Query Type Classification for Web Document Retrieval

In-Ho Kang
Division of Computer Science
Department of EECS
KAIST
ihkang@csone.kaist.ac.kr

GilChang Kim
Division of Computer Science
Department of EECS
KAIST
gckim@cs.kaist.ac.kr

ABSTRACT

The heterogeneous Web exacerbates IR problems and short user queries make them worse. The contents of web documents are not enough to find good answer documents. Link information and URL information compensates for the insufficiencies of content information. However, static combination of multiple evidences may lower the retrieval performance. We need different strategies to find target documents according to a query type. We can classify user queries as three categories, the topic relevance task, the homepage finding task, and the service finding task. In this paper, a user query classification scheme is proposed. This scheme uses the difference of distribution, mutual information, the usage rate as anchor texts, and the POS information 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.

Categories and Subject Descriptors

H.4.m [**Information Systems Applications**]: Miscellaneous

General Terms

Experimentation

Keywords

Combination of Multiple Evidences, Link Information, URL Information, Query Classification

1. INTRODUCTION

The Web is rich with various sources of information. It contains the contents of documents, web directories, multimedia data, user profiles and 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 weakness that can hurt the retrieval performance in certain situations [5]. 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 a single evidence [17]. Fusion IR studies have repeatedly shown that combining multiple sources of evidence can improve retrieval performance [5][17].

However, previous studies did not consider a user query in combining evidences [5][7][10][17]. Not only documents in the Web but also users' 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 the higher rank. For user query *'Britney's Fan Club'*, if we use content information too highly, yahoo or lycos's web directory pages get the higher rank, instead of the 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. User queries can be classified as three categories according to their intent [4].

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

The topic relevance task is a traditional ad hoc retrieval task where web documents are ranked by decreasing likelihood of meeting the information need provided in a user query [8]. For example, *'What is a prime factor?'* or *'prime factor'* is a 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 a 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

SIGIR'03, July 28–August 1, 2003, Toronto, Canada

web documents that provide the service described in a user query. For example, *'Where can I buy concert tickets?'* or *'buy concert tickets'* is a query of the service finding task. The goal of this query is finding documents where they can buy concert tickets.

Users may want different documents 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 documents with this query. In this paper, we calculate the probability that the class of a user query is the topic relevance task or the homepage finding task. Based on this probability, 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.

In this paper, we present a user query classification method and a combining method for each query type. In section 2, we describe various types of information (Content, Link, and URL information). 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 INFORMATION

In this section, we explain various sources of information for the web document retrieval. There are three types of information, Content information, Link information, and URL information.

2.1 Content Information

There are multiple types of representations for a document. These representations typically contain titles, anchor texts, and main body texts [5]. A title provides the main idea and the brief explanation of a web document. An anchor text provides the description of linked web documents and files. An anchor text often provides more accurate description of a web document than the document itself.

We usually use tf and df to calculate the relevance of a given web documents [1]. 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 df. These scoring algorithms include the normalization and the combination of each factor, tf and df.

2.2 Link Information

A hyperlink in a web document 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. 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 [13].

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

We assume page A has pages $T_1 \ldots T_n$ 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 [3].

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. Additionally, URLs of site entry pages tend to be higher in a server's directory tree than other web documents, i.e. the number of slashes ('/') in an entry page URL tends to be relatively small. Kraaij et al. suggested 4 types of URLs [16].

- 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/t9proc.html)

Kraaij et al. estimated a prior probability ($URLprior$) of being an entry page on the basis of the URL type for all URL types t (root, subroot, path, and file).

2.4 Combination of Information

We can combine results of each search engine or scores of each measure to get better results. Croft proposed the INQUERY retrieval system, based on the inference network, to combine multiple evidences [5]. The inference network model is a general model for combining information. It is data-level fusion. 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 [7][10][16].

$$score(d) = \sum_i \alpha_i s_i(d) \quad (2)$$

It requires training for the weight α_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 [16].

$$score(d) = 0.7 \times content + 0.3 \times URLprior \quad (3)$$

3. TOPIC RELEVANCE TASK AND HOMEPAGE FINDING TASK

In this section, we show properties of Content information, Link information, and URL information in each search task. Besides, we will propose the method for linearly combining information for each task.

We use TREC data collection, to show the differences of each search task. We made a simple search engine that use the variation of the OKAPI scoring function [15]. Given a query Q, the scoring formula is:

$$score = \sum_{t \in (Q \cap D_d)} TF_{d,t} \times IDF_t \quad (4)$$

$$TF_{d,t} = 0.4 + 0.6 \times \frac{tf_{d,t}}{tf_{d,t} + 0.5 + 1.5 \times \frac{doclen_d}{avg\ doclen}} \quad (5)$$

$$IDF_t = log(\frac{N + 0.5}{df_t}) / log(N + 1) \quad (6)$$

N is the number of documents in the collection. $tf_{d,t}$ is the number of occurrences of an index term t in a document d, and df_t is the number of documents in which t occurs.

We use the data for the web track, the 10-gigabyte WT10g collection [2], distributed by CSIRO [6]. We use TREC-2001 topic relevance task queries (topics 501-550) for the topic relevance task, and 145 queries for the homepage finding task [8]. For the homepage finding task, NIST found a homepage within WT10g and then composed a query designed to locate it.

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 evidences 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 (OKAPI score), URLprior, and PageRank. We call this interpolation as *CMB*.

$$rel(d) = 0.65 \times Content\ Information + 0.25 \times URL\ Information + 0.1 \times Link\ Information \quad (7)$$

We used *'and'* and *'sum'* operators for matching query terms [1]. *'and'* operator means that the result document has all query terms in it. *'sum'* operator means that a result document has at least one query term in it.

Table 1 shows the average precision of the topic relevance task and the MRR of the homepage finding task [8]. The first column in the table 1 means the method that we used for indexing and scoring. For example, *'Anchor and CMB'* means that we used the anchor text representation for indexing, *'and'* operator for query matching, and the OKAPI score, PageRank and URLprior for scoring. The average precision is defined as the average of the precision obtained at the rank of each relevant document.

$$P_{avg} = \frac{1}{|R|} \sum_{d \in R} \frac{R_{\leq r(d)}}{r(d)} \quad (8)$$

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. MRR (Mean Reciprocal Rank) is the main evaluation measure for the homepage finding task. MRR is based on the rank of the first correct document ($answer_i\ rank$) according to the following formula:

$$MRR = \frac{1}{\#queries} \sum_{i=1}^{\#queries} \frac{1}{answer_i\ rank} \quad (9)$$

Table 1: Topic Relevance Task vs. Homepage Finding Task

	Topic	Homepage
model	P_{avg}	MRR
Anchor and	0.031	0.297
Anchor and CMB	0.031	0.431
Anchor sum	0.034	0.351
Anchor sum CMB	0.034	0.583
Common and	0.131	0.294
Common and CMB	0.122	0.580
Common sum	**0.182**	0.355
Common sum CMB	0.169	**0.673**
MAX	0.226	0.774
AVG	0.145	0.432

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.

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'*. This combined result showed 0.730 in the homepage finding task. When we combined the results of *'Anchor and'* and *'Common sum'*, it showed 0.173 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 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 is useful. By combining results from main body text and anchor texts and titles we can have the better performance.

4. USER QUERY CLASSIFICATION

In this section, we present the method for making a language model for a 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 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.

$$Q = \{w_1, w_2, \ldots, w_n\} \tag{10}$$

To see the characteristics of each query class, we use two query sets. For the topic relevance task, TREC-2000 topic relevance task queries (topics 451-500) are used. For the homepage finding task, queries for randomly selected 100 homepages [1] are used. We call them $QUERY_{T-TRAIN}$ and $QUERY_{H-TRAIN}$.

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 [16], 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 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 equation.

$$\#DB_{TOPIC} = \#\ of\ documents\ in\ DB_{TOPIC} \tag{11}$$

$$\#DB_{HOME} = \#\ of\ documents\ in\ DB_{HOME} \times \frac{avg\ doclength_{HOME}}{avg\ doclength_{TOPIC}} \tag{12}$$

[1] available at http://www.ted.cmis.csiro.au/TRECWeb/Qrels/

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 [11].

$$Dist(w_1, \ldots, w_n) = \frac{n \times C(w_1, \ldots, w_n)}{\sum_{i=1}^{n} C(w_i)} \tag{13}$$

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

$$diff_{Dist}(Q) = \frac{Dist_{HOME}(Q)}{Dist_{TOPIC}(Q)} \tag{14}$$

If a query has only one term, we use the chi-square [11]. 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 at least two query 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. We calculate the possibility that a given query is in each class with the mean value and the standard deviation. However, there are queries that show high $diff_{DIST}$ in $QUERY_{T-TRAIN}$. For example, 'Jenniffer Aniston' and 'Chevrolet Trucks' showed 2.04 and 0.76 respectively. Usually proper names showed high $diff_{DIST}$ values. If a proper name is frequently used in the DB_{HOME}, then we can think of it as the name of the site.

4.3 Mutual Information

There are two or more words that co-occur frequently. These words may have syntactic or semantic relations to

```
if length(Q)=1 then
    calculate the χ² of Q
    if χ² > 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 diffDist(Q)
    if diffDist(Q) > α
        the homepage finding task
    else
        unknown
```

Figure 1: The Algorithm of Distribution Difference Method

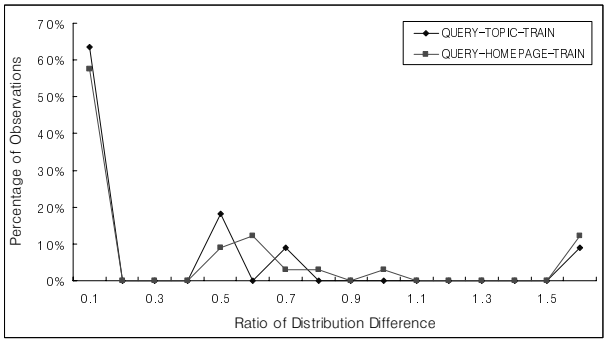

Figure 2: Distribution of Queries

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, $I(A;B)$ [9]. We use the pointwise mutual information $I(x,y)$ to calculate the dependency of terms in a query [11].

$$\begin{aligned} I(A;B) &= H(A) + H(B) - H(A,B) \\ &= \sum_{a,b} p(a,b) log \frac{p(a,b)}{p(a)p(b)} \end{aligned} \quad (15)$$

$$I(x,y) = log \frac{p(x,y)}{p(x)p(y)} \quad (16)$$

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.

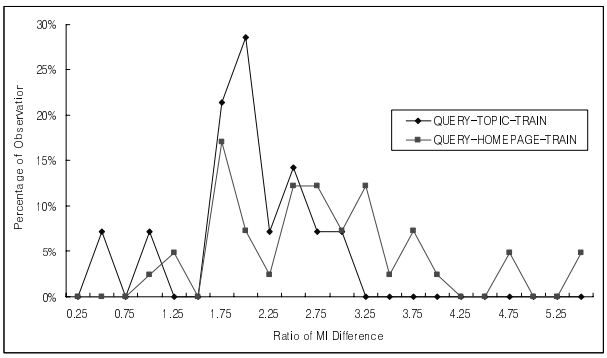

Figure 3: Mutual Information of Queries

$$\begin{aligned} I(A;B;C) &= H(A,B,C) - H(A) - H(B) - H(C) + \\ &\quad I(A;B) + I(B;C) + I(C;A) \\ &= \sum_{a,b,c} p(a,b,c) log \frac{p(a,b)p(b,c)p(c,a)}{p(a,b,c)p(a)p(b)p(c)} \end{aligned} \quad (17)$$

$$I(x,y,z) = log \frac{p(x,y)p(y,z)p(z,x)}{p(x,y,z)p(x)p(y)p(z)} \quad (18)$$

In principle, $p(x,y)$ means the probability that x and y are co-occurred in a specific distance [11]. 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 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.

$$diff_{MI}(Q) = \frac{MI_{HOME}(Q)}{MI_{TOPIC}(Q)} \quad (19)$$

Fig.3 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 that a given query is in each class with the mean value and the standard deviation.

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.

$$use_{Anchor}(w_1,\ldots,w_n) = \frac{C_{SITE\_ANCHOR}(w_1,\ldots,w_n) - C_{SITE}(w_1,\ldots,w_n)}{C_{SITE}(w_1,w_2,\ldots,w_n)} \quad (20)$$

$C_{SITE}(w)$ means the number of site entry documents that have w as an index term. $C_{SITE\_ANCHOR}(w)$ means the number of site entry documents and anchor texts that have w as an index term.

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.

4.6 Combination of Measures

The difference of distribution method can apply more queries than the difference of MI. The usage rate as anchor texts and the POS information show small coverage. 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.

$$S(Q) = \alpha \times diff_{Dist}(Q) + \beta \times diff_{MI}(Q) + \gamma \times use_{Anchor}(Q) + \delta \times POS_{info}(Q) \quad (21)$$

We choose α, β, γ, and δ with train data ($QUERY_{T-TRAIN}$ and $QUERY_{H-TRAIN}$). If 'S(Q)' score is not high or low enough, then we make no decision.

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 (Topic 501-550) and TREC-2001 homepage finding task queries (1-145) are used for testing (TEST). We call two test sets as $QUERY_{T-TEST}$ and $QUERY_{H-TEST}$. We used WT10g for making a classification model.

We classified queries with our proposed method. If the score 'S(Q)' is high enough to tell that a given query is in the topic relevance task or the homepage finding task query, then we assigned the query type to it. For other cases, we did not classify a query category. Table 2 shows the classification result of our proposed language model.

Table 2: Query Classification Result

QUERY	TRAIN		TEST	
Measure	Precision	Recall	Precision	Recall
Dist.	77.3%	38.7%	82.1%	28.2%
MI	90.9%	20.0%	78.2%	29.9%
Anchor	73.6%	35.3%	82.4%	35.9%
POS	100%	9.3%	96.4%	13.8%
All	81.1%	57.3%	91.7%	61.5%

By combining each measure, we could apply our method to more queries and increase precision and 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.

Table 3: Average Precision of the Topic Relevance Task

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

Table 4: MRR of the Homepage Finding Task

model	OKAPI	TF-IDF	KL_DIR	MIXFB_KL_D
Lemur	0.355	0.340	0.181	0.144
MLemur	0.673	0.640	0.447	0.360

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 Improvement of IR Performance

We used the *Lemur* Toolkit [12] 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* [14]. 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.

Table 5: The Retrieval Performance with Classification Method

		OKAPI		TF-IDF		MIXFB_KL_D	
Measure	DEFAULT	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

Then we combined URL information and Link information to reorder results with the equation Eq. 7. 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 3 and 4 show results of the topic relevance task and the homepage finding 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 5 shows the change of performance. 'DEFAULT' means the default category for an unclassified query. Digits in the TOPIC column and the HOME column are average precision and MRR respectively. From the result, the OKAPI algorithm and the homepage finding task as a default class method shows 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 may create two databases as proposed in this paper for indexing. We retrieve two result document sets from each database and classify a query type at the same time. And then according to the category of a query, merge two results. From table 1, merging the results of the anchor text representation and the common content representation shows good performance. We need more work to unify the query classification work and the document retrieval.

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 [7]. [10] proposed *CombMNZ*, Multiply by NonZeros, is better than other scoring algorithm, *CombSUM*, Summed similarity over systems. But if we consider the homepage finding task, we are in a different situation.

Table 6 and 7 show the improvement of performance of MetaSearch algorithms. We had an experiment with random samplings of 2, 3, 4, and 5 engine results. The score is the average improvement of 100 tests. *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 6: 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 7: 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, and the usage rate as anchor texts 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. We got 91.7% precision and 61.5% recall. 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.

8. REFERENCES

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM PRESS BOOKS, 1999.

[2] P. Bailey, N. Craswell, and D. Hawking. Engineering a multi-purpose test collection for web retrieval experiments. *Information Processing and Management*, to appear.

[3] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[4] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2), 2002.

[5] W. B. Croft. Combining approaches to information retrieval. In *Advances in Information Retrieval: Recent Research from the Center for Intelligent Information Retrieval*, pages 1–36. Kluwer Academic Publishers, 2000.

[6] CSIRO. Web research collections - trec web track. *www.ted.cmis.csiro.au /TRECWeb/*, 2001.

[7] E. Fox and J. Shaw. Combination of multiple searches. In *Text REtrieval Conference (TREC-1)*, pages 243–252, 1993.

[8] D. Hawking and N. Craswell. Overview of the trec-2001 web track. In *Text REtrieval Conference (TREC-10)*, pages 61–67, 2001.

[9] E. Jaynes. Information theory and statistical mechanics. *Physics Review*, 106(4):620–630, 1957.

[10] J. H. Lee. Analyses of multiple evidence combination. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 267–276, 1997.

[11] C. D. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 1999.

[12] P. Ogilvie and J. Callan. Experiments using the lemur toolkit. In *Text REtrieval Conference (TREC-10) http://www-2.cs.cmu.edu/ lemur*, pages 103–108, 2001.

[13] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[14] J. M. Ponte. Language models for relevance feedback. In W. B. Croft, editor, *Advances in Information Retrieval: Recent Research from the Center for Intelligent Information Retrieval*, pages 73–95. Kluwer Academic Publishers, 2000.

[15] S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *Text REtrieval Conference (TREC-2)*, pages 109–126, 1994.

[16] T. Westerveld, W. Kraaij, and D. Hiemstra. Retrieving web pages using content, links, urls and anchors. In *Text REtrieval Conference (TREC-10)*, pages 663–672, 2001.

[17] K. Yang. Combining text and link-based retrieval methods for web ir. In *Text REtrieval Conference (TREC-10)*, pages 609–618, 2001.