ASU at TREC 2006 Genomics Track

Luis Tari, Graciela Gonzalez, Robert Leaman, Shawn Nikkila,
Ryan Wendt, Chitta Baral
Department of Computer Science and Engineering,
Arizona State University

Abstract

This paper describes our experiments in the TREC 2006 Genomics track submitted by the ASU BioAI group, as well as experiments based on the improvements made after our submission. Some of the major issues we tried to address in our experiments are how to (1) extract keywords from natural language questions in the biomedical domain and (2) determine the relevancy of passages.

1. Introduction

The task for the TREC 2006 Genomics track is quite different from last year in the sense that the goal is to develop a system that can provide answers with enough context and supporting information to the questions. Another noticeable difference is that HTML full-text articles are provided rather than abstracts of the articles. The system we developed is fully automated: from processing natural language questions to presenting answers in the form of passages. Our main design principle of the system is to answer questions that are not limited to the questions provided by TREC. To achieve this goal, we experimented our system with some innovative approaches, such as taking advantages of the WordNet [1] resource to recognize keywords from natural language questions. Another highlight of our system is the use of subject-verb-object triplets to realize the relevancy of passages.

The paper is organized as follows. Section 2 provides a system overview, with details of some of the major components of the system discussed in the subsections. Section 3 describes our runs submitted to TREC 2006 Genomics track and our revised run after improvement of some of the components. Section 4 describes our revised run and section 5 provides analysis of how each of the major components affects the performance of the overall system. Finally, we summarize the new components we improved for our revised run, and discuss some of the issues and challenges we face in developing our system in section 6.

2. System Overview

Our system for TREC 2006 Genomics track can be divided into three major components: preprocessing, document retrieval and passage retrieval. We first provide an overview of these three major components and describe some of the innovative features of our system in details. The role of the preprocessing component is to convert HTML full-text articles provided by TREC into structured XML format of the articles, as shown in figure 1. Acronyms are resolved in the course of preprocessing the full-text articles. The details of the preprocessing component are described in subsection 2.1. The document retrieval component, as illustrated in figure 2, involves the processing of natural language questions to form queries (subsection 2.2) and expansion of biological entities identified by the question processor (subsection 2.3). Articles are retrieved using variants of queries generated from the original queries (subsection 2.4). Passages are retrieved by the passage retrieval component, which utilizes the subject-verb-object (SVO) triplets of the questions to determine the relevancy of the passages (subsection 2.5). Passages that are determined to be not relevant to the question are filtered out from the final results. The passage retrieval component is illustrated in figure 3.

2.1. Preprocessing

One of the major obstacles is to process the full-text articles provided by TREC, which are in HTML format. Our goal is to translate the HTML articles into XML format so that it can capture information such as which section a paragraph of text is originated from and the byte start and byte offset for each sentences.

A naïve algorithm is used to do such conversion based on section indicators. Section indicators are defined as a list of words such as “abstract”, “introduction” that indicate the beginning of sections in full-text articles. We outline the algorithm in this subsection.

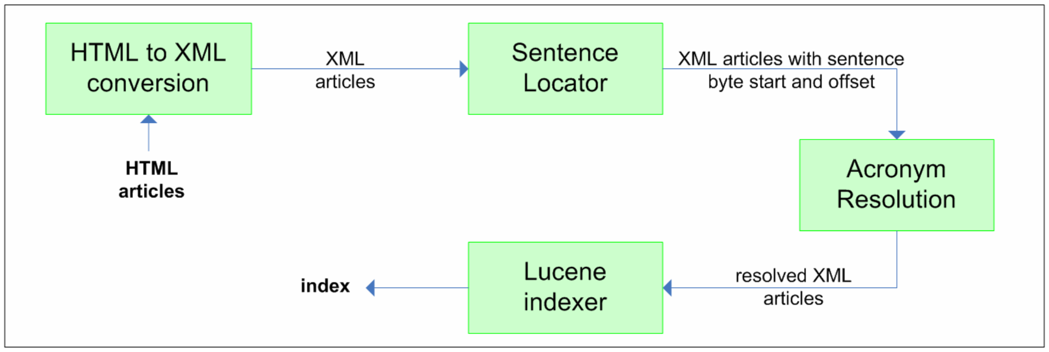

Figure 1 – An overview of the preprocessing component

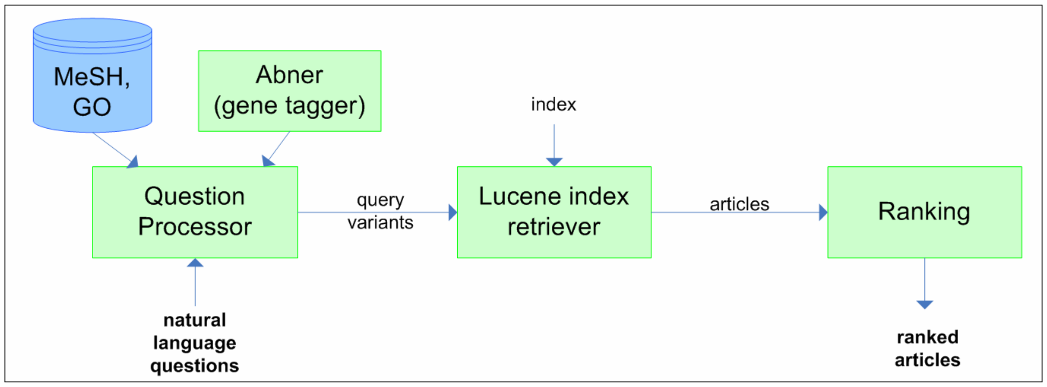

Figure 2 – An overview of the document retrieval component

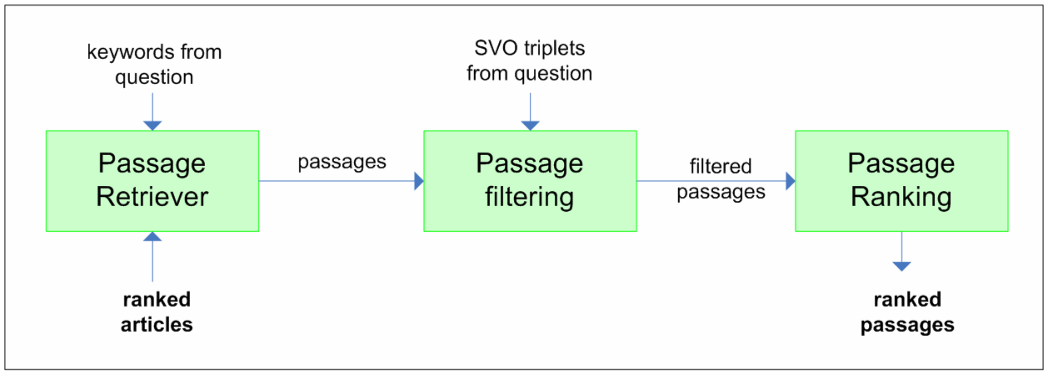

Figure 3 – An overview of the passage retrieval component

For each HTML full-text article, the content of the HTML file is extracted by a tool called HTMLParser[1]. If the article being converted does not have a table of contents, then the article is read line by line to find section indicators. When no such section indicators are found, the first paragraph of the full-text article is considered to be originated from the abstract and the rest of the text is considered as part of the introduction. Since figure captions usually contain rich and concise information, text from figure captions are included into the XML files. The XML files also include byte start and offset for each of the sentences that

[1] http://htmlparser.sourceforge.net/

correspond to the original HTML files. As acronyms are frequently used in the biomedical literature, resolving acronyms is essential to the task of retrieving articles and passages. The acronym resolution algorithm in [2] is used to resolve acronyms in the full-text articles, and the occurrences of the acronyms are stored in the XML files. The corresponding MeSH terms for each article are obtained from PubMed as well. These XML files are then indexed using Lucene [3].

2.2. Keyword extraction

One of the important aspects of our system is to process the given TREC Genomics questions which are in the form of natural language, so that keywords can be extracted from the questions to construct the corresponding queries. It is clear that extracting the right keywords from natural language questions is critical to the task of retrieval of documents and the resulting passages. The queries we are interested in generating are in the form of Lucene queries, which are boolean keyword queries.

Our approach of question processing incorporates the WordNet [1] dictionary together with the part-of-speech and biological entity taggers to extract keywords. WordNet is an extensive lexical dictionary for English that provides different meanings of words, commonly referred to as senses, as well as relationships between words. It is a common resource for QA systems, as the WordNet dictionary aims at non-specialized English words. It is suitable to our task of extracting keywords out of biological questions, as the goal here is to disallow common words in the questions from being selected as keywords. Our assumption behind the idea is that common English words usually have multiple senses. With this assumption, nouns that are not recognized as biological entities are examined using the WordNet dictionary. If an unrecognized noun has none or fewer than *k* different senses, our question processor considers the noun as a keyword. Using the question for topic 177 "*How does Sec61-mediated CFTR degradation contribute to cystic fibrosis?*", the tagger recognized "cystic fibrosis" as a disease, but missed Sec61 as a gene. Using our WordNet-based algorithm, "Sec61-mediated CFTR" and "CFTR degradation" are picked up as keywords. Further processing (breaking hyphenated words) resulted in recognition of Sec61. In our experiments using the TREC Genomics questions, we found that the threshold *k*=3 achieves the best performance for this purpose based on.

2.3. Expansion of biological entities

Variation of naming convention for biological entities is one of the challenges in text retrieval and mining of biomedical literature, in particular gene names. Even resources such as Entrez Gene provide comprehensive lists of synonyms of gene names, slight variation of the official gene names can commonly be found in the biomedical literature. To automatically recognize biological entities, we utilize a gene name tagger called Abner [4] and an efficient exact match of disease names and biological processes utilizing MeSH terms disease category and the Gene Ontology [5] biological process ontology.

Once a word is recognized as a biological entity such as a gene name, it is vital to be able to identify the actual gene name intended in order to find the corresponding synonyms. This is commonly known as gene normalization. Our approach of normalization of gene names is to utilize the fuzzy and proximity queries provided by Lucene. A fuzzy query allows matching words that are within certain edit distance between them. This is useful, for instance, to match the keyword "HNF4" to "HNF-4", in which their edit distance is 1. A proximity query is capable of matching words that are within a specific distance of each other. The gene name "hyprocretin receptor" can be matched to "hyprocretin (orexin) receptor" with 1 extra word in between the words of the gene name. All fields of the human gene names (i.e. genes with taxonomy id of 9606) in the Entrez gene dictionary are first indexed by Lucene. A simple algorithm is then used to find the intended gene name, as follows:

1. If the given gene name is single-word (we assume this is a gene symbol), then
    1.1. form a Lucene query that allows exact matching of the gene name.
    1.2. form a Lucene fuzzy query by adding "*~0.6" at the end of the given gene name.

    1.3. form a Lucene fuzzy query by adding "~0.6" at the end of the given gene name.
2. If the given gene name is multi-word (we assume this is a gene name), then
    2.1. form a Lucene proximity query with distance of 3 based on the given gene name.

Steps 1.2 and 1.3 differ in the sense that the query formed in step 1.2 gives preference to matching gene names that are from the same gene family. For instance, BOP is matched to BOP1 using the query formed in step 1.2, but the query formed in step 1.3 allows to match BOP to BEP.

However, simply adding synonyms from a gene name dictionary such as Entrez Gene is not enough, as variations of gene names might be used in the literature. For example, Nur77 can be used in an article instead of Nur-77, but this is not listed as a synonym of Nur77. As a result, potentially important abstracts can be missed if variants of gene symbols are not considered in the formation of queries. It is important to notice that gene symbol variants are names that do not appear in the gene name dictionary as official names or synonyms. We utilized an algorithm adapted from [6, 7] to generate gene symbol variants for a given gene identified as a keyword.

2.4. Article retrieval and ranking

Once the keywords are identified and expanded as described in the previous sections, Lucene queries are formed based on the keywords. To increase the precision of the queries, predefined lists of MeSH terms for different kinds of biological entities [8] are utilized in the formation of query variants. Multiple query variants are generated from the keywords of the original query, which are described as follows:

- Query variant 1: keywords with their synonyms and variants
- Query variant 2: keywords and their synonyms together with a predefined list of MeSH terms

There are cases when there are none or very few (less than 10) articles are retrieved for a topic. When this happens, the retrieval component switches to the "risky" mode automatically, in which we are willing to give up some precision in favor of higher recall by relaxing the queries. The revised queries allow, for example, fuzzy and proximity matches, so that biological process such as "cell growth" can be matched to "growth of cell". For words that are extracted by the question processor but their types are not identified, such as mutation, we relax the query so that a retrieved article may contain the word mutation. The relaxed query enables us to retrieve articles that must contain biological keywords and may contain the non-biological keywords. To unify and rank the hits retrieved by all query variants, the rank of an article is computed by adding the normalized ranking scores computed by Lucene from each query variants. This allows an article retrieved by multiple query variants to achieve a high rank.

2.5. Passage retrieval and ranking

A passage is defined as a contiguous list of sentences from a paragraph. In our case, we limit the maximum number of sentences in a passage to be 3. Our passage retrieval component takes top-*n* ranked articles relevant to the question, and finds sentences that have at least half of the keywords in the article. We call such sentences as *important sentences*. Neighboring sentences of the important sentences with at least one keyword are merged to form a passage. To avoid having too many passages, we set the limitation of a maximum number of 5 passages from each paragraph of the article.

An essential part of our system is the extraction of subject-verb-object (SVO) triplets from sentences of passages as well as questions. The idea of SVO triplets is commonly used in typical QA systems [9] to deal with the problem of semantic symmetry. The questions "What is the role of *X* in *Y*?" and "What is the role of *Y* in *X*?" are similar at the word level, where *X* and *Y* are some keywords. However, the answers we expect from these two questions can be quite different. Our current method of extracting SVO triplets utilizes constituent trees (also referred to as parse trees) produced by the Link Grammar parser [10], which describe the syntactic structure of a sentence. However, generating the constituent trees of the sentences is computationally expensive. We relax the constraint of determining the validity of passages by finding the occurrences of subject and object extracted from the corresponding question. Passages that do not have

sentences containing the subject or object of the question are filtered out from being considered as a passage related to the question. With this relaxed constraint, we only require passages that contain subjects and objects of the questions to be valid passages, while keywords such as "mutation" that are neither the subjects nor the objects of the question are not required to be in the valid passages.
Once passages are retrieved, we used two approaches for ranking passages. Approach 1 is based on the idea of [11] that takes into account of the keyword density and distance between keywords. To give preference to longer passages, we extended the approach to take into account of the number of words and sentences in each passage. Approach 2 utilizes the rank of an article to determine relevancy of a passage retrieved from the article. The assumption is that a passage retrieved from a highly ranked article is likely to be a relevant passage. So approach 2 is computed using the reciprocal of the rank of the article in which the passage is retrieved from. The scores of passages computed by the above two approaches are normalized so that the scores can be combined for different runs. The choices of the passage ranking approaches for each run are described in the next section.

3. Runs

We submitted 3 runs for our TREC 2006 Genomics track submission. Our baseline run (run 1) involved retrieving passages from the top 100 relevant articles and filtering out passages that do not have all keywords mentioned in the passages. Our second run includes natural language processing techniques to rank passages and determine the relevancy of the passages by using the SVO triplets of passages. Due to the length of processing SVO triplets from passages, only the top 25 retrieved relevant articles are used to retrieve passages. Our third run only filters out passages that do not have the subject mentioned in the sentences of the passages, and passages are retrieved from the top 100 retrieved relevant articles.
In this paper, we emphasize on all new results following the improvements of some of the components in our system. Some of the improved components involve the conversion of HTML to XML, gene synonym finding, filtering of passages, formation of query variants and variants of keywords from the index. We used these improved components to perform evaluation of various aspects. We called this as our revised run, with the results described in section 4.

4. Results

In this section, we describe the performance of our revised run for the 26 topics instead of the original 8 topics, as there are no passages for topics 173 and 180 in the gold standard provided by TREC. The document average precision (denoted as Doc AP), passage average precision (denoted as Psg AP) and aspect average precision (denoted as Asp AP) for each of the 26 topics are described in Table 1. We noticed that there is an improvement of our results by using our new components, and we further investigated the effects of the performance for each of the major component on the overall performance of our system.

5. Analysis

In this section, we analyzed how the performance of the major components of our system affects the overall performance of our system. These components include the recognition and expansion of biological entities, HTML to XML conversion, retrieval of articles and extraction of passages.

5.1. Recognition of biological entities

Finding and recognizing, also known as normalizing, the extracted keywords from natural language questions to the intended biological entities are essential to the formation of queries, which are used to retrieve articles. In this subsection, we evaluated the correctness of the normalization of gene names, disease names and biological processes based on the reference file containing normalized biological entities for each of the questions[2].

Among the 33 genes in the 28 questions, we noticed that 19 of the genes were normalized to the correct gene. In the case of disease names, 10 out of 12 disease names in the 28 questions are normalized correctly. For biological processes, only 4 out of 17 of them were normalized to the correct biological process. Analysis of the normalization of the three types of biological entities indicates that exact matching on biological processes with standardized vocabulary is not an ideal solution to normalize biological entities. On the other hand, exact matching of disease names with MeSH disease terms is adequate, due to the fewer variation of naming conventions for diseases. Normalizing gene names with fuzzy and proximity queries result in a fair performance.

Topic ID	Doc AP	Psg AP	Asp AP
160	0.4594	0.0102	0.0789
161	0.4693	0.0053	0.0294
162	0.2000	0.0000	0.0000
163	0.2987	0.0039	0.0299
164	0.2857	0.0000	0.0000
165	0.0500	0.0000	0.0000
166	0.0000	0.0000	0.0000
167	0.5150	0.0301	0.1023
168	0.5007	0.0867	0.0289
169	0.1343	0.0011	0.0043
170	0.5000	0.0021	0.0242
171	0.0000	0.0000	0.0000
172	0.2102	0.0009	0.0075
174	0.0402	0.0010	0.0051
175	0.0526	0.0000	0.0000
176	0.0000	0.0000	0.0000
177	0.0000	0.0000	0.0000
178	0.0000	0.0000	0.0000
179	0.0000	0.0000	0.0000
181	0.4224	0.0282	0.0167
182	0.0000	0.0000	0.0000
183	0.0000	0.0000	0.0000
184	0.0000	0.0000	0.0000
185	0.0000	0.0000	0.0000
186	0.3922	0.0097	0.1200
187	0.0000	0.0000	0.0000
Mean	0.1743	0.0069	0.0172

Table 1 – The document average precision (Doc AP), passage average precision (Psg AP) and aspect average precision (Asp AP) for each of the 26 topics for our revised run

5.2. HTML to XML conversion

The variety of the HTML formats for different journals in the TREC 2006 Genomics corpus is an obstacle for our HTML to XML conversion program. Out of the 162,259 articles in the corpus, only 147,723 articles

[2] Normalized representation of the TREC questions - Alex Morgan, Stanford and MITRE http://www.stanford.edu/~alexmo/slides/NormalizedTRECGen2006Questions.xls

were stored in our Lucene index, i.e. 8.96% of the articles were not utilized by our system. The number of articles stored in our Lucene index indicates that some of the converted XML articles are not in valid XML format, such as articles containing special characters and missing certain XML tags in the course of the conversion. With respect to the topics, there is an average of 16.30% or (a median of 14.29%) of the gold standard articles not being in our index.

To investigate on how the articles not indexed by our system affects our retrieval performance, we performed an experiment on evaluating our retrieval performance based on the gold standard articles that were indexed by our system. In Figure 4, the document average precision for each topic using all 162K articles (denoted as *Doc AP*) is compared with using the 147K articles that our system indexed (denoted as *Doc AP Mod_GS*). We noticed that there was a gain in the retrieval performance in each of the topic for *Doc AP Mod_GS*. In fact, the document MAP for *Doc AP Mod_GS* went up to 0.2142, as compared to 0.1743 for *Doc AP*. We can conclude from this experiment that the failure in indexing all articles due to the issues in converting HTML to XML format has an impact on our performance.

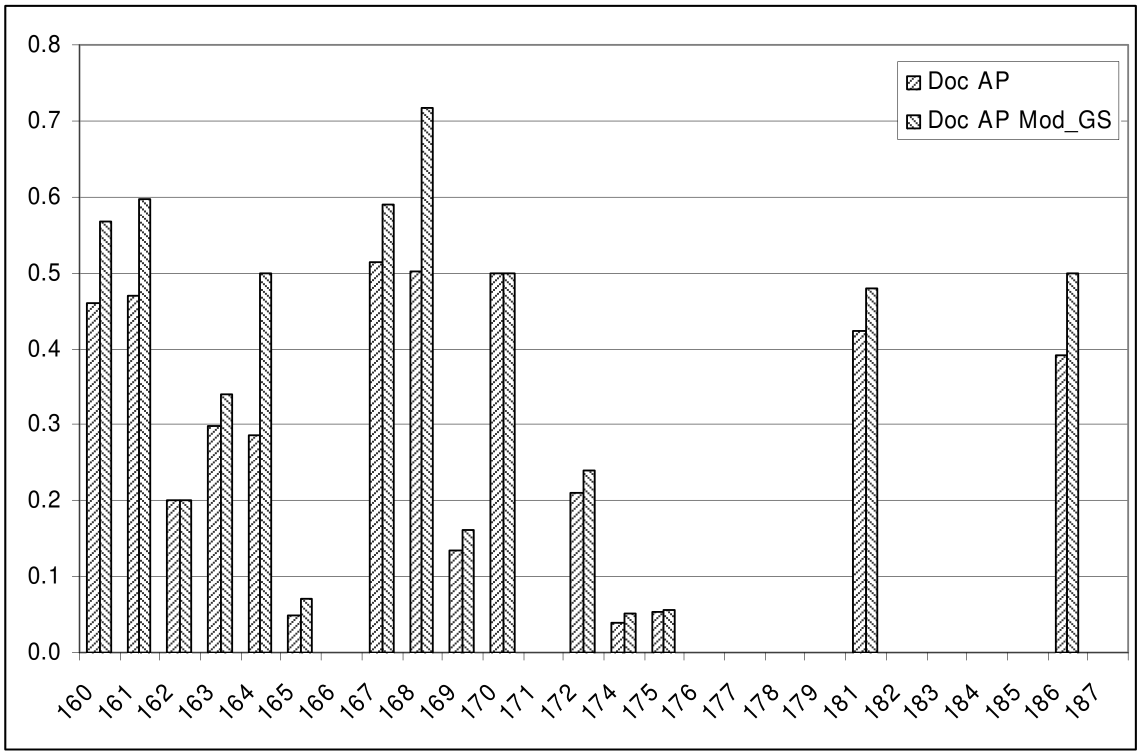

Figure 4 – The document average precision of all topics with respect to the gold standard based on all articles (denoted as Doc AP) and the gold standard based on only the articles our system indexed (denoted as Doc AP Mod_GS)

5.3. Retrieval of articles

In TREC Genomics 2006, precision of document retrieval is dependent on the submitted passages. As described in section 2.5, extraction of passages involves a process filtering out passages that are not determined as relevant with respect to the questions. This implies some of the documents can be filtered due to the absence of passages extracted from them, which can possibly be correct documents. We performed an experiment to evaluate the correctness of document retrieval without the passage filtering process. In Figure 5, we compare the document average precision for each topic without using the passage filtering process (denoted as *Bef_Psg*), i.e. retain all retrieved documents, and using the passage filtering process (denoted as *Aft_Psg*). We can see that there is an increase in document average precision for 6 of the 26 topics when the passage filtering process is used, while the document average precision decreases for another 16 topics. The performance remains unchanged for the remaining 4 topics. The document MAP for *Bef_Psg* is 0.2537, a huge increase from 0.1743 for *Aft_Psg*. The results of this experiment show that the passage extraction process can be too restricted so that documents that are in fact correct are filtered out as well.

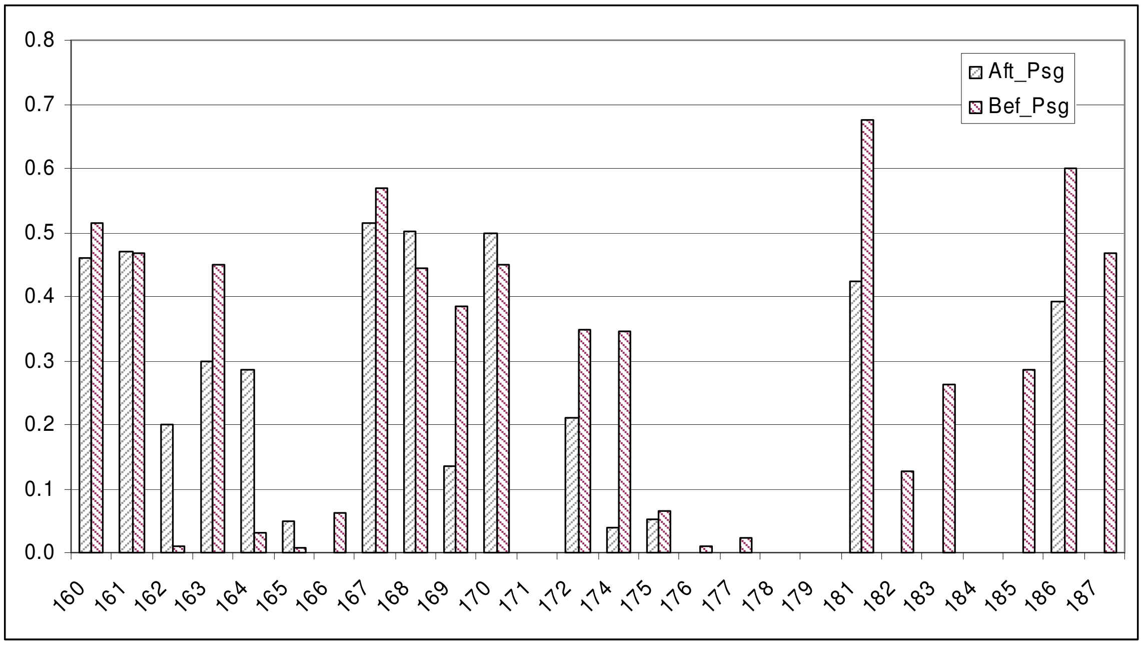

Figure 5 - The document average precision of all topics with passage filtering (denoted as Aft_Psg) and without passage filtering (denoted as Bef_Psg).

5.4. Extraction of passages

We performed analysis of our extraction performance, and we realized there is an issue with the correctness of byte start and offset for the passages. This issue is rooted from the conversion of HTML to XML format of the articles. It is unfortunate that we could not do a thorough analysis of this component, but we believe the issue of byte start and offset for the passages contribute to the poor passage average precision.

6. Discussion

From the analysis of our experiments, we realized that the component that performs the conversion of TREC Genomics articles from HTML to XML format is the major source of errors, thus reducing the performance of our system. Due to errors encountered in the course of the conversion, we noticed that almost 9% of the HTML articles were not indexed by our indexer, which expects to take valid XML files as input for indexing. A more serious flaw of the component was that there were cases when the byte start and offset of the sentences could not be computed correctly.

Other than the programming issues, the methodology of expansion of biological entities needs to be improved. Though there are cases when using the biological entities extracted from the natural language questions to form keywords is enough to form the correct queries, we realized that we performed poorly in questions that involve biological processes. This can imply our current way of expansion of biological processes is insufficient. On the other hand, we recognized from the experiments that some of the correct documents were dropped from the final results due to the absence of passages. That could mean there are issues in the process of filtering passages. Another issue with our system is that our current method of extracting passages tends to extract lengthy passages, which hurts our extraction performance when the gold standard passages tend to be concise.

Acknowledgement

We would like to acknowledge the availability of the ASU/TGen supercomputer for our processing tasks.

References

1. Fellbaum C: **WordNet: An Electronic Lexical Database**. *MIT Press* 1998.
2. Schwartz A, Hearst M: **A simple algorithm for identifying abbreviation definitions in biomedical texts**. *In Proceedings of the Pacific Symposium on Biocomputing (PSB 2003)* 2003, **8**:451-462.
3. **Lucene** [http://lucene.apache.org/java/docs/]
4. Settles B: **ABNER: an open source tool for automatically tagging genes, proteins and other entity names in text**. *Bioinformatics* 2005, **21**(14):3191-3192.
5. Ashburner M, Ball, C., Blake, J., Botstein, D., et al.: **Gene Ontology: tool for the unification of biology**. *Nature Genetics* 2000, **25**:25 – 29
6. Huang M, Zhu X, Li M: **A hybrid method for relation extraction from biomedical literature**. *International Journal of Medical Informatics*, **In Press, Corrected Proof**.
7. Bhalotia G, Nakov PI, Schwartz AS, Hearst MA: **BioText team report for the TREC 2003 Genomics track**.
8. Aronson AR: **Fusion of knowledge-intensive and statistical approaches for retrieving and annotating textual genomics documents**. In: *NIST Text Retrieval Conference (TREC).* Gaithersburg, Maryland: National Institute of Standards and Technology; 2005.
9. Katz B, Lin J: **Selectively Using Relations to Improve Precision in Question Answering**. In: *Proceedings of the EACL 2003 Workshop on Natural Language Processing for Question Answering: 2003; Budapest, Hungary*; 2003.
10. Grinberg D, Lafferty, J. , Sleator, D.: **A Robust Parsing Algorithm For LINK Grammars**. In.: CMU; 1995.
11. Lee GG, Seo J, Lee S, Jung H, Cho B-H, Lee C, Kwak B-K, Cha J, Kim D, An J *et al*: **SiteQ: Engineering high performance QA system using lexico-semantic pattern matching and shallow NLP**. *In Proceedings of the Tenth Text REtrieval Conference (TREC 2001)* 2001.