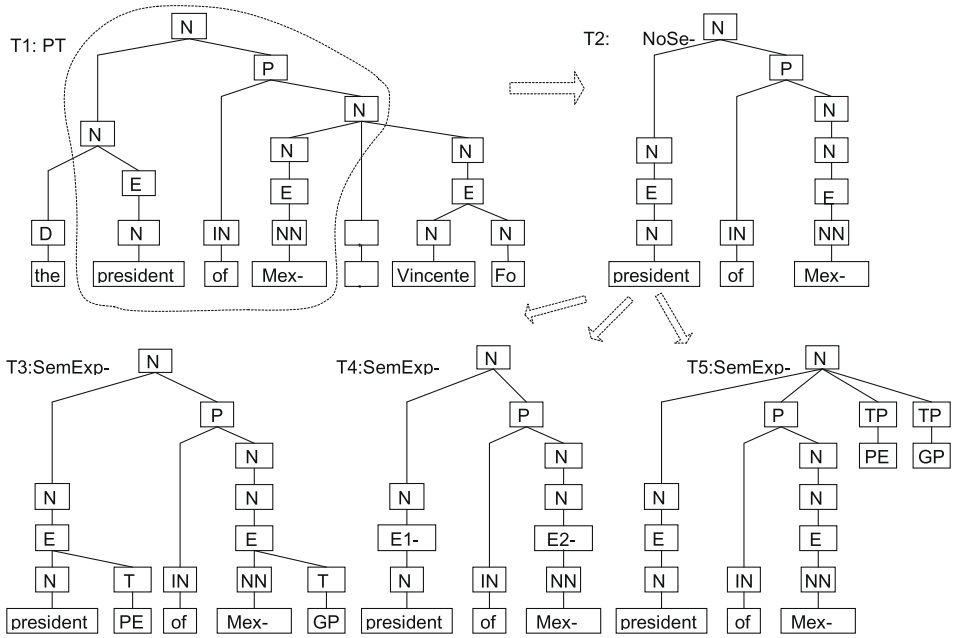

Fig. 3. Different semantic expansion setups for the relation instance "EMP-ORG.Employ-Executive" between the first entity "President" (PER) and the second entity "Mexico" (GPE).

discrete objects in terms of their sub-structures, such as the convolution tree kernel [5], the string kernel [12], and the graph kernel [18]. In particular, Collins and Duffy's convolution tree kernel (CTK) $K_{CTK}(T_1, T_2)$ counts the number of common sub-trees (a specific form of sub-structures) as the structural similarity between two parse trees T_1 and T_2 [5]:

$$K_{CTK}(T_1, T_2) = \sum_{n_1 \in N_1, n_2 \in N_2} \Delta(n_1, n_2), \tag{1}$$

where N_j is the set of nodes in tree T_j, and $\Delta(n_1, n_2)$ evaluates the common sub-trees rooted at n_1 and n_2. Here, each node n encodes the identity of a sub-tree rooted at n and, if there are two nodes in the tree with the same label, the summation will go over both of them. Therefore, $\Delta(n_1, n_2)$ can be computed recursively as follows:

(1) If the context-free productions (Context-Free Grammar [CFG] rules) at n_1 and n_2 are different, $\Delta(n_1, n_2) = 0$; otherwise go to 2.
(2) If both n_1 and n_2 are POS tags, $\Delta(n_1, n_2) = 1 \times \lambda$; otherwise go to 3.
(3) Calculate $\Delta(n_1, n_2)$ recursively as:

$$\Delta(n_1, n_2) = \lambda \prod_{k=1}^{\#ch(n_1)} (1 + \Delta(ch(n_1, k), ch(n_2, k))), \tag{2}$$

where $\#ch(n)$ is the number of children of node n; $ch(n, k)$ is the kth child of node n; and $\lambda (0 < \lambda < 1)$ is the decay factor in order to make the kernel value less variable with respect to different sub-tree sizes.

This standard CTK has been successfully applied in semantic role labeling [13] and semantic relation extraction [21,17]. However, one problem with this tree kernel is: The sub-trees involved in the tree kernel computation are context-free (i.e., they do not consider the information outside the sub-trees). This is in contrast to the context-sensitive tree kernel [6], which considers the path from the tree root node to the sub-tree root node. In order to integrate the advantages of both tree kernels and resolve the problem in the standard CTK, this paper proposes a context-sensitive CTK (CSCTK) by considering the ancestral information of the sub-trees. Again, we borrow the term "context-sensitive" from formal grammar, as we have done in the CSPT. In particular, the term "context-sensitive" in CSCTK means that the tree kernel computation considers the nodes beyond the sub-trees (more precisely, the nodes along the path from the sub-tree root to the tree root), in addition to those within the sub-trees, which are only considered by the default "context-free" example. More formally, our CSCTK can be computed by:

$$K_{CSCTK}(T_1, T_2) = \sum_{\substack{n_1 \in N_1 \\ n_2 \in N_2}} \sum_{i=1}^{m} w_i \cdot \Delta^i(n_1, n_2), \tag{3}$$

where m defines the maximal length of a root node path; and w_i is the weight coefficient for a context-sensitive sub-tree with a root node path of length i. Subsequently, $\Delta^i(n_1, n_2)$ measures the common context-sensitive sub-trees rooted at n_1 and n_2 in relation to path length i. Note that for normalization, the sum of all w_i must be equal to 1.

Slightly different from [5], our tree kernel computes $\Delta^i(n_1, n_2)$ recursively as follows:

(1) If the context-sensitive productions (modified CFG rules with root node paths as their left hand sides, for example, the production of "NP PP → IN NP" with $i = 2$) rooted at n_1 and n_2 are different, return $\Delta^i(n_1, n_2) = 0$; otherwise go to Step 2.
(2) If both n_1 and n_2 are POS tags, then $\Delta^i(n_1, n_2) = \lambda$; otherwise go to Step 3.
(3) Calculate $\Delta^i(n_1, n_2)$ recursively as:

$$\Delta^i(n_1, n_2) = \lambda \prod_{k=1}^{\#ch(n_1)} (1 + \Delta^i(ch(n_1, k), ch(n_2, k)). \tag{4}$$

It is worthwhile to compare our tree kernel with the previous ones. Obviously, our tree kernel is an extension of the standard CTK because it can be degenerated to the standard CTK by setting $m = 1$ and $w_1 = 1$ in Eq. (3). Our tree kernel not only counts the occurrence of each context-free sub-tree, which does not consider its ancestors, but also counts the occurrence of each context-sensitive sub-tree, which considers its ancestors. As a result, our tree kernel is not limited by the constraints in the previous tree kernels (as discussed in Section 2), such as in the examples of [5,21,22,6,2].

Finally, we would like to address the computational issue with our tree kernel. Although our tree kernel considers the context-sensitive sub-trees, it only slightly increases the computational burden, compared with the standard CTK. This is because $\Delta(n_1, n_2) = 0$ holds for the majority of context-free sub-tree pairs [5] and this fact can be fully exploited to speed up the kernel computation [14], as the computation for context-sensitive sub-tree pairs is necessary only when $\Delta^i(n_1, n_2) \neq 0$ and the context-sensitive sub-tree pairs have the same root node path.

5. Experimentation

This paper uses the ACE RDC 2003 and 2004 corpora provided by the Linguistic Data Consortium (LDC) in all experiments.

5.1. Experimental setting

The ACE RDC corpora are gathered from various newspapers, newswires, and broadcasts. In the ACE RDC 2003 corpus, the training set consists of 674 documents and 9683 positive relation instances, while the test set consists of 97 documents and 1386 positive relation instances. The ACE RDC 2003 corpus defines five entity types, five major relation types and 24 relation subtypes. All the reported performances in this paper on the ACE RDC 2003 corpus are evaluated on the test data. The ACE RDC 2004 corpus contains 451 documents and 5702 positive relation instances. It redefines seven entity types, seven major relation types and 23 relation subtypes. For comparison, we use the same experimental setting as employed in [24,19,22], by applying a five-fold cross-validation scheme on a subset (including both the BNEWS and NWIRE domains) of the ACE RDC 2004 data, containing 348 documents and 4400 relation instances. That is, all the reported results in this paper on the ACE RDC 2004 corpus are evaluated using five-fold cross-validation on this subset of the entire corpus.

Both corpora are parsed with the state-of-the-art Charniak parser [4], keeping the boundaries of all the entity mentions. This is accomplished first by representing all entity mentions by their headwords and then by restoring all the entity mentions after syntactic parsing. Please note that the final performance of semantic relation extraction may change greatly with a different range of syntactic parsing errors. We will address this issue in future research. In this study, we iterate over all pairs of entity mentions occurring in the same sentence to generate potential relation instances. Moreover, we only measure the performance of relation extraction on "true" mentions and explicitly model the argument order of the two mentions involved, as described in [23]. In our experimentation, SVM (SVMLight, [9]) is selected as our classifier. For simplicity, we apply the *one versus. others* strategy, which builds K classifiers so as to separate one class from all others, instead of the pair-wise strategy, which builds $K(K-1)/2$ classifiers so as to separate one class from another. Then an instance in the multiple binary classifiers will be assigned to the class with the maximal SVM output. In addition, all the training parameters, including the decay factor λ in formula (2), the path length m, and the weight coefficients w_i $(1 \leqslant i \leqslant m)$ in formula (3) and the forthcoming coefficient α in formula (5), are chosen using a two-fold cross-validation scheme on the ACE RDC 2003 training data. Thereafter, these parameters are applied to the test data of both the ACE RDC 2003 and 2004. The documents in both corpora are supposed to share the same linguistic characteristics regarding semantic relations between named entities because they come from the same news domain. In particular, λ in our tree kernel is fine-tuned to 0.4, in accordance with the work of Zhang et al. (2006). This suggests that a discount of approximately 60% is given as our tree kernel moves down one level in computing $\Delta^i(n_1, n_2)$.

5.2. Experimental results and discussion

Our experiments are organized in the following order:

1) Evaluation of the rich semantic relation tree structure using the standard CTK.
(2) Comparison of the context-sensitive CTK with the standard CTK.
(3) Evaluation of the complementary nature between our tree kernel and a state-of-the-art linear kernel via a composite kernel.
(4) Comparison of our system with the state-of-the-art systems in the literature.

To determine whether an improvement is significant, we also conduct significance testing using paired t-test. In this paper, '$\ggg$', '$\gg$', and '$>$' denote p-values of less than 0.01, 0.01–0.05, and greater than 0.05, which mean significantly better, moderately better, and slightly better, respectively.

5.2.1. Rich semantic relation tree structure

Table 1 presents the contributions of attaching only the entity type information to the widely-used SPT on the seven relation types of the ACE RDC 2004 corpus using the standard CTK. It shows that:

- The convolution tree kernel with the widely-used SPT achieves a performance of 68.3%/51.3%/58.6 in precision/recall/F-measure, respectively. This indicates the effectiveness of the standard CTK and the widely-used SPT in semantic relation extraction.
- Compared with the widely-used SPT, all types0 of SPT + semantics significantly ($\ggg$) improve the F-measure due to an increase in both precision and recall. This indicates that the entity type information is very discriminative for semantic relation extraction.
- Among the three kinds of SPT + semantics, attaching the entity type information as children of the top node performs best, and it significantly ($\ggg$) outperforms the other two alternatives. This may indicate that, with semantic information added as children of the top node, we can have features that are more general because the involved

Table 1
Contributions of attaching semantic information (entity type information only) to the SPT on the seven relation types of the ACE RDC 2004 corpus, using the standard CTK.

Semantic relation tree structure	*P* (%)	*R* (%)	*F*
SPT (Baseline)	68.3	51.3	58.6
SPT + semantics (entity type@BottomNode)	75.3	59.9	66.7 (⋙)
SPT + semantics (entity type@EntityNode)	76.3	59.8	67.1 (⋙)
SPT + semantics (entity type@TopNode)	76.8	62.1	68.6 (⋙)

tree fragments represent general syntactic information as well. That is, attaching the entity type information as children of the top node can generalize a larger sequence of constituents [15,16]; hence, semantic information provides a larger contribution than the other two cases. In the subsequent experiments, we will attach semantic information as children of the top node by default.

Besides the entity type information, we also explore more semantic information in semantic relation extraction, such as entity subtype, entity mention type, entity class, GPE role, entity headword, and the base form of the predicate nearest the second entity. Table 2 evaluates their contributions on the seven relation types of the ACE RDC 2004 corpus by incorporating them incrementally in decreasing order of their potential importance. It shows that our system achieves the best performance with 80.6%/65.1%/72.0 in precision/recall/*F*-measure, respectively. It also shows that:

- Among the explored semantic information, the entity type information is the most effective, and it significantly (⋙) increases the *F*-measure by 10 units. This is because the entity type information imposes strong constraints on relation types.
- Entity subtype information significantly (⋙) improves the *F*-measure by 1.2 units. This further shows that gracefully-defined entity type and subtype information can greatly improve performance.
- Mention level information is also useful, and it significantly (⋙) increases the *F*-measure by 1.6 units.
- It is surprising to observe that entity class, GPE role, and entity headword all decrease the *F*-measure. This suggests that such kinds of semantic information may either cause over-fitting (for entity headword) or be non-discriminative (for entity class and GPE role) and thus fail to achieve a positive effect. Thus, they are excluded from subsequent experiments.
- The predicate verb (in basic form) nearest the second entity moderately (≫) improves the *F*-measure by 0.6 units, largely due to the increase in recall. This suggests that moving verbs from the bottom to the top of the parse tree may be helpful.

Table 3 further presents the contributions of the different structural refinements on the seven relation types of the ACE RDC 2004 corpus by attaching them incrementally. It shows that our system achieves the best performance with 80.1%/69.1%/74.2 in precision/recall/*F*-measure, respectively. It also indicates that:

- Expanding the possessive structure after the second entity moderately (≫) improves the *F*-measure by 0.8 units due to the increase in both precision and recall. This suggests that the possessive structure in the context is very useful, and it should be incorporated.
- Removing the constituents (except the headwords) of the entities slightly (>) improves the *F*-measure by 0.4 units, largely due to the increase in recall (1.6%), even though the precision drops by 1.4%.
- Compressing single in-and-out nodes slightly (>) improves the *F*-measure by 0.4 units, largely due to the increase in recall. This means that proper handling of the nodes with single in-and-out arcs is beneficial to the simplification of the tree structure.

Table 2
Incremental contributions of attaching more semantic information to the SPT on the seven relation types of the ACE RDC 2004 corpus, using the standard CTK. Note: (i) All sorts of semantic information are attached at the top node; (ii) The '+' sign indicates a positive effect and the inclusion of the corresponding semantic information in subsequent experiments while the '−' sign indicates a negative effect and the exclusion of the corresponding semantic information from subsequent experiments.

Semantic relation tree structure	*P* (%)	*R* (%)	*F*
SPT (Baseline)	68.3	51.3	58.6
SPT + semantics(entity type@TopNode) (+)	76.8	62.1	68.6 (⋙)
SPT + semantics (entity subtype@TopNode) (+)	78.6	62.7	69.8 (⋙)
SPT + semantics (mention level@TopNode) (+)	80.4	64.2	71.4 (⋙)
SPT + semantics (entity class@TopNode) (−)	80.8	63.4	71.0
SPT + semantics (GPE role@TopNode) (−)	80.4	63.8	71.1
SPT + semantics (entity headword@TopNode) (−)	81.5	61.9	70.4
SPT + semantics (predicate@TopNode**) (+)**	80.6	65.1	72.0 (≫)

Table 3
Incremental contributions of the different structural refinements on the seven relation types of the ACE RDC 2004, using the standard CTK. Note that the '+' sign indicates a positive effect and the inclusion of the corresponding structural refinement in subsequent experiments while the '−' sign indicates a negative effect and the exclusion of the corresponding structural refinement from subsequent experiments.

Semantic relation tree structure	*P* (%)	*R* (%)	*F*
SPT + semantics (Baseline)	80.6	65.1	72.0
SPT + DEL_ENT_PRE(+)	79.2	66.7	72.4 (>)
SPT + DEL_PATH_PP$^{(-)}$	79.3	66.5	72.4
SPT + DEL_PATH_ADVP$^{(-)}$	79.1	66.5	72.2
SPT + CMP_SINGLE_INOUT(+)	79.2	67.3	72.8 (>)
SPT + CMP_NP_CC_NP(+)	79.0	68.0	73.1 (>)
SPT + CMP_VP_CC_VP$^{(-)}$	78.9	68.1	73.1
SPT + EXP_ENT2_POS(+)	80.0	68.7	73.9 (≫)
SPT + EXP_ENT2_COREF(+)	80.1	69.2	74.2 (>)
SPT + EXP_ENT1_PrePREP$^{(-)}$	80.1	69.2	74.2

- Compressing noun phrase-coordinated conjunctions can simplify the tree structure and slightly (>) improve the performance by 0.3 units in the *F*-measure.
- Expanding co-referential information can recover some useful information and slightly (>) improve the performance by 0.3 units in the *F*-measure.
- The remaining four refinements do not help or even slightly decrease the performance. This may be because these refinements mainly relate to the "predicate-linked" category, indicating the difficulty of detecting and classifying semantic relations from the "predicate-linked" category.

Table 4 presents the contribution of contextual expansion using the standard CTK. It shows that proper contextual expansion significantly (≫) improves the performance by ~1.0 in the *F*-measure. This suggests the usefulness of extending the semantic relation tree beyond the SPT for the "predicate-linked" category. In future research, we will further explore the expansion of the dynamic tree span beyond the SPT for the remaining categories. Table 4 also summarizes the contributions of semantic expansion, structural refinements and contextual expansion in constructing the rich semantic relation tree structure.

Finally, Table 5 presents the contribution of structural refinements and contextual expansion in our rich semantic tree structure over different structural categories on the seven relation types of the ACE RDC 2004 corpus. It also lists the number of test instances for each structural category and its percentage to the total test instances. It shows that:

(1) Structural improvement (i.e., structural refinements and contextual expansion), is useful for different structural categories, especially for "embedded", "semi-structured", and "predicate-linked". This suggests the effectiveness of pruning out noisy information and capturing necessary structural information in semantic relation extraction.
(2) Although the "predicate-linked" category has the worst performance due to the complex syntactic structures and diverse predicate verbs in this category, our structural improvement significantly (≫) enhances the performance by about 10 units in the *F*-measure, largely due to contextual expansion. This suggests the usefulness of contextual expansion and the necessity for better handling of this category in future research.

5.2.2. Context-sensitive convolution tree kernel

Table 6 evaluates the contribution of our context-sensitive CTK, compared with the standard CTK, on the seven relation types of the ACE RDC 2004 corpus, with the above rich semantic relation tree structure. It shows that context-sensitivity significantly (≫) increases the performance by 1.1 units in the *F*-measure, suggesting the usefulness of context-sensitive sub-trees in tree kernel-based semantic relation extraction. Here, m is fine-tuned to 3 while w_1, w_2, and w_3 are fine-tuned to 0.7, 0.2, and 0.1, respectively. This suggests that the parent and grandparent nodes of a sub-tree contain valuable information for semantic relation extraction, while considering further ancestral nodes may not help. The reason may be that although our experimentation on the training data indicates that more than 80% (on average) of sub-trees have a root node path longer

Table 4
Contribution of semantic expansion, structural refinements, and contextual expansion in constructing the rich semantic relation tree structure on the seven relation types of the ACE RDC 2003 corpus, using the standard CTK.

Semantic relation tree structure	*P* (%)	*R* (%)	*F*
SPT (baseline)	68.3	51.3	58.6
SPT + semantics	80.6	65.1	72.0 (≫)
SPT + semantics + structural refinements	80.1	69.2	74.2 (≫)
SPT + semantics + structural refinements + contextual expansion	80.7	70.5	75.2 (≫)

Table 5
Contributions of structural refinements and contextual expansion to our rich semantic relation tree structure over different structural categories on the seven relation types of the ACE RDC 2004 corpus, using the standard CTK.

Structural category	#Test	Percentage (%)	*R* (%)	
			SPT + semantics	SPT + semantics + structural refinements + contextual expansion
Embedded	158	18.2	79.7	84.8 (⋙)
PP-linked	215	24.8	69.3	70.7 (≫)
Descriptive	250	28.8	79.0	80.0 (≫)
Semi-structured	71	8.2	76.1	81.7 (⋙)
Predicate-linked	174	20.0	25.3	35.2 (⋙)

than 3 (as most of the sub-trees are deep from the root node, and more than 90% of the parsed trees in the training data are deeper than six levels), including a root node path longer than 3 may increase vulnerability to syntactic parsing errors and have a negative impact. Moreover, it is interesting to note that the improvement of the performance of employing the rich tree structure, compared with the widely-used SPT, is much greater than that of employing the context-sensitive CTK, compared with the standard CTK. This indicates that semantic relation extraction can greatly benefit from a proper tree structure in tree kernel-based methods.

5.2.3. Composite kernel

Generally, both feature vector-based methods and tree kernel-based methods have their own merits. It is usually easy to build a system using a feature vector-based method and achieve good performance, while tree kernel-based methods hold potential for further performance improvement. Therefore, it is always a good idea to integrate them via a composite kernel. In this paper, a composite kernel K_{COM} via polynomial interpolation, as described in [21,22], is applied to integrate the proposed context-sensitive CTK with a state-of-the-art linear kernel [25,27], which employs a variety of lexical, syntactic, and semantic features, such as word, entity type, mention level, overlap, base phrase chunking, syntactic path tree (either constituent- or dependency-based), and semantic information.

$$K_{\mathrm{COM}}(\langle T_1, F_1\rangle, \langle T_2, F_2\rangle) = \alpha \cdot K^p_{Linear}(F_1, F_2) + (1-\alpha)\cdot K_{CTK}(T_1, T_2). \tag{5}$$

Here, $\langle T_i, F_i\rangle$ denotes the tree structure and feature vector of the relation instance R_i, $K_{Linear}(F_1, F_2)$ and $K_{\mathrm{CTK}}(T_1, T_2)$ indicates the normalized linear kernel and context-sensitive CTK, respectively. $K^p(\bullet,\bullet)$ is the polynomial expansion of $K(\bullet,\bullet)$ with degree d=2, that is, $K^p(\bullet,\bullet) = (K(\bullet,\bullet)+1)^2$; and α is the weight coefficient (α is set to 0.3 using two-fold cross-validation on the ACE 2003 training data).

Table 7 presents the performance of the composite kernel. It shows that the composite kernel achieves an *F*-measure of 77.8 on the seven relation types of the ACE RDC 2004 corpus and significantly (⋙) outperforms both the state-of-the-art linear kernel and our tree kernel. This suggests that our tree kernel and the state-of-the-art linear kernel are quite complementary, and that the composite kernel can effectively integrate both flat and structural features.

5.2.4. Comparison with other systems

Finally, Tables 8 and 9 compare our system with other state-of-the-art systems on the ACE RDC 2003 and 2004 corpora, respectively. It is worth mentioning that all of the state-of-the-art systems apply certain kinds of entity-related information. This is not surprising as our experiments also show that using entity-related information results in a large performance improvement. Tables 8 and 9 show that our tree kernel-based system greatly outperforms the previous tree kernel-based systems. This is largely due to (i) the rich semantic relation tree structure, which incorporates necessary syntactic and semantic information; and (ii) the context-sensitive nature of our tree kernel, which overcomes the limitations of previous tree kernels. The tables also show that our tree kernel-based system outperforms the state-of-the-art feature vector-based systems. This proves the great potential for semantic relation extraction inherent in the tree structure, even though the total time for learning a tree kernel-based system is usually about 10 times longer than that required to learn a feature vector-based example, according to our empirical statistics. Finally, the tables also show that our composite kernel-based system outperforms all the others.

Table 6
Comparison of the context-sensitive CTK with the standard CTK on the seven relation types of the ACE RDC 2004 corpus, using the rich semantic relation tree structure.

Tree kernel	*P* (%)	*R* (%)	*F*
CTK (baseline)	80.7	70.5	75.2
Context-sensitive CTK	81.4	71.8	76.3 (⋙)

Table 7
Performance of the composite kernel on the seven relation types of the ACE RDC 2004 corpus via polynomial interpolation.

System	*P* (%)	*R* (%)	*F*
Linear kernel	78.2	63.4	70.1
Context-sensitive CTK	81.4	71.8	76.3
Composite kernel	82.7	73.5	77.8 (⋙)

Table 8
Comparison of the different systems on the ACE RDC 2003 corpus over the five relation types (outside the parentheses) and the 24 relation subtypes (inside the parentheses).

ACE RDC 2003	*P* (%)	*R* (%)	*F*
Ours: composite kernel	82.3 (66.3)	70.1 (56.4)	75.7 (60.9)
Zhang et al. [21,22]: composite kernel	77.3 (64.9)	65.6 (51.2)	70.9 (57.2)
Ours: context-sensitive CTK	82.1 (65.2)	67.2 (54.3)	73.9 (59.2)
Zhang et al. [21,22]: standard CTK	76.1 (62.4)	62.6 (48.5)	68.7 (54.6)
Bunescu and Mooney [2]: shortest path kernel	65.5 (–)	43.8 (–)	52.5 (–)
Culotta and Sorensen [6]: dependency kernel	67.1 (–)	35.0 (–)	45.8 (–)
Zhou et al. [25,27]: feature vector-based	77.2 (63.1)	60.7 (49.5)	68.0 (55.5)
Kambhatla [10]: feature vector-based	– (63.5)	– (45.2)	– (52.8)

Table 9
Comparison of the different systems on the ACE RDC 2004 corpus over the seven relation types (outside the parentheses) and the 23 relation subtypes (inside the parentheses).

ACE RDC 2004	*P* (%)	*R* (%)	*F*
Ours: composite kernel	83.1 (71.2)	73.5 (64.2)	77.8 (67.5)
Zhang et al. [21,22]: composite kernel	76.1 (68.6)	68.4 (59.3)	72.1 (63.6)
Zhao and Grishman [24]:[a] composite kernel	69.2 (–)	70.5 (–)	70.4 (–)
Ours: context-sensitive CTK	81.4 (69.8)	71.8 (63.3)	76.3 (66.4)
Zhang et al. [21,22]: standard CTK	72.5 (–)	56.7 (–)	63.6 (–)

[a] There may be some typing errors for the performance reported in [22] because *P*, *R*, and *F* do not match.

6. Conclusion

The use of structural information holds great potential for semantic relation extraction. This paper proposes a novel tree kernel-based method to resolve critical problems in previous tree kernel-based semantic relation extraction via a rich semantic relation tree structure and a context-sensitive CTK. Moreover, this paper evaluates the complementary nature between our tree kernel and a state-of-the-art linear kernel. The evaluation on the ACE RDC 2004 corpora shows that:

(1) Semantic information is very discriminative in semantic relation extraction and can be well incorporated via the rich semantic relation tree structure. Among all the types of explored semantic information, entity type information provides the greatest contribution. Moreover, it is interesting to note that when semantic information is placed higher in the tree structure, better performance is achieved.
(2) Proper structural refinements and contextual expansion are very useful.
(3) Semantic relations in the "predicate-linked" structural category are normally difficult to extract and given their popularity, they are thus worth exploring in the future research.
(4) Incorporating context-sensitivity into the standard CTK greatly improves its performance.
(5) Feature vector-based and tree kernel-based methods complement each other well. The composite kernel via polynomial interpolation well combines our tree kernel and a state-of-the-art linear kernel to integrate both flat and structural features in semantic relation extraction.

The contribution of this paper lies in two aspects: (1) Properly unifying various kinds of syntactic and semantic information into a single tree structure; and (2) Capturing such varieties via a novel context-sensitive convolution tree kernel. To our knowledge, this is the first research to demonstrate that, without extensive feature engineering, a tree kernel method can achieve much better performance than the state-of-the-art feature vector-based methods in semantic relation extraction. This shows the great potential of structural information in semantic relation extraction. Our approach takes a big stride in the right direction.

In future research, we will investigate more on semantic information in the semantic relation tree structure and explore new tree kernels so as to better capture syntactic and semantic information. In particular, the convolution tree kernel, as proposed in

this paper, maps a parse tree structure into the sub-tree (a specific form of sub-structure) space and computes the similarity between two parse trees by counting the number of common sub-trees. Therefore, it would be beneficial to automatically determining the crucial sub-structure space in effectively representing the parse tree structure. This can be done by employing locality-preserving projection-based manifold learning methods, such as the kernel class-wise example, in leveraging both local information and class information [11]. Moreover, we will study how to resolve data imbalance and sparseness issues from the viewpoint of the learning algorithm. Finally, the weight coefficients in both the context-sensitive convolution tree kernel and the kernel composition are chosen by cross-validation, which is only feasible for a small number of components. Therefore, it will be worthwhile to explore automatic coefficient learning such as multiple kernel learning [19].

Acknowledgements

This research is supported by Projects 60673041, 60873150, 60970056 and 90920004 under the National Natural Science Foundation of China and Project 2006AA01Z147 under the "863" National High-Tech Research and Development of China.

References

[1] ACE (1999–2008). Automatic Content Extraction. <http://www.ldc.upenn.edu/Projects/ACE/>.
[2] R. Bunescu, R.J. Mooney, A shortest path dependency kernel for relation extraction, in: Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing(HLT/EMNLP'2005), 6–8 October 2005, Vancouver, BC, 2005, pp. 724–731.
[3] R. Bunescu, R.J. Mooney, Subsequence kernels for relation extraction, in: Proceedings of Annual Conference on Neural Information Processing Systems (NIPS'2005), Vancouver, BC, December 2005, pp. 171–178.
[4] E. Charniak, Immediate-head parsing for language models. in: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'2001), 9–11 July 2001, Toulouse, France, 2001, pp. 129–137.
[5] M. Collins, N. Duffy, Convolution kernels for natural language, in: Proceedings of Annual Conference on Neural Information Processing Systems (NIPS'2001), Cambridge, MA, 2001, pp. 625–632.
[6] A. Culotta, J. Sorensen, Dependency tree kernels for relation extraction, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'2004), 21–26 July 2004, Barcelona, Spain, 2004, pp. 423–429.
[7] D. Haussler, Convolution kernels on discrete structures, Technical Report UCS-CRL-99-10, University of California, Santa Cruz, 1999.
[8] J. Jiang, C.X. Zhai, A systematic exploration of the feature space for relation extraction, in: Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT'2007), Rochester, NY, USA, 2007, pp. 113–120.
[9] T. Joachims, Text Categorization with Support Vector Machine: learning with many relevant features, in: Proceedings of European Conference on Machine Learning (ECML'1998), 21–24 April 1998, Chemnitz, Germany, 1998, pp. 137–142.
[10] N. Kambhatla, Combining lexical, syntactic and semantic features with Maximum Entropy models for extracting relations, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'2004) (Poster), 21–26 July 2004, Barcelona, Spain, 2004, pp. 178–181.
[11] J.B. Li, J.S. Pan, S.C. Chu, Kernel class-wise locality preserving projection, Information Sciences 178 (7) (2008) 1825–1835.
[12] H. Lodhi, C. Saunders, J. Shaw-Taylor, N. Cristianini, C. Watkins, Text classification using string kernel, Journal of Machine Learning Research 2002 (2) (2002) 419–444.
[13] A. Moschitti, A study on convolution kernels for shallow semantic parsing, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'2004), 21–26 July 2004, Barcelona, Spain, 2004.
[14] A. Moschitti, Making tree kernels practical for natural language learning, in: Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL'2006), 3–7 April 2006, Trento, Italy, 2006, pp. 113–120.
[15] A. Moschitti, D. Pighin, R. Basili, Tree kernel engineering in semantic role labeling systems, in: Proceedings of the Workshop on Learning Structured Information for Natural Language Applications, the Conference of the European Chapter of the Association for Computational Linguistics (EACL'2006), 3–7 April 2006, Trento, Italy, 2006, pp. 49–56.
[16] A. Moschitti, D. Pighin, R. Basili, Tree kernels for semantic role labeling, special issue on semantic role labeling, Computational Linguistics 34 (2) (2008) 194–224.
[17] L.H. Qian, G.D. Zhou, Q.M. Zhu. P.D. Qian, Exploiting constituent dependencies for tree kernel-based semantic relation extraction, in: Proceedings of International Conference on Computational Linguistics (COLING'2008), 18–22 August 2008, Manchester, UK, 2008, pp. 697–704.
[18] J. Suzuki, T. Hirao, Y. Sasaki, E. Maeda, Hierarchical directed acyclic graph kernel: methods for structured natural language data, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'2003), 7–12 July 2003, Sapporo, Japan, 2003, pp. 32–39.
[19] S. Sonnenburg, G. Raetsch, C. Schaefer, B. Schoelkopf, Large scale multiple kernel learning, Journal of Machine Learning Research 7 (2006) 1531–1565.
[20] D. Zelenko, C. Aone, Richardella, Kernel-based methods for relation extraction, Journal of Machine Learning Research 3 (February) (2003) 1083–1106.
[21] M. Zhang, J. Zhang, J. Su, G.D. Zhou, A composite kernel to extract relations between entities with both flat and structured features, in: Proceedings of the International Conference on Computational Linguistics and the Annual Meeting of the Association for Computational Linguistics (COLING-ACL'2006), 17–21 July 2006, Sydney, Australia, 2006, pp. 825–832.
[22] M. Zhang, G.D. Zhou, A.T. Aw, Exploring syntactic structured features over parse trees for relation extraction using kernel methods, Information Processing and Management 44 (2008) (2008) 687–701.
[23] M. Zhang, W.X. Che, G.D. Zhou, A.T. Aw, C.L. Tan, T. Liu, S. Li, Semantic role labeling using a grammar-driven convolution tree kernel, IEEE Transaction on Audio, Speech and Language Processing 16 (7) (2008) 1315–1329.
[24] S.B. Zhao, R. Grishman, Extracting relations with integrated information using kernel-based methods, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'2005), 25–30 June 2005, University of Michigan-Ann Arbor, USA, 2005, pp. 419–426.
[25] G.D. Zhou, J. Su, J. Zhang, M. Zhang, Exploring various knowledge in relation extraction, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'2005), 25–30 June 2005, Ann Arbor, Michgan, USA, 2005, pp. 427–434.
[26] G.D. Zhou, J. Su, M. Zhang, Modeling commonality among related classes in relation extraction, in: Proceedings of the International Conference on Computational Linguistics and the Annual Meeting of the Association for Computational Linguistics (COLING-ACL'2006), 17–21 July 2006, Sydney, Australia, 2006, pp. 121–128.
[27] G.D. Zhou, M. Zhang, Extracting relation information from text documents by exploring various types of knowledge, Information Processing and Management 43 (2007) (2007) 969–982.
[28] G.D. Zhou, M. Zhang, D.H. Ji, Q.M. Zhu, Hierarchical learning strategy in semantic relation extraction, Information Processing and Management 44 (2008) (2008) 1008–1021.