PPIExtractor: A Protein-Protein Interaction Extractor for Biomedical Literature

Zhihao Yang, Zhehuan Zhao, Yanpeng Li, Yuncui Hu, Hongfei Lin
College of Computer Science and Technology, Dalian University of Technology, Dalian, China 116023
yangzh@dlut.edu.cn

Abstract—Knowledge about protein-protein interactions (PPIs) unveils the molecular mechanisms of biological processes. In this paper, we present a PPI extraction system, termed PPIExtractor, which automatically extracts PPIs from biomedical text and visualizes them. Given a Medline record dataset, PPIExtractor first applies Feature Coupling Generalization (FCG) to tag protein names, next uses the extended semantic similarity-based method to normalize them, then combines feature-based, convolution tree and graph kernels to extract PPIs, and finally visualizes the PPI network. Experimental evaluations show that PPIExtractor can achieve state-of-the-art performance on a DIP subset with respect to comparable evaluations. PPIExtractor is freely available for academic purposes at: http://202.118.75.18:8080/PPIExtractor/.

Keywords-Information extraction; Protein–protein interaction; Feature Coupling Generalization; Multiple kernels learning

I. INTRODUCTION

Protein-protein interactions (PPIs) play a key role in various aspects of the structural and functional organization of the cell. Knowledge about them unveils the molecular mechanisms of biological processes. A number of databases such as MINT, BIND, and DIP have been created to store protein interaction information in structured and standard formats. However, the amount of biomedical literature regarding protein interactions is increasing rapidly and it is difficult for interaction database curators to detect and curate PPI information manually. Thus, most of the PPI information remains hidden in the unstructured text of the published papers. Correspondingly, many systems developed for extracting PPIs from the biomedical literature have been made publicly available.

The SUISEKI system uses regular expressions, with probabilities that reflect the experimental accuracy of each pattern to extract interactions into predefined frame structures [1]. BioBiblioMetrics uses co-occurrence of gene naming terms in Medline abstracts to generate semantic links between genes [2]. BioRAT uses manually engineered templates that combine lexical and semantic information to identify protein interactions [3]. PIE utilizes natural language processing techniques and machine learning method to extract PPIs [4]. OpenDMAP uses template matching backed by ontological resources to represent slots and potential slot fillers, etc. [5]. PPLook, for a given query protein name, can search for the interacting proteins using a keyword dictionary pattern-matching algorithm [6].

Table I shows the feature comparison of the PPI extraction systems publicly available. It should be noted that, in our opinion, a complete PPI extraction system should integrate Named Entity Recognition (NER) model, syntactic parser and other components it needs, and then it can take plain texts (e.g. Medline abstract) as input. In this way, ordinary biologists without a computer science backgound can also use it in their work. Some PPI extraction systems (e.g. AkaneRE [7]) can only process the dataset that has been pre-processed by dependency parser and named entity recognizer and are not listed in Table I). PPIExtractor is a PPI extraction system presented in this paper. Given a Medline record dataset, it first applies Feature Coupling Generalization (FCG) [8] to tag protein names, next uses the extended semantic similarity-based method to normalize them, then combines the feature-based, tree and graph kernels to extract PPIs, and visualizes the PPI network.

TABLE I. Feature comparison of PPI extraction systems publicly available. "Nor." denotes Normalization; "PPIIV" denotes PPI integration and visualization

	NER	Nor.	PPI Extraction	Output	PPIIV
Suiseki	protein name dictionary	N	template matching	PPI pairs	Y
BioBiblio-Metrics	protein name dictionary	N	co-occurrence	PPI pairs	Y
BioRAT	protein name dictionary	N	template matching	PPI pairs	N
PIE	protein name dictionary	N	convolution tree kernel	PPI sentences	N
OpenDMAP	ABNER system	N	template matching	PPI pairs	N
PPLook	GENIA tagger	N	template matching	PPI pairs	Y
PPIExtractor	CFG+CRFs	Y	multiple kernel Combination	PPI pairs	Y

In terms of NER, most of systems use protein name dictionary. Dictionary look-up is the most straightforward technique for NER, but its performance for entity recognition is inferior to those of machine learning methods [9]. OpenDMAP uses ABNER [10], a Conditional Random Fields (CRFs)-based open source tool for NER while PPIExtractor uses the NER method of FCG combined with CRFs model which has a better performance over ABNER [8] on BioCreative II Gene Mention test set [9]. PPLook uses the GENIA tagger [11] to tag protein name which, on the JNLPBA test set, has almost the same level performance with ABNER (72.79% to 72.6% in F-score on protein name tag, obtained on the website of GENIA Tagger and ABNER respectively (http://www.nactem.ac.uk/tsujii/GENIA/tagger/ and http://pages.cs. wisc. edu /~bsettles/abner/)).

In terms of Normalization, most of systems do not have this processing stage. However, it is a key step in the

integration of different knowledge sources, viz. unstructured textual and structured database information. Progress in normalizing biological entities recognized in text to specific database identifiers [12, 13] has made the output of text processing systems much more valuable. However, gene name normalization poses significant semantic difficulty: despite the existence of various standards bodies, there is great variability in how genes and gene products are mentioned in the literature; Ambiguity is inherent in the synonym dictionary, as well as being generated during the Normalization step that transforms string mentions. Even the same name may refer to different meanings in different articles. PPIExtractor presents a disambiguation method based on extended semantic similarity which enriches the gene descriptions in databases with the information extracted from Gene Ontology and Medline abstracts. The method was tested on BioCreative II Gene Normalization task dataset [13] and achieved a performance comparable to the current state-of-the-art in this task.

In terms of PPI extraction, most of systems are based on template matching, which usually yields high precision but low individual recall. OpenDMAP, using a set of manually created patterns, performed best in the BioCreative II PPI task [14], but was found to perform worst (compared to 5 other systems, including co-occurrence as baseline) on each of five corpora [15]. The PIE system utilizes the convolution tree kernel to extract PPI while PPIExtractor combines the feature-based, convolution tree and graph kernels to extract PPIs with appropriate weights and achieves much better performance than that of the convolution tree kernel alone as shown in Table II.

In addition, integration and visualization of PPI network help biologists to extract information from the data. A visual representation of the network has several advantages over viewing interactions as binary data. Visual analysis of PPIs can increase the confidence levels for individual interactions and allows the assignment of potential functions to uncharacterized proteins. It is also important for uncovering interactions that link diverse cellular processes [16]. Unfortunately, Suiseki, BioBiblioMetrics and PPLook can only implement PPI integration and visualization with protein names while PPIExtractor can also provide integration and visualization with EntrezGene IDs which is a key step in the integration of different knowledge sources.

As the best of our knowledge, PPIExtractor is the first PPI extraction system publicly available which integrates NER, Normalization, PPI extraction, integration and visualization. Moreover, the technique used in each stage can achieve comparable performance with other systems.

II. MEHTODS

PPIExtractor contains four modules: (i) NER module which aims to identify the protein names in the biomedical literature; (ii) Normalization module which determines the unique identifier of proteins identified in NER module; (iii) PPI extraction module which extracts the PPI information in the biomedical literature and (iv) PPI visualization module which displays the extracted PPI information in the form of a graph. Figure 1 shows the architecture of PPIExtractor.

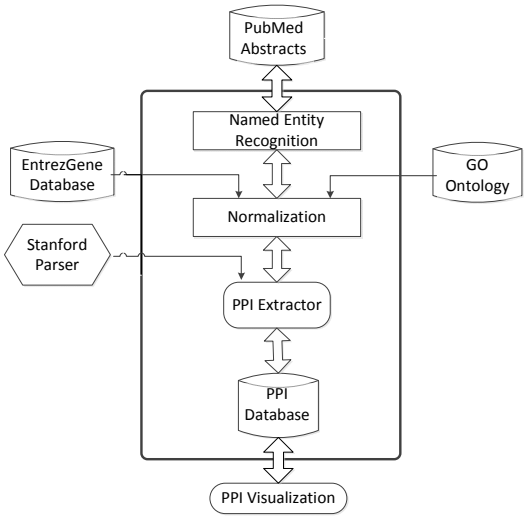

Figure 1. Architecture of PPIExtractor

A. NER

PPIExtractor uses a NER module based on the recently proposed semi-supervised learning strategy - FCG [8]. The general idea of FCG is to learn a novel feature representation from the co-occurrences of two special types of raw features: class-distinguishing features (CDFs) and example-distinguishing features (EDFs). CDFs and EDFs refer to strong indictors for examples and for classes respectively. In the case of NER, for example, when determining whether a term is a gene name, the pattern that the term ends with '*gene*' is an example of CDFs while the word or n-gram features which have the strong capability to differentiate the current example from others are examples of EDFs. Intuitively, their co-occurrences in huge unlabeled data will capture indicative information that could not be obtained from labeled training data due to data sparseness.

We examined its performance in a named entity classification task [8]. The results show that new features generated by FCG outperform lexical features by 5.97 percentage units in F-score. Also in this framework each extension yields significant improvements and the sparse lexical features can be transformed into both a lower dimensional and more informative representation. A forward maximum match method based on the refined dictionary produces an F-score of 86.2% on BioCreative II test set. Then the dictionary is combined with a CRFs-based gene mention tagger, achieving an F-score of 89.05%.

B. Normalization

The task of gene (protein) name normalization is to determine the unique identifier of genes (proteins) mentioned in biomedical literature, so as to create the linkage of these entities to biological databases. Gene name normalization poses significant semantic difficulty, as it requires detection of the actual gene intend, along with reporting of the gene in a standardized form. It is a challenging task even for the human experts. The difficulties lie in: (1) despite the existence of various standards bodies, there is great variability in how genes and gene products are mentioned in

the literature. Gene names are often described, rather than referred to by names or symbols by medical researchers, as in "protein homologous to the product of the C. elegans gene lin-7"; (2) ambiguity is inherent in the synonym dictionary, as well as being generated during the normalization step that transforms string mentions.

PPIExtractor adopts a disambiguation method based on the extended semantic similarity which enriches the gene descriptions in databases with the information extracted from Gene Ontology (GO) and Medline abstracts. Based on the idea that the information associated with a gene can be a good indicator of that gene, the method predicts the identifier with consideration of gene mention's context and gene identifier's extended semantic information. Besides short description from EntrezGene, the extended semantic information used for disambiguation is composed of manually curated GO annotation and knowledge derived from all Medline abstracts known to be relevant to the gene.

The gene synonym lexicon we used is provided by BioCreative II Gene Normalization task [13], which is composed of 32975 entries, and each entry is made up of multiple synonyms indexed by the same gene identifier (EntrezGene ID). Our Normalization method achieves an F-measure of 83 % (a precision of 82.5% and recall of 83.5%) on the dataset of BioCreative II task, which is comparable to the current state-of-the-art.

C. PPI extractor

Among the popular machine learning approaches, kernel-based methods including tree kernels [17], shortest path kernels [18], and graph kernels [19] have been proposed for PPIs extraction. These methods retain the original representation of objects and use the objects in algorithms only via computing a kernel function between objects.

In recent years, researchers have proposed the use of multiple kernels to retrieve the widest range of important information in a given sentence. Kim *et al.* suggested four kernels: predicate, walk, dependency and hybrid kernels to adequately encapsulate information required for a relation prediction based on the sentential structures involved in two entities [20]. Miwa *et al.* proposed a method to combine bag-of-words (BOW), subset tree and graph kernels based on several syntactic parsers, in order to retrieve the widest possible range of important information from a given sentence [21]. However, these methods assign the same weight to each individual kernel and their combined kernels fail to achieve the best performance. The reason is that, with the same weight, the introduction of the kernel with poor performance will deteriorate the overall performance of the combined kernel as in the cases of [20] and [21].

We proposed a weighted multiple kernel learning-based approach to extracting PPIs from biomedical literature [22]. The approach combines several appropriately weighed kernels, namely feature-based, convolution tree, and graph kernels, whereby the kernel with better performance is assigned higher weight.

1) Feature-based kernel

Word feature (words between and surrounding two protein names), protein name distance feature (the number of words between two protein names) and keyword feature (the existence of an interaction keyword (e.g. "bind," "interact," "inhibit," etc) between or surrounding two protein names) are used in our feature-based kernel.

2) Convolution tree kernel

As a specialized convolution kernel, convolution tree kernel K_C (T_1, T_2) counts the number of common sub-trees (sub-structures) as the syntactic structure similarity between two parse trees T_1 and T_2 [23]:

$$K_C(T_1, T_2) = \sum_{n_1 \in N_1, n_2 \in N_2} \Delta(n_1, n_2) \quad (1)$$

where N_j is the set of nodes in tree T_j, and $\Delta(n_1, n_2)$ evaluates the common sub-trees rooted at n_1 and n_2.

a) Parse tree kernel: By default, we adopt the shortest path tree (SPT) as our tree span. When the number of leaf nodes in the SPT is smaller than four, the SPT is expanded to a higher level, i.e. the parent node of the root node of the original SPT is used as the new root node.

b) Dependency path tree kernel: For dependency-based parse representations, a dependency path is encoded as a flat tree depicted as follows: (DEPENDENCY (NSUBJ (interacts ENTITY1)) (PREP (interacts with)) (POBJ (with ENTITY2))) corresponding to the sentence "ENTITY1 interacts with ENTITY2." Similar to the SPT parse tree, in some cases, dependency path tree also needs to be extented.

3) Graph kernel

A graph kernel calculates the similarity between two input graphs by comparing the relations between common vertices (nodes). The graph kernel used in our method is the all-paths graph kernel proposed by Airola *et al.* [19].

4) Combination of kernels

Each kernel utilizes a portion of the structures to calculate useful similarity. Combining the similarities can reduce the risk of missing important features and thus produces a new useful similarity measure. To realize the combination of different types of kernels based on different parse structures, we sum up the normalized output of several kernels K_m as:

$$K(x, x') = \sum_{m=1}^{M} \sigma_m K_m(x, x') \quad (2)$$

$$\sum_{m=1}^{M} \sigma_m = 1, \ \sigma_m \geq 0, \ \forall m \quad (3)$$

where M represents the number of kernel types, and σ_m is the experimentally determined weight of each K_m, adjusted until the overall best results are achieved. More complete details are presented in [22].

III. EXPERIMENTAL RESULTS

A. Experimental setting

We used the SVMLight package (http://svmlight.joachims.org/) developed by Joachims for our feature-based kernel. The polynomial kernel is chosen with parameter d = 4. Tree Kernel Toolkit developed by Moschitti with default

parameters is chosen for our convolution tree kernel (http://dit.unitn.it/~moschitt/Tree-Kernel.htm). For the graph kernel all-paths graph kernel proposed by Airola et al. (http://mars.cs.utu.fi/PPICorpora /GraphKernel.html) is used.

The binary classification performance of our PPI extraction module is tested on AImed corpus [18] since it is sufficiently large for training and reliably testing of machine learning methods. Further, we evaluated our method with 10-fold document-level cross-validation, which guarantees the maximal use of the available data and allows comparison to the earlier relevant work.

The majority of PPI extraction system evaluations use the balanced F-score measure for quantifying the performance of the systems and, therefore, we also use it in our evaluation.

B. The Binary classification performance of PPI extraction module

The performances of different kernels used in PPIExtractor are shown in Table II, showing that graph kernel has superior performance compared with other two kernels. The reason is that the graph kernel can treat the parser's output and word features at the same time. The performance of the feature-based kernel ranks second since it uses protein names distance and keyword feature in addition to words features.

The experimental results show that, when two or more individual kernels are combined, better performances are achieved. For example, when the graph kernel is combined with the feature-based kernel, the performance is improved by 1.82 percentage units in F-score. When further combined with the convolution tree kernel, the performance is improved by 4.88 percentage units in F-score. Since different kernels calculate the similarity between two sentences from different aspects, the combination of kernels covers more knowledge and is effective for PPI extraction.

TABLE II. Effectiveness of different kernels on AImed. The weights of each individual kernel in combined kernels are in the parentheses after the kernel name.

	P	R	F
Feature-based kernel	46.32	61.1	52.69
Tree kernel	43.71	64.65	52.24
Graph kernel	52.66	64.56	57.20
Tree kernel(0.5)+ Feature-based kernel(0.5)	50.44	68.49	58.05
Graph kernel(0.7) +Feature-based kernel(0.3)	51.33	69.58	59.02
Graph kernel(0.7)+ Tree kernel(0.3)	53.43	68.57	59.66
Feature-based kernel(0.2) + Tree kernel(0.2)+ Graph kernel(0.6)	57.4	70.75	63.9

In addition, we found that only when the kernels with better performances are assigned higher weights can the combined kernel produce the best result. In our experiments, the weights for the graph, feature-based, and convolution tree kernels are set to 0.6, 0.2, and 0.2 respectively in the descending order of their performance. These weights have been optimized by cross-validation.

The comparison with relevant results reported in related research is summarized in Table III. The best performing system combines multiple layers of syntactic information by using a combination of multiple kernels based on several different parsers and achieves an F-score of 63.5% [24]. PPIExtractor uses only the Stanford parser output to produce the parse tree and dependency structure (path and graph) information, yielding a comparable performance. This is due to the following two key reasons: 1) With the feature-based kernel, besides the commonly used word feature, protein names distance and keyword feature are introduced to improve the performance. 2) Our combined kernel can reduce the risk of missing important features, yielding new useful similarity measures. More specifically, the weighted linear combination of individual kernel used instead of assigning the same weight to each individual kernel is experimentally proven to contribute to the performance improvement.

TABLE III. Comparison on AImed. Precision, recall, F-score results for methods evaluated on AImed with the correct cross-validation methodology.

Method	P	R	F
Our method	57.4	70.75	63.9
Miwa et al. [24]	60.4	69.30	63.5
Miwa et al. [21]	58.7	66.1	61.9
Miyao et al. [25]	54.9	65.5	59.5
Airola et al. [19]	52.9	61.8	56.4

C. The overall performance of PPIExtractor on DIP

To explore the overall performance (which involves the performances of NER, Normalization and PPI extraction models) of PPIExtractor, experiments were performed on the DIP database. Blaschke and Valencia recommend using DIP as a way of evaluating biological IE systems, because it represents a realistic problem of practical interest to biological researchers [26]. IE researchers can use their systems to extract PPIs, and then compare these with the records in DIP.

The test set used in our experiments is the same DIP dataset used for the BioPPISVMExtractor [27], BioPPIExtractor [28], BioRAT [6] and IntEx [29] evaluation so that the results are comparable. For evaluation, 394 interactions were identified from the DIP database such that both proteins participating in the interaction had SwissProt entries. These interactions correspond to 229 abstracts from the PubMed. Since we did not obtain the corresponding full papers, PPIExtractor was tested only on the abstracts.

The training set used for our PPI extraction model is the combination of five publicly available corpora (AImed [18], BioInfer, HPRD50, IEPA and LLL) which includes 4174 positive and 12483 negative instances.

We evaluated the results in the similar way as the other four systems. The candidate interactions extracted by PPIExtractor were manually examined by a domain expert for precision and recall. Our domain expert manually

compared the candidate interactions to the source DIP records to measure the recall. For each record in DIP, our domain expert searched through the output of PPIExtractor corresponding to the same Medline abstract, and checked to see if the interaction mentioned in DIP has been identified. Precision is harder to measure than recall, because we need an estimate of the number of false positives. If a record produced by PPIExtractor is not found in DIP, it could be that a) it is a false-positive example, reducing the precision of PPIExtractor; or b) the record is missing from DIP. The latter case consists of interactions that are mentioned in Medline abstract, but have yet to be added to DIP. Our domain expert manually checked each interactions extracted by the system. If the two proteins in an interaction have a biologically relevant relationship between them, the interaction is used as a true positive, whether or not the information is in DIP.

The PPIExtractor evaluation results as compared with other four systems are shown in Table IV, in which the recalls, different from those in Table II and III (which are used to evaluate the classification performance of different kernels), are calculated as the ratio of extracted DIP interactions to 394 (the sum of all interactions selected from the DIP database). The performances of PPIExtractor before and after the Normalization are listed respectively. Since the corresponding EntrezGene Ids of some protein names may not be found in the Normalization stage, the recall of PPIExtractor after the Normalization is lower than that of before the Normalization. On the other side, the precision after the Normalization is higher than that of before the Normalization since the extracted PPI between two proteins with identified EntrezGene Ids are more accurate.

TABLE IV. PPIExtractor performance comparison with other four systems. The precision, recall, and F-score values are calculated with the same method for all the systems and achieved with the optimal threshold.

	R	P	F
PPIExtractor (before Normalization)	56.09	71.22	62.75
PPIExtractor (after Normalization)	39.59	79.23	52.80
BioPPISVMExtractor	72.84	50.86	59.90
BioPPIExtractor	41.62	55.41	47.53
IntEx	26.94	65.66	38.20
BioRAT	20.31	55.07	29.68

It should be noted that the performance of BioPPISVMExtractor is better than that introduced in [27] since, for comparison purpose, we used the same training set as PPIExtractor, i.e. the combination of five corpora instead of the IEPA corpus alone to re-train the PPI extraction model of BioPPISVMExtractor.

In terms of F-score, the performance of PPIExtractor (before the Normalization, 62.75%) is better than those of BioPPISVMExtractor (59.90%), BioPPIExtractor (46.33%), BioRAT (29.68%) and IntEx (38.20%). Compared with another machine learning-based system BioPPISVMExtractor, PPIExtractor achieves better performance (nearly 3 percentage points in F-score higher than that of BioPPISVMExtractor). This is due to the following two reasons: 1) The NER model in PPIExtractor combines the refined dictionary (achieved with FCG) with a CRFs model and outperforms the performance of the CRFs-based model used in BioPPISVMExtractor. 2) The PPI extraction model in PPIExtractor combines several appropriately weighed kernels while BioPPISVMExtractor only uses the feature-based kernel with Link path and Link Grammar extraction result features extracted by link grammar analysis. The combined kernel can reduce the risk of missing important features, and the weighted linear combination of individual kernel contributes to the performance improvement.

As to other three systems, their performances are far inferior to that of PPIExtractor. BioRAT uses templates and achieves a recall of 20.31% and a precision of 55.07%. IntEx using link grammar to identify interactions between proteins achieves a recall of 26.94% and a precision of 65.66%. Nevertheless, the performances of these manual pattern engineering and grammar engineering approaches are usually inferior to those of machine learning approaches [14]. Especially, both BioRAT and IntEx adopt the dictionary-based NER method, which leads to poor recall performance (20.31% and 26.94% respectively) since the dictionary-based method depends badly on the size and quality of the protein name dictionary. For example, the recall errors generated in NER account for about 60% and 45% respectively in BioRAT and IntEx. BioPPIExtractor, like IntEx, also uses a Link Grammar parser to identify the syntactic roles in sentences and then extracts interactions from these syntactic roles. However, by introducing a CRFs-based NER method, BioPPIExtractor achieves a much better recall of 41.62% and a still good precision of 55.41%.

D. Error Analysis

We also made a detailed analysis of all types of recall errors of PPIExtractor (after Normalization).

DIP contains protein interactions from both abstracts and full text. Since PPIExtractor was tested on the abstracts, it missed out on some interactions that were only present in the full text. This accounts for 23.53% of total recall errors. If those interactions are excluded, PPIExtractor can have a recall of 53.85% (39.59% currently).

In addition, recall errors occur in all the PPI extraction processing stages: NER, Normalization and PPI extraction. Among others, most errors are generated in PPI extraction stage (33.19%). The reason is that due to the complexity of the protein interaction expression as well as the limited size of training set, many interactions are missed out.

The recall errors generated in NER account for 13.45% of total recall errors. The number is about 60% and 45% in BioRAT and IntEx respectively since they adopt the dictionary-based NER method whose performance depends badly on the size and quality of the protein name dictionary. Compared with another machine learning-based system BioPPISVMExtractor, the recall errors generated in NER of PPIExtractor is fewer (13.45% to 15.32%) for its better NER model.

In addition, 29.83% recall errors are generated in Normalization stage since the corresponding EntrezGene Ids of some protein names are not found in this stage. The reason is that, despite the existence of various standards bodies, there is great variability in how genes and gene products are mentioned in the literature.

Most precision errors occur in the PPI extraction stage. The reason is that, confined to the complexity of the protein interaction expression as well as the quantity and quality of the training set, many false positives are generated.

IV. CONCLUSION

In this paper, we present a PPI extraction system, termed PPIExtractor, which automatically extracts PPI information from biomedical text and visualizes them. As the best of our knowledge, PPIExtractor is the first PPI extraction system publicly available which integrates NER, Normalization, PPI extraction and visualization and ordinary biologists without a computer science backgound can also use it in their work. Especially, the technique used in each stage can achieve comparable performance with other systems.

PPIExtractor can be used to extract PPIs from a large collection of biomedical text files and is a useful tool for functional bioinformatics.

ACKNOWLEDGMENT

This work is supported by grant from the Natural Science Foundation of China (No. 60973068 and 61070098), the Fundamental Research Funds for the Central Universities (No. DUT10JS09) and Liaoning Province Doctor Startup Fund (No. 20091015).

REFERENCES

[1] C. Blaschke and A. Valencia, "The frame-based module of the SUISEKI information extraction system," IEEE Intell. Syst., vol.17, pp.14-20, 2002.

[2] B.J. Stapley and G. Benoit, "Biobibliometrics: Information retrieval and visualization from co-occurrence of gene names in Medline abstracts," Proc. the Pacific Symposium on Biocomputing, Hawaii, pp.526-537, 2000.

[3] D.P. Corney, B.F. Buxton, W.B. Langdon, and D.T. Jones, "BioRAT: extracting biological information from full-length papers," Bioinformatics, vol. 17, pp. 3206-3213, 2004.

[4] S. Kim, et al., "PIE: an online prediction system for protein--protein interactions from text," Nucleic Acids Res, vol. 36, pp.W411-415, 2008.

[5] L. Hunter, et al., "OpenDMAP: An Open Source Ontology-Driven Concept Analysis Engine with Applications to Capturing Knowledge Regarding Protein Transport Protein Interactions and Cell-Type-Specific Gene Expression," BMC Bioinformatics, 2008, doi:10.1186/1471-2105-9-78

[6] S.W. Zhang, Y.J. Li, L.Xia, and Q. Pan, "PPLook: an automated data mining tool for protein-protein interaction," BMC Bioinformatics, 2010, doi:10.1186/1471-2105-11-326.

[7] R. Saetre, et al., "Extracting protein interactions from text with the unified AkaneRE event extraction system," IEEE/ACM Trans. Comput. Biol. Bioinfor., vol.7, pp. 442-453, 2010.

[8] Y.P. Li, H.F. Lin and, Z.H. Yang, "Incorporating rich background knowledge for gene named entity classification and recognition," BMC Bioinformatics, 2009, doi:10.1186/1471-2105-10-223.

[9] J.Wilbur., L. Smith, and L. Tanabe, "BioCreative 2. gene mention task," Proc. the Second BioCreative Challenge Evaluation Workshop, Madrid. pp. 7-16, 2007.

[10] B. Settles, "Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets," Proc. the COLING 2004 International Joint Workshop on Natural Language Processing in Biomedicine and its Applications, Geneva, pp.107-110, 2004.

[11] Y. Tsuruoka and T. Tsujii, "Bidirectional Inference with the Easiest-First Strategy for Tagging Sequence Data," Proc. the Human Language Technology and Empirical Methods in Natural Language Processing. Vancouver, Canada, pp. 467-474, 2005.

[12] M. Ashburner, et al., "Gene ontology: tool for the unification of biology," The Gene Ontology Consortium. Nat. Genet., vol.25, pp.25-29, 2000.

[13] A. Morgan, et al., "Overview of BioCreative II Gene Normalization," Genome Biol., 2008, doi:10.1186/gb-2008-9-s2-s3.

[14] M. Krallinger, F. Leitner, C. Rodriguez-Penagos, and A.Valencia, "Overview of the protein-protein interaction annotation extraction task of BioCreative II," Genome Biol., 2008, doi: 10.1186/gb-2008-9-s2-s4.

[15] R. Kabiljo, A. B. Clegg, and A. J. Shepherd, "A realistic assessment of methods for extracting gene/protein interactions from free text," BMC Bioinformatics, 2009, doi:10.1186/1471-2105-10-233.

[16] M.L. Mayer, and P. Hieter, "Protein networks-built by association," Nat. Biotechnol., vol.18, pp.1242-1243, 2000.

[17] A. Moschitti, "Making tree kernels practical for natural language processing," Proc. the 11th Conference of the European Chapter of the Association for Computational Linguistics, Trento, Italy, 2006.

[18] R.C. Bunescu and R.J. Mooney, "A shortest path dependency kernel for relation extraction," Proc. the Human Language Technology and Empirical Methods in Natural Language Processing. Morristown, USA, pp. 724-31, 2005.

[19] A. Airola, et al., "All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning," BMC Bioinformatics, 2008, doi:10.1186/1471-2105-9-S11-S2.

[20] S. Kim, J. Yoon, and J. Yang, "Kernel approaches for genic interaction extraction," Bioinformatics, vol.24, pp. 118-26, 2008.

[21] M. Miwa, R. Saetre, Y. Miyao, and J. Tsujii, "Protein-protein interaction extraction by leveraging multiple kernels and parsers," Int. J. Med. Inform., vol.78, pp. e39-46, 2009.

[22] Z.H. Yang, et al., "Multiple Kernel Learning in Protein-Protein Interaction Extraction from Biomedical Literature," Artif. Intell. Med., vol.51, pp.163-73, 2011.

[23] M. Collins, and N. Duffy, "Convolution Kernels for Natural Language," Proc. the 14th Conference on Neural Information Processing Systems, Cambridge, pp. 625-632, 2001.

[24] M. Miwa, R. Sætre, Y. Miyao, T. Ohta, and J.Tsujii, "Combining Multiple Layers of Syntactic Information for Protein-Protein Interaction Extraction," Proc. the Third International Symposium on Semantic Mining in Biomedicine. Turku, Finland, pp. 101-108, 2008.

[25] Y. Miyao, R. Sætre, K. Sagae, T. Matsuzaki, and J. Tsujii, "Evaluating contributions of natural language parsers to protein-protein interaction extraction," Bioinformatics, vol. 25, pp.394-400, 2009.

[26] C. Blaschke and A. Valencia, "Can bibliographic pointers for known biological data be found automatically? Protein interactions as a case study," COMP. FUNCT. GENOM., vol.2, pp.196–206, 2001.

[27] Z.H. Yang, H.F. Lin, and Y.P. Li, "BioPPISVMExtractor:A Protein-Protein Interaction Extractor for Biomedical Literature Using SVM and Rich Feature Sets," J. Biomed. Inform., vol.43, pp.88-96, 2010.

[28] Z.H. Yang, H.F. Lin, and B.D. Wu, "BioPPIExtractor: A Protein-Protein Interaction Extraction System for Biomedical Literature," Expert Syst. Appl., vol.36, pp. 2228-2233, 2009.

[29] S.T.Ahmed, D. Chidambaram, H. Davulcu, and C. Baral, "IntEx: A Syntactic Role Driven Protein-Protein Interaction Extractor for Bio-Medical Text," Proc. the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics, Detroit, Michigan, 2005.