Mining Protein Interactions from Text Using Convolution Kernels

Ramanathan Narayanan[1], Sanchit Misra[1], Simon Lin[2], and Alok Choudhary[1]

[1] Department of Electrical Engineering and Computer Science, Northwestern University
[2] Feinberg School of Medicine, Northwestern University

Abstract. As the sizes of biomedical literature databases increase, there is an urgent need to develop intelligent systems that automatically discover Protein-Protein interactions from text. Despite resource-intensive efforts to create manually curated interaction databases, the sheer volume of biological literature databases makes it impossible to achieve significant coverage. In this paper, we describe a scalable hierarchical Support Vector Machine(SVM) based framework to efficiently mine protein interactions with high precision. In addition, we describe a convolution tree-vector kernel based on syntactic similarity of natural language text to further enhance the mining process. By using the inherent syntactic similarity of interaction phrases as a kernel method, we are able to significantly improve the classification quality. Our hierarchical framework allows us to reduce the search space dramatically with each stage, while sustaining a high level of accuracy. We test our framework on a corpus of over 10000 manually annotated phrases gathered from various sources. The convolution kernel technique identifies sentences describing interactions with a precision of 95% and a recall of 92%, yielding significant improvements over previous machine learning techniques.

1 Introduction

Protein-Protein interactions are the associations of protein molecules with one another, and the study of these associations from the perspective of biochemistry, signal transduction and protein networks. Protein interactions form the basis for virtually every process in a living cell, and the study of interactions improves the understanding of diseases and provides researchers with the ability to analyze and study therapeutic approaches.

Over the past few years, a multitude of high throughput methods to detect protein interactions have been developed. Researchers all over the world are performing these experiments and reporting results on protein-protein interactions. Although a few large-scale studies are available, most of the protein-protein interactions come from thousands of smaller studies. Figure 1 shows that the number of articles in Pubmed over the last 50 years is steadily increasing. Consolidating the known list of protein-protein interactions will provide researchers with a powerful tool that will greatly enhance their understanding of these relationships on a genomic scale.

T. Theeramunkong et al. (Eds.): PAKDD Workshops 2009, LNAI 5669, pp. 118–129, 2010.

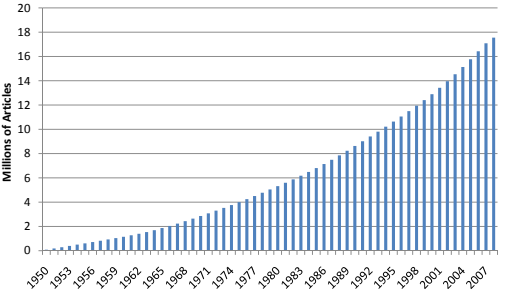

Fig. 1. Growth of the Pubmed Database over the last few years

There have been several attempts to develop databases of interacting proteins, and the supporting metadata that describes an interaction. Currently, there are several manually curated databases like DIP, BIND and MINT [10,1,4]. While these databases promise a high degree of accuracy, by providing supporting evidence reviewed by a subject expert, their coverage leaves much to be desired. DIP contains 56493 records referring to protein interactions whereas MINT describes 105537 interactions. However, there is a common consensus that the number of interactions available in these manually curated databases is miniscule compared to the total number of interactions described in the literature. Typically, manually curated databases cover about 3000-4000 articles. Considering that Pubmed, the biomedical literature database maintained by National Institute of Health (NIH) contains over 17 million documents, we can clearly see the limited usefulness of manually curated databases.

Machine learning and data mining techniques have been applied to develop automated and semi-automated methods for finding protein interactions on a large scale. The highly ambiguous nature of natural languages and lack of standards in research writing complicate this task tremendously. The lack of standards in protein nomenclature has led to a single protein being referred to by several synonyms, with no apparent similarities (eg. Trehalose-phosphatase, EC 3.1.3.12 and TPP refer to the same protein). Therefore, the use of standard exhaustive dictionaries to extract protein/gene references [2,17] have not been very effective. Existing machine learning approaches [15,14] have shown some success, but suffer from lack of scalability, inability to adapt to different domains and small test sets. Semi-automated techniques like PreBIND use SVMs to speed up manual expert reviewing by reducing the amount of information to be examined. The challenges of manual curation can be addressed by developing fully automated systems to extract interactions from abstracts or full text articles. However, fully automated methods suffer from low sensitivity and a lack of coverage. Since there are a large number of ways in which interactions can be described, simple rule-based approaches, relying on human-generated rules to recognize phrases, can only capture a limited percentage of interactions. At the same time, using a complex but accurate machine learning technique may not be computationally feasible, nor achieve the desired coverage. Previous efforts have also been hampered by the lack of quality training datasets.

However, we utilize the highly accurate protein interaction information in interaction databases like DIP and MINT to serve as a vast, diverse training platform for our supervised learning system. In this paper, our contributions are

- We propose a scalable hierarchical Support Vector Machine(SVM) based framework to efficiently mine protein interactions from Pubmed abstracts with high precision.
- We propose the use of Convolutions kernels in Support Vector Machines for mining protein-protein interactions.
- We validate our technique on a large corpus gathered from various manually annotated databases, and achieve high rates of precision and recall.

2 Kernel Methods and Support Vector Machines

Support vector machines (SVMs) are a set of supervised machine learning techniques used for classification and regression. SVMs have been shown to be highly successful for text classification[11]. [19] contains a comprehensive discussion of support vector machine theory. SVMs are able to solve a multitude of classification problems by using domain-specific and cost-sensitive kernel functions. In our work, we use the concept of tree kernels to efficiently determine the similarity between the syntactic parse trees of sentences. As a result, we are able to exploit the syntactic structure of natural language text and develop a better classification model when compared to a traditional SVM.

3 Hierarchical Approach

We tackle the problem of finding protein interactions in Pubmed abstracts using a three-stage hierarchical approach. Figure 2 shows a brief overview of our approach, and the various stages involved.

Stage I: Pubmed contains a vast number of abstracts (around 17 million), most of which do not contain descriptions of protein interactions. Comprehensive analysis of each of these abstracts is time-consuming and wasteful in terms of computational resources. Therefore we need a way to determine which documents are worth looking at in greater detail. This is a classic example of a text classification problem, and we propose a solution using a simple SVM formulation.

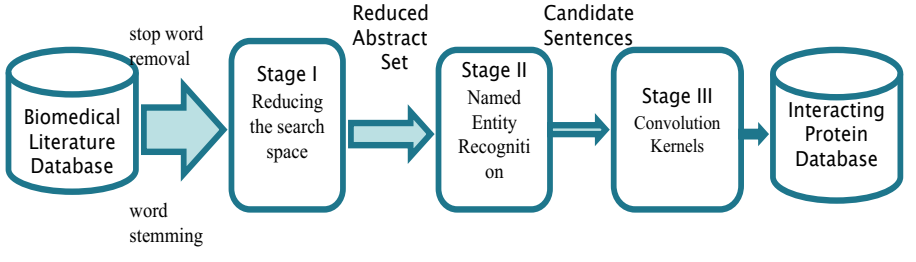

Fig. 2. System architecture for Mining Protein-Protein Interactions

Stage II: Once we have determined those abstracts which are likely to contain interactions, we need to narrow down the list of possibilities furthur. Occurrence of proteins (in general, biomedical named entities) is a strong requirement for description of an interaction. In other words, it is very likely that a sentence containing two or more protein names actually describes an interaction amongst them. Also a sentence containing no references to proteins in unlikely to contain an interaction. To find proteins in text, we use a comprehensive SVM-based method, that yields high recall, thus finding most protein references, and by implication, most sentences that describe interactions.

Stage III: Finally, to determine if a sentence describes an interaction or not, we must understand the inherent meaning of the sentence. This can be a tricky proposition, considering the richness and diversity of the English language. However, by studying the syntactic and semantic structure of the text, we can determine, with high accuracy, whether or not it describes an interaction. For this purpose, we introduce the concept of convolution kernels to accurately identify sentences describing interactions. While this is a resource intensive technique, we only apply it to those candidate sentences which have been obtained after the first two stages.

3.1 Reducing the Search Space

In our problem, we require a method to prune the search space, so that we can apply more accurate and complex machine learning techniques on a reduced candidate set. We have used SVMs with a *Bag-of-Words* approach to build a classifier model which successfully distinguishes those abstracts which may contain a protein interaction. The Bag-of-words technique is a simple approach that relies on word-frequency information to classify text documents. Each abstract is converted into a high dimensional feature vector, after performing necessary preprocessing steps like stemming and stop-word removal. We used the TFIDF [11] metric to represent features in the input vector. The details of the training and validation phases are provided in the results section. The end-result of this phase is that we are able to identify those documents which are likely candidates for containing protein-protein interactions, with an acceptable false-positive rate.

3.2 Biomedical Named Entity Recognition (NER)

Stage II involves the recognition of biomedical named entities(more specifically, proteins) in text. In this work, we do not distinguish between proteins and other named entities(NEs), since it is difficult to make such a distinction. Each word is represented by a binary feature vector, which consists of lexical features, orthographical and morphological features, Part-of-Speech features, and context features(as in [13]). The basic idea is that using these features, the SVM will be able to determine whether the word is a biomedical named entity. The features we use are similar, but not identical to those in [8]. Details of these features are discussed below:

Lexical Features: The lexical feature set of a word consist of three lists of terms: single-term list, functional-term list and general-term list. Each term in these lists corresponds to a feature. Single-terms are words which can be used as

an protein entity by themselves, such as 'NF-KappaB', 'astrogen' or 'ERK'. Functional terms are devised to describe the function and characteristics of NEs. They have no special orthographic features and consist of all lower case words which frequently appear in NEs (eg. 'protein','gene','receptor' are all functional words). General terms are a set of words that are classified neither as single-terms nor function terms. This is simply a list of words with frequency greater than three in the training corpus.

Orthological and Morphological Features: A large number of NEs contain surface clues (called orthological features), which may help in discriminating them. We use a list of sufficiently powerful orthological features in our model(see Figure 3). We also utilize commonly occurring suffix patterns in NEs as a predicting feature.

Part-of-Speech(POS) Features: The part-of-speech tags of a target word and its surrounding context words represent syntactic characteristics for composing an entity. It is commonly seen that named entities are tagged as nouns or adjectives, whereas they are rarely tagged as adverbs.

Contextual features: For boundary identification, we use neighboring words and the POS of neighboring words as contextual features. In our experiments, we considered the two words to the left and right of the target word. Context words are selected as features only if they belong to one of the lexical feature lists.

Therefore, the feature vector for a target word is obtained by composing all of the features described above. For each feature, the binary feature vector has a 1 if that feature is present, or 0 otherwise. We use a SVM with standard settings and a linear kernel. Once biomedical entities are recognized, we will be able to further narrow down the list of candidate sentences which may contain protein interactions.

3.3 Convolution Kernels

Stage III represents the most complex of all our SVM formulations. The input is a sentence containing several biomedical named entities. However, the mere presence of a protein does not guarantee that the sentence will describe a protein interaction (See Figure 5). Also the use of the Bag-of-Words and similar techniques does not allow us to exploit the syntactic structure of the phrase in question. In recent years, tree kernels have been shown to be interesting approaches for the modeling of syntactic information in natural language tasks [21,22,23].

Given an input sentence, we can obtain the corresponding syntactic parse tree using standard natural language processing techniques. Figure 4 shows the parse tree of the sentence 'IL-5 activated the Jak 2-STAT 1 signaling pathway'. The kernels that we consider represent parse trees in terms of their substructures(fragments). These substructures define the feature space of a tree, which is represented as a high-dimensional vector. The kernel function attempts to find the similarity between two trees by counting the number of their common fragments. When this kernel function is plugged into a SVM, it detects if a parse tree belongs to the feature space of the known examples belonging to the target class, thereby accomplishing the classification task.

Feature ID	Example
Cap + Digit	IL22
Alpha-numeric	Ama292
Cap + Digit + Cap	L809TR
Greek	NF-KappaB
Lower+digit+sym	110-nt
Unit	U/mg
HasDot	a.m
Has_special_symbol	Doc1/Apc10

Fig. 3. An example of Orthological Features used for NER

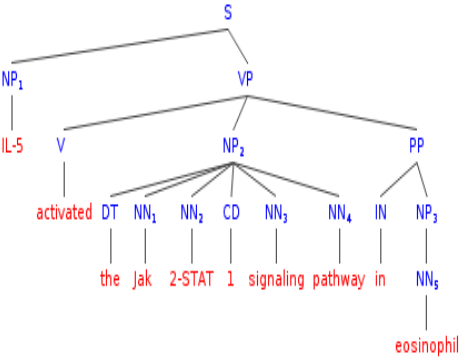

Fig. 4. Parse Tree

In a syntactic parse tree, each node with its children is associated with a grammar production rule, where the symbol on the left-hand side corresponds to the parent node, and the symbols on the right hand side are associated with the child nodes. We define as a subtree(ST) any node of a tree along with all its descendants. A subset tree (SST) is a tree obtained by applying the same grammatical rule set which generated the original tree. The main idea behind tree kernels is to compute the number of common substructures between two trees T_1 and T_2 without explicitly considering the whole fragment space. We base our tree kernel on the method proposed in [5], along with a modification that enables us to evaluate differences between features spaces generated from STs or SSTs.

We formally represent a parse tree feature space (ST or SST) as $T = (f_1, f_2, ... f_{|\mathcal{N}|})$, where f_i represents a ST or SST fragment. Define a function $F_i(n)$ to be 1 if fragment f_i is rooted at node n of T, and 0 otherwise. The kernel function K between two trees T_1, T_2 is defined as follows: $K(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \sum_{i=1}^{|\mathcal{N}|} F_i(n_1) F_i(n_2)$.

Here N_{T_1}, N_{T_2} represent node sets of T_1, T_2, and $|\mathcal{N}|$ represents the size of the feature space.

It can be shown that computing the kernel function as-it-is can lead to an exponential number of evaluations in the size of the input tree. However, we can compute the inner product ($\sum_{i=1}^{|\mathcal{N}|} F_i(n_1) F_i(n_2)$, designated as $IP(n_1, n_2)$) in an efficient manner as follows:

- If the productions at n_1 and n_2 are different, then $IP(n_1, n_2) = 0$
- If the productions at n_1 and n_2 are the same, and n_1 and n_2 have only leaf children, then $IP(n_1, n_2) = \lambda$
- If the productions at n_1 and n_2 are the same, and n_1, n_2 are not pre-terminals, then

$IP(n_1, n_2) = \lambda \prod_{j=1}^{numchild(n_1)} (\sigma + IP(c_{n_1}^j, c_{n_2}^j))$

where c_n^j is the j^{th} child of node n. When $\sigma = 0$, we evaluate the ST kernel, and when $\sigma = 1$, the SST kernel is evaluated. λ is a decay parameter that determines the importance of the tree fragment length in the kernel evaluation.

The complexity of evaluating the above kernel function is quadratic in the length of the trees. However using several optimization techniques, the kernel evaluation can run in linear time on average. Therefore, in Stage III we generate the Part-of-Speech tags for each sentence and then generate the corresponding syntactic parse tree. In addition to the parse tree representation, we also generate a bag-of-words feature vector for each candidate sentence. This is because we would like to combine the word-frequency information with the syntactic tree kernel to generate a hybrid kernel. The details of the training and test datasets, implementation and results are discussed in the next section.

4 Experimental Results

In order to verify the effectiveness of our system, we perform extensive testing to gauge the performance of all three stages individually and as a group. We use SVM-light [12] software for all our experiments. The metrics we use to determine performance are Precision, Recall and F-1 score.

Stage I: We generated a training set of 3730 Pubmed abstracts (2230 positive cases and 1500 negative cases). The positive examples were those abstracts in DIP(Database of Interacting Proteins) which have been manually curated and found to contain protein interactions. The negative examples were derived by selecting a subset of Pubmed abstracts which had no interaction keywords, and then were reviewed by a subject expert. After performing preprocessing steps like word stemming and stop word removal, the SVM was trained with these examples using a linear kernel and default settings. Testing the model on a set of 500 positive examples and 1000 negative examples(generated from the same sources as described above) yielded a precision of 97.5% and recall of 92%. For further testing, we obtained a set of 3121 positive examples from another manually curated database, MINT, and attempted to classify those abstracts using our SVM model. The model was able to successfully predict that 96.4% of these abstracts described protein interactions. Therefore, with a lightweight technique, we are able to dramatically reduce the search space, without generating too many false negatives.

Stage II: The Genia corpus [9] is a large human subject expert annotated database which serves as an ideal platform to train and test supervised learning systems. In the Genia corpus, 2000 abstracts from Pubmed have been manually annotated to denote biological entities like proteins, genes, DNA etc. as well as language constructs like sentences, abbreviations and titles. The number of 'protein' terms alone is 10504, which indicates the richness of this corpus.This serves as an ideal testing platform to measure the effectiveness of the Named entity recognition module. We used the GENIA POS tagger [9] for part-of-speech tagging, and computed the various lexicographic and orthographic features for each word as described in the earlier section. Inspite of inconsistent naming conventions and punctuation, our named entity recognition module was able to

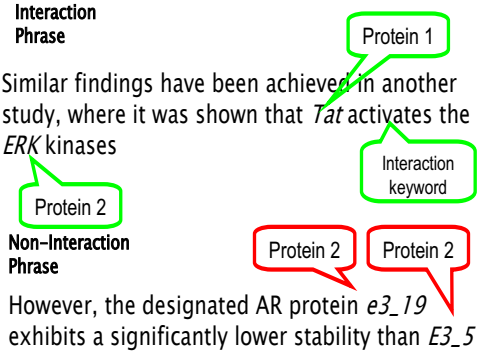

Fig. 5. Positive and Negative examples from the Input Corpus

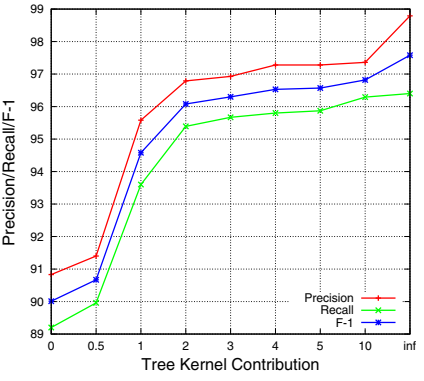

Fig. 6. Performance comparison of the Hybrid kernel for different values of τ

successfully identify biomedical named entities (NEs) with a precision of 79% and a recall of 77%.

Stage III: In order to generate an appropriate dataset to test the convolution kernel SVM formulation, we consider a list of protein interaction sentences generated by the BioText project [16]. This dataset contains a list of 3190 sentences which describe protein interactions of various kinds. A list of 7000 negative examples was generated in a semi-automated fashion by gathering sentences which contained protein references, yet did not contain interaction keywords. Figure 5 shows examples of positive and negative cases in the corpus. The sentences were divided into training and test datasets. The Genia POS tagger was used for part-of-speech tagging, and the Collins parser [6] was used to generate syntactic parse trees.

In our experiments, we first compare the performance of the convolution kernel based method with a Bag-of-words approach. Further, we use a combination of both these approaches to generate a hybrid model. The kernel formulation for two input examples x_1:(T_1, v_1), x_2:(T_2, v_2) where T_i represents the syntactic parse tree, and v_i represents the traditional bag-of-words feature vector is as follows:

$$K(x_1, x_2) = \tau K_t(T_1, T_2) + K_{bow}(v_1, v_2),$$

where τ is a parameter that represents the contribution of the tree kernel, K_t represents the tree kernel, and K_{bow} represents a traditional SVM kernel on a bag-of-words feature vector(radial,gaussian etc). Figure 6 shows the variation in precision and recall as we vary the contribution of the tree kernel (parameter σ is set to 0). It can be seen that the usage of a tree kernel significantly improves the precision and recall values. This is because the tree kernel takes into account the syntactic structure of a sentence, and is able to capture the inherent semantics in a superior way, as compared to a normal bag-of-words feature representation. In fact, the use of word frequency information may actually bias the classifier and produce erroneous results. The highest precision and recall values are when $\tau = \infty$, or in other words, when only the tree kernel is considered.

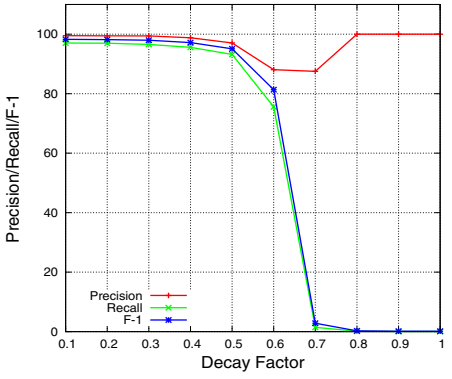

Fig. 7. Performance comparison of the Tree kernel for different decay factors

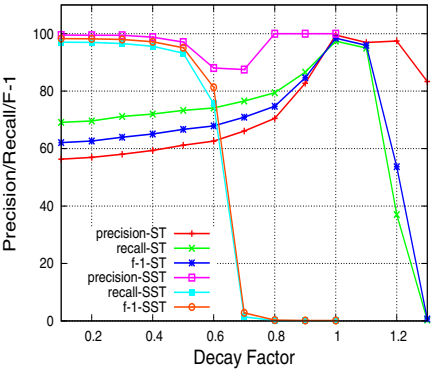

Fig. 8. Performance comparison of Subtree Kernel vs. SST Kernel

We also study how our classification model changes when we change the value of λ, the tree decay parameter. Intuitively, λ adjusts the importance of tree fragments according to their size. A higher value of λ indicates that we are concerned about the details of the fragment structure, especially for larger sentences. Figure 7 shows how the precision, recall and F-1 values vary for λ in the range (0,1]. The performance of the kernel remains relatively stable for λ in the range [0.1-0.5], with a dramatic drop around 0.6. At this stage, the precision values reach 100%, but the recall falls to near-zero levels. This indicates that our SVM based model performs better when we consider the local structure of the fragments, without giving too much weight to the details. The ideal choice of λ is determined to be 0.4.

We also ran experiments testing the performance of the subtree kernel as compared to the subset tree kernel. The difference between these kernels is the value of σ. For the subtree Kernel, $\sigma = 0$, and for the subset tree kernel $\sigma = 1$. The results described earlier are for the subset tree kernel. Figure 8 shows the comparative performance between the subtree and subset tree kernels for different values of the decay parameter. The subtree kernel performance for lower values of λ is poor, but it gradually improves till at $\lambda = 1$, the performance matches the best case performance of the subset tree kernel. The optimum choice of λ for the subtree kernel is 1. Overall, we were able to achieve an average precision of 95% and average recall of 92% using the convolution kernel technique.

Finally, to test the system as a whole, we ran experiments to evaluate the performance on a dataset of 1584 Pubmed abstracts. After Stage I, we were able to identify 95% of those abstracts which did contain an interaction. Similarly, Stage II was able to identify 82% of all protein references. Finally in Stage III we were able to identify 95% of those sentences which contained protein interactions. We also tested our technique on the Biocreative 2[27] dataset. However, the toughest task in this dataset is the Named Entity Recognition, and the precision and recall of the results are highly influenced by this step. Since that is not the major focus of this work, we have evaluated only Stage III of our technique using

this dataset. The convolution kernel technique yields a precision of 93% and a recall of 89% on a test set comprising of over 2000 sentences.

Previously used techniques like [18,8,20] make use of manually constructed interaction phrases, as well as context free grammars to describe interaction patterns between biological entities. The interactions generated by these techniques are limited by the coverage of the recognition rules, as well as the inherent complexity and variability in sentences describing interactions. These methods achieve high precision rates (above 90%), but suffer from extremely low recall rates (around 20-30%). Another popular approach is to use machine learning and statistical techniques [15,14], which have been shown to have higher recall. However, the major issues are scalability and portability. State-of-the-art text mining systems for protein interaction mining([3]) which offer full automation are tested on datasets of only a few hundred articles. In contrast, we have validated our results using a much larger corpus. Shin et al[28] also use tree kernels to identify interaction sentences. However, our work differs from theirs in that we explore a variety of different kernels (and combinations of convolution/bow kernels), and perform extensive performance comparisons. Though the convolution kernel technique performs impressively, as a whole, the overall recall of our system is low (around 60%). This is because there is loss of interaction information in each stage.

5 Conclusions and Future Work

In this paper, we have highlighted the need for a scalable, accurate method for predicting protein-protein interactions from text articles. We propose a hierarchical Support Vector Machine based system to efficiently mine these interactions from Pubmed abstracts. In our three-stage system, we use simple word-frequency based SVM formulation in Stage I, a slightly more complex Named Entity Recognition module in Stage II, and a sophisticated, accurate convolution kernel-based method in Stage III. At each stage, the complexity of the technique used increases. However, a large number of redundant documents are eliminated in the earlier stages, thereby reducing the overall workload. In doing so, we achieve a reasonable performance/accuracy tradeoff. Extensive testing on real-world datasets yields reasonable results. In the future, we plan to address the problem of extracting the specific interaction type. Also, while abstracts contain useful information, a vast body of knowledge lies hidden in full text articles. It will be useful to extend our technique to mine interactions from full text articles.

Acknowledgements

This work was supported in part by NSF grants CNS-0551639, IIS-0536994, NSF HECURA CCF-0621443, and NSF SDCI OCI-0724599 and DOE SCIDAC-2: Scientific Data Management Center for Enabling Technologies (CET) grant DE-FC02-07ER25808.

References

1. Alfarano, C., et al.: The Biomolecular Interaction Network Database and related tools 2005 update. Nucleic Acids Res. 33, D418–D424 (2005)
2. Blaschke, C., et al.: Automatic extraction of biological information from scientific text: protein-protein interactions. In: Proc. Int. Conf. Intell. Syst. Mol. Biol., pp. 60–67 (1999)
3. Brown, K.R., et al.: Online predicted human interaction database. Bioinformatics 21, 2076–2082 (2005)
4. Chatr-aryamontri, A., et al.: MINT: the Molecular INTeraction database. Nucleic Acids Res. 35, D572–D574 (2007)
5. Collins, M., Duffy, N.: New ranking algorithms for parsing and tagging: Kernels over discrete structures (2002)
6. Collins, M.: Head-Driven Statistical Models for Natural Language Parsing. Computational Linguistics (2003)
7. Donaldson, I., et al.: PreBIND and Textomy–mining the biomedical literature for proteinprotein interactions using a support vector machine. BMC Bioinformatics 4, 11 (2003)
8. Fukuda, K., et al.: Toward information extraction: identifying protein names from biological papers. In: Pac. Symp. Biocomput., pp. 707–718 (1998)
9. Genia Project: Mining literature for knowledge in molecular biology (2008), `http://wwwtsujii.is.s.u-tokyo.ac.jp/GENIA/home/wiki.cgi`
10. Gilfillan, I.: A database of proteins that are known to interact. Genome Biology 1; Reports220 (November 2000)
11. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998)
12. Joachims, T.: Making large-scale SVM learning practical. In: Advances in Kernel Methods-Support Vector Learning (1999)
13. Lee, K.J., Hwang, Y.S., Kim, S., Rim, H.C.: Biomedical named entity recognition using two-phase model based on SVMs. J. Bio. med. Inform. 37, 436–447 (2004)
14. Marcotte, E.M., et al.: Mining literature for protein-protein interactions. Bioinformatics 17, 359–363 (2001)
15. Ramani, A.K., et al.: Consolidating the set of known human protein-protein interactions in preparation for large-scale mapping of the human interactome. Genome Biol. 6, R40 (2005)
16. Rosario, B., Hearst, A.: Multi-way Relation Classification: Application to Protein-Protein Interaction. In: Human Language Technology Conference on Empirical Methods in Natural Language Processing (2005)
17. Rindflesch, T.C., et al.: Mining molecular binding terminology from biomedical text. In: Proc. AMIA Symp., pp. 127–131 (1999)
18. Temkin, J.M., Gilder, M.R.: Extraction of protein interaction information from unstructured text using a context-free grammar. Bioinformatics 19, 2046–2053 (2003)
19. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, Heidelberg (1995)
20. Yu, H., et al.: Automatic extraction of gene and protein synonyms from MEDLINE and journal articles. In: Proc. AMIA Symp., pp. 919–923 (2002)
21. Culotta, A., Sorensen, J.: Dependency Tree Kernels for Relation Extraction. In: Proceedings of ACL 2004 (2004)
22. Bunescu, R., Mooney, R.J.: Subsequence kernels for relation extraction. In: Proceedings of the 19th Conference on Neural Information Processing Systems, Vancouver, British Columbia (2005)

23. Collins, M., Duffy, N.: Convolution kernels for natural language. In: NIPS 2001 (2001)
24. Yuka, T., Tsujii, J.: Part-of-Speech Annotation of Biology Research Abstracts. In: The Proceedings of 4th International Conference on Language Resource and Evaluation (LREC 2004), Lisbon, Portugal, May 2004, pp. 1267–1270 (2004)
25. Collins, M.: A New Statistical Parser Based on Bigram Lexical Dependencies. In: Proceedings of the 34th Annual Meeting of the ACL, Santa Cruz
26. Porter, M.F.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
27. Biocreative 2: `http://biocreative.sourceforge.net/biocreative_2.html`
28. Shin, et al.: Identifying Protein-Protein Interaction Sentences Using Boosting and Kernel Method. In: Second BioCreative Challenge Evaluation Workshop (2007)