IPSJ Transactions on Bioinformatics Vol. 3 91–94 (Nov. 2010)

Original Paper

A Dynamic Programming Re-ranking Approach to Enhance PPI Interactor Extraction

Po-Ting Lai[†1] and Richard Tzong-Han Tsai[†1]

Experimentally verified protein-protein interactions (PPIs) cannot be easily retrieved by researchers unless they are stored in PPI databases. The curation of such databases can be made faster by employing text-mining systems to identify genes which play the interactor role in PPIs and to map these genes to unique database identifiers, also referred to as the interactor normalization task (INT). Our previous INT system won first place in the BioCreAtIvE II.5 INT challenge by exploiting the different characteristics of individual paper sections to guide gene normalization (GN) and using a support-vector-machine (SVM)-based ranking procedure. The best AUC achieved by our original system was 0.435 in the BioCreAtIvE II.5 INT offline challenge. After employing the proposed re-ranking algorithm, we have been able to improve our system's AUC to 0.447. In this paper, we present a new relational re-ranking algorithm that considers the associations among identifiers to further improve INT ranking results.

1. Introduction

Protein-protein interaction (PPI) has been studied extensively because of their crucial role in elucidating signal pathways, controlling central biological processes such as transcription factors involved in cell division and DNA transcription (Zhang, et al., 2009), and their implications in a range of human diseases including cancer and neurodegeneration (Liao, et al., 2009; Thalappilly, et al., 2008). To provide efficient widespread access to PPIs information, some organizations have begun collecting structured interaction annotation in public databases.

In 2009, as part of the BioCreAtIvE II.5 challenge (Krallinger, et al., 2009), CNIO announced the interactor normalization task (INT), a new task to map interactor proteins in well-formed full-text articles to UniProt identifiers, and to rank these identifiers according to their probability of being interactors.

INT can be divided into two subtasks: gene normalization (GN) (Morgan, et al., 2008) and interactor protein ranking. GN determines the unique database identifiers of genes and proteins mentioned in scientific literature. The second subtask of INT is to rank the normalized identifiers according to their probability of being interactors. To rank each normalized identifier, most approaches consider two factors: the confidence of the normalized identifier and its interactor likelihood score.

We introduce a relational re-ranking algorithm that considers co-occurrence among identifiers. According to (Jenssen, et al., 2001; Stephens, et al., 2001), co-mentioned genes are usually related in some way. If two gene names frequently occur alongside each other in the same sentence in an article, they are likely to have an association and influence each other's rank. Take a low-ranked interactor mentioned only twice in an article for example. If both mentions happen to be alongside the highest-ranked interactor in the article, then the low-ranked interactor's rank should be significantly boosted. Using a greedy computational approach, the re-ranking procedure requires large amounts of computer resources and time to calculate each identifier's rank simultaneously and find the best ranked list. Therefore, to maximize computational efficiency, we implemented our re-ranking algorithm using dynamic programming.

2. Methods

The well-formed full-text article is pre-processed to resolve the conjunction problems presented by Ref. 1). After pre-processing, the GN procedure is executed.

2.1 Interactor Ranking

We employed a multi-stage GN procedure, which was developed in our previous work 2). For interactor ranking, each candidate identifier from GN procedure is ranked by a SVM classifier. For each identifier, the corresponding context information such as the frequency which the identifier appears in the entire article and the sections where an identifier appears is used to extract features.

2.2 Re-ranking Algorithm

In order to further refine the ranking results, we created a re-ranking algorithm that takes into consideration the rank of an identifier and the genes that co-occur

†1 Department of Computer Science & Engineerining, Yuan Ze University, Taiwan

© 2010 Information Processing Society of Japan

alongside all instances of that identifier in a paper.

Our re-ranking algorithm accepts a ranked result list of n identifiers, R. In the following paragraphs, we describe the functions that are used in our re-ranking algorithm. **Table 1** defines the notations used.

$\mathit{association}(x, y)$: The function measures the association between two identifiers x and y within a given piece of text-sentence, paragraph, or entire document-and returns an association score. We use an unsupervised approach based on mutual information (MI) [3] to measure the association.

$\mathit{newrank}(x, R)$: For the identifier x, the procedure uses the *association* function to measure the association between x and x_i, for all $x_i \in R, x_i \neq x$, and returns a new ranked list NR_x.

For an identifier x whose rank is i, x can only determine a new ranked list from rank i to n. Take a ranked list R of four identifiers w, x, y, and z for example. For the convenience of explanation, we use w_1, x_2, y_3, and z_4 to represent that w, x, y, z are ranked first, second, third, and fourth, respectively. After employing the *newrank* procedure, the rank 1 identifier, w_1, can determine a new ranked list, NR_w, which must be one of the following six possible cases: $[w_1, x_2, y_3, z_4]$, $[w_1, x_2, z_3, y_4]$, $[w_1, y_2, x_3, z_4]$, $[w_1, y_2, z_3, x_4]$, $[w_1, z_2, x_3, y_4]$ or $[w_1, z_2, y_3, x_4]$ if the scores of the *association* procedure for (w, x), (w, y), and (w, z) are different and they co-occurred in a sentence. Consider another case: if two of the scores were the same, e.g., $\mathit{association}(w, y) = \mathit{association}(w, z)$, NR_w could only be one of the following two cases: $[w_1, x_2, y_3, z_3]$ or $[w_1, y_2, z_2, x_3]$. For x_2, if $\mathit{association}(x, w) \neq \mathit{association}(x, y)$ and x do not co-occur with z, x only determines a ranked list from either $[x_2, w_3, y_4]$ and $[x_2, y_3, w_4]$.

$\mathit{whodetermine}(x, r, NR)$: For an identifier x_r whose new rank r is determined by more than one identifiers, the function returns the highest ranked identifier.

Table 1 Notation definition.

Notation	Description
R	The ranked list generated by the SVM-based ranking procedure.
NR	A set of ranked lists which were determined by all identifiers of R.
RR	A re-ranked list.
NR_x	The ranked list determined by the identifier x; $NR_x \in NR$.
x_i	An identifier x whose rank is i. x can be any lowercase letters in italics.
$NR_x[r]$	All identifiers in rank r of NR_x.

For example, assumed that the identifier w_1 determines a ranked list $NR_w = [w_1, x_2, y_3, z_4]$ and the identifier x_2 determines $NR_x = [x_2, y_3, w_4]$, the identifier y will be ranked third. In this case, $\mathit{whodetermine}\left(y, 3, NR_{[w,x]}\right)$, will return w since w's rank (1) is higher than x's rank (2).

$\mathit{aggregate}(y, r, NR)$: Given an identifier y, rank r and a set of ranked lists NR, the function returns an agreement score based on the following equation:

$$\mathit{aggregate}(y, r, NR) = \frac{\sum_x \mathit{agree}(y, r, NR_x)}{\sum_x \sum_{r'} \mathit{agree}(y, r', NR_x)}$$

where x is an identifier in R, and $r' \in \{1, 2, 3, \ldots, n\}$

In the equation, $\mathit{agree}(y, r, NR_x)$ determines whether the identifier y is in rank r of NR_x (return 1) or not (return 0). Therefore, $\sum_x \sum_{r'} \mathit{agree}(y, r', NR_x)$ is the total number of the identifier y appeared in all ranks of NR. $\mathit{svmaccuracy}(x)$: Given an identifier x, the function returns the INT accuracy of x's rank in our SVM-based ranking. The accuracy is calculated based on a three-fold cross validation carried out on the training set. The score function for x to be re-ranked r is defined as follows:

$$\begin{aligned} &\mathit{score}(x, r, NR) \\ &\quad = \mathit{svmaccuracy}(x) \times \mathit{svmaccuracy}(\mathit{whodetermine}(x, r, NR)) \\ &\qquad \times \mathit{aggregate}(x, r, NR) \end{aligned} \tag{1}$$

Given a re-ranked list, $RR = [w_1, \ldots, z_n]$, the score for RR is defined as follows:

$$\mathit{overallscore}(RR, NR) = \prod_{r=1}^{n} \mathit{score}(RR[r], r, NR)$$

We can now formulate the re-ranking problem as an optimization problem that maximizes the overall scores over all possible rank orders:

$$\mathit{argmax}_{RR} \cdot \mathit{overallscore}(RR, NR) \tag{2}$$

2.3 Discussion of Optimization Problem

If the duplication of identifiers in a ranked list is permitted, the optimal ranked list can be directly found by choosing the identifier with the highest score value (Eq. (1)) for each rank. However, a legal ranked list cannot have any duplicates. To avoid duplication, we add a constraint on the score funciton: when estimating $\mathit{overallscore}(x, r, NR)$, if the identifier x has been de-

© 2010 Information Processing Society of Japan

Table 2 The newrank_candidates data structure.

newrank_candidates
A dictionary maps the rank (an integer) to a list of tuples: (id, score, overallscore, from). The dictionary's keys are the ranks in the re-ranked list. Values are lists contained tuples in which the stored information can be extracted by the following attributes:
Attributes
tuple.id: the identifier
tuple.score: the score of tuple.id
tuple.overallscore: the overall score of the ranked list after considering tuple.id
tuple.from: the identifier in the previous rank, which leads the optimal tuple.overallscore
Methods
nc[*key*]: Return the list of tuples in *nc* with key *key*.
nc[*key*][*i*]: Return the *i*th tuple in the list in *nc* with key *key*.

termined in the previous rank k, the $score\,(x, k, NR)$ function must return 0 (i.e., $overallscore\,(x, k, NR)$ equals 0). For example, consider two possible ranked lists: $RL1 = [x_1, w_2, y_3]$, and $RL2 = [x_1, y_2, z_3]$. Assume that $overallscore\,(RL1, NR) > overallscore\,(RL2, NR)$, and we now want to determine the value of $overallscore$ function when w is in rank 4, then $RL1$'s $overallscore$ becomes 0 because $score\,(w, 2, NR) = 0$ and $RL2$'s score stays the same. Therefore, even though in rank 3, $RL1$'s $overallscore$ is higher than $RL2$, the algorithm will not choose $RL1$ as an optimal sub-ranking when considering w in rank 4. Unfortunately, the duplication constraint increases the computational complexity of finding the optimal ranked list. In order to find the optimal RL and avoid computational overhead, we propose a dynamic-programming-based algorithm.

2.4 Dynamic Programming Algorithm

The re-ranking algorithm starts by generating all possible ranked lists for each identifier in R. For each rank, the corresponding identifiers and their scores are calculated and stored in a dictionary-like data structure, *newrank_candidates* (lines 3–5). **Table 2** shows the data structure and its attributes.

The algorithm computes the optimal overall score for each identifier in each rank, and finds the maximum overall score in $newrank\_candidates\,[i-1]$ in which the identifier, $ID\,(i, j)$, does not appear among rank 1 to rank $i-1$. In the following formula, $newrank\_candidates\ [i]\,[j]$.overallscore is shorted to $OverallScore\,(i, j)$, which is the optimal overall score from rank 1 to rank i when rank i's jth candidate is placed at rank i. $newrank\_candidates\,[i]\,[j]$.identifier is shorted to $ID\,(i, j)$, which stands for rank's jth candidate. The score can be recursively calculated as follows:

$$OverallScore\,(i, j)$$
$$= \begin{cases} score\,(ID\,(i, j)\,, i, NR) \times \max \left\{ \begin{array}{l} Overallscore\,(i-1, 0) \\ \vdots \\ Overallscore\,(i-1, k-1) \end{array} \right\} \text{if } i > 2 \\ score\,(ID\,(1, 0)\,, 1, NR)\, if\, i = 1, j = 0 \end{cases}$$

where k is the number of tuples in the $newrank\_candidates\,[i-1]$. Then it calculates the score. After all, the optimal ranking is reconstructed by tracing the "from" attribute of the tuple in the last rank with maximal overall score (the *optimal_end*) until the value of "from" is None.

3. Results

3.1 Dataset and Evaluation Metrics

The BioCreAtIvE II.5 Elsevier corpus (Hirschman, et al., 2005), which contains 1,190 journal articles selected mainly from FEBS Letters, is used for evaluation. The area under curve (AUC) of the interpolated precision/recall (iP/R) curve used in the BioCreAtIvE II.5 challenge is used to evaluate the proposed approach. The AUC of the iP/R-curve can be found in the BioCreAtIvE II.5 web site:

http://www.biocreative.org/tasks/biocreative-ii5/
biocreative-ii5-evaluation/

3.2 INT Test Set Performance

We first report the BioCreAtIvE II.5 INT evaluation results in **Fig. 1**, in which top three teams are showed. Our AUC performance is significantly higher. The AUC (IASL-IISR) achieved by our system was 0.435 outperforms the second best team's top score (Hakenberg, et al.) by 4.125%. In IASL-IISR+Re-rank, the proposed re-ranking algorithm is added. The Freq is a baseline method which ranks all identifiers according to their frequency is employed. If two or more identifiers have the same frequency, three criteria are employed sequentially to rank them: (1) matches the largest number of our PPI patterns (2) highest frequently in the Results sections (3) mentioned earliest in the article. Lastly,

© 2010 Information Processing Society of Japan

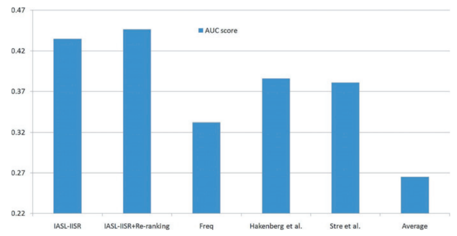

Fig. 1 The results of different ranking approach in INT.

Fig. 1 also shows the average AUC score of all BioCreAtIvE II.5 INT participants (Average).

As shown in Fig. 1, after employing our re-ranking algorithm, AUC performance increases by 1.16%. According to our analysis, before re-ranking, identifiers whose feature values rarely appear in the training set are often incorrectly ranked because their feature values are underweighted in the ranking model. However, if these identifiers co-occur with higher-ranked identifiers whose feature values frequently appear, our re-ranking algorithm is very likely to increase their ranks. This results in the improved AUC score.

4. Conclusions

In this paper, we have proposed a relational re-ranking algorithm that considers the associations among identifiers to further improve INT performance. We formulated the re-ranking problem as an optimization problem and solved it by using dynamic programming to reduce the computational complexity.

The highest AUC achieved by our system is 0.435 which is the highest in the BioCreAtIvE II.5 INT challenge. By employing the re-ranking algorithm, the AUC can be further improved to 0.447. In the future, we hope to integrate advanced parsing technologies which can extract deeper semantic information from full-text articles into our INT system.

References

1) William A. Baumgartner, J., Lu, Z., Johnson, H.L., Caporaso, J.G., Paquette, J., Lindemann, A., White, E.K., Medvedeva, O., Cohen, K.B. and Hunter, L.: An integrated approach to concept recognition in biomedical text, *Proc. Second BioCreative Challenge Evaluation Workshop*, pp.257–271 (2007).

2) Dai, H.-J., Lai, P.-T. and Tsai, R.T.-H.: Multistage Gene Normalization and SVM-Based Ranking for Protein Interactor Extraction in Full-Text Articles, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, Vol.7, No.3, pp.412–420 (2010).

3) Fano, R.M.: *Transmission of Information: A Statistical Theory of Communications*, MIT Press, Cambridge, MA (1961).

(Received July 14, 2010)
(Accepted August 26, 2010)
(Released November 11, 2010)

(Communicated by *Tatsuya Akutsu*)

Po-Ting Lai studied for his B.S. degree in Department of Computer Science and Engineering from Yuan Ze University in Taiwan. Since 2009, he has been an intern student at the Academia Sinica. His research interests include information retrieval, natural language processing, biological literature mining and Software Engineering.

Richard Tzong-Han Tsai received his B.S. degree in Computer Science and Information Engineering from National Taiwan University, Taipei, Taiwan, in 1997, the M.S. degree in Computer Science and Information Engineering from National Taiwan University in 1999, and, respectively, the Ph.D. degree in Computer Science and Information Engineering from National Taiwan University in 2006. He was a postdoctoral fellow at Academia Sinica from 2006 to 2007. He is now an assistant professor of Department of Computer Science and Engineering, Yuan Ze University, Zhongli, Taiwan. His research areas are natural language processing, cross-language information retrieval, biomedical literature mining, and information services on mobile devices.

© 2010 Information Processing Society of Japan