Combining Multiple Resources, Evidences and Criteria for Genomic Information Retrieval

Luo Si[1], Jie Lu[2] and Jamie Callan[2]
[1]Department of Computer Science, Purdue University, West Lafayette, IN 47907, USA
lsi@cs.purdue.edu
[2]Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA
{jielu, callan}@cs.cmu.edu

ABSTRACT
We participated in the passage retrieval and aspect retrieval subtasks of the TREC 2006 Genomics Track. This paper describes the methods developed for these two subtasks. For passage retrieval, our query expansion method utilizes multiple external biomedical resources to extract acronyms, aliases, and synonyms, and we propose a post-processing step which combines the evidences from multiple scoring methods to improve relevance-based passage rankings. For aspect retrieval, our method estimates the topical aspects of the retrieved passages and generates passage rankings by considering both topical relevance and topical novelty. Empirical results demonstrate the effectiveness of these methods.

1. INTRODUCTION
We describe in this paper the design of the system built for the passage retrieval and aspect retrieval subtasks of the TREC 2006 Genomics Track. The modules provided in the Lemur toolkit for language modeling and information retrieval (version 4.2)[1] constitute the backbone of our system. The Indri index was chosen for its support of conveniently indexing and retrieving various fields of the documents, and its rich query language that easily handles phrases and structured queries. New methods and tools were developed to equip the system with enhanced capabilities on collection pre-processing, indexing, query expansion, passage retrieval, result post-processing, and aspect retrieval.

Particularly, based on the success of query expansion indicated by the results of previous Genomics tracks, we continue the exploration of incorporating domain knowledge to improve the quality of query topics. Acronyms, aliases, and synonyms are extracted from external biomedical resources, weighted, and combined using the Indri query operators to expand original queries. A hierarchical Dirichlet smoothing method is used for utilizing passage, document, and collection language models in passage retrieval. A post-processing step that combines the scores from passage retrieval, document retrieval, and a query term matching-based method further improves the search performance. An external database constructed from MEDLINE abstracts is used to assign MeSH terms to passages for estimating topical aspects, based on which passage rankings are generated for aspect retrieval by optimizing topical relevance while reducing topical redundancy.

The following section describes various modules of the system developed for genomic passage retrieval and aspect retrieval. Section 3 presents some evaluation results, and Section 4 concludes.

2. SYSTEM DESCRIPTION
In this section we elaborate on the methods and tools used in different modules of our system, focusing on query expansion, post-processing, and aspect retrieval.

2.1 PRE-PROCESSING
The corpus for the TREC 2006 Genomics Track includes 160,472 biomedical documents from 59 journals. The documents are in html format. The formats of the html files are similar but not identical. For example, some documents use the tag "BIB" to indicate reference while other documents use the tag "Reference".

Since the TREC 2006 Genomics Track mainly focuses on passage retrieval, it is important to design an effective method to segment the biomedical documents into passages. A passage extraction method is developed to consider both paragraph boundaries and sentence boundaries. As required by the TREC 2006 Genomics Track, a biomedical document is first segmented into many paragraphs based on the tags "<p>"

[1] http://www.lemurproject.org/

or "</p>". Special treatment is applied to the reference part to make sure that the paragraphs for references are separate from the paragraphs in the main text part. Furthermore, the html tags and other irrelevant contents (e.g., scripts) are removed from the extracted paragraphs. Then each paragraph is segmented into many sentences with a modified version of a perl script[2]. The sentences are assigned to individual passages until the length of a passage exceeds 50 words. There is no overlap among the passages, which means that two consecutive passages contain different sentences. This procedure is applied to all the biomedical documents in the Genomics Track. Altogether, there are 2.1 million passages extracted from the biomedical documents. Each document on average contains 132 passages. A new version of each document is built by merging the passages extracted from the document. The identity of each passage is preserved by using special tags. A new text collection is built from all the new documents.

2.2 INDEXING

All the new documents generated by the pre-processing module are indexed by the indexing module. The document parser in the indexing module provided by the Lemur toolkit is modified to further process potential biomedical acronyms. Specifically, two additional operations are applied to the tokens recognized by the text tokenizer: *segmentation* and *normalization*.

Segmentation takes as input a token free of space or punctuation characters (including hyphen) and looks for boundaries between any two adjacent characters of the token, based on which it segments the token into multiple tokens. A boundary occurs between two adjacent characters of the token in the following cases: i) one is numeric and the other is alphabetic; or ii) both are alphabetic but of different cases, except for the case that the characters are the first two characters of the token with the first in uppercase and the second in lowercase. For example, taking the token "*hMMS2*" as input, the output of the segmentation operation includes three tokens "*h*", "*MMS*", and "*2*".

Normalization converts Roman digits into their corresponding Arabic numbers. For instance, the "*II*" in "*hMMS II*" is converted to "*2*" by the normalization operation.

2.3 QUERY EXPANSION

The query expansion module parses each of the original queries and utilizes several external resources to incorporate domain knowledge by expanding queries with acronyms, aliases, and synonyms. Both the original terms and the expanded terms are weighted. Each original term and its expanded terms are combined using the weighted synonym operator "*#wsyn*" into a "*#wsyn*" expression, and different "*#wsyn*" expressions for a query are combined using the "*#weight*" belief operator of the Indri query language[3].

The data provided by AcroMed[4], LocusLink[5], and UMLS[6] are processed to create three lexicons. In the AcroMed lexicon, entries are indexed by technical terms or phrases, and each entry is a list of acronyms associated with the corresponding technical term/phrase, accompanied by the frequencies of such associations. In the LocusLink lexicon, entries are indexed by acronyms, and each entry is a list of aliases that are only associated with the corresponding acronym but no other acronyms. In the UMLS lexicon, entries are indexed by technical terms or phrases, and each entry is a list of synonyms associated with the corresponding technical term/phrase.

For example, the phrase "*huntington's disease*" has an entry in the AcroMed lexicon "*1582 hd 2 h.d.*", indicating that the acronym "*hd*" is associated with "*huntington's disease*" 1582 times while "*h.d.*" is associated with "*huntington's disease*" only twice. An example for the LocusLink lexicon is that the acronym "*psen1*" corresponds to a list of aliases "*ps-1, pre1, psen, zfps1, zf-ps1*". The entry provided by UMLS for the phrase "*mad cow disease*" is "*bovine spongiform encephalopathy, bse, bovine spongiform encephalitis*", excluding the variants generated by varying the form or order of the words.

For each query, the lexicons are applied in the order of AcroMed, LocusLink, and UMLS for query expansion. Generally speaking, AcroMed is used to find the acronyms associated with a technical term or

---

[2] http://l2r.cs.uiuc.edu/~cogcomp/atool.php?tkey=SS
[3] http://www.lemurproject.org/lemur/IndriQueryLanguage.html
[4] http://medstract.med.tufts.edu/acro1.1/
[5] http://www.ncbi.nlm.nih.gov/projects/LocusLink/
[6] http://www.nlm.nih.gov/research/umls

phrase (gene or disease name) that occurs in the query. LocusLink is used to find the aliases of the acronyms identified by AcroMed. UMLS is used to find the synonyms of the technical terms or phrases not recognized by AcroMed or LocusLink. In addition, commonly observed synonyms for some "function" words such as "role" and "disease" that occur in the query are added as expansion terms, and multiple surface forms of the acronyms or aliases are included. Specifically, the following steps are performed in order to generate an expanded query:

1. If a string of word(s) in the original query has an entry in AcroMed, the string is considered a technical term/phrase. The acronyms in its entry in AcroMed whose frequencies are above a threshold (25) are added as expansion terms, with the weight of each acronym proportional to its frequency, normalized by the maximum frequency in the entry so the maximum weight is 1.00.
2. If an acronym included in the expanded query can locate in LocusLink its aliases, the aliases are included and their weights are equal to the weight of the acronym.
3. For the strings of words that occur in the original query but are not expanded by steps 1-2, UMLS is used to find possible synonyms to add to the expanded query, each with a weight of 0.50.
4. The same operations of *segmentation* and *normalization* used in the document parser are applied to the acronyms in the original query, and the acronyms and aliases in the expanded query. In addition, the segmented tokens of an acronym or alias output by the segmentation operation are fed into the *assembly* operation, which assembles the tokens to produce multiple variants with the same weight as the acronym or alias. For example, the acronym "*hMMS2*" is segmented into "*h*", "*MMS*", and "*2*", which are assembled into "*hmms2*", "*h mms2*", "*hmms 2*", and "*h mms 2*" in phrasal representations.
5. Each word or phrase that occurs in the original query and is expanded by the above steps uses the weighted synonym operator "*#wsym*" to combine itself (with the weight 1.00) and its expanded acronyms, aliases or synonyms (with the corresponding weights described in the above steps). The overall weight of the "*#wsym*" expression is 2.00.
6. A few "function" words commonly observed for genomic retrieval are grouped and the words within each group are regarded as synonyms. If any word in the original query occurs in a synonym group, the other words in the same group are added as expansion terms with weights half the value of the weight given to the original word. The weighted synonym operator "*#wsym*" is used to combine these terms. Particularly, two groups of synonyms are used: {role, affect, impact, contribute} and {disease, cancer, tumor}. The first group of synonyms has an overall weight of 1.00 for the "*#wsym*" expression, while the second group of synonyms has an overall weight of 2.00. Therefore, if the words "*role*" and "*cancer*" are in the original query, they will be expanded into "*1.00 #wsyn(1.00 role 0.50 affect 0.50 impact 0.50 contribute)*" and "*2.00 #wsyn(1.00 cancer 0.50 tumor 0.50 disease)*".
7. Finally, the "*#wsym*" expressions created during steps 1-6 and the words left in the original query are combined using the "*#weight*" belief operator, with the weight of each unexpanded word in the original query set to 2.00.

2.4 PASSAGE RETRIEVAL

A variant of the language modeling method is used for passage retrieval. This method extends the traditional Dirichlet smoothing method [Zhai and Lafferty, 2001] with hierarchal smoothing. Specifically, the log-likelihood of generating query Q from the j^{th} passage of the i^{th} document is calculated as follows:

$$\log\left(P(Q|\{\overrightarrow{psg}_j,\vec{d}_i\})\right)=\sum_w \log\left(\frac{psg\_tf_j(w)+d\_probtf_i(w)*u_1}{\left|\overrightarrow{psg}_j\right|+u_1}\right)*qtf(w)$$

where $psg\_tf_j(w)$ and qtf(w) indicate the term frequency of word w in the j^{th} passage and in the user query respectively, $\left|\overrightarrow{psg}_j\right|$ is the length of the j^{th} passage, u_1 is a normalization constant (set to 200 empirically), and $d\_probtf_i(w)$ introduces the evidence of word w from the document level, which is estimated as:

$$d\_probtf_i(w)=\frac{dtf_i(w)+p_c(w)*u_2}{\left|\vec{d}_i\right|+u_2}$$

where $dtf_i(w)$ represents the term frequency of word w in the i^{th} document, $\left|\vec{d}_i\right|$ is the length of the i^{th} document, $p_c(w)$ is the word probability in the whole text collection, which is calculated by the relative frequency of the word in the collection, and u_2 is a normalization constant (set to 1000 empirically).

2.5 POST-PROCESSING

The retrieved passages are first cleaned to remove passages that don't contain meaningful content and to get rid of unnecessary characters at the beginning or the end of each passage. Specifically, the following three steps are performed:

1. Simple word patterns are used to detect passages for acknowledgment, abbreviation list, keyword list, address, figure list, and table list, and these passages are discarded.
2. A throw-away list is constructed which includes words commonly used in section titles of papers, such as "introduction", "experiments", "discussions", etc. and their morphological variants. If these words occur at the beginning or the end of a retrieved passage, they are removed.
3. Regular expression patterns are used to identify tags, references, figures, tables, and punctuations at the beginning or the end of a retrieved passage in order to remove them.

The cleaned passages are rescored by combining the scores obtained for document retrieval, passage retrieval, and the query term matching-based scores recalculated for the passages. The details of calculating the query term matching-based score for a cleaned passage are given below.

The terms of an expanded query are classified into three types, namely type-0, type-1, and type-2. Type-0 terms are terms that occur in a "function" word list containing common terms for genomic queries such as "role", "contribute", "affect", "develop", "interact", "activity", "mutate", etc. and their morphological variants. Type-1 terms are non-type-0 terms added to the query during query expansion. Type-2 terms are non-type-0 terms in the original query.

The query term matching-based passage score is calculated by accumulating the adjusted frequencies of the matched query terms in the passage, which are computed differently for different types of terms:

$$\text{Type-0: } \min\{\text{weight}, 0.50\}\text{; Type-1: } 0.50\text{; Type-2: } \sum_{i=1}^{tf} \text{weight} \times (0.25)^{i-1}$$

where weight is each term's weight in the expanded query, and tf is the frequency of a term in the passage.

The query term matching-based scores of the retrieved passages for a query are normalized by:

$$S_{normalized} = \frac{S\text{-}2.00}{\min\{4.00, S_{max}\}\text{-}S_{min}}$$

where S is the score of a passage before normalization, S_{max} and S_{min} are the maximum and minimum scores of the retrieved passages for a query.

Because the weight of a type-2 term in the query is 2.00, the minimum score of a retrieved passage that matches at least one type-2 term is 2.00. If a retrieved passage fails to match any of the type-2 terms in the query, it is very likely to have a negative score unless it matches multiple type-1 terms. 4.00 is the minimum score of a retrieved passage that matches at least two distinctive type-2 terms. Using $\min\{4.00, S_{max}\}$ instead of S_{max} for normalization is to downgrade the differences between the passages that match two or more distinctive type-2 terms since these passages probably have the same degree of relevance.

The original scores of the retrieved passages and the original scores of their corresponding documents are normalized using the standard max-min normalization:

$$S_{normalized} = \frac{S\text{-}S_{min}}{S_{max}\text{-}S_{min}}$$

The final score of a retrieved passage is a weighted linear combination of the normalized passage retrieval score (with a weight of 0.85), the normalized document retrieval score of the document which contains the passage (with a weight of 0.05), and the normalized query term matching-based passage score (with a weight of 0.10). Passages for a query are reranked using the final scores.

2.6 ASPECT RETRIEVAL

Although topical relevance is the most important factor in information retrieval, an effective information retrieval system also needs to consider the topical aspects of the retrieved information. For example, a biomedical researcher would like to avoid seeing similar or even duplicated contents. Therefore, the redundant information should be removed. The retrieval performance is generally considered better if the top-ranked documents or passages are not only relevant but also cover a wide range of aspects.

Our aspect retrieval module considers both topical relevance and the coverage of aspects. Topical relevance can be directly measured by the scores from the passage retrieval module (with or without rescoring in the post-processing module). Topical aspects of each retrieved passage can be based on the MeSH (i.e., Medical Subject Heading) terms which best describe the semantic meaning of the passage. However, since MeSH terms are only associated with a whole document in the MEDLINE database, the MeSH terms of each retrieved passage need to be estimated.

The process of assigning appropriate MeSH terms to retrieved passages is viewed as a multiple-category classification problem in this work. Specifically, each retrieved passage is regarded as a query to locate similar documents from a subset of the MEDLINE database. As the similar documents found tend to share similar MeSH terms with this passage, we can assign MeSH terms to the passage based on the MeSH terms associated with the similar documents.

In our work, the subset of the MEDLINE database is formed by the data for the adhoc retrieval task of the TREC 2003 Genomics Track, which include 4,491,008 MEDLINE abstracts published during 1993-2003. Most of the abstracts are associated with MeSH terms assigned by domain experts. After some text pre-processing such as removing stopwords and stemming, the title field and the body text field of each abstract are indexed. The average document length is about 160.

For each user query, the content of each of the 500 top-ranked passages from the ranked list of passage retrieval is obtained and processed by removing stopwords and stemming to form a document query, which is used for locating similar documents from the subset of the MEDLINE database. The Okapi formula is used to retrieve the 50 top-ranked MEDLINE abstracts as the most similar documents. Then, the most common 15 MeSH terms are extracted from these MEDLINE abstracts. Each MeSH term is associated with a weight based on the number of occurrences of this term in the top-ranked MEDLINE abstracts. These 15 MeSH terms are further represented by a vector in a vector space formed by all the MeSH terms. We utilize the TF.IDF method as the term weighting scheme of the vector space:

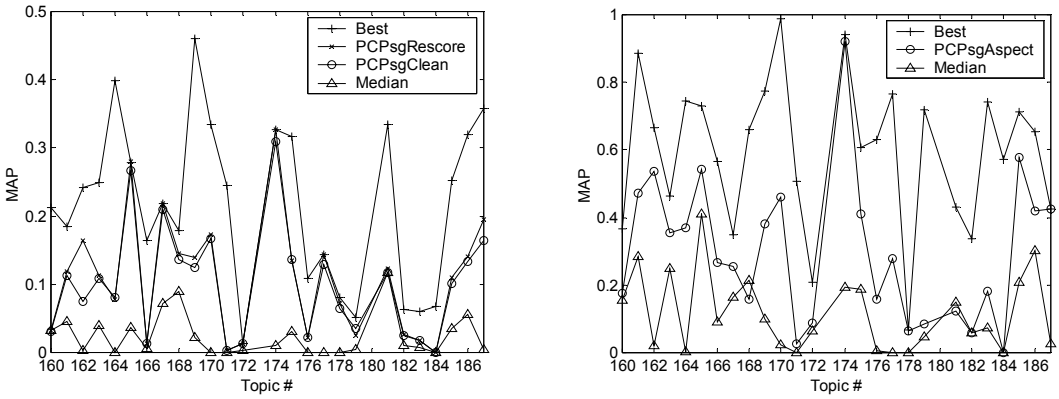
$$\text{val}(w)=\text{tf}(w) * \log \frac{\text{ctf}(w)}{|c|}$$

Where tf(w) is the number of occurrences of a specific MeSH term within the top-ranked MEDLINE abstracts, ctf(w) is the number of occurrences of this MeSH term within the subset of the MEDLINE database, and |c| is the total number of occurrences of all MeSH terms in the database.

In addition to extracting representative MeSH terms for each passage by analyzing the content similarity between the passage and the MEDLINE abstracts, the MeSH terms associated with the biomedical document that contains this passage can also be used to generate the MeSH terms for this passage. In order to utilize both evidences, we use a linear form to combine the vector representations of the MeSH terms from these two sources. More specifically, the two vectors are normalized respectively, and the normalized vectors are summed together with an equal weight (i.e., 0.5). The final representation is obtained by normalizing the sum.

The procedures described in the above paragraphs can be used to derive the MeSH representations for all the top-ranked passages (top 500 in this work) for a user query. These MeSH representations reflect the topical aspects of the passages. With both the topical aspects and the topical relevance information (i.e., passage retrieval scores), a new ranked list can be constructed by reranking the passage retrieval result.

Particularly, a procedure similar to the maximal marginal relevance method [Carbonell and Goldstern, 1998] is adopted here. This is a gradient-based search approach. At each step, a document is selected and added to the bottom of the current reranked list. A combination score is calculated for each passage by considering both the topical relevance information and the novelty information of the topical aspects with respect to the current reranked list (i.e., the selected passages).



(a) passage retrieval **(b) aspect retrieval**

Figure 1 The performance of our system compared with the best and median performance.

$$S_{comb}(\overrightarrow{psg_i}) = \lambda S_{rel}(\overrightarrow{psg_i}) - (1-\lambda) \max_{psg_j \in Sel} (Sim(\overrightarrow{psg_i}, \overrightarrow{psg_j})) \quad \overrightarrow{psg_i} \notin Sel$$

where S_{comb} represents the combination score, S_{rel} represents the normalized passage retrieval score (i.e., divided by the maximum score), λ is the factor to adjust the relative weights of the topical relevance information and the topical aspect information (set to 0.5 in this work), Sel is the current reranked list of the selected passages, Sim is a function that calculates the cosine similarity between the MeSH term representations of two passages to reflect their topical similarity.

At each step, the combination scores are calculated for the passages that are not in the current reranked list for a query. Since the passage with the maximum combination score reflects a good trade-off between topical relevance and topical novelty, it is added to the bottom of the current reranked list. Note that the above procedure is only applied to rerank the top 1-500 passages. The top 501-1000 passages are still the same as the passages in the ranked list of passage retrieval solely based on topical relevance.

3. EVALUATION

We submitted three runs using automatically constructed queries. "PCPsgClean" used the automatically expanded queries to conduct passage retrieval, and the retrieved passages were cleaned during post-processing but not rescored. "PCPsgRescore" reranked the results obtained from "PCPsgClean" using the rescoring method described in Section 2.5. "PCPsgAspect" further processed the results from passage retrieval to optimize performance for aspect retrieval based on the method described in Section 2.6. None of our results were optimized for document retrieval performance. Figure 1 shows the performance of our system compared with the best and median performance for passage retrieval and aspect retrieval.

4. CONCLUSION

Our results for passage retrieval show that query expansion based on external biomedical resources is an effective technique, and the hierarchical Dirichlet smoothing method that utilizes passage, document, and collection language models works reasonably well for passage retrieval. Reranking the retrieved passages by combining scores from passage retrieval, document retrieval, and the query term matching-based rescoring consistently further improves the performance of passage retrieval. Our method of estimating topical aspects of the retrieved passages and generating passage rankings by considering both topical relevance and topical novelty has an acceptable performance but still leaves room for improvement.

REFERENCES

[1] Carbonell, J. and Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM.

[2] Zhai, C. X. and Lafferty, J. (2001). A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM.