York University at TREC 2004: HARD and Genomics Tracks

Xiangji Huang[1], Yan Rui Huang[2], Miao Wen[2] and Ming Zhong[2]

[1]School of Information Technology, York University, Toronto, Ontario, Canada
e-mail: jhuang@yorku.ca

[2]Department of Computer Science, York University, Toronto, Ontario, Canada
e-mail: yhuang, mwen, ming@cs.yorku.ca

Abstract

York University participated in HARD and Genomics tracks this year. For both tracks, we used Okapi BSS (basic search system) as the basis. Our experiments mainly focused on exploiting various methods for combining document and passage scores, new term weighting formulae and feedback methods for query expansion. For HARD track, we built two levels of indexes, and search against both indexes for each topic. Then we combine these two searches into one. For Genomics track, we used a new strategy to automatically expand search terms by using relevance feedback and combined retrieval results from different fields into the final result. We achieved good results on the HARD task and average results on the Genomics task. For the HARD passage level evaluation, the automatic run 'yorku04ha1' obtained the best result (0.358) in terms of Bpref measure at 12K characters. The evaluation results show that *Algorithm* 1 is more effective than *Algorithm* 2 for the passage level retrieval,

1 Introduction

This is the first time York participated in TREC. We participated in both High Accuracy Retrieval from Documents (HARD) track and Genomics track. For both tracks, we used Okapi BSS (basic search system) as the basis. Our experiments mainly focused on exploiting various feedback methods for query expansion and term weighting formulae.

For the HARD task, both document level index and passage level index are built for retrieval purpose. We use Okapi BM25 for passage retrieval, but for document level retrieval, we applied a modified version of BM25, named BM50, by adding a correction factor which is based on the length of document. This correction factor is added at the end of the usual BM25 function, and serves as an adjusting factor that gives relative low weight to those documents with considerably short or long lengths.

Our experiments at the genomics track mainly focused on the following methodology: (1) We generated initial query terms automatically. (2) We used a new strategy to automatically expand search terms by using relevance feedback. (3) We built five indexes on the fields of "PMID", "Title", "Abstract", "Mesh heading" and "NameOfSubstance" for retrieval; (4) We designed a new retrieval strategy to combine retrieval results from different fields as the final result.

The rest of the paper is organized as follows. We first give a brief description of the Okapi information retrieval system in Section 2. Then, HARD track and Genomics track are presented in Section 3 and 4. Following that, experimental results for both HARD and Genomics tracks are provided in Section 5. Finally, the conclusion and future work are given in Section 6.

2 Retrieval Based on Probabilistic Model

We used Okapi BSS (Basic Search System) as our main search system. Okapi is an information retrieval system based on the probability model of Robertson and Sparck Jones [6]. The retrieval documents are ranked in the order of their probabilities of relevance to the query. Search term is assigned weight based on its within-document term frequency and query term frequency. There is a Go-See-List (GSL) file containing synonym for indexing. The weighting function used is BM25 [2]:

$$w = \frac{(k_1+1)*tf}{K+tf} * \log\frac{N-n+0.5}{n+0.5} * \frac{(k_3+1)*qtf}{k_3+qtf} \quad \oplus \quad k_2 * nq * \frac{(avdl-dl)}{(avdl+dl)} \quad (1)$$

where w is the weight for each query term, N is the number of indexed documents in the collection, n is the number of documents containing a specific term, tf is within-document term frequency, qtf is within-query term frequency, dl is the length of the document, $avdl$ is the average document length, nq is the number of query terms, the k_is are tuning constants (which depend on the database and possibly on the nature of the queries and are empirically determined), K equals to $k_1 * ((1-b)+b*dl/avdl)$, and $\oplus$ indicates that its following component is added only once per document, rather than for each term. In our experiments, the values of k_1, k_2, k_3 and b in the BM25 function are set to be 1.2, 0, 8 and 0.75 respectively.

3 HARD Track

The main task of HARD is to achieve high accuracy retrieval from documents. For this track, we explored some techniques that could enhance the search results. We mainly focused on the usage of the query information provided and interactive user feedback. We also investigated the effectiveness of different indexing and term weighting methods. Two levels of search, document-level and passage-level, are performed for each topic. Then we combined these two search results into one. This year, we submitted one baseline run, two clarification forms and four final runs. For the four final runs, two of them are automatic runs and the other two are manual runs. Our experiments were conducted on a double-processor server which has 2 Intel Xeon 2.40GHz CPU and 2G memory. The version of Linux kernel we used is version 2.4.26.

3.1 System Description

Figure 1 shows the system architecure of our search system. Search terms are first generated by parsing the original topics, metadata and the interactive user feedback from the clarification forms. These terms are used to generate initial search results. After initial results are generated, we use the blind feedback to do the query expansion. Then we choose the top 80 terms as the finalized search terms. These search terms are used to search the database again and newly generated results will be used as the final results.

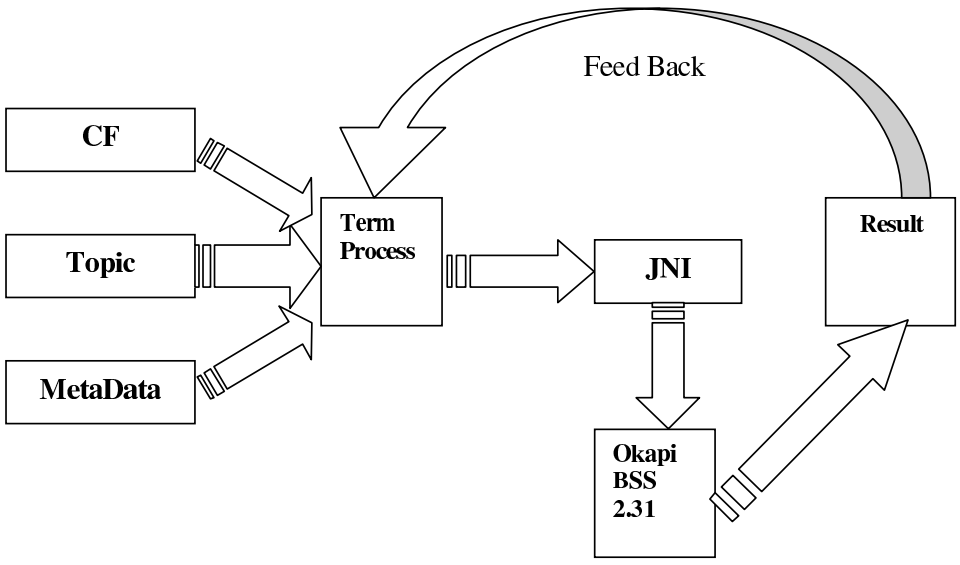

Figure 1: System Architecture

3.2 Topics, Clarification Forms and Metadata

There are 20 training topics and 50 evaluation topics. For the baseline run, each topic contains a short title, a sentence long query description and a paragraph long narrative. After the baseline was submitted, more detailed information called metadata was added for each topic. The metadata includes the following fields: Genre, Geography, Granularity, Familiarity, Subject, and Related Text. However, only the granularity, geography and related text information were used in our HARD experiment. The granularity information was used to decide if we should have document level retrieval or passage level retrieval. The geography information was used to filter out some documents[1] and the related text information was used to automatically expand the query terms.

Two sets of clarification forms (CFs) were submitted. The first one concentrated on short paragraph feedback and the second one concentrated on keyword feedback within the context. The CF results were then used in query expansion. The main goal of using CFs is to get interactive feedback from the users. Both positive and negative feedback can be obtained from CFs. One of the automatic run was generated by using only the positive feedback and the other one was generated by using both positive and negative feedback.

3.3 Term Selection and Expansion

The first task we need to perform before searching is to find the right terms to search. For the baseline run, there are only three fields provided for each topic: title, description and narrative. The title is a very short phrase, the description and narrative are provided in a natural language format. All the search terms should be extracted from these three fields. We use a list of stop words to filter out the words which will not be helpful for the topic searching. For each search term, we count the total number of occurrence within the topic and the number of occurrence in each field. The following structure is used for each term.

Search Term	Total No. of Occurrences	No. of Occurrences in Title	No. of Occurrences in Description	No. of Occurrences in Narrative

The above structure is extendable. For the baseline run, we only count the first three fields. After we get the feedback of the clarification form and related text, we extracted more terms from these two resources and expand the search terms. Then one counter for each new field is added for every search term, and total number of occurrences is updated correspondingly. These counters are indicators which show

[1]The Geography parameter restricts the region discussed in the returned articles. For example, articles concerned with the state of affairs in other countries will not be welcome returns for topics in which the US value has been selected. It is used as the U.S. filter in our experiments.

the importance of the term and will be used later to address the weight of the terms. More search terms can be added by using the blind feedback technique. So the final terms for searching include the terms extracted from original topics, the terms extracted from the metadata, the terms extracted from clarification forms and the terms extracted from the blind feedback. Currently, all the search terms are single words. No phrase has been extracted from the topic.

3.4 BM50 Function and Document Length Adjustments

The component shown below is called *correction factor* which was designed to take into account the length of a document. The value of the correction factor decreases with dl, from a maximum as $dl \rightarrow 0$, through zero, at which $dl = avdl$, and to a minimum as $dl \rightarrow \infty$

$$k_2 * nq * \frac{(advl - dl)}{(advl + dl)} \tag{2}$$

This design of the correction factor assumes that, the shorter the document is, the more value the correction factor should have, i.e., the terms in a short document becomes more significant for that document and thus the ranking of a short document is improved. However, under certain circumstance, the correction factor is misleading. For example, when the document length is 0 or extremely small, it doesn't contain sufficient information, and most likely this document is a noise. In this case, it should not be considered as the relevant document. Thus, it should not gain more weight from the correction factor than other documents [3].

In order to find a better solution, we conducted some analysis of the 2004 Hard Corpus. The statistical information is shown in Table 1 and 2:

Min Length	Max Length	Average Length
2	63870	2188.0344

Table 1: 2004 Hard Corpus

Min Length	Max Length	Average Length
131	26461	3320.695

Table 2: 2004 relevant dataset from the training topic

From the two tables above, we can find that the average documents length is shorter than the average relevant documents length. To illustrate the results graphically, we re-plot the data in Figure 2 and 3. Figure 2 shows the distribution of the

whole documents, while Figure 3 shows the distribution of the relevant documents. What we can do here is to find a function which best fits the curve of the relevant documents for the training topics so that a negative factor will be given for a very short document. For a document whose length is equal to the average length of the relevant document, a maximum correction factor will be assigned.

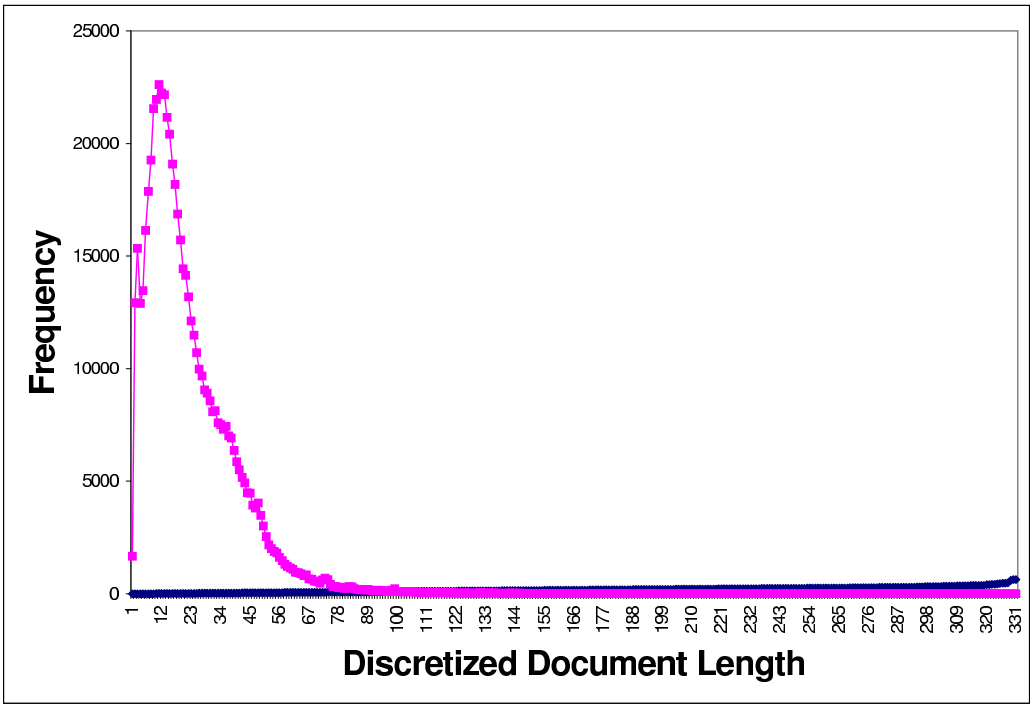

Figure 2: Distribution curve for the whole documents

Using the statistic software TableCurve 2D from Systat[2], we found the following function which best fits the curve of the relevant dataset for the training topics:

$$y = 4 * a * e^{-\frac{x-b}{c}} * (1 - e^{-\frac{x-b}{c}}) \tag{3}$$

This function is added to the end of the BM25 function. Thus, the refined BM25 function, BM50, is as follows:

$$w = w_1 + w_1 * y/k_l \tag{4}$$

[2]http://www.systat.com

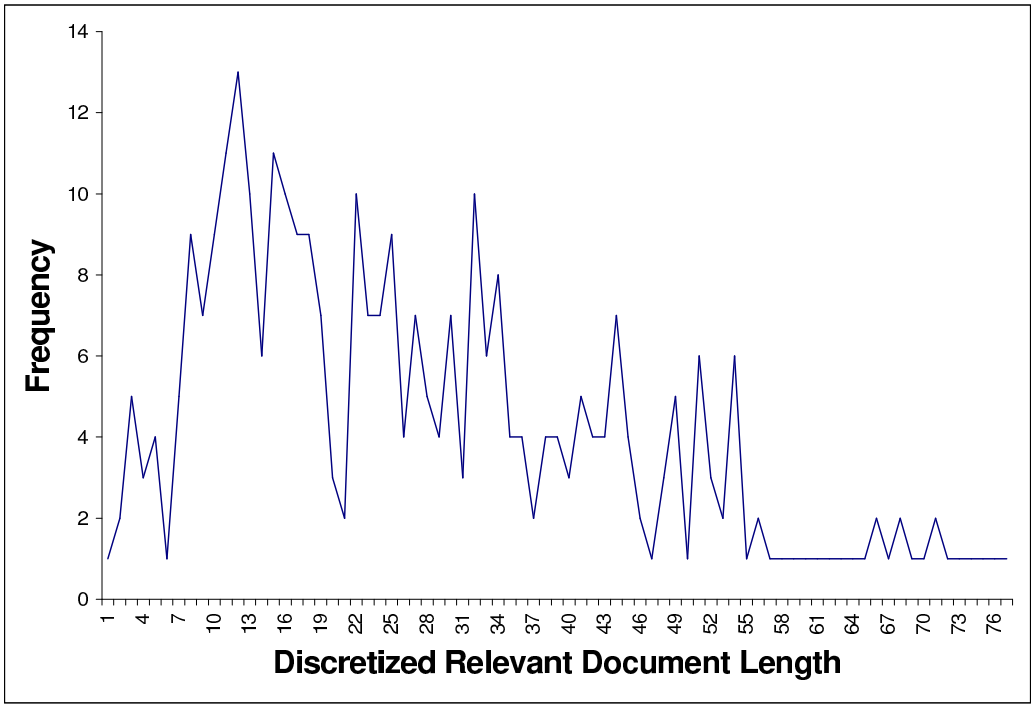

Figure 3: Distribution curve for the relevant documents

Where

$$w_1 = \frac{(k_1+1)\,tf}{K+tf} log \frac{N-n+0.5}{n+0.5} \frac{(k_3+1)\,qtf}{k_3+qtf} \tag{5}$$

The y/k_l component indicates the percentage of the final weights which is contributed by the correction factor. Here, we set a=13, b=1.3 and c=44 to match the distribution curve we get from the relevance dataset and set k_l=50 to make the correction factor contributes no more than 26% of the final weight.

The experiments show that the new correction factor changes the weight of the documents. The weight of documents which have very short and very long lengths will be decreased compared to the weight get from the original BM25 function.

3.5 Search

After search terms are extracted, we use them to search against the HARD corpus by using Okapi. We implemented a JNI (Java Native Interface) for our search engine

to interactive with Okapi BSS[3]. For the baseline run, we only use terms extracted from the original topic to search against the HARD corpus.

For the final runs, we first expand our query terms by using clarification forms and metadata. The new list of terms are searched against the HARD corpus, and the statistical information of these terms is collected. Each term's weight is calculated based on the Okapi weighting function and the importance of the term (e.g., the query term frequency). Then we choose the top 80 terms as our finalized search terms according to their weights.

After we finalize the top 80 search terms, an U.S filter is used first to filter out the documents if there is a geographical restriction on the topic. The U.S filter contains a list of states and cities of the United States, and all those documents which do not contain any name from the list are considered to be non-relevant and will be filtered out if there is a geography restriction. For the manual run, we also use a key filter. The format of the key filter is as the following: each line contains several keywords which can be a single term or phrase. The keywords in a single line are "OR" related. This key filter can be used to filter out some unrelated documents. Since the filter can be highly restrictive, it should be used with caution. In the manual experiment, we only applied it to the first round of search.

Other than the interactive user feedback that we obtain from the CFs, we also use blind feedback to expand our query terms. From the initial search result, we select the top 50 documents, calculate the RSV values of the terms within these documents, and add the top ranked terms into the existing search terms.

3.6 Combining Document and Passage Scores

The following two algorithms were applied in the final submissions. Both of them are based on the same topics and CF results.

Algorithm 1: For each topic, we do both document level search and passage level search. Then we combine these two searches into one. Our basic assumption for this combination is: if an article is hit by both searches, it should be assigned more weight than others that are hit by only one search. After initial results are generated, we use blind feedback to do the query expansion. Then we generate the final results by using the same algorithm. BM25 was used for passage level search and BM50 was used for document level search.

Algorithm 2: For each topic, we do only document level search or passage level search according to the value of "retrieval-element" (document or passage). The search terms are automatically extracted from topics and CF results. The terms extracted from CF results were classified into two sets: positive and negative terms. We assign positive weights to positive terms and negative weights to negative terms. BM25 was used for both document level search and passage level search.

[3]Okapi BSS is written in C and our main search engine is written in Java.

For *Algorithm* 1, we use different merge functions to update the weights for document and paragraph by combining the results from both indexes. If the granularity is "document", the following merge function is used:

$$W_{dnew} = (W_d + \frac{\sum_{x=1}^{k} W_{d.x}}{|P|}) * \log_{10}(10 * |P|) \tag{6}$$

where W_{dnew} is the new weight of the document, W_d is the weight obtained from the document level index, $W_{d.x}$ is the weight obtained from the paragraph level index, x ranges from 1 to k, where k equals to the total number of paragraphs retrieved from this document in the top 1000 paragraphs from the paragraph level index. $|P|$ is the total number of paragraphs retrieved from this document.

If the granularity is "passage" and the paragraphs found in a document are not adjacent, the following merge function is used to assign a new weight to each of these paragraphs:

$$W_{pnew} = (W_p + h_1 * W_d) * \log_{10}(10 * |P|) \tag{7}$$

where W_{pnew} is the new weight of the paragraph, W_p is the weight of the paragraph obtained from the paragraph level index, W_d is the weight of the document containing the paragraph, which is obtained from the document level index, $|P|$ is the total number of paragraphs retrieved from this document, and h_1 is a coefficient, which is set to be 3 in our experiments.

If there are n adjacent paragraphs found in a document, we merge these paragraphs into one and use the following function to assign a weight to the newly merged paragraph:

$$W_{pnew} = (W_{p_1} + h_1 * W_d) * \log_{10}(10 * |P|) + \frac{1}{2} \sum_{k=1}^{n} W_{p_k} \tag{8}$$

where W_{pnew} is the weight of the newly merged paragraph, W_{p_1} is the weight of the first of these adjacent paragraphs obtained from the paragraph level index, W_d is the weight of the document obtained from the document level index, $|P|$ is the total number of paragraphs retrieved from this document. W_{p_k} is the weight of the kth of these n adjacent paragraphs, and h_1 is a coefficient, which is set to be 3 in our experiments.

4 Genomics Track

In recent years, there has been a large amount of experiments and researches conducted in the area of genomics and its related disciplines. As a result, a vast amount of scientific literature has been published. One purpose of Genomics track is to study the retrieval task in the domain of genomics. Given the MEDLINE database,

how can we adapt the general purpose information retrieval system for the genomic domain? There are two tasks for this year's Genomics track. The primary task is an ad-hoc retrieval from MEDLINE corpus. The second task is categorization. This year, we only participated in the ad-hoc task. Our efforts concentrated on applying different techniques to Okapi system for improving the retrieval performance.

The goal of the York team to participate in TREC Genomics track is to evaluate the Okapi information retrieval system in the genomic domain. In the past TRECs, BM25 has been shown to be a very good probabilistic weighting scheme in text retrieval. However, the Okapi system has never been evaluated in the genomic domain.

4.1 Topics

The 2004 topics are completely different from the 2003 topics. There are 50 topics derived from interviews elicitingal information needs of real biologists. Each topic has the following format:

1. ID - 1 to 50
2. Title - abbreviated statement of information need
3. Information need - full statement information need
4. Context - background information to place information need in context

This year's topics are more varies and specific than last year's topics, which only require to find all MEDLINE references that focus on the basic biology of the gene or its protein products from the designated organism. This year two new fields are added for each topic: information need and context. Both of these fields are stated in a natural language form and give more specific instruction for each topic. Only five training topics are given this year, thus increasing the difficulty of the search compared to last year.

For each topic, we first need to extract the search terms from the topic. Then, we searched against the MEDLINE corpus using Okapi information retrieval system for each term. After the first round search, more terms are added by using blind feedback. A second round search is performed and its result will be used as the final result.

4.2 Term Selection

For query term generation, a simple and automatic method is used to select at most 110 query terms from genomic topics. The initial query term selection is not only based on their weights but also based on their frequencies in the topics. The blind feedback is used to expand the initial query terms after the first round search is

done. Those new terms are extracted from the top 10 first round search documents and are added into the original query terms. The expanded query terms are then used for the second round search. For each search term extracted from topics, we count its occurrences. An example is shown as follows:

transgenic 3 1 1 1

The first column is the term extracted from the topic. The second column contains the total number of occurrences of the terms in the topic. The third one indicates the number of occurrences of the term in field TITLE. The fourth and fifth indicated the number of occurrences of the term in the field NEED and CONTEXT correspondingly.

4.3 Search

For retrieval, the retrieval documents are ranked based on the Best Match function BM25 together with the frequency of the search term. The search results from different fields are assigned different weights. Four different retrieval results are generated for each topic by retrieving documents based on the indexes constructed according to Title, Abstract, Mesh terms and NameOfSubstance. Those four results are then combined into the final result. Since the lengths of document in the collection are more or less the same, we do not use global correction factor in our genomics experiments. An organism filter is used in the experiments to filter some the irrelevant documents.

After the first round search, we choose the top 10 documents resulting from the first round search and assume them as relevant documents. Then we rank these new terms extracted from the top 10 documents by using their RSV values and add them into the original search terms. By doing in this way, we can keep the terms from the original topics and also add some new terms extracted from the top 10 documents.

5 Experiments and Results

5.1 HARD Experiments

The original HARD corpus is around 1.6G. It contains one year (2003) of newswire data from eight sources: AFE (Agence France Presse - English), APE (Associated Press Newswire), CNE (Central News Agency Taiwan), LAT (Los Angeles Times), NYT (New York Times), SLN (Salon.com), UME (Ummah Press - English), and XIE (Xinhua News Agency - English). In order to use the original data on our search system, we first need to convert this raw data into a format which can be recognized by the Okapi system, called exchange format. Based on this exchanged database, a runtime database and indexes are built by using Okapi. Both document-level index and passage-level index are built for the document-level retrieval and passage-level

retrieval. A GSL file, which contains the most common synonyms, is used during the process of building index.

5.1.1 Evaluation Results at TREC 2004

We submitted five runs for the HARD track at TREC 2004, one baseline run, two automatic runs and two manually runs. The document-level evaluation results are presented in Table 3 and the passage-level evaluation results are given in Table 4.

Run	Description	Soft-rel R-Precision	Hard-rel R-Precision
york04hb1	baseline	0.2003	0.1796
york04ha1	automatic	0.3273	0.3163
york04ha2	automatic	0.3185	0.2805
york04hm1	manually	0.3336	**0.3308**
york04hm2	manually	**0.3440**	0.3012

Table 3: Document level evaluation results

Run	Description	Bpref@12K Char	PassagePrec@10	PassageRPrec
york04hb1	baseline	0.088	0.1532	0.0672
york04ha1	automatic	**0.358**	**0.4420**	0.2856
york04ha2	automatic	0.186	0.2729	0.2481
york04hm1	manually	0.352	0.3956	**0.3511**
york04hm2	manually	0.183	0.2778	0.2926
Average Best	Over all 136 runs	0.358	0.4420	0.3511
Average Median	Over all 136 runs	0.200	0.1732	0.1091

Table 4: Passage level evaluation results

The runs 'york04ha1' and 'york04hm1' were generated by Algorithm 1 and the runs 'york04ha2' and 'york04hm2' were generated by Algorithm 2. That is: the runs 'york04ha1' and 'york04hm1' were generated by (1) doing both document level search and passage level search and (2) using both BM25 and BM50 for term weighting. However, the runs 'york04ha2' and 'york04hm2' didn't use these two methods. Instead, it assigned negative weight to negative terms returned from the clarification form. The difference between automatic run 'york04a1' and manual run 'york04m1' is that for the run 'york04m1', a manually generated key filter is used at the very beginning of the search. It is easy to find from Table 3 and 4 that all the final runs

made significant improvements over the baseline run. For passage level retrieval, Algorithm 1 is obviously better than Algorithm 2. For the passage level evaluation, the automatic run 'yorku04ha1' achieves the best result (0.358) in terms of Bpref measure at 12K characters [1].

5.1.2 U.S. Filter and Related Text

More experiments are conducted in order to evaluate the effectiveness of using the U.S. filter and the Related Text metadata. Table 5 gives us a comparison of three runs. The first run 'york04ha1' is our official submission in TREC 2004. The other two runs from 'experiment 1' and 'experiment 2' are based on the first run. For 'experiment 1', the environment setting is the same as the one set for 'york04ha1' except that the U.S filter is turned off. This experiment is used to test the effectiveness of the U.S filter. As we can see, turning off the U.S. filter can improve the retrieval performance comparing to turning on the U.S.filter. The possible explanation to this phenomenon is that the documents without the geographical information are eliminated even though they are relevant. This may be caused by the high restriction of the U.S. filter. For 'experiment 2', the environment setting is the same as the one set for 'york04ha1' except that no related text information is used. Without using the related text information, the retrieval performance decreases on both document-level and passage-level. It is obvious that the related text information can make a positive contribution to the retrieval performance.

Run	Description	Bpref@12K Char	PassageRPrec	Hard-rel RPrec
york04ha1	submission	0.358	0.2856	0.3163
experiment 1	No U.S filter	**0.3781**	**0.2977**	**0.3378**
experiment 2	No Related Text	0.3353	0.2294	0.2700

Table 5: Experiments for U.S. Filter and Related Text

5.1.3 BM50 vs BM25

To study the effectiveness of our proposed BM50 term weighting function on document-level retrieval, we conducted a series of experiments by tuning the k_l constant in the term weighting function.

The experimental results are shown in Table 6. All the experiments in Table 6 are based on the official submission except that we do not use the blind feedback for retrieval. In the experiments, we found that the retrieval performance decreases for document-level retrieval by using blind feedback[4]. So the blind feedback was not used in our experiments for evaluating the BM50 term weighting function. From Table 6,

[4]This is because we use documents instead of passages in the blind feedback process for document-level retrieval.

we can observe that the document-level performance in terms of hard-rel R-precision achieves the best when the correction factor contributes 13% of the total weight. Clearly, these experiments demonstrate that BM50 can make a positive contribution to retrieval performance on the document level.

Run	Description	Hard-rel RPrec	Bpref@12K Char	PassageRPrec
ExperimentA	No BM50 & No BF	0.3132	0.3181	0.2864
ExperimentB	BM50 (6.5%)	0.3255	0.3144	0.2756
ExperimentC	BM50 (13%)	**0.3256**	0.311	0.2769
ExperimentD	BM50 (20%)	0.3185	0.3093	0.2800
ExperimentE	BM50 (25%)	0.3195	0.3088	0.2899

Table 6: Experiments for BM50

5.2 Genomics Experiments

MEDLINE is the bibliographical database of biomedical articles maintained by the National Library of Medicine (NLM). The subset of MEDLINE is used for the TREC 2004 Genomics Track. It consists of 10 years of completed citations from the database inclusive from 1994 to 2003. Records were extracted using the Date Completed (DCOM) field for all references in the range of 19940101 - 20031231. This provided a total of 4,591,008 records.

Each MEDLINE record contains a number of fields, but we are only interested in the following fives fields: (a) PMID, which is the unique identifier of the PubMed (NLM's database that incorporates MEDLINE). (b) Title, which contains the entire title of the journal article. (c) Abstract, which is taken directly from the published article. (d) Mesh, NLM's controlled vocabulary, which is used to characterize the content of the articles represented by MEDLINE citations. (e) NameOfSubstance, which is the controlled vocabulary as well.

We use Okapi BSS as our search baseline. The first thing we need to do is to convert the database to the format which can be read by the Okapi system. We have developed a small tool, which convert xml file to exchange format, the format which can be used by the Okapi system. It reads through the xml file and extracts the information we need.

We submit two runs for the Genomics track. One is automatic run and the other one is manually run. The mean precision of the automatic run is 0.1794 and the mean precision of the manually run is 0.2011. Detailed information for these two runs are shown as follows.

Run	Description	Mean Precision	> Median	Median	< Median
york04g1	automatic	0.1794	19	3	28
york04g2	manually	0.2011	22	15	13

Table 7: 2004 Genomic Track results

6 Conclusion and Future work

York Team participated in both Genomics and HARD tracks this year. For the Genomics track, we adopted the Okapi system in the genomic domain without using any biomedical knowledge. In the Genomics experiments, we did not incorporate domain expertise and did not use external biomedical resources. For the HARD track, we tested our new ideas on indexing and evaluated the effectiveness of different weighting formulae. The HARD experimental results showed that using two-level indexes improves the performance greatly for both passage level and document level retrieval. For the passage level evaluation, the automatic run 'yorku04ha1' achieves the best result (0.358) in terms of Bpref measure at 12K characters. BM50 can also make a positive contribution for document level retrieval.

For the HARD track, only the Geography and Related-text metadata are used. However, the retrieval performance decreases with U.S. filter turned on (the Geography field). The Related-text field is useful in our experiments for improving performance on both document and passage level retrieval. There is a potential to use other fields as well, e.g. the Genre and Subject fields. We can find some statistic information from the database and use it to boost the retrieval performance.

For both Genomics and HARD tracks, we only use a very simple topic extraction method to extract the search terms from the given topics. A simple count of the terms within topics is served as a query term frequency. No phrase is constructed from the topics. For the future work, we will design new algorithms to find the most relevant search terms and phrases from the given topics. We expect a better topic extraction algorithm will improve the performance.

Other methods can also be applied to improve the performance, e.g. designing new term weighting formulae and using machine learning methods. For the BM50 weighting function, we can try to test different values of the parameters to improve the retrieval performance.

7 Acknowledgement

This research is supported by research grants from the Natural Sciences and Engineering Research Council (NSERC) of Canada and Atkinson Faculty of York University. We also would like to thank Kai Zheng and Xin Gao for their help in this research.

References

[1] J. Allan. HARD Track Overview in TREC 2004. *The 13th Text Retrieval Conference*, NIST Special Publication, 2004.

[2] M. M. Beaulieu, M. Gatford, X. Huang, S. E. Robertson, S. Walker and P. Williams. Okapi at TREC-5. In D.K.Harman, editor, *Proceedings of the 5th Text Retrieval Conference*, NIST Special Publication, 1996.

[3] X. Huang, S E. Robertson. A Probabilistic Approach to Chinese Information Retrieval: Theory and Experiments. *Proceedings of the BCS-IRSG 2000: the 22nd Annual Colloquium on Information Retrieval Research*, pages 178-193, Cambridge, England, 2000.

[4] X. Huang, F. Peng, D. Schuurmans, N. Cercone and S. E. Robertson. Using Self-Supervised Word Segmentation in Chinese Information Retrieval. *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, August 11-15, 2002, Finland.

[5] X. Huang, F. Peng, D. Schuurmans, N. Cercone and S. E. Robertson. Using Machine Learning for Text Segmentation in Information Retrieval, *Information Retrieval*, 6 (3-4), pp. 333-362, 2003.

[6] S. E. Robertson, J. K. Sparck. Relevance weighting of search terms. *Journal of the American Society for Information Science* 27, May-June 1976, p129-146

[7] S. E. Robertson, H. Zaragoza, M. Taylor. Microsoft Cambridge at TREC-12: HARD Track. *The 12th Text Retrieval Conference*, NIST Special Publication, 2003.