BIOINFORMATICS ORIGINAL PAPER

Vol. 21 no. 15 2005, pages 3294–3300
doi:10.1093/bioinformatics/bti493

Data and text mining

Discovering patterns to extract protein–protein interactions from the literature: Part II

Yu Hao[1], Xiaoyan Zhu[1,*], Minlie Huang[1] and Ming Li[1,2,3]

[1]State Key Laboratory of Intelligent Technology and Systems (LITS), Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, China, [2]School of Computer Science, University of Waterloo, N2L 3G1, Canada and [3]City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong SAR

Received on February 21, 2005; revised and accepted on May 6, 2005
Advance Access publication May 12, 2005

ABSTRACT

Motivation: An enormous number of protein–protein interaction relationships are buried in millions of research articles published over the years, and the number is growing. Rediscovering them automatically is a challenging bioinformatics task. Solutions to this problem also reach far beyond bioinformatics.

Results: We study a new approach that involves automatically discovering English expression patterns, optimizing them and using them to extract protein–protein interactions. In a sister paper, we described how to generate English expression patterns related to protein–protein interactions, and this approach alone has already achieved precision and recall rates significantly higher than those of other automatic systems. This paper continues to present our theory, focusing on how to improve the patterns. A minimum description length (MDL)-based pattern-optimization algorithm is designed to reduce and merge patterns. This has significantly increased generalization power, and hence the recall and precision rates, as confirmed by our experiments.

Availability: http://spies.cs.tsinghua.edu.cn

Contact: zxy-dcs@tsinghua.edu.cn

1 INTRODUCTION

We aim at systematically developing a novel and effective methodology for automatically extracting protein–protein interaction information from the literature, including over 12 million articles at Medline. Our proposal is to automatically discover relevant English expression patterns, optimize them and use them to find protein–protein interaction information in research articles. The rationale is very simple. General literature mining at the semantic level is not technically feasible. For simpler problems in a restricted domain, such as the protein–protein interaction, our theory works just fine. This has already been demonstrated in our first paper (Huang *et al*., 2004), where we implemented the first part of our theory. We used dynamic programming to extract sentence patterns related to protein–protein interaction and, using these patterns, our system has achieved high precision and recall rates that have never been achieved by any other fully automatic system. The current paper studies and implements the second part of our theory: improving the patterns to achieve higher sensitivity and specificity. This is done by a novel application of the minimum description length (MDL) principle. Our approach is wholly data driven, and the MDL principle ensures that the resulting patterns have better generalization abilities. Experiments show that the number of patterns is greatly reduced by our algorithm, and the system's performance is significantly improved.

Databases such as BIND (the Biomolecular Interaction Network Database) (Bader *et al*., 2001) and PIR (the Database of Interacting Proteins) (Salwinski *et al*., 2004) are useful. However, there is a large volume of experimental data pertaining to protein, gene and small molecule interactions scattered in enormous volumes of the published literature in natural languages. Automatically rediscovering such information is invaluable, e.g. for protein pathway studies (Hirschman *et al*., 2002). Such an automatic system will also serve as a prototype for similar problems in other domains, say, on the Internet.

Many previous works exist (Ray and Craven, 2001). Natural language processing (NLP) techniques have been widely applied. These are parsing-based methods, with full and partial (or shallow) parsing strategies. A general full parser with grammars applied to the biomedical domain was used to extract interaction events by filling sentences into argument structures by Yakushiji *et al*. (2001). No recall or precision rate was given. Another full parsing method, using bidirectional incremental parsing with combinatory categorical grammar (CCG), was proposed (Park *et al*., 2001). This method first localizes the target verbs and then scans the left and right neighborhood of the verb. The recall and precision rates of the system were reported to be 48% and 80%, respectively. Another full parser, utilizing a lexical analyzer and context free grammar (CFG), extracts protein, gene and small molecule interactions with a recall rate of 63.9% and a precision rate of 70.2% (Temkin and Gilder, 2003) Similar methods such as preposition-based parsing to generate templates were proposed by Leroy and Chen (2002), processing only abstracts with a template precision of 70%. A partial parsing example is relational parsing for the inhibition relation (Pustejovsky *et al*., 2002), with a comparatively low recall rate of 57%. All these methods are inherently complicated and domain sensitive, requiring many resources and showing poor performances; some focus only on several special verbs.

Another approach uses pattern matching. As an example, a set of simple word patterns and part-of-speech rules were manually coded, for each verb, to extract a special kind of interactions from abstracts (Ono *et al*., 2001). This method is essentially a rule-based method, without any complicated parsing techniques; thus it is able to handle

*To whom correspondence should be addressed.

© The Author 2005. Published by Oxford University Press. All rights reserved. For Permissions, please email: journals.permissions@oupjournals.org

Downloaded from https://academic.oup.com/bioinformatics/article/21/15/3294/195397 by guest on 12 March 2021

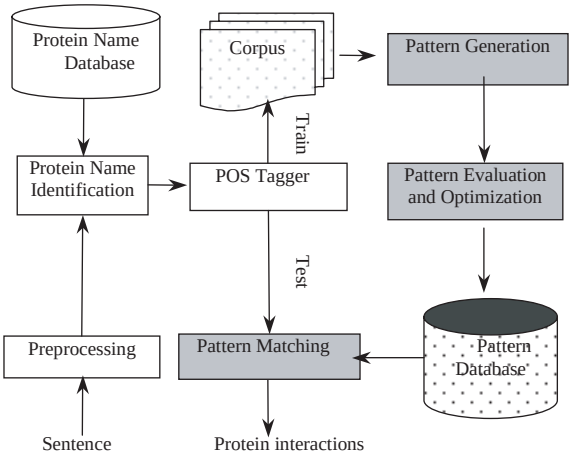

Fig. 1. System architecture.

long sentences, outperforming the traditional parsing methods. The method obtains a recall rate of ∼85% and a precision rate of ∼94% for yeast and *Escherichia coli*.[1] In GENIES, more complicated patterns with syntactic and semantic constraints are used (Friedman *et al*., 2001). GENIES also uses semantic information. GENIES' recall rate is low. In all the above methods, including our work (Yao *et al*., 2004) and several commercial systems, patterns are hand-coded without exception. Such systems are not flexible, not easily improvable, and hence have limited practicality (see also Ng and Wong, 1999; Thomas *et al*., 2000; Wong, 2001).

This paper is arranged as follows: in Section 2, the structure of our system is introduced; the MDL-based optimization algorithm is described in Section 3; four experiments which test the effectiveness of our algorithm are presented in Section 4; and Section 5 contains the discussion.

2 SYSTEM OVERVIEW

Our protein–protein interaction extraction system is divided into three phases (Figure 1): the data preparation phase, the pattern generation phase and the interaction extraction phase.

The data preparation phase first converts the input sentence into tagged sequences for pattern generation and interaction extraction. This phase consists of preprocessing, protein name identification and the part-of-speech (POS) tagger. For an input sentence, the preprocessing module first uses some filtering rules to remove useless expressions. Then protein names in the sentence are identified according to a protein name dictionary and the names are replaced with a unique label in the protein name identification module. Subsequently, the sentence is part-of-speech tagged by Brill's tagger (Brill *et al*., 1995). Last, the tag sequence is generated and added into the corpus for the pattern generation phase or used by the matching algorithm for interaction extraction.

The pattern generation phase mines the tagged sentences in the corpus and extracts patterns using a dynamic programming algorithm.

[1]The precision and recall rates in this paper and in (Huang *et al*., 2004) were calculated for individual verbs. This is different from our method, which measures all verbs at the same time using the MDL principle.

Table 1. Main tags used in the patterns

Tag name	Tag description
PTN	Special tag for protein name
NN	Noun, singular or mass
NNS	Noun, plural
IN	Preposition, subordinating conjunction
CC	Coordinating conjunction
TO	To
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-third-person singular present
VBZ	Verb, third-person singular present
RB	Adverb
JJ	Adjective

Patterns are also tag sequences, where each tag is called a component. In our patterns, the tag alphabet consists of two kinds of tags: part-of-speech tags, like those used by Brill's tagger (Brill *et al*., 1995), and the tag *PTN* for protein names. The main tags are listed in Table 1.

Except for *PTN*, each tag has a word set that contains the words with which the tag can be instantiated. For example, for a pattern {PTN VBZ IN PTN: *; binds, associates; to, with;*}, the word set of the tag *VBZ* is {binds, associates}, and that of the tag *IN* is {to, with}, as shown by Huang *et al*. (2004). The acquired patterns are then evaluated and optimized by an MDL-based algorithm to be presented here. The resulting patterns are stored in the Pattern Database to be used in the interaction extraction phase, which extracts interactions by dynamically matching the patterns with sentence tag sequences.

3 METHOD

Smallest consistent theory has the highest power to explain and generalize the data. Consider pattern set $P = \{p_1, p_2, \ldots, p_m\}$, which consists of candidate English expression patterns p_i that are extracted from the literature automatically, as in Huang *et al*. (2004), or manually, as in other systems. There is no guarantee that they are all correct and without any redundancy. If a pattern produces too many errors, it is a 'bad' pattern and should be removed or modified. If a pattern can be replaced by other patterns without affecting the system's performance, it is a 'redundant' pattern and should be deleted.

For example, consider pattern $p_i \in P$ and $p_i = \{$PTN VBZ IN PTN$\}$. (For simplicity, the word set of each component is omitted.) If we compare the pattern p_i with another pattern $p_i^* = \{$PTN VBZ PTN$\}$, we find that, since p_i^* is simpler than p_i, it has better generalization ability than p_i does. Suppose S and S^* are the sets of tagged sentences which can be matched by patterns p_i and p_i^*, respectively. It is obvious that $S \subseteq S^*$. If we replace p_i with p_i^* in P, P becomes simpler and can match more sentences; this means that P's generalization power improves over the operation, assuming the new pattern does not introduce new false positives.

The optimal pattern set should satisfy the following three criteria: least number of (false positive) errors in extracted interactions, least redundancy in patterns and maximum number of sentences which can be matched by at least one pattern. Obviously, these criteria cannot be achieved simultaneously. Thus, the optimization task becomes one of finding the best balance among those criteria.

Downloaded from https://academic.oup.com/bioinformatics/article/21/15/3294/195397 by guest on 12 March 2021

Assume $S = \{s_1, s_2, \ldots, s_n\}$ is a set of sentences, and $I = \{I_1, I_2, \ldots, I_n\}$ the set of interactions extracted from S through the pattern set $P = \{p_1, p_2, \ldots, p_m\}$. The pattern matching method is defined as a function F with parameters S and P, $I = F(S, P)$. If the true interaction set defined by S is $I^* = \{I_1^*, I_2^*, \ldots, I_n^*\}$, then the total expected extraction error R is

$$R(P) = \int_S L(S, P)\, \mathrm{d}G(S), \tag{1}$$

where $G(S)$ is the probability distribution of S, and $L(S, P) = |I^* - F(S, P)|$ is the lost function. Then the best pattern set P^* is the P which minimizes the expected risk $R(P)$:

$$P^* = \arg\min_P R(P) = \arg\min_P \int_S L(S, P)\, \mathrm{d}G(S). \tag{2}$$

Rissanen (1978) proposed the MDL principle as a tool to solve the trade-off problem between generalization power and accuracy. The MDL principle can be applied without the analytical form of the risk function, and hence is suitable in our case.

3.1 MDL principle

The MDL principle states that, given some data D, the best model (or theory) M_{mdl} in the set $\mathtt{M}$ of all models is the one that minimizes the sum of the length in bits of the description of the model and the length in bits of the description of the data with the aid of the model:

$$M_{\mathrm{mdl}} = \arg\min_{M \in \mathtt{M}} l(M) + l(D|M), \tag{3}$$

where $l(M)$ and $l(D|M)$ denote, respectively, the description length of the model M and that of data D using model M.

If we have a proper means to utilize regularities in the data, we can have a shorter representation for the model and for encoding the data using the model. The MDL principle can be viewed from the point of view of Kolmogorov complexity:

$$M_{\mathrm{mdl}} = \arg\min_M K(M) + K(D|M) \tag{4}$$

where $K(\,)$ is Kolmogorov complexity (Li and Vitanyi, 1997). The MDL principle looks for an optimal balance between the regularities (in the model) and the randomness remaining in the data, that is, a trade-off between the complexity of the model and the fitness of the model to the data.

The MDL principle serves as a general guide to solving the problems of model selection and parameter regression.

Without loss of generality, we assume the interaction set I to be a sequence given by $I = I_1 I_2 \ldots I_n$. The expected risk $R(P)$ is affected by the stochastic characteristic of sequence I, which means that, if we want to minimize $R(P)$, we should try to describe I in minimum length with the aid of P. We define $K(I) = K(P) + K(I|P)$ as the description length of I through P, where $K(P)$ is the description length of pattern set P and $K(I|P)$ is that of I given P. Then our optimizing task becomes one of trying to find a pattern set P^* which describes the interaction sequence I as briefly as possible, that is,

$$P^* = \arg\min_P K(I) = \arg\min_P K(P) + K(I|P). \tag{5}$$

In order to calculate $K(I|P)$, we first assume the expected interaction set I^* as a sequence given by $I^* = I_1^* I_2^* \ldots I_n^*$, similarly to I. Then, obviously, if $I = I^*$, the pattern set P is the perfect set for the sentence set S, no error happened and the description length of the sequence I is equal to the description length of pattern set P: $K(I) = K(P)$. If $I \neq I^*$, it means that there exist errors in the interaction sequence I. Here we define the Hamming distance of the two interaction sequences:

$$d(I, I^*) = \sum_{i=1}^{n} \delta(I_i, I_i^*), \tag{6}$$

where $\delta(I_i, I_i^*) = \begin{cases} 1, & I_i \neq I_i^* \\ 0, & I_i = I_i^* \end{cases}$.

It is shown that the sequence I can be represented by modifying the ideal sequence I^* in d positions where I and I^* are different. So, in order to describe the length I, we have to know how many differences exist, which are indicated by d, and the exact positions of these differences, which can be calculated as C_n^d. Thus the description length of sequence I is

$$\begin{aligned} K(I) &= K(P) + K(I|P) \\ &= K(P) + \log_2 d(I, I^*) + \log_2 C_n^d + \mathrm{c}, \end{aligned} \tag{7}$$

where c is a constant. Then the optimal pattern set P^* is obtained as follows:

$$\begin{aligned} P^* &= \arg\min_P K(P) + K(I|P) \\ &= \arg\min_P K(P) + \log_2 d(I, I^*) + \log_2 C_n^d. \end{aligned} \tag{8}$$

For simplicity, we usually take the *exception-based* MDL principle as an approximation of Equation (8), as follows:

$$M_{E-\mathrm{MDL}} = \arg\min_M K(M) + K(E|M), \tag{9}$$

where E are the exceptions from the expected result.

Vitányi and Li (2000) have proved that the exception-based MDL can be justified and reduced to the MDL principle of Equation (4) under circumstances of 'supervised learning'. It is effective for our task of pattern set optimization, and the optimal pattern set P^* is obtained as follows:

$$P^* = \arg\min_P K(P) + \log_2 d(I, I^*), \tag{10}$$

where I and I^* are the extracted and optimal interaction sequences, respectively, and $d(I, I^*)$ is the number of differences between I and I^*.

3.2 Pattern set optimization

The pattern set is optimized by the MDL principle as shown in Equation (10), which consists of two components, $K(P)$ and $d(I, I^*)$. For convenience, we assume $\mathrm{DL}(P) = K(P) + \log_2 d(I, I^*)$ is the description length of the system. Then the optimization task becomes one of minimizing $\mathrm{DL}(P)$ by adjusting parameter P. In order to get $\mathrm{DL}(P)$, $K(P)$ and $d(I, I^*)$ should be calculated, where $d(I, I^*)$ is the level of errors caused by using P to extract interactions. These errors include wrong interactions (false positives) and missing interactions (false negatives), shown as follows:

$$\begin{aligned} d(I, I^*) &= N_{\mathrm{wrong}} + N_{\mathrm{miss}} \\ &= (N_{\mathrm{extracted}} - N_{\mathrm{correct}}) + (N_{\mathrm{expected}} - N_{\mathrm{correct}}) \\ &= N_{\mathrm{extracted}} + N_{\mathrm{expected}} - 2\, N_{\mathrm{correct}} \end{aligned} \tag{11}$$

where N_{wrong} is the number of wrong interactions extracted, N_{miss} is the number of missed interactions, N_{expected} is the number of interactions expected to be extracted, $N_{\mathrm{extracted}}$ is the total number of interactions actually extracted, including the correct ones and erroneous ones, and N_{correct} is the number of interactions correctly extracted.

Since $K(P)$ is the Kolmogorov complexity of pattern set P and is non-computable, it is approximated by the code length of the pattern set $P = \{p_1, p_2, \ldots, p_m\}$, and $p_i = m_i^1\, m_i^2 \ldots m_i^{c(p_i)}$, where $c(p_i)$ is the number of components of pattern p_i such that

$$K(P) \approx \sum_{i=1}^{m} \sum_j |m_i^j|, \tag{12}$$

where $|m_i^j| = \begin{cases} 1, & m_i^j = PTN \\ \gamma / c(m_i^j), & \text{otherwise} \end{cases}$.

From (4), it is shown that if a component's word set includes more words, the pattern is more general, matching more sentences, and simpler. $|m_i^j|$ decreases when $c(m_i^j)$ increases, such that m_i^j contributes less to the complexity of P, and $K(P)$ is accordingly decreased.

Once $\mathrm{DL}(P)$ has been calculated, we can optimize the pattern set P by minimizing $\mathrm{DL}(P)$ taking the parameter of P. Since the modification of P is

Downloaded from https://academic.oup.com/bioinformatics/article/21/15/3294/195397 by guest on 12 March 2021

Input: pattern pair p_1 and p_2
Output: p_m, which is merged from p_1 and p_2
1. Sequence $A = (a_1, a_2, \ldots, a_k) = LCS(p_1, p_2)$
2. For each a_k, get the word set w_k by union of the corresponding word sets of a_k in p_1 and p_2
3. Let $p_m = \{a_1\ a_2 \ldots\ a_k : w_1; w_2; \ldots; w_k\}$
4. If illegal(p_m) then p_m = NULL, go to 6
5. If p_m is the same as one of patterns in the original pattern set, then p_m = NULL, go to 6
6. Output p_m

Fig. 2. Pattern merge algortihm.

ranged over the pattern set space P, the search space of the optimization methods is very large considering the infinite variation of the pattern components and word sets. For simplicity and efficiency, we take the 'superfluous then condense' strategy to guide the optimization process, which is as follow:

(1) Generate as many candidate patterns as possible, such that the initial pattern set covers all appropriate patterns in the system. This can be achieved by suspending all the restrictions imposed in the pattern generation phase.

(2) Merge the patterns obtained in (1), and add the new ones to the pattern set.

(3) Try to delete patterns in the pattern set by minimizing DL(P) such that the patterns remaining are all the most competitive 'good' patterns.

Pattern merging plays an important part in our pattern optimizing method. If patterns p_1 and p_2 cover most of their matched sentences, it is very likely that the two patterns are similar to each other. Then the newly merged pattern p_m is the longest pattern that can match all the sentences which p_1 and p_2 match. From the discussion above, p_m is simpler and has more generalization power than either p_1 or p_2. When p_1 and p_2 are replaced by p_m, the whole pattern set's generalization ability improves. In our algorithm, p_m is approximated by the LCS (longest common sequence) of the merged pattern pair and the detailed algorithm is shown in Figure 2.

In the merge algorithm, the function illegal(p) checks the merged pattern p using the following rules:

- At least two PTN exist in p.
- At least one VB or NN exists in p.

These rules are basic requirements of a pattern and impose no other limitations.

Then the pattern set, including both original and merged patterns, is optimized through the steepest gradient descent local search strategy. In our algorithm, the worst pattern, which incurs the largest increase of the description length of the system DL(P), is deleted in each iteration, until there no deletion of a pattern can lower DL(P). When DL(P) reaches the minimal value at pattern set P^*, the optimal pattern set P^* is the one that best fits our system. The whole algorithm is shown in Figure 3.

In our approach, a pattern is evaluated according to its effects to the whole pattern set P, rather than individually on the precision and recall rates of that pattern. Even if a pattern has good properties, it may be deleted if it is redundant and increases the system's description length. The final set P^* may not be the best pattern set in terms of minimizing the interaction error, but it has better generalizability and enhances the system's performance.

4 EXPERIMENTS

The corpus, consisting of 963 sentences, was collected by the following steps: we ran a web crawler program which is able to automatically download biomedical papers of interest to the user

Input: initial pattern set P^0, which contains the original patterns and the merged ones.
Output: optimal pattern set P^*
1. $P = P^0$
2. While P is not empty do
   (i) Calculate DL(P)
   (ii) MaxDL = –MAX_INTEGER
   (iii) For each p_i in P do
      (a) $P^1 = P - \{ p_i \}$
      (b) $\Delta DL = DL(P) - DL(P^1)$
      (c) If $\Delta DL > MaxDL$ then $MaxDL = \Delta DL$, $i^* = i$
   (iv) If $\Delta DL < 0$ then go to 3
   (v) $P = P - \{p_i^*\}$
3. Output $P^* = P$

Fig. 3. Algorithm for pattern optimization.

from the Internet by using the keywords 'protein–protein interaction', and the papers were sorted automatically according to their relevance to the keywords of the query; the first 8037 papers were selected, and full texts were segmented into 65 656 sentences; protein names in these sentences were identified based on an dictionary which contains about 60 000 items collected from the databases of PathwayFinder (Yao *et al.*, 2004); finally, those sentences with fewer than two protein names were discarded. It has to be mentioned that the protein name dictionary is far from complete, and its effect on the performance of the system is omitted from our experiments.

In these 963 sentences, 1435 interactions were labeled manually. Totally 192 patterns were generated as the initial pattern set without imposing any grammatical rules or optimization algorithm. By implementing our proposed optimization algorithm, patterns were merged and deleted until the optimal point was reached. The F-score is often used to evaluate a pattern set, indicating the overall performance in terms of both the precision and the recall rate. It is defined as follows:

$$F\text{-score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (13)$$

As it was hard to perform a comparative study with the limited number of 963 sentences and 1436 interactions, we implemented a strategy of cross-validation, and calculated the average performance over 10 runs of labeled sentences. First we divided the sentences into 10 equal sets. Then we randomly selected seven sets for training and three sets for testing. This procedure was repeated 10 times. Precision and recall rates were averaged over these 10 runs.

Our experiment is in four parts: first, the original pattern set is optimized from all 963 sentences with only the deletion method; the description length, precision and recall rates are shown in Figure 4. Second, the extraction result is compared with that of the rule-based approach in testing the deletion operation's effectiveness in reducing the pattern numbers. Third, our algorithm with both merging and deletion is evaluated. Finally, the generalizabilty of our algorithm is tested by comparing it with the empirical risk minimize (ERM) algorithm as a baseline.

4.1 Pattern set optimization

From Figure 4(a), the minimum description length is obtained at the deletion of pattern number 162, which means there are 30 patterns left

Downloaded from https://academic.oup.com/bioinformatics/article/21/15/3294/195397 by guest on 12 March 2021

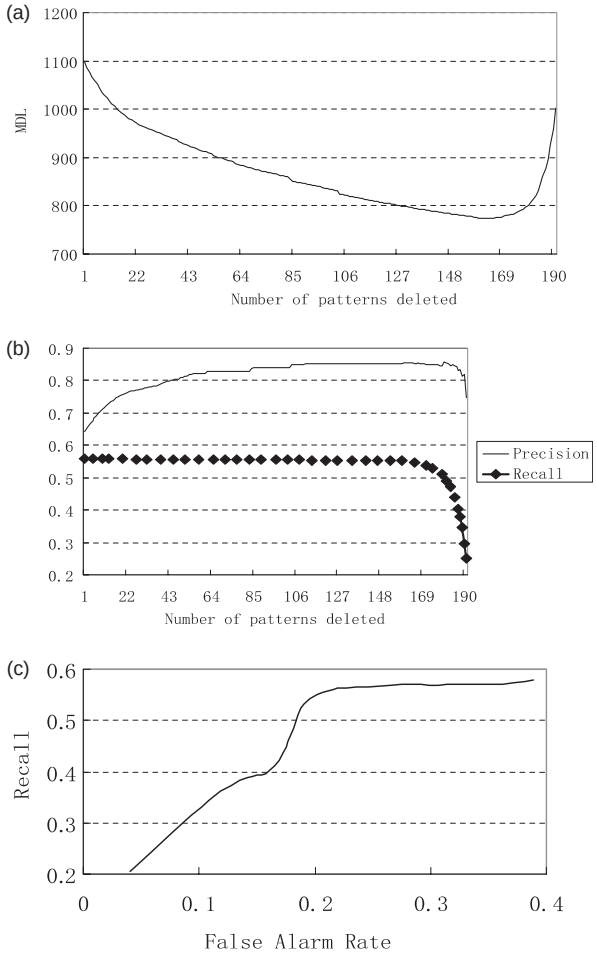

Fig. 4. Pattern set optimization. (a) MDL system description length vs. number of patterns deleted. (b) Precision and recall rates vs. number of patterns deleted. (c) ROC Curve.

in the optimal pattern set. The bottom of the curve (near the optimal point) is flatter than the beginning and ending parts of the curve because there exist many trivial patterns in the set which have no effect on the extraction accuracy. The beginning part of the curve is steeper because the deletion of bad patterns reduces erroneous interactions. The tail part is steepest when some 'huge' good patterns are deleted. For example, pattern {PTN VB IN PTN: *; interact associate; with; *} matches 88 interactions, 77 of which are correct. Most of the errors are introduced by some bad patterns, and a very few 'huge' patterns do most of the work of matching.

This also explains why the recall rate curve in Figure 4(b) drops dramatically immediately after the optimal pattern number is reached. In Figure 4(b), precision reaches its peak at the optimal point of 162 and drops gently as precision is determined more by the property of 'huge' patterns left in the pattern set. The ROC curve is shown in Figure 4(c).

4.2 Effectiveness compared with the rule-based approach

Our approach is effective in optimizing the pattern set. In accordance with Section 4.1, most of the erroneous and trivial patterns are deleted, with only those good and 'huge' patterns remaining in the set. Although manually drafted rules can filter the patterns, it is always hard to find the optimal rules which preserve only those good ones. We compared our approach with the rule-based one in (Huang *et al*., 2004), some rules of which are

(1) If a pattern has neither verb tag nor noun tag, reject it.
(2) If the last tag of a pattern is IN or TO, reject it.
(3) If the left neighborhood of a CC tag is not equal to the right neighborhood of the tag in a pattern, reject the pattern.

Table 2. System performance of the MDL-based approach and rule-based approach

Algorithm	No. of patterns	Precision (%)	Recall (%)	*F*-score (%)
Original	192	64.6	57.2	60.68
Rule	65	71.2	51.0	59.43
MDL*	30	85.1	55.8	67.40

Original is the pattern set originally generated. *Rule* is the optimized pattern set using the rule-based approach given by Huang *et al*. (2004). *MDL** is the optimized pattern set using the MDL approach from the original pattern set with only the deletion operation.

The result is shown in Table 2.

In the training set part of Table 2, the rule-based approach reduced the pattern number from 192 to 65, whereas the MDL-based deletion method reduced the number of patterns to 30. Although the precision of the rule-based approach improves by 6.6% from the original, the recall rate declines by 5.8%, which makes the *F*-score decrease by 1.25%. The MDL-based approach improves the precision rate by 22.5%, the recall rate declines by only 1.4% and the *F*-score increases by 6.72%. It is clear that the MDL-based approach has better performance than the rule-base one, even without the merging method.

The manually drafted rules abruptly remove some good patterns and allow many erroneous patterns. The precision of the rule-based system remained almost the same, but the recall rate dropped significantly. The following are examples of some good patterns that are deleted by the rule-based approach and otherwise preserved by the MDL-based approach with only the deletion method.

{IN PTN VBN IN PTN: that; *; conjugated interacted; with; *}
{PTN CC VB IN PTN: *; and; associates interacts; with; *}
{PTN VB CC IN PTN: *; interact interacts; and and/or; with; *}
{PTN VB IN PTN IN: *; associates interacts; with; *; in}

4.3 Merging

We applied the merge method described in Section 3 before condensing the pattern set. The result is shown in Table 3, and indicated by the name *MDL*. The number of patterns is now reduced to 14, as compared with 30 without merging. The precision reduces by 4.4% in the training set, and the recall rate increases by 5.2%, which makes the *F*-score increase by 2.15%. In the test set, the precision reduces by 4.2%, the recall rate increases by 5.2% and the *F*-score increases by 2.21%. It is shown that the system's performance is improved after the merge method is used. Each merged pattern covers more than one pattern of the pattern set; although it sacrifices a little precision, it discovers more correct interactions and has a much higher recall

Downloaded from https://academic.oup.com/bioinformatics/article/21/15/3294/195397 by guest on 12 March 2021

Table 3. System performance of the MDL-based approach, with/without merging, and the ERM-based approach

Algorithm	No. of patterns	Precision (%)	Recall (%)	F-score (%)
(a) Training set				
Original	192	64.6	57.2	60.68
ERM	134	82.7	55.7	66.57
MDL	14	80.7	61.1	69.55
(b) Test set				
Original	192	63.5	57.3	60.24
ERM	134	77.9	53.3	63.29
MDL	14	79.8	59.5	68.17

Original is the pattern set originally generated. *ERM* is the optimized pattern set using the ERM approach from the original pattern set. *MDL* is the optimized pattern set using the MDL approach from the original pattern set.

Table 4. An example of a merged pattern

Pattern	Pattern contents	Correct	Extracted
p_1	{PTN VB TO PTN: *; binds; to; *}	87	92
p_2	{PTN VB IN PTN: *; interact associate interacts associates; with ;*}	303	316
p_3	{PTN PTN VB PTN: *; *; form binds; *}	0	14
p_4	{PTN TO VB PTN: *; to; form inhibit; *}	1	19
p_5	{PTN VB PTN: *; transfer activates; *}	22	31
p_6	{PTN VB PTN PTN: *; activates bound; *; *}	4	4
Sum of p_1 to p_6	—	417	476
p_m	{PTN VB PTN: *; binds interact associate interacts associates form inhibit transfer activates bound;*}	514	598

rate. The merged patterns are simpler in nature, and their generalization power is better than that of the complex ones. Table 4 gives an example of a merged pattern.

In Table 4, p_{m} is merged from $p_1, p_2 \ldots p_6$ and extracts 97 more correct patterns than p_1, p_2 ... and p_6 collectively do, while introducing 25 erroneous ones.

With our merging method, the recall rate of MDL-based optimization increases by 3.9% from the original, the precision rate increases by 16.1% to 80.7% and the F-score attains its highest value of 69.55%, which indicates that our MDL-based optimization method is very effective.

4.4 Testing generalization ability

We have also implemented the well-known empirical risk minimize (ERM) algorithm to optimize the pattern set as a baseline for comparison. In the ERM algorithm the pattern set is optimized according to the empirical risk of the system defined as follows:

$$P^* = \arg\min_P d(I, I^*). \tag{14}$$

The average system performance is shown in Table 3.

From Table 3, our MDL-based approach reduces the pattern number from 192 to 14, whereas the ERM-based approach's optimal pattern number is 134, which is much larger. From Table 3(a), the F-score of the ERM-based approach is 2.98% lower compared with the MDL-based approach in the training set, and in the test set the MDL-based approach has a 6.2% edge over the ERM-based approach. Obviously, our approach has a better performance than the ERM-based one, especially in the test set, which means better generalization ability. In fact the ERM-based optimal pattern set contains too many trivial patterns which are in very complicated forms and can match to no more than one sentence. Those are problematic patterns that cause errors in the test set. Whereas the ERM-based algorithm cannot dispose of these patterns, the MDL-based algorithm can. Thus, although the ERM-based approach adapts to the data in the training set quite well, it has poorer generalization ability than our MDL-based approach. MDL solves the over-fitting problem of ERM.

5 DISCUSSION

We have presented a new paradigm of mining protein–protein interaction from the literature. Our method is fully automatic and works reasonably well. Our goal is also to demonstrate that our proposal of automatically generating and optimizing sentence patterns and using them to mine a targeted area of knowledge is feasible. Mining protein–protein interactions from the literature is not our final goal. We wish to demonstrate to readers, via our prototype for this particular domain, that this approach works in other domains, too. Specifically, answering Internet search queries beyond a simple keyword search comes to mind.

ACKNOWLEDGEMENTS

We would like to thank Jingbo Wang, Daming Yao and Dr Kunbin Qu and his team at Rigel for earlier collaboration on PathwayFinder. This work was supported by the Chinese Natural Science Foundation under grant nos 60272019 and 60321002, the Canadian NSERC grant OGP0046506, CITO, Canada Research Chair fund, and the Killam Fellowship.

Conflict of Interest: none declared.

REFERENCES

Bader,G.D. *et al*. (2001) BIND—the biomolecular interaction network database. *Nucleic Acids Res.*, **29**, 242–245.

Brill,E. (1995) Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. *Computational Linguistics*, **21**, 543–565.

Friedman,C. *et al*. (2001) Genies: a natural-language processing system for the extraction of molecular pathways from journal articles. *Bioinformatics*, **17**, S74–S82.

Hirschman,L. *et al*. (2002) Accomplishments and challenges in literature data mining for biology. *Bioinformatics*, **18**, 1553–1561.

Huang,M.L. *et al*. (2004) Discovering patterns to extract protein–protein interactions from full texts. *Bioinformatics*, **20**, 3604–3612.

Leroy,G. and Chen,H. (2002) Filling preposition-based templates to capture information from medical abstracts. In *Pacific Symposium on Biocomputing*, Hawaii, USA, Vol. **7**, pp. 350–361

Li,M. and Vitanyi,P. (1997) *An introduction to Kolmogorov complexity and its applications*. Springer-Verlag.

Marcotte,E.M. *et al*. (2001) Mining literature for protein–protein interactions. *Bioinformatics*, **17**, 359–363.

Ng,S.K. and Wong,M. (1999) Toward routine automatic pathway discovery from on-line scientific text abstracts. In *Proceedings of 10th International Workshop on Genome Informatics*, Tokyo, pp. 104–112.

Ono,T. *et al*. (2001) Automated extraction of information on protein–protein interactions from the biological literature. *Bioinformatics*, **17**, 155–161.

Downloaded from https://academic.oup.com/bioinformatics/article/21/15/3294/195397 by guest on 12 March 2021

Park,J.C., Kim,H.S. and Kim,J.J. (2001) Bidirectional incremental parsing for automatic pathway identify-cation with combinatory categorical grammar. In *Proceedings of the Pacific Symposium on Biocomputting*, Hawaii, USA, pp. 396–407.

Pustejovsky,J., Castano,J., Zhang,J., Kotecki,M. and Cochran,B. (2002) Robust relational parsing over biomedical literature: extracting inhibit relations. In *Proceedings of the 7th Pacific Symposium on Biocomputing 2002*, Hawaii, USA, pp. 362–373.

Ray,S. and Craven,M. (2001) Representing sentence structure in hidden markov models for information extraction. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence* (*IJCAI 2001*), Morgan Kaufmann, pp. 1273–1279.

Rissanen, J. (1978) Modelling by shortest data description. *Automatica*, **14**, 465–471.

Salwinski,L. *et al*. (2004) The database of interacting proteins: 2004 update. *Nucleic Acids Res.*, **32**, D449–D451.

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

Thomas,J., Milward,D., Ouzounis,C., Pulman,S. and Carroll,M. (2000) Automatic extraction of protein interactions from scientific abstracts. In *Proceedings of the Pacific Symposium on Biocomputing*, Hawaii, USA, pp. 541–551.

Vitányi,P. and Li,M. (2000) Minimum description length induction, Bayesianism and Kolmogorov Complexity. *IEEE transactions on information theory*, **47**, 446–464.

Wong,L. (2001) A protein interaction extraction system. In *Proceedings of Pacific Symposium on Biocomputing 2001*, Hawaii, pp. 520–530.

Yakushiji,A., Tateisi,Y., Miyao,Y. and Tsujii,J. (2001) Event extraction from biomedical papers using a full parser. *Proceedings of the sixth Pacific Symposium on Biocomputing 2001*, Hawaii, USA, pp. 408–419.

Yao,D., Wang,J., Lu,Y., Noble,N., Sun,H., Zhu,X., Lin,N., Payan,D.G., Li,M., Qu,K. *et al*. (2004) PathwayFinder: paving the way towards automatic pathway extraction. In Yi-Ping Phoebe Chen(eds), *Bioinformatics 2004: Proceedings of the 2nd Asia-Pacific Bioinformatics Conference* (*APBC*), **29** volume of *CRPIT*, pp. 53–62.

Downloaded from https://academic.oup.com/bioinformatics/article/21/15/3294/195397 by guest on 12 March 2021