BIOINFORMATICS ORIGINAL PAPER Vol. 27 no. 10 2011, pages 1422–1428
doi:10.1093/bioinformatics/btr155

Data and text mining

Advance Access publication March 30, 2011

Inference of transcriptional regulatory network by bootstrapping patterns

Hei-Chia Wang[1], Yi-HSiu Chen[1], Hung-Yu Kao[2] and Shaw-Jenq Tsai[3,4,*]

[1]Institute of Information Management, [2]Department of Computer Science & Information Engineering, [3]Institue of Basic Medical Science, College of Medicine and [4]Department of Physiology, College of Medicine, National Cheng Kung University, Tainan 701, Taiwan

Associate Editor: Jonathan Wren

ABSTRACT

Motivation: Transcriptional regulatory networks, which consist of linkages between transcription factors (TF) and target genes (TGene), control the expression of a genome and play important roles in all aspects of an organism's life cycle. Accurate prediction of transcriptional regulatory networks is critical in providing useful information for biologists to determine what to do next. Currently, there is a substantial amount of fragmented gene regulation information described in the medical literature. However, current related text analysis methods designed to identify protein–protein interactions are not entirely suitable for finding transcriptional regulatory networks.

Result: In this article, we propose an automatic regulatory network inference method that uses bootstrapping of description patterns to predict the relationship between a TF and its TGenes. The proposed method differs from other regulatory network generators in that it makes use of both positive and negative patterns for different vector combinations in a sentence. Moreover, the positive pattern learning process can be fully automatic. Furthermore, patterns for active and passive voice sentences are learned separately. The experiments use 609 HIF-1 expert-tagged articles from PubMed as the gold standard. The results show that the proposed method can automatically generate a predicted regulatory network for a transcription factor. Our system achieves an F-measure of 72.60%.

Availability: The software, training/test datasets and learned patterns are available at http://140.116.99.138/~hcw0901/PubMedSearch.php

Contact: seantsai@mail.ncku.edu.tw

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on November 22, 2010; revised on March 15, 2011; accepted on March 16, 2011

1 INTRODUCTION

Transcription factors (TFs) play an important role in regulatory systems. A substantial amount of biological research is continuously being conducted, and the cumulative results obtained from the subsequent analyses are increasing rapidly. Many of these results are reported in the medical literature (Raychaudhuri, 2006). However, given this explosive growth, it is becoming more difficult for biologists to search, read and assemble the information they need. Therefore, many high-throughput techniques for extracting useful information from the literature have been proposed. Much research (Bui *et al*., 2011; Chowdhary *et al*., 2009; Jang *et al*., 2006; Marcotte *et al*., 2001; Niu *et al*., 2010; Ono *et al*., 2001) has been conducted on predicting protein–protein interactions (PPIs) through text mining. These methods aim to extract more information than what is currently contained in curated databases such as BIND (Bader *et al*., 2001), KEGG (Kanehisa *et al*., 2002), SwissProt (Bairoch and Apweiler, 2000) and the Database of Interacting Proteins (Xenarios *et al*., 2000). However, current-related text analysis methods designed for finding PPI are not entirely suitable for finding transcriptional regulatory networks, because the presence of two proteins in a PPI sentence does not necessarily indicate these proteins have a TF and Target gene (TGene) relationship or that one controls the gene activity of the other. The descriptive terms used for determining PPIs may be different from those necessary to determine gene regulation. For example, the words 'associate' and 'interact', which are descriptions that are useful for predicting PPIs, will cause a high-throughput automatic method to find substantial amounts of false positives in a regulatory network. Moreover, the regulatory descriptions could be different in articles written in passive versus active voice. Other existing systems (Jensen *et al*., 2006; Leroy and Chen, 2002; Narayanaswamy *et al*., 2005; Ono *et al*., 2001; Saric *et al*., 2006) use manually defined patterns, a popular writing style for describing gene interactions. However, these systems have low recall values. Furthermore, existing methods rarely discuss how negative terms will cause false positives (Sanchez-Graillet and Poesio, 2007).

Therefore, this study takes advantage of bootstrapping (Abney, 2002; Chang and Choi, 2006; Feng and Chua, 2003; Yangarber *et al*., 2002) to automatically learn patterns for describing gene regulation relationships and generating a transcriptional regulatory network.

2 SYSTEM OVERVIEW

Our system architecture is composed of three tasks and is shown in Figure 1. In our previous work (Kooi, 2008), we trained a set of initial patterns that included positive and negative patterns from manually tagged hypoxia inducible factor-1 (HIF-1)-related articles. However, this method required human judgment to decide whether a sentence indicated a relationship between a TF and a TGene. In this article, we develop an automatic method for learning patterns. Manually tagged sentences from our previous research (Kooi, 2008)

*To whom correspondence should be addressed.

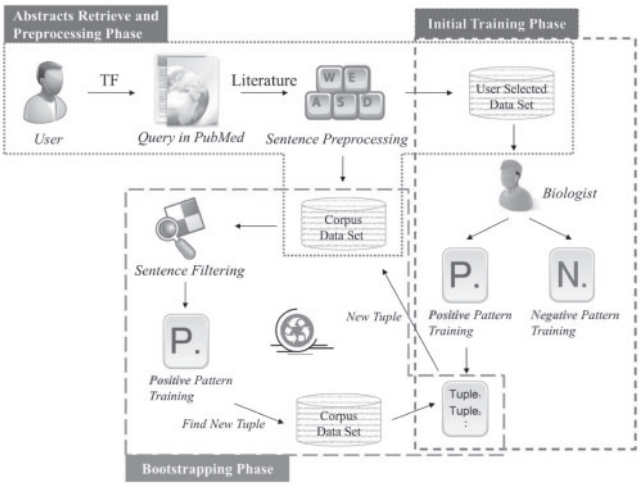

Fig. 1. Architecture of the proposed method.

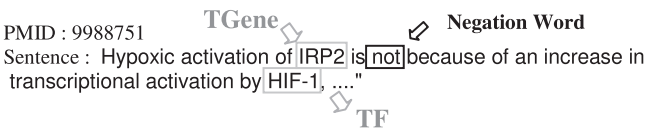

Fig. 2. An example of negative sentence.

will be used as the gold standard. The proposed method includes three phases:

(1) Abstract retrieval and preprocessing phase: a TF of interest is added into the system, and TF-related abstracts are collected from PubMed (http://www.ncbi.nlm.nih.gov/pubmed/). The system parses all sentences containing a (TF, TGene)-tuple. (A sentence is said to contain a tuple if a TF and TGene both appear in a sentence.) We use Sequence Retrieval Systems (SRS) from the European Molecular Biology Laboratory (EMBL) and HUGO Gene Nomenclature Committee (HGNC) databases to identify TFs and TGenes.

(2) Initial pattern training phase: all sentences in the articles containing a (TF, TGene)-tuple are listed for the user. The user tags positive and negative sentences (an example is shown in Fig. 2), which are then used for training. If there is no feedback from the user, the system randomly selects *n* articles that contain tuples for training. Based on our evaluation of different values of *n*, an *n* of 5 is suggested. The system will first filter out sentences by negative patterns, if any negative patterns have been learned. Presently, we use negative patterns that were learned from the manually tagged 1048 negatively marked sentences. The rest of the sentences are used to train positive patterns. Each sentence containing a tuple is parsed into the '<Before> Entity1 <Middle> Entity2 <After>' format as shown in Figure 3. This format includes three vectors, <Before>, <Middle> and <After>, which are then divided by the TF/TGene entity. Patterns are then trained from different vector combinations named <Before>, <Middle>, <After>, <Before+Middle>, <Before+After>, <Middle+After> and <Before+Middle+After>.

PMID：10777486
Sentence：Role of hypoxia-inducible factor-1 *in transcriptional activation of* ceruloplasmin *by iron deficiency.*
TF and TGene Tagging：Role of TF in transcriptional activation of TGene by iron deficiency.
POS Tagging: [NP Role] [PP of] [NP TF] [PP in] [NP transcriptional activation] [PP of] [NP TGene] [PP by] [NP iron deficiency] .

<Before>		$Entity_1$	<Middle>			$Entity_2$	<After>	
Role	of	TF	in	Transcriptional activation	of	TGene	by	Iron deficiency
NP	PP	NP	PP	NP	PP	NP	PP	NP

Fig. 3. An example sentence in the vector format for tuple (HIF-1, ceruloplasmin).

I <Before>: the sub-sentence vector located to the left of Entity1.
II Entity1: the TF or TGene that is between <Before> and <Middle>.
III <Middle>: the sub-sentence vector located between Entity1 and Entity2.
IV Entity2: the TGene or TF that is between <Middle> and <After>. For example, Entity1 is TF and Entity2 is TGene.
V <After>: the sub-sentence vector located to the right of Entity2.

(3) Bootstrapping phase: in Phase 2, patterns from initial selected articles are found. These patterns can then be used to search other, still unused, articles in the corpus dataset to determine if new (TF, TGene)-tuples can be found. If a new tuple is extracted from the dataset, the process goes back to Phase 2 to find new patterns. This process can be iterated until no new tuples are found in the tuple-finding step. The bootstrapping process is shown in Figure 4a. A real example is shown in Figure 4b.

After these three tasks, relationships between genes can be determined, as shown by Ring 1 in Fig. 5. A simplified, hypothetical example of gene relationships is shown in Figure 5. In this relationship, some of the TGenes could be also be TFs themselves. TFs found by the bootstrap search can then be used as inputs to continue the process and, in this way, a gene network can be constructed. The system interface allows the user to see the related genes and the corresponding sentences. Part of the HIF-1 gene network generated by this process is shown in Figure 6.

3 EVALUATION

3.1 Data source

Although there are several available benchmark datasets for PPI, such as AIMed, BioInfer, HPDR50, IEPA and LLL (Pyysalo *et al.* 2008), they are not suitable for evaluating transcriptional regulatory networks. The main reason is that PPI sentences do not necessarily indicate TF–TGene relationships or whether one gene controls the gene activity of the other. Therefore, we manually tagged a benchmark dataset for evaluating transcriptional regulatory networks. This study used 'HIF-1' as a keyword to search articles from PubMed. A total of 5651 sentences in 602 studies were found. Not all sentences with a (TF, TGene) tuple, only 1578 sentence with tuple can be used to train patterns. Moreover, not all sentences with

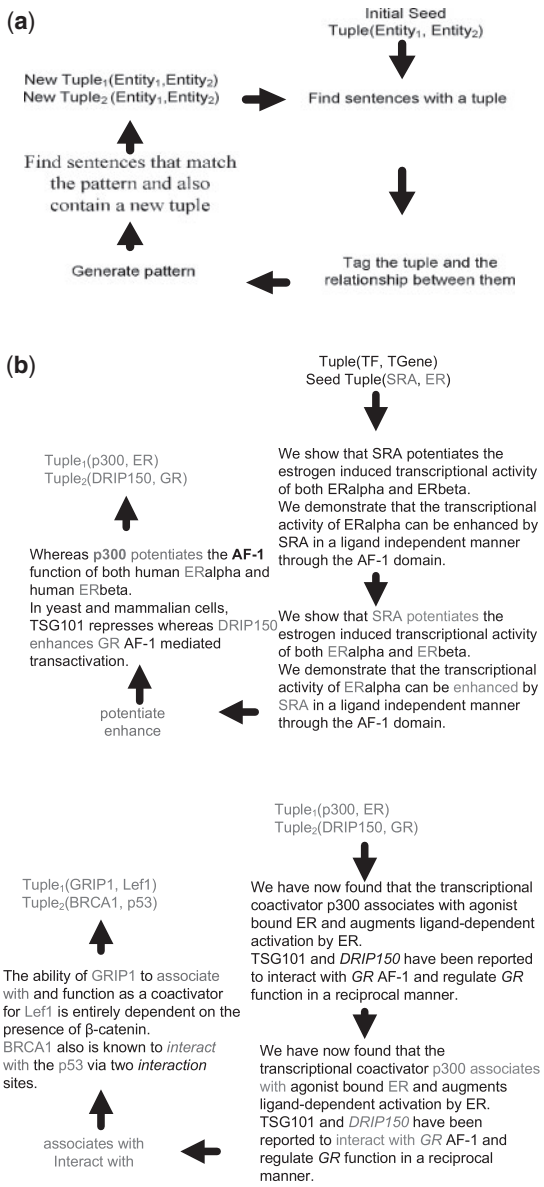

Fig. 4. (**a**) The bootstrapping process. (**b**) An example of the bootstrapping learning process. This example shows how positive patterns can be learned.

tuple should be train as 'positive' pattern because some sentences do not directly indicate whether a TF regulates a TGene. When sentences did not directly indicate regulation, they were tagged with negative marks and used for training 'negative' patterns. After manually tagging these sentences, 531 positive marks and 1047 negative marks were obtained for experimental and benchmarking purposes.

Sentences analyzed in this study were classified as either active or passive voice according to the location of the TF in the sentence. Active sentences were marked as TF–TGene-style, and passive sentences were marked as TGene–TF-style. In the data, there were 847 TF–TGene sentences composed of 316 positive sentences and 531 negative sentences. There were a total of 731 TGene–TF sentences comprising 215 positive sentences and 516 negative

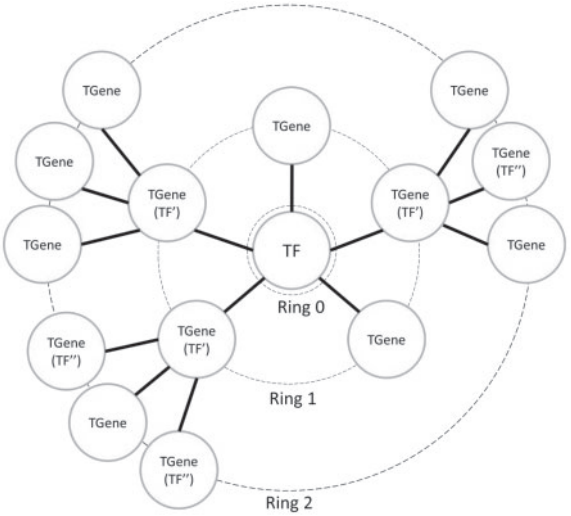

Fig. 5. Gene regulatory network.

Fig. 6. An example of a gene network.

sentences. Among these sentences, some included two TFs and one TGene (TF–TGene–TF-style) or two TGenes and one TF (TGene–TF–TGene-style) in a sentence. These relationships were counted in both TF–TGene and TGene–TF categories.

3.2 Results

3.2.1 Experiment 1 How much initial training data should be used for training positive patterns? This experiment identified how many manually tagged initial abstracts should be used for our proposed approach to reach an acceptable performance. We evaluated the results of the overall method from different vector combinations of TF–TGene and TGene–TF sentences. The results, shown in Table 1, indicate that three initial abstracts are the minimum requirement, and the F-measure reaches a stable condition when five abstracts are used.

We also used error rates to evaluate performance. In this study, we define error rate as follows:

(1) If genes in a tuple can sustain only one iteration and are then unable to proceed with bootstrap training, the identification of those genes is regarded as an error incident.

(2) If genes in a tuple can sustain more than two iterations but the F-measure is $<10\%$, it is treated as an error.

Table 1. Training abstract number and resulting performance in different vector combinations

TF–TGene-style							TGene–TF-style						
	Vector	Training abstract no.						Vector	Training abstract no.				
		1	2	3	4	5			1	2	3	4	5
Precision	B (%)	39.57	35.95	33.91	33.67	33.67	Precision	B (%)	35.22	31.03	31.32	28.97	28.42
	M (%)	46.36	43.98	40.08	40.28	40.26		M (%)	48.44	54.42	34.90	32.32	33.64
	A (%)	41.36	42.30	42.00	42.00	42.00		A (%)	34.04	33.12	32.54	31.54	31.18
	B+M (%)	56.76	49.02	43.94	45.12	43.98		B+M (%)	55.86	55.34	49.42	39.62	40.84
	B+A (%)	30.17	32.46	35.90	35.35	34.59		B+A (%)	26.39	31.19	41.54	39.68	35.52
	M+A (%)	36.39	40.27	37.35	38.54	36.95		M+A (%)	23.88	38.90	44.33	39.31	34.50
	B+M+A (%)	34.90	47.95	45.46	44.24	45.42		B+M+A (%)	29.36	41.73	44.42	56.01	47.07
Recall	B (%)	50.04	69.57	74.03	75.37	75.55	Recall	B (%)	42.19	61.29	71.40	71.73	77.22
	M (%)	72.54	84.86	92.60	93.26	92.58		M (%)	49.60	55.06	86.24	90.36	87.84
	A (%)	60.42	65.24	67.00	67.00	67.00		A (%)	39.66	54.28	57.00	61.58	63.08
	B+M (%)	46.02	69.12	82.30	82.26	84.98		B+M (%)	26.96	41.06	62.36	73.28	72.90
	B+A (%)	29.43	40.87	45.27	44.94	45.66		B+A (%)	16.03	28.58	37.11	44.44	45.57
	M+A (%)	38.75	55.55	67.42	65.37	67.65		M+A (%)	18.69	40.15	49.22	59.30	70.97
	B+M+A (%)	38.30	43.26	52.37	57.87	55.14		B+M+A (%)	9.95	21.68	30.64	34.30	42.19
F-measure	B (%)	31.36	43.34	45.70	46.55	46.58	*F*-measure	B (%)	23.68	33.89	38.17	38.73	40.84
	M (%)	45.10	51.56	55.86	56.14	56.00		M (%)	26.18	29.36	44.64	45.58	44.60
	A (%)	47.98	51.20	52.00	52.00	52.00		A (%)	32.40	38.62	39.58	41.26	41.64
	B+M (%)	32.36	47.44	56.34	56.70	57.96		B+M (%)	16.28	25.22	36.98	43.28	43.54
	B+A (%)	25.32	34.92	38.79	38.72	39.34		B+A (%)	12.28	21.53	28.00	33.37	34.26
	M+A (%)	28.00	39.59	48.06	46.06	47.78		M+A (%)	11.24	24.57	29.82	35.99	42.71
	B+M+A (%)	32.35	36.65	44.31	48.54	46.24		B+M+A (%)	9.29	19.24	28.03	30.97	37.09
Error	B (%)	34.00	8.00	2.00	0.00	0.00	Error	B (%)	48.00	24.00	10.00	10.00	2.00
	M (%)	22.00	8.00	0.00	0.00	0.00		M (%)	48.00	42.00	8.00	2.00	4.00
	A (%)	12.00	4.00	0.00	0.00	0.00		A (%)	52.00	22.00	16.00	6.00	2.00
	B+M (%)	34.00	14.00	2.00	2.00	0.00		B+M (%)	68.00	48.00	24.00	12.00	12.00
	B+A (%)	36.00	12.00	2.00	2.00	0.00		B+A (%)	70.00	46.00	28.00	16.00	14.00
	M+A (%)	42.00	18.00	0.00	4.00	0.00		M+A (%)	76.00	48.00	34.00	18.00	4.00
	B+M+A (%)	34.00	26.00	10.00	2.00	6.00		B+M+A (%)	76.00	56.00	38.00	32.00	18.00

B, M and A in the Vector column represent <Before>, <Middle> and <After>, respectively.

Note that the error rate for TGene–TF-style sentences is higher than that for TF–TGene-style sentences. The results also indicate that *F*-measures become acceptable when more initial abstracts are used for training.

3.2.2 Experiment 2 Initial dataset selection. Using the following three methods of selecting sentences to include in the dataset, we analyzed whether automatic processes for finding patterns are possible.

(1) Manually selected positive sentences for pattern training (manual-marked).
(2) A few manually selected abstracts that contain positive sentences for pattern training (semi-automatic).
(3) Randomly selected abstracts that include either TF–TGene-style or TGene–TF-style sentences for bootstrapping pattern training (automatic).

The results of this experiment, shown in Table 2, indicate that none of these three methods is clearly the best. This lack of consistent differences between the methods may be because the *F*-measures shown are the average values across 15 experiments. We compare averages because not every experiment is perfect and errors sometimes occur. If we observe just the averaged results, the difference between the three methods is not obvious. Additionally, if we randomly take a small number of abstracts as the initial dataset and proceed with iterative pattern learning through bootstrapping, the final patterns learned by the automatic or semi-automatic method are similar to those obtained from the manual method. In some cases, the semi-automatic or automatic methods were even better than the manual method. Therefore, the use of manual tagging instead of automatic or semi-automatic tagging does not dramatically influence the result of the gene network finding method. This result indicates that when choosing an initial dataset, it is not necessary to manually mark all of the sentences if the selected literature contains tuple sentences. This finding shows that it is possible to reduce the amount of human effort required to generate a regulatory network.

3.2.3 Experiment 3 Can qualified negative patterns help? In this experiment, sentences were filtered with qualified negative patterns learned from manually tagged literature according to the three types of initial datasets listed in Experiment 2. The use of negative patterns

Table 2. The average F-measure values of different initial data sources

TF–TGene-style							TGene–TF-style						
		Training abstract no.							Training abstract no.				
		1	2	3	4	5			1	2	3	4	5
B	Manual (%)	31.36	43.34	45.70	46.55	46.58	B	Manual (%)	23.68	33.89	38.17	38.73	40.84
	Semi-auto (%)	37.10	46.54	46.53	46.57	46.57		Semi-auto (%)	23.66	37.68	40.89	41.35	40.85
	Automatic (%)	38.37	46.54	46.04	45.89	46.56		Automatic (%)	23.31	38.52	39.81	37.82	41.36
M	Manual (%)	45.10	51.56	55.86	56.14	56.00	M	Manual (%)	26.18	29.36	44.64	45.58	44.60
	Semi-auto (%)	44.18	54.52	56.08	56.16	56.04		Semi-auto (%)	36.60	38.98	43.70	45.72	46.30
	Automatic (%)	23.34	41.06	45.90	45.18	54.88		Automatic (%)	24.84	37.48	39.62	45.94	46.20
A	Manual (%)	47.98	51.20	52.00	52.00	52.00	A	Manual (%)	32.40	38.62	39.58	41.26	41.64
	Semi-auto (%)	46.36	50.86	51.28	52.00	52.00		Semi-auto (%)	29.04	36.92	39.98	41.66	41.30
	Automatic (%)	46.76	50.56	52.00	52.00	52.00		Automatic (%)	33.44	38.68	39.96	40.98	41.34
B+M	Manual (%)	32.36	47.44	56.34	56.70	57.96	B+M	Manual (%)	16.28	25.22	36.98	43.28	43.54
	Semi-auto (%)	34.06	55.96	59.10	59.08	59.48		Semi-auto (%)	24.72	39.20	45.48	46.72	45.22
	Automatic (%)	36.98	54.46	57.06	58.32	58.16		Automatic (%)	21.32	33.86	37.42	39.80	45.88
B+A	Manual (%)	25.32	34.92	38.79	38.72	39.34	B+A	Manual (%)	12.28	21.53	28.00	33.37	34.26
	Semi-auto (%)	26.11	31.61	38.75	39.57	39.54		Semi-auto (%)	16.81	25.58	29.79	34.18	35.89
	Automatic (%)	25.38	35.62	37.89	38.05	39.39		Automatic (%)	12.98	21.80	25.78	35.78	36.48
M+A	Manual (%)	28.00	39.59	48.06	46.06	47.78	M+A	Manual (%)	11.24	24.57	29.82	35.99	42.71
	Semi-auto (%)	29.16	40.42	45.94	47.91	47.70		Semi-auto (%)	14.12	27.48	32.85	40.92	41.40
	Automatic (%)	30.23	37.77	44.47	46.38	47.79		Automatic (%)	15.68	31.67	37.42	34.95	39.44
B+M+A	Manual (%)	32.35	36.65	44.31	48.54	46.24	B+M+A	Manual (%)	9.29	19.24	28.03	30.97	37.09
	Semi-auto (%)	26.18	38.52	46.69	47.25	47.65		Semi-auto (%)	13.46	19.21	30.25	30.83	34.55
	Automatic (%)	30.41	37.11	46.25	48.25	47.50		Automatic (%)	12.91	12.18	24.00	27.48	35.67

B, M and A represent <Before>, <Middle> and <After>, respectively.

has been shown to be effective (Chapman *et al.*, 2001; Leroy *et al.*, 2003; Mutalik *et al.*, 2001; Sanchez-Graillet and Poesio, 2007). However, those learned patterns have rarely been evaluated to determine if they are qualified. In this study, we adapted the negative patterns obtained in our previous work (Kooi, 2008) because only qualified patterns should be used. This idea is based on the principle that a good pattern should cover sufficient correct sentences (Brin, 1998; Xiao *et al.*, 2003). In other words, if too many positive sentences match a negative pattern, the negative pattern should not be considered. The error rate of each negative pattern is calculated by the following equation:

$$\mathrm{PRate}(P_k) = \mathrm{Error}(P_k)/\mathrm{Correct}(P_k)$$

For negative pattern P_k, $\mathrm{Error}(P_k)$ means the number of positive sentences matched to negative pattern P_k, $\mathrm{Correct}(P_k)$ means the number of negative sentences matched to P_k, and $\mathrm{PRate}(P_k)$ means the ratio of incorrectly predicted sentences to correctly predicted ones. The threshold, $\mathrm{PRate}(P_k)$, is between 0 and 1.0. The results of this experiment, which are shown in Table 3, indicate that taking advantage of negative patterns in the <Middle> or <Before+Middle> vectors when the threshold is 0.1 can result in better performances of the gene network finding method, especially for TGene–TF-style sentences.

3.2.4 Experiment 4 Can we learn negative patterns automatically? Experiment 3 showed that using qualified negative patterns learned from manually marked sentences can improve the performance of the gene network finding method. Experiment 4 analyzed whether negative patterns can be learned automatically and whether negative patterns learned in this manner enhanced F-measures. Negative patterns are learned by excluding sentences matched to learned positive patterns. The comparison between using the gene network finding method with or without automatically learned negative patterns is shown in Table 4. After learning and filtering negative patterns, the precision corresponding to the <Middle>, <After> and <Before+Middle> vectors was slightly enhanced. However, the recall and F-measure values decreased.

The results of this experiment show that it is not possible to automatically learn negative patterns. We determined that published articles rarely include negative sentences in the abstracts. Therefore, given the small number of negative sentences available for training, automatically learning negative patterns is difficult.

4 CONCLUSION

This study introduced a pattern-learning method using bootstrapping. Based on our experimental results, we found that users do not need to manually mark sentences for pattern learning. In positive pattern training, data consisting of TF–TGene-style sentences resulted in better performance of the gene network finding method than data consisting of TGene–TF-style sentences. The difference may be due to the writing habits of biomedical researchers. The literature tends to record research achievements

Table 3. *F*-measure values of different thresholds of negative patterns

Threshold		0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
TGene–TF-style												
M	Manual (%)	39.78	50.28	48.92	40.42	36.54	33.28	28.28	24.02	24.12	24.14	24.14
	Semi-auto (%)	46.30	60.82	59.66	50.90	46.78	43.16	37.24	31.70	31.78	32.04	32.10
	Automatic (%)	46.20	60.82	59.54	50.66	46.66	43.02	37.12	31.56	31.66	31.88	31.88
A	Manual (%)	35.68	38.64	38.62	38.58	36.60	34.60	26.74	22.78	22.78	22.78	22.78
	Semi-auto (%)	41.30	45.14	45.16	45.02	43.14	41.04	33.32	28.40	28.40	28.46	28.46
	Automatic (%)	41.34	45.22	45.24	45.18	43.26	41.22	33.42	28.56	28.56	28.56	28.56
M+A	Manual (%)	43.54	55.84	54.42	44.62	40.68	37.50	31.82	27.16	27.22	27.36	27.36
	Semi-auto (%)	45.22	58.40	56.80	46.76	42.72	39.44	33.68	28.50	28.58	28.74	28.74
	Automatic (%)	45.88	59.64	58.14	48.08	44.20	40.86	34.82	29.32	29.38	29.66	29.66
TF–TGene-style												
M	Manual (%)	57.96	70.84	69.08	58.02	53.42	49.90	47.04	43.20	42.92	41.34	41.38
	Semi-auto (%)	56.04	70.20	68.94	58.80	55.16	51.64	48.52	43.70	43.08	41.72	41.74
	Automatic (%)	54.88	68.54	67.32	57.38	53.58	50.32	47.26	42.56	41.90	40.60	40.72
A	Manual (%)	52.00	55.00	55.00	53.00	50.00	50.00	47.92	44.00	43.00	42.00	41.00
	Semi-auto (%)	52.00	55.00	55.00	53.00	50.00	50.00	48.00	44.00	43.00	42.00	41.00
	Automatic (%)	52.00	55.00	55.00	53.00	50.00	50.00	48.00	44.00	43.00	42.00	41.00
M+A	Manual (%)	57.96	70.84	69.08	58.02	53.42	49.90	47.04	43.20	42.92	41.34	41.38
	Semi-auto (%)	59.48	72.60	70.70	58.98	54.32	50.86	47.58	44.00	43.26	42.04	42.06
	Automatic (%)	57.70	70.54	68.84	57.98	53.28	49.86	46.94	43.04	42.80	41.22	41.28

Table 4. Comparison of different vectors with or without filtering by learned negative patterns

Vector	Paper no.	TF–TGene						TGene–TF					
		R (%)	R (Filter) (%)	P (%)	P (Filter) (%)	F-M (%)	F-M (Filter) (%)	R (%)	R (Filter) (%)	P (%)	P (Filter) (%)	F-M (%)	F-M (Filter) (%)
< M>	5	92.41	91.78	40.15	40.21	55.97	55.91	90.40	89.70	31.00	31.05	45.94	45.88
	10	94.22	90.82	40.42	40.81	56.57	56.25	91.88	89.09	31.03	31.16	46.39	46.12
	15	93.55	88.58	40.37	41.03	56.39	55.92	92.01	86.75	30.94	31.08	46.30	45.70
< A>	5	67.40	66.73	42.04	42.02	51.78	51.56	63.28	62.91	31.12	31.11	41.48	41.38
	10	67.36	65.85	42.08	42.14	51.80	51.38	64.00	62.39	30.92	30.84	41.70	41.27
	15	67.31	64.25	42.12	42.37	51.81	51.04	63.97	62.73	30.94	30.94	41.70	41.44
<B+M>	5	84.72	84.00	46.08	46.13	59.56	59.41	73.71	73.49	38.83	38.83	43.53	43.50
	10	87.59	86.72	45.83	45.93	60.04	59.92	82.08	81.40	33.98	33.98	48.03	47.91
	15	86.53	84.39	46.31	46.62	60.21	59.88	82.50	80.76	33.95	33.88	48.07	47.68

(that is, positive results), and therefore authors are likely to describe their findings using active phrasing.

We also found that including only three to five abstracts for the initial training datasets generated acceptable results. This may be because many articles are published about the same TF and its specifically regulated genes; therefore, it is easy to collect such literature entries into an initial dataset. When a tuple is used for bootstrapping training, it is easy to find many sentences from which to learn new patterns.

Based on the retrieval effects from the seven vector styles in Experiment 1, the recall value of the <Middle> vector is the highest at 92.58% for TF–TGene-style sentences and 87.86% for TGene–TF-style sentences. The values of precision are 40.36% for TF–TGene-style sentences and 33.64% for TGene–TF-style sentences. This result shows that patterns generated by the <Middle> vector are the most representative. This may be because the keywords related to 'certain TF regulates this particular gene' almost always appears in the <Middle> vector for phrases such as 'activation of,' 'induction of,' 'transactivation of,' 'activate' and other similar phrases.

Experiment 2 indicated that the three methods for generating initial datasets do not result in obviously different performances of the gene network finding method. Performance using manually marked sentences is not always better than using automatically or semi-automatically marked sentences. In addition, the trained patterns were similar whether we manually marked results or used

a few abstracts to form the initial dataset and then adapted the bootstrapping method for pattern learning. Therefore, manually tagging the results does not greatly affect the performance of the gene network finding method. Rather, these results show that choosing the initial dataset does not necessarily involve manually marking sentences in advance or checking whether the literature contains positive sentences. Simply using an abstract that contains sentences with TF–TGene or TGene–TF relationships is sufficient for automatic training for positive patterns. Creating an initial dataset this way could reduce expenditures of time and human effort. Moreover, learning negative patterns can also help improve the performance of the gene network discovery method. However, the bootstrapping method of selecting sentences to include in the dataset is not suitable for training based on negative patterns. Data that consist of negative patterns must thus be manually marked.

In general, the bootstrapping method described in this study allows the gene network discovery method to effectively identify regulatory information regarding TFs and TGenes, thereby automatically finding positive patterns. Our results also indicate the <Before+Middle> and <Middle> vectors generate the most suitable positive patterns. Finally, our results indicate that if negative patterns are used to filter negative sentences, the precision of the gene network discovery method will be enhanced.

ACKNOWLEDGEMENTS

The authors would like to thank the Bioinformatics Center of National Cheng Kung University for their technical assistance.

Funding: National Science Council, Taiwan, grant (NSC 97-2627-B-006-004).

Conflict of Interest: none declared.

REFERENCES

Abney,S. (2002) Bootstrapping. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Pennsylvania, USA.

Bader,G.D. *et al.* (2001) BIND—the Biomolecular Interaction Network Database. *Nucleic Acids Res.*, **29**, 242–245.

Bairoch,A. and Apweiler,R. (2000) The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. *Nucleic Acids Res.*, **28**, 45–48.

Brin,S. (1998) Extracting patterns and relations from the world wide web. In *International Workshop on The World Wide Web and Databases*. Valencia, Spain, pp. 172–183.

Bui,Q.-C. *et al.* (2011) A hybrid approach to extract protein-protein interactions. *Bioinformatics*, **27**, 259–265.

Chang,D.S. and Choi,K.S. (2006) Incremental cue phrase learning and bootstrapping method for causality extraction using cue phrase and word pair probabilities. *Inform. Process. Manag.*, **42**, 662–678.

Chapman,W.W. *et al.* (2001) A simple algorithm for identifying negated findings and diseases in discharge summaries. *J. Biomed. Inform.*, **34**, 301–310.

Chowdhary,R. *et al.* (2009) Bayesian inference of protein–protein interactions from biological literature. *Bioinformatics*, **25**, 1536–1542.

Feng,H. and Chua,T.-S. (2003) A bootstrapping approach to annotating large image collection. In *Proceedings of the 5th ACM SIGMM international workshop on Multimedia information retrieval*. Berkeley, California, pp. 55–62.

Jang,H. *et al.* (2006) Finding the evidence for protein-protein interactions from PubMed abstracts. *Bioinformatics*, **22**, e220–e226.

Jensen,L.J. *et al.* (2006) Literature mining for the biologist: from information retrieval to biological discovery. *Nat. Rev. Genet.*, **7**, 119–129.

Kanehisa,M. *et al.* (2002) The KEGG databases at GenomeNet. *Nucleic Acids Res.*, **30**, 42–46.

Kooi,T.K. (2008) Using Text Mining to extract regulation between transcription factor and target gene. National Cheng Kung University Master Thesis, Taiwan.

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

Leroy,G. *et al.* (2003) A shallow parser based on closed-class words to capture relations in biomedical text. *J. Biomed. Inform.*, **36**, 145–158.

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

Mutalik,P.G. *et al.* (2001) Use of general-purpose negation detection to augment concept indexing of medical documents: a quantitative study using the UMLS. *J. Am. Med. Inform. Assoc.*, **8**, 598–609.

Narayanaswamy,M. *et al.* (2005) Beyond the clause: extraction of phosphorylation information from medline abstracts. *Bioinformatics*, **21**, I319–I327.

Niu,Y. *et al.* (2010) Evaluation of linguistic features useful in extraction of interactions from PubMed; application to annotating known, high-throughput and predicted interactions in I2D. *Bioinformatics*, **26**, 111–119.

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

Pyysalo,S. *et al.* (2008) Comparative analysis of five protein-protein interaction corpora. *BMC Bioinformatics*, **9**, Article No. S6.

Raychaudhuri,S. (2006) *Computational Text Analysis: For Functional Genomics and Bioinformatics*. Oxford University Press, Oxford, UK.

Sanchez-Graillet,O. and Poesio,M. (2007) Negation of protein-protein interaction: analysis and extraction. *Bioinformatics*, **23**, i424–i432.

Saric,J. *et al.* (2006) Extraction of regulatory gene/protein networks from Medline. *Bioinformatics*, **22**, 645–650.

Xenarios,I. *et al.* (2000) DIP: the database of interacting proteins. *Nucleic Acids Res.*, **28**, 289–291.

Xiao,J. *et al.* (2003) A global rule induction approach to information extraction. In *Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence*. Sacramento, California, USA.

Yangarber,R. *et al.* (2002) Unsupervised learning of generalized names. In *Proceedings of the 19th International Conference on Computational Linguistics*. Taipei, Taiwan.