Question Answers System in Biomedical Domain

S.Krishna Santosh Reddy – 200401030
Segu Revanth - 200401032
Abhinay Pandya
Evaluation Committee no: 6

***Abstract* - We are working on question answering system in biomedical domain which is retrieving biomedical passages from a corpus of documents. As with all the question answer systems our methodology is to build from the question, the query to information retrieval engines and fine-tune the results so obtained.**

Index Terms - biomedical, information retrieval, Lemur and Indri, question answer system, Text REtrieval Conference (TREC)

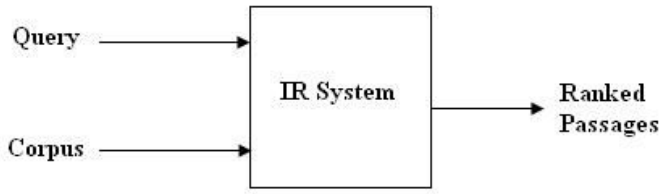
INTRODUCTION

The Text REtrieval Conference (TREC), co-sponsored by the National Institute of Standards and Technology (NIST) and U.S. Department of Defense, was started in 1992 as part of the TIPSTER Text program. Its purpose was to support research within the information retrieval community by providing the infrastructure necessary for large-scale evaluation of text retrieval methodologies.

Question answering (QA) is a type of information retrieval. Given a collection of documents the system should be able to retrieve answers to questions posed in natural language.

Our system is a fine grained Information Retrieval system which answers a given set of questions gives a corpus containing answers. In another words a next step to what Google does.

Note: Though our System is not a perfect Question answering system we named it as Question answering system as we got our problem statement and data from TREC, which named it as Question answer system.



Reason for choosing passages instead of one line answers (as given as TREC):

Research, conducted by TREC, shows that most of the end users, in biomedical domain, often prefer passages (a paragraph or a set of paragraphs) containing the answer to the question posed, instead of a direct answer, as the output. So, our system is made to output paragraphs containing relevant answer to the questions posed.

About the Data

We collected data for our system from Text REtrieval Conference (TREC) [1] genomics website. The data consists of a corpus consisting of 162,259 full-text HTML documents from the 49 journals and a set of questions and their corresponding answers (passages) which are manually evaluated from the corpus.

METHODOLOGY

The main motive behind building any QA system is improving its performance. Many different measures for evaluating the performance of information retrieval systems have been proposed. The measures require a collection of documents and a query and of all the methods proposed recall and precision are treated to the basic ones which every Information Retrieval (IR) system should provide.

Thus the efficiency of any Information Retrieval system is based on the measures of Precision and Recall it provides. Hence our system is built keeping these two parameters in mind and we will explain how we improved the performance of our system by improving these factors one by one.

Recall

Recall is the fraction of the documents that are relevant to the query that are successfully retrieved.

$$\text{Recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

Precision

Precision is the fraction of the documents retrieved that are relevant to the user's information need.

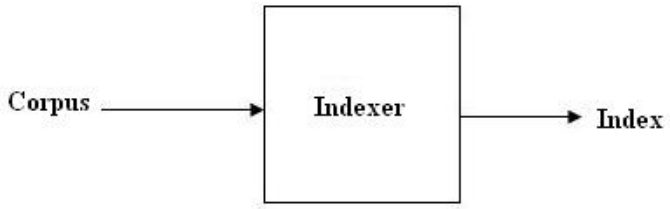
$$\text{Precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

ACHIEVING RECALL

We achieved recall through the following three phases. In all the three phases our main aim was to get all the relevant passages which are likely to answer the question.

INDEXING (PHASE I)

We indexed all the documents using Lemur Indri Indexer to give a single big Indri Index file which will be used in the later part of the process to retrieve the answers for the questions.



GENERATING QUERY (PHASE II)

I. Extracting NPs

When a user poses a question to our system first Noun Phrases(NPs) are extracted from the question

. Consider as an example topic 160:

What is the role of **Prnp** in **mad cow disease** ?

To get these bold faced noun phrases (NPs which contain the gene, biological and disease terms) present in the question we parsed the question using Geniatagger and filtered its output. Filtering is done to remove the non-medical words like what, the role.

II Collecting Synonyms

After obtaining a list of noun phrases in a topic description, the next step in our system is to expand the phrases into lists of synonyms and related terms. We expand the noun phrases into synonym lists by searching the MeSH [2] (Medical Subject Heading) database. The biological terms along with synonyms are passed into Porter Stemmer (a tool of used for stemming) to give stemmed form (root form) of the words.

For topic 160, the aforementioned expansion techniques produce the following synonym lists:

- **PrnP**: prnp protein, prion protein, gss protein, G1-dependent prion Protein.
- **mad cow disease**: encephalopathy, bovine spongiform encephalopathy, bse, bses, encephalitis, mad cow diseases, spongiform encephalopathy.

III .Building Indri Query

Query for our IR system is constructed using all the synonyms collected in the previous step.

We utilize several of the Indri structured query language [3][4][5] operators in building queries. We begin at the level of forming a query term based on a single synonym list. Specifically, we form a #syn term that treats each of the expressions it contains as synonyms. The #syn term contains each item in the synonym list as an exact phrase via the #1 operator.

This means we look for documents that contain at least one of the synonyms as an exact match. For example, we represent one of the topic-160 synonym lists as follows:

```
#syn(
        #1(mad cow disease) #1(BSE)
        #1(Bovine Spongiform Encephalopathy)
        #1(Spongiform Encephalopathy)
...
)
```

After forming terms corresponding to each synonym list, we combine the synonym lists using the #uw operator, which requires all of its operands (at least one entry from each synonym) to be present but in any order. For example, we join the topic-160 synonym lists as follows:

```
#uw(
        #syn(
                #1(mad cow disease) #1(BSE) ...
                )
        #syn(
                #1(PrnP) #1(prion protein) ...
                )
)
```

So far, our query says that we need to find at least one synonym for each important noun phrase in the topic. The #uw requires each #syn to return true, but this simply means one of the contained phrases must be found.

Then, we employ Indri's #combine operator. Unlike a simple Boolean AND, which gives a result of true or false, the #combine operator gives a higher score to results that contain more of its operands. We end up with a query of the general form shown below:

```
#combine(
                #uw(
                        #syn( #1(a) #1(b) )
                        #syn( #1(c) #1(d) )
                        ...
                        )
)
```

This query tells Lemur to search for documents that contain at least one synonym for each important noun phrase in the topic and they can be in any order and also rank them.

Finally we replaced #uw with #uw50 which means the noun phrases can contain no more than 50 words between them. This is made to increase the speed of retrieval at the cost of losing only a few documents. Also the more close the noun phrases are the more appropriate the answer is. So we end up with the following query:

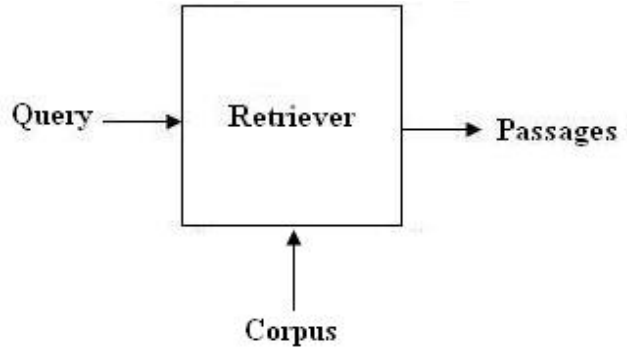
```
#combine(
        #uw50(
                #syn( #1(a) #1(b) )
                #syn( #1(c) #1(d) )
                ...
                )
)
```

The end result is that Lemur/Indri fetches all the documents meeting the #uw50 criteria and then ranks them.

RETRIEVAL (PHASE III)

After constructing queries as described above, we execute them against the Indri index built in Phase I. This produces a ranked list of files satisfying our query. In order to verify with the results we need byte offsets and lengths within the original documents. For this we run annotated search on Indri index rather than a simple search. The difference between normal search and annotated search is that in annotated search we get the starting (call it s1) and ending (call it e1) indexes within the document which match the given Indri query.

Once we get s1 and e1 we can compare with the actual answers provided by TREC. As of now we are comparing the results manually. We found out that though the results are good we are missing some of the documents because of factors like presence of lexical variants of query terms in the documents, missing some of the synonyms due to MeSH.



ACHIEVING PRECISION

Passages retrieved using the Indri Query are taken for re-ranking to increase the Precision of the retrieved documents.

RERANKING (PHASE IV)

We studied the latest paper on reranking which says that graph analysis algorithms such as PageRank and HITS have been successful in web environments because they are able to extract important inter-document relationships from manually-created hyperlinks. Thus if we can somehow build a web like structure among the documents which we have retrieved, then we can apply Pagerank and Hits for reranking the same. The paper considers the application of these algorithms to related document networks comprised of automatically-generated content-similarity links.

Specifically, this work tackles the problem of document retrieval in the biomedical domain, in the context of the PubMed search engine. A series of re-ranking experiments demonstrate that incorporating evidence extracted from link structure yields significant improvements in terms of standard ranked retrieval metrics[7].

Theoretical Study 1

As the most successful re-ranking algorithms, although mainly related to re-rank web documents, we studied Google's Page Rank and HITS algorithms for re-ranking the passages retrieved.

Page Rank:

In order to measure the relative importance of web pages, PageRank[8][9] is proposed , a method for computing a ranking for every web page based on the graph of the web. PageRank has applications in search, browsing, and traffic estimation. Every page has some number of forward links (outedges) and backlinks (inedges).

The reason that PageRank is interesting is that there are many cases where simple citation counting does not correspond to our common sense notion of importance. For example, if a web page has a link off the Yahoo home page, it may be just one link but it is a very important one. This page should be ranked higher than many pages with more links but from obscure places. PageRank is an attempt to see how good an approximation to "importance" can be obtained just from the link structure.

A page has high rank if the sum of the ranks of its backlinks is high. This covers both the case when a page has many backlinks and when a page has a few highly ranked backlinks.

HITS:

Hyperlink-Induced Topic Search (HITS) [9] [10] is a link analysis algorithm that rates web pages. It determines two values for a page: its authority, which estimates the value of

the content of the page, and its hub value, which estimates the value of its links to other pages.

Authority and hub values are defined in terms of one another in a mutual recursion. An authority value is computed as the sum of the scaled hub values that point to that page. A hub value is the sum of the scaled authority values of the pages it points to.

We dropped the idea of Page rank because in the actual web pages are interconnected (hyperlinked) and these interconnections are given manually for each web page. But in our case the passages retrieved are not interconnected and we can use some criteria to interconnect them. But these heuristic approaches of interconnecting the passages retrieved do not resemble web like links and hence PageRank and HITS are not efficient in our case. Though we got PUBMED (PubMed is a service of the U.S. National Library of Medicine that include links to full text articles and other related resources) which gives top five documents ranked on counting number of simultaneous visits to both documents even that is not an efficient way.

Theoretical Study II

Our IR Engine retrieves documents based on probabilistic retrieval for a query Q a document D ranking is done on the basis of P(D/Q) considering query as set of search words(stemmed form of biological words and their synonyms)

$$Q = (w_1, w_2, \ldots, w_n)$$

And

$$P\left(\frac{D}{Q}\right) = P\left(\frac{D}{w_1, w_2, \ldots, w_n}\right)$$

Considering the search words are independent of each other. Then the above equation turns into following equation

$$P\left(\frac{D}{Q}\right) = P\left(\frac{D}{w_1}\right) P\left(\frac{D}{w_2}\right) \cdots P\left(\frac{D}{w_n}\right)$$

The proof is as follows:

Applying Bayes Rule

$$P\left(\frac{D}{w_1, w_2, \ldots, w_n}\right) = P\left(\frac{w_1, w_2, \ldots, w_n}{D}\right) \frac{P(D)}{P(w_1, w_2, \ldots, w_n)}$$

Assuming $w_1, w_2, \ldots, w_n$ are independent of each other

$$P\left(\frac{w_1, w_2, \ldots, w_n}{D}\right) = P\left(\frac{w_1}{D}\right) P\left(\frac{w_2}{D}\right) \cdots P\left(\frac{w_n}{D}\right)$$

And

$$P(w_1, w_2, \ldots, w_n) = P(w_1)\, P(w_2) \cdots P(w_n)$$

$$P\left(\frac{D}{w_1, w_2, \ldots, w_n}\right) = \frac{P\left(\frac{w_1}{D}\right) P\left(\frac{w_2}{D}\right) \cdots P\left(\frac{w_n}{D}\right)}{P(w_1)\, P(w_2) \cdots P(w_n)} P(D)$$

$$= \frac{P\left(\frac{w_1}{D}\right)}{P(w_1)} \frac{P\left(\frac{w_2}{D}\right)}{P(w_2)} \cdots \frac{P\left(\frac{w_n}{D}\right)}{P(w_n)} P(D)$$

$$= \frac{P\left(\frac{w_1}{D}\right)}{P(w_1)} \frac{P\left(\frac{w_2}{D}\right)}{P(w_2)} \cdots \frac{P\left(\frac{w_n}{D}\right)}{P(w_n)} P(D)$$

By Bayes Rule

$$P\left(\frac{w_i}{D}\right) = P\left(\frac{D}{w_i}\right) \frac{P(w_i)}{P(D)}$$

$$\Rightarrow \frac{P\left(\frac{w_i}{D}\right)}{P(w_i)} = \frac{P\left(\frac{D}{w_i}\right)}{P(D)}$$

$$P\left(\frac{D}{w_1, w_2, \ldots, w_n}\right) = \frac{P\left(\frac{D}{w_1}\right)}{P(D)} \frac{P\left(\frac{D}{w_2}\right)}{P(D)} \cdots \frac{P\left(\frac{D}{w_n}\right)}{P(D)} P(D)$$

Every Document is retrived for some Query so $P(D) = 1$

$$P\left(\frac{D}{w_1, w_2, \ldots, w_n}\right) = P\left(\frac{D}{w_1}\right) P\left(\frac{D}{w_2}\right) \cdots P\left(\frac{D}{w_n}\right)$$

We calculate each of the independent probabilities by TF-IDF (Term frequency * Inverse Document Frequency).

As we can see if we consider the query words to be dependent, the rankings will be more appropriate and we thought of modifying the existing ranking algorithm by not assuming the independence of query words. But calculating the dependent probabilities of words for a document is not an easy task so we thought we will not be able to complete the task in the given time. Hence, we dropped this idea/

Note: The above model of ranking and retrieving the documents followed in our IR system is a combination of the language modeling and inference network retrieval frameworks.

Practical Work

As our ideas of implementing re-ranking failed and we could not find a better algorithm (which gives good results for us) we thought a new algorithm for re-ranking which is simple but which gave good results. The algorithm is explained in the following paragraph.

We collected top 1000 passages retrieved by our system and sent them again to our ranking system. Now the ranking

of passages changes because the ranking is done based on TF-IDF and IDF changes as the documents are less. We assume the top 1000 passages capture all the best results. So in a way we are getting the top 1000 passages initially from the corpus and then from these 1000 passages we are getting the top 100 passages. Note that the ranks of 1000 passages change after the re-ranking because IDF changes.

Theoretical Study III

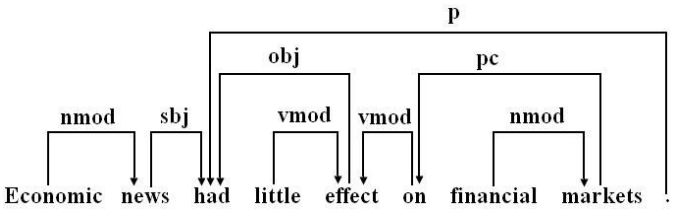
Though the above re-ranking gave good results we were not satisfied as even the above algorithm does not take into account the structural relationship between question and passage. Thus, after searching for different re-ranking algorithms we finally end up with a new idea of implementing re-ranking by finding the relevance between query and document structure. To find the relevance we decided to match the structures of\query and document using dependency graphs.

The basic driving force behind the idea is that till now we have not captured the structure of query and the document, which is crucial. Keeping in simple words, we want the answers obtained for the question "what is the role of prnp in mad cow disease?" to be different from those obtained for "what is the role of mad cow disease in prnp?" this can be achieved if we take the structure of the query and document into consideration.

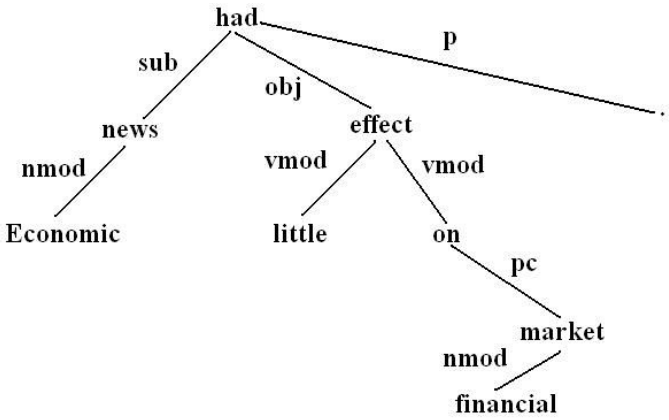
Dependency Graphs

Dependency graphs syntactic structure consists of lexical items, linked by binary asymmetric relations called dependencies.

The sentence is an organized whole, the constituent elements of which are words. Every word that belongs to a sentence ceases by itself to be isolated as in the dictionary. Between the word and its neighbors, the mind perceives connections, the totality of which forms the structure of the sentence. The structural connections establish dependency relations between the words. Each connection in principle unites a superior term and an inferior term. The superior term receives the name governor. The inferior term receives the name subordinate. Thus, in the sentence Alfred parle [. . .], parle is the governor and Alfred the subordinate.

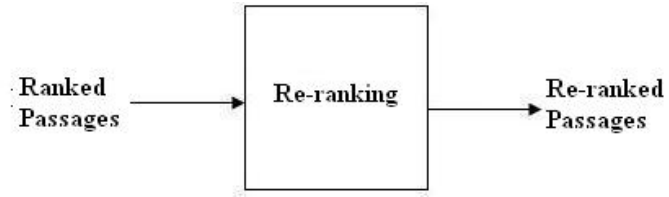


The same structure can be represented in other form as shown below



Properties of Dependency graphs

- Weekly Connected
- Acyclic
- Single-headed



Calculating MAP

Average Precision

Average precision emphasizes returning more relevant documents earlier. It is average of precisions computed after truncating the list after each of the relevant documents in turn, a document which is retrieved earlier (given better rank) has more weight than the later ones.

$$\text{Ave P} = \frac{\sum_{r=1}^{N} \left(P(r) \times rel(r) \right)}{\text{Number of relevant Documents}}$$

Mean of Average precision of calculated over all the questions gives MAP(Mean Average Precision) of the IR system.

FUTURE WORK

As we can see re-ranking based on dependency graphs captures the similarity between structures of query and document and hence gives good results but we did not explore

it to the full-extent if we get a chance to further work in this area we want to exploit dependency graphs to its full extent.

REFERENCES

[1] TREC Genomics Track - http://ir.ohsu.edu/genomics/

[2] MESH - www.nlm.nih.gov/mesh/MBrowser.html

[3] Indri Language - http://www.lemurproject.org/indri

[4] *Don Metzler* , “Indri Retrival Model Overview.”, http://ciir.cs.umass.edu/~metzler/indriretmodel.html

[5] Strohman, T., Metzler, D., Turtle, H., and Croft, W.B., "Indri: A language-model based search engine for complex queries (extended version)" CIIR Technical Report, 2005

[6] Metzler, D. and Croft, W.B., "Combining the Language Model and Inference Network Approaches to Retrieval," *Information Processing and Management Special Issue on Bayesian Networks and Information Retrieval*, 40(5), 735-750, 2004.

[7] Jimmy Lin,” PageRank without Hyperlinks: Reranking with Related Document Networks”, Technical Report LAMP-TR-146/HCIL-2008-01, University of Maryland, College Park, January 2008.

[8] S. Brin, L. Page, “The PageRank Citation Ranking:Bringing Order to the Web", International World Wide Web Conference, 1998

[9] Amy N. Langville and Carl D.Meyer, ”Google’s Page Rank and Beyond : The Science of Search Engine Rankings”.

[10] Jon M. Kleinberg,” Authoritative Sources in a Hyperlinked Environment”, Journal of the ACM (JACM),1999

[11] David A. Grossman ,Ophir Frieder” Information Retrieval Algorithms and Heuristics”

[12] Chin-Yew Lin,” The Effectiveness of Dictionary and Web-Based Answer Reranking”, Association for Computational Linguistics Morristown, NJ, USA, 2002

[13] Dragomir Radev , Weiguo Fan , Hong Q i, Harris Wu, Amardeep Grewal ,” Probabilistic question answering on the Web”, 10 Feb 2005.

[14] Ganesh Ramakrishnan, Soumen Chakrabarti, Deepa Paranjpe, Pushpak Bhattacharyya,” Is Question Answering an Acquired Skill?”, Journal of the ACM (JACM) ,2004

[15] Bo Han, Slobodan Vucetic, Zoran Obradovic,” Reranking MEDLINE Citations by Relevance to a Difficult Biological Query”

[16] Jaime Carbonell, Jade Goldstein” The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries”, Journal of the ACM (JACM),1998

[17] William Hersh, Aaron M. Cohen, Phoebe Roberts, Hari Krishna Rekapalli” TREC 2006 Genomics Track Overview”,2006