

# Integration of Multiple Evidences based on a Query Type for Web Search

In-Ho Kang<sup>\*</sup>, Gil Chang Kim

*Division of Computer Science*

*Department of EECS, KAIST*

*373-1 Guseong-dong, Yuseong-gu, Daejeon 305-701, Republic of Korea*

---

## Abstract

The massive and heterogeneous Web exacerbates IR problems and short user queries make them worse. The contents of web pages are not enough to find answer pages. PageRank compensates for the insufficiencies of content information. The content information and PageRank are combined to get better results. However, static combination of multiple evidences may lower the retrieval performance. We have to use different strategies to meet the need of a user. We can classify user queries as three categories according to users' intent, the topic relevance task, the homepage finding task, and the service finding task. In this paper, we present a user query classification method. The difference of distribution, mutual information, the usage rate as anchor texts and the POS information are used for the classification. After we classified a user query, we apply different algorithms and information for the better results. For the topic relevance task, we emphasize the content information, on the other hand, for the homepage finding task, we emphasize the Link information and the URL information. We could get the best performance when our proposed classification method with the OKAPI scoring algorithm was used.

*Keywords:* Information Retrieval, Content Information, PageRank, Query Type

---

## 1 Introduction

The Web is rich with various sources of information. It contains the contents of documents, web directories (e.g. Yahoo), multimedia data, user profiles and

---

<sup>\*</sup> Corresponding author.

*Email addresses:* `ihkang@csone.kaist.ac.kr` (In-Ho Kang),  
`gckim@cs.kaist.ac.kr` (Gil Chang Kim).

so on. The massive and heterogeneous web document collections as well as the unpredictable querying behaviors of typical web searchers exacerbate Information Retrieval (IR) problems. Retrieval approaches based on the single source of evidence suffer from weaknesses that can hurt the retrieval performance in certain situations (Croft, 2000). For example, content-based IR approaches have a difficulty in dealing with the diversity in vocabulary and the quality of web documents, while link-based approaches can suffer from an incomplete or noisy link structure. Combining multiple evidences compensates for the weakness of single evidence (Westerveld et al., 2001, Yang, 2001). Fusion IR studies have repeatedly shown that combining multiple evidences can improve the retrieval performance (Croft, 2000).

However, previous studies did not consider user queries in combining multiple evidences. Not only the Web but also user's queries are diverse. For example, for user query '*Mutual Information*', if we count on link information too highly, well-known site that has '*mutual funds*' and '*information*' as index terms gets a higher rank. For user query '*Britney's Fan Club*', if we use content information too highly, yahoo or lycos's web directory pages get a higher rank, instead of '*Britney's Fan Club Site*'. Like these examples, combining content information and link information is not always good. We have to use different strategies to meet the need of a user. If we can classify a user query, we can refine candidate pages and rank them delicately. User queries can be classified as three categories (Broder, 2002).

- topic relevance task (informational)
- homepage finding task (navigational)
- service finding task (transactional)

The topic relevance task is a traditional ad hoc retrieval task where documents (web pages) are ranked by decreasing likelihood of meeting the information need provided in a user query (Hawking & Craswell, 2001). Users are interested in finding as much information as possible, in the topic relevance task. For example, '*What is a prime factor?*' or '*prime factor*' is the query of the topic relevance task. The goal of this query is finding the meaning of '*prime factor*'.

The homepage finding task is a known-item task where the goal is to find the homepage (or site entry page) of the site described in a user query. Users are interested in finding a certain site. For example, '*Where is the site of John Hopkins Medical Institutions?*' or '*John Hopkins Medical Institutions*' is the query of the homepage finding task. The goal of this query is finding the entry page of '*John Hopkins Medical Institutions*'.

The service finding task is a task where the goal is to find documents that provide the service described in a user query. Users are interested in the service of a result page. For example, '*Where can I buy concert tickets?*' or '*buy concert*

*tickets*' is the query of the service search. The goal of this query is finding documents where they can buy concert tickets.

Users may want different pages with the same query. We cannot always tell the class of a query clearly. But we can tell most people want a certain kind of pages with this query. In this paper, we calculate the possibility whether the class of a user query is the topic relevance task or the homepage finding task. Based on this possibility, we combine multiple evidences dynamically. In this paper, we consider the topic relevance task and the homepage finding task only. Because the proposed method is based on the difference of databases, we can apply the same method to classify the service finding task.

Usually we extract result documents from one database. We refine result documents by the change of scoring algorithms. From general web documents, we can extract site entry pages by the change of a scoring function. However, we may create different databases for each query category. But in this case, we need human intervention and special knowledge to make databases. In this paper, we extract result documents from one database. Also we extract classifiers for a query classification from two databases that are automatically constructed from the original database.

In this paper, we present a user query classification method and a combining method for each class. In section 2, we describe various types of evidence for IR and previous methods of combining multiple evidences. Section 3 lists the differences of search tasks and the properties of Content, Link and URL Information. In section 4, we present the model of a query classification. In section 5, we experiment with our proposed model. Conclusion is described in section 6.

## 2 Multiple Sources of Evidence

In this section, we explain three types of evidence for IR.

### 2.1 Content Information

There are multiple types of representations for a document. These representations typically contain titles, anchor texts, and main body texts (Croft, 2000). A title provides the main idea and the brief explanation of a web document. An anchor text is the underlined and highlighted text of a hyperlink in web documents. This anchor text can be added to linked documents. It provides the description of linked web documents and files. An anchor text often pro-

vides more accurate description of a web document than the document itself. An anchor text may exist for a document that cannot be indexed by a text-based search engine, such as images, programs, and databases. This makes it possible to return web pages that have not actually been crawled.

We usually use TFIDF,  $tf$  and  $(1/df)$ , to calculate the relevance of given web documents (Baeza-Yates & Ribeiro-Neto, 1999).  $tf$  is the raw frequency of a given term inside a document. It provides one measure of how well that term describes the document contents.  $df$  is the number of documents in which the index term appears. The motivation for using an inverse document frequency is that terms that appear in many documents are not very useful for distinguishing a relevant document from a non-relevant one. There are various scoring algorithms that use  $tf$  and  $(1/df)$ . These scoring algorithms include the normalization and the combination of each factor,  $tf$  and  $df$ . OKAPI is the one of well-known scoring algorithms (Robertson et al., 1994).

$$w_t = q_t \times tf \times \frac{\log\left(\frac{N-df+0.5}{df+0.5}\right)}{2 \times \left(0.25 + 0.75 \times \frac{doclength}{avg\ doclength}\right) + tf} \quad (1)$$

where  $w_t$  is the relevance weight assigned to a document due to query term  $t$ , and  $q_t$  is the weight attached to the term by the query.  $N$  is the total number of documents.

The Kullback-Leibler (KL) divergence algorithm is also a famous scoring algorithm (Manning & Schutze, 1999). Documents are ranked according to the negation of the divergence of the query's language model from the document's language model (Lafferty & Zhai, 2001). Given two probability mass functions  $p(x)$  and  $q(x)$ , the Kullback-Leibler divergence between  $p$  and  $q$ , denoted  $D(p||q)$ , is defined as follows.

$$D(p||q) = \sum_x p(x) \log \frac{p(x)}{q(x)} \quad (2)$$

Let  $\theta_Q$  be the language model for the query  $Q$  and  $\theta_D$  be the language model for a document  $D$ . Documents are ranked by  $D(\theta_Q||\theta_D)$  (Zhai & Lafferty, 2001).

$$D(\theta_Q||\theta_D) = - \sum_w p(w|\theta_Q) \log \frac{p(w|\theta_D)}{p(w|\theta_Q)} \quad (3)$$

## 2.2 Link Information

A hyperlink on the Web is a kind of citation. The essential idea is that if page  $u$  has a link to page  $v$ , then the author of  $u$  is implicitly assigning some importance to page  $v$ . We can represent the Web as a directed graph by representing each page as a node, and putting a directed edge between two pages if there is a link from one to the other (Kleinberg, 1999). This graph allows rapid calculation of a web document's '*PageRank*', an objective measure of its citation importance that corresponds well with people's subjective idea of importance. Since we can represent the Web as a graph, we can use graph theories to help us make a search engine that returns the most important pages first. The *PageRank* or  $PR(A)$  of a page  $A$  is given as follows (Brin & Page, 1998).

$$PR(A) = (1 - d) + d(PR(T_1)/C(T_1) + \dots + PR(T_n)/C(T_n)) \quad (4)$$

We assume page  $A$  has pages  $T_1 \dots T_n$  that point to it. The parameter  $d$  is a damping factor that can be set between 0 and 1. Also  $C(A)$  is defined as the number of links going out of a page  $A$ .  $PR(A)$  can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the Web (Page et al., 1998).

## 2.3 URL Information

The URL string of a site entry page often contains the name or acronym of the corresponding organization. Therefore, an obvious way of exploiting URL information is trying to match query terms and URL terms. We can add URLs to the text. Additionally, URLs of site entry pages tend to be higher in a server's directory tree than other web pages, i.e. the number of slashes ('/') in an entry page URL tends to be relatively small. Westerveld et al. (2001) suggested 4 types of URLs.

- root: a domain name  
(e.g. <http://trec.nist.gov>)
- subroot: a domain name followed by a single directory  
(e.g. <http://trec.nist.gov/pubs/>)
- path: a domain name followed by an arbitrarily deep path  
(e.g. <http://trec.nist.gov/pubs/trec9/papers>)
- file: anything ending in a filename other than 'index.html'  
(e.g. <http://trec.nist.gov/pubs/trec9/t9proceedings.html>)

Table 1. Distributions of Entry Pages and WT10g over URL Types

URL type	#entry pages	#WT10g
root	38 (71.7%)	11,680 (0.6%)
subroot	7 (13.2%)	37,959 (2.2%)
path	3 (5.7%)	83,734 (4.9%)
file	3 (5.7%)	1,557,719 (92.1%)

Westerveld et al. (2001) estimated a prior probability ( $URLprior$ ) of being an entry page on the basis of the URL type  $P(entrypage|URLtype = t)$  for all URL types  $t$ . Table 1 shows the distribution of each URL type.

#### 2.4 Combination of Multiple Evidences

We can combine results of each search engine or scores of each measure to get better results.

##### 2.4.1 Merge Scores

Researchers are interested in combining multiple evidences. Croft (2000) proposed the INQUERY retrieval system, based on the inference network, to combine multiple evidences. The inference network model is a general model for combining information. The model is based on probabilistic updating of the values of nodes in the network, and many retrieval techniques and information can be implemented by configuring the network properly.

Several researchers have experimented with linearly combining the normalized relevance scores ( $s_i$ ) given to each document (Westerveld et al., 2001).

$$score(d) = \sum_i \alpha_i s_i(d) \quad (5)$$

It requires training for the weight  $\alpha_i$  given to each input system. For example, we can get a better result by combining content information and URL type information with the following weight (Westerveld et al., 2001).

$$score(d) = 0.7 \times content + 0.3 \times URLprior \quad (6)$$

##### 2.4.2 Merge Results

Fox & Shaw (1994) tried fusing based on the unweighted *min*, *max*, *median*,

or *sum* of each document’s normalized relevance scores over the constituent systems. They also experimented with ways to more or less heavily weight a merged score. ‘*CombANZ* (Average of NonZeros)’ is the average similarity over systems. ‘*CombSum*’ is the summed similarity over systems. ‘*CombMNZ* (Multiply by NonZeros)’ is the multiplied similarity by the number of systems that return a given document. Lee (1997) showed that *CombSum* can be used to obtain improved results when combining two runs from the same retrieval system. If we use different parameters for the two runs or different query expansion techniques, then we can have different results. He also performed experiments with the other *Comb* algorithms, arguing that “different runs retrieve similar sets of relevant documents but retrieve different sets of non-relevant documents”. He further argues that the *CombMNZ* combination rule appropriately takes advantage of this feature of variant systems’ ranked lists by heavily weighting common documents. His experiments showed significant improvement for both *CombSum* and *CombMNZ*.

### 3 Topic Relevance Task and Homepage Finding Task

In this section, we present the properties of Content, Link, and URL Information.

#### 3.1 TREC

We use TREC data collection, to show the differences of each search task. The Text REtrieval Conference (TREC) was started in 1992 as part of the TIPSTER Text program. Its purpose is to support research within the information retrieval community by providing the infrastructure necessary for large-scale evaluation of text retrieval methodologies. The web track is a division of TREC. Its goal is to investigate retrieval behavior when the collection to be searched is a large hyperlinked structure such as the World Wide Web. The web track contained two tasks in 2001, the ‘topic relevance task’ and the ‘homepage finding task’ (Hawking & Craswell, 2001). The topic relevance task is a traditional ad hoc retrieval task. The goal is to find documents that meet the information need of a user. On the other hand, the goal of the homepage finding task is to find the homepage of the site described in the query. The query for the homepage finding task consists of a single phrase such as ‘*HKUST Computer Science Dept.*’. For each query, the system should retrieve the homepage of the entity described by the query. For the example query, the system should retrieve the homepage for the computer science department of the Hong Kong University of Science and Technology. The homepage of a related site of a different granularity, such as the homepage for the entire

university or a project within the computer science department, is counted as incorrect for this task.

The data for the web track are the 10-gigabyte WT10g collection (Bailey et al., to appear), distributed by CSIRO (CSIRO, 2001). We use four query sets. For the topic relevance task, TREC-2000 topic relevance task queries (topics 451-500) and TREC-2001 topic relevance task queries (topics 501-550) are used. We call them  $QUERY_{T-TRAIN}$  and  $QUERY_{T-TEST}$ . For the homepage finding task, queries for randomly selected 100 homepages<sup>1</sup> and 145 TREC-2001 homepage finding task queries are used. We call them  $QUERY_{H-TRAIN}$  and  $QUERY_{H-TEST}$ . We use  $QUERY_{T-TRAIN}$  and  $QUERY_{H-TRAIN}$  for training and  $QUERY_{T-TEST}$  and  $QUERY_{H-TEST}$  for testing.

Although there are many metrics used in IR, raw performances of a retrieval system are perhaps often measured with average precision and Mean Reciprocal Rank (Hawking & Craswell, 2001). The topic relevance task is measured with average precision. Since we are considering not sets of documents but ranked lists, we need a measure that takes the rank of each document into account. We first define precision at rank  $i$  to be the precision of the set of documents with rank  $i$  or better. For example, ‘pre@5’ means the precision with top 5 documents. Now, average precision is defined as the average of the precision obtained at the rank of each relevant document. The average precision of one system’s ranked list corresponding to one query is:

$$P_{avg} = \frac{1}{|R|} \sum_{d \in R} \frac{|R_{\leq r(d)}|}{r(d)} \quad (7)$$

where  $R$  is the set of all relevant documents and  $R_{\leq r(d)}$  is the set of relevant documents with rank  $r(d)$  or better. Usually researchers report the results of an experiment in terms of *mean average precision*, the mean of average precision obtained over each query in the experiment (Montague & Aslam, 2001). We call *mean average precision* itself simply average precision. Mean Reciprocal Rank (MRR) is the main evaluation measure for the homepage finding task. MRR is based on the rank of the first correct document (*answer<sub>i</sub> rank*) according to the following formula:

$$MRR = \frac{1}{\#queries} \sum_{i=1}^{\#queries} \frac{1}{answer_i \text{ rank}} \quad (8)$$

In addition to MRR, we use *%top10* and *%fail* for the homepage finding task. *%top10* is the proportion of queries for which a right answer is found in the

<sup>1</sup> available at <http://www.ted.cmis.csiro.au/TRECWeb/Qrels/>

top 10 results.  $\%fail$  is the proportion of queries in which no right answer is found in the top 100 results.

### 3.2 Differences of Search Tasks

In this section, we present the properties of each information. We tested each information for each query category.

We made a simple search engine that use the variation of OKAPI scoring function (Robertson et al., 1994). We changed the coefficient of Eq. 1.

$$w_t = q_t \times \left( 0.4 + 0.6 \times \frac{tf}{tf + 0.5 + 1.5 \times \frac{doclength}{avg\ doclength}} \right) \times \frac{\log\left(\frac{N+0.5}{df_t}\right)}{\log(N+1)} \quad (9)$$

We used the anchor text representation (*Anchor*) and the common content text representation (*Common*) for indexing. Every document in the anchor text representation has anchor texts and the title as content, and excludes a body text. Consequently the anchor text representation has brief or main explanations of a document.

We used two other types of information for a scoring function besides the OKAPI score. One is *URLprior* for URL information and the other is *PageRank* for Link Information. We linearly interpolated content information, *URLprior*, and *PageRank*. We call this interpolation as ‘*CMB*’.

$$rel(d) = 0.65 \times Content\ Information + \quad (10) \\ 0.25 \times URL\ Information + 0.1 \times Link\ Information$$

We used ‘*and*’ and ‘*sum*’ operators for matching query terms (Baeza-Yates & Ribeiro-Neto, 1999). ‘*and*’ operator means that a result document has all query terms in it. ‘*sum*’ operator means that a result document has at least one query term in it.

Table 2 shows the average precision of the topic relevance task and the MRR of the homepage finding task. For example, ‘*Anchor and CMB*’ implies that we used the anchor text representation for indexing, ‘*and*’ operator for query matching, and the OKAPI score, *PageRank* and *URLprior* for scoring. *MAX* represents the best score of a search engine that submitted in TREC-2001. *AVG* represents the average score of all search engines that submitted in TREC-2001.

Table 2. Topic Relevance Task vs. Homepage Finding Task

	Topic Relevance Task			Homepage Finding Task		
model	avg Pre.	prec@5	prec@10	MRR	%top10	%fail
<i>Anchor and</i>	0.0305	0.0519	0.0370	0.297	47.6	42.8
<i>Anchor and CMB</i>	0.0305	0.0519	0.0370	0.431	53.8	41.4
<i>Anchor sum</i>	0.0343	0.1388	0.1143	0.351	60.0	23.4
<i>Anchor sum CMB</i>	0.0338	0.1388	0.1143	0.583	70.3	19.3
<i>Common and</i>	0.1312	0.3208	0.2833	0.294	51.7	28.3
<i>Common and CMB</i>	0.1220	0.3125	0.2729	0.580	67.6	24.1
<i>Common sum</i>	<b>0.1819</b>	<b>0.3510</b>	<b>0.3265</b>	0.355	62.1	15.9
<i>Common sum CMB</i>	0.1685	0.3347	0.3163	<b>0.673</b>	<b>79.3</b>	<b>11.0</b>
MAX	0.226	0.432	0.362	0.774	88.3	4.8
AVG	0.145	0.328	0.256	0.432	58.5	22.7

We got the better result with the common content text representation than the anchor text representation in the topic relevance task. A title and anchor texts do not have enough information for the topic relevance task. On the other hand, we could get the similar performance with the anchor text representation in the homepage finding task.

URL information and Link information are good for the homepage finding task but bad for the topic relevance task. In the topic relevance task, we lost our performance by combining URL and Link information.

The query of the topic relevance task usually consists of main keywords that are relevant to some concept or the explanation of what they want to know. However, we cannot assume that other people use same expressions and keywords to explain what a user wants to know. Therefore, we could not get a good result with ‘*and*’ operator in the topic relevance task. But on the other hand the query of the homepage finding task consists of entity names or proper nouns. Therefore we could have good results with ‘*and*’ operator when we can have a result document. However, the MRR of ‘*Anchor and CMB*’ is lower than that of ‘*Common sum CMB*’ in the homepage finding task. ‘*Anchor and CMB*’ method did not retrieve a document for 31 queries. To compensate for this sparseness problem, we combined the results of ‘*Anchor and CMB*’ and ‘*Common sum CMB*’ with *CombSUM* method. This combined result showed 0.730 MRR in the homepage finding task. When we combined the results of ‘*Anchor and*’ and ‘*Common sum*’, it showed 0.173 average precision in the topic relevance task. This implies that the result documents with ‘*and*’ operator are good and useful in the homepage finding task.

We can conclude that we need different retrieval strategies according to the category of a query. We have to use the field information (title, body, and anchor text) of each term, and combine multiple evidences dynamically to get good results. In the topic relevance task, the body text of a document is good for indexing, ‘*sum*’ operator is good for query term matching and combining URL and Link information are useless. On the other hand, in the homepage finding task, anchor texts and titles are useful for indexing, ‘*and*’ operator is also good for query term matching, and URL and Link information are useful. By combining results from main body text and anchor texts and titles, we can have the better performance.

## 4 User Query Classification

### 4.1 Preparation for Language Model

We may use the question type of a query to classify the category of a user query. For example, “*What is a two electrode vacuum tube?*” is a query of the topic relevance task. “*Where is the site of SONY?*” is a query of the homepage finding task. We can assume the category of a query with an interrogative pronoun and cue expressions (e.g. ‘the site of’). However, people do not provide natural language queries to a search engine. They usually use main keywords for their queries. It is not easy to anticipate natural language queries. In this paper, we assume that users provide only main keywords for their queries. We define a query  $Q$  as the set of words.

$$Q = \{w_1, w_2, \dots, w_n\} \tag{11}$$

We use four measures to determine the class of a given query. Four measures are as follows.

- Distribution of terms in a query
- Mutual Information of each term in a query
- The usage rate of each term as anchor texts
- The POS information

We divided WT10g into two sets,  $DB_{TOPIC}$  and  $DB_{HOME}$ . If the URL type of a document is ‘root’ type, we put this document to  $DB_{HOME}$ . Others are added to  $DB_{TOPIC}$ . According to the report of Westerveld et al. (2001), our division method can get site entry pages with 71.7% precision. Additionally we put virtual documents into  $DB_{HOME}$  with anchor texts. If a linked document is in  $DB_{TOPIC}$ , then we make a virtual document that consists of anchor texts

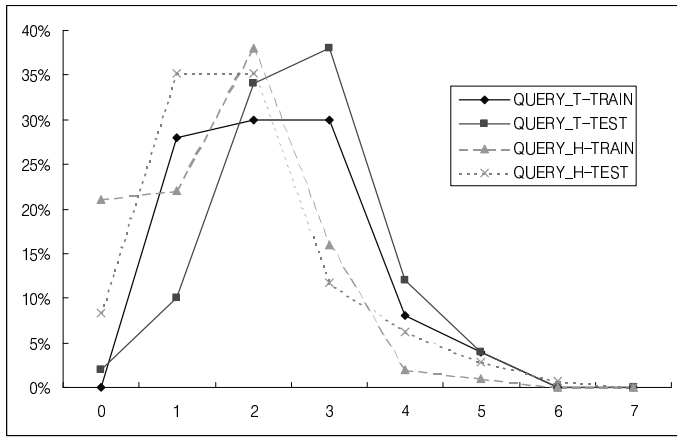


Fig. 1. The Number of Terms in Queries

and put it into  $DB_{HOME}$ . If a linked document is in  $DB_{HOME}$ , then we add anchor texts to the original document. Usually a site entry page does not have many words. It is not an explanatory document for some topic or concept, but the brief explanation of a site. We can assume that site entry pages have the different usage of words. If we find distinctive features for site entry pages, then we can discriminate the category of a given query.

$\#DB_{TOPIC}$  and  $\#DB_{HOME}$  mean the number of documents in the  $DB_{TOPIC}$  and  $DB_{HOME}$  respectively. However, most documents in the  $DB_{HOME}$  have a short length, we normalized the number of documents with the following equations.

$$\#DB_{TOPIC} = \# \text{ of documents in } DB_{TOPIC} \quad (12)$$

$$\begin{aligned} \#DB_{HOME} &= \# \text{ of documents in } DB_{HOME} \quad (13) \\ &\times \frac{\text{avg doclength}_{HOME}}{\text{avg doclength}_{TOPIC}} \end{aligned}$$

Fig. 1 shows the number of terms in topic relevance task queries and homepage finding task queries. Stop words are excluded from queries. If all terms are stopwords, then the number of terms is zero. It shows that there is no significant difference between the numbers of terms in each query set. This implies that we cannot distinguish a query type by the number of query terms.

#### 4.2 Distribution of Query Terms

‘*Earthquake*’ occurs more frequently in  $DB_{TOPIC}$ . But ‘*Hunt Memorial Library*’ shows the high relative frequency in  $DB_{HOME}$ . General terms tend to have same distribution regardless of the database. If the difference of distribution is larger than expected, this tells whether a given query is in the topic

relevance task class or the homepage finding task class. We can calculate the occurrence ratio of a query with the following equation (Manning & Schutze, 1999).

$$Dist(w_1, \dots, w_n) = \frac{n \times C(w_1, \dots, w_n)}{\sum_{i=1}^n C(w_i)} \quad (14)$$

$C(w)$  is the number of documents that have  $w$  as an index term.  $df$  of  $w$  is used for  $C(w)$ .  $C(w_1, \dots, w_n)$  is the number of documents that have all  $w_1, \dots, w_n$  as index terms. To see the distribution difference of a query, we use the following ratio equation.

$$diff_{Dist}(Q) = \frac{Dist_{HOME}(Q)}{Dist_{TOPIC}(Q)} \quad (15)$$

If a query has only one term, we use the chi-square (Manning & Schutze, 1999). We make a 2-by-2 table for the given word ‘ $w$ ’.

	word= $w$	word $\neq w$
$DB_{TOPIC}$	$a$	$b$
$DB_{HOME}$	$c$	$d$

$a + b = \#DB_{TOPIC}$  and  $c + d = \#DB_{HOME}$ . ‘ $a$ ’ is the frequency of the word ‘ $w$ ’ in the  $DB_{TOPIC}$  and ‘ $c$ ’ is the frequency of the word ‘ $w$ ’ in the  $DB_{HOME}$ . The chi-square value shows the dependence of the word ‘ $w$ ’ and DB. If the chi-square value of the word ‘ $w$ ’ is high, then ‘ $w$ ’ is a special term of  $DB_{TOPIC}$  or  $DB_{HOME}$ . We classify these words that have a high chi-square value according to the  $df$ . If ‘ $w$ ’ has a high  $df$  then the word ‘ $w$ ’ is the topic relevance task query. Otherwise ‘ $w$ ’ is the homepage finding task query. For example, ‘*fast*’ shows the high chi-square value, since it is used a lot to modify proper names. However, one word ‘*fast*’ is not the proper name. We classify a word that has a high chi-square and a high  $df$  into the topic relevance task. If the chi-square value of the word ‘ $w$ ’ is low, then ‘ $w$ ’ is a general term.

Fig. 2 shows the results of  $diff_{Dist}$  of queries that have two or more terms. The mean values of  $QUERY_{T-TRAIN}$ ’s  $diff_{Dist}$  and  $QUERY_{H-TRAIN}$ ’s  $diff_{Dist}$  are 0.5138 and 1.1 respectively. As the value of  $diff_{Dist}$  of a given query is higher, we can have confidence that the query has special terms. On the other hand, if the score of  $diff_{Dist}$  is near the mean value of  $QUERY_{T-TRAIN}$ , it means the query has general terms, not a special expression. However, there are queries that show high  $diff_{Dist}$  in  $QUERY_{T-TRAIN}$ . For example, ‘*Jennifer Aniston*’ and ‘*Chevrolet Trucks*’ showed 2.04 and 0.76 respectively. Usually

proper names showed a high positive value. If a proper name is frequently used in the  $DB_{HOME}$ , then we can think of it as the name of the site.

We calculate the possibility,  $DIST-INFO$ , that a given query is in each class with the mean value and the standard deviation.  $DIST-INFO$  is between -1 and 1. Negative  $DIST-INFO$  implies that a given query is the topic relevance task and positive  $DIST-INFO$  implies that a given query is the homepage finding task.

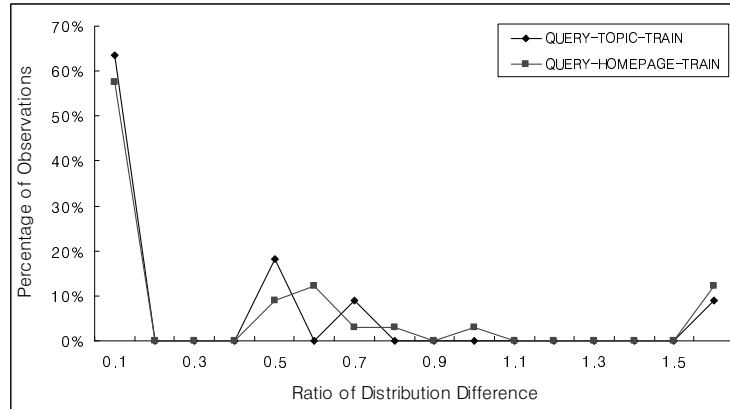


Fig. 2. Distribution Difference of Queries

```

if length(Q)=1 then
  calculate the  $\chi^2$  of Q
  if  $\chi^2 > 18$  then
    if df of a query > 65
      the topic relevance task
    else
      the homepage finding task
  else
    the topic relevance task
else
  calculate distributions of a query in each database
  calculate  $DIST-INFO$ 
  if  $DIST-INFO > \alpha$ 
    the homepage finding task
  else if  $DIST-INFO < \beta$ 
    the topic relevance task
  else
    unknown

```

Fig. 3. The Algorithm of Distribution Difference Method

### 4.3 Mutual Information

There are two or more words that co-occur frequently. These words may have syntactic or semantic relations to each other. We say these words have some dependency. For example, ‘*tornadoes formed*’ shows similar dependency regardless of the database. But ‘*Fan Club*’ has a high dependency in  $DB_{HOME}$  set. This means that ‘*tornadoes formed*’ is a general usage of words but ‘*Fan Club*’ is a special usage in  $DB_{HOME}$ . Therefore, the dependency of ‘*Fan Club*’ can be the key clue of guessing the category of a user query. If the difference of dependency of each term is larger than expected, this tells whether a given query is the topic relevance task or the homepage finding task. For two variables  $A$  and  $B$ , we can calculate the dependency with mutual information,  $I(A; B)$  (Jaynes, 1957). We use the pointwise mutual information  $I(x, y)$  to calculate the dependency of terms in a query (Manning & Schutze, 1999).

$$\begin{aligned} I(A; B) &= H(A) + H(B) - H(A, B) \\ &= \sum_{a,b} p(a, b) \log \frac{p(a, b)}{p(a)p(b)} \end{aligned} \quad (16)$$

$$I(x, y) = \log \frac{p(x, y)}{p(x)p(y)} \quad (17)$$

We extend pointwise mutual information for three variables. We use the set theory to calculate the value of an intersection part, like two variables problem.

$$\begin{aligned} I(A; B; C) &= H(A, B, C) - H(A) - H(B) - H(C) + \\ &\quad I(A; B) + I(B; C) + I(C; A) \\ &= \sum_{a,b,c} p(a, b, c) \log \frac{p(a, b)p(b, c)p(c, a)}{p(a, b, c)p(a)p(b)p(c)} \end{aligned} \quad (18)$$

$$I(x, y, z) = \log \frac{p(x, y)p(y, z)p(z, x)}{p(x, y, z)p(x)p(y)p(z)} \quad (19)$$

In principle,  $p(x, y)$  means the probability that  $x$  and  $y$  are co-occurred in a specific distance (Manning & Schutze, 1999). Usually  $x$  and  $y$  are consecutive words. Since the number of words and documents are so huge in IR domain, it is not easy to keep statistics. Our measure assume that  $x$  and  $y$  are co-occurred in a document. We use  $df$  of a given term to calculate the number

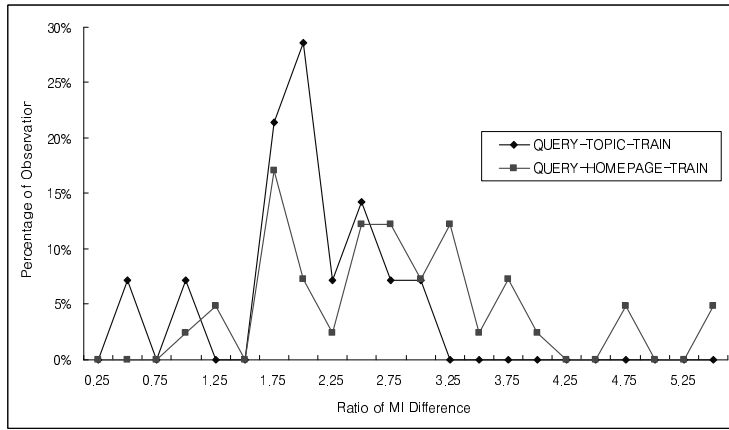


Fig. 4. Mutual Information of Queries

of documents that contain a term. Like the distribution difference measure, we use the ratio difference equation to see the difference of MI. If pointwise mutual information is below zero then we use zero.

$$diff_{MI}(Q) = \frac{MI_{HOME}(Q)}{MI_{TOPIC}(Q)} \quad (20)$$

Fig. 4 shows the results of  $diff_{MI}$ . The mean values of  $QUERY_{T-TRAIN}$ 's  $diff_{MI}$  and  $QUERY_{H-TRAIN}$ 's  $diff_{MI}$  are 1.9 and 2.7 respectively. For example, the topic relevance task query 'mexican food culture' showed 1.0, but the homepage finding task query 'Newave IFMO' showed 7.5.  $QUERY_{H-TRAIN}$  gets a slightly high standard deviation. It means that the query of  $QUERY_{H-TRAIN}$  has different MI in  $DB_{HOME}$ . As the value of  $diff_{MI}$  is higher, we can have confidence that the query has a special dependency. We calculate the possibility,  $MI-INFO$ , that a given query is in each class with the mean value and the standard deviation.  $MI-INFO$  is between -1 and 1. Like  $DIST-INFO$ , negative  $MI-INFO$  implies that a given query is the topic relevance task and positive  $MI-INFO$  implies that a given query is the homepage finding task.

#### 4.4 Usage Rate as an Anchor Text

If query terms appear in titles and anchor texts frequently, this tells the category of a given query is the homepage finding task. Titles and anchor texts are usually entity names or proper nouns, the usage rate shows the probability that given terms are special terms.

$$use_{Anchor}(w_1, \dots, w_n) = \quad (21)$$

$$\frac{C_{SITE\_ANCHOR}(w_1, \dots, w_n) - C_{SITE}(w_1, \dots, w_n)}{C_{SITE}(w_1, w_2, \dots, w_n)}$$

$C_{SITE}(w)$  means the number of site entry documents ('root' type pages) that have  $w$  as an index term.  $C_{SITE\_ANCHOR}(w)$  means the number of documents that have  $w$  as an index term in  $DB_{HOME}$ . We calculate the possibility, *ANCHOR-INFO*, that a given query is the homepage finding task. *ANCHOR-INFO* is between -1 and 1.

#### 4.5 POS information

Since the homepage finding task queries are proper names, they do not usually contain a verb. However, some topic relevance task queries include a verb to explain what he or she wants to know. For example, "*How are tornadoes formed?*" or briefly '*tornadoes formed*' contain a verb '*formed*'. If a query has a verb except the '*be*' verb, then we classified it into the topic relevance task. We calculate the possibility, *POS-INFO*, that a given query is the topic relevance task. *POS-INFO* is -1 or 0.

#### 4.6 Combination of Measures

The difference of distribution method can be applied to more queries than the difference of MI. The usage rate as anchor texts and the POS information can be applied to small number of queries. However, four measures cover different queries. Therefore, we can have more confidence and more coverage by combining these measures. We use a different combination equation as the number of query terms. If the query has 2 and 3 terms in it, we use pointwise mutual information also.

$$S(Q) = \alpha_1 \times DIST-INFO + \alpha_2 \times MI-INFO + \alpha_3 \times ANCHOR-INFO + \alpha_4 \times POS-INFO \quad (22)$$

If ' $S(Q)$ ' score is bigger than threshold,  $\gamma (> 0)$ , then a given query is classified as the homepage finding task. If ' $S(Q)$ ' score is smaller than threshold,  $\delta (< 0)$ , then a given query is classified as the topic relevance task. If ' $S(Q)$ ' is between  $\delta$  and  $\gamma$ , then we do not classify a query type. We choose  $\alpha_i$ ,  $\gamma$ , and  $\delta$  with train data ( $QUERY_{T-TRAIN}$  and  $QUERY_{H-TRAIN}$ ).

## 5 Experiments

In this section, we show the efficiency of a user query classification.

### 5.1 Query Classification

We used four query sets for experimenting our query classification method.  $QUERY_{T-TRAIN}$  and  $QUERY_{H-TRAIN}$  are used for training (TRAIN). TREC-2001 topic relevance task queries,  $QUERY_{T-TEST}$ , and TREC-2001 homepage finding task queries,  $QUERY_{H-TEST}$ , are used for testing (TEST). We used WT10g for making a classification model. We classified queries with our proposed method. Table 3 shows the classification result of our proposed language model. Precision and recall are calculated with the following equations.

$$Prec = \frac{\#correct\ classification}{\#total\ trial} \quad (23)$$

$$Recall = \frac{\#correct\ classification}{\#queries} \quad (24)$$

‘All’ in the table 3 implies that we combine all measures for the classification. By combining each measure, we could apply our method to more queries and increase the precision and the recall. Our proposed method shows the better result in the test set. This is due to the characteristics of the query set. There are 7 queries that have a verb in  $QUERY_{T-TRAIN}$  and 28 queries in  $QUERY_{T-TEST}$ . We can assume that the POS information is good information. Our proposed method showed good performance on finding the homepage finding task, but low performance on finding the topic relevance task. We need more useful measures for the topic relevance task.

Table 3. Query Classification Result

QUERY	$QUERY_{TRAIN}$				$QUERY_{TEST}$			
	TOPIC		HOMEPAGE		TOPIC		HOMEPAGE	
Measure	Prec.	Recall	Prec.	Recall	Prec.	Recall	Prec.	Recall
<i>DIST</i>	40.9%	18.0%	92.5%	49.0%	30.8%	8.0%	94.4%	35.2%
<i>MI</i>	16.7%	2.0%	100%	28.0%	0.0%	0.0%	100%	30.3%
<i>ANCHOR</i>	34.8%	16.0%	91.8%	45.0%	15.4%	4.0%	94.4%	46.9%
<i>POS</i>	100%	14.0%	0.0%	0.0%	100%	56.0%	0.0%	0.0%
<i>All</i>	52.9%	36.0%	94.4%	67.0%	77.8%	56.0%	96.8%	63.4%

The main reason of misclassification is wrong division of WT10g. Since our method usually gives the high score to the proper name, we need correct information to distinguish a proper name from a site name. We tried to make  $DB_{HOME}$  automatically. However, some root pages are not site entry pages. We need a more sophisticated division method.

There is a case that a verb is in the homepage finding task query. ‘*Protect & Preserve*’ is the homepage finding task query but ‘*protect*’ and ‘*preserve*’ are verbs. However, ‘*Protect*’ and ‘*Preserve*’ start with a capital letter. We can correct wrong POS tags.

There are queries in  $QUERY_{T-TEST}$  that look like queries of  $QUERY_{H-TEST}$ . For example, ‘*Dodge Recalls*’ is used to find documents that report on the recall of any dodge automobile products. But user may want to find the entry page of ‘*Dodge Recall*’. This is due to the use of main keywords instead of a natural language query.

There are 6 queries in  $QUERY_{T-TEST}$  and 6 queries in  $QUERY_{H-TEST}$  that do not have a result document that has all query terms in it. We could not use our method to them. WT10g is not enough to extract probability information for these two query sets. To make up this sparseness problem, we need a different indexing terms extraction module. We have to consider special parsing technique for URL strings and acronyms in a document. Also we need a query expansion technique to get a better result.

## 5.2 The IR Performance with the Classification

We used the *Lemur* Toolkit (Ogilvie & Callan, 2001) to make a general search engine for the topic relevance task. The *Lemur* Toolkit is an information retrieval toolkit designed with language modeling in mind. The *Lemur* Toolkit supports several retrieval algorithms. These algorithms include a dot-product function using TF-IDF weighting algorithm, the *Kullback-Leibler (KL)* divergence algorithm, the *OKAPI* retrieval algorithm, the feedback retrieval algorithm and the mixture model of Dirichlet smoothing, *MIXFB-KL-D* (Ponte, 2000). For the homepage finding task, we add the *URLprior* probability of a URL string to the *Lemur* Toolkit. Besides Link information, we add the *PageRank* of a document. We normalized *PageRank* values, so the max value is 100 and the min value is 0. First we extracted top 1,000 results with the *Lemur* Toolkit. Then we combined URL information and Link information to reorder results with the equation Eq. 10. We presented top 1,000 documents as the answer in the topic relevance task, and 100 documents in the homepage finding task. We call this modified Toolkit as *MLemur* Toolkit.

Table 4 and 5 show results of the topic relevance task and the homepage finding

Table 4. Average Precision of the Topic Relevance Task

model	OKAPI	TF-IDF	KL_DIR	MIXFB_KL_D
<i>Lemur</i>	0.182	0.170	0.210	0.219
<i>MLemur</i>	0.169	0.159	0.200	0.209

Table 5. MRR of the Homepage Finding Task

model	OKAPI	TF-IDF	KL_DIR	MIXFB_KL_DIR
<i>Lemur</i>	0.355	0.340	0.181	0.144
<i>MLemur</i>	0.673	0.640	0.447	0.360

task that use the *Lemur* Toolkit and the *MLemur* Toolkit. *MIXFB\_KL\_D* showed the good result in the topic relevance task but showed the poor result in the homepage finding task. We can say that a good information retrieval algorithm for the topic relevance task is not always good for the homepage finding task. We chose three algorithms, the *OKAPI*, the *TF-IDF*, and the *MIXFB\_KL\_D* that got the best and worst score in each task, for the test of performance improvement by query type classification.

Table 6 shows the change of performance. ‘DEF’ means the default category for an unclassified query. Digits in the ‘TOPIC’ column and the ‘HOME’ column are average precision and MRR respectively. Because the precision of MI measure is 1.0 and the precision of POS measure is 0.0 in finding the homepage finding task, ‘DEFAULT’ *HOME* method showed the best performance in the homepage finding task. Since the POS information is too effective in the test set, we did not have much improvement by combining measures. From the result, the *OKAPI* algorithm and the homepage finding task as a default class (*‘home’*) method showed the good performance.

### 5.3 Discussion

To classify a query type, we need the document frequency of a query term in each database. This lowers the system efficiency. However, we can classify a user query and retrieve result documents at the same time. First, we create two databases (*Common* and *Anchor*) as proposed in this paper for indexing. In addition, web pages that are the root type are inserted into *Anchor*. We treat *Common* and *Anchor* as  $DB_{TOPIC}$  and  $DB_{HOME}$ . An index term of *Anchor* has a field information that indicates whether it is from a main body, a title, or an anchor text. With this field information, we can obtain frequencies for the classification. For *Common*, we used ‘*sum*’ operator and for *Anchor*, we used ‘*and*’ operator and ‘*CMB*’ method. We retrieve two result document sets

Table 6. The Retrieval Performance with Classification Method

		OKAPI		TF-IDF		MIXFB_KL_D	
Measure	DEF	TOPIC	HOME	TOPIC	HOME	TOPIC	HOME
DIST	topic	0.178	0.469	0.168	0.447	0.216	0.226
DIST	home	0.174	0.666	0.164	0.633	0.212	0.359
MI	topic	0.179	0.465	0.168	0.445	0.218	0.233
MI	home	0.169	0.673	0.159	0.640	0.209	0.360
ANCHOR	topic	0.176	0.513	0.165	0.489	0.215	0.232
ANCHOR	home	0.169	0.666	0.159	0.633	0.209	0.359
POS	topic	0.182	0.355	0.170	0.340	0.219	0.144
POS	home	0.173	0.673	0.163	0.640	0.212	0.354
All	topic	0.180	0.552	0.168	0.528	0.217	0.280
All	home	0.173	0.666	0.163	0.633	0.212	0.353

from each database and classify a query type at the same time. And then according to the category of a query, we merge two results. If the type of a query is ‘the topic relevance task’, then we use the result of *Common*. If the type of a query is ‘the homepage finding task’, then we merge results of *Common* and *Anchor*. We emphasize the result of *Anchor* in merging results. Table 7 shows the performance of the merging two results. We could obtain better performance by combining each result.

Table 7. The Performance of Results Combination

DEF	TOPIC	HOME
topic	0.180	0.620
home	0.173	0.753

In this paper, we proposed a user query classification method for the topic relevance task and the homepage finding task. The queries of the homepage finding task usually consist of entity names or proper nouns. However queries of the service finding task have verbs for the service definition. For example, “*Where can I buy concert tickets?*” has ‘*buy*’ as the service definition. To find these cue expressions, we need more sophisticated analysis of anchor texts. Since the service in the Web is provided as a program, there is a trigger button. Mostly these trigger buttons are explained by anchor texts. We have to distinguish an entity name and an action verb from anchor texts. We have to change measures for the query classification from a word unit to entity and action units.

User query classification can be applied to various areas. MetaSearch is the search algorithm that combines results of each search engine to get the better result (Fox & Shaw, 1994). Lee (1997) proposed *CombMNZ*, Multiply by NonZeros, is better than other scoring algorithm, *CombSUM*, Summed similarity over systems. But if we consider MetaSearch for the homepage finding task, *CombMNZ* does not show good performance.

Table 8 and 9 show the improvement of performance of MetaSearch algorithms. We had an experiment with random samplings of 2, 3, 4, and 5 engine results. 97 topic relevance task results and 43 homepage finding task results that are submitted in TREC-2001 were used (Hawking & Craswell, 2001). The score is the average performance improvement of 100 tests. The improvement of performance is calculated with the following equation (Montague & Aslam, 2001).

$$I = \frac{P_f - P_b}{P_b} \quad (25)$$

$P_f$  is the raw performance of the MetaSearch algorithm and  $P_b$  is the raw performance of the best input system being merged. *CombMNZ* was good for the topic relevance task, but *CombSUM* was good for the homepage finding task. It also tells, we need different strategies for MetaSearch as the class of a query.

Table 8. Performance of MetaSearch in the Topic Relevance Task

engine #	2	3	4	5
CombSUM	-2.4%	4.4%	3.7%	4.8%
CombMNZ	-1.2%	5.7%	5.3%	5.8%

Table 9. Performance of Metasearch in the Homepage Finding Task

engine #	2	3	4	5
CombSUM	-4.5%	0.7%	-0.9%	0.8%
CombMNZ	-6.0%	-0.4%	-4.5%	-2.4%

## 6 Conclusions

We have various forms of resources in the Web, and consequently purposes of user queries are diverse. We can classify user queries as three categories, the topic relevance task, the homepage finding task, and the service finding task. Search engines need different strategies to meet the purpose of a user query. For example, URL information and Link information are bad for the

topic relevance task, but on the other hand, they are good for the homepage finding task. We made two representative databases,  $DB_{HOME}$  and  $DB_{TOPIC}$ , for each task. To make databases, we divided text collection by the URL type of a web document. If the URL of a document contains a host name only, then we put it into  $DB_{HOME}$ . Also we make a virtual document with an anchor text and put it into  $DB_{HOME}$ . Other documents are put into  $DB_{TOPIC}$ . If given query's distributions in  $DB_{HOME}$  and  $DB_{TOPIC}$  are different, then this tells a given query is not a general word. Therefore, we can assume the category of a given query is in the homepage finding task. Likewise, the difference of dependency, Mutual Information, the usage rate as anchor texts, and the usage of a verb tell whether a given query is in the homepage finding task or not. We tested the proposed classification method with two query sets,  $QUERY_{T-TEST}$  and  $QUERY_{H-TEST}$ . The usage rate as anchor texts and the POS information show small coverage. On the other hand, distribution difference and dependency showed good precision and coverage. Also each classifier applied to different queries. We could get the better precision and recall by combining each classifier. After we classified the category of a query, we used different information for a search engine. For the topic relevance task, Content information such as TFIDF is used. For the homepage finding task, Link information and URL information besides content information are used. We tested our dynamic combining method. From the result, our classification method showed the best result with the OKAPI scoring algorithm.

## 7 Acknowledgments

We would like to thank Jamie Callan for providing useful experiment data and the Lemur toolkit.

## References

- Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. ACM Press.
- Bailey, P., Craswell, N., & Hawking, D. (to appear). Engineering a multi-purpose test collection for web retrieval experiments. *Information Processing and Management*.
- Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7), 107-117.
- Broder, A. (2002). A taxonomy of web search. *SIGIR Forum*, 36(2).
- Croft, W. B. (2000). Combining approaches to information retrieval. In W. B. Croft (Ed.), *Advances in information retrieval: recent research from*

- the center for intelligent information retrieval* (pp. 1–36). Kluwer Academic Publishers.
- CSIRO. (2001). Web research collections - trec web track. [www.ted.cmis.csiro.au/TRECWeb/](http://www.ted.cmis.csiro.au/TRECWeb/).
- Fox, E., & Shaw, J. (1994). Combination of multiple searches. In *Text REtrieval conference (trec-3)* (pp. 105–108). Gaithersburg, Maryland.
- Hawking, D., & Craswell, N. (2001). Overview of the trec-2001 web track. In *Text REtrieval conference (trec-10)* (pp. 61–67). Gaithersburg, Maryland.
- Jaynes, E. (1957). Information theory and statistical mechanics. *Physics Review*, 106(4), 620–630.
- Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5), 604–632.
- Lafferty, J., & Zhai, C. (2001). Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th annual international acm sigir conference on research and development in information retrieval* (p. 111-119). Gaithersburg, Maryland.
- Lee, J. H. (1997). Analyses of multiple evidence combination. In *Proceedings of the 20th annual international acm sigir conference on research and development in information retrieval* (pp. 267–276). Philadelphia.
- Manning, C. D., & Schütze, H. (1999). *Foundations of statistical natural language processing*. The MIT Press.
- Montague, M., & Aslam, J. (2001). Models for metasearch. In *Proceedings of the 24th annual international acm sigir conference on research and development in information retrieval* (p. 276-284). New Orleans, LA.
- Ogilvie, P., & Callan, J. (2001). Experiments using the lemur toolkit. In *Text REtrieval conference (trec-10)* <http://www-2.cs.cmu.edu/~lemur> (pp. 103–108). Gaithersburg, Maryland.
- Page, L., Brin, S., Motwani, R., & Winograd, T. (1998). *The pagerank citation ranking: Bringing order to the web* (Tech. Rep.). Stanford Digital Library Technologies Project.
- Ponte, J. M. (2000). Language models for relevance feedback. In W. B. Croft (Ed.), *Advances in information retrieval: recent research from the center for intelligent information retrieval* (pp. 73–95). Kluwer Academic Publishers.
- Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M., & Gatford, M. (1994). Okapi at trec-3. In *Text REtrieval conference (trec-2)* (pp. 109–126). Gaithersburg, Maryland.
- Westerveld, T., Kraaij, W., & Hiemstra, D. (2001). Retrieving web pages using content, links, urls and anchors. In *Text REtrieval conference (trec-10)* (p. 663-672). New Orleans, LA.
- Yang, K. (2001). Combining text and link-based retrieval methods for web ir. In *Text REtrieval conference (trec-10)* (p. 609-618). New Orleans, LA.
- Zhai, C., & Lafferty, J. (2001). A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th annual international acm sigir conference on research and development in information retrieval* (pp. 334–342). Gaithersburg, Maryland.