A Feature-Enriched Tree Kernel for Relation Extraction

Le Sun and **Xianpei Han**
State Key Laboratory of Computer Science
Institute of Software, Chinese Academy of Sciences
HaiDian District, Beijing, China.
{sunle, xianpei}@nfs.iscas.ac.cn

Abstract

Tree kernel is an effective technique for relation extraction. However, the traditional syntactic tree representation is often *too coarse* or *ambiguous* to accurately capture the semantic relation information between two entities. In this paper, we propose a new tree kernel, called *feature-enriched tree kernel (FTK)*, which can enhance the traditional tree kernel by: 1) refining the syntactic tree representation by annotating each tree node with a set of discriminant features; and 2) proposing a new tree kernel which can better measure the syntactic tree similarity by taking all features into consideration. Experimental results show that our method can achieve a 5.4% F-measure improvement over the traditional convolution tree kernel.

1 Introduction

Relation Extraction (RE) aims to identify a set of predefined relations between pairs of entities in text. In recent years, relation extraction has received considerable research attention. An effective technique is the tree kernel (Zelenko et al., 2003; Zhou et al., 2007; Zhang et al., 2006; Qian et al., 2008), which can exploit syntactic parse tree information for relation extraction. Given a pair of entities in a sentence, the tree kernel-based RE method first represents the relation information between them using a proper sub-tree (e.g., SPT – the sub-tree enclosed by the shortest path linking the two involved entities). For example, the three syntactic tree representations in Figure 1. Then the similarity between two trees are computed using a tree kernel, e.g., the convolution tree kernel proposed by Collins and Duffy (2001). Finally, new relation instances are extracted using kernel based classifiers, e.g., the SVM classifier.

Unfortunately, one *main* shortcoming of the traditional tree kernel is that the syntactic tree representation usually cannot accurately capture the relation information between two entities. This is mainly due to the following two reasons:

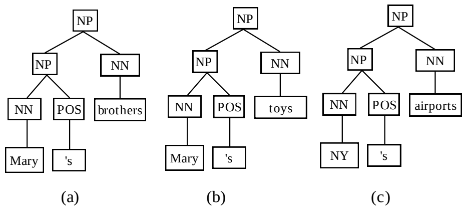

Figure 1. The ambiguity of possessive structure

1) *The syntactic tree focuses on representing syntactic relation/structure, which is often too coarse or ambiguous to capture the semantic relation information*. In a syntactic tree, each node indicates a clause/phrase/word and is only labeled with a Treebank tag (Marcus et al., 1993). The Treebank tag, unfortunately, is usually too coarse or too general to capture semantic information. For example, all the three trees in Figure 1 share the same possessive syntactic structure, but express quite different semantic relations: where "*Mary's brothers*" expresses *PER-SOC Family* relation, "*Mary's toys*" expresses *Possession* relation, and "*New York's airports*" expresses *PHYS-Located* relation.

2) *Some critical information may lost during sub-tree representation extraction*. For example, in Figure 2, when extracting SPT representation, all nodes outside the shortest-path will be pruned, such as the nodes *[NN plants]* and *[POS 's]* in tree *T1*. In this pruning process, the critical information "word *town* is the possessor of the possessive phrase *the town's plants*" will be lost, which in turn will lead to the misclassification of the *DISC* relation between *one* and *town*.

This paper proposes a new tree kernel, referred as *feature-enriched tree kernel (FTK),* which can effectively resolve the above problems by enhancing the traditional tree kernel in following ways:

1) We refine the syntactic tree representation by annotating each tree node with a set of discriminant features. These features are utilized to

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 61–67,
Baltimore, Maryland, USA, June 23-25 2014. ©2014 Association for Computational Linguistics

better capture the semantic relation information between two entities. For example, in order to differentiate the syntactic tree representations in Figure 1, FTK will annotate them with several features indicating "*brother is a male sibling*", "*toy is an artifact*", "*New York is a city*", "*airport is facility*", etc.

2) Based on the refined syntactic tree representation, we propose a new tree kernel – *feature-enriched tree kernel*, which can better measure the similarity between two trees by also taking all features into consideration.

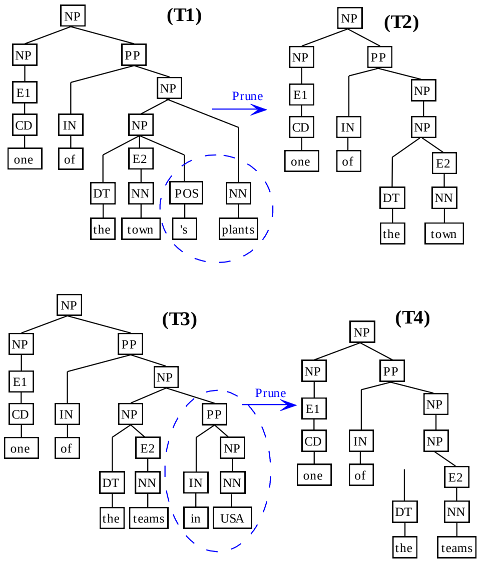

Figure 2. SPT representation extraction

We have experimented our method on the ACE 2004 RDC corpus. Experimental results show that our method can achieve a 5.4% F-measure improvement over the traditional convolution tree kernel based method.

This paper is organized as follows. Section 2 describes the feature-enriched tree kernel. Section 3 presents the features we used. Section 4 discusses the experiments. Section 5 briefly reviews the related work. Finally Section 6 concludes this paper.

2 The Feature-Enriched Tree Kernel

In this section, we describe the proposed feature-enriched tree kernel (FTK) for relation extraction.

2.1 Refining Syntactic Tree Representation

As described in above, syntactic tree is often too coarse or too ambiguous to represent the semantic relation information between two entities. To resolve this problem, we refine the syntactic tree representation by annotating each tree node with a set of discriminant features.

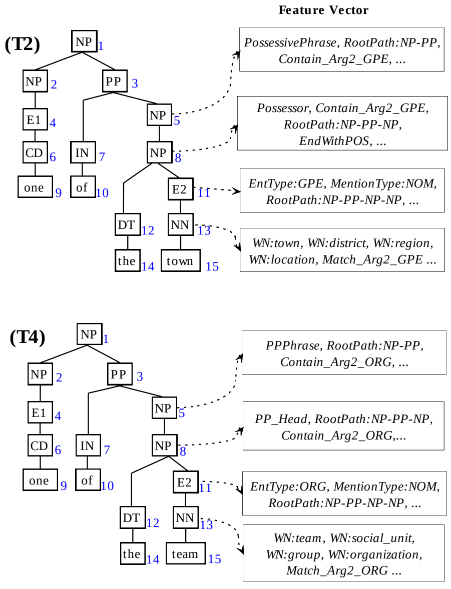
Feature Vector
(T2)
NP 1
NP 2
PP 3
E1 4
NP 5
CD 6
IN 7
NP 8
one 9
of 10
E2 11
DT 12
NN 13
the 14
town 15
PossessivePhrase, RootPath:NP-PP, Contain_Arg2_GPE, ...
Possessor, Contain_Arg2_GPE, RootPath:NP-PP-NP, EndWithPOS, ...
EntType:GPE, MentionType:NOM, RootPath:NP-PP-NP-NP, ...
WN:town, WN:district, WN:region, WN:location, Match_Arg2_GPE ...
(T4)
NP 1
NP 2
PP 3
E1 4
NP 5
CD 6
IN 7
NP 8
one 9
of 10
E2 11
DT 12
NN 13
the 14
team 15
PPPhrase, RootPath:NP-PP, Contain_Arg2_ORG, ...
PP_Head, RootPath:NP-PP-NP, Contain_Arg2_ORG,...
EntType:ORG, MentionType:NOM, RootPath:NP-PP-NP-NP, ...
WN:team, WN:social_unit, WN:group, WN:organization, Match_Arg2_ORG ...

Figure 3. Syntactic tree enriched with features

Specifically, for each node n in a syntactic tree T, we represent it as a tuple:

$$R_n = (L_n, F_n)$$

where L_n is its phrase label (i.e., its Treebank tag), and F_n is a feature vector which indicates the characteristics of node n, which is represented as:

$$F_n = \{f_1, f_2, ..., f_N\}$$

where f_i is a feature and is associated with a weight $w_i \in (0,1)$. The feature we used includes characteristics of relation instance, phrase properties and context information (See Section 3 for details).

For demonstration, Figure 3 shows the feature-enriched version of tree *T2* and tree *T4* in Figure 2. We can see that, although *T2* and *T4* share the same syntactic structure, the annotated features can still differentiate them. For example, the NP_5 node in tree *T2* and the NP_5 node in tree *T4* are differentiated using their features *Possessive-Phrase* and *PPPhrase*, which indicate that NP_5 in *T2* is a possessive phrase, meanwhile NP_5 in *T4* is a preposition phrase.

2.2 Feature-Enriched Tree Kernel

This section describes how to take into account the annotated features for a better tree similarity.

In Collins and Duffy's convolution tree kernel (CTK), the similarity between two trees T_1 and T_2 is the number of their common sub-trees:

$$K_C(T_1, T_2) = \sum_{t_1 \in T_1} \sum_{t_2 \in T_2} \delta(t_1, t_2)$$

Using this formula, CTK only considers whether two enumerated sub-trees have the identical syntactic structure (the indicator $\delta(t_1, t_2)$ is *1* if the two sub-trees t_1 and t_2 have the identical syntactic structure and *0* otherwise). Such an assumption makes CTK can only capture the syntactic structure similarity between two trees, while ignoring other useful information.

To resolve the above problem, the *feature-enriched tree kernel (FTK)* compute the similarity between two trees as the sum of the similarities between their common sub-trees:

$$F_{tk}(T_1, T_2) = \sum_{t_1 \in T_1} \sum_{t_2 \in T_2} k(t_1, t_2)$$

where $k(t_1, t_2)$ is the similarity between enumerated sub-trees t_1 and t_2, which is computed as:

$$k(t_1, t_2) = \delta(t_1, t_2) \times \prod_{(n_i, n_j) \in E(t_1, t_2)} (1 + sim(n_i, n_j))$$

where $\delta(t_1, t_2)$ is the same indicator function as in CTK; $(n_i, n_j)$is a pair of aligned nodes between t_1 and t_2, where n_i and n_j are correspondingly in the same position of tree t_1 and t_2; $E(t_1, t_2)$ is the set of all aligned node pairs; $sim(n_i, n_j)$ is the feature vector similarity between noden_i and n_j, computed as the dot product between their feature vectors F_{n_i} and F_{n_j}.

Notice that, if all nodes are not annotated with features, $k(t_1, t_2)$ will be equal to $\delta(t_1, t_2)$. In this perspective, we can view $k(t_1, t_2)$ as a similarity adjusted version of $\delta(t_1, t_2)$, i.e., $\delta(t_1, t_2)$ only considers whether two nodes are equal, in contrast $k(t_1, t_2)$ further considers the feature similarity $sim(n_i, n_j)$ between two nodes.

The Computation of FTK. As the same as CTK, FTK can be efficiently computed as:

$$F_{tk}(T_1, T_2) = \sum_{n_1 \in N_1, n_2 \in N_2} \Delta(n_1, n_2)$$

where N_j is the set of nodes in tree T_j, and $\Delta(n_1, n_2)$ evaluates the sum of the similarities of common sub-trees rooted at node n_1 and node n_2, which is recursively computed as follows:

1) If the production rules of n_1 and n_2 are different, $\Delta(n_1, n_2)$ = 0;
2) If both n_1 and n_2 is pre-terminal nodes, $\Delta(n_1, n_2) = (1 + sim(n_1, n_2)) \times \lambda$; Otherwise go to step 3;
3) Calculate $\Delta(n_1, n_2)$ recursively as:

$$\Delta(n_1, n_2) = \lambda \times (1 + sim(n_1, n_2)) \times \sum_{k=1}^{\#ch(n_1)} (1 + \Delta(ch(n_1, k), ch(n_2, k)))$$

3 Features for Relation Extraction

This section presents the features we used to enrich the syntactic tree representation.

3.1 Instance Feature

Relation instances of the same type often share some common characteristics. In this paper, we add the following instance features to the root node of a sub-tree representation:

1) **Syntactico-Semantic structure**. A feature indicates whether a relation instance has the following four syntactico-semantic structures in (Chan & Roth, 2011) – *Premodifiers, Possessive, Preposition, Formulaic* and *Verbal*.

2) **Entity-related information of arguments**. Features about the entity information of arguments, including: a) *#TP1-#TP2*: the concat of the major entity types of arguments; b) *#ST1-#ST2*: the concat of the sub entity types of arguments; c) *#MT1-#MT2*: the concat of the mention types of arguments.

3) Base phrase chunking features. Features about the phrase path between two arguments and the phrases' head before and after the arguments, which are the same as the phrase chunking features in (Zhou, et al., 2005).

3.2 Phrase Feature

As discussed in above, the Treebank tag is too coarse to capture the property of a phrase node. Therefore, we enrich each phrase node with features about its lexical pattern, its content information, and its lexical semantics:

1) **Lexical Pattern.** We capture the lexical pattern of a phrase node using the following features: a) *LP_Poss:* A feature indicates the node is a possessive phrase; b) *LP_PP*: A feature indicates the node is a preposition phrase; c) *LP_CC*: A feature indicates the node is a conjunction phrase; d) *LP_EndWithPUNC*: A feature indicates the node ends with a punctuation; e) *LP_EndWithPOSS*: A feature indicates the node ends with a possessive word.

2) **Content Information**. We capture the property of a node's content using the following features: *a) MB_#Num*: The number of mentions contained in the phrase; b) *MB_C_#Type*: A feature indicates that the phrase contains a mention with major entity type *#Type*; c) *MW_#Num*: The number of words within the phrase.

3) Lexical Semantics. If the node is a preterminal node, we capture its lexical semantic by adding features indicating its WordNet sense information. Specifically, the first WordNet sense of the terminal word, and all this sense's hyponym senses will be added as features. For example, WordNet senses {*New York#1, city#1, district#1,* *region#1, ...*} will be added as features to the *[NN New York]* node in Figure 1.

3.3 Context Information Feature

The context information of a phrase node is critical for identifying the role and the importance of a sub-tree in the whole relation instance. This paper captures the following context information:

1) **Contextual path from sub-tree root to the phrase node**. As shown in Zhou et al. (2007), the context path from root to the phrase node is an effective context information feature. In this paper, we use the same settings in (Zhou et al., 2007), i.e., each phrase node is enriched with its context paths of length 1, 2, 3.

2) **Relative position with arguments**. We observed that a phrase's relative position with the relation's arguments is useful for identifying the role of the phrase node in the whole relation instance. To capture the relative position information, we define five possible relative positions between a phrase node and an argument, corresponding *match*, *cover*, *within*, *overlap* and *other*. Using these five relative positions, we capture the context information using the following features:

a) *#RP_Arg1Head_#Arg1Type*: a feature indicates the relative position of a phrase node with argument 1's head phrase, where *#RP* is the relative position (one of *match*, *cover*, *within*, *overlap*, *other*), and *#Arg1Type* is the major entity type of argument 1. One example feature may be *Match_Arg1Head_LOC*.

b) *#RP_Arg2Head_#Arg2Type*: The relative position with argument 2's head phrase;

c) *#RP_Arg1Extend_#Arg1Type*: The relative position with argument 1's extended phrase;

d) *#PR_Arg2Extend_#Arg2Type*: The relative position with argument 2's extended phrase.

Feature weighting. Currently, we set all features with an uniform weight $w \in (0, 1)$, which is used to control the relative importance of the feature in the final tree similarity: the larger the feature weight, the more important the feature in the final tree similarity.

4 Experiments

4.1 Experimental Setting

To assess the feature-enriched tree kernel, we evaluate our method on the ACE RDC 2004 corpus using the same experimental settings as (Qian et al., 2008). That is, we parse all sentences using the Charniak's parser (Charniak, 2001), relation instances are generated by iterating over all pairs of entity mentions occurring in the same sentence. In our experiments, we implement the feature-enriched tree kernel by extending the SVM^{light} (Joachims, 1998) with the proposed tree kernel function (Moschitti, 2004). We apply the *one vs. others* strategy for multiple classification using SVM. For SVM training, the parameter C is set to 2.4 for all experiments, and the tree kernel parameter λ is tuned to 0.2 for FTK and 0.4 (the optimal parameter setting used in Qian et al.(2008)) for CTK.

4.2 Experimental Results

4.2.1 Overall performance

We compare our method with the standard convolution tree kernel (CTK) on the state-of-the-art context sensitive shortest path-enclosed tree representation (CSPT, Zhou et al., 2007). We experiment our method with four different feature settings, correspondingly: 1) FTK with only instance features – *FTK(instance)*; 2) FTK with only phrase features – *FTK(phrase)*; 3) FTK with only context information features – *FTK(context)*; and 4) FTK with all features – *FTK*. The overall performance of CTK and FTK is shown in Table 1, the F-measure improvements over CTK are also shown inside the parentheses. The detailed performance of FTK on the 7 major relation types of ACE 2004 is shown in Table 2.

	P(%)	**R(%)**	**F**
CTK	77.1	61.3	68.3 (-------)
FTK(instance)	78.5	64.6	70.9 (+2.6%)
FTK(phrase)	78.3	64.2	70.5 (+2.2%)
FTK(context)	80.1	67.5	73.2 (+4.9%)
FTK	**81.2**	**67.4**	**73.7 (+5.4%)**

Table 1. Overall Performance

Relation Type	**P(%)**	**R(%)**	**F**	**Impr**
EMP-ORG	84.7	82.4	83.5	5.8%
PER-SOC	79.9	70.7	75.0	1.0%
PHYS	73.3	64.4	68.6	7.0%
ART	83.6	57.5	68.2	1.7%
GPE-AFF	74.7	56.6	64.4	4.3%
DISC	81.6	48.0	60.5	6.6%
OTHER-AFF	74.2	36.8	49.2	1.0%

Table 2. FTK on the 7 major relation types and their F-measure improvement over CTK

From Table 1 and 2, we can see that:

1) By refining the syntactic tree with discriminant features and incorporating these features into the final tree similarity, FTK can significantly improve the relation extraction performance: compared with the convolution tree kernel baseline *CTK*, our method can achieve a 5.4% F-measure improvement.

2) All types of features can improve the performance of relation extraction: FTK can correspondingly get 2.6%, 2.2% and 4.9% F-measure improvements using instance features, phrase features and context information features.

3) Within the three types of features, context information feature can achieve the highest F-measure improvement. We believe this may because: ① The context information is useful in providing clues for identifying the role and the importance of a sub-tree; and ② The context-free assumption of CTK is too strong, some critical information will lost in the CTK computation.

4) The performance improvement of FTK varies significantly on different relation types: in Table 2, most performance improvement gains from the *EMP-ORG*, *PHYS*, *GPE-AFF* and *DISC* relation types. We believe this may because the discriminant features will better complement the syntactic tree for capturing *EMP-ORG*, *PHYS*, *GPE-AFF* and *DISC* relation. On contrast the features may be redundant to the syntactic information for other relation types.

System	**P(%)**	**R(%)**	**F**
Qian et al., (2008): composite kernel	83.0	72.0	77.1
Zhou et al., (2007): composite kernel	82.2	70.2	75.8
Ours: FTK with CSPT	**81.2**	**67.4**	**73.7**
Zhou et al., (2007): context sensitive CTK with CSPT	81.1	66.7	73.2
Ours: FTK with SPT	**81.1**	**66.2**	**72.9**
Jiang & Zhai (2007): MaxEnt classifier with features	74.6	71.3	72.9
Zhang et al., (2006): composite kernel	76.1	68.4	72.1
Zhao & Grishman, (2005): Composite kernel	69.2	70.5	70.4
Zhang et al., (2006): CTK with SPT	74.1	62.4	67.7

Table 3. Comparison of different systems on the ACE RDC 2004 corpus

4.2.2 Comparison with other systems

Finally, Table 3 compares the performance of our method with several other systems. From Table 3, we can see that FTK can achieve competitive performance: ① It achieves a 0.8% F-measure improvement over the feature-based system of Jiang & Zhai (2007); ② It achieves a 0.5% F-measure improvement over a state-of-the-art tree kernel: context sensitive CTK with CSPT of Zhou et al., (2007); ③ The F-measure of our system is slightly lower than the current best performance on ACE 2004 (Qian et al., 2008) – *73.7 vs. 77.1*, we believe this is because the system of (Qian et al., 2008) adopts two extra techniques: composing tree kernel with a state-of-the-art feature-based kernel and using a more proper sub-tree representation. We believe these two techniques can also be used to further improve the performance of our system.

5 Related Work

This section briefly reviews the related work. A classical technique for relation extraction is to model the task as a feature-based classification problem (Kambhatla, 2004; Zhou et al., 2005; Jiang & Zhai, 2007; Chan & Roth, 2010; Chan & Roth, 2011), and feature engineering is obviously the key for performance improvement. As an alternative, tree kernel-based method implicitly defines features by directly measuring the similarity between two structures (Bunescu and Mooney, 2005; Bunescu and Mooney, 2006; Zelenko et al, 2003; Culotta and Sorensen, 2004; Zhang et al., 2006). Composite kernels were also be used (Zhao and Grishman, 2005; Zhang et al., 2006).

The main drawback of the current tree kernel is that the syntactic tree representation often cannot accurately capture the relation information. To resolve this problem, Zhou et al. (2007) took the ancestral information of sub-trees into consideration; Reichartz and Korte (2010) incorporated dependency type information into a tree kernel; Plank and Moschitti (2013) and Liu et al. (2013) embedded semantic information into tree kernel. Bloehdorn and Moschitti (2007a, 2007b) proposed Syntactic Semantic Tree Kernels (SSTK), which can capture the semantic similarity between leaf nodes. Moschitti (2009) proposed a tree kernel which specify a kernel function over any pair of nodes between two trees, and it was further extended and applied in other tasks in (Croce et al., 2011; Croce et al., 2012; Mehdad et al., 2010).

6 Conclusions and Future Work

This paper proposes a *feature-enriched tree kernel*, which can: 1) refine the syntactic tree representation; and 2) better measure the similarity between two trees. For future work, we want to develop a feature weighting algorithm which can accurately measure the relevance of a feature to a relation instance for better RE performance.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants no. 61100152 and 61272324, and the Open Project of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research under Grants no. ICDD201204.

References

Agichtein, E. and Gravano, L. 2000. *Snowball: Extracting relations from large plain-text collections*. In: Proceedings of the 5th ACM Conference on Digital Libraries, pp. 85–94.

Plank, B. and Moschitti, A. 2013. *Embedding Semantic Similarity in Tree Kernels for Domain Adaptation of Relation Extraction*". In: Proceedings of ACL 2013.

Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M. and Etzioni, O. 2007. *Open information extraction from the Web*. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 2670–2676.

Bunescu, R. and Mooney, R. 2005. *A shortest path dependency kernel for relation extraction*. In: Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing, pp.724–731.

Bloehdorn, S. and Moschitti, A. 2007a. *Combined Syntactic and Semantic Kernels for Text Classification*. In: Proceedings of the 29th European Conference on Information Retrieval (ECIR).

Bloehdorn, S. and Moschitti, A. 2007b. *Structure and semantics for expressive text kernels*. In: Proceeding of ACM 16th Conference on Information and Knowledge Management (CIKM).

Bunescu, R. and Mooney. R., 2006. *Subsequence kernels for relation extraction*. In: Advances in Neural Information Processing Systems 18, pp. 171–178.

Charniak, E., 2001. *Immediate-head parsing for language models*. In: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics, pp. 124-131.

Chan, Y. S. and Roth, D. 2010. *Exploiting background knowledge for relation extraction*. In: Proceedings of the 23rd International Conference on Computational Linguistics, pp. 152–160.

Chan, Y. S. and Roth, D. 2011. *Exploiting syntactico-semantic structures for relation extraction*. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, pp. 551–560.

Croce, D., Moschitti, A. and Basili, R. 2011. *Structured lexical similarity via convolution kernels on dependency trees*. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pp. 1034–1046.

Croce, D., Moschitti, A., Basili, R. and Palmer, M. 2012. *Verb Classification using Distributional Similarity in Syntactic and Semantic Structures*. In: Proceedings of ACL 2012, pp. 263-272.

Culotta, A. and Sorensen, J. 2004. *Dependency tree kernels for relation extraction*. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, pp. 423–429.

Grishman, R. and Sundheim, B. 1996. *Message understanding conference-6: A brief history*. In: Proceedings of the 16th International Conference on Computational Linguistics, pp. 466–471.

Collins, M. and Duffy, N., 2001. *Convolution Kernels for Natural Language*. In: Proceedings of NIPS 2001.

Liu, D., et al. 2013. *Incorporating lexical semantic similarity to tree kernel-based Chinese relation extraction*. In: Proceedings of Chinese Lexical Semantics 2013.

Jiang, J. and Zhai, C. 2007. *A systematic exploration of the feature space for relation extraction*. In: Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, pp. 113–120.

Joachims, T. 1998. *Text Categorization with Su pport Vector Machine*: learning with many relevant features. ECML-1998: 137-142.

Kambhatla, N. 2004. *Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations*. In: the Proceedings of 42st Annual Meeting of the Association for Computational Linguistics, pp. 178–181.

Krause, S., Li, H., Uszkoreit, H., & Xu, F. 2012. *Large-scale learning of relation-extraction rules with distant supervision from the web*. In: Proceedings of ISWC 2012, pp. 263-278.

Marcus, M. P., Marcinkiewicz, M. A., & Santorini, B. 1993. *Building a large annotated corpus of English: The Penn Treebank*. Computational linguistics, 19(2), 313-330.

Moschitti, A. 2004. *A study on Convolution Kernels for Shallow Semantic Parsing*. In: Proceedings of the 42-th Conference on Association for Computational Linguistic (ACL-2004).

Moschitti, A. 2009. *Syntactic and semantic kernels for short text pair categorization*. In: Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009), pp. 576–584.

Mehdad, Y., Moschitti, A. and Zanzotto, F. 2010. *Syntactic/Semantic Structures for Textual Entailment Recognition*. In: Proceedings of Human Language Technology - North American chapter of the Association for Computational Linguistics.

Mintz, M., Bills, S., Snow, R. and Jurafsky D. 2009. *Distant supervision for relation extraction without labeled data*. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pp. 1003–1011.

Qian L., Zhou G., Kong F., Zhu Q., and Qian P., 2008. *Exploiting constituent dependencies for tree kernel based semantic relation extraction*. In: Proceedings of the 22nd International Conference on Computational Linguistics, pp. 697-704.

Reichartz, F. and H. Korte, et al. 2010. *Semantic relation extraction with kernels over typed dependency trees*. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining.

Zelenko, D., Aone, C., and Richardella, A. 2003. *Kernel methods for relation extraction*. Journal of Machine Learning Research, 3:1083–1106.

Zhang, M., Zhang, J., and Su, J. 2006. *Exploring syntactic features for relation extraction using a convolution tree kernel*. In: Proceedings of the Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics, pages 288–295.

Zhang, M., Zhang, J., Su, J. and Zhou, G. 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 for Computational Linguistics, pages 825–832.

Zhao, S. and Grishman, R. 2005. *Extracting relations with integrated information using kernel methods*. In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, pages 419–426.

Zhou, G., Su, J., Zhang, J., and Zhang, M. 2005. *Exploring various knowledge in relation extraction.* In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, pages 427–434.

Zhou, G. and Zhang M. 2007. *Extracting relation information from text documents by exploring various types of knowledge*. Information Processing & Management 43(4): 969--982.

Zhou, G., et al. 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, pp. 728–736.