Exploiting Constituent Dependencies for Tree Kernel-based Semantic Relation Extraction

Longhua Qian Guodong Zhou Fang Kong Qiaoming Zhu Peide Qian
Jiangsu Provincial Key Lab for Computer Information Processing Technology
School of Computer Science and Technology, Soochow University
1 Shizi Street, Suzhou, China 215006
{qianlonghua,gdzhou,kongfang,qmzhu,pdqian}@suda.edu.cn

Abstract

This paper proposes a new approach to dynamically determine the tree span for tree kernel-based semantic relation extraction. It exploits constituent dependencies to keep the nodes and their head children along the path connecting the two entities, while removing the noisy information from the syntactic parse tree, eventually leading to a dynamic syntactic parse tree. This paper also explores entity features and their combined features in a unified parse and semantic tree, which integrates both structured syntactic parse information and entity-related semantic information. Evaluation on the ACE RDC 2004 corpus shows that our dynamic syntactic parse tree outperforms all previous tree spans, and the composite kernel combining this tree kernel with a linear state-of-the-art feature-based kernel, achieves the so far best performance.

1 Introduction

Information extraction is one of the key tasks in natural language processing. It attempts to identify relevant information from a large amount of natural language text documents. Of three subtasks defined by the ACE program[1], this paper focuses exclusively on Relation Detection and Characterization (RDC) task, which detects and classifies semantic relationships between predefined types of entities in the ACE corpus. For example, the sentence "*Microsoft Corp.* is based in *Redmond,* WA" conveys the relation "GPE-AFF.Based" between "*Microsoft Corp.*" [ORG] and "*Redmond*" [GPE]. Due to limited accuracy in state-of-the-art syntactic and semantic parsing, reliably extracting semantic relationships between named entities in natural language documents is still a difficult, unresolved problem.

In the literature, feature-based methods have dominated the research in semantic relation extraction. Featured-based methods achieve promising performance and competitive efficiency by transforming a relation example into a set of syntactic and semantic features, such as lexical knowledge, entity-related information, syntactic parse trees and deep semantic information. However, detailed research (Zhou et al., 2005) shows that it's difficult to extract new effective features to further improve the extraction accuracy. Therefore, researchers turn to kernel-based methods, which avoids the burden of feature engineering through computing the similarity of two discrete objects (e.g. parse trees) directly. From prior work (Zelenko et al., 2003; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005) to current research (Zhang et al., 2006; Zhou et al., 2007), kernel methods have been showing more and more potential in relation extraction.

The key problem for kernel methods on relation extraction is how to represent and capture the structured syntactic information inherent in relation instances. While kernel methods using the dependency tree (Culotta and Sorensen, 2004) and the shortest dependency path (Bunescu and Mooney, 2005) suffer from low recall performance, convolution tree kernels (Zhang et al., 2006; Zhou et al., 2007) over syntactic parse trees achieve comparable or even better performance than feature-based methods.

However, there still exist two problems regarding currently widely used tree spans. Zhang et al. (2006) discover that the Shortest Path-

[1] http://www.ldc.upenn.edu/Projects/ACE/

enclosed Tree (SPT) achieves the best performance. Zhou et al. (2007) further extend it to Context-Sensitive Shortest Path-enclosed Tree (CS-SPT), which dynamically includes necessary predicate-linked path information. One problem with both SPT and CS-SPT is that they may still contain unnecessary information. The other problem is that a considerable number of useful context-sensitive information is also missing from SPT/CS-SPT, although CS-SPT includes some contextual information relating to predicate-linked path.

This paper proposes a new approach to dynamically determine the tree span for relation extraction by exploiting constituent dependencies to remove the noisy information, as well as keep the necessary information in the parse tree. Our motivation is to integrate dependency information, which has been proven very useful to relation extraction, with the structured syntactic information to construct a concise and effective tree span specifically targeted for relation extraction. Moreover, we also explore interesting combined entity features for relation extraction via a unified parse and semantic tree.

The other sections in this paper are organized as follows. Previous work is first reviewed in Section 2. Then, Section 3 proposes a dynamic syntactic parse tree while the entity-related semantic tree is described in Section 4. Evaluation on the ACE RDC corpus is given in Section 5. Finally, we conclude our work in Section 6.

2 Related Work

Due to space limitation, here we only review kernel-based methods used in relation extraction. For those interested in feature-based methods, please refer to Zhou et al. (2005) for more details.

Zelenko et al. (2003) described a kernel between shallow parse trees to extract semantic relations, where a relation instance is transformed into the least common sub-tree connecting the two entity nodes. The kernel matches the nodes of two corresponding sub-trees from roots to leaf nodes recursively layer by layer in a top-down manner. Their method shows successful results on two simple extraction tasks. Culotta and Sorensen (2004) proposed a slightly generalized version of this kernel between dependency trees, in which a successful match of two relation instances requires the nodes to be at the same layer and in the identical path starting from the roots to the current nodes. These strong constraints make their kernel yield high precision but very low recall on the ACE RDC 2003 corpus. Bunescu and Mooney (2005) develop a shortest path dependency tree kernel, which simply counts the number of common word classes at each node in the shortest paths between two entities in dependency trees. Similar to Culotta and Sorensen (2004), this method also suffers from high precision but low recall.

Zhang et al. (2006) describe a convolution tree kernel (CTK, Collins and Duffy, 2001) to investigate various structured information for relation extraction and find that the Shortest Path-enclosed Tree (SPT) achieves the F-measure of 67.7 on the 7 relation types of the ACE RDC 2004 corpus. One problem with SPT is that it loses the contextual information outside SPT, which is usually critical for relation extraction. Zhou et al. (2007) point out that both SPT and the convolution tree kernel are context-free. They expand SPT to CS-SPT by dynamically including necessary predicate-linked path information and extending the standard CTK to context-sensitive CTK, obtaining the F-measure of 73.2 on the 7 relation types of the ACE RDC 2004 corpus. However, the CS-SPT only recovers part of contextual information and may contain noisy information as much as SPT.

In order to fully utilize the advantages of feature-based methods and kernel-based methods, researchers turn to composite kernel methods. Zhao and Grishman (2005) define several feature-based composite kernels to capture diverse linguistic knowledge and achieve the F-measure of 70.4 on the 7 relation types in the ACE RDC 2004 corpus. Zhang et al. (2006) design a composite kernel consisting of an entity linear kernel and a standard CTK, obtaining the F-measure of 72.1 on the 7 relation types in the ACE RDC 2004 corpus. Zhou et al. (2007) describe a composite kernel to integrate a context-sensitive CTK and a state-of-the-art linear kernel. It achieves the so far best F-measure of 75.8 on the 7 relation types in the ACE RDC 2004 corpus.

In this paper, we will further study how to dynamically determine a concise and effective tree span for a relation instance by exploiting constituent dependencies inherent in the parse tree derivation. We also attempt to fully capture both the structured syntactic parse information and entity-related semantic information, especially combined entity features, via a unified parse and semantic tree. Finally, we validate the effectiveness of a composite kernel for relation extraction, which combines a tree kernel and a linear kernel.

3 Dynamic Syntactic Parse Tree

This section discusses how to generate dynamic syntactic parse tree by employing constituent dependencies to overcome the problems existing in currently used tree spans.

3.1 Constituent Dependencies in Parse Tree

Zhang et al. (2006) explore five kinds of tree spans and find that the Shortest Path-enclosed Tree (SPT) achieves the best performance. Zhou et al. (2007) further propose Context-Sensitive SPT (CS-SPT), which can dynamically determine the tree span by extending the necessary predicate-linked path information outside SPT. However, the key problem of how to represent the structured syntactic parse tree is still partially resolved. As we indicate as follows, current tree spans suffer from two problems:

(1) Both SPT and CS-SPT still contain unnecessary information. For example, in the sentence "…bought *one* of town's two meat-packing *plants*", the condensed information "*one* of *plants*" is sufficient to determine "DISC" relationship between the entities "*one*" [FAC] and "*plants*" [FAC], while SPT/CS-SPT include the redundant underlined part. Therefore more unnecessary information can be safely removed from SPT/CS-SPT.

(2) CS-SPT only captures part of context-sensitive information relating to predicate-linked structure (Zhou et al., 2007) and still loses much context-sensitive information. Let's take the same example sentence "…bought *one* of *town*'s two meat-packing plants", where indeed there is no relationship between the entities "*one*" [FAC] and "*town*" [GPE]. Nevertheless, the information contained in SPT/CS-SPT ("*one* of *town*") may easily lead to their relationship being misclassified as "DISC", which is beyond our expectation. Therefore the underlined part outside SPT/CS-SPT should be recovered so as to differentiate it from positive instances.

Since dependency plays a key role in many NLP problems such as syntactic parsing, semantic role labeling as well as semantic relation extraction, our motivation is to exploit dependency knowledge to distinguish the necessary evidence from the unnecessary information in the structured syntactic parse tree.

On one hand, lexical or word-word dependency indicates the relationship among words occurring in the same sentence, e.g. predicate-argument dependency means that arguments are dependent on their target predicates, modifier-head dependency means that modifiers are dependent on their head words. This dependency relationship offers a very condensed representation of the information needed to assess the relationship in the forms of the dependency tree (Culotta and Sorensen, 2004) or the shortest dependency path (Bunescu and Mooney, 2005) that includes both entities.

On the other hand, when the parse tree corresponding to the sentence is derived using derivation rules from the bottom to the top, the word-word dependencies extend upward, making a unique head child containing the head word for every non-terminal constituent. As indicated as follows, each CFG rule has the form:

$P \rightarrow L_n \ldots L_1 \boldsymbol{H} R_1 \ldots R_m$

Here, P is the parent node, $\boldsymbol{H}$ is the head child of the rule, $L_n \ldots L_1$ and $R_1 \ldots R_m$ are left and right modifiers of $\boldsymbol{H}$ respectively, and both n and m may be zero. In other words, the parent node P depends on the head child $\boldsymbol{H}$, this is what we call *constituent dependency*. Vice versa, we can also determine the head child of a constituent in terms of constituent dependency. Our hypothesis stipulates that the contribution of the parse tree to establishing a relationship is almost exclusively concentrated in the path connecting the two entities, as well as the head children of constituent nodes along this path.

3.2 Generation of Dynamic Syntactic Parse Tree

Starting from the Minimum Complete Tree (MCT, the complete sub-tree rooted by the nearest common ancestor of the two entities under consideration) as the representation of each relation instance, along the path connecting two entities, the head child of every node is found according to various constituent dependencies. Then the path nodes and their head children are kept while any other nodes are removed from the tree. Eventually we arrive at a tree called Dynamic Syntactic Parse Tree (DSPT), which is dynamically determined by constituent dependencies and only contains necessary information as expected.

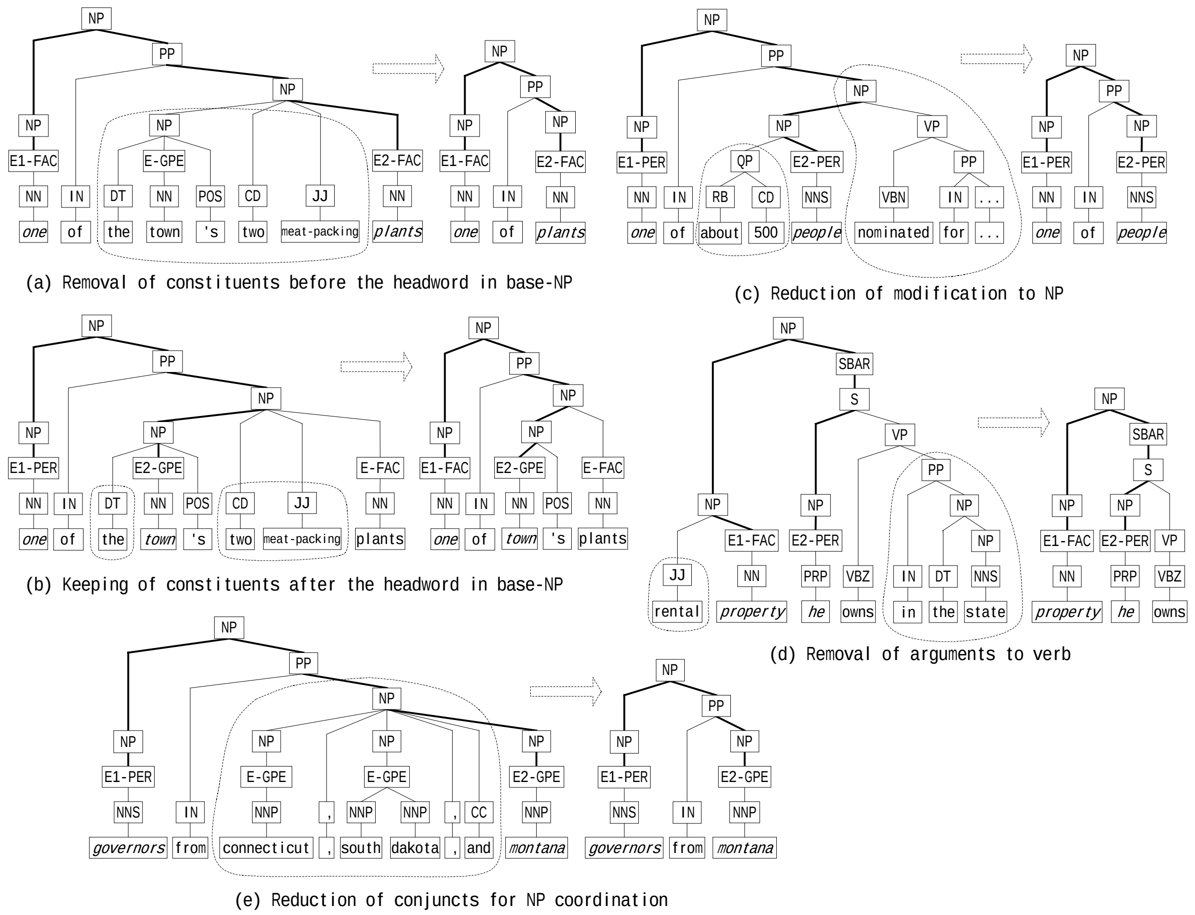
There exist a considerable number of constituent dependencies in CFG as described by Collins (2003). However, since our task is to extract the relationship between two named entities, our focus is on how to condense Noun-Phrases (NPs) and other useful constituents for relation extraction. Therefore constituent dependencies can be classified according to constituent types of the CFG rules:

(1) **Modification within base-NPs**: base-NPs mean that they do not directly dominate an NP themselves, unless the dominated NP is a possessive NP. The noun phrase right above the entity headword, whose mention type is nominal or name, can be categorized into this type. In this case, the entity headword is also the headword of the noun phrase, thus all the constituents before the headword are dependent on the headword, and may be removed from the parse tree, while the headword and the constituents right after the headword remain unchanged. For example, in the sentence "…bought *one* of town's two meat-packing *plants*" as illustrated in Figure 1(a), the constituents before the headword "*plants*" can be removed from the parse tree. In this way the parse tree "*one* of *plants*" could capture the "DISC" relationship more concisely and precisely. Another interesting example is shown in Figure 1(b), where the base-NP of the second entity "*town*" is a possessive NP and there is no relationship between the entities "*one*" and "*town*" defined in the ACE corpus. For both SPT and CS-SPT, this example would be condensed to "*one* of *town"* and therefore easily misclassified as the "DISC" relationship between the two entities. In the contrast, our DSPT can avoid this problem by keeping the constituent "'s" and the headword "*plants*".

(2) **Modification to NPs**: except base-NPs, other modification to NPs can be classified into this type. Usually these NPs are recursive, meaning that they contain another NP as their child. The CFG rules corresponding to these modifications may have the following forms:

NP* → *NP *SBAR*	[relative clause]
NP* → *NP *VP*	[reduced relative]
NP* → *NP *PP*	[PP attachment]

Here, the NPs in bold mean that the path connecting the two entities passes through them. For every right hand side, the NP in bold is modified by the constituent following them. That is, the latter is dependent on the former, and may be reduced to a single NP. In Figure 1(c) we show a sentence "*one* of about 500 *people* nominated for …", where there exists a "DISC" relationship between the entities "*one*" and "*people*". Since the reduced relative "nominated for …" modifies and is therefore dependent on the "*people*", they can be removed from the parse tree, that is, the right side ("NP VP") can be reduced to the left hand side, which is exactly a single NP.



(a) Removal of constituents before the headword in base-NP

(b) Keeping of constituents after the headword in base-NP

(c) Reduction of modification to NP

(d) Removal of arguments to verb

(e) Reduction of conjuncts for NP coordination

Figure 1. Removal and reduction of constituents using dependencies

(3) **Arguments/adjuncts to verbs**: this type includes the CFG rules in which the left side includes S, SBAR or VP. An argument represents the subject or object of a verb, while an adjunct indicates the location, date/time or way of the action corresponding to the verb. They depend on the verb and can be removed if they are not included in the path connecting the two entities. However, when the parent tag is S or SBAR, and its child VP is not included in the path, this VP should be recovered to indicate the predicate verb. Figure 1(d) shows a sentence "… maintain rental *property he* owns in the state", where the "ART.User-or-Owner" relation holds between the entities "*property*" and "*he*". While PP can be removed from the rule ("VP→ VBZ PP"), the VP should be kept in the rule ("S→ NP VP"). Consequently, the tree span looks more concise and precise for relation extraction.

(4) **Coordination conjunctions**: In coordination constructions, several peer conjuncts may be reduced into a single constituent. Although the first conjunct is always considered as the headword (Collins, 2003), actually all the conjuncts play an equal role in relation extraction. As illustrated in Figure 1(e), the NP coordination in the sentence ("*governors* from connecticut, south dakota, and *montana*") can be reduced to a single NP ("*governors* from *montana*") by keeping the conjunct in the path while removing the other conjuncts.

(5) **Modification to other constituents**: except for the above four types, other CFG rules fall into this type, such as modification to PP, ADVP and PRN etc. These cases are similar to arguments/adjuncts to verbs, but less frequent than them, so we will not detail this scenario.

In fact, SPT (Zhang et al., 2006) can be arrived at by carrying out part of the above removal operations using a single rule (i.e. all the constituents outside the linking path should be removed) and CS-CSPT (Zhou et al., 2007) further recovers part of necessary context-sensitive information outside SPT, this justifies that SPT performs well, while CS-SPT outperforms SPT.

4 Entity-related Semantic Tree

Entity semantic features, such as entity headword, entity type and subtype etc., impose a strong constraint on relation types in terms of relation definition by the ACE RDC task. Experiments by Zhang et al. (2006) show that linear kernel using only entity features contributes much when combined with the convolution parse tree kernel. Qian et al. (2007) further indicates that among these entity features, entity type, subtype, and mention type, as well as the base form of predicate verb, contribute most while the contribution of other features, such as entity class, headword and GPE role, can be ignored.

In order to effectively capture entity-related semantic features, and their combined features as well, especially bi-gram or tri-gram features, we build an Entity-related Semantic Tree (EST) in three ways as illustrated in Figure 2. In the example sentence "*they* 're *here*", which is excerpted from the ACE RDC 2004 corpus, there exists a relationship "Physical.Located" between the entities "*they*" [PER] and "*here*" [GPE.Population-Center]. The features are encoded as "TP", "ST", "MT" and "PVB", which denote type, subtype, mention-type of the two entities, and the base form of predicate verb if existing (nearest to the 2nd entity along the path connecting the two entities) respectively. For example, the tag "TP1" represents the type of the 1st entity, and the tag "ST2" represents the subtype of the 2nd entity. The three entity-related semantic tree setups are depicted as follows:

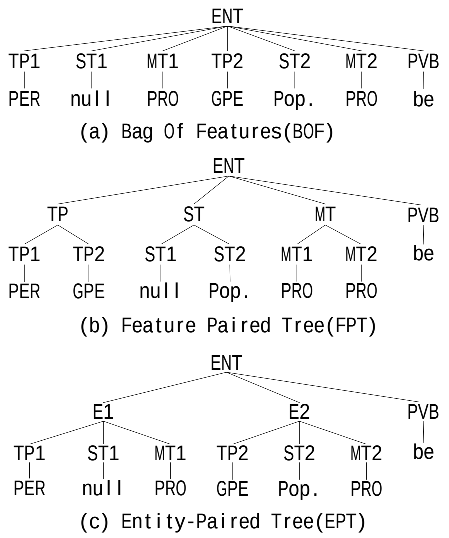

ENT
TP1 ST1 MT1 TP2 ST2 MT2 PVB
PER null PRO GPE Pop. PRO be
(a) Bag Of Features(BOF)
ENT
TP ST MT PVB
TP1 TP2 ST1 ST2 MT1 MT2 be
PER GPE null Pop. PRO PRO
(b) Feature Paired Tree(FPT)
ENT
E1 E2 PVB
TP1 ST1 MT1 TP2 ST2 MT2 be
PER null PRO GPE Pop. PRO
(c) Entity-Paired Tree(EPT)

Figure 2. Different setups for entity-related semantic tree (EST)

(a) Bag of Features (BOF, e.g. Fig. 2(a)): all feature nodes uniformly hang under the root node, so the tree kernel simply counts the number of common features between two relation instances. This tree setup is similar to linear entity kernel explored by Zhang et al. (2006).

(b) Feature-Paired Tree (FPT, e.g. Fig. 2(b)): the features of two entities are grouped into different types according to their feature names, e.g. "TP1" and "TP2" are grouped to "TP". This tree setup is aimed to capture the additional similarity of the single feature combined from different entities, i.e., the first and the second entities.

(c) Entity-Paired Tree (EPT, e.g. Fig. 2(c)): all the features relating to an entity are grouped to nodes "E1" or "E2", thus this tree kernel can further explore the equivalence of combined entity features only relating to one of the entities between two relation instances.

In fact, the BOF only captures the individual entity features, while the FPT/EPT can additionally capture the bi-gram/tri-gram features respectively.

Rather than constructing a composite kernel, we incorporate the EST into the DSPT to produce a Unified Parse and Semantic Tree (UPST) to investigate the contribution of the EST to relation extraction. The entity features can be attached under the top node, the entity nodes, or directly combined with the entity nodes as in Figure 1. However, detailed evaluation (Qian et al., 2007) indicates that the UPST achieves the best performance when the feature nodes are attached under the top node. Hence, we also attach three kinds of entity-related semantic trees (i.e. BOF, FPT and EPT) under the top node of the DSPT right after its original children. Thereafter, we employ the standard CTK (Collins and Duffy, 2001) to compute the similarity between two UPSTs, since this CTK and its variations are successfully applied in syntactic parsing, semantic role labeling (Moschitti, 2004) and relation extraction (Zhang et al., 2006; Zhou et al., 2007) as well.

5 Experimentation

This section will evaluate the effectiveness of the DSPT and the contribution of entity-related semantic information through experiments.

5.1 Experimental Setting

For evaluation, we use the ACE RDC 2004 corpus as the benchmark data. This data set contains 451 documents and 5702 relation instances. It defines 7 entity types, 7 major relation types and 23 subtypes. For comparison with previous work, evaluation is done on 347 (nwire/bnews) documents and 4307 relation instances using 5-fold cross-validation. Here, the corpus is parsed using Charniak's parser (Charniak, 2001) and relation instances are generated by iterating over all pairs of entity mentions occurring in the same sentence with given "true" mentions and coreferential information. In our experimentations, SVM^{light} (Joachims, 1998) with the tree kernel function (Moschitti, 2004) [2] is selected as our classifier. For efficiency, we apply the *one vs. others* strategy, which builds K classifiers so as to separate one class from all others. For comparison purposes, the training parameters C (SVM) and λ (tree kernel) are also set to 2.4 and 0.4 respectively.

5.2 Experimental Results

Table 1 evaluates the contributions of different kinds of constituent dependencies to extraction performance on the 7 relation types of the ACE RDC 2004 corpus using the convolution parse tree kernel as depicted in Figure 1. The MCT with only entity-type information is first used as the baseline, and various constituent dependencies are then applied sequentially to dynamically reshaping the tree in two different modes:

--[M1] Respective: every constituent dependency is individually applied on MCT.

--[M2] Accumulative: every constituent dependency is incrementally applied on the previously derived tree span, which begins with the MCT and eventually gives rise to a Dynamic Syntactic Parse Tree (DSPT).

Dependency types	P(%)	R(%)	F
MCT (baseline)	75.1	53.8	62.7
Modification within base-NPs	76.5 (59.8)	59.8 (59.8)	67.1 (67.1)
Modification to NPs	77.0 (76.2)	63.2 (56.9)	69.4 (65.1)
Arguments/adjuncts to verb	77.1 (76.1)	63.9 (57.5)	69.9 (65.5)
Coordination conjunctions	77.3 (77.3)	65.2 (55.1)	70.8 (63.8)
Other modifications	**77.4** (75.0)	**65.4** (53.7)	**70.9** (62.6)

Table 1. Contribution of constituent dependencies in respective mode (inside parentheses) and accumulative mode (outside parentheses)

The table shows that the final DSPT achieves the best performance of 77.4%/65.4%/70.9 in precision/recall/F-measure respectively after applying all the dependencies, with the increase of F-measure by 8.2 units compared to the baseline MCT. This indicates that reshaping the tree by exploiting constituent dependencies may significantly improve extraction accuracy largely due to the increase in recall. It further suggests that constituent dependencies knowledge is very effec-

[2] http://ai-nlp.info.uniroma2.it/moschitti/

tive and can be fully utilized in tree kernel-based relation extraction. This table also shows that:

(1) Both modification within base-NPs and modification to NPs contribute much to performance improvement, acquiring the increase of F-measure by 4.4/2.4 units in mode M1 and 4.4/2.3 units in mode M2 respectively. This indicates the local characteristic of semantic relations, which can be effectively captured by NPs near the two involved entities in the DSPT.

(2) All the other three dependencies show minor contribution to performance enhancement, they improve the F-measure only by 2.8/0.9/-0.1 units in mode M1 and 0.5/0.9/0.1 units in mode M2. This may be due to the reason that these dependencies only remove the nodes far from the two entities.

We compare in Table 2 the performance of Unified Parse and Semantic Trees with different kinds of Entity Semantic Tree setups using standard convolution tree kernel, while the SPT and DSPT with only entity-type information are listed for reference. It shows that:

(1) All the three unified parse and semantic tree kernels significantly outperform the DSPT kernel, obtaining an average increase of ~4 units in F-measure. This means that they can effectively capture both the structured syntactic information and the entity-related semantic features.

(2) The Unified Parse and Semantic Tree with Feature-Paired Tree achieves the best performance of 80.1/70.7/75.1 in P/R/F respectively, with an increase of F-measure by 0.4/0.3 units over BOF and EPT respectively. This suggests that additional bi-gram entity features captured by FPT are more useful than tri-gram entity features captured by EPT.

Tree setups	P(%)	R(%)	F
SPT	76.3	59.8	67.1
DSPT	77.4	65.4	70.9
UPST (BOF)	80.4	69.7	74.7
UPST (FPT)	**80.1**	**70.7**	**75.1**
UPST (EPT)	79.9	70.2	74.8

Table 2. Performance of Unified Parse and Semantic Trees (UPSTs) on the 7 relation types of the ACE RDC 2004 corpus

In Table 3 we summarize the improvements of different tree setups over SPT. It shows that in a similar setting, our DSPT outperforms SPT by 3.8 units in F-measure, while CS-SPT outperforms SPT by 1.3 units in F-measure. This suggests that the DSPT performs best among these tree spans. It also shows that the Unified Parse and Semantic Tree with Feature-Paired Tree perform significantly better than the other two tree setups (i.e., CS-SPT and DSPT) by 6.7/4.2 units in F-measure respectively. This implies that the entity-related semantic information is very useful and contributes much when they are incorporated into the parse tree for relation extraction.

Tree setups	P(%)	R(%)	F
CS-SPT over SPT[3]	1.5	1.1	1.3
DSPT over SPT	1.1	5.6	3.8
UPST (FPT) over SPT	3.8	10.9	8.0

Table 3. Improvements of different tree setups over SPT on the ACE RDC 2004 corpus

Finally, Table 4 compares our system with other state-of-the-art kernel-based systems on the 7 relation types of the ACE RDC 2004 corpus. It shows that our UPST outperforms all previous tree setups using one single kernel, and even better than two previous composite kernels (Zhang et al., 2006; Zhao and Grishman, 2005). Furthermore, when the UPST (FPT) kernel is combined with a linear state-of-the-state feature-based kernel (Zhou et al., 2005) into a composite one via polynomial interpolation in a setting similar to Zhou et al. (2007) (i.e. polynomial degree d=2 and coefficient α=0.3), we get the so far best performance of 77.1 in F-measure for 7 relation types on the ACE RDC 2004 data set.

Systems	P(%)	R(%)	F
Ours: composite kernel	83.0	72.0	77.1
Zhou et al., (2007): composite kernel	82.2	70.2	75.8
Zhang et al., (2006): composite kernel	76.1	68.4	72.1
Zhao and Grishman, (2005):[4] composite kernel	69.2	70.5	70.4
Ours: CTK with UPST	80.1	70.7	75.1
Zhou et al., (2007): context-sensitive CTK with CS-SPT	81.1	66.7	73.2
Zhang et al., (2006): CTK with SPT	74.1	62.4	67.7

Table 4. Comparison of different systems on the ACE RDC 2004 corpus

[3] We arrive at these values by subtracting P/R/F (79.6/5.6/71.9) of Shortest-enclosed Path Tree from P/R/F (81.1/6.7/73.2) of Dynamic Context-Sensitive Shortest-enclosed Path Tree according to Table 2 (Zhou et al., 2007)

[4] There might be some typing errors for the performance reported in Zhao and Grishman (2005) since P, R and F do not match.

6 Conclusion

This paper further explores the potential of structured syntactic information for tree kernel-based relation extraction, and proposes a new approach to dynamically determine the tree span (DSPT) for relation instances by exploiting constituent dependencies. We also investigate different ways of how entity-related semantic features and their combined features can be effectively captured in a Unified Parse and Semantic Tree (UPST). Evaluation on the ACE RDC 2004 corpus shows that our DSPT is appropriate for structured representation of relation instances. We also find that, in addition to individual entity features, combined entity features (especially bi-gram) contribute much when they are combined with a DPST into a UPST. And the composite kernel, combining the UPST kernel and a linear state-of-the-art kernel, yields the so far best performance.

For the future work, we will focus on improving performance of complex structured parse trees, where the path connecting the two entities involved in a relationship is too long for current kernel methods to take effect. Our preliminary experiment of applying certain discourse theory exhibits certain positive results.

Acknowledgements

This research is supported by Project 60673041 under the National Natural Science Foundation of China, Project 2006AA01Z147 under the "863" National High-Tech Research and Development of China, and the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No. 20060285008. We would also like to thank the excellent and insightful comments from the three anonymous reviewers.

References

Bunescu, Razvan C. and Raymond J. Mooney. 2005. A Shortest Path Dependency Kernel for Relation Extraction. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (EMNLP-2005)*, pages 724-731. Vancover, B.C.

Charniak, Eugene. 2001. Intermediate-head Parsing for Language Models. In *Proceedings of the 39th Annual Meeting of the Association of Computational Linguistics (ACL-2001)*, pages 116-123.

Collins, Michael. 2003. Head-Driven Statistics Models for Natural Language Parsing. *Computational linguistics*, 29(4): 589-617.

Collins, Michael and Nigel Duffy. 2001. Convolution Kernels for Natural Language. In *Proceedings of Neural Information Processing Systems (NIPS-2001)*, pages 625-632. Cambridge, MA.

Culotta, Aron and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting of the Association of Computational Linguistics (ACL-2004)*, pages 423-439. Barcelona, Spain.

Joachims, Thorsten. 1998. Text Categorization with Support Vector Machine: learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning (ECML-1998)*, pages 137-142. Chemnitz, Germany.

Moschitti, Alessandro. 2004. A Study on Convolution Kernels for Shallow Semantic Parsing. In *Proceedings of the 42nd Annual Meeting of the Association of Computational Linguistics (ACL-2004)*. Barcelona, Spain.

Qian, Longhua, Guodong Zhou, Qiaoming Zhu and Peide Qian. 2007. Relation Extraction using Convolution Tree Kernel Expanded with Entity Features. In *Proceedings of the 21st Pacific Asian Conference on Language, Information and Computation (PACLIC-21)*, pages 415-421. Seoul, Korea.

Zelenko, Dmitry, Chinatsu Aone and Anthony Richardella. 2003. Kernel Methods for Relation Extraction. *Journal of Machine Learning Research,* 3(2003): 1083-1106.

Zhang, Min, 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 the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association of Computational Linguistics (COLING/ACL-2006)*, pages 825-832. Sydney, Australia.

Zhao, Shubin and Ralph Grishman. 2005. Extracting relations with integrated information using kernel methods. In *Proceedings of the 43rd Annual Meeting of the Association of Computational Linguistics (ACL-2005)*, pages 419-426. Ann Arbor, USA.

Zhou, Guodong, Jian Su, Jie Zhang and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting of the Association of Computational Linguistics (ACL-2005)*, pages 427-434. Ann Arbor, USA.

Zhou, Guodong, Min Zhang, Donghong Ji and Qiaoming Zhu. 2007. Tree Kernel-based Relation Extraction with Context-Sensitive Structured Parse Tree Information. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP/CoNLL-2007)*, pages 728-736. Prague, Czech.