Intelligent Web Page Retrieval Using Wikipedia Knowledge

Falah H. Al-akashi
University of Ottawa
800 King Edward Av.
Ottawa, ON, K1N6N5, Canada
1-613-5625800
falak081@uottawa.ca

Diana Inkpen
University of Ottawa
800 King Edward Av.
Ottawa, ON, K1N6N5, Canada
1-613-5625800
diana@eecs.uottawa.ca

ABSTRACT
In this paper, we present an intelligent web retrieval system that is able to rank webpages by using Wikipedia knowledge to enhance a standard vector space model. Our index contains separate information about the frequency of the terms in Wikpedia articles, in home pages, and in other types of web pages, instead of using a generic term frequency for the whole text collection.We also filter out spam. We present results on the ClueWeb collection, for two sets of queries, for an adhoc retrieval task and for a diversity task (which aims at retrieving not only relevant information, but also information for different aspects of the queries).

Categories and Subject Descriptors
H.3.3 [Information Search and Retrieval] - *retrieval models, query formulation, relevance feedback, search process.*

General Terms
Algorithms, Performance, Design, Experimentation.

Keywords
Web retrieval, indexing, Wikipedia knowledge, home pages, vector space model, query expansion.

1. INTRODUCTION
In order to experiment with new intelligent search model for the web, we use a very large text collection named the ClueWeb[1] dataset. It consists of about 1 billion web pages in ten languages that were collected in January and February 2009. We can evaluate our models by using the test and training queries made available by the National Institute for Standards and Technology (NIST) in 2011, as part of TREC[2] (Text Retrieval Conference) for the web search track[3]. The advantage of using these queries is that NIST also made available expected solutions, called relevance judgments, consisting of lists of document that are relevant answers to each query.

We focus on two tasks: a classic text retrieval task, called adhoc retrieval, and a diversity task. The diversity task is similar to the adhoc retrieval task, but differs in its judging process and evaluation measures. The goal of the diversity task is to return a ranked list of pages that together provide complete coverage for a query, while avoiding excessive redundancy in the result list. For this task, the probability of relevance of a document is conditioned on the documents that appear before it in the result list, since the goal is to cover as different aspects of the relevant information, without repetitions.

For evaluation, we used two sets of queries:36 training queries that we used to test our system while we developed it (these queries are in fact the 2010 test queries from the Web track at TREC), and the 50 test queries from 2011, that we used for the final testing.

Here are some examples of three queries from the 2010 set:

- “how to build a fence”
- “to be or not to be that is the question”
- “pvc”

Here are other examples of three queries from the 2011 set:

- “ritzcarlton lake lasvegas”
- "uplift at yellowstone national park”
- “trombone for sale”

For each dataset, we used the expected solutions built by the human assessors from NIST, in order to evaluate the results of our system. The evaluation for the ad-hoc retrieval task uses the human judgments about how relevant is each document as an answer to reach query. For the diversity task, the evaluation is more complicated because the assessors have to look at different sub-topics (possible meanings of each query).

The rest of the paper described each component of our system. Section 2 describes our indexing model. Sections 3 explain how we expanded and processed the queries. Section 4 explains how we matched documents to queries in order to obtain answers to the queries. A re-ranking based on reputation is done on the list of selected documents, as explained in Section 5.Section 6 explains how we filter out spam form the results. Section 7 presents our results; we compare them to related work in Section 8, followed by conclusions in Section 9.

[1]http://lemurproject.org/clueweb09/

[2]http://trec.nist.gov/

[3]http://plg.uwaterloo.ca/~trecweb/2011.html

2. INDEXING MODEL

Indexing is crucial for the task of finding relevant information on the Web. Various indexing methods are used in a wide range of applications, such as Home-page finding, Entity finding, and Web pages classification. The design of highly-scalable indexing algorithms is needed, especially with an estimate of one billion pages currently accessible on the web. Previous work classifies indexing of web documents in two types: word-based and phrase-based indexing [1].

In word-based indexing, single words are used to build an index table and to find a set of relevant pages according to some computations. Recently, published work on very large scale retrieval systems has been dealing with a metric that counts occurrences for all distinct terms, with the goal to find all web pages that are most relevant to a particular topic.

In order to find keyphrases that are important for each topic, a set of articles can be assembled from a particular dictionary. This was used recently for document clustering, entity finding, or document classification in small collections of documents, but it has not been used on the large scale webpage indexing.

Moreover, the current approach of word-based indexing does not scale well for phrasal queries, because the positions of the words need to be recorded, requiring a lot of storage space. On another hand, indexing without considering word location for proximity search causes two documents to seem similar if they have words in common, even if they are on different in topics. Term frequency is important in determining the topic of documents; but this is not case for all documents configurations, because documents may contain different topics located on different parts. Only a few systems consider this aspect, for example by using a sliding window for each topic [2]. Using phrases in the index in addition to words eliminates is another way to deal with this problem.

We categorized the documents in our repository index into three types: Wikipedia articles, home pages, and other documents. Each type holds a particular type of document. We will describe the indexing algorithm that we used for each category.

2.1 Wikipedia Repository Indexing

Wikipedia is a collective knowledge source of approximately 5 million articles in English. The data in each article is structured into several fields, and sometimes it has a relationship with other articles using tags or links to expand a certain topic. Each article has a unique vocabulary name (identifier or ID); sometimes Wikipedia uses different faceted vocabulary terms to describe the same article. We extracted the important information (keyphrases) from each article and used it as setup information to rank each leaf node in our customized index. Our system scanned through the whole Wikipedia repository (the version that is included in the ClueWeb collection) and computed the following normalized vectors for each document:

- Words-occurrences (frequencies); we assigned a threshold value to exclude all low frequency terms and to keep the ones with frequency higher than the threshold.
- Outgoing-links occurrence frequencies; outgoing-links are all URLs that point internally to other Wikipedia articles. For us, if an article points to other articles frequently, at more than one position in the content, then all these articles are related or similar in topic.
- All external URLs.

Items such as html markups, navigation links, and stop-words were removed. As a result, the outputs were transformed and represented as vectors of the following structure:

$$D^{w} = \{D^{URL}, D^{ID}, D^{lf}, D^{eURL}, D^{tf}\}$$

whereD^{URL} is a document URL, D^{ID} is a documents identifier, D^{lf} is represents the outgoing-link frequencies (for the links that are repeated frequently at more positions in the content), D^{eURL} refers to all external links, and D^{tf} represents all terms that occurred with high frequency.

Thus, all documents in Wikipedia repository are transformed into several tables of vectors available in the leaves of the index.

2.2 Home Pages Indexing

The use of page content for home-page finding is problematic for several reasons. Often the first page in a site is a home page which mostly contains navigational links for sitemap. Some potentially useful evidence for home page finding is query-dependent. This includes the presence of query words in the document's text, which is referring to anchor texts, or in the document's URL. It is known that full-text relevance ranking is not particularly effective for home page finding [4]. Other potentially useful evidence is query-independent. This was demonstrated in the TREC-2001 home page finding task [4]. The best run was submitted by Westerveld et al. from UTwente/TNO [5] and used the pages URLs as evidence. Generally, query-dependent and query-independent are not the case for all situations of home-page finding, because URLs sometimes use shortcuts or abbreviation terms to represent some underlying meanings beyond the domain name, e.g., "nist.com". We used different representations when we processed URLs for home-pages. We used two basic methods for the home-page finding task: the first method is standard and it uses the structure of URL names and the second method is suitable for most abbreviated and embedded terms.

2.2.1 Processing URL Structures

According to the general structure of URLs, we classified them into five categories (after stripping off all the symbols, numbers, and all the trailing terms such as index, default, and welcome), depending on the location of the term in the URL:

- If a term is located in the main domain section, e.g., "www.diana.com/".
- If a term is mixed or embedded with other terms, such as Air France in "airfrance.ca".
- If a term is represented as a shortcut or an abbreviation, e.g., "www.uottawa.ca".
- If a term is located in the sub-domain section, e.g., "trec.nist.com/".
- Any an URL was ended at document's name and preceded by a symbol "~", e.g., "www.uottawa.ca/~cadams".

Thus, we have only five possibly evidences in the URL forms. We used different scores for each status; we assigned the ranking value "1", "2", or "3" for a term that is located on the main-domain, sub-domain, and document-name, respectively.

Home-page finding methods require finding equivalent pages by converting hyperlinks might to a canonical form, for example: "http://bmo.com" "http://www.bmo.com/"

"http://www.bmo.com:80/"

"http://www.bmo.com/index.htm/"
"http://www.bmo.com/welcome/"

"http://www.bmo.com/default"

"http://www.bmo.com/ (language code)/ index.html"

should be all represented as "www.bmo.com/".

2.2.2 Embedded Keyword Extraction

URL keywords or terminology extraction is a challenging task. Researchers employed different algorithms, such as a statistical "n-grams" [5] or natural language processing methods for tokenizing and analyzing URL data to extracting keywords that can be utilized to index content. Besides web page popularity, we exploit using query log files assembled by Alexa.com[4] for finding out the original keywords behind the embedded keywords in domain names. Alexa.com computes traffic for all popular search engines. We used the tf method for computing the occurrence of frequent queries in log file. The log file contains the important queries for each site accessed by users ; for instance "airfrance", "uottawa", and "nist" are defined in log file as queries: "Air France", "University of Ottawa", and "National Institute of Standards and Technology", respectively.

Our method was tested for home page finding for the TREC 2011 webtrack queries. For example the query "jax chemical company" involved retrieving all home pages available in the corpus; therefore our system was obtained a precision p@5=1.0 and p@10=1.0 for this query.

2.3 Other Collection Indexing

Usually, web pages are indexed using their content, but not all pages are useful for indexing their content, for instance multimedia pages may contain videos, sound, or images; home-pages sometimes contain little text; other web pages may contain topical content such as programming code or navigational links which are useless for indexing. Generally, we used two types of index structures: word based index and key-phrase based index.

2.3.1 Word Based Index

Often the meaning of a document is conveyed by words located in meta-content (such as URLs, titles, and headers). Normally there is at least one term shared between meta-content and the document content. If we can represent the shared word by a short vector and the document content by another vector, it is possible compute the similarity and the impact of that term in the document. Basically, meta-content is available in three fields ("title", "headers", and "URL") and it is necessary to manipulate all these fields together, because (i) we assume that not all documents contain important terms in alone of these fields (ii) usually keywords in the title, headers, and URL are complementary to each other. If the header h1 is not available or it is similar to the title, we chose "h2" or "h3", alternatively. A very short meta-text may not contain enough information and a long text may contain unnecessary or redundant information. Also, it is necessary to index the main content in order to provide a comprehensive indexing. We use meta-keywords to trigger the document's topic, and then to go inside the index for other query terms.

[4] http://www.alexa.com

Two documents from different sites might have meta-content with different impact, even they have similar meta-content. Documents whose content has higher similarity to its meta-content should be judged more relevant. Our model was modeled to measure the closeness of any document content to its meta-content, by computing the cosine similarity between the document content and each meta-term, with the cosine similarity between two vectors [8].

Before computing the similarity measure for each term in meta-content, we processed the document content as follows:

- Stripping off all the html codes from the content of the document
- Removing stop words, symbols, and numbers,
- Removing stemming characters from each term.
- Computing the frequency of occurrence *tf* of each term in the document content.

Once the *tf* value of each term was computed, we used the cosine similarity to quantify the impact of each term from the meta-content. Each meta-term is assigned its cosine similarity measure with the document content. To determine which meta-term is significant, we use a specific threshold value to choose the best terms and ignore others. On other hand, as we mentioned before, not all terms in the content are available in the meta-content; likewise it is rare to find all the query terms located in meta-content. Therefore, it is impractical to rank web documents only by their meta-contents; we need to add the term frequencies in the content of the document. To reduce the storage needs, we ignored all the terms that occurred only once. Each term in meta-content has a vector with a certain dimension; including the cosine measure, and the significant terms frequencies, in addition to the "docID" and the "termID". Document relevance is computed by summing up the cosine similarity for the first query term which is available in the meta-content; otherwise, only terms frequencies are computed. Sometimes, queries contain digits when looking for more precise results, e.g. "hp mini 2140"; therefore we address this issue by adding one extra dimension to the vector to yield document title. Hence, the meta-content of "n" terms is broken down to "n" vectors; and then each vector is transmitted to a corresponding node in the index, as shown below:

{docID}{TermID}{SC(Term)}{<t1,f><t2,f><t3,f>,….<tn,f>}{Title}

2.3.2 Key-Phrase Based Index

Our method does not employ computations for the terms that occurred only once; but they can still be used for phrased queries. Some documents are based on a fixed interval of sequence terms; these terms could occur only once or could be repeated in the document's content. Terms do not need to occur at more than one position in the content; for instance the query "map of brazil" is sometimes located once at one position in the document; hence terms occurrences are not important for the document's relevance. However, our method uses the key-phrase index with the respect of computing terms frequencies for all the terms in the content. For example, the query "Martha Stewart and imclone" requires to compute the proximity search for all terms; computing the term frequency for the term "imclone" and the key-phrase frequency for the phrase "Martha Stewart" is important for computing document's relevance. To compute the key-phrase frequency, first we strip off all stop-words, symbols, characters, and single letters from the document content. Next, we compute the frequency of single terms, double contiguous terms, then length three, four, etc., as far as terms occurred together frequently. Then, for each key-phase we compose a vector of fixed dimension, as shown below:

{docID}{Key-phraseID, f}{<t1,f><t2,f><t3,f>….<tn,f>}{Title}

where key-phraseID is a hash key that is generated using the same algorithm that generated the hash keys for each node in our index.

3. QUERY PROCESSING AND EXPANSION

Query processing is an important processing step and it includes: detecting the type of the query, query normalization and query expansion. Basically, we have five types of queries according to the: title, domain, frequency.

- **Title**: this means that relevant pages contain all query terms in core positions, as full keyphrases, e.g. "arkadelphia health club" or "map of brazil".
- **Domain**: this means that relevant documents are located in a particular site or domain, e.g., "jax chemical company".
- **Occurrence:** this means that relevant documents were judged using the occurrences of query terms in the document, e.g., "fact of uranus" or "Martha stewart and imclone".

Each query is processed into three types of indexes mentioned in section 2. We use overlapped term positions besides the priority factor for each term in the query. Our system uses the following criteria for processing a query:

- If the query length is one term, searching is done in two indexes: the home-page index and the word-based index, because a one-term query could look for a home page, e.g., "uottawa", or, the term could be frequent in a document's content, regardless if it is a home page or not, e.g., "afganistan".
- If the query length is two or three terms, searching occurs in three indexes: the home-index, the word-based index, and the key-phrase based index, e.g., "Map of Brazil" or "Ralph Owen Brewster".
- If the query length is four or more, searching occurs in two indexes: the word-based and the key-phrase based index, e.g., "Ritz Carlton Lake Las Vegas". In the case of keyphrase-based index, the query would be searched as: "Ritz Carlton Lake Las Vegas", "Ritz Carlton lake Las", and "Ritz Carlton Lake"; whereas in case of the word-based index, first, a node "Ritz" is located and the system goes through each document vector to find the other query terms. Next, a node "Carlton" is located and then the system walks through each document vector to finding other query terms; and so on, regarding other terms query. The results from all the search situations are aggregated in one list, without duplication.
- If the query involves a prepositional or conjunctional term, then the first term and the single term that occurred before or after the conjunction is weight more than other terms in the query, for example the terms "uplift" and "Yellowstone" on the query "uplift at Yellowstone national park"; or a term "french" and "casino" in a query "french lick resort and casino".

Search engines use query expansion to increase the quality of the search results. It is assumed that users do not always formulate search queries using the best terms. The goal of query expansion is to increase recall, without decreasing too much the precision, by including in the result pages which are more relevant (higher quality), or at least equally relevant. In the same time, many of the current commercial search engines use word frequency (tf-idf) to assist in ranking. By ranking the occurrences of both the user entered words and synonyms and alternate morphological forms, documents with a higher density (high frequency and close proximity) tend to migrate higher up in the search results, leading to a higher quality of the search results near the top of the final ranked list.

The trade-off between precision and recall is one of the problems of query expansion. However, to improve retrieval performance in our system, we used query expansion for those queries that were classified as Wikipedia articles; anchor terms or phrases that frequently occurred in each article were used to expand the query topic (anchor terms have been indexed previously); for instance, the topic "all men are created equal" that ranked our system as high precision P@5=0.8 and P@10=0.7, because the query was expanded with the phrases "Gettysburg Address" and "Declaration of Independence" from related Wikipedia articles. In this way the system succeeded to retrieve relevant documents that were not retrieved without the query expansion step, because they contained the query terms with low frequency (all the terms occurred once).

4. DOCUMENT RANKING

In this section, we explain our custom model for ranking webpages which uses cosine measure similarity. As we said, not all query terms have equal impact or weight. For each query, there is one term has more impact than others; for instance the query, ***"Martha stewart and imclone"*** is focused on a term "imclone" more than on the other term.

However, we used the following formula for ranking documents for each query:

$$Rank(D_i,Q) = SC(D_i, imclone) + \frac{\sum_{j=1}^{t} W_{ji}}{100}$$

where $SC(D_i, imclone)$is the cosine similarity for the query term "imlcone" in document Di, Wji is the weight of query term j in document i; and, t is the number of query terms.We added the sum of all the weights of query terms because not only the term "imclone" is important in the query, but other terms are also important. That is, we used the cosine similarity for the term that has more impact than others in the query, plus the sum of the weights if all the query terms (the value is divided by 100 is to find the percentage value).Finally, our system biases the final ranking list by the site's reputation, Wikipedia preferences, and other preferences, as explained in the following subsections.

5. REPUTATION RANKING MODEL

Traditional methods for ranking documents are not optimal in terms of search engine optimization (SEO). Smoothing ranked list and changing documents positions in our search results was used based on a number of factors designed to provide end-users with helpful and accurate search results. In our method, we used two strategies based on human references to improve our rankings.

5.1 Using alexa.com

With the massive amount of data available on the web, not all data are reliable and valuable. There are a lot of sites with untruthfull content that might be ranked high by our model. Informational queries, for example, are always looking for reliable and valuable information; this information is usually available in sites that can be trusted. Summarizing, site reputation, ranked locally and globally, are important in our relevancy algorithm. We used this factor for enhancing our ranking algorithm by filtering out all the poor sites. We exploited the information from www.alexa.com by assembling all reputation values for the main domains in our corpus.

5.2 Using Wikipedia

As we mentioned earlier in this paper, documents that are classified as home pages or documents that are important articles in the Wikipedia might change their rank in our final ranking list. Human references are robust arguments to bias the ranking towards some documents. Since we previously indexed all the important external references to Wikipedia articles, the ranking algorithm will make a match between the ranked list and the archived indexed documents that existed in each node for the search query. As a result, the matching documents will get higher positions in the final list.

6. SPAM FILTERING

The ClueWeb09 collection contains a lot of spam documents. We filtered out spam documents that would hurt the quality of our retrieval. Cormack et al. [7] studied the spam filtering in the "ClueWeb" collection and showed that the spam filtering could significantly improve the performance of a system. Therefore, computing term frequency and cosine term similarity in our system could detect spam documents because they use many junk words that affect the impact of each term (the cosine term similarity in our method). If the cosine term frequency is lower than a threshold, the document is considered junk; if it is higher than a second threshold, the document is considered spam. In between the two threshold values, the document is kept in the list of relevant documents. We chose appropriate thresholds based on a small development set.

7. EXPERIMENTAL RESULTS

We present results for the adhoc and the diversity tasks of the web track. Our system is based on the collection of document Category B of the ClueWeb09 corpus (50 million documents).

Table 1 and 2 list the results of different metrics for both the diversity and the adhoc metrics, as average for the 36 training queries (which are in fact the 2010 test queries). Tables 3 and 4 show the results for the 50 test queries (the 2011 test queries).

We use standard evaluation measures used in TREC. The primary effectiveness measure for the adhoc task is expected reciprocal rank in the first k documents retrieved (ERR@k) [9] [10]. We also report anDCG [9], as well as standard binary measures, including mean average precision (MAP) and precision at rank k (P@k). The primary effectiveness measure for the diversity task is a variant of intent-aware expected reciprocal rank (ERR-IA) [9]. We also report a number of other intent-aware measures appearing in the literature, including αnDCG@k (Discount Cumulative Gain), NRBP (Rank-biased Precision), and MAP-IA [10].

Table 1. Diversity task results for the 36training queries

αnDCG@10	**αnDCG@20**	**ERR-IA@10**	**P-IA@10**	**P-IA@20**
0.5464	0.5564	0.4377	0.3989	0.3110
MAP-IA	**NRBP**	**ERR-IA@20**	**strec@10**	**strec@20**
0.1024	0.4181	0.4406	0.5705	0.6102

Table 2. Ad-hoc task results for the 36 training queries

NDCG@20	**ERR@20**	**P@10**	**P@20**	**MAP**
0.4024	0.2286	0.5200	0.454	0.0638

Table 3: Diversity task results for the 50 test queries

αnDCG@10	**αnDCG@20**	**ERR-IA@10**	**P-IA@10**	**P-IA@20**
0.4380	0.4675	0.3578	0.2414	0.2098
MAP-IA	**NRBP**	**ERR-IA@20**	**strec@10**	**strec@20**
0.0685	0.3207	0.3670	0.6731	0.7155

Table 4: Adhoc task results for the 50 test queries

NDCG@20	**ERR@20**	**P@10**	**P@20**	**MAP**
0.1743	0.09639	0.2909	0.2636	0.0838

For some queries, our system obtained good results, but for a few of the queries, the precision of the retrieved document list was zero because relevance judgments contained only documents selected from the other part of the collection (Category "A"). For other queries, our method was not able to model the meaning of the queries. In some cases, the retrieved documents looked relevant to us, but they were not relevant according to the relevance judgments. This happened because it is difficult to capture all relevant documents that satisfy all users' needs in one relevance judgment file, since users might have different points of view at different moments in time.

8. COMPARISON TO RELATED WORK

In this section we compare our system to other systems, on the same task and for the same dataset. We look at both the ad-hoc and the diversity retrieval tasks. We chose the results of the best systems presented in the TREC 2010 web track overview paper [9] and in the TREC 2011 overview paper [11].

The results in Table 5 and Table 6 are for the training queries (which were the 2010 test queries). According to the tables, our current system obtains better results than the other systems that used the same subset of the data, namely the part B of the ClueWeb collection. The reason we and a few other systems that we compare with used the part B of the ClueWeb collection is that it requires 5TB of disk space, while the whole collection requires 25TB of disk space. We did not have 25TB of disk space available.

Table 5. Comparison of the ad-hoc retrieval task results for the training queries (the 2010 test queries) for three other systems and for our current system

Group	ERR@20	nDCG@20	P@20	MAP
ISI	0.134	0.225	0.379	0.133
IRRA	0.126	0.260	0.443	0.133
UAmsterdam	0.110	0.145	0.237	0.043
Our system	0.228	0.402	0.454	0.063

Table 6. Comparison of the diversity retrieval task results for the training queries (the 2010 test queries) for four other systems and for our current system

Group	ERR-IA@20	α-nDCG @20	NRBP	MAP-IA
uogTr	0.298	0.418	0.262	0.074
UAmsterdam	0.242	0.341	0.210	0.026
UCDSIFT	0.210	0.312	0.170	0.062
qirdcsuog	0.210	0.312	0.170	0.062
Our system	0.440	0.556	0.418	0.102

For the test queries from 2011, our run submitted at TREC 2011 was considered only for the ad-hoc task. We can compare our results at the web track at TREC 2011 with other systems that worked with the subset of the data collection named category B. It is not fair to compare to the systems that used the whole data collection (category A), because some documents that were in the expected solution could not be retrieved by our system since they were not in the reduced dataset. Table 7 presents the comparative results, according to the track's overview paper [11]. Our system at TREC 2011 had different parameter settings than the system described in this paper[5]; this is why the results in Table 7 are not the same as in Table 4.

Table 7. Comparison of the ad-hoc retrieval task results for the testing queries (the 2011 test queries) for the best two systems from TREC 2011 and for our system submitted to TREC 2011.

Group	ERR@20	nDCG@20	P@20	MAP
Name not Disclosed	0.131	0.233	0.298	0.110
Univ. of Ottawa	0.122	0.204	0.275	0.079
Univ. of Amsterdam	0.119	0.202	0.273	0.085

9. CONCLUSIONS

Our method used our own custom indexing and ranking model based on Wikipedia knowledge. This model provides a variety of analytic capabilities, including: concept extraction, concept correlation, text summarization, spam filtering, and term to document similarity.

We improved several aspects of our system. We kept stopwords in the key-phrase index. This allowed us to successfully process queries such as "to be or not to be, that is the question". The conjunctions and prepositions also allowed us to separate important terms in some queries, e.g., for the queries: "Martha Stewart and imclone" and "earn money at home", the important terms are: "imclone" and "home", respectively.

In future work, we plan to experiment with more types of queries and more ways of including knowledge from Wikipedia in our retrieval system.

10. ACKNOWLEDGMENTS

We thank NIST for evaluating the submitted TREC results. We thank NSERC and Iraqi Embassy in Canada / Cultural Office for supporting our research.

[5]Nearly every ranking algorithm has parameters that can be turned to improve the effectiveness of the results. For example, Okapi BM25 has the parameters k1, k2, and b used in term weighting, and query likelihood. Ranking algorithms for web search can have hundreds of parameters, the weights for the associated features. In our system, we used two parameter values (thresholds). If we use the same parameter value in the training queries and testing queries, the results would be as shown in tables 1-6 above. Our submitted system at TREC 2011 had different parameters. The type of queries in 2011 is different from the type of queries in 2010. Most of the queries in 2010 target the diversity task (short queries), whilst most queries in 2011 target the adhoc task (longer queries). The parameters have default reasonable values in our system, but the optimum parameter values vary with both the length of list being ranked (the number of documents in that list) and the length of the queries, so the system is able to improve the effectiveness of search by tuning the appropriate values of the parameters.

11. REFERENCES

[1] Xiangji Huang, Damon Sotoudeh-Hosseini, HashmatRohian and Xiangdong An, "York University at TREC 2007: Genomics Track", School of Information Technology, York University, Toronto, Ontario, Canada, Department of Computer Science & Engineering, York University, Toronto, Ontario, Canada, 2007.

[2] Xu Chen, ZeyingPeng, Jianguo Wang, XiaomingYu,Yue Liu, HongboXu, Xueqi Cheng, "ICTNET at Web Track 2010 Ad-hoc Task", Institute of Computing Technology, Chinese Academy of Sciences, Beijing, Graduate School of Chinese Academy of Sciences, Beijing, 2010.

[3] DjoerdHiemstra and Claudia Hau, "MapReduce for Experimental Search", University of Twente, 2010.

[4] N. Craswell, D. Hawking, and S. Robertson. 2001. Effective site finding using link anchor information. In Proceedings of ACM SIGIR'01 (New Orleans, LA). 250–257, 2010.

[5] Nick Craswell and David Hawking, "Query-Independent Evidence in Home Page Finding", TrystanUpstill, Australian National University and CSIRO Mathematical and Information Sciences, 2010.

[6] EdaBaykan, Monika Henzinger, Ludmila Marian, lngmar Weber, "Purely URL-based Topic Classification", EcolePolytechnique, Google, Lausanne, Switzerland. 2009.

[7] G. V. Cormack, M. D. Smucker, and C. L. A. Clarke. "Efficient and effective spam filtering and re-ranking for large web datasets", April 2010.

[8] David A. Grossman, OphirFrieder, "Information Retrieval Algorithms and Heuristics", Second Edition, Illinois Institute of Technology, Chicago, IL, USA, Springer, 2004.

[9] C.L.A. Clarke, N. Craswell, I. Soboroff, and G. V. Cormack, Overview of the TREC 2010 Web Track. In Proceedings of TREC 2010, NIST Special Publication: SP 500-294, 2010.

[10] Olivier Chapelle, Donald Metzler, Ya Zhang, and Pierre Grinspan. "Expected Reciprocal Rank for Graded Relevance", Yahoo Labs and Google Inc, Santa Clara CA, Sunnyvale CA, and San Bruno CA. ACM, 2009.

[11] C.L.A. Clarke, N. Craswell, I. Soboroff, and E.M. Voorhees, NIST, Overview of the TREC2011 Web Track. In Proceedings of TREC 2011, the Twentieth Text Retrieval Conference, NIST Special Publication: SP 500-295, 2011.