A Term Dependency-Based Approach for Query Terms Ranking

Chia-Jung Lee, Ruey-Cheng Chen, Shao-Hang Kao and Pu-Jen Cheng
Department of Computer Science and Information Engineering
National Taiwan University, Taiwan
{cjlee1010, rueycheng, denehs}@gmail.com
pjcheng@csie.ntu.edu.tw

ABSTRACT

Formulating appropriate and effective queries has been regarded as a challenging issue, since a large number of candidate words or phrases could be chosen as query terms to convey users' information needs. In this paper, we propose an approach to rank a set of given query terms according their effectiveness, wherein top ranked terms will be selected as an effective query. Our ranking approach exploits and benefits from the underlying relationship between the query terms, and thereby the effective terms can be properly combined into the query. Two regression models which capture a rich set of linguistic and statistical properties are used in our approach. Experiments on NTCIR-4 ad-hoc retrieval tasks demonstrate that the proposed approach can significantly improve retrieval performance, and can be well applied to other problems such as query expansion and querying by text segments.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information Search and Retrieval – *Query formulation*

General Terms

Algorithms, Measurement, Performance, Experimentation

Keywords

Query terms ranking, query formulation, query term combination

1. INTRODUCTION

It is clear that not all query terms involved in a retrieval process have the same retrieval effectiveness, and thus different combinations of query terms may lead to diverse performance results for information retrieval (IR). Query terms ranking is a research task aiming to rank a set of given query terms according to their effectiveness of retrieval. The task endeavors to discover effective or ineffective query terms, and assists users in formulating better queries by combining top ranked terms or removing ineffective terms from original queries. The obtained ranking list can additionally benefit many IR applications. For those search engines that only accept a few keywords in their search boxes, or those IR systems that only adopt a limited number of terms from feedback documents as expansion terms, the top ranked terms of the ranking list serve as good candidates to form an appropriate query. Previous work [15] attempts to sort query terms according to their effectiveness based on a greedy local optimal solution. It assumes each query term is independent of other terms being present in the same query. Thus, potential influence of query term dependency is neglected. In this paper, we propose a more general approach that takes term dependency into account to produce a preferable terms ranking list, accompanied by two applications of query terms ranking, including query expansion and querying by text segments.

To reveal the importance of capturing underlying term relations, let us examine the following expression of the user's information need, which is the description query of topic 25 in NTCIR-4: "*Find articles containing contents from reports on the decline of the unemployment rate as South Korea overcame the foreign exchange crisis.*" After removing stop words, we obtain "*contents, reports, decline, unemployment rate, South Korea, overcame, foreign exchange, crisis*" as query, which scores a mean average precision (MAP) of 0.0859. As not all of the query terms are equally effective in the retrieval process, for each term in the original query without stop words, we would rank a term t_i in front of another t_j when the drop of MAP is larger by removing t_i than removing t_j from the query, and then get the following ranking list:

> Ranking List-1: *unemployment rate, reports, contents, crisis, South Korea, overcame, decline, foreign exchange*

This ranking list is constructed by considering the effectiveness of a single term independently as in [15]. As we can see, terms "*South Korea*" and "*foreign exchange*" are ranked 5 and 8 respectively, which collide with our common sense that named entities and nouns may be more effective in IR. The reason for this ranking result stems from that each of the two terms is not good enough for distinguishing the relevant documents from the irrelevant ones. However, if "*South Korea*" or "*foreign exchange*" is properly combined together with other query terms to form a discriminative concept, another ranking list can be produced as follows:

> Ranking List-2: *unemployment rate, South Korea, foreign exchange, contents, crisis, reports, overcame, decline*

CIKM'09, November 2–6, 2009, Hong Kong, China.
Copyright 2009 ACM 978-1-60558-512-3/09/11…$10.00.

The second ranking list takes into account the underlying combination of terms that might be beneficial for terms ranking. It has been shown that "*South Korea*" or "*foreign exchange*" individually leads to weak MAP in the benchmark; however, once they are combined with "*unemployment rate*", "*unemployment rate, South Korea*" and "*unemployment rate, foreign exchange*" greatly advance to achieve MAP as 0.2992 and 0.1620 respectively. The top 2 ranked terms in Ranking List-2 show that the removal of "*unemployment rate, South Korea"* will result in severe information loss. Moreover, if we choose top 3 ranked terms from the two lists as the queries, Ranking List-2 obviously outperforms Ranking List-1, where "*unemployment rate, reports, contents*" and "*unemployment rate, foreign exchange, South Korea*" respectively obtain MAP of 0.1793 and 0.2701. This observation indeed points out the importance of modeling the underlying relationships between terms in the problem of query terms ranking.

Different combinations of query terms intrinsically bear unequal amount of information and thus behave distinctly in response to IR systems. Unfortunately, there are no explicit clues to help users determine what terms or which combinations are effective in IR. Previous works that attempt to measure the effectiveness of query terms include identifying key concepts within long and verbose queries [3], removing redundant query terms from original queries [10], finding good sub-queries [11,12], and selecting effective terms for query formulation [15]. Most of these works do not address the problem of term combination and analyze its impact on retrieval performance. There are some other works [2,5,20] focusing on predicting the difficulty of queries, but these works merely focus on evaluating the performance of a whole query and do not give insight into the impact of each query term in the retrieval process.

Provided with a set of possible query terms describing the users' information needs, the primary goal of this paper is to rank terms from the set according to their effectiveness, where top k ranked terms are selected as an appropriate query. Our ranking approach extends previous work [15] by learning two regression models from training data to predict the IR effectiveness of one term (regression model r_1) and two terms (regression model r_2) respectively. Model r_1 treats all terms independently as a bag of words, whereas model r_2 reveals the hidden relationships among combination of two terms. By proper integration (the hybrid model) of the two models, we can produce ranking lists that enjoy the benefits brought by both a single term and terms combination, and eventually formulate effective queries. Also, our approach comprehensively takes into consideration various factors that are essential in determination of retrieval performance, inclusive of linguistic properties and statistical relationships in a document collection. Experiments on NTCIR-4 ad-hoc IR tasks reveal that retrieval performance can be significantly improved based on our approach, compared to the performance of the original queries given in the benchmarks together with two other previous works [3, 15]. Finally, we also successfully apply the proposed approach to two IR problems, consisting of query expansion and querying by text segments, which thereby shows the extensibility of our ranking scheme.

In the rest of this paper, we first make a brief review on related work in Section 2, and describe our term dependency-based approach for query terms ranking in Section 3. The experimental results are presented in Section 4. Section 5 demonstrates the two applications, namely query expansion and querying by text segments, based on our ranking approach. Finally, in Section 6, we give our discussions and conclusions.

2. RELATED WORK

Query terms ranking intrinsically is a task that measures how effective each term inside the query is. Therefore, methods that intend to estimate the effectiveness of either a single term or the entire query are regarded as our related work, including:

Key concepts detection. Detection of key concepts is important in long queries for reducing noise and highlighting focus. [1] recognizes core terms of description queries based on linguistic and statistical methods. The appearance of a core term in a document makes the document relevant. [3] adopts machine learning methods for identifying weighted key concepts among verbose queries. Each noun phrase (candidate key concepts) represents the original verbose query with different degree of confidence, which is predicted by an AdaBoost.M1 classifier. Experiments in [3] demonstrate that retrieval performance is enhanced by adding two weighted concepts to original queries. These works lay emphasis on the extraction of key concepts from noun phrases and then re-weight the key concepts in queries to improve IR performance. Our term ranking approach differs in that (1) no weight assignments are needed and thus conventional retrieval models could be easily incorporated; (2) in addition to noun phrases, our approach takes other parts of speech (POS) and named entities into account simultaneously.

Query reformulation. Our goal of finding effective and informative terms among a query resembles [11,12], which improve MAP by visiting all possible sub-queries based on a user-interactive approach. Their approach determines optimal sub-query by constructing a maximum spanning tree with mutual information as the weight of its edge. However, their focus is to evaluate performance of a whole (sub) query whereas we consider units at the level of terms. Similarly, [10] attempts to predict what words in a query should be deleted based on query logs. [13] assigns weights to query terms, which can be subsequently added to original queries as an extension. Unfortunately, these methods cannot explain what properties make a query term significant or effective for search. [4] uses a supervised learning method for selecting good expansion terms from a number of candidate terms generated by the Indri model. With the same goal of selecting good terms with ours, however, (1) what [4] focuses is the relation between original queries and expansion terms, (2) consideration of linguistic features is absent in [4], and (3) query formulation based on the terms ranking list does not introduce extra terms outside original queries. As mentioned before, we extend previous work [15] that also ranks terms in original queries; nevertheless, [15] does not capture hidden term relation which is potentially beneficial for query terms ranking.

Predicting query performance. Predicting query performance [2] draws much attention for its connection with the capability of IR systems, and provides possible solutions to poorly-performing queries. Pre-retrieval predicting methods measure the performance of a query based on various characteristics of the query and document collection. [9] presents several pre-retrieval predictors, which predict the performance by computing relative entropy of query and document language models. Other post-retrieval predictors [5,6,19,20,21] measure the overlap of retrieved documents between using individual query term and the full query. In addition to the statistical-based methods mentioned above, [16] analyzes lexical properties of queries. [17] examines 16 kinds of linguistic features of query terms and [14] estimates the content load of lexical n-grams based on the amount of information carried by a POS block. In this paper, with the same goal of predicting performance, we differ from these works in that we make estimation on the effectiveness of a query term, instead of the whole query.

3. QUERY TERMS RANKING

3.1 Ranking Approach

Assume that, given a query topic, a user has a set of possible usable terms $T = \{t_1, t_2, \ldots, t_n\}$ that is relevant to the topic. Our major goal is to find a ranking function r: $T \rightarrow R$, which ranks $\{t_1, t_2, \ldots, t_n\}$ based on their effectiveness in retrieval. Once the ranking list is obtained, top k query terms will be selected to form a query. Note that the source of query term space T is not limited to the user's original query. T could be the set of terms from a long query. For a short query, T could include the terms from the query together with a set of expansion terms, the terms extracted from feedback documents initially returned from the given short query. Similarly, to discover key terms of a news title, T could also be the set of the terms appearing in the title. Three different sources of query term space T and various threshold k's have been examined in our experiments and applications in Sections 4 and 5.

An intuitive solution to find r is to maximize the following:

$$t_i^* = \arg\max_{t_i \in T} (\varphi(\mathrm{T}) - \varphi(\mathrm{T} - \{\mathrm{t_i}\})) / \varphi(\mathrm{T}) \qquad (1)$$

where $\varphi(\cdot)$ is a performance measure function such as MAP, which is used in this paper. It is believed that leaving out a relevant term can make the query semantics less accurate and result in decreased performance. Thus, query terms ranking can be carried out as follows. Once the most effective term t_i is chosen among T, it is removed from the term space. The second best term is extracted from T-$\{t_i\}$ in the next step. The process continues until the ranking list is fully generated.

One problem of Eq.(1) is that the selection of each query term t_i is determined independently, lacking consideration of latent terms relations. However, one less important term may become significant in retrieval when properly combined with another. To deal with the problem, we generalize Eq.(1) to the following:

$$c_m^* = \arg\max_{\mathrm{c_m} \subset \mathrm{T}} (\varphi(\mathrm{T}) - \varphi(\mathrm{T} - \{\mathrm{C_m}\})) / \varphi(\mathrm{T}) \qquad (2)$$

where c_m is a subset of T such that $c_m = \{t_i \mid t_i \in T\}$, and the number of terms in c_m is m, i.e., $|\ c_m\ | = m$, and $1 \leq m < |T|$. Specifically, Eq.(1) is equal to Eq.(2) when m=1. To compute Eq.(2), for each c_m, we develop a regression model r_m: $T \rightarrow R$ by learning examples in the form of

$$< f(c_m), (\varphi(\mathrm{T}) - \varphi(\mathrm{T} - \{\mathrm{c_m}\})) / \varphi(\mathrm{T}) >$$

where $f(c_m)$ is the set of features for c_m, which will be described in Section 3.2. Thus, r_m is able to predict how effective a group of m terms c_m is, and thereby can decide whether or not one m-term group is more effective than another m-term group. Given the number of query terms in a group, say m, query term space T is divided into $\lceil n/m \rceil$ groups $\{c_m^1, c_m^2, \ldots, c_{n\%m}^{\lceil n/m \rceil}\}$. Based on r_m, the $\lceil n/m \rceil$ groups are ranked into a sequence of

$$c_m^{\pi(1)} > c_m^{\pi(2)} > \cdots > c_{n\%m}^{\pi(\lceil n/k \rceil)}$$

where π is a permutation on {1, 2, …, $\lceil n/m \rceil$}. The sequence means that any term specified in group $c_m^{\pi(i)}$ is more effective than any other term specified in $c_m^{\pi(j)}$ for any i < j; that is, $c_m^{\pi(i)}$ is more effective than $c_m^{\pi(j)}$. Note that the least effective group contains n modulo m terms which may be less than m terms. Similar to Eq.(1), once the most effective group $c_m^{\pi(1)}$ is decided by r_m, terms in this group are removed from T. r_m keeps going on the selection of $c_m^{\pi(2)}$ among T-$c_m^{\pi(1)}$ and so on. More specifically, to obtain $c_m^{\pi(i)}$, we apply regression model r_m to any m-term combination from the subset of query term space $\{t_k \mid t_k \in T, t_k \notin c_m^{\pi(j)}, j < i\}$, and choose the currently best group as $c_m^{\pi(i)}$.

To produce final term list in order, for each $c_m^{\pi(i)}$ generated in previous step, we apply regression function r_{m-1} to derive the most effective (m-1)-term-group $c_{m-1}^{\pi(1)}$ and a single leave-out-term $c_m^{\pi(i)} - c_{m-1}^{\pi(1)}$ from the m terms contained in $c_m^{\pi(i)}$. Similarly, $c_{m-2}^{\pi(1)}$ is produced by the selection of the most effective m-2 terms specified in $c_{m-1}^{\pi(1)}$, using function r_{m-2}. This selection process is recursively performed until there exists merely one term in $c_1^{\pi(1)}$, namely, the most effective term in $c_m^{\pi(i)}$. The whole process is based on a "divide-and-conquer" method, wherein for each $c_m^{\pi(i)}$, we can get the sequence of

$$\begin{aligned}
&c_m^{\pi(i)} \\
&\rightarrow c_{m-1}^{\pi(1)} > c_m^{\pi(i)} - c_{m-1}^{\pi(1)} \\
&\rightarrow c_{m-2}^{\pi(1)} > c_{m-1}^{\pi(1)} - c_{m-2}^{\pi(1)} > c_m^{\pi(i)} - c_{m-1}^{\pi(1)} \rightarrow \cdots \\
&\rightarrow c_1^{\pi(1)} > c_2^{\pi(1)} - c_1^{\pi(1)} > \cdots > c_m^{\pi(i)} - c_{m-1}^{\pi(1)}
\end{aligned}$$

Notice that $c_{n\%m}^{\pi(\lceil n/k \rceil)}$ can be calculated similarly. The final terms ranking list is generated by combining all of $c_m^{\pi(i)}$ and $c_{n\%m}^{\pi(\lceil n/k \rceil)}$ in order, wherein terms belonging to each group of m terms are also well sorted. That is, query terms in the entire T are ranked according to their effectiveness.

In practice, due to insufficient training data, m is set to be 2 in this paper, that is, we train two regression models, including the 1-term model r_1 and the 2-term model r_2. The time complexity of using only r_1, i.e., Eq.(1), is O(n^2) while that of using r_1+r_2, i.e., Eq.(2), is O(n^3). By proper integration (the hybrid model) of r_1 and r_2, the terms ranking list can enjoy the benefits brought by both a single term and terms combination. The regression model we adopt in this paper is Support Vector Regression (SVR) [18], which is a regression analysis technique suitable for limited amount of training data based on SVM [8].

3.2 Features Used for Regression Models

We utilize linguistic and statistical features of one term t_i and term pair (t_i, t_j) for training the regression models r_1 and r_2 described in Section 3.1.

Linguistic Features: Our approach adopts parts of speech (POS), named entities (NE), acronym, phrase, and size (i.e., the number of words in a term) as the linguistic features. In our experiment, the POS features contain noun, verb, adjective, and adverb, while the NE features include person names, locations, organizations, and time. POS and NE in our experiments are labeled manually; nevertheless, it can be alternatively labeled automatically for the purpose of efficiency. For model r_1, the values of the linguistic features for term t_i are binary except for the size feature. For model r_2, we examine possible combinations of POS tags and NEs and label term pair (t_i, t_j) 1 if it satisfies certain linguistic properties. For example, if both t_i and t_j are nouns (or person names), then the feature pos_nn (or ne_pp) will be marked positive. The size feature of t_i and t_j is the mean of lengths of each. Additionally, we introduce the features pis and nenum for term pair (t_i, t_j) to respectively compute the weighted POS score and the number of NEs.

Statistical Features: Statistical features of term t_i or term pair (t_i, t_j) refer to the statistical information about the term(s) in a document collection. The information could be about the term(s) itself such as term frequency (TF) and inverse document frequency (IDF). Also, in order to capture the relationship between one term (or term pair) and the rest of terms in query term space *T*, we define four categories of the statistical features for both one term and term pair. The four categories include *term-term co-occurrence, term-topic co-occurrence, term-term context,* and *term-topic context* features. Note that the features defined for one term and two terms are respectively used for training models r_1 and r_2.

- *term-term co-occurrence features:*

Features in this category measure how often terms appear together in the document collections. For model r_1, the feature of term t_i depends on the co-occurrences of t_i and t_j ($t_j \in T$ and $t_i \neq t_j$). For term pair (t_i, t_j) in model r_2, it resembles the feature for the single term approach in r_1 except that the computation is required for both terms.

- *term-topic co-occurrence features*

The feature computes the co-occurrences of term t_i and T-{t_i} for model r_1 in the document collection. For model r_2, it similarly calculates co-occurrence of term pair (t_i, t_j) and T-{(t_i, t_j)} in the collection.

- *term-term context features*

This category relies on so-called context vectors from the search results. For model r_1, we calculate cosine similarity values over the context vector of t_i and that of all other t_j in *T* ($t_i \neq t_j$). For model r_2, given each term pair (t_i, t_j), context vectors of both (t_i, t_j) and other terms in *T* are needed to be computed pairwisely.

- *term-topic context features*

The feature computes the similarity between the context vectors of t_i and T-{t_i} for model r_1. For model r_2, the feature is defined as the similarity between the context vectors of (t_i, t_j) and T-{(t_i, t_j)}.

We further discuss into details about how these features are practically carried out and what meanings these features stand for. From now on, for simplicity, we symbolize either term t_i or term pair (t_i, t_j) by Γ and notate their corresponding complementary terms in *T*, that is T-{t_i} or T-{(t_i, t_j)}, by Γ'. Also, for any single term in Γ or Γ', we respectively denote it as γ or γ'. The *term-term* (or *term-topic*) *co-occurrence features* are used to measure whether or not term(s) in Γ could be replaced with γ' (or the whole Γ'), and its value shows how confident the substitution is. Terms that can hardly be replaced by others are thought to be key terms in *T*. In practice, we adopt three measures, including point-wise mutual information (PMI), Chi-square statistics (X^2), and log-likelihood ratio (LLR), to estimate the co-occurrences between γ and γ' for *term-term features*, and meanwhile the co-occurrences between Γ and Γ' for *term topic features*. Again, for simplicity, γ or Γ shall be recognized as *Y*, and its counterpart γ' or Γ' shall be marked as *Z*. Here we denote the total number of documents as *N* in the collection, the number of documents containing both *Y* and *Z* as *a*, the number of documents containing *Y* but not *Z* as *b*, the number of documents containing *Z* but not *Y* as *c*, and *d* is the number of documents containing neither *Y* nor *Z*, i.e., *d=N-a-b-c*.

PMI is a measure of association which quantifies the discrepancy between the dependent joint distribution and the independent individual distributions. Thus, PMI indicates that how much term(s) in *Y* would tell us about *Z*.

$$PMI(Y,Z)=\log\frac{\mathrm{p}(Y,Z)}{\mathrm{p}(Y)\mathrm{p}(Z)}\approx\log\frac{a\times N}{(a+b)(a+c)}.$$

X^2 compares the observed frequencies with frequencies expected for independence, and is a statistical method that tests whether two (or more) variables are independent or homogeneous.

$$\chi^2(Y,Z)=\frac{N\times(a\times d\text{-}b\times c)^2}{(a+b)\times(a+c)\times(b+d)\times(c+d)}.$$

LLR is a statistical test for making a decision between two hypotheses of dependency or independency based on the value of this ratio.

$$\begin{aligned}-2log\ LLR(Y,Z)= {}& a\times log\frac{a\times N}{(a+b)(a+c)}+b\times log\frac{b\times N}{(a+b)(b+d)}\\ &+c\times log\frac{c\times N}{(c+d)(a+c)}+d\times log\frac{d\times N}{(c+d)(b+d)}.\end{aligned}$$

When generating the *term-term co-occurrence features* for γ over all possible term pairs (γ, γ'), we make use of their average, minimal, and maximal values as follows:

$$f^{\theta}_{avg}(\gamma)=\frac{1}{|\Gamma||\Gamma'|}\sum\nolimits_{\gamma\in\Gamma,\gamma'\in\Gamma'}\theta(\gamma,\gamma')$$

$$f^{\theta}_{\max}(\gamma)=\max\nolimits_{\gamma\in\Gamma,\gamma'\in\Gamma'}\theta(\gamma,\gamma')$$

$$f^{\theta}_{\min}(\gamma)=\min\nolimits_{\gamma\in\Gamma,\gamma'\in\Gamma'}\theta(\gamma,\gamma')$$

where *θ* is *PMI*, *LLR* or X^2. In addition, for each *θ*, we sort all the γ according to the normalized feature values, and associate each γ with a ranking number as a new feature. We produce these new features for the purpose of avoiding domination of some certain training query terms.

The co-occurrence features are more reliable for estimating the relationship between high frequency query terms. Unfortunately, terms in Γ are probably not co-occurring with terms in Γ' in the document collection at all. Thus, we resort to *term-term* (*term-topic*) *context features*, which are helpful for low frequency query terms that yet share common contexts in search results. More specifically, we generate the context vectors from the search results of *Y* and *Z* respectively. The context vector is composed of a list of pairs <document ID, relevance score>, which can be obtained from the search results returned by IR systems. The contextual relationship between *Y* and *Z* can be determined by the cosine similarity of their context vectors. Note that much more computation time is required to extract the context features, since the retrieval process is involved. In contrast, the co-occurrence features can be quickly obtained from the indices of IR systems. Also, the effectiveness of context features is deeply influenced by the goodness of retrieval models.

4. EXPERIMENTS

4.1 Experimental Data

We conduct several experiments to measure the effectiveness and reliability of our terms ranking approach. The data used in the experiments is NTCIR-4 English-English ad-hoc IR tasks, whose statistics in data collection can be found in Table 1. Description queries are adopted for evaluation, and its average length is 14 query terms. Note that in Section 5, we shall use the rest of queries, i.e., NTCIR-5, as shown in Table 1. To examine the robustness of our approach across different frameworks, three retrieval models are used throughout our experiments and are constructed using the Lemur Toolkit[1], including the vector space model (TFIDF), the language model (Indri) and the probabilistic model (Okapi). Also, we stem both queries and documents with Porter stemmer and remove stop words from original queries. The remaining query terms in each query topic form a query term space *T*. We use MAP as performance metric evaluating over top 1000 documents retrieved. Also, we filter the poorly-performing queries whose MAP is below 0.02 to ensure the quality of our training data. Table 2 summarizes the settings for training instances. In Table 2, one can see that there are different numbers of training and test instances in different models, which results from that different retrieval models have different MAP on the same queries. To balance the ratio of positive and negative instances, we up-sample the positive instances by repeating them up to the same number as the negative ones.

Table 1. Statistics of NTCIR-4 and NTCIR-5 datasets.

Setting		Adopted dataset after data clean		
		Number of query topics	Number of distinct terms	Number of query terms
NTCIR4	title	44	216	4.90
	desc	58	865	14.90
NTCIR5	title	35	198	5.65
	desc	47	623	13.20

Table 2. Numbers of training and testing instances (positive : negative) in NTCIR4 <desc>.

	Indri	**TFIDF**	**Okapi**
Training for model r_1			
Original	674(156:518)	702(222:480)	687(224:463)
Upsample	1036(518:518)	960(480:480)	926(463:463)
Train	828(414:414)	768(384:384)	740(370:370)
Test	208(104:104)	192(96:96)	186(93:93)
Training for model r_2			
Original	804(210:594)	788(259:529)	778(277:501)
Upsample	1188(594:594)	1058(529:529)	1002(501:501)
Train	950(475:475)	846(423:423)	802(401:401)
Test	238(119:119)	212(106:106)	200(100:100)

4.2 Performance of Regression Models

For statistical models whose central purpose is the prediction of future outcomes on the basis of observed data, the coefficient of determination R^2 measures the proportion of variability in a data set. It serves as a measure of how well future outcomes are likely to be predicted by the model. In our case, the R^2 statistics ($R^2 \in [0, 1]$) is used to evaluate the prediction accuracy of regression model r_2, and is defined as one minus the ratio of the residual sum of squares and the total sum of squares:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

Hence, R^2 statistics explains the variation between true label $y_i = (\varphi(\mathrm{T}) - \varphi(\mathrm{T} - \{\mathrm{c_m}\}))/\varphi(\mathrm{T})$ and fit value $\hat{y}_i = wf(t_i) + b$ for each testing query term $t_i \in \mathrm{T}$, or $\hat{y}_i = wf(t_i, t_j) + b$ for each term pair (t_i, t_j) as explained in Section 3.2. $\bar{y}$ is the mean of the ground truth. In the training process, we use 5-fold cross validation for training and testing regression models r_1 and r_2. We also guarantee that all the training instances are different from instances of the test set to avoid inside test due to up-sampling.

Figure 1 shows the R^2 results of regression model r_2. Distribution of r_1 [15] generally resembles that of r_2; however, r_1 achieves even higher R^2 value in average, caused by the fact that it is harder to capture complicated interleaving term relations than individual term. In Fig 1, two extra features are introduced for even boosting performance, namely, m-CL and m-SCS. The modified content load (m-CL) sets weight of a noun as 1 and the weights of adjectives, verbs, and participles as 0.147 in the issued query. This feature adopts previous definition of Content Load (CL) [14] that gives unequal importance to words of different POS. Our m-SCS references the simplified clarity score (SCS) [9] by calculating the relative entropy between query and collection level distributions (unigram language models).

Figure 1 shows that no matter what retrieval model is used, the more the features are included for training, the larger the R^2 values tend to become. In addition, the statistical features consistently achieve higher R^2 values than the linguistic features do. It is caused by that the statistical features reflect the underlying distribution of the query terms in the document collection. Further, we can tell that the improvement brought by m-CL and m-SCS is not obvious, which comes from their similarities to other features. As the linguistic and statistical features are complementary, we use all of the features in the following experiments.

4.3 Feature Analysis

One of the interesting findings of this work is to discover which features of query terms are influential on retrieval and responsible for their IR effectiveness. We analyze correlation between the features and MAP with three standard measurements, namely Pearson's product-moment, Kendall's tau and Spearman's rho.

Figure 2 shows our analytical results of model r_2. In all cases, removing a term having high context feature value from the topic leads to high deviation in the result set. Specifically, context features "cosine_avg" (*term-term*) and "cosine_topic" (*term-topic*) are found to be highly related to MAP (ρ>0.5). These observations imply that the context features are more discriminative in estimating the effectiveness of query terms than the others, but such features suffer from the cost of higher computation time. Figure 2 also shows the co-occurrence features such as PMI, LLR and X^2 have strong connection to MAP. Moreover, the "coccur" feature which measures how often the two terms of each term pair appear together in the collection has moderate correlation with MAP. These results support that dependence between query terms is helpful in predicting the importance of query terms in retrieval.

[1] Lemur Project: http://www.lemurproject.org/

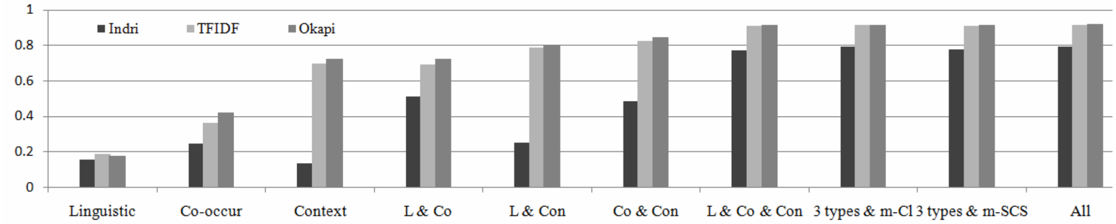

Figure 1. R^2 value of regression model *r2*, L: Linguistics, Co: Co-occur features, Con: Context features, 3 types: L & Co & Con

For the linguistic features, a longer term or part of phrase is intuitively more useful than the shorter ones, in terms of IR performance. In general, it is believed that the longer a term is, the less ambiguity and the more information it contains. This explains why the linguistic features "size" and "phrase" positively correlates to MAP ($0.3< \rho <0.5$). We notice that the feature "pos_nn" has higher correlation to MAP than other linguistic features do. This conforms to a common belief in search that nouns are more important than the others. However, other features like "ne_pp", "ne_go", and "ne_gt", which are combinations of two named entities, do not exhibit this property. We consider this as a result of insufficient training data. Finally, Fig. 2 also shows the high correlation to MAP of features "tf", "idf" and "m-SCS". Overall, the statistical features are more powerful in estimation of query term effectiveness than the linguistic ones.

4.4 Evaluation on Information Retrieval

In this section, we conduct experiments for evaluating our query terms ranking approach in IR. We use NTCIR-4 as the dataset and topic <desc> as the queries.

Table 3. MAP of NTCIR4<desc>. TFIDF and Okapi models have PRF involved, Indri model does not. T-test with $p < 0.01$ (**) and $p< 0.05$ (*). Best MAP of each retrieval model is marked bold.

Method	Indri	TFIDF	Okapi	Avg.
BL1: original	0.1742	0.2660	0.2718	0.2373
BL2: noun	0.1773	0.2622	0.2603	0.2332
UpperBound	0.2233	0.3052	0.3234	0.2839
KeyConcept	**0.2065****	0.2719*	0.2710	0.2498*
One_term: r_1	0.1954**	*0.2861***	0.2875*	0.2563**
Hybrid: $r_1 + r_2$	0.2029**	***0.2880*****	***0.2917*****	0.2609**

The results of the experiments with 5-fold cross-validation are given in Table 3. Two baseline methods are included in our experiments: "BL1" method simply selects all the query terms in *T* as one query string, whereas "BL2" method formulates queries by choosing terms whose POS tags are nouns. Besides, for each topic, we permute all sub queries and discover the sub-query with the highest MAP value, denoted as "UpperBound". We have implemented the method "KeyConcept" [3] for performance comparison, where two weighted key concepts are added to original description query. Note that, however, since the KeyConcept method demands different weighting on different terms, which is not applicable for TDIDF and Okapi models (e.g., lack support of Indri query language), we use equal weights for the two selected concepts in these two models. The rest of methods are based on either one term model r_1, where query terms are independently ranked [15], or the proposed hybrid model r_1+r_2, which emphasizes terms relation. The retrieval results are presented in terms of MAP. We also run the two-sample pairwise significance test for each method (against BL1).

As we can see in Table 3, two baseline methods share similar MAP results. It is inferred that some nouns may still be noisy while some terms of other POS categories may be helpful for IR. Further, one term model r_1 and hybrid model r_1+r_2 significantly outperform the baseline methods with progress by 7.79% to 11.87% of MAP. It is important to note that the proposed hybrid model consistently performs better than one term model. This again proves our assumption that the hybrid model considering term relation with r_2 is more preferable in query terms ranking. Also, all the methods show significant improvements when applied to dissimilar retrieval models, thereby revealing the reliability and robustness of our ranking approach.

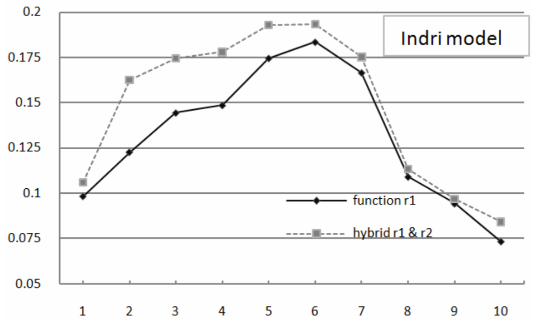

Figure 3. MAP curves based on one term model r_1 and hybrid model r_1+r_2 with NTCIR-4 <desc> query. X coordinate is the number of query terms and y coordinate illustrates MAP.

Figure 3 shows the MAP curve for each ranking scheme by connecting the dots at (1, $MAP^{(1)}$), … , and (n, $MAP^{(n)}$), where $MAP^{(i)}$ is the MAP of top *i* query terms selected as the issued query. From Fig. 3, two MAP curves share an interesting tendency: the curves keep going up in the first few iterations, while after the maximum (locally to each method) is reached, they begin to go down quite rapidly. Thus, from a general perspective, the findings might informally establish the validity of our assumption that a longer query topic might encompass more noisy terms. Yet if we inspect some query topics into detail to observe micro-phenomenon, we can discover that MAP again climbs up after the "up-and-down" pattern. This discovery is somehow not surprising even though it is assumed that our algorithm may select terms of higher effectiveness during earlier iteration. The reason for rising MAP is that these terms act as terms suggested by query

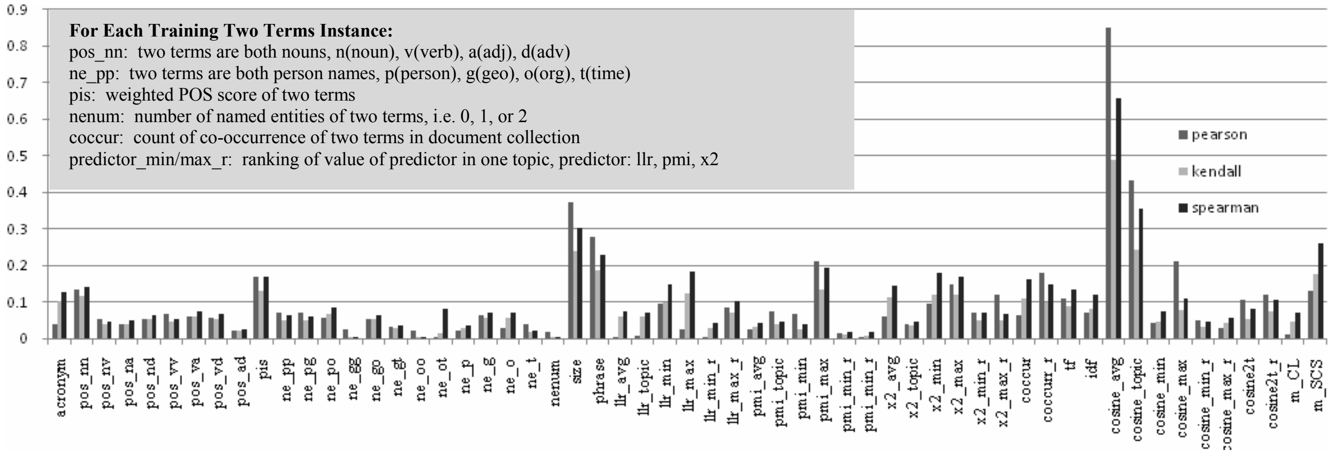

Figure 2: Three correlation values between features and MAP of regression model r_2 on Okapi retrieval model

expansion, and thus once again bring up MAP by providing extra information to IR systems. Finally, it is clearly told that the hybrid model r_1+r_2 boosts MAP more rapidly than r_1 does, by inspecting Fig 3. This again points out the importance of relation existing between query terms captured by r_2.

4.5 Analysis of Term Combination

As stated in Section 3.2, in order to explore the impact of the combination of query terms on IR, we extract the linguistic features of individual term and combine these features together such as ne_pp (two person names) to train model r_2. To analyze the relationship, we first remove one category of NE (*X*) from original query *Q*, which is followed by secondly removing another category of NE (*X* plus *Y*) from *Q*. We introduce a relationship ratio for measurement:

$$\text{Ratio } \ R = \frac{\Delta\text{MAP(Q,X)} - \Delta\text{MAP(Q,XY)}}{\Delta\text{MAP(Q,X)}}$$

$$\Delta_1 = \Delta\text{MAP(Q,X)} = \text{MAP(Q)} - \text{MAP(Q}-\text{X)}$$

$$\Delta_2 = \Delta\text{MAP(Q,XY)} = \text{MAP(Q)} - \text{MAP(Q}-\text{XY)}$$

Δ_1 itself gives us how important a certain category NE *X* is, and *X* has a greater impact on IR if its Δ_1 is larger than other categories. In addition, since $\Delta\text{MAP(Q,XY)} \geq \Delta\text{MAP(Q,X)}$ (as each category causes a drop of MAP, combination of two categories *X* and *Y* may cause a larger drop), we find that the larger the absolute value of ratio *R*, the stronger connection between *X* and *Y* is, given a certain *X*. Table 4 summarizes the results.

From Table 4, we can see that Δ_1 of "org" is the largest among all NEs, and thus organization name is considered mostly important in NTCIR-4 on the TFIDF model. The same experiment has been conducted on NTCIR-5, yet the result shows that person name is the most important category, which reveals that the result is collection-dependent. Furthermore, we discover that (1) given a category *X*, there exists a category *Y* that contributes a most significant drop of MAP by removing *X* and *Y* from *Q*, with a largest absolute *R* (marked bold for each *X*). For example, |*R*| of combination of (person, organization) is the largest in both categories "person" (1.992) and "org" (0.6763), which implies the relation of the two kinds of NEs is tight and together often constructs a concrete concept.

We also observe that (2) Δ_1 of category "geo" is relatively small, yet once "geo" is combined with "org", the resulting value of Δ_2 climbs up to 0.0475, causing a large |*R*| of 2.3929. It shows that "geo" and "org" together form a combination relation. Consider topic 031, which concerns military operation of organization NATO (org) in Yugoslavia (geo). Since the action of NATO bombing took place in Yugoslavia, combining NATO with Yugoslavia implies the event occurring in Yugoslavia.

Lastly, we notice that (3) category "time" has pretty large |*R*| values. This is because there are insufficient training data for "time", and thus of Δ_1 of "time" is very small, causing much greater |*R*| values than other categories.

Table 4. Relation between combinations of NEs on TFIDF model. Original NTCIR4 <desc> queries have MAP 0.2660 on TFIDF model.

Remove X (MAP)	Δ_1	Y	Remove XY (MAP)	Δ_2	Ratio R
person (p) 0.2409	0.0251	g	0.2253	0.0407	-0.6215
		o	0.1909	0.0751	**-1.9920**
		t	0.2402	0.0258	-0.0279
geo (g) 0.2520	0.0140	p	0.2253	0.0407	-1.9071
		o	0.2185	0.0475	**-2.3929**
		t	0.2514	0.0146	-0.0429
org (o) **0.2212**	0.0448	p	0.1909	0.0751	**-0.6763**
		g	0.2185	0.0475	-0.0603
		t	0.2203	0.0457	-0.0201
time (t) 0.2658	0.0002	p	0.2402	0.0258	-128.00
		g	0.2514	0.0146	-72.000
		o	0.2203	0.0457	**-227.50**

Moreover, as we have identified some important combinations of NEs, we further explore, inside these combinations, which term combination (as opposed to category combination) is more important for IR. Take terms of combination of (person, organization) as example. We find some cause obvious drop of MAP while some do not. Consider topic 006 of "*Find articles* *containing the reasons for NBA Star Michael Jordan's retirement and what effect it had on the Chicago Bulls*", if we remove separately "*Michael Jordan, NBA Star*" (person+org) or "*Michael Jordan, Chicago Bulls*" (person+org) from the original query, we get MAP 0.2275 and 0.0778, respectively. This remarkable difference of MAP indicates that: (a) not all term combinations are equally informative even if they share the same linguistic features, and (b) the statistical features may make term combination such as "*Michael Jordan, Chicago Bulls*" more important for IR, which stems from the fact that "*Michael Jordan*" and "*Chicago Bulls*" have a tendency to appear together in document collections to convey the concept of a "*retirement*" event. We have learned a lesson from this example that statistical features, when defined appropriately, allow us to use sub-queries to capture key query concepts whilst also reduce the information noises, which is a task that linguistic features barely accomplish.

5. IR Applications

5.1 Query Expansion

In this section, we will show that our term ranking scheme can be applied to query expansion; that is, the source of term space T comes not only from the description field in benchmark dataset as in Section 4.4, but from an arbitrary external expansion set. We devise two experiments such that (1) the proposed hybrid model runs on the expansion set and selects top k terms as expansion terms to description queries, and (2) the proposed hybrid model runs on the description field of NTCIR-4 and selects top k terms as expansion terms to corresponding title queries. Detailed data sets adopted in this experiment can be found in Table 1.

As aforementioned in Section 2, [4] proposes a method for selecting good expansion terms based on an SVM classifier. Our approach is also applicable to the selection of effective query expansion terms. Given a set of candidate expansion terms which are generated by conventional approaches such as TF and IDF, we apply our hybrid model to the expansion set, inside which terms are ranked according their effectiveness (with the NTCIR-4 5-fold cross validation regression model). Table 5 shows the MAP results of the hybrid model and the baseline method (BL), where BL simply adds all high-frequency expansion terms to original queries. Note that, as we have shown the superiority of the hybrid model to one term model in Section 4.4, we merely adopt the hybrid model here for query terms ranking in NTCIR-4 and NTCIR-5. From Table 5, the hybrid model outperforms BL under different retrieval models and datasets, improving MAP by 2.57% to 8.05% compared to the baseline. Moreover, though extra terms are introduced for query formulation, we can see that certain MAP results in Table 3 still outperform those in Table 5 (marked *italic*). It is therefore inferred that, it is still important to filter out noisy terms in original queries even though good expansion terms are selected. Finally, note that we use the NTCIR-4 5-fold cross validation regression model, which is trained to fit the target performance gain in NTCIR-4 dataset, rather than instances in the query expansion terms set. However, results in Table 5 show that this model works satisfactorily in the selection of good expansion terms, which ensures that our approach is robust in different environments and applications such as query expansion.

Next, we focus on how the hybrid model helps the title queries in NTCIR-4 in terms of retrieval performance. The hybrid model attempts to rank query terms from description field of NTCIR-4, and adds top k effective terms regarded as expansion terms to the corresponding title queries (with the NTCIR-4 5-fold cross validation regression model). Table 6 shows the experimental result, by which we can tell that better MAP results are consistently acquired than all original title queries (BL1), the description queries (BL2), and even the title plus description queries (BL3). Also, no matter what retrieval model is used, the hybrid model is capable of choosing effective expansion terms, thereby improving 10.1% to 16.2% of MAP. In this experiment, we verify the adaptability and feasibility of our mechanism of learning effectiveness of query terms and describe the extensibility to other applications such as pseudo-relevance feedback (PRF).

Table 5. MAP of query expansion based on hybrid model in NTCIR-4 and NTCIR-5 <desc>. T-test with p < 0.01 (**) and p< 0.05 (*) against baseline method.

Setting	Method	Indri	TFIDF	Okapi	Avg.
NTCIR-4 <desc>	BL	0.2470	0.2642	0.2632	0.2581
	Hybrid:r_1 +r_2	0.2610**	0.2860**	0.2899**	0.2789
NTCIR-5 <desc>	BL	0.1795	0.1891	0.1913	0.1866
	Hybrid:r_1 +r_2	0.1880*	0.1918*	0.1945*	0.1914

Table 6. MAP of query expansion from NTCIR-4 <desc> to title queries. All three models have PRF involved. T-test with p < 0.01 (**) and p< 0.05 (*) against BL2.

Setting	Method	Indri	TFIDF	Okapi	Avg.
NTCIR-4 <title> <desc>	BL1:<title>	0.2143	0.2417	0.2664	0.2408
	BL2:<desc>	0.2252	0.2660	0.2718	0.2543
	BL3:<t + d>	0.2295	0.2461	0.2777	0.2511
	Hybrid:r_1 +r_2	0.2611*	0.2683**	0.3107**	0.2800

5.2 Querying by Text Segments

As we have developed a term ranking scheme which is proved to be effective on different benchmark collections, we are now interested in realizing if this approach can be well applied to Web environments. Web pages are often composed of text segments in form of a body of words such as keywords and sentences. When users are interested in more information about certain text segments, they naturally formulate their own query (viewed as their information need T) based on these text segments to search web pages. In this experiment, 10 text segments with 74.8 words in average are manually selected from Google news[2]. 13 subjects are asked to read the 10 given text segments, and generate their own queries (UQ) to find related documents about the segments. Similarly, our approach tends to select top effective terms from the text segments as queries (AQ). The generated queries UQ and AQ are sent to Google[3] and their search results UR and AR are returned. The subjects are required to score the quality of AQ and AR, and judge the accuracy of AR and UR with a score varying from 1 (worst) to 5 (best).

From Table 7, it can be seen that although the subjects think that AQ only has moderate similarity (3.2/5.0) to UQ, AQ still looks reasonable to them (3.5/5.0). When simply checking the search results returned from Google, the subjects agree that AR is highly

[2] http://news.google.com.tw

[3] http://www.google.com

relevant to the content of the text segments (4.0/5.0). If they carefully examine each search result, i.e., download the full page, AR achieves the best performance at P@3, P@5 and P@7. Table 8 shows a real example of the text segment, UQ, and AQ. We find that the subject seems to select highly discriminative terms for the player of Kuroda Hiroki, “right shoulder” and “starter,” based on their linguistic features. But it is hard for her to know which terms are statistically important. That is why AR performs better than UR. The precision of using the entire text segments as a query is not high because such method often tries to match as many as terms in search results, causing that few documents can be found.

Table 7. User study on querying by text segments.

Quality of AQ and AR	**Score**	**Description**	
Reasonability of AQ	3.5	Is AQ reasonable to subject?	
Similarity of AQ to UQ	3.2	Is AQ similar to UQ?	
Relevancy of AR	4.0	Is AR relevant to text segment?	
Compare AR and UR	**P@3**	**P@5**	**P@7**
Performance of UR	0.84	0.80	0.78
Performance of AR	0.93	0.90	0.83
Perf. of entire text seg.	0.80	0.72	0.64

Table 8. An example of text segment, UQ, and AQ.

Text segment	According to Japanese media reports, the Los Angeles Dodgers pitcher Kuroda Hiroki has told the supervision Hara by telephone, he was unable to recover from right shoulder injury so cannot to take part in next year's World Baseball Classic. Hara listed Kuroda as the “No. 4 starter” in the 1st alternate list of 34 people, now has to be adjusted again.
UQ	Kuroda Hiroki, right shoulder, starter
AQ	Hara, Hiroki, Dodgers, World Baseball Classic

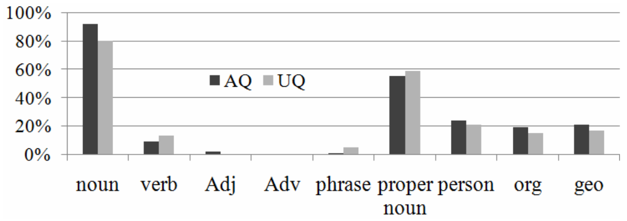

Figure 4. Distributions of AQ and UQ terms.

We make several further investigations on the relationship between AQ and UQ: (1) We find that more than 99% terms in UQ are generated from the given text segments. It is, therefore, reasonable that our approach extracts terms mainly from the text segments as queries. (2) The average number of terms in UQ is about 3.0, which is close to 4.0 of AQ. The numbers are close to those of real web queries whose average query length are about 2.3 words in English and 3.18 characters in Chinese. (3) There are averagely about 1.24 terms in AQ also appearing in UQ. The overlapping percentage of AQ and UQ is 41.3%. (4) The distributions of AQ and UQ over different linguistic characteristics are very similar. Figure 4 shows that both AQ and UQ prefer nouns, including proper nouns. Named entities like person, organization, and location names seem to carry abundant informative content. (5) AQ often contains terms with wrong boundaries in Chinese due to the errors produced by Chinese word segmentation. This is why the reasonability of AQ is scored by 3.5/5.0 only. However, segmentation errors do not affect the retrieval performance much. (6) More importantly, the average time spent by the subjects to generate one query is about **34.4** seconds, compared to the time of **2.0** seconds in average required by our hybrid approach to generate a query. Most of the time taken by our approach is to compute the values of the features such as named entity recognition.

6. DISCUSSIONS AND CONCLUSIONS

In this paper, we measure and predict the importance of query terms while as well construct effective queries based on this knowledge, namely the terms ranking list. In addition to the term-independent assumption (bag-of-words model), we advance to take into account the relationship of combination of terms, which captures underlying dependencies that are beneficial to IR performance. Our experiments show that the proposed approach is robust and effective in formulating good queries and the performance gain is consistent across different retrieval models and document collections. Another contribution of this work is that we capture certain types of terms combinations which convey representative concepts in original queries and assist IR performance, as well that we provide insights to identify what kind of the characteristics of query terms play important roles in retrieval tasks. Finally, we also show that our ranking scheme works satisfactorily on some external sources of term space *T* (e.g., query expansion). The user study of querying by text segments also points out that our ranking approach is applicable to form proper queries using text fragments in Web pages.

Our approach practically approximates global optimal ranking list of terms by iteratively selecting the best candidate terms or best pair of terms (as described in Section 3). Such a local optimization scheme trades some IR quality for running time. In addition, as mentioned in Section 4.2, the more complicated relation of terms combination is considered, the more difficult the training of regression models can be carried out. The insufficient data problem becomes even severe when more terms combination are included (in this paper, we at most consider two terms simultaneously). Also, Section 4.5 points out what kind of term combination has more influence on IR performance; however, the results are collection-dependent, which cannot generally explain common behaviors. Meanwhile, we are not able to automatically choose the best value for parameter *k*, which is anticipated to optimize the retrieval performance for each query topic in our algorithms (*k* is manually selected to optimize each topic in this paper). Though given the difficulty of automatic determination of *k*, it turns out that a fixed value 4 still works acceptably on all retrieval models in our experiments. Finally, like all other training-based algorithms, we have to obtain the Web corpus for statistical features before applying our method to Web applications. We leave these limitations as our future work.

ACKNOWLEDGES

This work was partially supported by the National Science Council, Taiwan, under contact NSC97-2221-E-002-222-MY2.

REFERENCES

[1] J. Allan, J. Callan, W. B. Croft, L. Ballesteros, J. Broglio, J. Xu, and H. Shu. INQUERY at TREC-5. In *Proc. of the Fifth Text Retrieval Conference TREC-5*, pages 119-132, 1997.

[2] G. Amati, C. Carpineto, and G. Romano. Query difficulty, robustness, and selective application of query expansion. In *Proc. of 26th European Conference on IR Research*, pages 127-137, 2004.

[3] M. Bendersky and W. B. Croft. Discovering key concepts in verbose queries. In *Proc. of 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 491-498, 2008.

[4] G. Cao, J. Y. Nie, J. F. Gao, and S. Robertson. Selecting good expansion terms for pseudo-relevance feedback. In *Proc. of 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 243-250, 2008.

[5] D. Carmel, E. Yom-Tov, and I. Soboroff. SIGIR WORKSHOP REPORT: Predicting query difficulty - methods and applications. In *Proc. of the ACM SIGIR 2005 Workshop on Predicting Query Difficulty - Methods and Applications*, pages 25-28, 2005.

[6] D. Carmel, E. Yom-Tov, A. Darlow, and D. Pelleg. What makes a query difficult? In *Proc. of 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 390-397, 2006.

[7] D. Carmel, E. Farchi, Y. Petruschka, A. and Soffer. Automatic query refinement using lexical affinities with maximal information gain. In *Proc. of 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 283-290, 2002.

[8] C. C. Chang and C. J. Lin. LIBSVM. http://www.csie.ntu.edu.tw/~cjlin/libsvm , 2001.

[9] B. He and I. Ounis. Inferring query performance using pre-retrieval predictors. In *Proc. of 11st International Conference of String Processing and Information Retrieval*, pages 43-54, 2004.

[10] R. Jones and D. C. Fain. Query word deletion prediction. *Proc. of 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 435-436, 2003.

[11] G. Kumaran and J. Allan. Effective and efficient user interaction for long queries. In *Proc. of 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 11-18, 2008.

[12] G. Kumaran and J. Allan. Adapting information retrieval systems to user queries. *Information Processing and Management*, pages 1838-1862, 2008.

[13] K. L. Kwok. A new method of weighting query terms for ad-hoc retrieval. In *Proc. of 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 187-195, 1996.

[14] C. Lioma and I. Ounis. Examining the content load of part of speech blocks for information retrieval. In Proc. of *COLING/ACL 2006 Main Conference Poster Sessions*, pages 532-53, 2006.

[15] C. J. Lee, Y. C. Lin, R. C. Chen, and P. J. Cheng. Selecting effective terms for query formulation. In *Proc. of the Fifth Asia Information Retrieval Symposium*, 2009.

[16] T. Mandl and C. Womser-Hacker. Linguistic and statistical analysis of the CLEF topics. In *3rd Workshop of the Cross-Language Evaluation Forum CLEF*, 2002.

[17] Mothe, J., Tanguy, L. Linguistic features to predict query difficulty. In *Proc. of the ACM SIGIR 2005 Workshop on Predicting Query Difficulty - Methods and Applications*, 2005.

[18] V. N. Vapnik. Statistical Learning Theory. John Wiley & Sons, 1998.

[19] E. Yom-Tov, S. Fine, D. Carmel, A. Darlow, and E. Amitay. Juru at TREC 2004: Experiments with prediction of query difficulty. In *Proc. of 13th Text Retrieval Conference*, 2004.

[20] Y. Zhou and W. B. Croft. Query performance prediction in web search environments. In *Proc. of 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 543-550, 2007.

[21] Y. Zhou and W. B. Croft. Ranking robustness: A novel framework to predict query performance. In *Proc. of 15th ACM International Conference on Information and Knowledge Management*, pages 567-574, 2006.