Tree Kernel-based Protein-Protein Interaction Extraction Considering both Modal Verb Phrases and Appositive Dependency Features

Changlin Ma
School of Computer
Central China Normal University
Wuhan, Hubei 430079, China
+862767868318
clma@mail.ccnu.edu.cn

Yong Zhang
School of Computer
Central China Normal University
Wuhan, Hubei 430079, China
+862767868318
ychang@mail.ccnu.edu.cn

Maoyuan Zhang
School of Computer
Central China Normal University
Wuhan, Hubei 430079, China
+862767868318
zhangmy@mail.ccnu.edu.cn

ABSTRACT

Protein-protein interaction plays an important role in understanding biological processes. In order to resolve the parsing error resulted from modal verb phrases and the noise interference brought by appositive dependency, an improved tree kernel-based PPI extraction method is proposed in this paper. Both modal verbs and appositive dependency features are considered to define some relevant processing rules which can effectively optimize and expand the shortest dependency path between two proteins in the new method. On the basis of these rules, the effective optimization and expanding path is used to direct the cutting of constituent parse tree, which makes the constituent parse tree for protein-protein interaction extraction more precise and concise. The experimental results show that the new method achieves better results on five commonly used corpora.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Systems *–Textual databases;* H.2.8 [**Database Management**]: Database Applications *–Data mining;* I.2.6 [**Artificial Intelligence**]: Learning *–Knowledge acquisition*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing *–Language parsing and understanding, Text analysis.*

Keywords

Protein-protein interaction extraction, Modal verb phrases, Appositive dependency features, EOEP-CPT algorithm.

1. INTRODUCTION

Protein-protein interaction (PPI) reveals the molecular mechanisms of biological processes and is critical for the understanding of vital movement. In recent years, various kinds of biomedical research papers regarding protein interaction have been rapidly expanding, which makes it difficult for researchers to find related PPI information from a mass of biomedical literature. Furthermore, most of the existing PPI databases are manually crafted, such as BIND [1], MINT [2], and IntAct [3]. The rapidly growing number of literature brings inconvenience to the update and maintenance of these databases. Therefore, automatic extraction of PPI information has been a widely researched topic in the biomedical natural language processing (BioNLP) field.

Early studies employed rule-based method in which the system performance depended on the quality and size of predefined rules. Alert et al. [4] adopted the rule of co-occurrence where two protein names and an interaction vocabulary were in one sentence, which only achieved 22% P-score. Huang et al. [5] attempted to automatically extract PPI patterns from training corpora with part of speech tagging and achieved good performance.

With the advent of more and more sophisticated and powerful NLP techniques, machine learning methods have been widely used for the task of PPI extraction, which are divided into feature-based methods and kernel-based methods. However, feature-based methods [6-10] cannot avoid the complex construction and mapping process of feature vectors. Therefore, current researches focus on kernel methods due to their capability to directly use structured information. Several kernels are proposed for PPI extraction, mainly including subsequence kernels [11], tree kernels [12], and graph kernels [13] etc.

Bunescu and Mooney [14] first proposed the idea of using kernel methods to extract PPI and a kernel based on the shortest dependency path was adopted for PPI extraction in their paper. Erkan et al. [15] defined cosine similarity and edited distance functions over dependency paths between two entities for PPI extraction via semi-supervised learning. Airola et al. [13] employed an all-dependency-paths graph kernel to comprehensively analyze the complex dependency relationships among words and achieved great success. However, their system had the problem of high time complexity and difficulty in calculation and implementation. Kim et al. [16] introduced a walk-weighted subsequence kernel based on previous research to explore the effect of various substructures for PPI extraction. Miwa et al. [17] proposed a method to combine multiple kernels for the purpose of improving performance.

Most of the above kernels are based on the dependency information derived from sentences, which shows that dependency information plays a critical role in PPI extraction. Nevertheless, taking only dependency information into consideration is not enough because of the complexity of biological information expression. The constituent parse tree (CPT) should be considered because it includes rich syntactic and structural features which are important for structured information representation of PPI extraction.

In order to solve the aforementioned problems, some researchers used tree kernels over CPT for semantic relation extraction in newswire domain. After analyzing and comparing five tree setups, Zhang et al. [18] concluded that the tree kernel using the shortest path-enclosed tree (SPT) as structured information achieved the best performance. SPT is a part of CPT enclosed by two entities. Although this cutting strategy has certain rationality for PPI

Copyright is held by the International World Wide Web Conference Committee (IW3C2). IW3C2 reserves the right to provide a hyperlink to the author's site if the Material is used in electronic media.
WWW 2015 Companion, May 18–22, 2015, Florence, Italy.
ACM 978-1-4503-3473-0/15/05.
http://dx.doi.org/10.1145/2740908.2741705

extraction, most of the sentences in biomedical literature are too long and the entities span a relatively long distance [19], which cause some noise information unrelated to PPI extraction or even interfering with PPI extraction remain in SPT.

For making better use of structured information, Zhou et al. [20] extended SPT by including some necessary predicate-linked information and then proposed the context-sensitive SPT (CS-SPT), which further improved the performance of PPI extraction. CS-SPT may be effective for sentences in which two entities are closed to each other, but it lacks guidance for the extension of complex sentences. Furthermore, the extended context information related to predicate link may not be associated with PPI extraction.

For the purpose of overcoming the shortcomings of SPT and CS-SPT, Qian et al. [21] adopted dependency information to generate a dynamic relation tree (DRT). In their method, the dependencies of ACE corpus were first divided into five categories. Then some hand-crafted rules according to these dependency types were used to determine what information should be retained and what information should be removed from CPT. However, many problems still exist in CPT. Firstly, these rules are manually crafted based on news corpora, which may not be applicable to the biomedical domain. For example, some dependencies are important for semantic relation extraction in newswire domain, but it may be useless for PPI extraction. Secondly, it is too coarse to divide the dependencies into only five categories, which fails to reflect the difference between similar dependency types.

Introducing CPT into newswire domain has achieved great success in the task of semantic relation extraction. However, it fails to reach the expected performance in PPI extraction of biomedical researches due to the unique writing style of biomedical literature in which protein entities span a long distance in the parse tree and the relationship between entities even spans several clauses. They may cause the obtained structure information to be filled with much noise information unassociated with PPI extraction.

In order to resolve these problems, Qian et al. [22] combined dependency information with CPT and proposed to employ the shortest dependency path (SDP) for the cutting of CPT to make the generated tree structure representation more applicable to PPI extraction. The approach was motivated by the following two aspects: one was the importance of the SDP to PPI extraction, the other was the great success of employing dependency information to cut CPT in newswire domain. The main idea of their shortest dependency path-directed constituent parse tree (SDP-CPT) algorithm was that only the words appearing on the SDP and their associated constituents in CPT were kept as the part of final tree structure, which achieved significant results over several commonly used corpora. Nevertheless, several shortcomings still exist in the structured information representation of SDP-CPT algorithm. Some noise information still remains in the process of cutting CPT via SDP while some critical information expressing PPI was omitted.

Considering the aforementioned disadvantages, an improved tree kernel-based PPI extraction method is proposed in this paper. Some relevant processing rules are defined to effectively optimize and expand SDP between two proteins in the new method. These processing rules are aimed at resolving two problems: (1) The missing of key verbs resulted from the parsing error of modal verb phrases; (2) The noise interference brought by appositive dependency. The proposed processing rules can remove noise information while retaining critical information, which makes the effective optimization and expanding path-directed constituent parse tree (EOEP-CPT) more precise and concise.

2. EOEP-CPT ALGORITHM DESCRIPTION

The SDP-CPT algorithm [22] took advantage of SDP between two proteins to direct the cutting of CPT, in which only the words appearing on SDP and their associated constituents were kept. It combined the simplicity of SDP information with rich structural features inherent in CPT. Generally speaking the CPT structure generated by SDP-CPT algorithm can accurately express PPI. However, when there are modal verb phrases (such as “be able to”) between two protein entities to represent PPI relationship, the shortest dependency path in SDP-CPT may miss the verbs behind the modal verb phrases, which tend to be critical for expressing PPI. Moreover, SDP fails to remove the noise interference brought by the “appos” dependency type. The following example is used to illustrate the above limitations of SDP-CPT.

Example 1: A bacterially expressed 318-amino acid fragment, PROT1 (418-736), containing the amphipathic helix region, was able to bind PROT2.

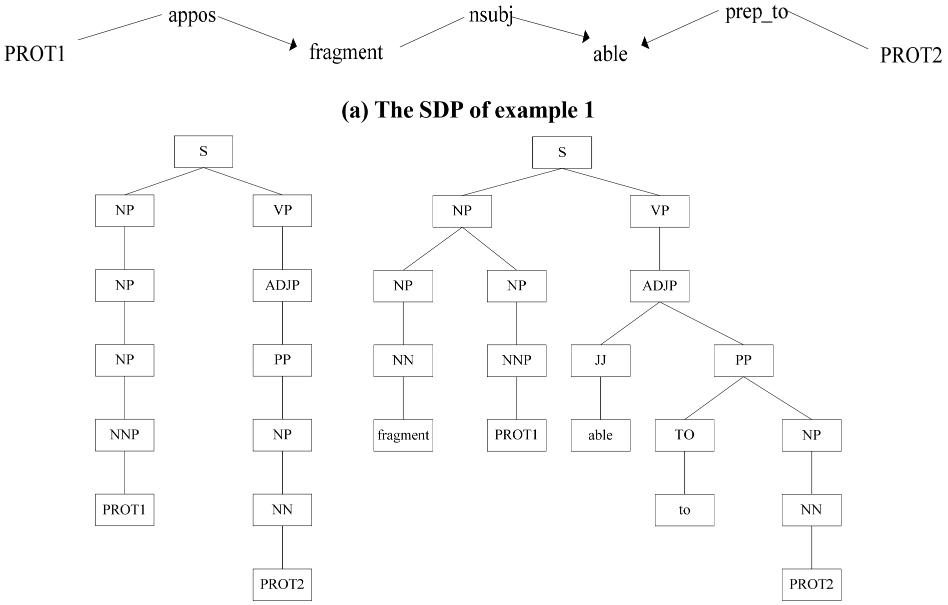
The real names of two proteins are replaced with PROT1 and PROT2. Figure 1 depicts the processing results of SDP-CPT algorithm on example 1.

The SDP and shortest constituent path (SCP) generated by the Stanford parser are shown in Figure 1 (a) and (b) respectively. According to SDP-CPT algorithm, all the words in SDP and their associated constituents in CPT should be added to SCP. Apart from PROT1 and PROT2, only “fragment” and “able” two words appear in SDP of example 1. Thus, the words, “fragment” and “able”, and their associated constituents need to be added to SCP. As a result, two paths, “fragment→NN→NP→NP→S” and “able→JJ→ADJP →VP→S”, are linked to SCP. Since the dependency relation type between “able” and “PROT2” is “prep_to”, the preposition “to” and its associated constituents should also be included in SCP. The final step is to merge two consecutive NP/VP nodes along the SDP-CPT paths into a single one when the parent node has only one child node. In the obtained tree structure, there are two consecutive NP nodes on the path from PROT1 to S. The merged SDP-CPT is shown in Figure 1 (c). It is easy to see that the SDP-CPT tree structure is unable to represent the interaction relationship between PROT1 and PROT2 because SDP ignores the key word “bind” which can express interaction. Moreover, there is still noise information like “fragment” in SDP-CPT. The noise information not only is unhelpful for PPI extraction, but also increases the complexity of tree structure for kernels.

In order to solve the above problems, some improvements are made on SDP-CPT. An effective optimization and expanding path-directed constituent parse tree (EOEP-CPT) is proposed in this paper. At first a related definition is given as follows.

Definition 1: The symbolic words of commonly used modal verb phrases are defined as modal-verb-phrase keywords. The set composed of these keywords is defined as modal-verb-phrase-keyword set which is denoted as MVPK. For example, the words “able”, “ability” and “necessary” in modal verb phrases “be able to”, “ability to” and “be necessary to” are keywords.

The main idea of EOEP-CPT is described as follows. Firstly, the Stanford parser is applied to generate CPT and dependency graph (DG). The direct output results from the dependency parser are SD CCprocessed relation tuples, which should be further converted into DG for later processing.



(a) The SDP of example 1

(b) The SCP of example 1 **(c) The SDP-CPT of example 1**

Figure 1. The processing results of SDP-CPT algorithm

On the basis of generating CPT, SCP between two proteins is extracted from CPT, where SCP is the path starting from the two proteins to find their common ancestor node along CPT. SCP only contains two protein nodes and their constituents in CPT and does not include any dependency information. It is the simplest tree structure and the initial value of EOEP-CPT is set to SCP. Then EOEP-CPT finds SDP between two proteins from DG by using the shortest path method in graph theory. The path with length 1 will not be taken into account since this path cannot provide any information about PPI and makes no sense to PPI. Therefore, when there is a line connecting two proteins, the line should be deleted.

From the above discussions we know that there are two limitations of SDP-CPT: 1. the appositive dependency relation produces noise interference; 2. the parsing error of modal verb phrases leads to the missing of key information. For the purpose of resolving the two problems, the following processing rules are defined in EOEP-CPT to optimize and expand the generated SDP.

(1) For limitation 1, the algorithm first judges whether an appositive dependency relation appears in SDP or not. If it actually appears in SDP, this case indicates that the two words connected by the dependency relation are the same semantic constituent for the sentence. It will produce useless noise interference that the two words simultaneously appear in SDP. Thus, one of them should be removed from SDP. In EOEP-CPT algorithm, the removing rules consist of the following three cases:

(a) If the appositive dependency relation appears in the form of PROT1 → appos → word or PROT2 → appos → word, the relative position of PROT1/PROT2 and the word is generally like “word, PROT1/PROT2” in the original sentence. In this case, PROT1/PROT2 is the summary of the word and they all refer to the same thing. Therefore, the word should be removed from the SDP. Accordingly, the “appos” dependency relation should be removed from dependency type set (SDT).

(b) If the appositive dependency relation appears in the form of PROT1 ← appos ← word or PROT2 ← appos ← word, the relative position of PROT1/PROT2 and the word is generally like “PROT1/PROT2, word” in the original sentence. In this case, the word is the supplement of PROT1/PROT2 and expresses useful information about the proteins. For example, in the sentence “In addition, a possible role for PROT1, the product of PROT2”, the appositive “product” represents the relationship between PROT1 and PROT2. Thus, PROT1/PROT2 and the word cannot be deleted.

(c) If the appositive dependency relation appears in the form of PROT1 →……word1 → appos → word2 or PROT2 →……word1 → appos → word2, the effect of deleting word1 or word2 is similar in this case. In EOEP-CPT algorithm, the word which is far from PROT1/PROT2 will be deleted from the SDP because this kind of relationship is weaker.

(2) For limitation 2, the algorithm first judges whether SDP contains some keywords in set MVPK or not. If not, nothing will be done; otherwise, some measures will be taken to find the verb behind the keyword from the original sentence. Secondly, the algorithm judges whether the verb appears in SDP. If it is true, noting will be done; otherwise, the verb should be inserted into the back of the keyword.

The SDP processed through the above rules is denoted as the effective optimization and expanding path (EOEP). The set of dependency types is denoted as EOET. Then EOEP is used to direct the cutting of CPT. For each word in EOEP except PROT1 and PROT2, its corresponding node is found from CPT and the path from the node to the root of SCP is added to EOEP-CPT. In addition, if the dependency relation between two words is “prep_xx”, the path from the “xx” to the root of SCP should also be added to EOEP-CPT because this kind of dependency relation is important for PPI extraction. The final step is to merge two consecutive NP/VP nodes along EOEP-CPT paths into a single one when the parent node has only one child node. Figure 2 describes the specific steps of EOEP-CPT algorithm.

Now we still use example 1 to demonstrate the EOEP-CPT generating process. From SDP in Figure 1 (a) we can conclude that the dependency relation between PROT1 and “fragment” is “appos”. According to EOEP-CPT, the word “fragment” and dependency relation “appos” should be removed from SDP while PROT1 is connected to “able”. Furthermore, because the keyword “able” appears in SDP while the verb “bind” behind it doesn’t appear in SDP, the verb “bind” needs to be inserted into the back of “able”. Meanwhile, the dependency relation between “able” and “bind” is replaced with “xcomp” and that between “bind” and “PROT2” is replaced with “dobj”. The processing results are shown in Figure 3. Compared with SDP in Figure 1 (a), EOEP in Figure 3 (a) retains the key information “bind” while it removes the noise interference “fragment”. Compared with SDP-CPT in Figure 1 (c), it is more accurate and concise to use the EOEP-CPT structure (in Figure 3 (c)) representing the PPI information.

3. EXPERIMENTS

In this paper, we select five widely used corpora in related work as the experimental data sets: AIMed [19], BioInfer [23], HPRD50 [24], IEPA [25] and LLL [26]. These PPI corpora lack unified annotation standard, which will lead to the large difference and incompatibility of the corpora format. For the convenience of comparison, we use the conversion system [27] to convert all the corpora into the unified XML format.

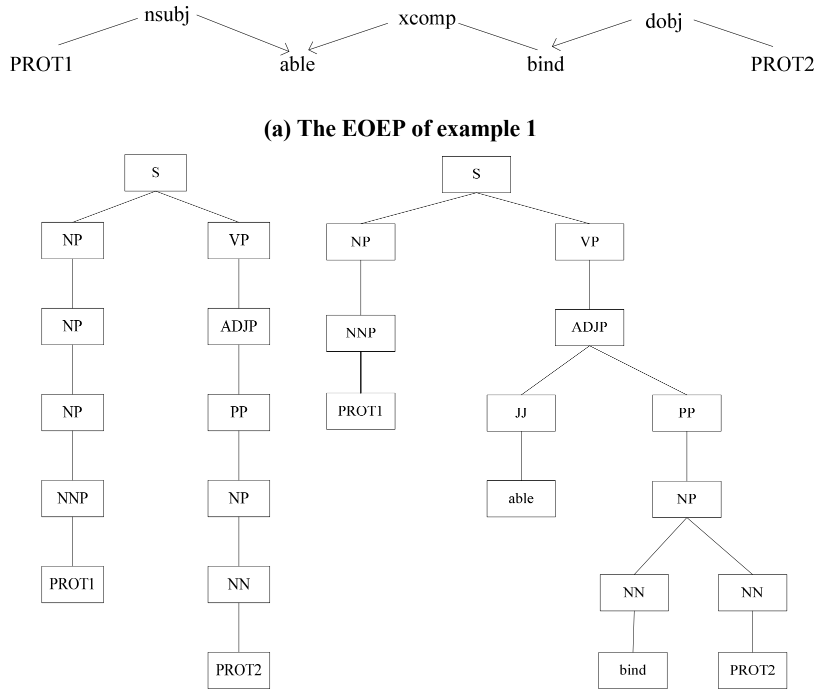
For the preprocessing of corpora, the paper adopts the same strategies [22] as most related work, in which all self-interaction instances are removed and nest protein names are retained in all corpora. For a sentence with multiple protein entities, the involved two proteins are replaced with PROT1 and PROT2 respectively to hide protein entities and facilitate machine learning.

The Stanford parser is used to generate CPT and SD CCprocessed dependency relation tuples for sentences in the above corpora. Qian et al. [22] demonstrated that SDP in the form of SD CCprocessed had the best performance for PPI extraction among the four Stanford dependency schemes. Furthermore, this paper employs the latest version of Stanford parser whose performance is different from that of the old version used in [22]. All the following experimental results are conducted on the new version. In this paper, we choose SVMLight [28] classifier with convolution tree kernel functions as the classifier.

In order to make full use of the data set, we adopt the widely used 10-fold document-level cross-validation. For the problem of considering the multiple occurrences of a PPI as one relation pair or multiple pairs, we employ OAOD (One Answer per Occurrence in a Document) strategy, which means that each occurrence is taken as a PPI instance. The commonly used evaluation metrics for PPI extraction are Precision (P), Recall (R) and their harmonic mean F-score (F). However, according to [13], F-score has a critical weakness that it’s sensitive to the distribution of positive instances and negative instances in corpora. Even for the same PPI extraction system, F-score on various corpora is of great difference. As a substitute of F-score, AUC [29] (area under the receiver operating characteristics curve), which is invariant to the different distribution of corpora and recommended to evaluate the performance of PPI extraction, is provided in this paper.

```
Algorithm EOEP-CPT
Input: a sentence S and two proteins PROT1 and PROT2 in it
Output: EOEP-CPT
Steps:
(1) For given S, generate the constituent parse tree CPT and the dependency graph DG using the
    Stanford parser
(2) Extract the shortest constituent path SCP between PROT1 and PROT2 from DG, denoted as
    SCP={$scp_i$}, i=1,⋯,L with its root $scp_r$
(3) Find the shortest dependency path SDP between PROT1 and PROT2 from DG, where the
    path with length 1 is undesirable. The path and its sequence of dependency types are denoted as:
    SDP={$sdp_i$}, i=1,⋯,N, where $sdp_1$=PROT1, $sdp_N$=PROT2
    SDT={$sdt_i$}, i=2,⋯,N, where $sdt_i$ is the dependency relation between $sdp_{i-1}$ and $sdp_i$
(4) For each $sdt_i$=appos in SDT, Repeat
        If PROT1 → appos → word or PROT2 → appos → word, then remove word from SDP and
        appos from SDT
        Else if PROT1 ← appos ← word or PROT2 ← appos ← word, then nothing will be done
        Else if PROT1 →⋯⋯word1 → appos → word2 or PROT2 →⋯⋯word1 → appos → word2,
        then remove the word far away from PORT1 or PORT2
(5) Define a modal verb phrase keyword set MVPK
    If SDP includes a word $sdp_i$ in MVPK, then find the verb v behind it in the original sentence
        If v exists in SDP, then nothing will be done
        Else insert v into the back of $sdp_i$
(6) After the step (4) and (5) treatment, the obtained effectively optimized and expended path and
    dependency types are denoted as EOEP, EOET
(7) Set EOEP-CPT=SCP
(8) For each $eoep_i$ in EOEP—{PROT1, PROT2}, Repeat
      (a) Find the leaf node $n_w$ in CPT corresponding to $eoep_i$
      (b) Add the path from $n_w$ to $scp_r$ into EOEP-CPT
      (c) If the dependency type $eoet_i$ is “prep_xx”, then
            i   Extract the word xx from $eoet_i$
            ii  Find the node $n_{xx}$ in CPT corresponding to xx
            iii Add the path from $n_{xx}$ to $scp_r$ into EOEP-CPT
(9) Merge two consecutive NP/VP nodes along the EOEP-CPT paths into a single one, when the parent
    node has only one child node
(10) Return EOEP-CPT
```

Figure 2. EOEP-CPT algorithm



(a) The EOEP of example 1

(b) The SCP of example 1 **(c) The EOEP-CPT of example 1**

Figure 3. The processing results of EOEP-CPT algorithm

We compare the results obtained by EOEP-CPT with those of using other state-of-the-art PPI extraction systems on the AIMed corpus. Specially, the experimental results of EOEP-CPT and SDP-CPT are conducted on the Stanford parser v2.0.4. The new version of Stanford parser improves the overall performance but decreases the parsing precision, which affects the experimental results of PPI extraction systems. Even so, the PPI extraction system in this paper outperforms systems in [6, 12, 22, 30] which are shown in table 1. Compared with SDP-CPT, P, R and F-score of our method are increased by 1.4%, 1.1% and 1.1% respectively.

The performance comparison of our system and other PPI extraction systems over five corpora is shown in table 2. It indicates that EOEP-CPT achieves better results than SDP-CPT [22] and Tikk [30] on four corpora except HPRD50. Furthermore, the results of EOEP-CPT are better than Airola [13] on three of five corpora although an all-paths graph kernel was applied in [13] with high complexity. It demonstrates that EOEP-CPT algorithm is effective in removing noise interference caused by appositive dependency relation and retaining critical information for improvement of PPI extraction. There is a gap between our method and system of Miwa et al. [31]. However, their system combined multiple parsers and kernels, which had high computation complexity and difficulty in implementation.

Table 1. Performance comparison on the AIMed corpus.

PPI extraction systems	*P*(%)	*R*(%)	*F*
EOEP-CPT	58.2	46.4	51.4
SDP-CPT[22]	56.8	45.3	50.3
Dependency tree[12]	56.9	39.0	46.3
Constituent parse tree[30]	39.2	31.9	34.6
Mitsumori et al. [6]	54.2	42.6	47.7

In order to evaluate the performance of the learning model on other corpora, experiments for cross-corpus are also conducted in this paper. We first train a model on one corpus and then use the trained model to test on the other four corpora. F-score for cross-corpus is shown in table 3 and AUC in table 4. It is obvious that F-score tested on larger corpus (AIMed and BioInfer) are lower than that of other corpora in table 3.

This is because that the model trained on smaller corpus cannot make up for the difference with large corpus. In addition, each corpus has different tagging strategy and is incompatible with each other.

Table 2. Performance comparison on multiple corpora.

	EOEP-CPT		**SDP-CPT**		**Airola**		**Tikk**		**Miwa**	
	F	***AUC***	***F***	***AUC***	***F***	***AUC***	***F***	***AUC***	***F***	***AUC***
AIMed	51.4	79.4	50.3	79.6	56.4	84.8	34.6	77.6	64.2	89.1
BioInfer	62.8	80.9	60.9	80.4	61.3	81.9	47.6	73.3	67.6	86.1
HPRD50	65.1	81.9	65.3	81.7	63.4	79.7	69.7	84.0	69.7	82.8
IEPA	68.7	81.6	68.1	80.3	75.1	85.1	70.7	81.0	74.4	85.6
LLL	82.3	87.2	79.8	85.2	76.8	83.4	79.1	86.8	80.5	86.0

Except HPRD50, the testing results using the models of other corpora are lower than the internal 10-fold cross-validation of the same corpus. This is not surprising because using the trained model to test on the same corpus will certainly achieve better results. The same conclusion can be drawn from table 4.

4. ACKNOWLEDGMENTS

This work is supported by NSFC (National Natural Science Foundation of China) under Grant no. 61003192.

Table 3. F-score for cross-corpus evaluation.

	AIMed	BioInfer	HPRD50	IEPA	LLL
AIMed	51.4	47.3	48.0	41.6	45.6
BioInfer	46.2	62.8	62.1	61.4	62.7
HPRD50	40.4	47.6	64.7	54.2	54.2
IEPA	38.1	53.4	66.3	68.5	67.8
LLL	34.9	47.9	64.2	68.1	82.3

（Row: the training corpus. Column: the testing corpus）

Table 4. AUC for cross-corpus evaluation.

	AIMed	BioInfer	HPRD50	IEPA	LLL
AIMed	79.4	75.8	78.2	67.1	71.8
BioInfer	73.6	80.9	79.3	78.5	80.2
HPRD50	70.5	71.4	81.9	65.0	67.7
IEPA	67.9	72.6	74.3	81.5	77.2
LLL	65.2	71.9	73.4	74.0	87.2

（Row: the training corpus. Column: the testing corpus）

5. REFERENCES

[1] Mathivanan, S., Periaswamy, B., Gandhi, T., Kandasamy, K., Suresh, S., Mohmood, R., Ramachandra,Y., and Pandey, A. 2006. An evaluation of human protein–protein interaction data in the public domain. *BMC Bioinformatics*. 7, 19, Suppl. 5 (2006).

[2] Ceol, A. and Aryamontri C. 2010. MINT, the molecular interaction database: 2009 update. *Nucl Acids Res*. 38, D532 (2010).

[3] Aranda, B., Achuthan, P., and Alam-Faruque, Y. 2010. The IntAct molecular interaction database in 2010. *Nucl Acids Res*. 38, D525 (2010).

[4] Albert, S., Gaudan, S., and Knigge, H. 2003. Computer-assisted generation of a protein-interaction database for nuclear receptors. *Mol Endocrinol*. 17 (2003), 1555-1567.

[5] Huang, M., Zhu, X., and Hao, Y. 2004. Discovering patterns to extract protein-protein interactions from full texts. *Bioinformatics*. 20 (2004), 3604-3612.

[6] Mitsumori, T., Murata, M., Fukuda, Y., Doi, K., and Doi, H. 2006. Extracting protein–protein interaction information from biomedical text with SVM. *IEICE Transactions on Information and Systems*. E89, 8 (2006), 2464-2466.

[7] Giuliano, C., Lavelli, A., and Romano, L. 2006. Exploiting shallow linguistic information for relation extraction from biomedical literature. *In Proceedings of EACL'2006*. (2006), 401-408.

[8] Niu, Y., Otasek, D., and Jurisica, I. 2010. Evaluation of linguistic features useful in extraction of interactions from PubMed: application to annotating known, high-throughput and predicted interactions in I2 D. *Bioinformatics*. 26, 1 (2010), 111-119.

[9] Liu, B., Qian, L. H., Wang, H. L., and Zhou, G.D. 2010. Dependency-driven feature-based learning for extracting protein–protein interactions from biomedical Text. *In Proceedings of COLING'2010*. (2010), 757-765.

[10] Bui, Q. C., Katrenko, S., and Sloot, P. M. 2011. A hybrid approach to extract protein–protein interactions. *Bioinformatics*. 27, 2 (2011), 259-265.

[11] Bunescu, R. and Mooney, R. 2005. Subsequence kernels for relation extraction. *In Proceedings of NIPS'2005*. (2005), 171-178.

[12] Chowdhury, F. M., Lavelli, A., and Moschitti, A. 2011. A study on dependency tree kernels for automatic extraction of protein–protein interaction. *In Proceedings of BioNLP'2011*. (2011), 124-133.

[13] Airola, A., Pyysalo, S., Björne, J., Pahikkala, T., Ginter, F, and Salakoski, T. 2008. All-paths graph kernel for protein–protein interaction extraction with evaluation of cross corpus learning. *BMC Bioinformatics*. 9, Suppl. 1 (2008).

[14] Bunescu, R. and Mooney, R. 2005. A shortest path dependency kernel for relation extraction. *In Proceedings of EMNLP'2005*. (2005), 724-731.

[15] Erkan, G., Özgür, A., and Radev, D. R. 2007. Semi-supervised classification for extracting protein interaction sentences using dependency parsing. *In Proceedings of EMNLP-CoNLL'2007*. (2007), 228-237.

[16] Kim, S., Yoon, J., Yang, J., and Park, S. 2010. Walk-weighted subsequence kernels for protein–protein interaction extraction. *BMC Bioinformatics.* 11 (2010), 107.

[17] Miwa, M., Sætre, R., Miyao, Y., and Tsujii, J. 2009. Protein–protein interaction extraction by leveraging multiple kernels and parsers. *Int J Med Inform 2009*. 78 (2009), 39-46.

[18] 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. *In Proceedings of ACL-COLING'2006*. (2006), 825-832.

[19] Bunescu, R., Ge, R., Kate, R., Marcotte, E., Mooney, R., and Ramani, A. 2005. Comparative experiments on learning information extractors for proteins and their interactions. *J Artif Intell Med*. 33, 2 (2005), 139-155.

[20] Zhou, G. D., Zhang, M., Ji, D. H., and Zhu, Q. M. 2007. Tree kernel-based relation extraction with context-sensitive structured parse tree information. *In Proceedings of EMNLP/CoNLL'2007*. (2007), 728-736.

[21] Qian, L. H, Zhou, G. D., Zhu, Q. M., and Qian, P. D. 2008. Exploiting constituent dependencies for tree kernel-based semantic relation extraction. *In Proceedings of COLING'2008*. (2008), 697-704.

[22] Qian, L. H. and Zhou, G. D. 2012. Tree kernel-based protein-protein interaction extraction from biomedical literature. *Journal of Biomedical Informatics*. 45 (2012), 535-543.

[23] Pyysalo, S., Ginter, F., Heimonen, J., Björne, J., Boberg, J., and Jarvinen, J. 2007. BioInfer: a corpus for information extraction in the biomedical domain. *BMC Bioinformatics*. 8, 50 (2007).

[24] Fundel, K., Küffer, and R., Zimmer, R. 2007. RelEx—relation extraction using dependency parse trees. *Bioinformatics*. 23, 3 (2007), 365-371.

[25] Ding, J., Berleant, D., Nettleton, D., and Wurtele, E. 2002. Miningmedline: abstracts, sentences, or phrases?. *In Pacific Symposium on Biocomputing*. (2002), 326-337.

[26] Nédellec, C. 2005. Learning language in logic-genic interaction extraction challenge. *In Proceedings of the LLL'05 Workshop*. (2005), 97-99.

[27] Pyysalo, S., Airola, A., Heimonen, J., Björne, J., Ginter, F., and Salakoski, T. 2008. Comparative analysis of five protein–protein interaction corpora. *BMC Informatics*. 9, Suppl. 3, S6 (2008).

[28] Joachims, T. 1998. Text categorization with support vector machine: learning with many relevant features. *In Proceedings of ECML'1998*. (1998), 137-142.

[29] Bradley, A. P. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition.* 30, 7 (1997), 1145-1159.

[30] Tikk, D., Thomas, P., Palaga, P., Hakenberg, J., and Leser, U. 2010. A comprehensive benchmark of kernel methods to extract protein–protein interactions from literature. *PLoS Computational Biology*. 6, 7 (2010).

[31] Miwa, M., Sætre, R., Miyao, Y., and Tsujii, J. 2009. A rich feature vector for protein–protein interaction extraction from multiple corpora. *In Proceedings of EMNLP'2009*. (2009), 121-130.