An Interaction Pattern Kernel Approach for Protein-Protein Interaction Extraction from Biomedical Literature

Yung-Chun Chang[1,2], Yu-Chen Su[2], Nai-Wen Chang[1,3], and Wen-Lian Hsu[1]

[1] Institute of Information Science, Academia Simica
No. 128, Sec. 2, Academia Rd., Taipei City 11529, Taiwan (R.O.C)
[2] Department of Information Management, National Taiwan University
No. 1, Sec. 4, Roosevelt Rd., Taipei City 10617, Taiwan (R.O.C)
[3] Graduate Institute of Biomedical Electronics and Bioinformatics
No. 1, Sec. 4, Roosevelt Rd., Taipei City 10617, Taiwan (R.O.C)
{changyc,hsu}@iis.sinica.edu.tw,
{b99705029,d00945020}@ntu.edu.tw

Abstract. Discovering the interactions between proteins mentioned in biomedical literature is one of the core topics of text mining in the life sciences. In this paper, we propose an interaction pattern generation approach to capture frequent PPI patterns in text. We also present an interaction pattern tree kernel method that integrates the PPI pattern with convolution tree kernel to extract protein-protein interactions. Empirical evaluations on LLL, IEPA, and HPRD50 corpora demonstrate that our method is effective and outperforms several well-known PPI extraction methods.

Keywords: Text Mining, Protein-Protein Interaction, Interaction Pattern Generation, Interaction Pattern Tree Kernel.

1 Introduction

With a rapidly growing number of research papers, researchers have difficulty finding the papers that they are looking for. Relationships between entities, mentioned in these papers, can help biomedical researchers find the specific papers they need. Among biomedical relation types, protein–protein interaction (PPI) extraction is becoming critical in the field of molecular biology due to demands for automatic discovery of molecular pathways and interactions in the literature. The goal of PPI extraction is to recognize various interactions, such as transcription, translation, post translational modification, complex and dissociation between proteins, drugs, or other molecules from biomedical literature.

Most PPI extraction methods can be regarded as supervised learning approaches. Given a training corpus containing a set of manually-tagged examples, a supervised classification algorithm is employed to train a PPI classifier to recognize whether an interaction exists in the text segment (e.g., a sentence). Feature-based approaches and

S.-M. Cheng and M.-Y. Day (Eds.): TAAI 2014, LNAI 8916, pp. 36–46, 2014.

kernel-based approaches are frequently used for PPI extraction. Feature-based methods exploit instances of both positive and negative relations in a training corpus to identify effective text features for protein-protein interaction extraction. For instance, Van et al. [16] propose a rich-feature-based kernel which applies feature vectors in combination with automated feature selection for protein-protein interaction extraction. In addition, a co-occurrence-based method is introduced by Airola et al. [1], which explores co-occurrence features of dependency graphs for representing the sentence structure.

However, feature-based methods often have difficulty finding effective features to extract entity relations. In order to address this problem, the kernel-based methods have been proposed to implicitly explore various features in a high dimensional space by employing a kernel to directly calculate the similarity between two objects. In particular, kernel-based methods can be effective in reducing the burden of feature engineering for structured objects in Natural Language Processing (NLP) research, such as the tree structure in PPI extraction. For instance, Erkan et al. [6] define two kernel functions based on the cosine similarity and the edit distance among the shortest paths between protein names in a dependency parse tree. Moreover, Satre et al. [19] develop a system called AkanePPI, which extracts features using the combination of a deep syntactic parser to capture the semantic meaning of the sentences with a shallow dependency parser for the tree kernels, in order to automatically create rules to identify pairs of interacting proteins from a training corpus.

Current research attempt to use tree kernel-based methods mainly due to its capability to effectively utilize the structured information derived from sentences, especially for the constituent dependencies knowledge. Vishwanathan et al. [17] propose a subtree (ST) kernel which considers all common subtrees in the tree representation of two compared sentences. Here a subtree comprises a node with all its descendants in the tree, and two subtrees are identical if labels of the node and order of their children are identical for all nodes. Likewise, Collins et al. [3] introduce a subset tree (SST) kernel that relaxes the constraint that requires all leaves to be included in the substructures at all times. In the meanwhile it preserves the grammatical rules. For a given tree node, either none or all of its children have to be included in the resulting subset tree. In addtion, Moschitti et al. [13] adopt a partial tree kernel (PT) which is more flexible by virtually allowing any tree sub-structures; the only constraint is the order of child nodes must be identical. Both SST and PT kernels are convolution tree kernels. Kuboyama et al. [9] propose a spectrum tree kernel (SpT) which put emphasis on the simplest syntax-tree substructures among these four tree kernels. It compares all directed vertex-walks, that is, sequences of edge connected syntax tree nodes, of length q as the unit of representation. When comparing two protein pairs, the number of shared sub-patterns called tree q-grams are measured as similarity score.

To extract PPI from biomedical literature effectively, we modeled interaction extraction as a classification problem. We proposed an interaction pattern generation approach to capture frequent PPI patterns. Furthermore, to identify interactions between proteins, we developed an interaction pattern tree kernel that integrates the shortest path-enclosed tree (SPT) structure with generated PPI patterns to support vector machines (SVM). The results of experiments demonstrate that the iteractive pattern tree kernel method is effective in extracting PPI. In addition, the proposed interaction pattern generation approach successfully exploits the interaction semantics of text by capturing frequent PPI patterns. Consequently, the method outperforms the tree kernel-based PPI method [3, 9, 13, 17]; the feature-based PPI method [1, 16]; and the shortest path-enclosed tree (SPT) detection method which is widely used to identify relations between named entities.

2 Our System Architecture

Figure 1 shows the proposed interaction extraction method, which is comprised of two key components: *interaction pattern generation* and *interaction pattern tree construction*. We regard interaction extraction as a classification problem. The interaction pattern generation component aims to automatically generate representative patterns of mention interactions between proteins. Then, the interaction pattern tree construction integrates the syntactic and content information with generated interaction patterns for representation of text. Finally, the convolution tree kernel measures similarity between interaction pattern tree structures for SVM to classify interactive expressions. We discuss each component in detail in the following sections.

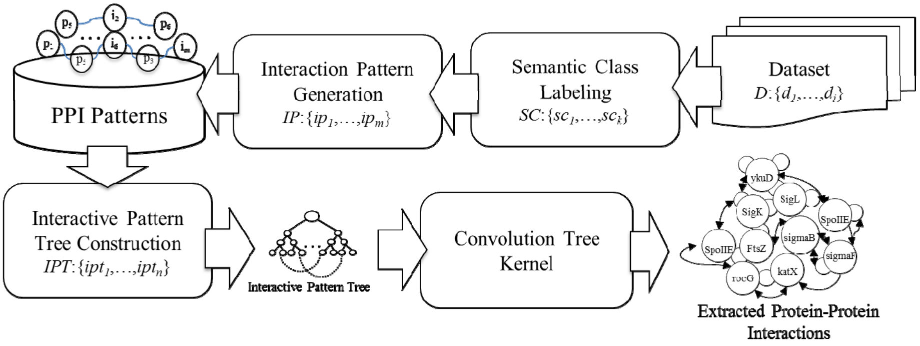


Fig. 1. The interaction extraction method

3 Interaction Pattern Generation

The human perception of a protein-protein interaction is obtained through the recognition of important events or semantic contents to rapidly narrow down the scope of possible candidates. For example, when an expression contains strongly correlated words like "*beta-catenin*", "*alpha-catenin 57-264*" and "*binding*" simultaneously, it is natural to conclude that this is a protein-protein interactive expression, with a less likelihood of a non-interactive one. This phenomenon can explain how humans can skim through an article to quickly capture the interactive expression. In light of this rationale, we proposed an interaction pattern generation approach that aims to automatically generate representative patterns from sequences of expression of protein-protein interactions.

We formulate interaction pattern generation as a frequent pattern mining problem. First of all, the instances undergo the semantic class labeling process. To illustrate the process of semantic class labeling, consider the instance I_n = "*Abolition of the gp130 binding site in hLIF created antagonists of LIF action*", as shown in Fig. 2. First, "*gp130*" and "*hLIF*" are two given protein names, as tagged *PROTEIN1* and *PROTEIN2* respectively. Then, we stem remaining tokens by using porter stemming algorithm [15]. Finally, trigger words "bind" and "antagonist" are labeled with their corresponding types by using our compiled trigger word list which extracts from a BioNLP corpus [8]. Evidently the SCL can group the synonyms together by the same label. This enables us to find distinctive and prominent semantic classes for PPI expression in the following stage.

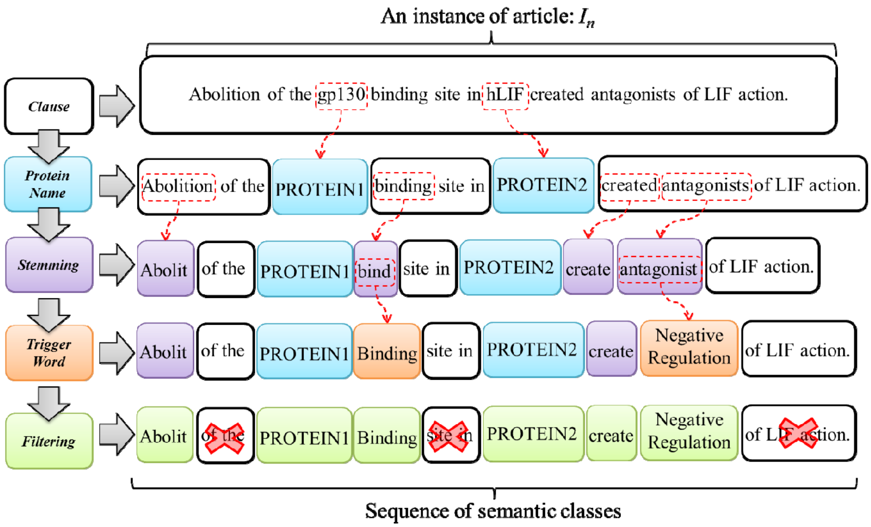

An instance of article: I_n
Clause
Abolition of the gp130 binding site in hLIF created antagonists of LIF action.
Protein Name
Abolition of the PROTEIN1 binding site in PROTEIN2 created antagonists of LIF action.
Stemming
Abolit of the PROTEIN1 bind site in PROTEIN2 create antagonist of LIF action.
Trigger Word
Abolit of the PROTEIN1 Binding site in PROTEIN2 create Negative Regulation of LIF action.
Filtering
Abolit of the PROTEIN1 Binding site in PROTEIN2 create Negative Regulation of LIF action.
Sequence of semantic classes

Fig. 2. Semantic class labeling process

After labeling semantic classes, we based on the co-occurrence of semantic classes to construct a graph to describe the strength of relations between them. Since semantic classes are of an ordered nature, the graph is directed and can be made with association rules. In order to avoid the generation of frames with insufficient length, we empirically set the minimum support of a semantic class as 20 and minimum confidence as 0.5 in our association rules. Thus, an association rule can be represented as (1). Fig. 3 is an illustration of a semantic graph. In this graph, vertices (SC_x) represent semantic classes, and edges represent the co-occurrence of two classes, SC_i and SC_j, where SC_i precedes SC_j. The number on the edge denotes the confidence of two connecting vertices. After constructing all of the semantic graphs, we then generate semantic frames by applying the random walk theory [13] in search of high frequency and representative classes for each topic. Let a semantic graph G be defined as G=(V,E) (|V|=p, |E|=k), a random walk process consisting of a series of random selections on the graph. Every edge (SC_n, SC_m) has its own weight M_{nm}, which denotes the probability of a semantic class SC_n, followed by another class SC_m. For each class, the sum of weight to all neighboring classes N(SC_n) is defined as (2), and the whole graph's probability matrix is defined as (3). As a result, a series of a random walk process becomes a Markov Chain. According to [4], the cover time of a random walk process on a normal graph is $\forall SC_n, C_{SC_n} \leq 4k^2$. We select frequent semantic classes and their neighborhoods as start nodes of a random walk process. We can conclude that using random walk to find frequent patterns on the interactive graph would help us capture even the low probability combinations and shorten the processing time.

$$\text{confidence}(SC_i \Rightarrow SC_j) = P(SC_j|SC_i) = \frac{\text{support}\,(SC_i \cup SC_j)}{\text{support}\,(SC_i)}, \tag{1}$$

where $support_{min}$=20, $confidence_{min}$=0.5

$$\forall SC_n \sum_{m \in N(SC_n)} M_{nm} = 1 \tag{2}$$

$$Pr = [X_{t+1} = SC_m \,|\, X_t = SC_n, X_{t-1} = SC_k, \ldots, X_0 = SC_i] = Pr[X_{t+1} = SC_m \,|\, X_t = SC_n] = M \tag{3}$$

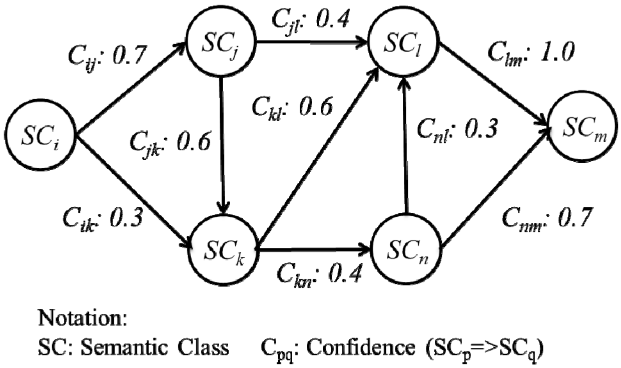


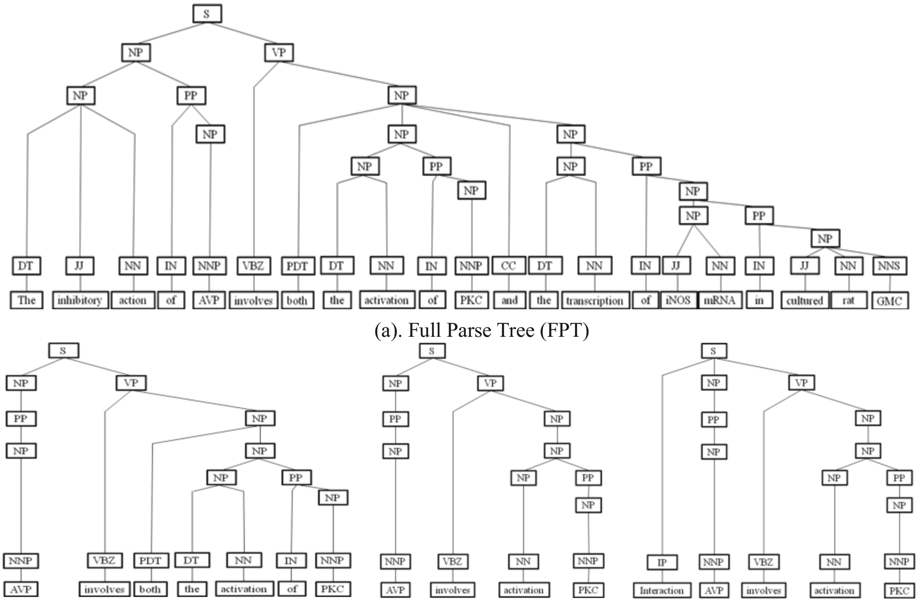
Fig. 3. An interactive graph for pattern generation

Although the random walk process can help us generate frames from frequent patterns in semantic graphs, it can also create some redundancy. Hence, a merging procedure is required to eliminate the redundant results by retaining the patterns, with long length and high coverage, and dispose of bigram patterns that are completely covered by another pattern. For example, the pattern [*PROTEIN1*]->[*Binding*] is completely covered by the pattern [*PROTEIN1*]->[*Binding*]->[*Regulation*]->[*Transcription*]->[*PROTEIN2*]. Thus, the former pattern is incorporated. Otherwise, if a bigram pattern partially overlaps with another, the overlapping part is concatenated to form a longer pattern. For instance, the pattern [*Positive_regulation*]->[*Regulation*] partially overlaps with [*Regulation*]->[*Gene_expression*]->[*PROTEIN1*], thus the two patterns are merged into another single pattern

[*Positive_regulation*]->[*Regulation*]-> [*Gene_expression*]->[*PROTEIN1*]. Moreover, the reduction of the semantic classes space provided by pattern selection is critical. It allows the execution of more sophisticated text classification algorithms, which lead to improved results. Those algorithms cannot be executed on the original semantic classes space because their execution time would be excessively high, making them impractical [1]. Therefore, to select patterns closely associated with an interaction would improve the performance of PPI extraction. We use the pointwise mutual information (PMI) [1], a popular statistical approach used in feature selection, to discriminate semantic classes for PPI instances. Given a training dataset comprised of positive instances, the PMI calculates the likelihood of the occurrence of a semantic class in the expressions of PPI. A semantic class with a large PMI value is thought to be closely associated with the interaction. Lastly, we rank the interaction patterns in the training dataset based on a sum of semantic classes PMI values and retain the top 20 for representing protein-protein interactions.

4 Interaction Pattern Tree Construction

A PPI instance is represented by the interaction pattern tree (IPT) structure, which is the shortest path-enclosed tree (SPT) of the instance enhanced by following steps. To facilitate comprehension of the construction process, the positive instance shown in Fig. 4(a), which mentions the interaction between "AVP" and "PKC", serves as an example.


S
NP VP
NP PP NP
NP NP NP
NP PP NP PP
NP NP
NP PP
NP
DT JJ NN IN NNP VBZ PDT DT NN IN NNP CC DT NN IN JJ NN IN JJ NN NNS
The inhibitory action of AVP involves both the activation of PKC and the transcription of iNOS mRNA in cultured rat GMC
S NP VP PP NP NP NP NP PP NP
NNP VBZ PDT DT NN IN NNP
AVP involves both the activation of PKC
S NP VP PP NP NP NP NP PP NP
NNP VBZ NN NNP
AVP involves activation PKC
S NP VP PP NP NP NP NP PP NP
IP NNP VBZ NN NNP
Interaction AVP involves activation PKC

(a). Full Parse Tree (FPT)

(b). Shortest Path-enclosed Tree (SPT) (c). IPT Pruning (d). IPT Ornamenting

Fig. 4. The interaction pattern tree construction procedure for a PPI instance “The inhibitory action of AVP involves both the activation of PKC and the transcription of iNOS mRNA in cultured rat GMC”

In [15], the authors show that the SPT is effective in identifying the relation between two entities mentioned in a segment of text. Given an instance, therefore we first construct the smallest common sub-tree including the two proteins. In other words, the sub-tree is enclosed by the shortest path linking the two proteins p_i and p_j in the parse tree, which as shown in Fig. 4(b). Next, in order to make the IPT concise and clear, we remove indiscriminative IPT elements. Frequent words are not useful for expressing interactions between proteins. For instance, the word "*both*" in Fig. 4(c) is a common word and cannot discriminate interactive expressions. To remove stop words and the corresponding syntactic elements from the IPT, we sort words according to their frequency in the text corpus. Then, the most frequent words are used to compile a stop word list. Moreover, to refine the list, protein names and verbs are excluded from it because they are key constructs of protein-protein interactions. Finally, the generated interaction patterns can help us capture the most prominent and representative patterns for expressing PPI. Highlighting interaction patterns closely associated with PPIs in an IPT would improve the interaction extraction performance. For each IPT that matched an interaction pattern, we add an IP tag as a child of the tree root to incorporate the interactive semantics into the IPT structure (as shown in Fig. 4(d)).

A convolution kernel aims to capture structured information in terms of substructures. Generally, we can represent a parse tree T by a vector of integer counts of each sub-tree type (regardless of its ancestors):

$$\phi(T) = (\#subtree_1(T), ..., \#subtree_i(T), ..., \#subtree_n(T)), \tag{4}$$

where $\#subtree_i(T)$ is the occurrence number of the i^{th} sub-tree type ($subtree_i$) in T. Since the number of different sub-trees is exponential with the parse tree size, it is computationally infeasible to directly use the feature vector $\phi(T)$. To solve this computational issue, we leverage the convolution tree kernel [3] to capture the syntactic similarity between the above high dimensional vectors implicitly. Specifically, the convolution tree kernel K_{CTK} counts the number of common sub-trees as the syntactic similarity between two rich interactive trees IPT_1 and IPT_2 as follows:

$$K_{CTK}(IPT_1, IPT_2) = \sum_{n_1 \in N_1, n_2 \in N_2} \Delta(n_1, n_2), \tag{5}$$

where N_1 and N_2 are the sets of nodes in IPT_1 and IPT_2 respectively. In addition $\Delta(n_1, n_2)$ evaluates the common sub-trees rooted at n_1 and n_2 and is computed recursively as follows:

(1) if the productions (i.e. the nodes with their direct children)at n_1 and n_2 are different, $\Delta(n_1, n_2) = 0$;

(2) else if both $n1$ and $n2$ are pre-terminals (POS tags), $\Delta(n_1, n_2)=1\times\lambda$;

(3) else calculate $\Delta(n_1, n_2)$ recursively as:

$$\Delta(n_1, n_2) = \lambda \prod_{k=1}^{\#ch(n_1)} (1 + \Delta(ch(n_1, k), ch(n_2, k))), \tag{6}$$

where *#ch(n₁)* is the number of children of node n_1; *ch(n, k)* is the k^{th} child of node n; and $\lambda(0<\lambda<1)$ is the decay factor used to make the kernel value less variable with respect to different sized sub-trees. The parse tree kernel counts the number of common sub-trees as the syntactic similarity measure between two relation instances. The time complexity for computing this kernel is $O(|N_1| \cdot |N_2|)$.

5 Experiments

5.1 Experimental Setting

We evaluated our method with three publicly available corpora that contain PPI annotations: LLL [13], IEPA [4] and HPRD50 [6] (the distribution of corpora are shown as the Fig.5). All the corpora are parsed using Stanford parser (http://nlp.stanford.edu/software/lex-parser.shtml) to generate the output of parse tree and part-of-speech tagging. In our implementation, we use Moschitti's tree kernel toolkit [1] to develop the convolution kernel of an IPT. To derive credible evaluation results, we utilize the 10-fold cross validation method [1] on all of the corpora. This guarantees the maximal use of the available data and allows comparison to the earlier relevant work. The evaluation metrics are the precision rate, recall rate, and F1-measure [1]. The F1 value is used to determine relative effectiveness of the compared methods. We exploit the macro-averaged score to indicate the overall performance across three different corpora for each evaluation metric.

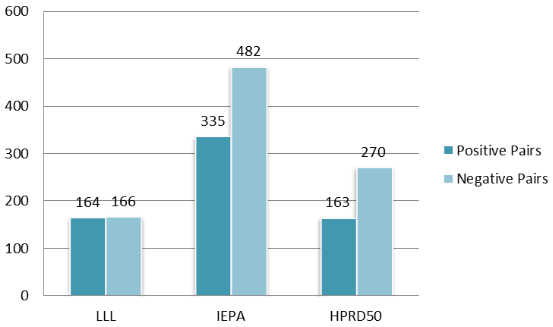


Fig. 5. Distribution of 3 corpora used for performance evaluation of PPI extraction

5.2 Results and Discussion

The proposed interaction pattern tree kernel uses the PPI patterns to enhance the SPT. In the following, we compare it with several feature-based and kernel-based PPI extraction methods reported by [17] to demonstrate the effectiveness. As shown in Table 1, the proposed method significantly outperforms SPT and AkanePPI. Furthermore, the syntax tree-based kernel methods (ST, SST, PT, and SpT) only examine the syntactic structures of text and cannot sense the semantics of protein interactions. By contrast, our method analyzes the semantics and content (i.e., PPI patterns) of text to identify protein-protein interactions. Hence, our performance is superior to that of them. It is noteworthy that syntax tree-based kernel methods oftentimes are just on par with the co-occurrence approach in terms of F1-measure. On the very small LLL, their results practically coincide with co-occurrence. The rich-feature-based and Cosine also outperform SPT, AkanePPI and syntax tree-based kernel methods as it incorporates dependency features to distinguish protein-protein interactions. Although Cosine can accomplish higher performance by further considering term weighting, it is difficult to represent word relations. By contrast, our method can extract word semantics, and generate PPI patterns that can capture long distance relations among them. Consequently, we can achieve a better outcome than other methods.

To summarize, the proposed interaction pattern tree kernel approach successfully integrates the syntactic and semantic information in text to identify protein-protein interactions. Hence, it achieves the best performance among the compared methods, as shown in Table 1.

Table 1. The interaction extraction performance of the compared methods

System	***LLL***	***IEPA***	***HPRD50***	***Macro-average***
	Precision, Recall, F1-measure (%)			
SPT	56.4 / 96.1 / 69.6	55.5 / 28.8 / 37.1	46.2 / 13.4 / 20.8	52.7 / 46.1 / 42.5
AkanePPI [19]	**76.7** / 40.2 / 52.8	**66.2** / 51.3 / 57.8	52.0 / 55.8 / 53.8	**65.0** / 49.1 / 54.8
co-occ. [1]	55.9 / **100.** / 70.3	40.8 / **100.** / 57.6	38.9 / **100.** / 55.4	45.2 / **100.** / 61.1
PT [13]	56.2 / 97.3 / 69.3	63.1 / 66.3 / 63.8	54.9 / 56.7 / 52.4	58.1 / 73.4 / 61.8
SST [3]	55.9 / **100.** / 70.3	54.8 / 76.9 / 63.4	48.1 / 63.8 / 52.2	52.9 / 80.2 / 62.0
ST [17]	55.9 / **100.** / 70.3	59.4 / 75.6 / 65.9	49.7 / 67.8 / 54.5	55.0 / 81.1 / 63.6
SpT [9]	55.9 / **100.** / 70.3	54.5 / 81.8 / 64.7	49.3 / 71.7 / 56.4	53.2 / 84.5 / 63.8
rich-feature-based [16]	72.0 / 73.0 / 73.0	64.0 / 70.0 / 67.0	**60.0** / 51.0 / 55.0	65.3 / 64.7 / 65.0
Cosine [6]	70.2 / 81.7 / **73.8**	61.3 / 68.4 / 64.1	59.0 / 67.2 / 61.2	63.5 / 72.4 / 66.4
Our method	59.9 / 94.4 / 71.6	52.2 / 88.1 / **65.2**	59.3 / 83.0 / **67.3**	57.1 / 88.5 / **68.0**

6 Concluding Remarks

Automated extraction of protein-protein interactions is an important and widely studied task in biomedical text mining. To this end, we proposed an interaction pattern generation approach for acquiring PPI patterns. We also developed a method that combines the shortest path-enclosed tree structure with the generated PPI patterns to analyze the syntactic, semantic, and content information in text. It then exploits the derived information to identify protein-protein interactions in biomedical literatures. Our experiment results demonstrate that the proposed method is effective and also outperforms well-known PPI extraction methods.

In the future, we will investigate the syntactic dependency tree in text to incorporate further syntactic and semantic information into the interactive pattern tree structures. We will also utilize information extraction algorithms to extract interaction tuples from positive instances and construct an interaction network of proteins.

Acknowledgements. This research was supported by the National Science Council of Taiwan under grant NSC102-3113-P-001-006, MOST103-2319-B-010-002 and MOST103-3111-Y-001-027.

References

1. Airola, A., Pyysalo, S., Björne, J., Pahikkala, T., Ginter, F.: All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. BMC Bioinformatics 9, S2 (2008)
2. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval: The Concepts and Technology Behind Search. Addison Wesley (2011)
3. Collins, M., Duffy, N.: Convolution kernels for natural language. In: Proceedings of Annual Conference on Neural Information Processing Systems, pp. 625–632 (2001)
4. Cooper, C., Frieze, A.M.: The cover time of random regular graphs. SIAM Journal on Discrete Mathematics 18, 728–740 (2005)
5. Ding, J., Berleant, D., Nettleton, D., Wurtele, E.: MEDLINE: abstracts, sentences, or phrases? In: Proceedings of Pacific Symposium on Biocomputing, pp. 326–337 (2002)
6. Erkan, G., Özgür, A., Radev, D.R.: Semi-supervised classification for extracting protein interaction sentences using dependency parsing. In: Proceedings of the 2007 Joint Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 228–237 (2007)
7. Fundel, K., Küffner, R., Zimmer, R.: RelEx – Relation extraction using dependency parse trees. Bioinformatics 23(3), 365–371 (2007)
8. Kim, J.D., Ohta, T., Pyysalo, S., Kano, Y., Tsujii, J.: Overview of BioNLP'09 shared task on event extraction. In: Proceeding of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task, pp. 1–9 (2009)
9. Kuboyama, T., Hirata, K., Kashima, H., Aoki-Kinoshita, K.F., Yasuda, H.: A spectrum tree kernel. Information and Media Technologies 2, 292–299 (2007)
10. Lovász, L.: Random walks on graphs: a survey, vol. 2, pp. 1–46. Janos Bolyai Mathematical Society, Budapest (1993)
11. Manning, C.D., Schütze, H.: Foundations of statistical natural language processing, 1st edn. MIT Press, Cambridge (1999)
12. Moschitti, A.: A study on convolution kernels for shallow semantic parsing. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, pp. 21–26 (2004)
13. Moschitti, A.: Efficient convolution kernels for dependency and constituent syntactic trees. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 318–329. Springer, Heidelberg (2006)
14. Nédellec, C.: Learning language in logic-genic interaction extraction challenge. In: Proceedings of the 4th Learning Language in Logic Workshop, pp. 31–37 (2005)
15. Porter, M.F.: An algorithm for suffix stripping. In: Jones, K.S., Willet, P. (eds.) Readings in Information Retrieval. Morgan Kaufmann, San Francisco (1997)
16. Van Landeghem, S., Saeys, Y., De Baets, B., Van de Peer, Y.: Extracting protein-protein interactions from text using rich feature vectors and feature selection. In: Proceedings of 3rd International Symposium on Semantic Mining in Biomedicine, pp. 77–84 (2008)
17. Tikk, D., Thomas, P., Palaga, P., Hakenberg, J., Leser, U.: A Comprehensive Benchmark of Kernel Methods to Extract Protein–Protein Interactions from Literature. PLoS Computational Biology 6(7), 1–19 (2010)

18. Vishwanathan, S.V.N., Smola, A.J.: Fast kernels for string and tree matching. In: Proceedings of Neural Information Processing Systems (NIPS 2002), pp. 569–576 (2002)
19. Satre, R., Sagae, K., Tsujii, J.: Syntactic features for protein-protein interaction extraction. In: Proceedings of the 2nd International Symposium on Languages in Biology and Medicine, pp. 6.1-6.14 (2007)