Term Impact-Based Web Page Ranking

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.Inkpen@uottawa.ca

ABSTRACT
Indexing Web pages based on content is a crucial step in a modern search engine. A variety of methods and approaches exist to support web page rankings. In this paper, we describe a new approach for obtaining measures for Web page ranking. Unlike other recent approaches, it exploits the meta-terms extracted from the titles and urls for indexing the contents of web documents. We use the term impact to correlate each meta-term with document's content, rather than term frequency and other similar techniques. Our approach also uses the structural knowledge available in Wikipedia for making better expansion and formulation for the queries. Evaluation with automatic metrics provided by TREC reveals that our approach is effective for building the index and for retrieval. We present retrieval results from the ClueWeb collection, for a set of test queries, for two tasks: for an adhoc retrieval task and for a diversity task (which aims at retrieving relevant pages that cover different aspects of the queries).

Categories and Subject Descriptors
H.3.3 [Information Search and Retrieval] - *indexing model, query expansion and formulation, search process.*

General Terms
Algorithms, Performance, Design, Experimentation.

Keywords
Web retrieval, indexing, searching, Wikipedia anchors, term impact, vector space model, query expansion.

1. INTRODUCTION
Urls and titles of document contain essential keywords that describe document's content [15]. How are the urls and the titles of the documents important and how are they related to document's content? How to reflect the document's topics starting from its meta-data? How can the links in Wikipedia be used for detecting topics in documents? These are the questions we are addressing in our web page retrieval task.

Indexing documents by representative keywords is still a difficult and complex task. Researchers attempted to use the large amounts of information from external resources to assist in finding a small number of keywords that represent each document's topics [2, 3, 5, 18]. Other researchers used knowledge base acceleration (data stream filtering) that exploits bi-directional links between knowledge base entries and documents [1]. Other attempts to define the document topics used keywords extraction techniques based on the content of the document. Some techniques used only keywords from meta-data, while others exploited the anchor texts [3] [5, 6, 7, 8, 9, 10, 11]. In either solution, researchers did not consider how to relation between the weight of the terms extracted from meta-data or anchors and document's content. On another side, query expansion for reflecting the user needs is also a challenging task. It either requires collecting information for user's behavior (from query logs, if available), or using external knowledge sources. These difficulties and limitations motivate our approach to use the document's content and Wikipedia links to develop a new method to compute the weights of the index terms.

Our approach to document indexing and ranking is based on the relation between the meta-content and document's content. Unlike the traditional approaches based on term frequency and on Web graphs, our approach exploits the impact of the terms within the document's content using the meta-data available in the url and in the title of the document. Computing the impact of each keyword from the meta-data helps to reflect the topic in the document's content. We also use Wikipedia to measure the importance of the selected terms for different topics that appear in the documents.

In order to test our approach to the task of the web search, we used a subset of very large text collection named the ClueWeb09[1] dataset used in the web search track in TREC evaluation campaigns. The full dataset consists of one billion web pages in ten languages that were crawled in January and February 2009. We used the subset B which is 10% of the whole English collection (500 million web pages). In the next sections, we discuss related work; followed by a description of our approach, and its evaluation on a set of test queries from TREC web track 2012. We conclude with a discussion of the strengths and weakness of our approach.

WIMS'14, June 2-4, 2014, Thessaloniki, Greece.

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

2. RELATED WORK

Models for Web indexing provided by several major search engines are relevant to our work. Unfortunately, the details of these approaches are not publicly available. Also highly relevant are indexing approaches used in the systems that participated in the web track at TREC. Researchers [14] experimented with the latent concepts underlying query models by using Latent Dirichlet Allocation (LDA) to extract specific query-related topics from pseudo-relevant feedback documents. Other researchers [20] experimented with a formal model of word meaning and association to enhance the query representation of the topic titles. University of Delaware's system [21] experimented with two strategies: the first strategy combined the ontology and unstructured data to extract integrated subtopics. Then, a coverage-based diversification function was used to diversify documents based on the integrated subtopics. The second strategy exploited the structured information in ontology for diversification. The Chinese Academy of Sciences' model [11] used the Golaxy[2] framework which is a high performance distributed search platform deployed over several servers. The researches experimented with the BM25 model and Learning-To-Rank to combine multiple features from documents. A novel framework within Terrier was used by researchers in [19]. This framework deployed the state-of-the-art "LambdaMART" learning-to-rank-technique which exploits feature extraction techniques from the documents, such as: term-dependence features, spam features, quality features, url and link analysis features, field-based features, and weighting features (BM25). Each feature was assigned a particular weight. However, most experiments combined features extracted from the titles, the text of the incoming anchors, and the text of any incoming redirect links; others deployed several features from web documents, such as the score of Google virtual document models, the BM25 score between the query and document's content, the cosine similarity between query and title, etc. Finally, University of Twente's system [22] experimented with the Hadoop framework which is an ensemble clustering approach aimed to improve the quality of document clusters. The ensemble approach run obtained better results than the LDA-based diversification.

3. OUR APPROACH

The key idea for our index is to store data in an efficient way inspired by the block-oriented storage contexts called B-trees[3]. We aimed to avoid the rebalancing issues of some indexing trees, e.g., binary trees. Our index uses sub-trees in a fixed interval. The first level of each internal node is labeled by 3 letters from the first term (single word or phrase); whereas the second level is labeled by a full phrase or term. Each node has a collection of documents that represent the documents indexed around the term or phrase node. This means that each leaf contains several vectors, and each vector represents the document that is relevant to the index term or phrase. The depth from the root to each leaf corresponds the term or phrase. We used five index classes to automatically build the index. Each class stores a particular type of indexed data, as listed below:

- Wikipedia Node: Index class that contains 3 subclasses: Common-Tags, Terms' Impact, and CRC-Dictionary. These subclasses hold the indexed Wikipedia documents from the ClueWeb09 collection.
- Home-Page Node: Index class that contains 2 subclasses: Domain-Name and Wikipedia-External-Links. These subclasses hold all the indexed home pages from the ClueWeb09 collection.
- Document-Title Node: Index class that contains all documents that were indexed using the phrases from their titles.
- Terms-Combination Node: Index class for which its nodes were labeled with the keywords selected from the urls and the titles of the documents; the content of the nodes hold vectors for significant terms.
- Topical-Keywords Node: Index class that holds all other documents, except Wikipedia pages and home pages. The documents in this class were indexed based on our collective phrases collected from Wikipedia titles and from the One Million Query Track at TREC; it also holds the domain names.

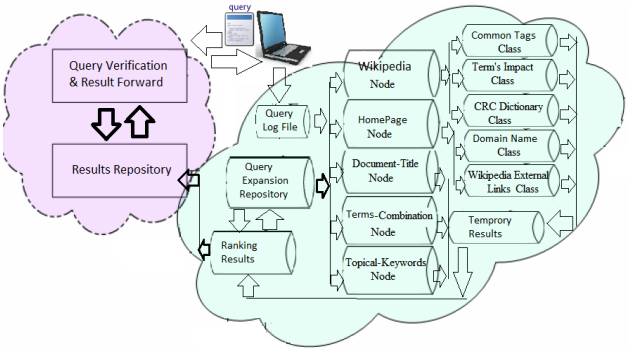


Figure 1: Our Index Structure

3.1 Wikipedia Index Node

Wikipedia is a collective knowledge reference of approximately 5 million pages to represent approximately 3 million articles in English. The data in each article is structured into several fields, and sometimes it has relationships with other articles using tags or links to expand certain topics. Each page has a unique vocabulary identifier. Sometimes Wikipedia uses different faceted vocabulary names to describe the same article. To efficiently index Wikipedia documents, we used three methods for grouping and clustering similar articles, in order to reduce the index size and the time required for indexing and searching. The first method is based on the document tags; this means that the documents that share similar tags are grouped together in one cluster; the second method is based on document's content; this means that the documents that share the significant terms (impacts) in their contents are grouped together; finally, the last method is based on the document titles; this means that the CRC code (see section 3.1.3) for the first paragraph of each document's content is computed; and then, the documents that share the same CRC values are grouped together. The second method is essential for retrieving the initial results from the Wikipedia index; whereas the two other methods are responsible for retrieving the corresponding (similar) documents.

[2] Golaxy Search Engine - http://www.golaxy.cn

[3] http://en.wikipedia.org/wiki/B-tree

3.1.1 Common-Tag Class

Some Wikipedia articles share the same tags (bold text). In our prospective, these articles are similar in topic. Thus, the content of each article was scanned for vocabulary terms and significant tags. Overall, each article name and its significant tags were used for creating and titling the index nodes; whereas the content of each node accumulates the document identifiers (as encoded by TREC). Hence, the index class contains vectors that represent documents.

3.1.2 Terms-Impact Class

Term frequency is important for detecting the topic of documents; but we need to look at different documents configurations in order to better measure the importance of the terms; for instance, two documents with the same terms frequency have different relevancy because two terms may occur in different topics and in documents of different lengths. To tackle this issue, we proposed to use term impact instead of term frequency. Term impact is our novel method based on using the impact of each term in the document's content. First of all, our model computed the frequency of each term in each document. Then, each term is mapped to the content by computing the cosine similarity between two vectors: the first vector represents the frequency of one meta-term, at a time (all values in the vector are zero, except the value of that meta-term which is represented by its frequency in the content); whilst the second vector was assigned by the frequencies of terms in the document's content. The length of both vectors is equal to the number of terms in the document's content. Since we have one value in the first vector and others are zeros, we used a normalized cosine similarity formula, as follows:

$$\text{Similarity (D, T)} = \sum_{j=1}^{m} \frac{(T_j)^2}{\sqrt{(T_j)^2} * \sqrt{\sum_{i=1}^{n} tf(T_i)^2}} \qquad (1)$$

where *m* is the number of meta-terms in document D, T_j is the term frequency of meta-term *j* in document D, *n* is the number of terms in the document-content D, and $tf(T_i)$ is the term frequency of term *i* in document *D*.

In order to filter out non-relevant articles (too short Wikipedia entries), our model assigned a strict threshold value to select the best similarity impact; it selects only the documents that have a total impact for the best terms greater than 3.5, regardless how many terms were summed; but the frequency of each term must be greater than 15. Otherwise, the documents were ignored and flagged as short articles (irrelevant).

The example below shows a sample of a class named "Martha"; where each vector represents a document; the first value represents the impact of term "martha" in the document's content, the second column denotes the document's TREC id; and the last column represents the document's title.

0.38 | Doc ID | Helen Lorraine
0.31 | Doc ID | prison, 0.27 | stewart, 0.40 | ImClone stock trading case
0.38 | Doc ID |Helen_Finney
0.32 | Doc ID | ray, 0.32 | The Honeymoon Killers
0.32 | Doc ID | raye, 0.37 | Martha Raye
0.31 | Doc ID | jefferson, 0.30 |wayles,0.27 | Martha Wayles Skeltonon

3.1.3 CRC-Dictionary Class

A cyclic redundancy check (CRC) is an algorithm to detect highly-similarity texts. We exploited this technique to find duplication in Wikipedia documents. Generally, Wikipedia articles that are very similar repeat the first paragraphs; whereas other paragraphs may or may not be repeated. We think that the documents that share the first paragraphs share similar topics. As a consequence, our approach used the CRC function with polynomial length of 16 to detect the documents that share the same paragraphs. Our system scanned through each document's content in the Wikipedia part of the ClueWeb09 corpus and generated the CRC value using only the header paragraph. We built a dictionary (a hash table) on the fly, with the generated CRC values used for representing the keys, and the value of each key holds the document identifier (the TREC identifier). In this way, through scanning all Wikipedia documents, the documents that share the same CRC values were aggregated in the contents of the keys. Finally, the dictionary was transferred to the index as vectors.

$$CRC=\sum_{j=1}^{m}\sum_{i=1}^{n} K_j(D_i) \qquad (2)$$

where m is the number of nodes (groups) in the index class, n is the number of documents that belong to the same group j, K is a key in the index class for group j; and D is the list of n documents that belong to group j. D was indexed as: D= {URL, Title, TREC-id}. TREC-id is the document's name in the collection.

In fact, this method supports other methods (Common-Tag and Term-Impact) for finding similar documents. In traditional search engines, displaying similar documents to the users is not preferred, but in the TREC data, the duplication needs because similar documents have different TREC IDs expected as solution. Our model uses this method to find similar documents (if a document was retrieved by the Term-Impact method, we also retrieved the ones with the same CRC, for the TREC submissions only). Otherwise, this method allows our search engine to remove duplication in the Wikipedia documents retrieved in the final ranked list.

3.2 Home-Pages Index Node

Home page finding is a challenging task. It is known that full-text relevance indexing is not particularly effective for home page finding. This was demonstrated in the TREC-2001 home page finding task. The best system used evidence from the URLs evidence in order to predict the home pages [12]. However, a home page is not restricted to the home domains, but it should be the first page on the final ranked list. Since the first page requires more effort to process, we used two methods for predicting the home pages: the first method is based on domain names, and the second method is based on Wikipedia articles. The domain names are assigned a higher priority in the results list.

3.2.1 Domain Names Class

Domain names and urls are very important for the home page finding task [13]. For each domain, the home page is defined by the shortest url. We used two phases for aggregating the sparse documents. The first phase indexed the documents

that belong to the same domain in the same index node; whilst the second phase generated a tree of urls, in alphabetic order.

However, the documents that involved the shortest urls were classified as the home pages. We created a node in domain name class for each home page. The name of node was named by the combination of domain name and document title; whereas the content of node holds the TREC id for the home page and all documents that belong to that domain (urls and titles of documents were also indexed). We wrapped the titles of the documents to the domain names because some of the domain names involve abbreviations; for instance the title "civil rights movement" is referred to the domain name "crmvet". Thus, the index node could be accessed by either query string.

$$DNC=\sum_{i=1}^{n} K(D_i) \quad (3)$$

where n is the number of domain names in the index class, *K* is a node in the index class for domain's name *i*; and *D* is the list of documents that belong to domain *i*. *D* was indexed as: D= {URL, Title, TREC-id}

3.2.2 Wikipedia External-Links Class

Wikipedia is often a good reference for most home pages. Researchers used the external links in the Wikipedia repository for the home-page finding task and potentially work better than searching anchor texts for the same task [12]. Many official home pages are referenced by the Wikipedia writers in the external links section. Since Wikipedia is a large portion of the ClueWeb09 corpus, our model aimed to index all external anchor texts and their urls. Wikipedia uses the terms "Official", "Website", and "Home page" to represent the external home pages for the related articles. Our model used the JavaScript "JS Regex" function to match these terms and to extract the corresponding urls and its anchors. Consequently, each home page was used to create the node in the Wikipedia external-links class. The nodes were labeled by the titles of the referred home pages. The content of each node holds vectors of home pages; each vector is represented by anchor text, url, and the TREC id.

3.3 Document's Title Index Node

The titles of documents may contain terms and phrases relevant for indexing; especially with the documents that consist of only a few words in their content for indexing. The phrases in the titles are often connected together by using conjunctional words, i.e., "or" , "and" , "at" , "in" , "on" , "by" , "with" , "from" , or "for"; or a punctuation characters, i.e., ":" , "|" , "(" , ")" , "-" , "," , or "&". Thus, segmenting the titles of documents into phrases is essential in order to find the most important key phrases for the documents. We used these characters and function words for partitioning the titles into list of terms and phrases. More concretely, the terms and phrases do not have equal importance; some of the terms are more important than others. Also, the same phrases available in two documents do not have the same importance. To address this issue, we used our module "1" (section 3.1.2) for computing the impact of each segment (term or phrase) in its content; we computed the similarity between the vector represented by each segment separately and the vector that represents the document content (the similarity issue was based on term frequency). Finally, the fragment title was used to create and name the class node; whereas the content of each node holds a set of vectors represented by the term impact, document id (TREC id), and url.

Let's consider that we have three documents with the following titles ""civil rights movement- period 1", ""civil rights movement- period 2", ""civil rights movement- period 3". The fragment "civil rights movement" is repeated in three documents. The impact of this fragment in each of the three documents, as computed by equation 1, is as follow:

[0.537 | doc id | url], [0.152 | doc id | url], [0.421 | doc id | url]

3.4 Terms Combination Index Node

Usually, a query is combined from keywords that are located in different positions in the documents; for instance, some terms are located in the urls; whereas the remaining terms are located in the document's content or in the title. The combination index class focuses on these types of queries. First of all, the frequency of each term in the document was computed. Second, the keywords of url and title of document were combined and parsed in one vector to remove the repeated terms. The normalized vector was split into terms. Then, for each term, the impact value was computed (module 1) and specified in a range (0.5-1.0) to be a representative for the document (the best three values were selected). Finally, the best three terms were chosen for creating and naming the index nodes; whereas the content of nodes is a set of vectors; each vector was composed from the impact value of a representative term and all terms with their frequencies in the document content (TREC id is also wrapped). Precisely, for each document, we used a strict cut-off weighting value (threshold) ranged between 0.9 and 3.0. If the representative terms have impacts higher than 3.0, the document was flagged as spam; likewise, if the total impact for representative terms was less than 0.9, the document was ranked as a junk, as shown in figure 2.

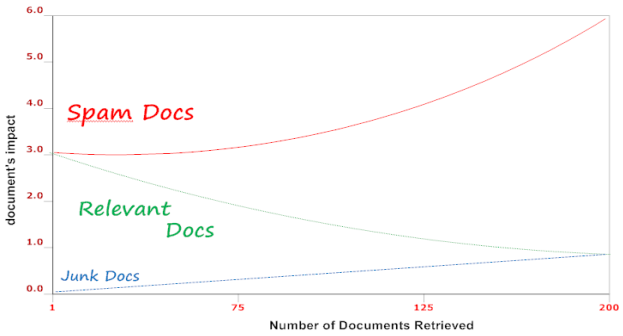


Figure 2 : Document Categories vs. Document Impact

3.5 Topical Keywords Index Node

Topical phrases are difficult to collect and they require a high level of parsing. Generally, topical phrases are available in the external resources; such as web-based encyclopedia. Our index class in this type was based on collecting phrases from the three following resources:

- Wikipedia titles: because Wikipedia collection is rich on marked phrases; our model gathered all the titles from the Wikipedia articles which have significant term impacts using

"Terms-Impact Class" and the documents that did not have representative weights in their terms were discarded;

- Query logs: the query logs from the "Million Query Track"[4] also contains rich keywords about frequently used phrasal queries; and finally, the keywords in the main URL domains were collected.
- The keywords in the main URL domains were collected.

The gathered collection was combined in one list (the duplicated words were removed). The final size for all elements in the list was approximately 7 million topical phrases and terms. In this class, we aimed to index all the documents that focus on each element in the list. Our model used two phases for completely indexing the whole collection. The first phase was responsible for detecting the validity of the topics in each document in our corpus, whereas the second phase was responsible for detecting the weights (impacts) of keywords/keyhprases in the documents and then in its domain, as explained below.

3.5.1 Document-Keywords Validation

The first step in this phase was detecting the validity of the index terms in the documents, and then assigning each document to a set of terms. We used three hash tables ($\partial 1, \partial 2,$ and $\partial 3$) that function cooperatively. Briefly, the first table $\partial 1$ assigned the position value for each element in the list. The second table $\partial 2$ tokenized the elements in the table $\partial 1$ into keywords and wrapped each single term with its position. Then, the terms of the first paragraph in each document in the collection were parsed sequentially and distributed in table $\partial 2$. Then, table $\partial 3$ aggregated the documents with the same key from the table $\partial 1$ and with the same position from table $\partial 2$.

3.5.2 Weighting the Keywords/Keyphrases

Often, keywords/Keypharses available in the documents are not enough to distinguish their topics; for instance, if a site focuses on a topic "civil right movement", all documents that belong to that site may contain the phrase "civil right movement". Generally, researchers use the inverse document frequency (*idf*) which is based on the number of documents that contain a term in that site, to find terms with high discriminant power. In our case, through processing all documents in the collection, the content of dictionary $\partial 3$ already captured the documents that belong to the same site. Our method is different from the standard *idf*; it uses the inverse document frequency based on the frequency of topical phrases not individual words, because it is more robust. In this section, we will focus on computing the impact of phrasal keywords in the document with respect to its domain.

a) Document-Topic Weighting

Long keyphrases may have different distributions in some documents. Therefore the same topical keyphrases might be compiled for different topics. The dictionary $\partial 3$ computed the impact of each term in its document's content. The results were saved in the vector space format, as follows:

$[K_1] \rightarrow [D_1,\ url,\ SV]\ [D_2,\ url,\ SV] \ldots [D_n,\ url,\ SV]$

$[K_2] \rightarrow [D_1,\ url,\ SV]\ [D_2,\ url,\ SV] \ldots [D_m,\ url, SV]$

$[K_j] \rightarrow [D_1,\ url,\ SV]\ [D_2,\ url,\ SV] \ldots [D_l,\ url, SV]$

where K is the keyword in dictionary $\partial 3$ to represent set of documents, D is the document TREC-id, url is the document link, and *SV* is the similarity (impact) value, j is the number of keys in the dictionary $\partial 3$.

b) Domain-Topic Weighting

Finally, all the keywords that belong to the same domain (site) were meet at the same keyphrases in the dictionary $\partial 3$. Computing the inverse document frequency based on phrases reflects the topic of document/site better than *idf* based on single terms. In fact, *idf* computes the frequency in the whole site, not in a specific subdomain; but not all documents in the same domain have similar contribution to the topic. For example, the query "University of Phoenix" refers to the domain "phoenix.edu", but not all the documents in the domain "phoenix" have equal relevancy to query "phoenix". To address this, our model computed the impact of a document in its subdomain with respect to the term frequency. First of all, the documents that belong to each keyword in the dictionary $\partial 3$ were sorted alphabetically. Thus, each key in the dictionary $\partial 3$ was structured hierarchically (main domain, subdomains, subdirectories, and files). The size of a subdirectory (subtree) is important for computing the weight of the nested documents. We applied a top-down traversal algorithm[5] for computing the weights of subtrees.

Each parent in the subtree counted the nested documents and summed their impacts. Then, the tree sorted its nested subtrees (or documents) with respect to the number of their children. To customize the content of subtrees optimally regarding the number of documents, our model used an automatic cutoff value. The cutoff value was based on the number of documents in subtrees, as well as the contributions (impact) of subtrees. Initially, the elimination started at frequency 1 and then it increased until the number of documents in the whole tree was equal to 200. We chose a threshold value of 200 to keep a balance between the precision and the recall value. However, the old tree was pruned and bounded to a new tree, and each parent had an optimal subtree size. The total impact weight of each subdomain was computed as follows:

$$Weight\ (D, I) = \sum_j w_j / N \qquad (4)$$

where w is the impact weight of document j in the subtree I and N is the number of documents in the subtree I.

The final tree was assigned to the index class; the root of the tree (the main parent) created and labeled the main node in the index class; whereas the children (documents) were bounded in the space of vectors represented as: total impact value, document TREC-id, and url.

[4] http://trec.nist.gov/data/million.query.html

[5] http://www.cs.umd.edu/~hjs/pubs/SametPAMI85b.pdf

4. QUERY PROCESSING and DOCUMENT RANKING

Query processing is an essential part in a search engine. It includes detecting the type of the query, query searching, query normalization and expansion. First of all, our approach processes all queries that have only one term. Then, for query length of more than one term, the search process removes the spaces between terms and combines the query terms into one term (with and without hyphens to replace the spaces). Then, the query is forwarded to the domain-name index class. If the query matches any node in the index class, the home page is extracted and forwarded to the topical keyword-class index to find the node of the extracted domain and to find its documents that have good impacts for the terms. If the query does not match the nodes in the domain-name class, the original query is forwarded to the Wikipedia index class. If the query matches any node in that class, the home page is extracted, and the domain name is forwarded to the Domain-Name index class to retrieve the relevant pages. Often, some Wikipedia articles do not contain home pages; in this case, the external pages in the Wikipedia class were flagged to be used later for result enhancement. Finally, if the query does not match any node in the Wikipedia index class, it is forwarded to the Term-Combination index class and then to the Document-Title index class.

Some queries match more than one index class and others match only one. The flowchart from Figure 3 shows the query processing stages.

As an example, we present several queries form the TREC Web Track 2012 and their corresponding index class types:

- The Domain-Name index class: e.g., "arkansas", "quitsmoking", and "newyork-hotels".
- The Wikipedia index class: e.g., "churchhill downs" and "indiana state fairgrounds".
- The Title-Based index class: e.g., "becoming a paralegal".
- The Term-Combination index class: e.g., the query "gs pay rate" in "www.gspay.com" or the query "brooks brothers clearance" that refers to the site "brooksbrothers.com" where the terms "rate" and "clearance" are extracted from their content, respectively.
- The Topical Keywords-based index class: e.g., the queries like "black history", "septic system design", "dogs clean up bags" or "furniture for small spaces".

Typically, each search engine has a certain criterion for manipulating search results. There are two types of search preferences: user dependent and user independent. User dependent works when users add preferences to query results; whilst user independent means that search engines use their own preferences to bias one site over others. In our model, we tried to redistribute the documents in the final list to compromise the preferences that satisfy the user needs for both tasks (adhoc and diversity):

- Homepages (".com", ".gov", ".org", ".edu", .., etc.).
- Wikipedia results whose titles matched the query literally.
- Site Preferences ("about.com" and "answers.com"), if they are situated in the top 20 in the ranked list (we used these sites as user's preferences because TREC prefers them as valuable informative sites in the diversity task).
- Top ten results that ranked high, regardless of the type of sites.
- Other Wikipedia results that ranked high based on their contents.

Figure 3 shows the flowchart for query processing steps. As we can see, the results pass through index nodes based on the result from each node, and the final result determine the type of task: adhoc, diversity, or both. Figure 3 shows five index nodes, as explained in figure 1 above (4 index nodes as well as two index classes, in which the "Domain Name" and "Wikipedia External Links" classes belong to the 'Home Page" node).

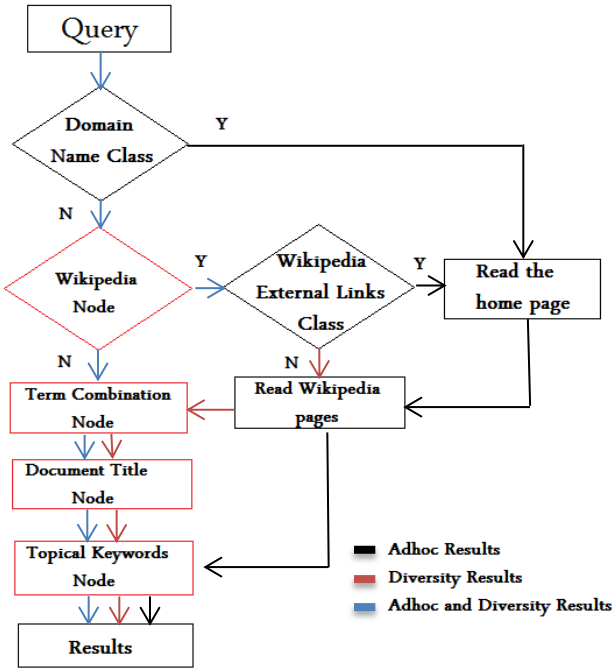


Figure 3: Query Processing Flowchart

5. QUERY EXPANSION

Search engines use query expansion to increase the quality of search results. It is assumed that users do not always formulate search queries using the best terms [17]. User's feedback is important for query expansion. Most search engines use this technique to expand the initial results and satisfy the user needs. Our approach uses an implicit user's feedback; that is, it computes the behaviour of some Wikipedia writers who adopted the preferences in the dynamic properties of the Wikipedia collection. Our approach used two algorithms for query expansion: the shared-links and the manner of titling the similar articles:

- **Using Shared-Links:**

We assume if an article (A) in the Wikipedia has a link that points to an article (B), and the article (B) has a link that points backward to the article (A), the two articles A and B are related,

topically. Therefore, our approach indexed all incoming and outgoing links for each article, by building a hash-table in the fly, firstly. The article names represent the keys of the table and the outgoing links are stored in the contents of corresponding keys. Secondly, the index was mapped into the physical disk.

Figure 4 shows an example for expanding the query "Global Warming" by the terms "Carbon Dioxide", "Air Pollution", "Greenhouse Gas", and "Alternative Fuel".

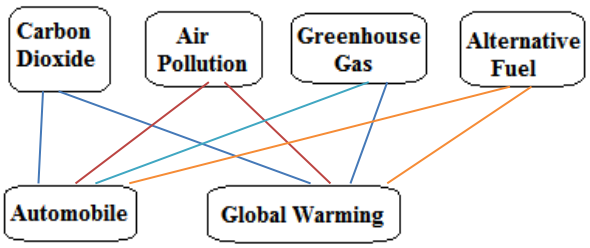


Figure 4: Global Warming and related articles

- **Using Titling-Variation Aspect**

Often, Wikipedia writers use different strategies for entitling the articles. Often, highly similar articles are named by different titles. Hence, these variations have been used to expand queries in our model; this means that if the initial ranked list contains results from Wikipedia, the title of article will be used to collect the other corresponding titles. For example, an article "Lipoma" is titled by Wikipedia writers as: "Fatty Tumor", "Fatty Lipoma", "Lypoma", "Lipomatous Neoplasm", "Lipomas", and "Lipomatosis"; where the contents of these articles are similar. If our model is queried by one of these titles, it will use the other terms for query expansion.

However, query expansion and reformulation in our model is not used for all queries; it is used only when the initial ranking list is short (less than 200 pages). This works well when the initial query retrieves at least one Wikipedia document from the index. For instance, the initial result list for the query "angular chelitis" is too short; the expansion with "angular stomatitis" increased the number of results retrieved by our model.

6. EXPERIMENTAL EVALUATION

Since TREC evaluation campaigns provide sets of queries and relevance judgments (expected solution as a list of relevant documents), we submitted the results of our new model to TREC web track 2012, for evaluation, for the two tasks: adhoc, and diversity. The diversity task is similar to the adhoc retrieval task, but differs in its judging process and evaluation metrics. 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. The primary effectiveness measure for the adhoc task is expected reciprocal rank (ERR) as defined by [15]. We also report a variant of nDCG [25], as well as standard binary measures, including mean average precision (MAP) and precision at rank k (P@k). TREC computes ERR at rank k (ERR@k) and nDCG at rank k (nDCG@k), as follows:

$$\text{ERR@k}=\sum_{i=1}^{k}\left(\frac{R(g_i)}{i}\prod_{j=1}^{i-1}(1-R(g_i))\right) \quad (5)$$

where $R(g)=\frac{2^{g}-1}{16}$ and $g_1, g_2, g_3, .., g_n$ are the relevance grades associated with the top k documents.

$$\text{nDCG@k}=\frac{\text{DCG@k}}{\text{ideal DCG@k}} \quad (6)$$

where:

$$\text{DCG@k}=\sum_{i=1}^{k}\frac{2^{g_i}-1}{\log_2(1+i)} \quad (7)$$

The primary effectiveness measure for the diversity task is the intent-aware expected reciprocal rank (ERR-IA) [16]. 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.

For some queries, our system obtained good precision, but for a few queries, the precision was low because the relevance judgments[6] contained only documents selected from the category the whole ClueWeb09 collection (category A), while we used only a part of the collection (category B). In a few 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. TREC evaluated all the systems on the 50 testing queries from 2012, without separating those that used full dataset (category A) from those who used the subset B. The test set of queries were selected by TREC to represent both tasks and involve different complexity of topics.

For each task, TREC used four evaluation metrics; each metric uses different strategy of raking to represent different views. Table 1 and 2 show the results of the automatic evaluation for both the adhoc and diversity tasks, on the training set of queries (50 queries) for the top systems that participated in the TREC 2012 Web track. Figure 5 shows more details about our results for both tasks, for each query in the upper part and for the 50 test queries in the lower part. The 50 queries are numbered from 150 to 200 according to the TREC Web track topics.

7. PERFORMANCE EVALUATION

The evaluation metrics that we discussed previously concerned the correctness of the retrieved results. In this section we look at the execution time (speed) for our approach.

Figures 5 and 6 show the progress graphs regarding the indexing data per day and the query processing response time (in millisecond) that we were able to achieve it by this system. As

[6] The relevance judgment file is made by the TREC assessors and consists of a list of documents that are relevant answers to each query.

we can see, the indexing speed is high at the beginning of the processing because at the initial point the index was empty and the indexer only accessed the disk once (writing). At every step, the writing time also depended on the size of files been indexed. During the indexing, the process took a bit longer in some places, because the indexer required accessing the disk twice (reading and writing) and the operation was appending (reading the old node, appending the new data, and creating the new node).

Likewise, query response time differs from query to query and it is variable because the response time depends on the size of the index node, as well as on the type of query (the diversity queries take a bit longer than the others).

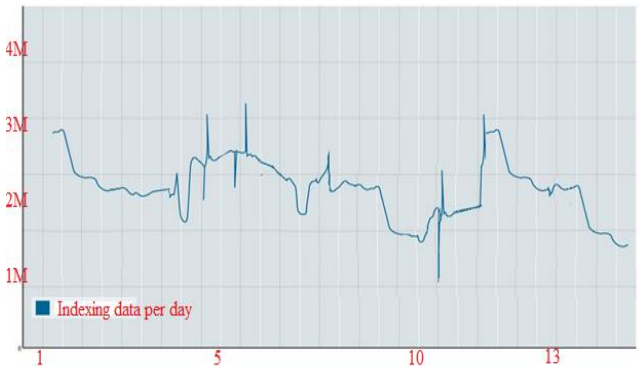


Figure 5: Number of documents indexed per day

Figure 6: Query processing response time

8. COMPARISON WITH OTHER MODELS

We compare our approach to the other approaches described briefly in the related work section. Table 1 and 2 compare our model with other models using the same 50 test queries. Only the top 8 systems are shown, as presented in the TREC 2012 web track overview paper [25]. From the tables 1 and 2, we can conclude that the approaches that obtained high precision are those that used document's content for indexing, but they used different assumptions about how to use it in order to measure document relevancies. Different features were used by researchers to target documents for specific topics. Often document-topic finding is implemented as part of a content filtering process, where only desirable content is kept in the cache for indexing, and undesirable content is prevented from being indexed. The documents whose meta-contents have strong integration and coordination with document's content seem to be highly relevant to the underlying topics because they are focused on the few keywords available in the document titles and urls.

Table 1: Top adhoc task results ordered by ERR@20 for the best systems over 48 systems.

Group	CAT	ERR@20	nDCG@20	P@20	MAP
uogTr	A	0.313	0.238	0.453	0.212
Srchvrs	A	0.305	0.176	0.315	0.126
uOttawa	B	0.299	0.214	0.405	0.120
QUT_Para	B	0.290	0.167	0.305	0.117
Utwente	B	0.219	0.133	0.221	0.061
ICTNET	A	0.215	0.110	0.257	0.078
IRRA	B	0.173	0.143	0.367	0.153
Qutir12	B	0.166	0.146	0.308	0.131

Table 2: Top diversity task results ordered by ERR-IA@20 for the best systems over 48 systems.

Group	CAT	ERR-IA@20	α-nDCG@20	NRBP
uogTr	A	0.505	0.606	0.463
uOttawa	B	0.431	0.525	0.394
Utwente	B	0.405	0.508	0.357
Srchvrs	A	0.386	0.485	0.340
ICTNET	A	0.326	0.422	0.280
Udel	A	0.325	0.419	0.282
LIA	A	0.318	0.424	0.268
Udel_fang	B	0.300	0.420	0.241

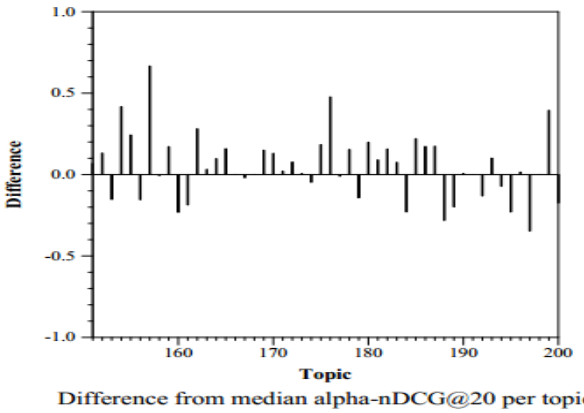
To enforce topic finding in document's content, several features and sources of evidences were proposed by researchers, and methods were ranked according to the type of features used. For example, University of Glasgow's approach 'uogTr' was ranked first because they use many features, including term weighting, url and link terms, spam detection, term dependencies, etc. Our approach used the first three features; but others seem making additional impacts for result's better precision. University of Twente approach 'Utwente' also captured document content with anchor texts in the indexing process. Other systems that ranked high used different strategies for indexing document content with anchor texts and url; that is, they separated the indexing process from the ranking process and each process used different types of data; for example document content was used for indexing and anchor text was used for ranking. In fact, anchor texts are implicitly similar to, or part of, the texts in the titles and urls of target documents and they help if they are used in the same process. Our approach is different than other approaches; in which, it measures the impact of the query terms in the document's content. Our approach considers that each query must be weighted differently and we enhance the results based on the initial results. We used five index classes and the query is passed through each class (see figure 3); so the initial result that obtained from each class is based on the priority of that class, that corresponds to the type of query, and is also based on the corresponding results from each class and how we feed the query to other classes based on the results from current class. Researchers at University of Twente indicated as future direction to investigate the per-query

processing as a way to obtain better precision. The Chinese Academic of Science's system 'Srchvrs' also used document's content, title, and url; but they combined for indexing, and the anchor texts were aggregated for ranking. By contrast, our approach used the impact of each keyword from the meta-data to reflect the topic in the document's content. Another significant step in our approach was using the result-distribution per query. We used different distributions for the ranked lists based on the type of query. The type of query was computed through passing the results among index classes, as explained above (figure 3).

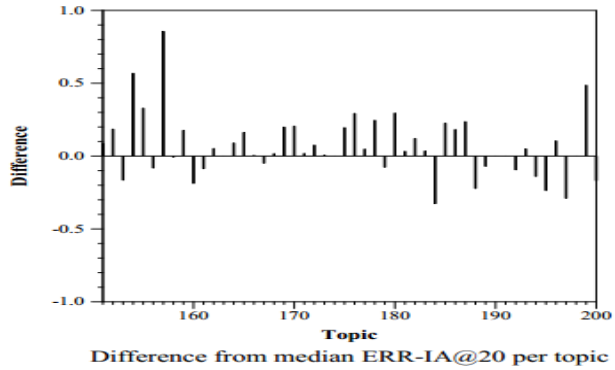
The TREC evaluation reported four metrics; each metric uses a different strategy of raking to represent different views of the users. Specifically, TREC evaluated each topic (query) separately for each of the two tasks (adhoc and diversity). As we can see in figure 7, for some queries the results were lower because our system could not retrieve relevant answers (many of them were in the part of the collection that we did not index, since we only indexed subset B); whereas other results were very good (most of the relevant documents were available in the subset B). Overall, as we showed in table 1 and 2 above, our approach "uOttawa" is better than other approaches for the subset B of the ClueWeb09 collection for both tasks, but it is second on the list when compared with all models submitted to TREC regardless what size of data was used, for the diversity task, and it is third on the list when compared with all models for the adhoc task. The subset collection "B" requires 5TB of disk space, while the whole collection "A" requires 25TB of disk space. We did not have 25TB of disk space available; this is why we indexed only the subset B. Nonetheless, we did not lose much in terms of result. We present our results in comparison with the best results over all the 48 systems submitted to TREC for 50 testing queries, though table 1 and 2 show only top 8 systems.

Summary Statistics	
Run ID:	DFalah121D
Task :	diversity
Run type :	automatic
Document collection category:	B
External resources used:	no additional resources
Number of topics:	50

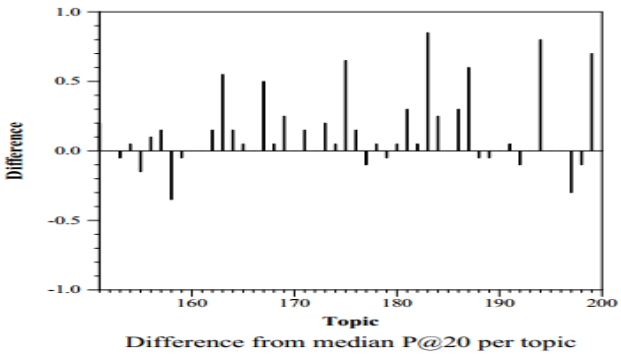
Adhoc measures		Diversity measures	
Retrieved	13109	α-nDCG@10	0.4910
Relevant	3523	α-nDCG@20	0.5253
Relevant retrieved	802	ERR-IA@10	0.4222
Prec@10	0.4420	P-IA@10	0.3414
Prec@20	0.4050	P-IA@20	0.3177
MAP	0.1203	MAP-IA	0.1052
NDCG@20	0.2135	NRBP	0.3940
ERR@20	0.2992	ERR-IA@20	0.4315



Difference from median alpha-nDCG@20 per topic



Difference from median ERR-IA@20 per topic



Difference from median P@20 per topic

Figure 7: TREC Adhoc and Diversity measures for our model.

9. CONCLUSIONS

In this paper, we presented our custom indexing and ranking model, developed for the TREC 2012 web track. We explained how the Wikipedia content and document phrases can cooperatively support finding of relevant results with low disk overhead and increased retrieval performance. Our method is based on topic identification in the documents' content through the investigation of keywords/keyphrases from meta-data with high impact values. Our model provides a variety of analytic capabilities, including: phrases extraction, keywords correlation, web page topic finding, documents grouping, phrase-based inverse document frequency, and document versus domain topics weighting. Unlike our approaches in the previous years 2010 and 2011 [28], this approach is more sophisticated and more robust for processing all types of queries.

In future work, we plan to experiment with new types of queries using additional types of resources; we also plan to investigate the topics in the documents based on the document / site topic correlation.

10. ACKNOWLEDGMENTS

This work is part of MOHESR-Iraq Scholarship. We extended our thanks to the anonymous reviewers from WIMS and to the TREC Web Track organizers for evaluating our approach and for providing us with the dataset.

11. REFERENCES

[1] Dalton, J. and Dietz, L., (2012), "Bi-directional Linkability FromWikipedia to Documents and Back

Again", In Proceeding of TREC 2012 Knowledge Base Acceleration Track.

[2] Kamps, J., Kaptein, R., and Koolen, M., (2010), "Using Anchor Text, Spam Filtering and Wikipedia for Web Search and Entity Ranking", In Proceeding of 2010 Web TREC Track.

[3] Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R., Peng, Y., Reddivari, P., Doshi, V., and Sachs, Y., (2004) "Swoogle: A Search and Metadata Engine for the Semantic Web", In Proceedings of ACM 1581138741/04/0011.

[4] Lawrence, S. and Giles, C., (1999), "Accessibility of information on the web". Macmillan Magazines Ltd.

[5] Amitay, E., (2001), "What Lays in the Layout: Using anchor-paragraph arrangements to extract descriptions of Web documents". Doctoral thesis, Macquarie University.

[6] Anh, V. and Moffat, A., (2010), "The Role of Anchor Text in ClueWeb09 Retrieval". In Proceedings of the 18th Text Retrieval Conference (TREC).

[7] Craswell, N., Hawking, D., and Robertson, S. (2010), "Effective site finding using link anchor information". In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, pages 250-257.

[8] Kaptein, R., Koolen, M., Kamps, J. (2010), "Result Diversity and Entity Ranking Experiments: Anchors, Links, Text and Wikipedia". In Proceedings of the 19th Text Retrieval Conference (TREC).

[9] MacKinnon, I. and Vechtomova, O. (2008), "Improving Complex Interactive Question Answering Enhanced with Wikipedia Anchor Text". In Proceedings of Advances in Information Retrieval, 30th European Conference on IR Research (ECIR).

[10] Xing, Y. and James, A. (2010), "A Content based Approach for Discovering Missing Anchor Text for Web Search", In Proceedings of SIGIR'10, pages 19–23.

[11] Heyuan L., Yuanhai X., Shaohua G., Feng G., Xiaoming Y., Yue L., Xueqi C., (2012), "ICTNET at Web Track 2012 Ad-hoc Task", In Proceedings of TREC 2012 Web Track.

[12] Craswell, N. and Hawking, D., (2010), "Query-Independent Evidence in Home Page Finding", Trystan Upstill, Australian National University and CSIRO Mathematical and Information Sciences.

[13] Baykan, E., Henzinger, M., Marian, L., Weber, L. (2009) "Purely URL-based Topic Classification", Ecole Polytechnique, Google, Lausanne, Switzerland.

[14] Deveaud, R., Juan, E., and Bellot, P. (2012) "LIA at TREC 2012 Web Track: Unsupervised Search Concepts Identification from General Sources of Information", In Proceedings of TREC 2012 Web Track.

[15] Chapelle, O., Metzler, D., Zhang, Y., and Grinspan, P. (2009). "Expected Reciprocal Rank for Graded Relevance", Yahoo Labs and Google Inc, Santa Clara CA, Sunnyvale CA, and San Bruno CA. ACM.

[16] Teerapong Leelanupab, Guido Zuccon, and Joemon M. Jose. "Is Intent-Aware Expected Reciprocal Rank Sufficient to Evaluate Diversity?". Advances in Information Retrieval, 35th European Conference on IR Research, ECIR 2013, pp 738-742.

[17] Mark Sanderson, Monica Lestari Paramita, Paul Clough, Evangelos Kanoulas. "Do user preferences and evaluation measures line up?". In proceedings of SIGIR'10, 2010, ACM 978.

[18] Hang Cui, Ji-Rong Wen, Jian-Yun Nie, and Wei-Ying Ma. "Query Expansion by Mining User Logs". In Proceedings of e IEEE Computer Society, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, 2003.

[19] Limsopatham, N., McCreadie, R., Albakour, M., Macdonald, C., Santos, R., and Ounis, I. (2012). "University of Glasgow at TREC 2012: Experiments with Terrier in Medical Records, Microblog, and Web Tracks". In Proceedings of TREC Web Track.

[20] Symonds, M. Zuccon, G., Koopman, B., and Bruza, P. (2012). "QUT Para at TREC 2012 Web Track: Word Associations for Retrieving Web Documents", In Proceedings of TREC 2012 Web Track.

[21] Zheng, W. and Fang, H. (2012). "Exploiting Ontologies for Search Result Diversification". In Proceedings of TREC 2012 Web Track.

[22] Dong Nguyen and Djoerd Hiemstra, "Ensemble Clustering for Result Diversification". In Proceedings of TREC 2012 Web Track.

[23] R. Sarikaya, A. Gravano, and Y. Gao, " Rapid Language Model Development Using External Resources for New Spoken Dialog Domains", IEEE International Conference (Volume 1), 2005, Pages 573 – 576.

[24] D. Mioduser, R. Nachmias, O. Lahav, and A. Oren, (2000). "A Web-based learning environments: Current pedagogical and technological state", Journal of Research on Computing in Education, page 55.

[25] C.L.A. Clarke, N. Craswell, and E.M. Voorhees, "Overview of the TREC 2012 Web Track". In Proceedings of TREC 2012, the Twenty-First Text Retrieval Conference, NIST Special Publication: SP 500-298, 2012.

[26] Francisco João Pinto and Carme Fernández Pérez-Sanjulián. "Automatic query expansion and word sense disambiguation with long and short queries using WordNet under vector model". Actas de los Talleres de las Jornadas de Ingeniería del Software y Bases de Datos, Vol. 2, No. 2, 2008.

[27] Paha, N. and Gulati, P. ; Gupta, P. "Ontology driven conjunctive query expansion based on mining user logs". Proceeding of International Conference on Methods and Models in Computer Science, 2009. ICM2CS 2009., Pages 1-4.

[28] Al-akashi, F. and Inkpen, D. (2012). "Intelligent Web Page Retrieval Using Wikipedia Knowledge". In Proceedings of the 2nd Web Intelligence, Mining and Semantics (WIMS) International Conference.