A novel term weighting scheme based on discrimination power obtained from past retrieval results

Sa-kwang Song*, Sung Hyon Myaeng

Korea Institute of Science and Technology Information, 245 Daehak-ro, Yuseong-gu, Daejeon 305-806, South Korea
Division of Web Science and Technology, Korea Advanced Institute of Science and Technology, 291 Daehak-ro, Yuseong-gu, Daejeon 305-701, South Korea

A R T I C L E   I N F O
Article history:
Received 31 March 2011
Received in revised form 19 March 2012
Accepted 21 March 2012
Available online 18 April 2012

Keywords:
Term weighting
Evidential weight
Discrimination power
Language model
Probabilistic model

A B S T R A C T
Term weighting for document ranking and retrieval has been an important research topic in information retrieval for decades. We propose a novel term weighting method based on a hypothesis that a term’s role in accumulated retrieval sessions in the past affects its general importance regardless. It utilizes availability of past retrieval results consisting of the queries that contain a particular term, retrieved documents, and their relevance judgments. A term’s evidential weight, as we propose in this paper, depends on the degree to which the mean frequency values for the relevant and non-relevant document distributions in the past are different. More precisely, it takes into account the rankings and similarity values of the relevant and non-relevant documents. Our experimental result using standard test collections shows that the proposed term weighting scheme improves conventional TF-IDF and language model based schemes. It indicates that evidential term weights bring in a new aspect of term importance and complement the collection statistics based on TF-IDF. We also show how the proposed term weighting scheme based on the notion of evidential weights are related to the well-known weighting schemes based on language modeling and probabilistic models.

1. Introduction

Term weighting for document retrieval and ranking has been a key research issue in information retrieval for decades (Cummins & O’Riordan, 2006; Gerard Salton, 1988; Robertson, 2004; Robertson & Sparck Jones, 1976). Most popular term weighting methods are based on document oriented statistics such as term frequency (TF) in a document and inverse document frequency (IDF) from a collection. For term weighting, TF-IDF based schemes and their variations have been proven to be robust and difficult to be beaten, even by much more carefully designed models and theories (Amati, 2003; Gerard Salton, 1988; Pérez-Aguera, Arroyo, Greenberg, Iglesias, & Fresno, 2010; Robertson, 2004; Robertson & Sparck Jones, 1976; Robertson & Zaragoza, 2009). Term frequency is a document-specific local measure and typically computed as follows:

$$TF(t) = \frac{rtf}{\max freq}$$

where rtf is the raw term frequency and the max freq is the frequency of the most common term in the document.

The inverse document frequency or IDF (Robertson, 2004) is based on counting the number of documents containing the term in a query in the collection being searched. The most commonly cited form of IDF is as follows:

$$IDF(t) = \log \left( \frac{|D|}{\sum_{d \in D} \mathbf{1}(t \in d)} \right)$$

where D is the set of all documents and t ∈ d indicates that term t occurs in document d.
IDF(t) = \log \frac{N}{df_t}

where \( N \) and \( df_t \) are the total number of documents in the collection and the frequency of documents containing a term \( t \), respectively. Calculation of the weight of a term \( t \) in a document can be described as follows (Cummins & O’Riordan, 2006):

\[
\text{Weight}(t) = W_l(t) + W_g(t) = \text{TF}(t) \times \text{IDF}(t)
\]

where \( W_l(t) \) and \( W_g(t) \) represent local and global weights, respectively.

As an attempt to complement the document collection statistics, we propose to include external information derived from accumulated retrieval experiences and introduce a term’s evidential weight in addition to local and global information. It is obtained from a term’s history of separating relevant from non-relevant documents, which can be found in past retrieval results. A term’s history can be obtained directly from conventional test collections constructed through TREC, CLEF, and NTCIR or indirectly from click-through data (see Xue et al. (2004) and Joachims (2003) for example) that have been made available with commercial search engines.

IDF is considered to have a selective power since a term with a low IDF value appears in the majority of documents in a collection and hence fails in selecting a small subset to be retrieved. However, it is different from the concept of evidential weight in that it has nothing to do with the ability of selecting only relevant documents. A high IDF term in a query can select a small number of documents containing it, but its specificity carries no information about how useful it may be in retrieving relevant documents. When a high-IDF term in a query has two senses with roughly equal numbers of occurrences in a collection, for example, about a half of the retrieved documents would be irrelevant. When observed over a large number of queries, two terms with similar IDF values may play different roles in separating relevant from non-relevant documents. We attempt to show that this role of a term in past queries predicts its value in future queries, which is measured as discrimination power (DP).

Assuming that we can compute such a discrimination power for each term, the term weight formula in (1) can be rewritten to incorporate the new term weight based on DP as follows:

\[
\text{Weight}(t) = \text{TF}(t) \times \text{IDF}(t) \times \text{DP}(t)
\]

In this paper, we introduce a method for computing DP values using a set of queries and the retrieved documents for which relevance judgments are available. DP values are computed using the ranks or the similarity values of the relevant and non-relevant documents retrieved by a query.

2. Related work

Relevance feedback (Allan, 1996; Belkin et al., 1996; ChengXiang and Lafferty, 2001; Iwayama, 2000; Joachims et al., 2005; Robertson et al., 1996; Salton & Buckley, 1990) has been a major research area in IR for a long time as an attempt to incorporate users’ information needs indirectly with local relevance judgments. In general, the contents of relevant and non-relevant documents retrieved by a query.

These approaches focus on improving query quality based on local feedback information, i.e., the retrieved documents for the query at hand, whereas the proposed work in this paper attempts to gather evidence for the overall importance, like IDF, of a term in a document from an accumulation of relevance judgments. Given that the evidential weight of a term is used to measure its overall importance across queries, it is proposed as an auxiliary factor working together with the well-known term weighting method as in formula (2), where it complements the strictly local information (TF) and generic document discrimination information (DF). Instead of requiring relevance feedback information for the current query, which may not be available, it utilizes relevance information that can be collected over many past queries.

Recent work on Learning to Rank (see Richardson et al. (2006), Duh (2009), Yeh et al. (2007) and Cao et al. (2007) for example) also uses accumulated relevance information. Given a set of queries and their relevant documents, its goal is to automatically learn an optimal ranking function of a retrieval engine through various kinds of machine learning approaches. From the machine learning perspective, the proposed approach is to learn the empirical importance of individual terms based on their past roles in separating relevant from non-relevant documents. That is, while the Learning to Rank approach finds one best ranking function from the entire training data, ours determines goodness of the entire term separately.

3. Discrimination power

The premise of the notion of evidential weights is that importance of a term has something to do with its roles in the past queries in which it appeared. A query term in a retrieved document is supposed to play an important role on the relevance of
the document. But its general impact on relevance of any document, or importance, must depend not only on its degree of ambiguity but also the actual user information need, the collection being searched, time-sensitive trends, etc. While there are many factors that help a term discriminate relevant from irrelevant documents, we conjecture that a term’s overall discrimination power can be obtained from its role in the past queries. In other words, we attempt to compute a term’s discriminating power by accumulating its roles in separating relevant from non-relevant documents over many queries in the past.

Given a set of queries and top-ranked (for example, hundred), retrieved documents per query, which can be divided into two subsets of relevant and non-relevant documents, we can obtain two distributions for a term with its frequency (or weight) in relevant and non-relevant documents, respectively. Fig. 1 shows actual histograms of two subsets of relevant and non-relevant documents with respect to a term ‘nation’. The weights shown in Fig. 1 are normalized in the range of [0, 1] across queries.

Given a pair of curves for a term in a query, which represent distributions of term frequencies for relevant documents and non-relevant documents, respectively, we can take a sum of all the statistics for a term in the past queries containing the term to result in a graph like the one in Fig. 2. The difference between the two averages representing the sets of term frequencies obtained from relevant and non-relevant documents can be regarded as a discrimination power (DP) of the term. The idea of comparing two distributions has been shown to be useful in past research. For example, Robertson’s work utilized the difference of two distributions of matching function values of both relevant and irrelevant documents in order to enhance query expansion performance (Robertson, 1990). The distributions represent probability density of matching function values, not individual term weights or frequencies as in our case, and the method proposed there is to estimate the effect of adding a new term on the mean difference of the two distributions for a specific query. Our approach, on the other hand, attempts to measure the overall quality of a term by comparing two distributions of a term’s weight in both relevant and irrelevant documents accumulated over past queries.

The evidential weight $w_k$ of a term can be obtained by averaging the DP values from all the past queries. In estimating DP of a term, we consider the ratio, instead of difference, of the means of the term frequency distributions of relevant and non-relevant documents, so that it serves as a multiplier in formula (2) with its value around 1.0. As a result, an initial DP value for a term $t_k$ contained in a given query $q_i$ can be computed as follows:

$$DP_{init}(t_k|q_i) = \left( \frac{1}{u_q} \sum_{d_r \in \text{Rel}} w(t_k|d_r,q_i) / \frac{1}{v_q} \sum_{d_i \in \text{Nonrel}} w(t_k|d_i,q_i) \right)$$

(3)

where $0 < w(t_k|d_i,q_i) < 1$ denotes a weight (e.g. normalized TF or TF-IDF) of the term $t_k (1 \leq k \leq \# \text{ of unique terms in a collection})$ in the relevant (or non-relevant) document $d_i$ (or $d_r$) retrieved by the $i$th (1 $\leq i \leq \# \text{ of queries}$ query $q_i$, $u_i$ and $v_i$ are the numbers of relevant (Rel) and non-relevant (Nonrel) documents in the search results ($|\text{Rel}| + |\text{Nonrel}| = \# \text{ of search results}$) retrieved by the $i$th query $q_i$, respectively, the range of $DP_{init}$ is greater than zero and less than positive $\infty$. As a way to control the value, we apply the following based on sigmoid function to the $DP_{init}$ resulting in the range of (0, 2).

$$DP_{sigmoid}(t_k/q_i) = 2\left[ \frac{1}{1 + e^{-3(DP_{init}(t_k|q_i) - 1)}} \right]$$

$DP_{init}$ for a term not occurring in either relevant or non-relevant documents is not calculated but assigned 1.0 as a default value. The reason is related to how a final DP score is used for final term weighting. As in (2), the DP score is used as an additional evidence to the term weight calculated in a conventional way (e.g. TF-IDF); it is multiplied to adjust the TF-IDF weight. Since DP is an accumulated statistical value obtained from a set of past retrieval results, no change should be made to the

![Fig. 1. Histograms of weights of a word ‘nation’. Left and right graphs show the frequency-based weight histograms for relevant and non-relevant documents, respectively.](image)
overall term weighting in case of no known data so that the original weight is preserved. Using the default value 1.0 will not affect the term's weight one way or the other. By the same token, the default value is assigned to a term when there is no difference in the accumulated evidence between relevant and irrelevant sets. Such terms are construed to be 'neutral' rather than 'useless' in $DP$ calculated from the past retrieval results so that other indicators for their importance should be respected as they are. While such terms may be seen as useless in discriminating relevant from irrelevant documents when the distributions are for a specific query as in Robertson (1990), we take a conservative position because the past query statistics may not be always reliable. In sum, a term having $OPT_{rank}$ values less than 1.0 can be seen as less likely to retrieve relevant documents than those with $DP$ values greater than 1.0, and hence its final weight should be lowered.

A weakness of $DP_{sigmoid}$, however, is that it does not consider how the relevant and non-relevant documents were ranked according to the given query $q_i$. An ideal query should retrieve all the relevant documents before any non-relevant documents are retrieved. Furthermore, such a query should ensure that the difference between the similarity values of relevant documents and those of non-relevant ones be as large as possible. Instead of trying to devise a better ranking function (Cao et al., 2007; Duh, 2009; Robertson, 1990; Yeh et al., 2007), we attempt to judge the quality of a query including a particular term $t_k$ from the ranks of the retrieved results so that other indicators for their importance should be re-

To define rank optimality of a query, suppose there are $m$ ranked documents including $u$ relevant and $v$ non-relevant ones ($m = u + v$), each of which contains the term, in the retrieval result list for a query. In an ideal case, all the $u$ relevant documents should be ranked (positioned) first followed by the $v$ non-relevant ones. Given that there is no information about the degree of relevance in the judgments, we enforce that in an ideal situation, all the relevant documents be ranked at the first position and all the non-relevant ones at the last position. We define rank optimality $OPT_{rank}(t_k, q_i)$, as the rank ratio of relevant over non-relevant documents, for a query $q_i$ containing a particular term $t_k$:

$$OPT_{rank}(t_k, q_i) = \sqrt{\frac{\sum_{d_i \in \text{Rel}} Rank_{normalized}(d_i | t_k, q_i)}{\sum_{d_i \in \text{Nonrel}} Rank_{normalized}(d_i | t_k, q_i)}} \times \frac{\text{freq}(t_k)}{|q_i|}$$

where $\text{Rank}_{normalized}(d | t_k, q_i)$ stands for the rank of document $d$ in the retrieved results by the query $q_i$ containing term $t_k$. $0 \leq \text{Rank}_{normalized}(d_i | t_k, q_i) \leq 1$ is the normalized rank of a relevant (or non-relevant) document containing a term $t_k$ for query $q_i$. $u$ and $v$ are the numbers of relevant and non-relevant documents, respectively. $M$ is the number of the retrieved documents or the maximum number of rank, $\text{freq}(t_k)$ is the frequency of $t_k$ in the query $q_i$, and $|q_i|$ is the number of terms occurring in the query $q_i$. $\text{freq}(t_k)$ stands for relative importance of term $t_k$ in the query $q_i$ since the retrieved results are produced by the entire terms in the query $q_i$. We applied square root, $\sqrt{\text{rt}}$, to the original rank ratio because the ratio drastically changes and the ratio with $\sqrt{\text{rt}}$ performed well empirically.

Alternatively, we can define similarity optimality $OPT_{sim}(t_k, q_i)$, in place of rank similarity, of ranked documents for a given query containing a particular term $t_k$: 

$$OPT_{sim}(t_k, q_i) = \sqrt{\frac{\sum_{d_i \in \text{Rel}} \text{Sim}_{normalized}(d_i | t_k, q_i)}{\sum_{d_i \in \text{Nonrel}} \text{Sim}_{normalized}(d_i | t_k, q_i)}} \times \frac{\text{freq}(t_k)}{|q_i|}$$
\[
\text{Sim}_{\text{normalized}}(d|q_i) = \left( \frac{\text{Sim}(d|t_k, q_i) - \text{MinSim}}{\text{MaxSim} - \text{MinSim}} \right)
\]

where \(\text{Sim}(d|t_k, q_i)\) stands for the similarity value of document \(d\) in the retrieved results by the query \(q_i\) containing term \(t_k\), \(0 \leq \text{Sim}_{\text{normalized}}(d|t_k, q_i) \leq 1\) is the normalized similarity of a relevant (or non-relevant) document containing a term \(t_k\) for \(q_i\). \(u_i\) and \(v_i\) are the numbers of relevant and non-relevant documents, respectively, \(\text{MaxSim}\) and \(\text{MinSim}\) are maximum and minimum similarity scores respectively, \(\text{freq}(t_k)\) is the number of \(t_k\) in the query \(q_i\), and \(|q_i|\) is the number of all terms in the query \(q_i\). \(\frac{\text{freq}(t_k)}{|q_i|}\) stands for relative importance of term \(t_k\) in the query \(q_i\). The value of \(\text{OPTsim}\) is maximized when all the \(u\) relevant documents are ranked (positioned) first followed by the \(v\) non-relevant ones based on the decreasing order of the similarity values.

From Eq. (4), now a \(DP\) value for a term can be computed as:

\[
\text{DP}(t_k|q_i) = \text{DP}_{\text{sigmoid}}(t_k|q_i) \times \sqrt{\frac{\frac{1}{u_i} \sum_{d \in \text{Rel}} \text{Rank}_{\text{normalized}}(d|t_k, q_i)}{\frac{1}{v_i} \sum_{d \in \text{Nonrel}} \text{Rank}_{\text{normalized}}(d|t_k, q_i)}} \times \frac{\text{freq}(t_k)}{|q_i|}
\]

Alternatively, it can be computed using \(\text{OPTsim}\) as follows

\[
\text{DP}(t_k|q_i) = \text{DP}_{\text{sigmoid}}(t_k|q_i) \times \sqrt{\frac{\frac{1}{u_i} \sum_{d \in \text{Rel}} \text{Sim}_{\text{normalized}}(d|t_k, q_i)}{\frac{1}{v_i} \sum_{d \in \text{Nonrel}} \text{Sim}_{\text{normalized}}(d|t_k, q_i)}} \times \frac{\text{freq}(t_k)}{|q_i|}
\]

To compute \(W_k\) for a query term, we take an average of \(\text{DP}(t_k|q_i)\) values over all the queries containing the term as follows:

\[
W_k = \text{DP}(t_k) = \frac{\sum_{i=1}^{n_q} \text{DP}(t_k|q_i)}{n_q} \times A
\]

where \(n_q\) is the number of queries containing term \(t_k\), and \(A\) is an amplifier to control the range of the values when it is used together with the local and global weights.

4. Experiments

To demonstrate the value of the proposed term-weighting method utilizing evidential weights based on term’s \(DP\) values, we ran experiments using two systems, Terrier\(^1\) and Indri\(^2\) with and without applying the evidential weight component in computing term weights. We chose them because they are well-known open source search engines that have been validated (Jadh Ounis et al., 2005). The \(DP\) weighting is applied to the ranking procedure by replacing their original weighting functions as in Eq. (1) into the proposed one in Eq. (2).

4.1. Experimental setup and data sets

Terrier is a highly flexible, efficient, and effective open source search engine, readily deployable on large-scale collections of documents. It provides several state-of-the-art document ranking models including DFR_BM25 (Divergence From Randomness) (Amati, 2003), \(TF*IDF\), language modeling approach, etc. In addition, it supports a number of parameter-free DFR term weighting models for automatic query expansion plus Rocchio’s query expansion.

We modified Terrier’s term weighting module so as to apply the proposed \(DP\) based method and show its impact. In particular, \(TF*IDF\) and DFR_BM25 for probabilistic models, and Hiemstra language modeling method (Hiemstra, 2001) were modified in the experiment. The \(DP\) values were applied according to the Eq. (2) in this model.

We also used Indri, one of the most well-known language modeling-based search engines, which combines the language modeling and Inference Network approaches to retrieval (Metzler & Croft, 2004). Since it has shown excellent results in the TREC conference, we decided to use it as a variation of a language modeling-based search engine, in addition to the Hiemstra language model. The \(DP\)-based scheme is employed in such a way that it estimates the term’s (called node in Inference Network) belief score. It is calculated as follows:

\[
\text{Belief}(t_k) = \delta + (1 - \delta) \times \text{TF}(t_k) \times \text{IDF}(t_k)
\]

where \(\delta\) is a default belief that ensures term belief values to be above a certain value.

\(DP\) is computed as in the following equation if \(t_k\) has been found in the collection of past queries. Otherwise, a default \(DP\) value 1.0 is given.

\[
\text{Belief}(t_k) = \delta + (1 - \delta) \times \text{TF}(t_k) \times \text{IDF}(t_k) \times \text{DP}(t_k)
\]

In order for the experimental results to be comparable with previous performance values, we used three TREC collections: TREC-3, TREC-4, and TREC-5. Table 1 shows the basic statistics about them.

4.2. Training/testing procedure

The ranking procedure consists of four basic steps as depicted in Fig. 3.

– Factoring a query into a set of terms (or phrases).
– Calculating individual term’s weight in a document.
– Merging term weights based on the query structure.
– Ranking documents based on the merged weight.

In order to apply DP values to the original weight calculation, we need to train a term’s DP value from the past retrieved results with relevance judgments. All the three collections were used for both training and testing. Since each collection had to be used for both computing DP scores and conducting retrieval experiments, the leave-one-out method was used. That is, after computing DP scores for terms using $q - 1$ out of $q$ queries, the remaining query was used for testing. This process was repeated $q$ times.

At the training phase, we gathered three additional pieces of information: individual term’s weight at step 5, retrieved results including rank and similarity scores at step 6, and relevance judgments at step 7. We calculated DP values for individual query terms using the two values according to the Eq.(10) (or Eqs. (9)) and (11). At the testing phase, the trained DP values are multiplied to the original weighting scheme. As mentioned earlier, the default DP for terms having no DP score, i.e. no previous history, is 1.0.

In general, an individual term’s weight in a document is produced based on different kinds of weighting schemes such as TF*IDF, Okapi BM25, etc. The weight of a term $t_i$ on a relevant document can be conceived as $p(t_i|r)$, the probability score given that the document is relevant. In the same way, the weight on a non-relevant document can be treated as $p(t_i|r')$. As a result, we can estimate the basic ratio $p(t_i|r)/p(t_i|r')$ to calculate the DP-based weight.

4.3. Coping with limited training set

DP values are obtained from terms’ weights included in the given queries, not in the entire document collection. It means that the number of terms for which a DP value (henceforth DP terms) is available depends on the number of unique terms in the query sets. In reality, however, the number of query terms in a collection may be quite small compared to that of the terms in the collection-wide term dictionary. As a result, a query for which none of the terms appear in the past queries
is not influenced at all by the DP-based weighting scheme. In fact, this limitation of the training set is more severe with the experimental environment because only three collections are used.

In order to cope with this limitation, especially for the experiments, we employed pseudo relevance feedback, an automatic query expansion method. The DP values were computed for not only the original query terms but also expanded terms. This way, the DP-based weighting scheme was applied more widely. The two search engines, Terrier and Indri, provide well-known pseudo relevance feedback mechanisms. The former supports a number of parameter-free DFR term weighting models for automatic query expansion, in addition to Rocchio’s query expansion. The latter, on the other hand, provides automatic pseudo relevance feedback mechanism, an adaptation of Lavrenko’s relevance models (Lavrenko & Croft, 2001).

Table 2 shows the numbers of DP terms for different weighting models and test collections. In TF/IDF search, for example, the number of DP terms is at most 712 (TREC-3) and increased to 1249 with query expansion. The last row shows the minimum and maximum percentages of DP terms for the collections, which are not high.

5. Results

5.1. Rank vs. similarity based DP

Since we designed two optimality calculation methods for DP in Eqs. (5) and (7), the first task was to compare them with TF/IDF weighting in the three collections. The result was that the similarity optimality based DP method is better without exception as shown in Table 3 although the paired t-test of the difference between Rank Optimality and Similarity Optimality was not statistically significant at 5% significance level. One thing focused on in the result table is that applying DPinit alone showed unsatisfactory performance. It is because DPinit and similarity (or rank) optimality do not work independently. Based on the result in Table 3, the subsequent experiments were conducted with the similarity optimality based DP method.

5.2. DP for language modeling approach

To show the effect of applying the DP-based term weighting scheme to the language modeling method, we compared two language modeling-based search engines: the Hiemstra language model implemented in the Terrier engine and the Indri language model in the Indri engine. Table 4 shows the results of the experiments performed on each TREC collection. There are two baselines: ‘Baseline’ refers to the basic search while ‘Baseline w/QE’ means that the search was performed with query expansion using pseudo relevance feedback (PRF). The experiment with ‘Baseline w/QE’ shows the contribution made by DP scores over and above the query-specific expansion (PRF in this case). We use the default values for the system parameters. The performance values are all in Mean Average Precision (MAP), and numbers in parentheses are percentage increase over the baseline results.

While the magnitudes of the improvements are not significantly high except for one case (TREC-3/HLM), the results are consistent without any performance decrease with or without query expansion across all the collections. Given that the number of queries in the collections and hence the training set are relatively small, we consider this is very promising. The biggest improvement (20.3%) for TREC-5/HLM is due to the lowered performance of the baseline when query expansion was used, not entirely due to the DP-based scheme.

The effect of using query expansion is interesting by itself in that we still obtained improvement over the performances of “baseline w/QE”, which resulted in significant improvements over the “baseline”. While not consistent, the percentage increases are bigger for some cases. It at least partially coincides with the conjecture that the increased coverage of training query set would increase the positive effect of the DP-based weighting scheme.

5.3. DP for probabilistic models

To show the effect of DP-based weighting scheme on the probabilistic models, we made comparisons with two probabilistic search methods: TFIDF and DFR_BM25 in the Terrier engine. Table 5 shows the results of the experiments performed with the two search engines on each TREC collection.

Table 2
The numbers of applied DP terms.

<table>
<thead>
<tr>
<th>Model</th>
<th>TREC-3</th>
<th>TREC-4</th>
<th>TREC-5</th>
</tr>
</thead>
<tbody>
<tr>
<td>TFIDF</td>
<td>712</td>
<td>377</td>
<td>406</td>
</tr>
<tr>
<td>TFIDF w/QE</td>
<td>1249</td>
<td>729</td>
<td>768</td>
</tr>
<tr>
<td>DFR_BM25</td>
<td>713</td>
<td>385</td>
<td>389</td>
</tr>
<tr>
<td>DFR_BM25 w/QE</td>
<td>1195</td>
<td>663</td>
<td>748</td>
</tr>
<tr>
<td>Hiemstra_LM</td>
<td>672</td>
<td>378</td>
<td>369</td>
</tr>
<tr>
<td>Hiemstra_LM w/QE</td>
<td>1193</td>
<td>703</td>
<td>717</td>
</tr>
<tr>
<td>Indri LM</td>
<td>426</td>
<td>381</td>
<td>402</td>
</tr>
<tr>
<td>Indri LM w/QE</td>
<td>933</td>
<td>730</td>
<td>780</td>
</tr>
<tr>
<td>% of DP terms (Min/Max)</td>
<td>0.07/0.21</td>
<td>0.06/0.12</td>
<td>0.05/0.10</td>
</tr>
</tbody>
</table>
The overall trend is similar to that in Table 4 although the magnitudes of the improvements are a bit smaller. One interesting observation across the two results is that the effect of \( DP \)-based scheme is collection-dependent. The effect of query expansion is smaller for TREC-3 across all different retrieval models whereas it is increased for TREC-4 after expansion. It stays almost the same for TREC-5. A thorough investigation on the effect of test collections including query set sizes is left for future research.

5.4. Effect of the number of \( DP \) terms

We investigated how MAP varies as the number of \( DP \) terms increases. Fig. 4 shows that the MAP (solid line) tends to slowly increase as the number of \( DP \) terms (dotted line) increases. It shows one of the three experimental results we have done using TREC collections; the others also show a similar shape of graphs. It indicates that it’s likely to see better performance if we gather more \( DP \) terms from the previous search results. It should be noted that the maximum number of \( DP \) terms is only 1249 in Table 2.

5.5. \( DP \) for web blogs

The experiments mentioned above show usefulness of the \( DP \)-based weighting scheme when the three TREC collections were used. Noting that the TREC collections we used are different in their nature with the web search environment, we ran experiments with ClueWeb-09-T09B, the TREC 2009 Blog Track collection, which consists of 50 million documents (size of documents: 250 GB compressed). The experiment used 592 queries (# of average terms: 2.58) containing relevant documents.

We expect that the proposed method would be most useful with Web search when click-through data is available. In general, user click-through data can be extracted from a large amount of search logs accumulated by web search engines. These logs typically contain user-submitted search queries and the URL for user-clicked Web pages available in the search result pages. Although these click-through data do not necessarily reflect the exact relevance information, they provide indications

\[ \text{Table 3} \]
Comparison of MAP between rank based \( DP \) and similarity based \( DP \).

<table>
<thead>
<tr>
<th></th>
<th>TREC-3</th>
<th>TREC-4</th>
<th>TREC-5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.2906</td>
<td>0.2053</td>
<td>0.1669</td>
</tr>
<tr>
<td>( DP ) only</td>
<td>0.2917</td>
<td>0.2040</td>
<td>0.1693</td>
</tr>
<tr>
<td>( DP )-rank optimality</td>
<td>0.3049</td>
<td>0.2073</td>
<td>0.1741</td>
</tr>
<tr>
<td>( DP )-similarity optimality</td>
<td>0.3054</td>
<td>0.2089</td>
<td>0.1745</td>
</tr>
</tbody>
</table>

\[ \text{Table 4} \]
Experimental results with language modeling.

<table>
<thead>
<tr>
<th>TREC</th>
<th>Model</th>
<th>Baseline</th>
<th>( DP ) applied</th>
<th>Baseline w/QE</th>
<th>( DP ) applied</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>HLM</td>
<td>0.2003</td>
<td>0.2181(8.9)( ^i )</td>
<td>0.2365</td>
<td>0.2472(4.5)( ^i )</td>
</tr>
<tr>
<td></td>
<td>ILM</td>
<td>0.2654</td>
<td>0.2679(0.9)</td>
<td>0.3164</td>
<td>0.3224(1.9)</td>
</tr>
<tr>
<td>4</td>
<td>HLM</td>
<td>0.1926</td>
<td>0.1939(0.7)</td>
<td>0.2083</td>
<td>0.2146(3.0)( ^i )</td>
</tr>
<tr>
<td></td>
<td>ILM</td>
<td>0.1760</td>
<td>0.1875(6.5)( ^i )</td>
<td>0.2019</td>
<td>0.2102(4.1)( ^i )</td>
</tr>
<tr>
<td>5</td>
<td>HLM</td>
<td>0.1454</td>
<td>0.1561(7.4)( ^i )</td>
<td>0.1315</td>
<td>0.1582(20.3)( ^i )</td>
</tr>
<tr>
<td></td>
<td>ILM</td>
<td>0.1969</td>
<td>0.2056(4.4)( ^i )</td>
<td>0.1822</td>
<td>0.1968(4.6)</td>
</tr>
</tbody>
</table>

\( ^i \) The increase is statistically significant at the 5% significance level using paired t-test.

\[ \text{Table 5} \]
Experimental results with probabilistic models.

<table>
<thead>
<tr>
<th>TREC</th>
<th>Model</th>
<th>Baseline</th>
<th>( DP ) applied</th>
<th>Baseline w/QE</th>
<th>( DP ) applied</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>TFIDF</td>
<td>0.2906</td>
<td>0.3046(4.8)( ^i )</td>
<td>0.3399</td>
<td>0.3476(2.3)</td>
</tr>
<tr>
<td></td>
<td>DFR</td>
<td>0.2950</td>
<td>0.3068(4.0)( ^i )</td>
<td>0.3426</td>
<td>0.3448(0.6)</td>
</tr>
<tr>
<td>4</td>
<td>TFIDF</td>
<td>0.2053</td>
<td>0.2083(1.5)</td>
<td>0.2497</td>
<td>0.2575(3.0)( ^i )</td>
</tr>
<tr>
<td></td>
<td>DFR</td>
<td>0.2088</td>
<td>0.2132(2.1)( ^i )</td>
<td>0.2515</td>
<td>0.2640(5.0)( ^i )</td>
</tr>
<tr>
<td>5</td>
<td>TFIDF</td>
<td>0.1669</td>
<td>0.1741(4.3)( ^i )</td>
<td>0.1910</td>
<td>0.2034(6.5)( ^i )</td>
</tr>
<tr>
<td></td>
<td>DFR</td>
<td>0.1696</td>
<td>0.1754(3.4)( ^i )</td>
<td>0.2515</td>
<td>0.2655(5.6)( ^i )</td>
</tr>
</tbody>
</table>

\( ^i \) The increase is statistically significant at the 5% significance level using paired t-test.

5.4. Effect of the number of \( DP \) terms

We investigated how MAP varies as the number of \( DP \) terms increases. Fig. 4 shows that the MAP (solid line) tends to slowly increase as the number of \( DP \) terms (dotted line) increases. It shows one of the three experimental results we have done using TREC collections; the others also show a similar shape of graphs. It indicates that it’s likely to see better performance if we gather more \( DP \) terms from the previous search results. It should be noted that the maximum number of \( DP \) terms is only 1249 in Table 2.

5.5. \( DP \) for web blogs

The experiments mentioned above show usefulness of the \( DP \)-based weighting scheme when the three TREC collections were used. Noting that the TREC collections we used are different in their nature with the web search environment, we ran experiments with ClueWeb-09-T09B, the TREC 2009 Blog Track collection, which consists of 50 million documents (size of documents: 250 GB compressed). The experiment used 592 queries (# of average terms: 2.58) containing relevant documents.

We expect that the proposed method would be most useful with Web search when click-through data is available. In general, user click-through data can be extracted from a large amount of search logs accumulated by web search engines. These logs typically contain user-submitted search queries and the URL for user-clicked Web pages available in the search result pages. Although these click-through data do not necessarily reflect the exact relevance information, they provide indications

---

3 ClueWeb09 Dataset, http://www.boston.lti.cs.cmu.edu/Data/clueweb09/.
of the users’ intention by associating a set of query terms with a set of web pages (Joachims, 2003; Xue et al., 2004). Since user queries in Web search tend to follow Power Law distribution (Lempel & Moran, 2003), DP values can reflect which terms were heavily used for a certain time period and which were found frequency in clicked pages. While this calls for an interesting experiment, such data is not easily available outside the small community of Web portal companies. ClueWeb-09 was the best possible resource available to us to get close to the Web search situation.

In order to make our experiment possible, we had to use only 10% of 50 million documents in the collection; dealing with the entire collection in our experimental setting was prohibitively expensive. The subset consists of randomly selected 10% of non-relevant documents and entire relevant documents. The experiment is applied to 592 queries that have relevant documents. The ranking algorithms used were TFIDF, DFR_BM25, and Hiemstra Language Model in Terrier. Since 90% of the non-relevant documents were removed, the retrieval task became easier than the original one. As such, the effect of the DP values was expected to be smaller.

Table 6 shows the results of the experiments. The Base MAP indicates the MAP produced by the original ranking algorithms, while DP Applied MAP indicates the performance of the proposed DP applied search. The performance increase in the parentheses is promising because the number of applied DP terms per query is 1.31 on an average, which is smaller than that of the previous results on TREC 3–5. The total number of DP terms in each experiment was about 1000, which is a very small portion of the vocabulary size 17 M in the collection.

5.6. Parameter selection

The DP Eq. (11) has an important parameter A, which is an amplifier to control the range of the values when it is used together with the local and global weights. The parameter value, 0.8, was used for the experiments based on a preliminary experiment whose purpose was to investigate its effect on MAP variations for all the collections and retrieval models. Since the results show similar trends, only the result of the Hiemstra language model search is depicted in Fig. 5 as a representative. As mentioned earlier, A is an amplifier to control the range of the DP when it is used together with the local and global weights. Consequently, the value 0.8 is actually narrowing the DP range.

5.7. DP vs. IDF

DP and IDF share a common characteristic: they are term properties that are computed globally from a corpus or relevance judgments and intended to help separating relevant from non-relevant documents. However, they are distinct from each other in their statistical characteristics. In addition to the retrieval performance improvements, which show that DP seems to play a unique role beyond IDF, their values are distinct from each other as Table 7 shows examples of query terms and their IDF and DP value pairs extracted from TREC-3.

Besides the anecdotal evidence, the Pearson correlation measure (Rodgers & Nicewander, 1988) between them, 0.03, indicates their values are not correlated. For an intuitive understanding, a scatter plot between DP and IDF is shown in Fig. 6 where the straight line is a result of linear regression.

Table 6
Experimental results on web blogs.

<table>
<thead>
<tr>
<th></th>
<th>tfidf</th>
<th>DFR_BM25</th>
<th>HLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base MAP</td>
<td>0.4301</td>
<td>0.4750</td>
<td>0.3811</td>
</tr>
<tr>
<td>DP Applied MAP</td>
<td>0.4381 (1.9%)</td>
<td>0.4849 (2.1%)</td>
<td>0.3889 (2.0%)</td>
</tr>
<tr>
<td># of DP terms</td>
<td>1004</td>
<td>1098</td>
<td>1004</td>
</tr>
<tr>
<td>Average # of applied DP terms per query</td>
<td>1.31</td>
<td>1.30</td>
<td>1.31</td>
</tr>
</tbody>
</table>

DFR_BM25 and HLM stand for divergence from randomness BM25 and Hiemstra language model, respectively.
5.8. Storage and efficiency

A search engine with the proposed approach requires more space to store additional data as well as more time to calculate the additional weights of terms. For storage, the amount of additional storage for $DP$ scores is at most proportional to the number of unique terms in the search engine index Database. That is, the necessary increase in space is $O(n)$ where $n$ is the number of index terms. Considering that the size of the dictionary part is usually less than 1% of the total storage required for an index in a search engine (Witten, Moffat, & Bell, 1994), the increase is almost negligible. However, the proposed scheme
requires a history of users’ relevance judgments be kept. In a contemporary search engine, it amounts to storing click-through data that have been shown very important for various schemes for query understanding (Joachims, 2003, 2005; Xue et al., 2004). For efficiency, computing DP scores in addition to the traditional weighting scheme (e.g. \(TF \cdot IDF\)) is just to multiply a constant value when a term’s weight is computed as in Eq. (2). Since \(DP(t)\) is computed once for a term like \(IDF(t)\) when the index is constructed, the online time required for query process is negligible. However, \(DP(t)\) values need to be computed periodically offline, like \(IDF(t)\), as the user relevance judgments data is updated.

6. Conclusion

Term weighting schemes are crucial for document retrieval effectiveness in that they are the core of document weighting and ranking. We propose a novel term weighting method that utilizes availability of past retrieval results containing relevance judgments. A term’s evidential weight depends on the degree to which the mean frequency values for the relevant and non-relevant document distributions are different. It also takes into account the rankings or similarity values of the documents. In a sense, it has the effect of accumulating local relevance feedback information across many queries to determine its global weight. Added to the local relevance feedback is the similarity or ranking information of individual retrieved documents that contain the term at hand.

The experiments using standard test collections show that the proposed weighting scheme indeed improves retrieval effectiveness. It is interesting to note that we obtained the performance increase with only a small number of terms found in the relatively small number of past queries. Further analysis shows the notion of evidential weight, not based on the entire collection but based on the relevance-judged documents, is clearly distinct from \(IDF\).

The new weighting scheme opens up new research directions. First of all, we will have to investigate on different statistical properties of relevance-judged documents to refine the evidential weight calculation method. Second, we need to explore further the effect of the size of the relevance-judged document collection on retrieval effectiveness. Third, an interesting and promising area is to apply this scheme to Web search, especially attempting to compute evidential weights in a time window so that term weights vary with time. Finally, we will explore how best evidential weights can be mixed with other local and global weights by understanding the relationships among them.

Acknowledgements

This research was supported by WCU (World Class University) program under the National Research Foundation of Korea and funded by the Ministry of Education, Science and Technology of Korea (Project No.: R31-30007).

References


