Evolutionary Hypernetwork Classifiers for Protein-Protein Interaction Sentence Filtering

Jakramate Bootkrajang
Biointelligence Lab
School of Comp. Sci.& Eng.
Seoul National University
Seoul 151-744, Korea
+82-2-880-1847
jakramate@bi.snu.ac.kr

Sun Kim
Biointelligence Lab
School of Comp. Sci.& Eng.
Seoul National University
Seoul 151-744, Korea
+82-2-880-1847
skim@bi.snu.ac.kr

Byoung-Tak Zhang
Biointelligence Lab
School of Comp. Sci.& Eng.
Seoul National University
Seoul 151-744, Korea
+82-2-880-1847
btzhang@bi.snu.ac.kr

ABSTRACT

Protein-Protein Interaction (PPI) extraction, among ongoing biomedical text mining challenges, is becoming a topic in focus because of its crucial role in providing a starting point to understand biological processes. Machine learning (ML) techniques have been applied to extract the PPI information from biomedical literature. Although they have provided reasonable performance so far, more features are required for real use. In particular, many ML-approaches lack human understandability for learned models. Here, we propose a novel method for classifying PPI sentences. Our approach utilizes the modified hypernetwork model, a hypergraph with weighted hyperedges that are calibrated via an evolutionary learning method. The evolutionary hypernetwork memorizes fragments of training patterns while self-adjusting its own structure for detecting PPI sentences. For experiments, we show that our approach provides competitive performance compared to other ML methods. Apart from its superior classification performance, the evolving hypernetwork model comes with a highly interpretable structure. We show how significant PPI patterns can be naturally extracted from the learned model. We also analyze the discovered patterns.

Categories and Subject Descriptors

I.5.2 [**Computing Methodology, Pattern Recognition, and Design Methodology**]: Classifier design and evaluation

General Terms

Algorithms, Performance, Experimentation

Keywords

Hypernetwork classifier, Evolutionary learning, Protein-protein interaction sentence filtering

GECCO'09, July 8–12, 2009, Montréal, Québec, Canada.

1. INTRODUCTION

Protein-Protein Interaction (PPI) information is critical for understanding the function of individual protein as well as relationships between proteins in biological processes [9]. As a consequence, a number of databases have been built to maintain the rapidly growing PPI information. However, as the information accumulate and databases become large, extracting PPI information manually is no longer an effective method [2], such that the development of more robust alternative is at the highest priority. The earlier works for automatic classification of PPI sentence are primarily based on the use of rule-bases system [1, 12]. However, this approach is very limited and cannot effectively deal with new PPI sentences. The syntactic-aware based on natural language processing (NLP) techniques also have been applied to the PPI task [3, 18, 19], but the techniques usually suffer from classification performance issues. Recently, it has been shown that PPI information has its own patterns at sentence level [8]. The finding encourages scientists to apply machine learning (ML) technique to extract PPI sentences from biomedical documents. ML technique is well known for its robustness in discovering hidden patterns while providing a mechanism for recognizing unknown patterns. For example, SVM classifiers and its variations have been widely used to tackle PPI task [5, 15]. Some researchers used SVMs incorporating with dependency parse trees, which analyzes the path between two proteins [7]. The system called PreBIND has also been built to identify existence of protein interaction using both SVM and naive Bayes classifiers [4]. Despite more satisfactory performance is obtained from these new techniques, one important feature still left unconcerned; i.e., the human understandability of learned model.

It has been claimed that the challenge of information extraction and classification involves with both classification correctness and interpretability of the classifiers [16]. It would be better if we can inspect and know what is going on inside our model. In some sensitive cases, it is more desirable for machine learning algorithms to be able to explain and provide supports for their answer (e.g., classification of medical data). Although many ML techniques applied to PPI extraction problem provide reasonable classification correctness, the learned model is hard to interpret or even not understandable at all. Let us consider support vector machine (SVM) and naive Bayes, none of those algorithms provide human interpretable structure but only mathematical functions or complex network of numbers. This limita-

tion motivates us to search for classifier that also provides human understandable structure.

Unlike other ML techniques, the hypernetwork classifier satisfies both requirements by two reasons. First, the similar properties between hypernetwork structure and structure of natural language sentence ensure its classification performance. That is, text sentences can be viewed as a collection of building blocks [17] in which each block holds related words and contributes to the meaning of the whole sentence at various degrees. Likewise, the hypernetwork model [20] is a collection of hyperedges in which each hyperedge represents the relationship among features (i.e., words). In particular, the hypernetwork classifier works as a committee machine, where each hyperedge contributes to the final decision. Second, the hypernetwork provides human interpretable structure, which is another required feature for text pattern analysis. Thus, the similar properties between hypernetwork structure and word blocks and the interpretability of the hypernetwork form the solid supports for hypernetwork as a PPI classifier. The first guarantees its ability to store semantics of language resides in the sentences while the latter allows humans to extract and interpret information underlying the learned model.

Despite having many advantages, the conventional hypernetwork model suffers from its static characteristic. That is, when one hyperedge is sampled, it remains in the library forever regardless of its importance. This results in an inaccurate prediction due to votes from these unimportant hyperedges. In this paper, we propose an evolutionary approach of the hypernetwork model to overcome this difficulty. The evolutionary learning algorithm is designed to find the best hypernetwork, i.e., the best combination of hyperedges, by weakening the role of irrelevant hyperedges throughout a learning session.

For experiments, we apply the evolutionary hypernetwork classifier to PPI sentence classification. The goal is to accurately identify text sentences in data set whether they are PPI sentences. The experimental results show that the proposed model provides good performance compared to naive Bayes, SVM with various kernels. We also analyze discovered PPI patterns.

The rest of the paper is organized as follows: In section 2, we present the hypernetwork model, the evolutionary learning algorithm and illustrate operations of the classifier. Experimental results are reported in section 3, including data preprocessing, parameters and discussion of the results. Section 4 draws conclusions from this study.

2. HYPERNETWORK MODEL

A hypernetwork is a probabilistic graphical model based on hypergraph [20]. A hypergraph is an undirected graph G whose edges connect a non-null number of vertices. The hypergraph is defined as $H = \{V, E\}$ where V is a set of vertices such that $V = \{v_1, v_2, ..., v_n\}$ and E is a set of hyperedges such that $E = \{e_1, e_2, ..., e_m\}$. A hyperedge is an edge in the hypergraph in which more than two vertices can be connected, hence the name *hyper*. Each hyperedge is expressed as $e_i = \{v_{i1}, v_{i2}, ..., v_{ik}\}$ where k indicates a cardinality of the hyperedge. A hyperedge of cardinality k will be referred to as k-hyperedge. The advantage of using hyperedges is that it offers additional degree of freedom in representing the network while preserving mathematical method provided by the graph theory.

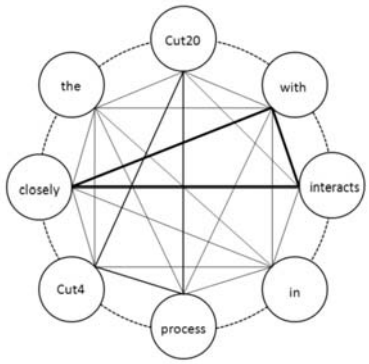

Figure 1: An example of hypernetwork model constructed from "Cut20 interacts closely with Cut4 in the assembly process of cyclosome".

The hypernetwork is considered as a generalization of the hypergraph such that it assigns weight values to its hyperedges to express how strong vertices are connected. Formally, the hypernetwork model can be defined as a triple $H = \{X, E, W\}$, where X is set of all words in data set, $E = \{e_1, e_2, ..., e_n\}$ is a set of hyperedges. A hyperedge $e_i = \{x_{i1}, x_{i2}, ..., x_{im}, y_i\}$ is a set containing m words from word set X and a class label y_i, and $W = \{w_1, w_2, ..., w_n\}$ is the set of weight value for each hyperedge. A hypernetwork H that consists of only k-hyperedges is said to be k-uniform and may be referred to as k-hypernetwork. One advantage of hypernetwork is that the higher-order correlation terms can be explicitly represented by hyperedges. That is an attractive feature in machine learning paradigm, especially in text classification domain. An instance of 3-hypernetwork constructed from example sentence from corpus is illustrated in Figure 1. Here, three words are randomly selected to form a hyperedge. The links, which connects vertexes in the same hyperedge, indicate the relationship between words whereas degree of correlation between words is represented by their thickness. More details about hypernetwork models can be found in [20, 21, 22].

2.1 Evolutionary hypernetwork classifiers

The hypernetwork classifier relies heavily on two processes, constructing initial hypernetwork and learning via an evolutionary method to archive an optimal hypernetwork. In evolutionary computation framework, a hypernetwork classifier can be considered as a population, while its hyperedges representing individuals. The crux of evolutionary learning method is that all individuals will go through the simplified selection and replacement processes with equal probability. The selection is based on matching a hyperedge sampled from training example to the hyperedges (individuals) in the library (population). The matched hyperedges are said to be selected. The replacement is done by adjusting weight of selected individual towards maximizing classification accuracy. By calibrating weights, the contribution of each individual to final output can be manipulated. In contrary to conventional replacement operation where replaced individuals vanished, our replacement algorithm does not completely eliminate individual from population set but rather weaken its existence. When a number of epochs have passed,

some redundant individuals could be thought as being disappear from population set (i.e., their contributions are so small comparing to others). In effect, this variation of replacement method works very similar to the conventional scheme.

We are now giving the method of constructing initial hypernetwork. Basically, hypernetwork model can be viewed as a probabilistic memory to store a data set $D = \{x^{(n)}\}_{n=1}^{N}$, where $x^{(n)}$ denotes the n-th pattern to store using its hyperedges and their weight values. By taking into account that hypernetwork is a multi-set of hypergraph, the number of all possible hyperedges for k-hypergraph is,

$$|E| = \sum_{k=0}^{n} C(n,k) = \sum_{k=0}^{n} \frac{n!}{k!(n-k)!} = 2^n, \tag{1}$$

which result in the total searching space of

$$|\Omega| = 2^{\kappa \cdot 2^n}, \tag{2}$$

where κ is the number of duplication of each hyperedges. In the situation that n and κ grows very large, i.e., in text classification where n represents the number of all words appeared in the document, we will end up with a very huge problem space Ω.

In order to effectively search this problem space, we use a stochastic approach based on random graph theory to generate a random hypernetwork. A random graph process chooses a single graph at random with equal probability from a set of all possible graphs.

The probability space can be view as a product of $C(n,k)$ binary space. The random hypernetwork can be generated by binomial random graph process. Given a real number $p, 0 \leq p \leq 1$, the binomial random hypergraph $\mathcal{G}(n,p)$ is defined by taking as Ω the set of all hypergraphs on vertex set $[n]$ and setting

$$P(\mathcal{G}) = p^{|E(\mathcal{G})|}(1-p)^{C(n,k)-|E(\mathcal{G})|}, \tag{3}$$

where $|E(\mathcal{G})|$ stands for the number of edges of $\mathcal{G}$. Repeating random hypergraph process generates the random hypernetwork. The difference of random hypernetwork from original random graph process is that we insert the hyperedges that instantiated from training data set into hypernetwork to construct the model of the problem being solved.

After that, we allow the random hypernetwork to learn. As the training sample $(\mathbf{x}, y)$ is observed, the hypernetwork has to determine the class y^* of the example $\mathbf{x}$. The prediction procedure is formally described as follows. First, we define the energy of the hypernetwork as

$$\mathcal{E}(\mathbf{x}^{(n)}; W) = \sum_{i=1}^{|E|} w_{i_1 i_2 \ldots i_{|E_i|}} \mathbf{x}_{i_1}^{(n)} \mathbf{x}_{i_2}^{(n)} \ldots \mathbf{x}_{i_{|E_i|}}^{(n)} \tag{4}$$

where W represents the parameters (hyperedge weight) for the hypernetwork model. The probability governing data being generated from hypernetwork is given as Gibbs' distribution

$$P(\mathbf{x}^{(n)}|w) = \frac{1}{Z(W)} \exp\{-\mathcal{E}(\mathbf{x}^{(n)}; W)\}, \tag{5}$$

where $\exp\{-\mathcal{E}(\mathbf{x}^{(n)}; W)\}$ is the Boltzmann factor and $Z(W)$ is the normalizing term. In order to classify a data $(\mathbf{x}, y) \in D$ that consists of set of features x_i and class label y, we use the fact that the hypernetwork can be represented by adding a vertex y to the set of vertices X and formulate a joint probability $P(X, y)$ as

$$P(\mathbf{x}, y) = \frac{1}{Z(W)} \exp\{-\mathcal{E}(\mathbf{x}, y; W)\}. \tag{6}$$

Thus, given training input $\mathbf{x}$, the classifier calculates the probability of each class y and output as its prediction y^* (i.e., the class with highest conditional probability).

$$y^* = \arg\max_y \frac{P(\mathbf{x}, y)}{P(\mathbf{x})} = \arg\max_y P(\mathbf{x}, y). \tag{7}$$

Generally, the learning method is simply a classification trial on training data set where hypernetwork makes a prediction and adjusts weight values using the evolutionary learning algorithm. It begins by generating new hyperedges from training examples, then searching and matching newly generated hyperedges to existing hyperedges in the library (selection step). Each time the match occurs, it collects weight values of matched hyperedges and makes a prediction based on accumulated weight values. In our method, two hyperedges are said to match if either all the elements are matched or at most one mismatch occurred, preserving order. During learning process, weight values of hyperedges should be adjusted (replacement step) to fine-tune the model when an incorrect prediction is made.

1. Construction step.
   Generate hyperedge e_i from sentences in training data set by random hypergraph process.
   Begin with $H = \{X, E, W\} = \{\emptyset, \emptyset, \emptyset\}$
   a. $E \leftarrow E \cup \{e_i\}$
   b. $X \leftarrow X \cup \{x_i \mid x_i \in e_i\}$
   c. $W \leftarrow W \cup \{w_i \mid w_i \in w_{init}\}$
   d. Repeat until $|E|$ reaches predefined size m
2. Prediction step.
   Generate a hyperedge set T from sentences in validation data set by random hypergraph process.
   Let y and $y*$ be the correct class and predicted class, respectively and let ans_0 be the weight sum of class 0 and ans_1 be the weight sum of class 1
   a. Compare $t_i \in T$ to every $e_i \in E$, allowing at most one word mismatch.
   b. If t_i matches e_i, then $ans_y \leftarrow ans_y + w_i$
   c. Repeat step b. and c. for $t_i \in T$

   $y* = 1$ if $ans_1 > ans_0$ else $y* = 0$
3. Learning step.
   Let M_0 and M_1 be a set of matched edges of class 0 and class 1 in previous step.
   a. If $y* \neq y$, $w_i \leftarrow w_i + learningParam$ for all $e_i \in M_y$
4. Prediction and Learning steps are repeated until terminate condition is met.

Figure 2: Learning algorithms for hypernetwork classifiers.

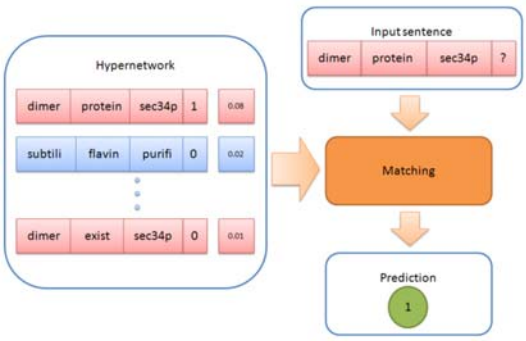

Figure 3: The classification mechanism of the hypernetwork.

Validating the learned model is as the same as training it but sentences from unseen data set are used. The procedure to construct and train the evolutionary hypernetwork classifier for a two class problem is summarized in Figure 2.

In effect, the evolutionary learning algorithm performs gradient search to find an optimal hypernetwork for training data. Assuming that we use the sigmoid function $\mathcal{E}(\mathbf{x}^{(n)}; W)$ as the energy function we have

$$\mathcal{E}(\mathbf{x}; W) = \frac{1}{1 + e^{-f(\mathbf{x}, W)}}, \tag{8}$$

where

$$f(\mathbf{x}, W) = \sum_{i=1}^{|E|} w_{i_1 i_2 \ldots i_{|E_i|}} \mathbf{x}_{i_1} \mathbf{x}_{i_2} \ldots \mathbf{x}_{i_{|E_i|}}. \tag{9}$$

The error function associate to this energy function can be stated as

$$\begin{aligned} G(W) &= -\sum_{n=1}^{N} (y^{(n)} \ln \mathcal{E}(\mathbf{x}^{(n)}; W) + \\ &\quad (1 - y^{(n)}) \ln(1 - \mathcal{E}(\mathbf{x}^{(n)}; W))). \end{aligned} \tag{10}$$

Notice that the term $\mathbf{x}_{i_1} \mathbf{x}_{i_2} \ldots \mathbf{x}_{i_{|E_i|}}$ in Eqn. 9 is simply a combination of k elements of the data $\mathbf{x}$ which is represented as k-hyperedges in the network. Here, if we take the partial derivative of function G in Eqn. 10, we will have

$$\frac{\partial G}{\partial w_i} = \sum_{n=1}^{N} -(y^{(n)} - y^{*(n)}) x^{(n)}. \tag{11}$$

Notice that $(y^{(n)} - y^{*(n)})$ is the error on an input instant. W needs to be adjusted in the direction that minimizes this error. From Eqn. 11, we successfully show that algorithm in learning step of Figure 2. is a simplified version of the gradient search. However, since the hypernetwork encompasses two basic evolutionary operators (i.e., selection and replacement), it can be regarded as evolutionary method as well. Figure 3 illustrates the classification procedure of the hypernetwork. The hyperedge {*dimer, protein*} is being compared to all hyperedges in hypernetwork. It matched with one hyperedge that was instantiated from PPI sentence (class 1), i.e., {*dimer, protein*} which has the weight value of 0.08 and hyperedge that was instantiated from non-PPI sentence {*dimer, substrat*} which has the weight value of 0.01. Without having matched to any other hyperedges, the hypernetwork model predicts that current example belongs to class 1 since total weight summation of class 1 is higher.

3. RESULTS AND DISCUSSION

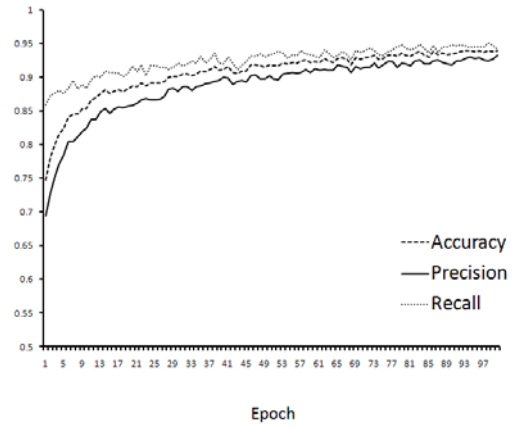

Figure 4: Learning graphs of the hypernetwork model.

The proposed method was applied to a PPI sentence corpus, which is the combination of the BioCreAtIvE II workshop data set, Anne Lise Veuthey corpus, Prodisen interaction corpus and manually selected PPI sentences [15]. Preprocessing is performed to shrink the problem space without affecting classification performance by deleting redundant components and stemming words. Insignificant words, i.e., stop words were also removed from the sentences.

We use a data set (X, y) where $X = (x_1, x_2, ..., x_n) \in \mathbb{N}$ and $y \in \{0, 1\}$. The elements in set X represent words appeared in the sentence in numerical form while y represents its class and that $y = 1$ indicates PPI sentences. Although hypernetworks can store any type of feature values, the numerical data increase efficiency in searching and matching, which those are intensively performed in learning process. Hence, we convert string values into numerical values using a customized dictionary.

We have trained and evaluated 3-uniform hypernetwork to classify the PPI sentences. The hyperedges were generated using random hypernetwork process, previously introduced. The size of hyperedge set was set to 1,200,000 and the number of duplicate edges was not limited. The learning parameter was set to 0.01 throughout the experiment, which lasted for 100 epochs.

Evaluation measures

The common performance measures, accuracy, precision, and recall are calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \times 100, \tag{12}$$

$$Precision = \frac{TP}{TP + FP} \times 100, \tag{13}$$

$$Recall = \frac{TP}{TP + FN} \times 100, \tag{14}$$

Table 1: Precision at 10-th, 20-th and 30-th rank sentences on Biocreative I (BC), Cristine Brun (CB) and N-PPI.

	BC			CB			N-PPI		
	10	20	30	10	20	30	10	20	30
Hypernetwork	90.00	75.00	76.67	100.00	85.00	76.67	90.00	80.00	66.67
Tree	90.00	85.00	76.67	100.00	90.00	90.00	80.00	75.00	66.67
Naive Bayes	70.00	75.00	70.00	100.00	100.00	93.33	90.00	75.00	66.67
RBF	70.00	75.00	76.67	100.00	95.00	90.00	70.00	65.00	76.67

Table 2: $Precision_{interp}$ at 10% recall point.

	BC	CB	N-PPI
Hypernetwork	90.91	91.67	100.00
Tree	86.98	84.68	77.27
Naive Bayes	73.68	80.20	90.91
RBF	81.48	78.86	82.93

where TP is the number of correct positive predictions, FP is the number of incorrect positive predictions, FN is the number of incorrect negative predictions, and TN is the number of correct negative predictions.

However, providing performance evaluation based on binary prediction only is not enough for real use. For example, in Web applications, the performance on top ranked results are more important than that on whole data set. This property affects text classification tasks in the same way. Hence, in this paper, interpolated precision and precision at Nth sentences are used to show system performance, i.e.,

$$Precision_{interp}(r) = \max_{r' \geq r} Precision(r'), \quad (15)$$

$$Precision_N = \text{Precision at } N\text{th rank}, \quad (16)$$

where $Precision_{interp}(r)$ is an interpolated precision at a certain recall r, and $Precision_N$ is a precision at Nth sentences. $Precision_{interp}$ is a measure of the number of PPI sentences before a certain percentage of relevant PPI sentences have been identified. Higher $Precision_{interp}$ indicates that true PPI sentences are located more in the high ranks. In addition, it is another good performance measure for text retrieval system because there is always a trade-off between precision and recall. $Precision_N$ account for the quality of ranking by using top N sentences.

Rank is judged based on confidence value of each prediction. The value can be calculated using the following steps:

1. From prediction step in Figure 2,
   let $|M| = |M_0| + |M_1|$,
   let ans_0 and ans_1 be the weight sum of class 0 and class 1, respectively and
   let $conf$ be the confidence value of this prediction.

2. $conf$ = (ans_1 - ans_0) / $|M|$.

3.1 PPI sentence classification

We now present the performance of the proposed model using 10-fold cross validation. Figure 4 illustrates the learning performance curve on training examples. After 100 iterations, the learning is converged and our model achieves 93% average accuracy while precision and recall rates are about 94%.

Table 3: Top 10 hyperedges based on weight values.

Rank	Vertex 1	Vertex2	Vertex 3
1	purifi	complex	enhanc
1	protein	form	specif
1	purifi	complex	atpas
2	character	protein	complex
2	protein	dimer	protein
3	yeast	sec34p	sec35p
3	protein	ubiquitin	specif
3	transamidas	complex	gaa1
3	subunit	exist	contain
3	ar	function	ubiquitin

We also performed the classification task on unseen data set and compared the results to the output from conventional ML algorithms. Table 1 reports precision rates at top Nth rank sentences for Biocreative I corpus (BC) [13], Christine Brun corpus (CB) [10] and Negative Cases of Protein-Protein interaction corpus (N-PPI) [14], respectively. The hypernetwork classifier outputs acceptable results compared to SVM with tree kernel, naive Bayes, and SVM with RBF kernels. The proposed method ranks first on N-PPI corpus and almost tops the performance on BC. We also find that the results based on interpolated precision at 10% recall point is very good. It outperforms every other ML techniques. The results are summarized in Table 2.

The results suggest that hypernetwork exhibits both consistency and stability in performance for all test corpora. Therefore, we can conclude that the proposed model is effective for sentence filtering tasks with high correctness.In next sub-section, we will look further into the structure of hypernetwork and demonstrate how the model can be interpreted.

3.2 PPI pattern analysis

Inspired by biological mechanism, the hypernetwork model naturally provides interpretable structure of the learned model such that analysis of the significant word patterns can be carried out easily. As described previously that the hypernetwork learning method is a simplified version of gradient search in which each weight values of hyperedges are adjusted to acquire optimal model and that the weight values are tuned towards detecting PPI sentences. As a result, after the learning process, hyperedges that carry high weight values are likely to play an essential part in determining whether a sentence is PPI. To prove this hypothesis, we sorted the hyperedges of hypernetwork by their weight values in descending order. The top 10 hyperedges are shown in Table 3. Note that, the words presented here are still in a stemmed form and are all in lowercase.

Table 4: High frequency terms from top 100 hyperedges.

Rank	Term	Rank	Term
1	protein	6	contain
2	complex	7	corepressor
3	subunit	8	form
4	specify	9	ar
5	activate	10	thi

The results reveal that a majority of the words appear in top ranked hyperedges are related to PPI information. For example, *Atpas* (stemmed from *ATPase*) is an enzyme whose central function is catalyzing the composition of *adenosine triphosphate (ATP)* into *adenosine diphosphate (ADP)*. *ATPase* is regarded as one type of protein. *Transamidas* (stemmed from *transamidase*) is an enzyme whose role is to catalyze the transfer of an amide group from one molecule to another. A term *ubiquitin*, a small protein found in eukaryotic cells, is also witnessed within the high-ranked hyperedges. Notice that not only terms that refer to protein are explicitly found in those hyperedges but also the terms that indicate particular interaction between them, such as *contain*, *subunit* or *enhanc (enhance)* are playing a crucial role in the problem context. The manifestation of words such as *exist* or *character*, which seems insignificant and irrelevant to our goal, might result from the allowance of a minor degree of freedom, i.e., a single feature mismatch during matching process.

The learned model was further analyzed by retrieving the most frequent words that appeared on top of our library, say top 100 hyperedges. Those words are presented in Table 4. The word *protein* was spot most of the time and that because it is the word representing our topic of concern. Others interesting words appeared here also represent the relationship between proteins. For example, the word *specify* clearly indicates the interaction of proteins. The word *activ (activate)* also gives the sense of interaction or *subunit* presents the interaction as well. Strictly speaking, the hypernetwork classifier primarily uses these keywords when deciding whether the sentence is PPI. These significant keywords that our classifier detected and stored in its library were also claimed by [11] to be the important keywords in PPI sentence classification domain. They found that the word *protein* and *complex* are among the top frequency words found by their approach. Interestingly, their results coincide with part of our outcomes. From this fact, we can loosely claim that the hypernetwork is an appropriate model for PPI sentence classification just like any other approach. In some cases, the hypernetwork even outperformed those widely introduced and more sophisticated classification methods.

A highly interpretable structure of hypernetwork also benefits us when we would like the get some clues why the prediction has been made in such way. As previously described, to classify a new example, we generate random hyperedges from traning example and match them with hyperedges reside in the library. In this case, the clues we are interested in are just within those matched hyperedges. For illustration purpose, let us examine how hypernetwork manages to classify the sentence "A ORF50 does not stimulate the phosphorylation of STAT3" from the N-PPI test corpus which was initially labeled as PPI sentence.

Table 5: Top 10 hyperedges based on weight values.

Rank	Vertex 1	Vertex2	Vertex 3
1	stimul	phosphoryl	camp
2	stimul	phosphoryl	affect
2	activ	stimul	phosphoryl
2	stimul	phosphoryl	protein
2	stimul	phosphoryl	kei
2	stimul	phosphoryl	matur
2	factor	stimul	phosphoryl
2	did	stimul	phosphoryl
2	icmt	stimul	phosphoryl
2	stimul	phosphoryl	erk1

We populated matched hyperedges whose contribution to the whole answer are dominant (Table 5). It can readily be seen from the result that a significant term is the term that appear more frequent, which are *stimul (stimulate)* and *phosphoryl (phosphorylation)*. The result indicates that the contribution to final decision of hypernetwork are mostly come from these two terms. Indeed, the term phosphorylation, which refers to the process of adding phosphate into proteins and thus refer to the interaction between proteins, plays a key role here and intuitively justify the prediction. It is also very interesting that, although there are two protein names in the sentence (e.g., ORF50 and STAT3), they do not play such important role during classification. By not depending only on protein names but also focus on discovering regularities in pattern and associated terms, hypernetwork can deal with the unseen sentences more accurate. This scheme used by the classifier should be regarded as another hallmark of the hypernetwork. It is somehow worth mentioning how human might deal with the problem of identifying PPI sentences. By common sense, humans might look for protein names first and see whether those proteins interact with each other in a certain ways. Likewise, many PPI classification algorithms also use dictionary of protein names as a technique for PPI sentence classification [6], which is ineffective due to its static characteristic of knowledge domain. It should be clear from previous example to see how hypernetwork justifies its answer. By all means, the experimental results, the discussion and the classification example should convey the idea of how evolutionary hypernetwork classifier might has the advantage over other techniques.

4. CONCLUSIONS

We proposed an evolutionary classifier for text filtering and information extraction in biomedical domain. The model is based on the hypernetwork, a generalization of hypergraph model. Here, word fragments in training data are randomly sampled to construct a random hypernetwork using a random graph process. An evolutionary approach is designed to find the best hypernetworks without exhaustive search in limited resources.

The proposed method was applied to protein-protein interaction corpora. Enhanced by the evolutionary learning algorithm, the hypernetwork has been proved to be able to effectively search large problem space. The experimental results show that the evolutionary hypernetwork classifier exhibits competitive performance to other machine learning algorithms. These empirical evidences support the fact that hypernetworks see relationships, capture associations

and discover regularities in patterns. Apart from classification performance, the hypernetwork model comes with an attractive feature that allows us to make an analysis of the learned model. Thus, the hypernetwork can be regarded as an effective classifier that provides both accurate classification and interpretable structure.

One major research focus in the future is to study the effect of preprocessing techniques. We believe that through a more sophisticated feature selection procedure we can achieve higher overall performance. We expect that this future will provide an alternative method for text sentence classification in both biomedical and related domains.

5. ACKNOWLEDGMENTS

This work was supported in part by IITA through the MARS project (IITA-2009-A1100-0901-1639), by KOSEF through the National Research Laboratory Program (No. M10400000349-06J0000-34910), by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (KRF-2008-314-D00377) and by Ministry of Education and Human Resources Development under the BK21-IT Program. The ICT at Seoul National University provided research facilities.

6. REFERENCES

[1] C. Blaschke, M. A. Andrade, C. Ouzounis, and A. Valencia. Automatic extraction of biological information from scientific text: protein-protein interactions. *Proc Int Conf Intell Syst Mol Biol*, pages 60–67, 1999.

[2] A. M. Cohen and W. R. Hersh. A survey of current work in biomedical text mining. *Brief Bioinform*, 6(1):57–71, March 2005.

[3] N. Daraselia, A. Yuryev, S. Egorov, S. Novichkova, A. Nikitin, and I. Mazo. Extracting human protein interactions from MEDLINE using a full-sentence parser. *Bioinformatics*, 20(5):604–611, March 2004.

[4] I. Donaldson, J. Martin, B. de Bruijn, C. Wolting, V. Lay, B. Tuekam, S. Zhang, B. Baskin, G. D. Bader, K. Michalickova, T. Pawson, and C. W. Hogue. Prebind and textomy–mining the biomedical literature for protein-protein interactions using a support vector machine. *BMC Bioinformatics*, 4, March 2003.

[5] J.-H. Eom, S. Kim, S.-H. Kim, and B.-T. Zhang. A tree kernel-based method for protein-protein interaction mining from biomedical literature. *Lecture Notes in Computer Science*, 3886:42–52, 2006.

[6] J.-H. Eom and B.-T. Zhang. Mining protein interaction from biomedical literature with relation kernel method. *Lecture Notes in Computer Science*, 3973:642–647, 2006.

[7] G. Erkan, A. Ozgur, and D. R. Radev. Semi-supervised classification for extracting protein interaction sentences using dependency parsing. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 228–237, 2007.

[8] H. Jang, J. Lim, J. H. Lim, S. J. Park, K. C. Lee, and S. H. Park. Finding the evidence for protein-protein interactions from PubMed abstracts. *Bioinformatics*, 22(14), July 2006.

[9] S. Kim, S.-Y. Shin, I.-H. Lee, S.-J. Kim, R. Sriram, and B.-T. Zhang. PIE: an online prediction system for protein-protein interactions from text. *Nucl. Acids Res.*, 36:W411–W415, May 2008.

[10] M. Krallinger, F. Leitner, and A. Valencia. Assessment of the second biocreative PPI task: automatic extraction of protein-protein interactions. In *Proceedings of the 2nd BioCreAtIvE Workshop*, pages 41–54, 2007.

[11] E. M. Marcotte, I. Xenarios, and D. Eisenberg. Mining literature for protein-protein interactions. *Bioinformatics*, 17(4):359–363, April 2001.

[12] T. Ono, H. Hishigaki, A. Tanigami, and T. Takagi. Automated extraction of information on protein-protein interactions from the biological literature. *Bioinformatics*, 17(2):155–161, February 2001.

[13] C. Plake, J. Hakenberg, and U. Leser. Optimizing syntax patterns for discovering protein-protein interactions. In *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*, pages 195–201, New York, NY, USA, 2005. ACM.

[14] O. Sanchez-Graillet and M. Poesio. Negation of protein protein interactions: analysis and extraction. *Bioinformatics*, 23(13):i424–i432, July 2007.

[15] S.-Y. Shin, S. Kim, J.-H. Eom, B.-T. Zhang, and R. Sriram. Identifying protein-protein interaction sentences using boosting and kernel methods. In *Proceedings of the 2nd BioCreAtIvE Workshop*, pages 187–192, 2007.

[16] I. Sondhauss and C. Weihs. Business phase classification and prediction: How to compare interpretability of classification methods? *Sonderforschungsbereich*, 475, 2004.

[17] R. Swanson and A. S. Gordon. A comparison of alternative parse tree paths for labeling semantic roles. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 811–818, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[18] J. M. Temkin and M. R. Gilder. Extraction of protein interaction information from unstructured text using a context-free grammar. *Bioinformatics*, 19(16):2046–2053, November 2003.

[19] A. Yakushiji, Y. Miyao, Y. Tateisi, and J. Tsujii. Biomedical information extraction with predicate-argument structure patterns. In *SMBM*, 2005.

[20] B.-T. Zhang. Random hypergraph models of learning and memory in biomolecular networks: shorter-term adaptability vs. longer-term persistency. *The First IEEE Symposium on Foundations of Computational Intelligence*, pages 344–349, 2007.

[21] B.-T. Zhang. Hypernetworks: A molecular evolutionary architecture for cognitive learning and memory. *Computational Intelligence Magazine, IEEE*, 3(3):49–63, August 2008.

[22] B.-T. Zhang and J.-K. Kim. DNA hypernetworks for information storage and retrieval. In *Lecture Notes in Computer Science, DNA12*, pages 298–307, 2006.