Semi-Supervised Learning for Relation Extraction

ZHOU GuoDong LI JunHui QIAN LongHua ZHU Qiaoming
Jiangsu Provincial Key Lab for Computer Information Processing Technology
School of Computer Science and Technology
Soochow Univ., Suzhou, China 215006
Email : {gdzhou, lijunhui, qianlonghua, qmzhu}@suda.edu.cn

Abstract

This paper proposes a semi-supervised learning method for relation extraction. Given a small amount of labeled data and a large amount of unlabeled data, it first bootstraps a moderate number of weighted support vectors via SVM through a co-training procedure with random feature projection and then applies a label propagation (LP) algorithm via the bootstrapped support vectors. Evaluation on the ACE RDC 2003 corpus shows that our method outperforms the normal LP algorithm via all the available labeled data without SVM bootstrapping. Moreover, our method can largely reduce the computational burden. This suggests that our proposed method can integrate the advantages of both SVM bootstrapping and label propagation.

1 Introduction

Relation extraction is to detect and classify various predefined semantic relations between two entities from text and can be very useful in many NLP applications such as question answering, e.g. to answer the query “Who is the president of the United States?”, and information retrieval, e.g. to expand the query “George W. Bush” with “the president of the United States” via his relationship with “the United States”.

During the last decade, many methods have been proposed in relation extraction, such as supervised learning (Miller et al 2000; Zelenko et al 2003; Culota and Sorensen 2004; Zhao and Grishman 2005; Zhang et al 2006; Zhou et al 2005, 2006), semi-supervised learning (Brin 1998; Agichtein and Gravano 2000; Zhang 2004; Chen et al 2006), and unsupervised learning (Hasegawa et al 2004; Zhang et al 2005). Among these methods, supervised learning-based methods perform much better than the other two alternatives. However, their performance much depends on the availability of a large amount of manually labeled data and it is normally difficult to adapt an existing system to other applications and domains. On the other hand, unsupervised learning-based methods do not need the definition of relation types and the availability of manually labeled data. However, they fail to classify exact relation types between two entities and their performance is normally very low. To achieve better portability and balance between human efforts and performance, semi-supervised learning has drawn more and more attention recently in relation extraction and other NLP applications.

This paper proposes a semi-supervised learning method for relation extraction. Given a small amount of labeled data and a large amount of unlabeled data, our proposed method first bootstraps a moderate number of weighted support vectors from all the available data via SVM using a co-training procedure with random feature projection and then applies a label propagation (LP) algorithm to capture the manifold structure in both the labeled and unlabeled data via the bootstrapped support vectors. Compared with previous methods, our method can integrate the advantages of both SVM bootstrapping in learning critical instances for the labeling function and label propagation in capturing the manifold structure in both the labeled and unlabeled data to smooth the labeling function.

The rest of this paper is as follows. In Section 2, we review related semi-supervised learning work in relation extraction. Then, the LP algorithm via bootstrapped support vectors is proposed in Section 3 while Section 4 shows the experimental results. Finally, we conclude our work in Section 5.

2 Related Work

Generally, supervised learning is preferable to unsupervised learning due to prior knowledge in the annotated training data and better performance. However, the annotated data is usually expensive to obtain. Hence, there has been growing interest in semi-supervised learning, aiming at inducing classifiers by leveraging a small amount of labeled data and a large amount of unlabeled data. Related work in relation extraction using semi-supervised learning can be classified into two categories: bootstrapping-based (Brin 1998; Agichtein and Gravano 2000; Zhang 2004) and label propagation(LP)-based (Chen et al 2006).

Currently, bootstrapping-based methods dominate semi-supervised learning in relation extraction. Bootstrapping works by iteratively classifying unlabeled instances and adding confidently classified ones into labeled data using a model learned from augmented labeled data in previous iteration. Brin (1998) proposed a bootstrapping-based method on the top of a self-developed pattern matching-based classifier to exploit the duality between patterns and relations. Agichtein and Gravano (2000) shared much in common with Brin (1998). They employed an existing pattern matching-based classifier (i.e. SNoW) instead. Zhang (2004) approached the much simpler relation classification sub-task by bootstrapping on the top of SVM. Although bootstrapping-based methods have achieved certain success, one problem is that they may not be able to well capture the manifold structure among unlabeled data.

As an alternative to the bootstrapping-based methods, Chen et al (2006) employed a LP-based method in relation extraction. Compared with bootstrapping, the LP algorithm can effectively combine labeled data with unlabeled data in the learning process by exploiting the manifold structure (e.g. the natural clustering structure) in both the labeled and unlabeled data. The rationale behind this algorithm is that the instances in high-density areas tend to carry the same labels. The LP algorithm has also been successfully applied in other NLP applications, such as word sense disambiguation (Niu et al 2005), text classification (Szummer and Jaakkola 2001; Blum and Chawla 2001; Belkin and Niyogi 2002; Zhu and Ghahramani 2002; Zhu et al 2003; Blum et al 2004), and information retrieval (Yang et al 2006). However, one problem is its computational burden, especially when a large amount of labeled and unlabeled data is taken into consideration.

In order to take the advantages of both bootstrapping and label propagation, our proposed method propagates labels via bootstrapped support vectors. On the one hand, our method can well capture the manifold structure in both the labeled and unlabeled data. On the other hand, our method can largely reduce the computational burden in the normal LP algorithm via all the available data.

3 Label Propagation via Bootstrapped Support Vectors

The idea behind our LP algorithm via bootstrapped support vectors is that, instead of propagating labels through all the available labeled data, our method propagates labels through critical instances in both the labeled and unlabeled data. In this paper, we use SVM as the underlying classifier to bootstrap a moderate number of weighted support vectors for this purpose. This is based on an assumption that the manifold structure in both the labeled and unlabeled data can be well preserved through the critical instances (i.e. the weighted support vectors bootstrapped from all the available labeled and unlabeled data). The reason why we choose SVM is that it represents the state-of-the-art in machine learning research and there are good implementations of the algorithm available. In particular, SVMLight (Joachims 1998) 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. Another reason is that we can adopt the weighted support vectors returned by the bootstrapped SVMs as the critical instances, via which label propagation is done.

3.1 Bootstrapping Support Vectors

This paper modifies the SVM bootstrapping algorithm BootProject(Zhang 2004) to bootstrap support vectors. Given a small amount of labeled data and a large amount of unlabeled data, the modified BootProject algorithm bootstraps on the top of SVM by iteratively classifying unlabeled instances and moving confidently classified ones into labeled data using a model learned from the augmented labeled data in previous iteration, until not enough unlabeled instances can be classified confidently. Figure 1 shows the modified BootProject algorithm for bootstrapping support vectors.

```
Assume:
  $L$: the labeled data;
  $U$: the unlabeled data;
  $S$: the batch size (100 in our experiments);
  $P$: the number of views(feature projections);
  $r$: the number of classes (including all the relation (sub)types and the non-relation)

BEGIN
  REPEAT
    FOR i = 1 to P DO
      Generate projected feature space $F_i$ from the original feature space $F$;
      Project both $L$ and $U$ onto $F_i$, thus generate $L_i$ and $U_i$;
      Train SVM classifier $SVM_{ij}$ on $L_i$ for each class $r_j (j = 1 \dots r)$;
      Run $SVM_{ij}$ on $U_i$ for each class $r_j (j = 1 \dots r)$
    END FOR
    Find (at most) $S$ instances in $U$ with the highest agreement (with threshold 70% in our experiments) and the highest average SVM-returned confidence value (with threshold 1.0 in our experiments);
    Move them from U to L;
  UNTIL not enough unlabeled instances (less than 10 in our experiments) can be confidently classified;
  Return all the (positive and negative) support vectors included in all the latest SVM classifiers $SVM_{ij}$ with their collective weight (absolute alpha*y) information as the set of bootstrapped support vectors to act as the labeled data in the LP algorithm;
  Return U (those hard cases which can not be confidently classified) to act as the unlabeled data in the LP algorithm;
END
```

Figure 1: The algorithm for bootstrapping support vectors

In particular, this algorithm generates multiple overlapping "views" by projecting from the original feature space. In this paper, feature views with random feature projection, as proposed in Zhang (2004), are explored. Section 4 will discuss this issue in more details. During the iterative training process, classifiers trained on the augmented labeled data using the projected views are then asked to vote on the remaining unlabeled instances and those with the highest probability of being correctly labeled are chosen to augment the labeled data.

During the bootstrapping process, the support vectors included in all the trained SVM classifiers (for all the relation (sub)types and the non-relation) are bootstrapped (i.e. updated) at each iteration. When the bootstrapping process stops, all the (positive and negative) support vectors included in the SVM classifiers are returned as bootstrapped support vectors with their collective weights (absolute a*y) to act as the labeled data in the LP algorithm and all the remaining unlabeled instances (i.e. those hard cases which can not be confidently classified in the bootstrapping process) in the unlabeled data are returned to act as the unlabeled data in the LP algorithm. Through SVM bootstrapping, our LP algorithm will only depend on the critical instances (i.e. support vectors with their weight information bootstrapped from all the available labeled and unlabeled data) and those hard instances, instead of all the available labeled and unlabeled data.

3.2 Label Propagation

In the LP algorithm (Zhu and Ghahramani 2002), the manifold structure in data is represented as a connected graph. Given the labeled data (the above bootstrapped support vectors with their weights) and unlabeled data (the remaining hard instances in the unlabeled data after bootstrapping, including all the test instances for evaluation), the LP algorithm first represents labeled and unlabeled instances as vertices in a connected graph, then propagates the label information from any vertex to nearby vertex through weighted edges and finally infers the labels of unlabeled instances until a global stable stage is achieved. Figure 2 presents the label propagation algorithm on bootstrapped support vectors in details.

```
Assume:
  $Y$ : the $n*r$ labeling matrix, where $y_{ij}$ represents the probability of vertex $x_i (i = 1 \ldots n)$ with label $r_j (j = 1 \ldots r)$ (including the non-relation label);
  $Y_L$ : the top $l$ rows of $Y^0$. $Y_L$ corresponds to the $l$ labeled instances;
  $Y_U$ : the bottom $u$ rows of $Y^0$. $Y_U$ corresponds to the $u$ unlabeled instances;
  $\bar{T}$ : a $n*n$ matrix, with $\bar{t}_{ij}$ is the probability jumping from vertex $x_i$ to vertex $x_j$;

BEGIN (the algorithm)
  Initialization:
    1) Set the iteration index $t = 0$;
    2) Let $Y^0$ be the initial soft labels attached to each vertex;
    3) Let $Y_L^0$ be consistent with the labeling in the labeled (including all the relation (sub)types and the non-relation) data, where $y_{ij}^0$ = the weight of the bootstrapped support vector if $x_i$ has label $r_j$ (Please note that $r_j$ can be the non-relation label) and 0 otherwise;
    4) Initialize $Y_U^0$;
  REPEAT
    Propagate the labels of any vertex to nearby vertices by $Y^{t+1} = \bar{T}Y^t$;
    Clamp the labeled data, that is, replace $Y_L^{t+1}$ with $Y_L^0$;
  UNTIL $Y$ converges(e.g. $Y_L^{t+1}$ converges to $Y_L^0$);
  Assign each unlabeled instance with a label: for $x_i (l \prec i \le n)$, find its label with $\arg\max_j y_{ij}$;
END (the algorithm)
```

Figure 2: The LP algorithm

Here, each vertex corresponds to an instance, and the edge between any two instances x_i and x_j is weighted by w_{ij} to measure their similarity. In principle, larger edge weights allow labels to travel through easier. Thus the closer the instances are, the more likely they have similar labels. The algorithm first calculates the weight w_{ij} using a kernel, then transforms it to $t_{ij} = p(j \to i) = w_{ij} / \sum_{k=1}^{n} w_{kj}$, which measures the probability of propagating a label from instance x_j to instance x_i, and finally normalizes t_{ij} row by row using $\bar{t}_{ij} = t_{ij} / \sum_{k=1}^{n} t_{ik}$ to maintain the class probability interpretation of the labeling matrix Y.

During the label propagation process, the label distribution of the labeled data is clamped in each loop using the weights of the bootstrapped support vectors and acts like forces to push out labels through the unlabeled data. With this push originates from the labeled data, the label boundaries will be pushed much faster along edges with larger weights and settle in gaps along those with lower weights. Ideally, we can expect that w_{ij} across different classes should be as small as possible and w_{ij} within the same class as big as possible. In this way, label propagation happens within the same class most likely.

This algorithm has been shown to converge to a unique solution (Zhu and Ghahramani 2002), which can be obtained without iteration in theory, and the initialization of Y_U^0 (the unlabeled data) is not important since Y_U^0 does not affect its estimation. However, proper initialization of Y_U^0 actually helps the algorithm converge more rapidly in practice. In this paper, each row in Y_U^0 is initialized to the average similarity with the labeled instances.

4 Experimentation

This paper uses the ACE RDC 2003 corpus provided by LDC for evaluation. This corpus is gathered from various newspapers, newswires and broadcasts.

Method	LP via bootstrapped (weighted) SVs	LP via bootstrapped (un-weighted) SVs	LP w/o SVM bootstrapping	SVM	(BootProject) SVM Bootstrapping
5%	46.5 (+1.4)	44.5 (+1.7)	43.1 (+1.0)	35.4 (-)	40.6 (+0.9)
10%	48.6 (+1.7)	46.5 (+2.1)	45.2 (+1.5)	38.6 (-)	43.1 (+1.4)
25%	51.7 (+1.9)	50.4 (+2.3)	49.6 (+1.8)	43.9 (-)	47.8 (+1.7)
50%	53.6 (+1.8)	52.6 (+2.2)	52.1 (+1.7)	47.2 (-)	50.5 (+1.6)
75%	55.2 (+1.3)	54.5 (+1.8)	54.2 (+1.2)	53.1 (-)	53.9 (+1.2)
100%	56.2 (+1.0)	55.8 (+1.3)	55.6 (+0.8)	55.5 (-)	55.8 (+0.7)

Table 1: Comparison of different methods using a state-of-the-art linear kernel on the ACE RDC 2003 corpus (The numbers inside the parentheses indicate the increases in F-measure if we add the ACE RDC 2004 corpus as the unlabeled data)

4.1 Experimental Setting

In the ACE RDC 2003 corpus, the training data consists of 674 annotated text documents (~300k words) and 9683 instances of relations. During development, 155 of 674 documents in the training set are set aside for fine-tuning. The test set is held out only for final evaluation. It consists of 97 documents (~50k words) and 1386 instances of relations. The ACE RDC 2003 task defines 5 relation types and 24 subtypes between 5 entity types, i.e. person, organization, location, facility and GPE. All the evaluations are measured on the 24 subtypes including relation identification and classification.

In all our experiments, we iterate over all pairs of entity mentions occurring in the same sentence to generate potential relation instances[1]. For better evaluation, we have adopted a state-of-the-art linear kernel as similarity measurements. In our linear kernel, we apply the same feature set as described in a state-of-the-art feature-based system (Zhou et al 2005): word, entity type, mention level, overlap, base phrase chunking, dependency tree, parse tree and semantic information. Given above various lexical, syntactic and semantic features, multiple overlapping feature views are generated in the bootstrapping process using random feature projection (Zhang 2004). For each feature projection in bootstrapping support vectors, a feature is randomly selected with probability p and therefore the eventually projected feature space has $p*F$ features on average, where F is the size of the original feature space. In this paper, p and the number of different views are fine-tuned to 0.5 and 10 [2] respectively using 5-fold cross validation on the training data of the ACE RDC 2003 corpus.

4.2 Experimental Results

Table 1 presents the F-measures[3] (the numbers outside the parentheses) of our algorithm using the state-of-the-art linear kernel on different sizes of the ACE RDC training data with all the remaining training data and the test data[4] as the unlabeled data on the ACE RDC 2003 corpus. In this paper, we only report the performance (averaged over 5 trials) with the percentages of 5%, 10%, 25%, 50%, 75% and 100%[5]. For example, our LP algorithm via bootstrapped (weighted) support vectors achieves the F-measure of 46.5 if using only 5% of the ACE RDC 2003 training data as the labeled data and the remaining training data and the test data in this corpus as the unlabeled data. Table 1

[1] In this paper, we only measure the performance of relation extraction on “true” mentions with “true” chaining of co-reference (i.e. as annotated by the corpus annotators) in the ACE corpora. We also explicitly model the argument order of the two mentions involved and only model explicit relations because of poor inter-annotator agreement in the annotation of implicit relations and their limited number.

[2] This suggests that the modified BootProject algorithm in the bootstrapping phase outperforms the SelfBoot algorithm (with p=1.0 and m=1) which uses all the features as the only view. In the related NLP literature, co-training has also shown to typically outperform self-bootstrapping.

[3] Our experimentation also shows that most of performance improvement with either bootstrapping or label propagation comes from gain in recall. Due to space limitation, this paper only reports the overall F-measure.

[4] In our label propagation algorithm via bootstrapped support vectors, the test data is only included in the second phase (i.e. the label propagation phase) and not used in the first phase (i.e. bootstrapping support vectors). This is to fairly compare different semi-supervised learning methods.

[5] We have tried less percentage than 5%. However, our experiments show that using much less data will suffer from performance un-stability. Therefore, we only report the performance with percentage not less than 5%.

also compares our method with SVM and the original SVM bootstrapping algorithm BootProject(i.e. bootstrapping on the top of SVM with feature projection, as proposed in Zhang (2004)). Finally, Table 1 compares our LP algorithm via bootstrapped (weighted by default) support vectors with other possibilities, such as the scheme via bootstrapped (un-weighted, i.e. the importance of support vectors is not differentiated) support vectors and the scheme via all the available labeled data (i.e. without SVM bootstrapping). Table 1 shows that:

1) Inclusion of unlabeled data using semi-supervised learning, including the SVM bootstrapping algorithm BootProject, the normal LP algorithm via all the available labeled and unlabeled data without SVM bootstrapping, and our LP algorithms via bootstrapped (either weighted or un-weighted) support vectors, consistently improves the performance, although semi-supervised learning has shown to typically decrease the performance when a lot of (enough) labeled data is available (Nigam 2001). This may be due to the insufficiency of labeled data in the ACE RDC 2003 corpus. Actually, most of relation subtypes in the two corpora much suffer from the data sparseness problem (Zhou et al 2006).
2) All the three LP algorithms outperform the state-of-the-art SVM classifier and the SVM bootstrapping algorithm BootProject. Especially, when a small amount of labeled data is available, the performance improvements by the LP algorithms are significant. This indicates the usefulness of the manifold structure in both labeled and unlabeled data and the powerfulness of the LP algorithm in modeling such information.
3) Our LP algorithms via bootstrapped (either weighted or un-weighted) support vectors outperforms the normal LP algorithm via all the available labeled data w/o SVM bootstrapping. For example, our LP algorithm via bootstrapped (weighted) support vectors outperforms the normal LP algorithm from 0.6 to 3.4 in F-measure on the ACE RDC 2003 corpus respectively when the labeled data ranges from 100% to 5%. This suggests that the manifold structure in both the labeled and unlabeled data can be well preserved via bootstrapped support vectors, especially when only a small amount of labeled data is available. This implies that weighted support vectors may represent the manifold structure (e.g. the decision boundary from where label propagation is done) better than the full set of data – an interesting result worthy more quantitative and qualitative justification in the future work.
4) Our LP algorithms via bootstrapped (weighted) support vectors perform better than LP algorithms via bootstrapped (un-weighted) support vectors by ~1.0 in F-measure on average. This suggests that bootstrapped support vectors with their weights can better represent the manifold structure in all the available labeled and unlabeled data than bootstrapped support vectors without their weights.
5) Comparison of SVM, SVM bootstrapping and label propagation with bootstrapped (weighted) support vectors shows that both bootstrapping and label propagation contribute much to the performance improvement.

Table 1 also shows the increases in F-measure (the numbers inside the parentheses) if we add all the instances in the ACE RDC 2004[6] corpus into the ACE RDC 2003 corpus in consideration as unlabeled data in all the four semi-supervised learning methods. It shows that adding more unlabeled data can consistently improve the performance. For example, compared with using only 5% of the ACE RDC 2003 training data as the labeled data and the remaining training data and the test data in this corpus as the unlabeled data, including the ACE RDC 2004 corpus as the unlabeled data increases the F-measures of 1.4 and 1.0 in our LP algorithm and the normal LP algorithm respectively. Table 1 shows that the contribution grows first when the labeled data begins to increase and reaches a maximum of ~2.0 in F-measure at a certain point.

Finally, it is found in our experiments that critical and hard instances normally occupy only 15~20% (~18% on average) of all the available labeled and unlabeled data. This suggests that, through bootstrapped support vectors, our LP algo-

[6] Compared with the ACE RDC 2003 task, the ACE RDC 2004 task defines two more entity types, i.e. weapon and vehicle, much more entity subtypes, and different 7 relation types and 23 subtypes between 7 entity types. The ACE RDC 2004 corpus from LDC contains 451 documents and 5702 relation instances.

rithm can largely reduce the computational burden since it only depends on the critical instances (i.e. bootstrapped support vectors with their weights) and those hard instances.

5 Conclusion

This paper proposes a new effective and efficient semi-supervised learning method in relation extraction. First, a moderate number of weighted support vectors are bootstrapped from all the available labeled and unlabeled data via SVM through a co-training procedure with feature projection. Here, a random feature projection technique is used to generate multiple overlapping feature views in bootstrapping using a state-of-the-art linear kernel. Then, a LP algorithm is applied to propagate labels via the bootstrapped support vectors, which, together with those hard unlabeled instances and the test instances, are represented as vertices in a connected graph. During the classification process, the label information is propagated from any vertex to nearby vertex through weighted edges and finally the labels of unlabeled instances are inferred until a global stable stage is achieved. In this way, the manifold structure in both the labeled and unlabeled data can be well captured by label propagation via bootstrapped support vectors. Evaluation on the ACE RDC 2004 corpus suggests that our LP algorithm via bootstrapped support vectors can take the advantages of both SVM bootstrapping and label propagation.

For the future work, we will systematically evaluate our proposed method on more corpora and explore better metrics of measuring the similarity between two instances.

Acknowledgement

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

References

ACE. (2000-2005). Automatic Content Extraction. http://www.ldc.upenn.edu/Projects/ACE/

Agichtein E. and Gravano L. (2000). Snowball: Extracting relations from large plain-text collections. Proceedings of the 5th ACM International Conference on Digital Libraries (ACMDL'2000).

Belkin, M. and Niyogi, P. (2002). Using Manifold Structure for Partially Labeled Classification. *NIPS 15*.

Blum A. and Chawla S. (2001). Learning from labeled and unlabeled data using graph mincuts. *ICML'2001*.

Blum A., Lafferty J., Rwebangira R and Reddy R. (2004). Semi-supervised learning using randomized mincuts. *ICML'2004*.

Brin S. (1998). Extracting patterns and relations from world wide web. *Proceedings of WebDB Workshop at 6th International Conference on Extending Database Technology*:172-183.

Charniak E. (2001). Immediate-head Parsing for Language Models. *ACL'2001*: 129-137. Toulouse, France

Chen J.X., Ji D.H., Tan C.L. and Niu Z.Y. (2006). Relation extraction using label propagation based semi-supervised learning. *COLING-ACL'2006*: 129-136. July 2006. Sydney, Australia.

Culotta A. and Sorensen J. (2004). Dependency tree kernels for relation extraction. *ACL'2004*. 423-429. 21-26 July 2004. Barcelona, Spain.

Hasegawa T., Sekine S. and Grishman R. (2004). Discovering relations among named entities form large corpora. *ACL'2004*. Barcelona, Spain.

Miller S., Fox H., Ramshaw L. and Weischedel R. (2000). A novel use of statistical parsing to extract information from text. *ANLP'2000*. 226-233. 29 April - 4 May 2000, Seattle, USA

Moschitti A. (2004). A study on convolution kernels for shallow semantic parsing. *ACL'2004*:335-342.

Nigam K.P. (2001). Using unlabeled data to improve text classification. Technical Report CMU-CS-01-126.

Niu Z.Y., Ji D.H., and Tan C.L. (2005). Word Sense Disambiguation Using Label Propagation Based Semi-supervised Learning. ACL'2005:395-402., Ann Arbor, Michigan, USA.

Szummer, M., & Jaakkola, T. (2001). Partially Labeled Classification with Markov Random Walks. *NIPS 14*.

Yang L.P., Ji D.H., Zhou G.D. and Nie Y. (2006). Document Re-ranking using cluster validation and label propagation. *CIKM'2006*. 5-11 Nov 2006. Arlington, Virginia, USA.

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

Zhang M., Su J., Wang D.M., Zhou G.D. and Tan C.L. (2005). Discovering Relations from a Large Raw Corpus Using Tree Similarity-based Clustering, *IJCNLP'2005, Lecture Notes in Artificial Intelligence (LNAI 3651)*. 378-389.

Zhang M., Zhang J., Su J. and Zhou G.D. (2006). A Composite Kernel to Extract Relations between Entities with both Flat and Structured Features. *COLING-ACL-2006*: 825-832. Sydney, Australia

Zhang Z. (2004). Weakly supervised relation classification for information extraction. CIKM'2004. 8-13 Nov 2004. Washington D.C. USA.

Zhao S.B. and Grishman R. (2005). Extracting relations with integrated information using kernel methods. *ACL'2005*: 419-426. Univ of Michigan-Ann Arbor， USA， 25-30 June 2005.

Zhou G.D., Su J. Zhang J. and Zhang M. (2005). Exploring various knowledge in relation extraction. *ACL'2005*. 427-434. 25-30 June, Ann Arbor, Michgan, USA.

Zhou G.D., Su J. and Zhang M. (2006). Modeling commonality among related classes in relation extraction, *COLING-ACL'2006*: 121-128. Sydney, Australia.

Zhu, X. and Ghahramani, Z. (2002). Learning from Labeled and Unlabeled Data with Label Propagation. *CMU CALD Technical Report*. CMU-CALD-02-107.

Zhu, X., Ghahramani, Z. and Lafferty, J. (2003). Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. *ICML'2003*.