Mining Protein Interaction from Biomedical Literature with Relation Kernel Method

Jae-Hong Eom and Byoung Tak Zhang

Biointelligence Laboratory,
School of Computer Science and Engineering,
Seoul National University Seoul 151-744, South Korea
{jheom, btzhang}@bi.snu.ac.kr

Abstract. Many interaction data still exist only in the biomedical literature and they require much effort to construct well-structured data. Discovering useful knowledge from large collections of papers is becoming more important for efficient biological and biomedical researches as genomic research advances. In this paper, we present a relation kernel-based interaction extraction method to extract knowledge efficiently. We extract protein interactions of from text documents with relation kernel and *Yeast* was used as an example target organism. Kernel for relation extraction is constructed with predefined interaction corpus and set of interaction patterns. The proposed method only exploits shallow parsed documents. Experimental results show that the proposed kernel method achieves a recall rate of 79.0% and precision rate of 80.8% for protein interaction extraction from biomedical document without full parsing efforts.

1 Introduction

It is known that protein interactions are primary biochemical reactions in the organisms and they take part in important functions since they determine the biological processes [1]. Therefore, broad description and in depth analysis of these interactions would significantly help the understanding of many important biological phenomena. For *S.cerevisiae* (budding yeast), many researchers have undertaken the task of functionally analyzing the yeast genome comprising more than 6,300 proteins (YPD) [2] and abundant interaction data have been produced by many research groups after the completion of the genome sequences.

Many machine learning-based promising methods have been successfully applied to the analysis of these data. One of these methods is knowledge discovery from text data called '*text mining*.' Even though there are many databases for protein interactions, many interaction data still exist only in the biomedical literatures. They are spread in biomedical literature written in the form of natural languages and they require much effort such as data mining for constructing well-structured data forms. Thus, the studies on the methodology of how to extract protein interactions from biomedical literature have been an active research subject. Many approaches widely adopted natural language processing (NLP) techniques for this problem. Many of these methods can be regarded as the parsing based methods and both full and shallow parsing strategies have been attempted.

A general full parser approach with domain specific grammars was proposed by Yakushiji *et al.* [3]. They used general full parser with grammars for biomedical domain to extract interaction events by filling sentences into augmented structures. Park *et al.* [4] proposed bidirectional incremental full parsing with combinatory categorical grammars. Temkin *et al.* [5] also utilized a lexical analyzer and context-free grammar to extract gene, protein, and interactions of small molecules. They achieved a recall rate of 63.9% and precision rate of 70.2%. As a similar method, template generating method via preposition-based parsing proposed by Leroy *et al.* [6]. They achieved precision of 70% for biomedical literature abstract processing. For a partial parsing method, Pustejovsky *et al.* [7] used the relational parsing for the inhibition relation with recall rate of 57%. But, these methods are intrinsically complicated and the performance was not satisfactory.

In this paper, we present a relation kernel-based interaction extraction method which extract protein interactions of *S.cerevisiae* from biomedical literature. We define kernel for protein relation extraction, called '*relation kernel*', with predefined interaction corpus and set of interaction patterns. Proposed kernel method for interaction extraction only exploits shallow parsing results.

This paper is organized as follows. In Section 2, we describe the basic concept of kernel method. In Section 3, relation kernel for protein interaction extraction is explained. In Section 4, we show experimental results of interaction extraction. Finally, Section 5 we present concluding remarks and future directions.

2 Concept of Kernel Method

Kernel methods are an alternative to feature-based methods. Kernel methods preserve the original representation of objects and use the object in algorithms only via computing a kernel function between a pair of objects. Kernel function K over the object space X is binary function K: $X{\times}X \rightarrow [0, \infty]$ mapping a pair of objects x, $y \in X$ to their similarity score $K(x, y)$. This is embedding procedure of data items into a vector space F, called a *feature space*, and searching for linear relation in this space. This embedding is defined implicitly by specifying an inner product for the feature space via a symmetric and positive semidefinite *kernel function*: $K(x, y) = \langle \Phi(x), \Phi(y) \rangle$, where $\Phi(x)$ and $\Phi(y)$ are the embeddings of data items x and y. Additionally, any kernel function implicitly calculates the dot product of feature vectors of objects in high-dimensional feature spaces and a function defined as a dot product of the corresponding feature vectors is necessarily a kernel function [8].

In many cases, it is possible to compute the dot product of certain features without enumerating all the features. An excellent example is that of subsequence kernels [9]. In the subsequence kernels, the objects are strings of characters, and the kernel function computes the number of common subsequences of characters in two strings, where each subsequence match is additionally decreased by the factor reflecting how spread out the matched subsequence in the original sequences. Despite of exponential number of features (subsequences), it is possible to compute the subsequence kernel in polynomial time. Therefore, one can exploits long-range features in string without enumerating the features explicitly.

There are a number of learning algorithms that can operate only using the dot product of examples. The models produced by the learning algorithms are also expressed using only dot product of example. Substituting a particular kernel functions in place of dot product defines a specific instantiation of such learning algorithms

The Support Vector Machine (SVM) is known as a learning algorithm that not only allows for a dual formulation, but also provides a rigorous rationale for resisting overfitting. For a kernel-based algorithms working in extremely rich feature spaces, it is crucial to deal with the problems of overfitting. Many experimental results indicate that SVM is able to generalize the classification boundary very well and avoid overfitting in high dimensional feature spaces effectively. So, here we use SVM and relation kernel (extended version of subsequence kernel for shallow parsing) to extract protein interactions.

3 Relation Kernel for Protein Interaction Extraction

Here we use the basic notion and the idea of subsequence kernel and extend it to operate on shallow paring for protein relation extraction according to the work of Zelenko *et al.* [8] and Lodhi *et al.* [9].

For the example sentence, "In those pathways, usually the yeast protein RAP1 suppresses the up-regulation of SH3", we can generate appropriate parse tree(s) for this sentence and compare with the other parse trees generated from other sentences by conventional shallow parsing approach. The form of target relation that we want to extract in this example sentence is 'RAP – suppress – SH3' which generally has the form of 'entity 1 – action verb – entity 2.'

In this paper, we use shallow parsing only for identifying its key elements such as entities and interactions instead of providing full interpretation of the sentence because this approach may provides us a fairly robust shallow parsing and the generation ability of structured representations even for ungrammatical sentences.

Next, we convert the shallow parse tree into examples for the '*entity–interaction–entity*' relation. This relation describes protein interactions such as gene–gene, gene–protein, and protein–protein. The type of the former and the later entity includes those names of gene and protein, and the type 'relation' holds between a two 'entity' and an 'interaction.' For each example sentence, we check whether the sentence has complete relation structure which has two entities and one interaction verb structures.

Each node of example (parse tree of sentence) has its 'type' and 'role.' We define *Type* = {φ, GeNP, GeN, PrNP, PrN, VerbPos, VerbNeg, VerbUnk, PNP, NP, Prep, Adj, Det} and *Role* = {φ, Entity_1, ActionVerb, Entity_2, UNK}. The GeNP and GeN of 'type' mean a noun phrase or a noun which represents a name of gene or protein. Verbs are classified into three categories; positive (e.g., activate, up-regulate, etc.), negative (e.g., inhibit, prohibit, etc.), and other unknown action verbs (VerbUnk). Generally, 'Type' represents the category of par-of-speech (POS) tagged results. On the other hand, 'Role' represents the role of 'type' in the sentence to construct the structure of protein interaction such as '*entity–interaction–entity*.'

For two relation examples P_1, P_2, we use the similarity function $K(P_1, P_2)$ in terms of similarity function of the parent nodes. That is,

$$K(P_1,P_2)=\begin{cases}0, \text{ if } t(P_1.p,P_2.p)=0,\\ k(P_1.p,P_2.p)+K_c(P_1.c,P_2.c), \text{ otherwise.}\end{cases} \tag{1}$$

where, $t(\cdot,\cdot)$ is a matching function and $k(\cdot,\cdot)$ is a similarity function on nodes and it has '1' if two nodes have same type and role in the parsing tree, and '0', otherwise. The function, $k(\cdot,\cdot)$, has '1' if the two nodes have same text and '0', otherwise [9].

We also use similarity function K_c of the children node of parse tree in terms of similarities of children subsequences. That is,

$$K_c(P_1.c,P_2.c)=\sum_{\mathbf{i},\mathbf{j},l(i)=l(j)}\lambda^{d(\mathbf{i})+d(\mathbf{j})}K(P_1[\mathbf{i}],P_2[\mathbf{j}])\prod_{s=1,\ldots,l(\mathbf{i})}t(P_1[i_s].p,P_2[j_s].p). \tag{2}$$

where, $\mathbf{i}$ denote a sequence of indices and $d(\mathbf{i}) = l(\mathbf{i}) = i_n-i_1+1$ denote length of the sequence $\mathbf{i}$. In the Equation 2, the term $P[\mathbf{i}]$ represents the sequence of children $[P[i_1],\ldots,P[i_n]]$ and $K(P_1[\mathbf{i}],P_2[\mathbf{j}])$ stands for summing up $K(P_1[i_s],P_2[j_s])$ for all $s = 1,\ldots,l(\mathbf{i})$ [8]. Equation 2 considers all subsequences of relation example children with matching parents and it accumulates the similarity for each subsequence by adding similarities of the corresponding child examples. The equation also reflects the amount of spread of the subsequences within children sequences by decreasing overall similarity with the factor λ which has the value from 0 to 1.

Generally, the index ($\mathbf{i}$ and $\mathbf{j}$) of two example parse tree will be different. So, we can't use Equation 2 for similarity function K_c of the children node of parse tree directly. Thus, here we use the derived recurrences of Zelenko *et al.* [8] and its original construction recurrences from Lodhi *et al.* [9] to calculate K_c. That is,

$$\begin{aligned}
&K_C=\sum_{q=1,\ldots,\min(m,n)}K_{c,q}(m,n).\\
&K_{c,q}(i,j)=\lambda K_{c,q}(i,j-1)+\sum_{s=1,\ldots i}t(P_1[s].p,P_2[j].p)\lambda^2C_{q-1}(s-1,j-1,K(P_1[s],P_2[j])).\\
&C_q(i,j,a)=aC_q(i,j)+\sum_{r=1,\ldots q}C_{q,r}(i,j).\\
&C_q(i,j)=\lambda C_q(i,j-1)+C'_q(i,j).\\
&C'_q(i,j)=t(P_1[i],P_2[j])\lambda^2C_{q-1}(i-1,j-1)+\lambda C'_q(i,j-1).\\
&C_{q,r}(i,j)=\lambda C_{q,r}(i,j-1)+C'_{q,r}(i,j).\\
&C'_{q,r}(i,j)=\begin{cases}t(P_1[i],P_2[j])\lambda^2C_{q-1,r}(i-1,j-1)+\lambda C'_{q,r}(i,j-1) & \text{if } q\neq r,\\ t(P_1[i],P_2[j])\lambda^2K(P_1[i],P_2[j])C_{q-1}(i-1,j-1)+\lambda C'_{q,r}(i,j-1) & \text{if } q=r.\end{cases}
\end{aligned} \tag{3}$$

where, a condition $m \geq n$ is assumed for the number of children of P_1 and P_2. The boundary conditions for calculating K_c are the same with the conditions of Lodhi *et al.* [9] and not presented here.

4 Experimental Results

Data

We used 504 labeled MEDLINE records containing at least one protein interaction as positive examples (50% for training, 50% for testing). About 320 negative examples were collected from MEDLINE which contain gene or protein names but any interactions at all. The shallow parses of each record was generated by our custom shallow parsing system and the Brill tagger trained with GENIA corpus was used to guarantee more accurate POS tagging. Table 1 shows the statistics of positive examples.

Table 1. The statistics of positive examples for the experiment

Categories	# of Positive Interaction	# of Negative Interaction	Total
# of Interaction	584	302	886

Algorithm Settings

We set parameter λ of kernel computation to 0.5 from experimental search to reflect the amount of spread of interaction structure (gene names, interaction verbs, and protein names). We evaluated kernels based relation extraction result in comparison with the general feature based baseline classifier, the Naïve Bayes. Full parsing results were used as a feature set for the Naïve Bayes. In this paper, this baseline model represents the method of full parsing based relation extraction approach.

Results

The extraction performance was measured by *precision*, *recall*, and *F-measure*. Precision is the ratio of the number of correctly predicted positive examples to the number of predicted positive examples. Recall is the ratio of the number of correctly predicted positive examples to the number of true positive examples. F-measure combines precision and recall as follows:

$$F = \frac{2 * recall * precision}{(recall + precision)}. \tag{4}$$

Table 2 shows the performance results of the protein relation extraction experiments.

Table 2. The extraction performance of relation kernel method on interaction extraction task for extracting interactions represented by major five interaction verbs.

Interaction Verb	Recall (%)		Precision (%)		Total *F* (%)	
	Kernel	*baseline*	*Kernel*	*baseline*	*Kernel*	*baseline*
Interact	81.11	79.16	80.27	79.36	80.69	79.25
Bind	80.83	78.55	82.12	78.54	81.46	78.54
Up-regulate	76.24	72.12	80.14	74.22	78.14	73.15
Suppress	79.92	76.96	81.91	81.84	80.90	79.32
Inhibit	76.91	76.54	79.90	78.79	78.37	77.64
Average	**79.00**	76.66	**80.8**	78.5	**79.91**	77.58

From Table 2, kernel based relation extraction outperform full parsing feature based baseline method (Naïve Bayes) overall. The number of interaction verb which generally describes protein interactions in the biomedical literature and recognized by this experiment was about 150 including 'activate', 'bind', 'control', 'enhance', 'inhibit', 'interact', 'prevent', etc. Some verbs occurred frequently but other occurred rarely.

5 Conclusions

In this paper, we presented kernel based protein relation extraction method and the result shows that kernel based method slightly outperform conventional feature based (full parsing information based) approach. The presented relation kernel only requires domain specific knowledge when we define matching function and similarity kernel function. We used the dictionary of protein names and its aliases as a domain specific knowledge. So it would be not difficult to expand this method to other domain by changing this domain knowledge simply.

But, this kernel method on NLP problem has one big drawback for practical use because it requires relatively high computational costs. Thus, the more efficient kernel computation method should be devised.

Acknowledgement

This research was supported in part by the National Research Laboratory Program of the Korea Ministry of Science and Technology.

References

1. Deng, M., et al.: Inferring Domain–Domain Interactions from Protein–Protein Interactions. Genome Res. **12** (2002) 1540–1548
2. Goffeau, A., et al.: Life with 6000 Genes. Science **274** (1996) 546–567
3. Yakushiji, A., et al.: Event Extraction from Biomedical Parsers Using a Full Parser. In: Proc. of the 6th Pacific Symposium on Biocomputing (2001) 408–419
4. Park, J.C., et al.: Bidirectional Incremental Parsing for Automatic Pathway Identification with Combinatory Categorical Grammar. In Proc. of the 6th Pacific Symposium on Biocomputing (2001) 396–407
5. Temkin, J.M., et al.: Extraction of Protein Interaction Information from Unstructured Text Using a Content–free Grammar. Bioinformatics **19(16)** (2003) 2046–2053
6. Leroy, G., et al.: Filling Preposition–based Templates to Capture Information from Medical Abstracts. In: Proc. of the 7th Pacific Symposium on Biocomputing (2002) 350–361
7. Pustejovsky, J., et al.: Robust Relational Parsing Over Biomedical Literature: Extracting Inhibit Relations. In Proc. of the 7th Pacific Symposium on Biocomputing (2002) 362–373
8. Zelenko, D., et al.: Kernel Methods for Relation Extraction. J. Machine Learning Res. **3** (2003) 1083–1106.
9. Lodi, H., et al.: Text Classification Using String Kernels. J. Machine Learning Res. **2** (2002) 419–444.