Extraction of Gene/Protein Interaction from Text Documents with Relation Kernel*

Jae-Hong Eom and Byoung-Tak Zhang

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

Abstract. Even though there are many databases for gene/protein interactions, most such data still exist only in the biomedical literature. They are spread in biomedical literature written in natural languages and they require much effort such as data mining for constructing well-structured data forms. As genomic research advances, knowledge discovery from a large collection of scientific papers is becoming more important for efficient biological and biomedical researches. In this paper, we present a relation kernel based interaction extraction method to resolve this problem. We extract gene/protein interactions of Yeast (*S.cerevisiae*) from text documents with relation kernel. Kernel for relation extraction is constructed with predefined interaction corpus and set of interaction patterns. Proposed relation kernel for interaction extraction only exploits shallow parsed documents. Experimental results show that the proposed kernel method achieves a recall rate of 78.3% and precision rate of 79.9% for gene/protein interaction extraction without full parsing efforts.

1 Introduction

With the advancement of genomic technology and genome–wide analysis of organisms, one of the great challenges to post–genomic biology is to understand how genetic information of proteins results in the predetermined action of gene products, both temporally and spatially, to accomplish biological function and how they act together with each other to build an organism. Also, it is known that gene/protein interactions are fundamental biochemical reactions in the organisms and play an important role since they determine the biological processes [1]. Therefore, comprehensive description and detailed analysis of these interactions would significantly contribute to the understanding of many important biological phenomena and problems.

After the completion of the genome sequence of *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 and many machine learning-based promising methods have been successfully applied to the analysis of these data.

Recently, there also have been many accomplishments in literature data mining for these kinds of biology applications. Many of these applications focus on extracting gene/protein interactions that are scattered throughout the biomedical literature to

* This research was supported by the NRL Program of the Korea Ministry of Science and by the BK21-IT Program from the Ministry of Education and Human Resources Development of Korea. The ICT at Seoul National University provided research facilities for this study

R. Khosla et al. (Eds.): KES 2005, LNAI 3682, pp. 936–942, 2005.

construct well-refined interaction databases of research results. Many research projects have been devised to collect information on gene/protein interactions and many databases have been constructed to store such data. However, most of data in these databases were accumulated manually which need high costs. Nevertheless, scientists continue to publish their discoveries on new gene/protein interactions and modified interactions of previous ones on various domains in scientific papers without submitting their results to specific public databases [3]. As a result, we can infer that most gene/protein interaction information still exists only in these papers.

Even though there are many databases for gene/protein interactions, most such data still exist only in the biomedical literatures. They are spread in biomedical literature written in natural languages and they require much effort such as data mining for constructing well-structured data forms. As genomic research advances, furthermore, knowledge discovery from a large collection of scientific papers is becoming more important for efficient biological and biomedical researches.

Thus, how to extract gene/protein interactions from biomedical literature has been an active research subject. Many approaches widely adopted natural language processing (NLP) techniques to resolve this problem. These methods can be regarded as parsing based methods. Both full and shallow parsing strategies have been attempted.

Yakushiji *et al*. [4] used a general full parser with grammars for biomedical domain to extract interaction events by filling sentences into augmented structures. Park *et al*. [5] proposed bidirectional incremental full parsing with combinatory categorical grammar (CCG) which localizes target verbs and then scans the left and right neighborhood of the verb respectively to find interaction events in the sentences. Temkin *et al*. [6] also utilized a lexical analyzer and context-free grammar (CFG) to extract gene, protein and small molecule interactions with a recall rate of 63.9% and precision rate of 70.2%. As a similar method, preposition-based parsing to generate templates also proposed by Leroy *et al*. [7] and they achieved precision of 70% for biomedical literature abstract processing. For a partial parsing method, Pustejovsky *et al*. [8] used the relational parsing for the inhibition relation with recall rate of 57%. But, these methods are inherently complicated, requiring many resources, and performances are not satisfactory.

In this paper, we present a relation kernel based interaction extraction method to resolve these interaction extraction problems. In the proposed method, we extract gene/protein interactions of *S.cerevisiae* from documents. To doing this, we define kernel for gene/protein relation extraction, so called '*relation kernel*', with predefined interaction corpus and set of interaction patterns. Proposed kernel method for interaction extraction only exploits shallow parsed documents without full parsing efforts.

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

2 Kernel Method

An object can be transformed into a collection of features $f_1,\ldots,f_N$ which produce a N-dimensional feature vectors. In many cases, however, it is difficult to express data via

features. In most NLP problems, for example, feature based representations produce inherently local representations of objects and it is computationally infeasible to generate features involving long-range dependencies [9].

Kernel methods are an attractive alternative to feature-based methods. Kernel methods retain the original representation of objects and use the object in algorithms only via computing a kernel function between a pair of objects. A kernel function is a similarity function which has certain properties. That is, 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 (e.g., genes, proteins, molecular compounds, etc.) into a vector space F, called a *feature space*, and searching for linear relation in such a 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 [9].

It can be shown that any kernel function implicitly calculates the dot product of feature vectors of objects in high-dimensional feature spaces. That is, if there exist features $f(\cdot) = (f_1(\cdot), f_2(\cdot),\ldots)$, then f_i: $X \rightarrow R$, so that $K(x, y) = \langle f(x), f(y) \rangle$. Conversely, given features $f(\cdot) = (f_1(\cdot), f_2(\cdot),\ldots)$, a function defined as a dot product of the corresponding feature vectors is necessarily a kernel function [9].

In many cases, it may be possible to compute the dot product of certain features without enumerating all the features. An excellent example is that of subsequence kernels [10]. 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 [10]. 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 algorithms that process examples only via computing their dot products are sometimes called *dual* 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 very well and avoid overfitting in high dimensional feature spaces. Thus, here we use SVM and relation kernel (extended version of subsequence kernel for shallow parsing) to extract gene/protein interactions.

3 Relation Kernel for Gene/Protein Interaction

Here we use the basic notion and the idea of subsequence kernel and extend it to operate on shallow paring for gene/protein relation extraction according to the work of Lodhi *et al*. [9] and Dmitry *et al*. [10].

We can consider the sentence, "In those pathways, usually the yeast protein RAP1 suppresses the up-regulation of SH3." We can generate appropriate parse tree for this sentence and compare with the other parse trees generated from other sentences by conventional shallow parsing. 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 about the gene/protein interactions such as gene–gene, gene–protein, and protein–protein interactions. 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.

Here we use the term 'example' for parse tree of sentence. Each node of example 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' is for noun phrase or noun which represents a name of gene or protein. Verbs are classified into three categories; positive (e.g., activate), negative (e.g., inhibit), 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 gene/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. The matching function has '1' if two nodes have same type and role in the parsing tree and '0', otherwise. The similarity 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,

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

where, **i** denote a sequence of indices and $d(\mathbf{i}) = l(\mathbf{i}) = i_n – i_1 + 1$ denote length of the sequence **i**. In the Equation 2, the term $P[\mathbf{i}]$ represent the sequence of children $[P[i_1],\ldots,P[i_n]]$ and $K(P_1[\mathbf{i}],P_2[\mathbf{j}])$ stand for summing up $K(P_1[i_s],P_2[j_s])$ for all $s = 1,\ldots,l(\mathbf{i})$ [9].

By using these definitions, Equation 2 consider 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 (**i** and **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 Dmitry *et al*. [9] and its original construction recurrences from Lodhi *et al*. [10] to calculate K_c. That is,

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

where, for the number of children of P_1 and P_2, condition $m \geq n$ is assumed. The boundary conditions of calculation of K_c are as follows:

$$
\begin{aligned}
&K_{c,q}(i,j) = 0, \text{ if } q > \min(i,j), && C_q(i,j) = 0, \text{ if } q > \min(i,j) \\
&C_0(i,j) = 1, && C'_q(i,j) = 0, \text{ if } q > \min(i,j) \\
&C_{q,r}(i,j) = 0, \text{ if } q > \min(i,j) \text{ or } q < r, && C'_{q,r}(i,j) = 0, \text{ if } q > \min(i,j) \text{ or } q < r
\end{aligned}
\tag{4}
$$

4 Experimental Results

Data

We used 260 labeled MEDLINE records which have in any case one sentence containing at least one gene/protein interaction as positive examples (50% were used for training, 50% were used for testing). The negative examples, about 100 examples, were also collected from MEDLINE records which contain any interaction at all. The shallow parses of each record was generated by out custom shallow parsing system. We also used Brill tagger trained with GENIA biomedical corpus to guarantee more accurate POS tagging. Table 1 shows the statistics of these examples.

Table 1. Statistics of positive and negative examples of the corpus which was used for gene/protein interaction extraction task

Categories	# of Positive Relation	# of Negative Relations	Total
# of Relation	314	139	453

Algorithm Settings

For kernel learning, we used Support Vector Machine (SVM) [11] and LIBSVM algorithm implementation [12]. We set parameter of kernel computation to 0.5 to reflect the amount of spread of interaction structure (gene names, interaction verbs,

and protein names). We evaluated kernel based relation extraxtion result in comparison with the general feature based baseline classifier, Naive Bayes. Full parsing results were used as a feature set for Naive Bayes. Here, these baseline model represent the method of full parsing based relation extraction (classification) approach.

Results

For information extraction problems, the system performance is usually reflected using the performance measure of information retrieval: *precision*, *recall*, and *F-measure*. Precision is the ration 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:

$$F = \frac{2 * recall * precision}{(recall + precision)} \tag{5}$$

Table 2 shows the precision, recall, and F-measure for the gene/protein relation extraction experiments.

Table 2. The precision and recall of experiments. *TP* is the number of correctly extracted true interactions and *TN* is the number of correctly extracted negative interactions in the test documents, and we calculated *Precision* = *TP*/(*TP*+*FP*) and *Recall* = *TP*/(*TP*+*FN*). In this table, *K* is stand for kernel method and *N* is stand for Naïve Bayes. In this table, we showed the result for the major four general interaction verbs (The value of 'all verbs*' represent the total average value including the value of four interaction verbs)

Interaction Verb	Recall (%)		Precision (%)		Total *F* (%)	
	K	*N*	*K*	*N*	*K*	*N*
Interact	82.3	79.1	80.4	79.3	81.3	79.2
Bind	80.8	78.5	84.6	78.5	82.7	78.5
Up-regulate	76.2	72.1	82.8	74.2	79.4	73.1
Suppress	79.9	76.9	81.9	81.8	80.9	79.3
All verbs*	**78.3**	74.2	**79.9**	75.5	**79.1**	74.8

As we can see in Table 2, kernel based relation extraction outperform full parsing feature based Naïve Bayes method overall. The number of interaction verb which generally describes gene/protein interactions in the biomedical literature and recognized by this experiment was about 150. These recognized interaction verb include, for example, 'activate', 'abolish', 'accelerate', 'affect', 'alter', 'amplify', 'assemble', 'associate', 'bind', 'block', 'conjugate', 'control', 'down regulate', 'enhance', 'inactive', 'induce', 'infect', 'inhibit', 'interact', 'ligate', 'localize', 'mediate', 'modify', 'prevent', 'prohibit', 'phosphorylate', 'regulate', 'stimulate', 'suppress', 'transactive', 'ubiquitinate', 'upregulate', etc. Among these verbs some verb occurs frequently but other occurs rarely.

5 Conclusions

In this paper, we presented kernel based gene/protein relation extraction method and the result that kernel based method outperform conventional feature based (full pars-

ing information based) approach. Presented relation kernel only requires domain specific knowledge on defining matching and similarity kernel function. In this paper, we used the dictionary of gene/protein names and its aliases. So it is not difficult to expand this method to other domain by changing this simple domain knowledge.

But, this kernel method on NLP problem has one big disadvantage for practical use of algorithm because it requires high computational costs. Thus, the more efficient kernel computation method should be devised for more efficient kernel computation and easily usable functionality to the problems of various domains, and we'll try this as future works.

References

1. Deng, M., *et al*.: Inferring domain–domain interactions from protein–protein inter-actions. *Genome Res*. **12** (2002) 1540–1548.
2. Goffeau, A., *et al*.: Life with 6000 genes. *Science* **274** (1996) 546–567.
3. Huang, M., *et al*.: Discovering patterns to extract protein-protein interactions from full texts. *Bioinformatics* **20(18)** (2004) 3604–3612
4. Yakushiji, A., *et al*.: Event extraction from biomedical parsers using a full parser. In *Proc. of the 6th Pacific Symposium on Biocomputing* (PSB 2001). (2001) 408–419.
5. 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* (PSB 2001). (2001) 396–407.
6. Temkin, J.M., *et al*.: Extraction of protein interaction information from unstructured text using a content-free grammar. *Bioinformatics* **19(16)** (2003) 2046–2053.
7. Leroy, G., *et al*.: Filling preposition-based templates to capture information from medical abstracts. In *Proc. of the 7th Pacific Symposium on Biocomputing* (PSB 2002). (2002) 350–361.
8. Pustejovsky, J., *et al*.: Robust relational parsing over biomedical literature: extracting inhibit relations. In *Proc. of the 7th Pacific Symposium on Biocomputing* (PSB 2002). (2002) 362–373.
9. Dmitry, Z., *et al*.: Kernel methods for relation extraction. *Journal of Machine Learning Res.* **3** (2003) 1083–1106.
10. Lodi, H., *et al*.: Text classification using string kernels. *Journal of Machine Learning Res.* **2** (2002) 419–444.
11. Cortes, C, *et al*.: Support-vector networks. *Machine Learning* **20(3)** (1995) 273–297.
12. LIBSVM, http://www.csie.ntu.edu.tw/~cjlin/libsvm/