Biological Question Answering with Syntactic and Semantic Feature Matching and an Improved Mean Reciprocal Ranking Measurement

Ryan T.K. Lin[1], Justin Liang-Te Chiu[13], Hong-Jei Dai[14], Min-Yuh Day[1], Richard Tzong-Han Tsai[2], and Wen-Lian Hsu[1].
[1]Institute of Information Science, Academia Sinica, Taipei, Taiwan, R.O.C.
[2]Dept. of Computer Science & Engineering, Yuan Ze Univ., Taoyuan, Taiwan, R.O.C.
[3]Dept. of Computer Science & Information Engineering, National Taiwan Univ., Taipei, Taiwan, R.O.C.
[4]Dept. of Computer Science, National Tsing-Hua Univ., Hsinchu, Taiwan, R.O.C.

Abstract

Specific information on biomolecular events such as protein-protein and gene-protein interactions is essential for molecular biology researchers. However, the results derived by current keyword-based information retrieval engine contain a great deal of noisy information, which forces biologists to use a combination of several keywords to locate information. To resolve this problem,, we propose a question answering (QA) system that offers more efficient and user-friendly ways to retrieve desired information. In addition, QA system measurements may suffer from the same score problem, so the evaluation of a QA system may be unfair. An improved mean reciprocal rank (MRR) measurement, mean average reciprocal rank (MARR), and an efficient formula to reduce the computational complexity of the MARR are proposed to address the same score problem. With our syntactic and semantic features, our system achieves a Top-1 MARR of 74.11% and Top-5 MARR of 76.68%. Compared to the baseline system, Top-1 MARR and Top-5 MARR increase by 16.17% and 18.61% respectively.

1. Introduction

Molecular biologists are primarily interested in relevant molecular pathways and underlying mechanisms [1]. Since molecular biology is a rapidly developing and changing field, researchers must be able to search recently published literature efficiently and effectively. Currently, most researchers use keyword-based search engines such as PubMed and Google [2]. However, with the overwhelming amount of new biomedical literature being published and the increasing complexity of molecular pathway descriptions, it is becoming much harder to find specific and relevant information about molecular interactions using these tools.

Some studies need to identify entities that are strongly associated with query terms. The most studied type of entity is people (also known as expert search), which has been addressed by [3, 4]. In a previous work [5], we proposed the BESearch system to assist users in searching for the entities participating in a specific molecular event.

In this paper, we propose a QA system that can answer questions about biomolecular events, such as gene and protein interactions. The answers to such questions usually consist of short pieces of information such as times, locations, as well as biomedical named entities (NEs) like protein, DNA, RNA, or cell. Accordingly, our system only focuses on factoid questions, to which the answers are NEs.

1.1. System Workflow

The proposed QA system is comprised of four components, namely Question Processing, Passage Retrieval, Candidate Extraction and Feature Generation, and Answer Ranking, which we describe in detail in the following sub-sections.

1.1.1. Question Processing

Question processing transforms natural language questions into search keywords and extracts features for answer ranking. In our work, question processing involves five steps: named entity recognition (NER) [6-9], semantic role labeling (SRL) [10, 11], question classification, and query modification.

The NER step extracts named entities (NEs), such as protein and gene names, from the original question. Then, the SRL step extracts predicates (e.g., a verb) and corresponding arguments (e.g., noun phrases) from the question. Both the NEs and the SRL information will be transformed into features and used by the answer ranking module.

In the question classification step, hand-crafted patterns are used to identify the target NE type, including protein, cell, DNA, and RNA, requested by the question (i.e., the answer's NE type). The classified NE type is then sent to the ranking module, which filters out unmatched answer candidates. Each word in the remaining phrases is then examined and will be eliminated if it appears on the stop

978-1-4244-2660-7/08/$25.00 ©2008 IEEE

word list. The remaining phrase segments are sent to the passage retrieval module as keywords for a Google search.

Query modification is used to improve recall for queries where Google returns pages. First, using WordNet [12] and Longman's dictionary [13], we expand queries to generate a list of synonyms and other tenses for the main verb of the question. Then, we repeat the web search with the expanded query terms. If there are still no returned pages, we begin removing keywords to improve recall.

1.1.2. Passage Retrieval

The passage retrieval module is a Google-interfacing program that can send queries to Google and retrieve a collection of web pages, which we then sent to the answer extraction module. In the passage retrieval stage, we only retrieve pages from Google's index of the PubMed database on the NCBI website to avoid unnecessary noise.

1.1.3. Candidate Extraction and Feature Generation

In this stage, we use two extraction technologies, NER and SRL, to extract candidate NEs and their corresponding features. NER extracts named entities for answer candidates, and generates features to help match a query with passages containing relevant NEs. We employ the GENIA Tagger [9] to identify four types of NE: protein, DNA, RNA, and cell. Biomolecular events in nominal form (e.g., protein expressions), in which the relevant NEs are involved, are also extracted. In our system, each candidate is output with the sentence containing it, and the sentence is treated as its supporting evidence.

Table 1. Argument Types and Their Descriptions

Type	Description
Arg0	agent
Arg1	direct object / theme / patient
Arg2-5	not fixed
ArgM-NEG	negation marker
ArgM-LOC	location
ArgM-TMP	time
ArgM-MNR	manner
ArgM-EXT	extent
ArgM-ADV	general-purpose
ArgM-PNC	purpose
ArgM-CAU	cause
ArgM-DIR	direction
ArgM-DIS	discourse connectives
ArgM-MOD	modal verb
ArgM-REC	reflexives and reciprocals
ArgM-PRD	marks of secondary predication

We developed an SRL component [11] (F-score: 84.76%) to generate semantic features for answer ranking. SRL can recognize the predicate of a sentence and its corresponding argument phrases, such as the agent, recipient, and location. The argument types recognized by our SRL component, and their descriptions are listed in Table 1.

The SRL step also verifies whether answer candidates extracted by our NER component are the expected type. By comparing a candidate's semantic argument type with the expected type, we can eliminate many incorrect candidates and improve the overall accuracy. All the entity candidates along with their features are delivered to the answer ranking module after the extraction step has been completed.

1.1.4. Answer Ranking

Each NE extracted in the previous step is treated as an answer candidate, and the answer ranking module is responsible for calculating each candidate’s score. We employ a linear model to calculate a candidate’s score based on its features. To estimate feature weights more precisely, we propose a supervised weight tuning procedure, which we describe in the next section.

2. Method

2.1. Our Linear Answer Ranking Model

To calculate an answer candidate’s score, the proposed ranking module uses a linear function (combination of features) to calculate the weighted sum of the candidate's features. Each candidate *c* identified in the candidate extraction step is represented as a binary feature vector $\mathbf{f}_c$. The *i*th dimension of $\mathbf{f}_c$ (f_{c_i}) indicates whether *c* meets the criterion of the binary feature function f_i, which has a corresponding weight w_i. Therefore, the score of a candidate *c* is calculated as follows:

$$\text{score}(c) = \mathbf{f}_c \bullet \mathbf{w} = \sum_i f_{c_i} w_i$$

where w is the weight vector that corresponds to $\mathbf{f}_c$.

2.2. Tuning Feature Weights

To improve the ranking results, we apply a weight tuning procedure. The procedure first generates all possible weight combinations of the eight features, whose weights have integer values between 1 and 10. This yields 108 different combinations. To avoid generating too many weight vectors with the same score, we use the top-5 MARR as the measurement of each weight vector, instead of the top-1 accuracy.

Next, for each of the top 20 weight vectors, new vectors are created by changing the weights upward or downward by 0.5 or by leaving the weight of a given vector unchanged. This produces 3^n-1 new vectors (*n* denotes the dimension number). The process is then repeated with an upward or downward change of 0.25, and the algorithm is iterated repeatedly until the weight decrement reaches 0.125. We take the combination of eight weights with the highest top-5 MARR as the final feature weight set.

2.3. Features

Our QA system employs 8 features: Verb_Match (f_{VM}), Argument_Match (f_{ARGM}), NE_Match (f_{NEM}), NE_Similarity (f_{NES}), KeyWord_Similarity (f_{KWS}), Argument_Similarity (f_{ARGS}), Consecutive_Word_Match (f_{CWM}), and Google_Reciprocal_Rank (f_{GRR}). We denote the answer candidate as *c*, the query as *q*, the sentence containing *c* as *s*, and the page containing *s* as *p*. The eight feature types are defined as follows:

$$f_{\text{VM}}(c)=\begin{cases}1 & \text{if } c\text{'s verb matches } q\text{'s main verb}\\ 0 & \text{otherwise}\end{cases}$$

$$f_{\text{ARGM}}(c)=\begin{cases}1 & \text{if } c\text{'s semantic role matches the target role}\\ 0 & \text{otherwise}\end{cases}$$

$$f_{\text{NEM}}(c)=\begin{cases}1 & \text{if } c\text{'s NE type matches the target type}\\ 0 & \text{otherwise}\end{cases}$$

$$f_{\text{CWM}}(q,s)=\frac{\text{\# of consecutive words in } s \text{ that match consecutive words in } q}{\text{\# of keywords in } q}$$

$$f_{\text{NES}}(c,q,s)=\frac{\text{\# of NEs in } s \text{ that match NEs in } q}{\text{\# of NEs in } q}$$

$$f_{\text{KWS}}(c,q,s)=\frac{\text{\# of keywords in } s \text{ that match keywords in } q}{\text{\# of keywords in } q}$$

$$f_{\text{ARGS}}(c,q,s)=\frac{k}{\text{\# args in } q \text{ exclusing the target argument}}$$

k = # args in s that match arguments in q excluding the target argument

$$f_{\text{GRR}}(p)=\text{the Google reciprocal rank of } p$$

The first three features listed above are binary features. The next three represent the similarity between *q* and *s*. Another denotes adjacent keywords from *s* match that of *q*; and the last feature is *p*'s Google reciprocal rank. The values of the last four features range between 0 and 1.

2.4. An Improved Evaluation Measurement

We consider two commonly used measurements of a QA system's performance: the top-1 accuracy and the top-5 mean reciprocal rank (MRR). For a question set Q, the, top-1 accuracy reports the average accuracy of the top-1 answers for all the questions. It is defined as follows:

top-1 accuracy = # of correct top-1 answers / |Q|

The top-5 mean reciprocal rank (MRR) [14] is calculated as follows:

$$r_i=\begin{cases}rank(q_i), & rank(q_i)\le 5\\ \infty, & rank(q_i)>5\end{cases}$$

$$RR(q_i)=\frac{1}{r_i}$$

$$MRR(Q)=\frac{\sum_{q\in Q} RR(q)}{|Q|}$$

where q_i is the *i*th question; $rank(q_i)$ is the rank of the first correct answer in the list of answer candidates for q_i. In addition, top-1 MRR uses 1 in place of 5 in above first formula.

However, there may be many answer candidates with the same score and all in the leading five places. This results in multiple answer candidate sequences with different RR score for a question but all based on the same score. Randomly selecting a candidate sequence and calculating its RR to represent all sequences may solve this problem. However, the true system performance may be overestimated or underestimated. Therefore, we propose a new measurement-the average reciprocal rank (ARR). The ARR score is the average of all possible sequences' MRR scores, which is defined as follows:

$$r_i(s)=\begin{cases}rank(q_i), & rank(q_i)\le 5\\ \infty, & rank(q_i)>5\end{cases}$$

$$ARR(q_i)=\frac{\sum_{s\in S}\frac{1}{r_i(s)}}{|S|}$$

$$MARR(Q)=\frac{\sum_{q\in Q} ARR(q)}{|Q|}$$

where *S* is the set of all possible sequences including all the answer candidates for q_i; and *s* is any sequence in *S*.

To further explain the differences between MRR and MARR, we use the following example to compare the result of the RR and ARR method. Suppose we have three answer candidates A, B, and C; and they all have the highest score. However, only A is the correct answer candidate. Using the RR measurement, we could get different answer candidate sequences for A, B, and C. There are 3! = 6 sequences, as shown in Table 2.

Table 2. All Possible Sequences

Sequence	Top 1	Top 2	Top 3	RR
1	**A**	B	C	1
2	**A**	C	B	1
3	B	**A**	C	1/2
4	B	C	**A**	1/3
5	C	**A**	B	1/2
6	C	B	**A**	1/3

The RR score for each sequence is listed in the last column. In this case, the QA system has one sixth probability to get one of these sequences. Consequently, each run of multiple experiments may produce different evaluation results.

However, by using ARR, we can sum up all RR scores and divide the result by *S*. Therefore, we have:

$$\text{ARR}(q_i) = (1+1+1/2+1/3+1/2+1/3)/6 = 11/18$$

which is a fixed value. In contrast with RR method, ARR can evaluate the QA systems' performance precisely.

However, the above ARR method has two limitations. (1) If the value of |*S*| is very large, calculating the ARR score directly is inefficient. For example, if 170 answer candidates have the highest score, we need to expand totally 170! permutations. (2) Technologically, there are no numerical data types can fit such large value.

To solve the above problems in calculating the ARR score, we employ the following efficient formula:

$$ARR(q_i) = \sum_{r}^{\min(r+m-1,5)} \frac{n(m-r)!(m-n)!}{r(m-n-r+1)!m!}$$

where *m* represents the number of answer candidates with the same score; *n* denotes the number of correct answer candidates with the same score; and *r* indicates the number of answer candidates with the same score are calculated from which position.

Using the above formula, we calculate the ARR score as follows (r=1, m=3, n=1):

$$\sum_{r=1}^{3} \frac{1(3-r)!(3-1)!}{r(3-1-r+1)!3!} = 11/18 \ .$$

Here, the score is equal to that of previous ARR method. It provides a convenient way of calculating a large number of answer candidates with the same score.

3. Results

3.1. Dataset

To the best of our knowledge, there are no well-established online factoid QA systems dedicated to the biomolecular domain. Hence, it is difficult to obtain a representative set of user queries for use as a benchmark. To create a question set, biologists in our laboratory referred to the TREC Genomics Track [15] to choose appropriate abstracts and generate candidate questions.

An independent committee composed of several other biologists then selected questions from among the generated candidates. The answer types of 400 biomolecular event questions cover four NE classes, namely protein, DNA, RNA, and cell (including cell line and cell type). Furthermore, each question is based on one of 30 common biomolecular verbs described in [11].

The following two sentences are examples of the selected questions:

1. [$_{\text{R-Arg0}}$Which protein] [$_{\text{predicate}}$increases] [$_{\text{Arg1}}$levels of active nuclear NF-kappa B complex]?
2. [$_{\text{R-AM-LOC}}$In which type of cell] does [$_{\text{Arg0}}$human immunodeficiency virus type 1 Nef protein] [$_{\text{predicate}}$inhibit] [$_{\text{Arg1}}$NF-kappa B induction]?

3.2. Experiment Design

We designed several experiments to find the best settings for our QA system. In the following experiments, we cache the query results of all configurations to eliminate the influence of updates to Google's index. We take f_{VM}, f_{NES}, f_{NEM}, and f_{KWS} as the baseline system.

First, we test the effectiveness of our query-modification methods. To assess the contribution of each feature, we compare its related features by adding f_{ARGM}, f_{ARGS}, f_{CWM} and f_{GRR} to the baseline configuration individually. The five configurations are Baseline, ARGM, ARGS, CWM and GRR. Furthermore, we incorporate all the features into a sixth configuration, ALL. Finally, using the development set, we determine the best combination of all eight features, evaluate it on the test set, and then we compare the results with development-set performance.

3.3. Experiment Results

Table 3. Comparison of Different Features

Config.	f_{ARGM}	f_{ARGS}	f_{CWM}	f_{GRR}	top-1 MARR (%)	top-5 MARR (%)
Baseline					57.94	58.07
ARGM	+				63.79	65.37
ARGS		+			62.46	64.16
CWM			+		64.47	66.08
GRR				+	59.71	61.38
ALL	+	+	+	+	74.11	76.68

Table 3 shows the performance comparison of f_{ARGM}, f_{ARGS}, f_{CWM}, f_{GRR} and ALL. When applied individually, f_{CWM} and f_{ARGM} are the most effective features. f_{ARGS} can also improve the performance when used alone or with other features. By incorporating all features, the top-1 MARR and top-5 MARR results are 74.11% and 76.68%, respectively. The actual weights of the ALL configuration determined by our tuning procedure are listed in Table 4.

Table 4. Actual Tuned Weights

Feature	f_{VM}	f_{NEM}	f_{NES}	f_{KWS}	f_{ARGM}	f_{ARGS}	f_{CWM}	f_{GRR}
Weight	1.0	7.8	2.5	3.0	10.8	1.0	7.7	1.0

4. Discussion

In this section, we provide some examples to demonstrate the effectiveness of each proposed syntactic or semantic feature and describe an additional experiment that compares MRR with MARR.

4.1. The Effects of Using f_{CWM}

Question:
Which protein inhibits the synthesis of Ig mRNA?
Answer:

Baseline	CWM	Passage
16	20.17	These findings demonstrate that [***TGF-beta***] decreases B lymphocyte Ig secretion by inhibiting the synthesis of Ig mRNA and inhibiting the switch from the membrane form to the secreted forms of mu and gamma mRNA .
16	18.50	Transforming growth factor-beta suppresses [***human B lymphocyte Ig***] production by inhibiting synthesis and the switch from the membrane form to the secreted form of Ig mRNA .

The f_{CWM} feature reinforces the keyword match feature by considering consecutive matches between questions and passages. The experiment result shows that f_{CWM} increases accuracy by approximately 7%. We use the above example to demonstrate f_{CWM}'s effectiveness. The first and second columns show the scores of the Baseline and CWM configurations, respectively. The third column shows answer candidates (the phrases in brackets) and their passages. In the baseline configuration, both candidates achieve a score of 16. However, after adding f_{CWM}, the first candidate achieves a better score than the second. This is because the first passage's CWM value is higher than that of the second. In the first passage, there is a consecutive five-word match, "the synthesis of Ig mRNA", which is underlined. In the second passage, the length of the consecutive match "Ig mRNA" is only three words. This example demonstrates that f_{CWM} is very useful for disambiguating candidates with similar contexts.

4.2. The Effects of Using *f*ARGM

Here we give an example to illustrate how *f*ARGM significantly enhances the performance of the top-1 accuracy and top-5 MARR. In the question, "Which protein interacts with the alpha subunit of TFIIA?", we tag the semantic roles in addition to named entity tagging, so the question becomes:

[$_{R\text{-}Arg0}$Which protein] [$_{predicate}$interacts] [$_{Arg1}$with the alpha subunit of TFIIA]?

Because of SRL, our QA system can determine the target role (Arg0) in addition to the answer type (protein); hence, it can, search for an Arg0 protein. The following is a sample answer text.

First, [$_{Arg0}$Tax] was found to [$_{predicate}$interact] [$_{Arg1}$with the 35-kDa (alpha) subunit of TFIIA] [$_{ArgM\text{-}LOC}$in the yeast two-hybrid interaction system]. Our QA system immediately identifies the "Tax" protein since it is an Arg0 protein.

4.3. Comparing MRR with MARR

To compare MARR with MRR, we conducted 30 additional experiments on the ALL configuration described in Section 3.4, and using the same dataset. Figure 1 shows the results. We observe that

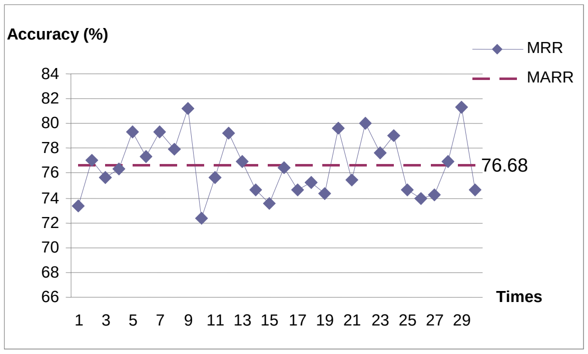

Figure 1. Comparison between MRR and MARR

MARR yields a stable evaluation result (Accuracy: 76.68%), while MRR arbitrarily changes in response to each experiment. Figure 1 demonstrate that the proposed method can evaluate any QA system precisely and avoid the same score problem that characterizes MRR.

5. Conclusion

We have presented a QA system that provides biologists with another way to obtain the information they need. After combining all the syntactic and semantic features, the performance of our QA system achieves 74.11% top-1 MARR and 76.68% top-5 MARR.

Although our system considers eight features with different weights, it still experiences the same score problem that affects widely used measurement methods. To resolve the problem, we propose a new measurement—the average reciprocal rank (ARR) which is the average of all possible RR score sequences. However, expanding all permutations to calculate the ARR is inefficient, so we further proposed an efficient formula and shown the equality of the results.

In our future work we plan to: (1) increase the variety of answer types by including more NE classes, such as diseases and viruses; (2) expand our corpus sources from short abstracts to full papers or other authoritative biomedical digital libraries; and (3) link the extracted answers directly to other databases or resources to provide biologists with related information in a fast and efficient manner.

6. Acknowledgement

This research was supported in part by the National Science Council under Center of Excellence Grant NSC 96-2752-E-001-001-PAE and thematic program of Academia Sinica under Grant AS95ASIA02.

7. References

[1] K. B. Cohen and L. Hunter, "Natural Language Processing and Systems Biology," *Artificial Intelligence Methods and Tools for Systems Biology*, 2005.

[2] X. Yang, J. S. Guodong Zhou, and C. L. Tan, "Improving Noun Phrase Coreference Resolution by Matching Strings," *Natural Language Processing-IJCNLP 2004: First International Joint Conference.*, pp. 22-31, 2005.

[3] C. S. Campbell, P. P. Maglio, A. Cozzi, and B. Dom, "Expertise identification using email communications," *Proceedings of the twelfth international conference on Information and knowledge management*, pp. 528-531, 2003.

[4] G. V. Cormack and T. R. Lynam, "Statistical precision of information retrieval evaluation," *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 533-540, 2006.

[5] R. T. H. Tsai, H. J. Dai, H. C. Hung, R. T. K. Lin, W. C. Chou, Y. S. Su, M. Y. Day, and W. L. Hsu, "BESearch: A Supervised Learning Approach to Search for Molecular Event Participants," *The 2007 IEEE International Conference on Information Reuse and Integration*, pp. 412-417, 2007.

[6] J. Finkel, S. Dingare, H. Nguyen, M. Nissim, C. Manning, and G. Sinclair, "Exploiting Context for Biomedical Entity Recognition: From Syntax to the Web," *Joint Workshop on Natural Language Processing in Biomedicine and Its Applications at Coling*, 2004.

[7] B. Settles, "Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets," *COLING 2004 International Joint workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP)*, 2004.

[8] R. T. H. Tsai, C. L. Sung, H. J. Dai, H. C. Hung, T. Y. Sung, and W. L. Hsu, "NERBio: using selected word conjunctions, term normalization, and global patterns to improve biomedical named entity recognition," *BMC Bioinformatics*, 2007.

[9] Y. Tsuruoka, "Bidirectional inference with the easiest-first strategy for tagging sequence data," *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 467-474, 2005.

[10] Y. Kogan, N. Collier, S. Pakhomov, and M. Krauthammer, "Towards Semantic Role Labeling & IE in the Medical Literature," *AMIA Annual Symposium Proceedings*, vol. 2005, p. 410, 2005.

[11] R. T. H. Tsai, W. C. Chou, Y. S. Su, Y. C. Lin, C. L. Sung, H. J. Dai, I. T. H. Yeh, W. Ku, T. Y. Sung, and W. L. Hsu, "BIOSMILE: A semantic role labeling system for biomedical verbs using a maximum-entropy model with automatically generated template features," *BMC Bioinformatics*, vol. 8, p. 325, 2007.

[12] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, "Introduction to WordNet: An On-line Lexical Database*," *International Journal of Lexicography*, vol. 3, pp. 235-244, 2004.

[13] B. Boguraev, T. Briscoe, J. Carroll, D. Carter, and C. Grover, "The derivation of a grammatically indexed lexicon from the Longman Dictionary of Contemporary English," *Proceedings of the 25th conference on Association for Computational Linguistics*, pp. 193-200, 1987.

[14] E. Voorhees and D. Tice, "The TREC-8 question answering track evaluation," *Text Retrieval Conference TREC*, vol. 8, 2000.

[15] W. Hersh, A. M. Cohen, P. Roberts, and H. K. Rekapalli, "TREC 2006 genomics track overview," *The Fifteenth Text Retrieval Conference*, 2006.