Using Profile Matching and Text Categorization for Answer Extraction in TREC Genomics

Haiqing Zheng HAYDEN.ZHENG@GAMIL.COM
Department of Computer Science and Engineering, Fudan Univerisity
220 Handan Road, Shanghai 200433, China

Chen Lin CHEYENNE.LIN@GMAIL.COM

Lishen Huang LISHENHUANG@GMAIL.COM

Jun Xu XJ.MANU@GMAIL.COM

Jiaqian Zheng JQZHENG@FUDAN.EDU.CN

Qi Sun 052021188@FUDAN.EDU.CN

Junyu Niu JYNIU@FUDAN.EDU.CN

Abstract

TREC'06 genomics track was focusing on text mining and passage retrieval. WIM lab participated in this year's TREC genomics track. Our system consists of five parts: preprocessing, sentence generation, document retrieval, answer extraction and answer fusion. And we developed two different method: a automated profile matching-based method and a text categorization-based method to do the text mining, we will compare the performances between those two methods.

1. Introduction

TREC genomics track is always focusing on the text processing in biomedical fields. And this year's task was mainly trying to find a specific passage for one query, here, the query was propose in a natural language way and the passages were composed by two or more short sentences which close to each other. And there is also some measurements about grouping the submitted passages into several different aspects.(Hersh, 2006)

We are going to give an introduction to our genomic text mining system. Firstly, we will give a brief overview of our system, and then will give a more detailed description of each part. And the result also be given in the next part. At last, we do some conclusion to this year's track.

2. System Overview

Our system is mainly contains 5 parts: preprocessing, sentence generation, document retrieval, text mining and answer fusion. The architecture of our system is below:

2.1. Preprocessing & Sentence Generation

The corpus of TREC 2006 is the electronic edition paper from Highwire press. And the submitted result should give the displacement of the start offset and the length of the relevant passages.

We firstly remove all the structure labels of the html and the information have nothing to do with the main part. After this, we convert the html format files into pure text.

Then we parsed all the text files, for the submitted results should give out the offsets of the paragraphs. We simply treated dot as the separates of the sentences. And for convenience, we also saved the start displace-

Appearing in *Proceedings of the 2006 Text Retrieval Conference (TREC 2006)*

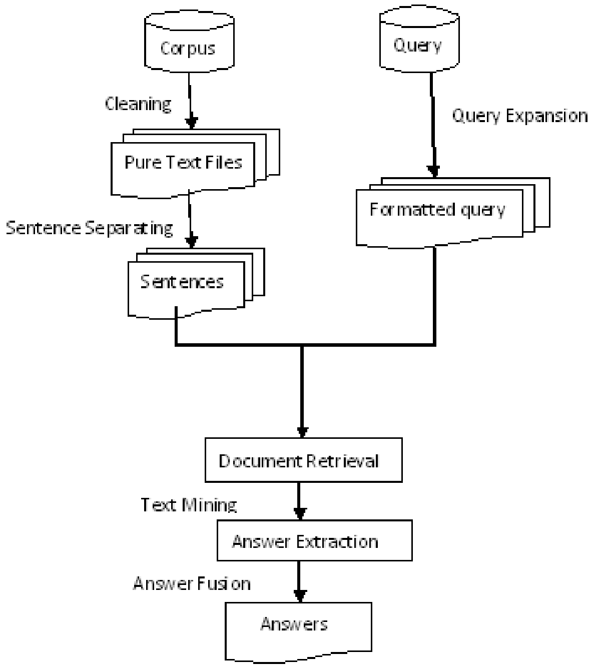

Figure 1. The architecture of the system

ment information of the sentences and its length.

2.2. Query Expansion

The topics of this year's TREC genomics is the same as the last year- generic topic templates (GTTs) which are derived from an analysis of the topics from the 2004 track and other known biologist information needs.(Hersh, 2006)

There're for types of this year's topics:

(1) Information describing the role(s) of one or more genes involved in a given disease.

(2)Information describing the role of a gene in a specific biological process.

(3)Information describing interactions (e.g., promote, suppress, inhibit, etc.) between two or more genes in the function of an organ or in a disease.

(4)Information describing one or more mutations of a given gene and its biological impact.

There're a lot entity names such as gene names, protein names, and disease names in the topics. And there are a lot of synonyms and abbreviations of the entity names, so for the initial topics query expansion is badly needed in this task.

Firstly, we picked out all the entity names, and get the synonyms from the PubMed databases. All the synonyms and its different abbreviations are expanded as the new input query for the retrieval system.

Secondly, for some words which seem not belong to the biology field such as 'effect', 'migrate' etc. We used word-net to find out the synonyms and also put them into the initial query(Voorhees, 1994).

Based on the two query expansion steps we mentioned before, the new query was formed which contains more information which could help the performance of the document retrieval.

2.3. Document Retrieval

For document retrieval we took the widely used Lemur toolkits as our search engine. The Indri query language, based on the Inquery query language, can handle both simple keyword queries and also complex queries. Such a query language sets Indri apart from many other available search engines. It allows complex phrase matching, synonyms, weighted expressions, Boolean filtering, numeric (and dated) fields, and the extensive use of document structure (fields), among others.
Taken topic 166: What is the role of Transforming growth factor-beta1 (TGF-beta1) in cerebral amyloid angiopathy (CAA) ?

we transformed it into the normalized query form such as : #weight(2.0#1(Transforming growth factor beta) #1(TGF beta) Tgfb Tgfb-1 2.0 #1(Cerebral Amyloid Angiopathy)).

3. Answer Extraction

3.1. Profile-based mining

One of the most popular method using in answer extraction is the profile-based methods. In TREC QA a lot profiles had been developed either manually or automatically to find out the most fit sentences to the profiles for different types of questions.

In this year's genomics track, we developed a profile based method to extract the most proper passages for each topic in automatic way. Firstly, we did a sentence-level retrieval to find out the most relevant sentences, and then we used a parsing tools to parse all the topics and also the submitted rank first N sentences by last step. And we checked the similarity between the sentences and the topic.

3.1.1. Sentence retrieval

After the document retrieval step, we got the ranked document list. But this year's task is trying to extract the most relevant paragraphs, which means that we should measure the relevance of each single sentences(Stefanie Tellex, 2003).

In this step we calculate the score of each sentence which indicates if the sentence is tightly related to the given query. The original algorithm was proposed in (A Ittycheriah, 2000). The score is composed of four parts,
(1) match score: The sum of the scores for each matched word (which means the word appeared in the query expansions) in the sentence using formula:
$\text{S}_{match}=\sum_{i=1}^{\|matchwords\|} tf_i \times (log_{10}\frac{N}{df_i})$

(2) mismatch score: The sum of the scores for each mismatched word (which means the word misses in the query expansions)) in the sentence using formula as:
$\text{S}_{match}=\sum_{i=1}^{\|mismatchwords\|} tf_i \times (log_{10}\frac{N}{df_i})$

(3) score for cluster Compute the number of words that appeared adjacently in both query and sentences.

(4) score for dispersion Compute the number of words that appear between the match words.

Each score has a weight ((1) and (3) are positive and (2) and (4) are negative) and we add them together to get a final score for each sentence.
$Score_{sen} = \alpha S_{match} + \beta S_{clu} - \gamma S_{mis} - \delta S_{dipersion}$

There are several aspects to be considered
(a) When a word repeat several times in one single sentence, it's score for match should decline each time when the score is added.
(b)When the disease name and gene name in the query expansions appear at the same time in the candidate sentence, the sentence should have some bonus score, for it is more likely to indicate the relationships between the gene and the disease.

3.1.2. Parsing

Mini-par(Lin,) was used as parsing tool in the experiment which was is a broad-coverage efficient parser for the English language. We here use it to parse all the topics and the return sentences by sentence retrieval.

3.1.3. Profile matching

In order to find out the most relevant passage to the query, we should find the most proper sentences which have the similar grammatical structure to the query and also using the term relationship information to extract the exact passage(Hang Cui, 2005). So we decided using a parsing tool to parse all the queries and find out their grammatical structure. Based on the query's structure, we can construct the profile of each topic.

By using external tools 'minipar', sentence in English can be converted into tree-like structure. Compared with POS tagging, these works go steps forward to the sentence parsing by approaching semantic level.

On applying tree-like structure to QA system, we have to find a suitable critical function to evaluate the similarity between two English sentences. We construct this function base on gene-node-relation algorithm, described as follow:

A1: parsing the question sentence into POS tree, splitting gene1, gene2 into word token.
A2: finding the enclosure of gene1 & gene2 in the POS tree
A3: if enclosure(gene1) and enclosure(gene2) are connective, return false;
A4: finding the co-parent node of two enclosures, named as COP
A5:ensuring the dependency relationship;
A6: getting the shortest connectivity path between two enclosures (crossing COP) named SCP
A7: for each answer-candidate, calculate the COP, TYPE, SCP, comparing them with that of questions, then matching the similarity mark.

PS1: matching TYPEs if equals, mark++; else mark–;
PS2: matching COP if equals, mark++; else mark–;
PS3: matching CSP finding the topest prep in the CSP, and then comparing the prep between them, if equals or both-NULL, mark++; else mark–;

3.2. Text Categorization-based Extraction

For the returned 1000 documents for each topic, we implemented a text categorization based method for each one.

Firstly, we parsed all the the returned relevant documents, and all those documents are divided into tens of thousands of sentences. We then using the Lemur toolkits to index them. Then we input the queries used the retrieval step, and get to first N sentences as the relevant ones.

We supposed that the information relevant to the topic is no only just in the keywords which presented in the query but also in the context of each related passages. And a lot text mining methods didn't consider the context of the relevant answers.

Then, we use a SVM-light(support vector machine) classifier(Joachims, 1999) to find the relevant answers to the topic. The methods we took will be in described details next:

1) we picked up the ranked first n$_p$ as the positive samples for a certain topic t$_i$ to the classifier; and the last ranked n$_n$ as the negative samples. and the left $N - n_p - n_n$ are the unlabeled data which should be classified. And the idea could be very easily understand, for the most relevant sentences to one topic, it can be regarded as the same class of the topic;
2)the weighting method we used here is a tf*idf based method: in the jth sentence, term t$_i$ has the weight as:
$\boldsymbol{weight(t_{i,j})=stf(t_{i,j})*}\frac{sidf(t_i)}{idf(t_i)}/(\mathrm{LENGTH(j)})$
in which stf(t$_{i,j}$) is the number term t$_i$ exits in the jth sentence,
and sidf(t$_i$) is the number sentences in which term t$_i$ exits in the N sentences,
LENGTH(j) stands for the length of sentence j.

After classifying the unlabeled data, every sentence got a score, and for each topic tp$_i$, we defined a threshold TH$_i$, for those sentences whose score is greater that this threshold TH$_i$, it will be regarded as a relevant sentence and would be returned as a relevant answer.

3.3. Answer Fusion

The two text mining systems were returning the separated sentences each with a score. So, we must combine the sentences in the neighborhood into a single passage.

Here, for the thousands of sentences returned by mining step, we check each sentence's SenId which was assigned in the preprocessing step. If sentence i and sentence j are closed to each other, and i has the score S$_i$, and j has the score S$_j$, the fused passage which consisted of i and j will have the score as S$_(pasg)=\frac{s_i+s_j}{2}$. In generally, if a passage pasg$_i$ was consist of sentence from i$\cdots$j, and which has the score s$_i$, $\cdots$, s$_j$, the score of the passage is s$_(pasg_i)=\frac{\sum_i^j(s_t)}{|pasg_i|}$,
where $|pasg_i|$ stands for the number of sentences in this passage.

4. Results and Analysis

We have submitted three runs of this years TREC genomics: trecgen1, trecgen2 and trecgen3. The first two runs are based on the text categorization method, and the last one is based on the profile matching method. The differences between the first two runs are the numbers of the positive samples and the negative samples are not the same, trecgen2 was with fewer samples used for categorization.

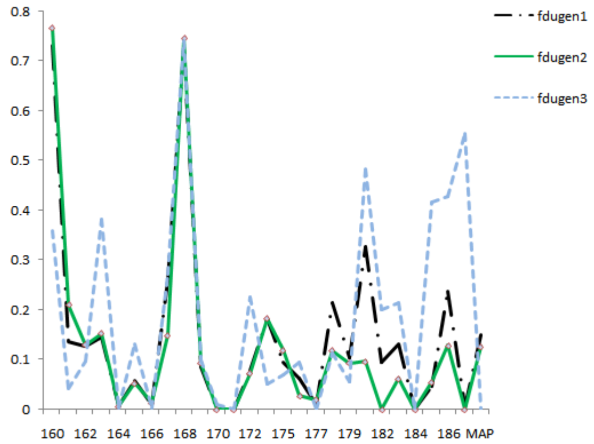

Figure 2. The document-level MAP of TREC 2006 Genomics

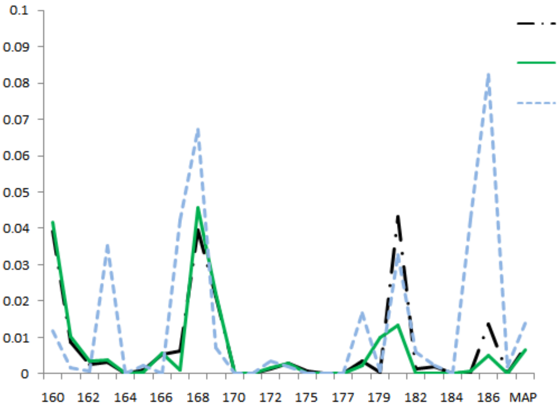

Figure 3. The passage-level MAP of TREC 2006 Genomics

This year's TREC's evaluation is based on three different levels of retrieval performance: passage retrieval, aspect retrieval, and document retrieval. And each of these provides insight into the overall performance for a user trying to answer the given topic questions. Based on this year's protocol, each level would be measured by some variant of mean average precision (MAP).

The passage-level retrieval performance was using character-based MAP, while the aspect-level retrieval performance was using aspect-based MAP and the document-level retrieval performance was using document-based MAP.

Fig2, fig3 and fig4 are our submitted runs distribution, from which we can see that the first two runs(trecgen1 and trecgen2) are no performs as good as the trec-

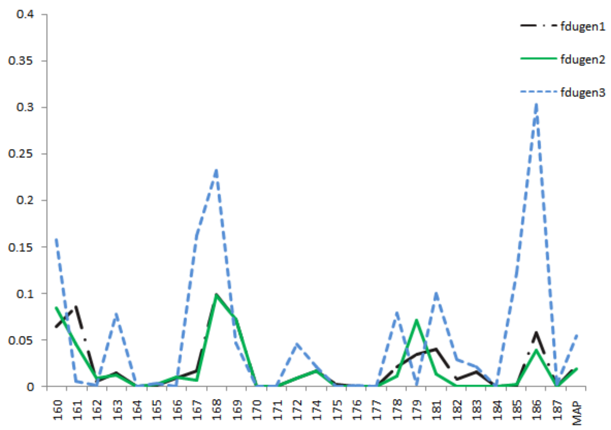

Figure 4. The aspect-level MAP of TREC 2006 Genomics

gen3, which seems that the swallow natural language processing will helps us improve the results. But we can also see that there's no relevant topic for few topic, and we checked out that has lot to do with the initial document level retrieval. For the first step of document level retrieval can not return relevant results, so the next steps can not get a good result.

5. Further Work

Text mining, especially in the bio-medical field is now really a new and hot research area(Aaron M. Cohen, 2005). For the fast increasing bio-medical articles, it will help researchers to find out the useful information which maybe hide deeply in the literatures, and this will save a lot of energy and time for them avoiding to do the same experiments that have had been done. And this year's track has provide a good example for this kind of work. Finding out the most relevant passages to the query and giving the aspect of this passage, which seems very closely to the requirement of real world.

In out experiment, we have done a lot on finding the most relevant passages, but few work has done about the aspects. In the next experiments, we are planning to use some text clustering methods to cluster the relevant passages in to some smaller sets while the passages in the same set maybe providing the same aspect answer to a certain topic.

References

A Ittycheriah, S. R. (2000). Ibm's statistical question answering system-trec 11. *Proceedings of the Eleventh Text Retrieval Conference.* Gaitherburg, MD.

Aaron M. Cohen, W. R. H. (2005). A survey of current work in biomedical text mining. *Briefs in Bioinfomatics*, *6*, 57–71.

Hang Cui, Min-Yen Kan, T.-S. C. (2005). Generic soft pattern model for definitional question answering. *Proceedings of the 28th Annual International ACM-SIGIR Conference* (pp. 384–391). Salvador, Brazil: Morgan Kaufmann.

Hersh, B. (2006). *Trec 2006 genomics track protocol* (Technical Report). http://ir.ohsu.edu/genomics/2006protocol.html.

Joachims, T. (1999). *Making large-scale svm learning practical. advances in kernel methods - support vector learning.* Cambridge, USA: MIT-Press.

Lin, D. Dependency-based evaluation of minipar. *In Workshop on the Evaluation of Parsing Systems.* Granada, Spain.

Stefanie Tellex, Boris Katz, J. L. (2003). Quantitative evaluation of passage retrieval algorithms for question answering. *Proceedings of the 26th Annual International ACM-SIGIR Conference* (pp. 41–47). Toronto, Canada: Morgan Kaufmann.

Voorhees, E. M. (1994). Query expansion using lexical-semantic relations. *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval* (pp. 61 – 69). Dublin, Ireland: ACM/Springer.