Simple BM25 Extension to Multiple Weighted Fields

Stephen Robertson, Hugo Zaragoza and Michael Taylor
Microsoft Research
Cambridge, U.K.
{ser,hugoz,mitaylor}@microsoft.com

ABSTRACT

This paper describes a simple way of adapting the BM25 ranking formula to deal with structured documents. In the past it has been common to compute scores for the individual fields (e.g. title and body) independently and then combine these scores (typically linearly) to arrive at a final score for the document. We highlight how this approach can lead to poor performance by breaking the carefully constructed non-linear saturation of term frequency in the BM25 function. We propose a much more intuitive alternative which weights term frequencies *before* the non-linear term frequency saturation function is applied. In this scheme, a structured document with a title weight of two is mapped to an unstructured document with the title content repeated twice. This more verbose unstructured document is then ranked in the usual way. We demonstrate the advantages of this method with experiments on Reuters Vol1 and the TREC dotGov collection.

Categories and Subject Descriptors

H.3.3 [**Information storage and retrieval**]: Information search and retrieval—*Retrieval models*

General Terms

Experimentation, Theory

1. INTRODUCTION

Textual data is most often found in a structured form; for example, documents are often subdivided into fields such as title, author, abstract, body, etc. Practitioners have found it beneficial (sometimes crucial) to exploit the document's internal structure to improve retrieval performance. Although there has been a number of IR frameworks proposed for this, their complexity and radical departure from standard ranking algorithms renders their application difficult. In practice, many systems exploit structure in an ad-hoc manner, by implementing a *linear combination of the scores* obtained from scoring every field. This however can be quite dangerous for several reasons, the most important one being the nonlinear treatment of term frequencies by ranking functions.

In this paper we discuss these dangers in detail and propose a very simple solution to extend standard ranking functions to multiple weighted fields. The idea is to compute a single score for a *linear combination of term frequencies*, instead of combining the scores. The approach is so simple that it may not merit a paper on the subject; however, we were encouraged to write it after seeing many papers following the more dangerous linear combination of the scores.

The proposed approach has many advantages, from the point of view of simplicity, interpretation, speed of computation and reduction of the number of parameters; furthermore it yields higher performance for the tasks considered in this paper.

In the following section we briefly review prior art. In section 2 we discuss in some detail the dangers of using a linear combination of scores. Section 3 describes our proposed method. In section 4, we discuss the problem of repeatable fields. Section 5 reports on some experiments with the two methods in two test collections.

1.1 Prior Art

Much work on structured document retrieval deals with the combination of field scores, decoupling the problem from the scoring function itself. Much of the discussion about combining scores for the same document has taken place in the context of meta-search systems, where the information available to the combining engine may not include any details about exactly which query components contributed to which score. In these circumstances, some form of post-hoc combination of scores is required; this may be more complex than a simple linear combination. Ogilvie and Callan [15] give a good overview of the issues, and propose and test various combinations.

One of the earlier empirical studies of field weighting is the work of Wilkinson [19] where he evaluates different ways to weight and combine the scores obtained on the different fields of a document.

A few authors have developed more formal frameworks to combine information from structured documents in a principled manner. For example, Lalmas [5] exploits the theory of evidence as a framework for information aggregation. Myaeng [6] extended the InQuery model (an IR formalism based on Bayesian Networks) to incorporate structural information. More recently Piwowarski [16] has proposed the

CIKM'04, November 8–13, 2004, Washington, DC, USA.

use of Bayesian Networks for this purpose; his paper gives a good review of work on this area.

However for the most part ad-hoc retrieval systems apply standard (non-structured) ranking algorithms and tackle the problem of structure by combining in some way the scores obtained from the different fields, in a similar manner to Wilkinson. As examples, we could cite many systems developed for the TREC Web Track (see [2]), systems dealing with DBMS aspects of structured IR (see [4] and references therein) and systems for XML retrieval [1]. In the case of the TREC Web Track, each of the following papers includes at least one linear combination of scores obtained from different text fields: [7, 8, 9, 10, 11, 12, 13, 14].

The aim of this paper is not to propose a comprehensive approach to structured document retrieval, but rather to point out an elegant and simple alternative to score combinations.

Much closer to the spirit of this paper, Ogilvie and Callan [15] discuss (as an alternative to meta-search) combining document representations from different sources in the language model, comparing it to score combination with the probabilistic model. This is essentially what we do in this paper for a wider range of ranking algorithms, by concentrating on transforming the statistics of the document collection rather than re-parametrising a specific ranking function (which they do for the language model).

2. THE PROBLEM

In this section we describe the dangers we foresee when using combinations of scores. Section 2.1 describes the notation used in the paper and defines mathematically the problem being addressed. Section 2.2 describes the standard method of score combination, which is criticized in section 2.3.

2.1 Documents and Weighted Fields

Consider first an unstructured document $\bar{d}$ belonging to a collection C. We may regard this as a vector $\bar{d} = (d_1, ..., d_V)$, where d_j denotes the *term frequency* of the jth term in $\bar{d}$ and V is the total number of terms in the vocabulary.

In order to score such a document against a query, most ranking functions define a term weighting function $w_j(\bar{d}, C)$, which exploits term frequency as well as other factors such as the document's length and collection statistics. An example of such a function is BM25 [18], which will be used throughout this paper. For ad-hoc retrieval, and ignoring any repetition of terms in the query, this function can be simplified to:

$$w_j(\bar{d}, C) := \frac{(k_1+1)d_j}{k_1((1-b)+b\frac{dl}{avdl})+d_j} log \frac{N - df_j + 0.5}{df_j + 0.5} \quad (1)$$

where df_j is the document frequency of term j, dl is the document length, $avdl$ is the average document length across the collection, and k_1 and b are free parameters.

The document score is then obtained by adding the document term weights of terms matching the query q:

$$W(\bar{d}, q, C) = \sum_j w_j(\bar{d}, C) \cdot q_j \quad (2)$$

This general dot-product form covers many different ranking functions, including traditional *tf–idf*, most forms of cosine scoring, BM25, some forms of the language model, etc.

Consider now a collection with a set of field types $T = \{1, ..., f, ..., K\}$. For example $f = 1$ may denote Title, $f = 2$ Abstract, etc. For now we are dealing only with non-repeatable and non-hierarchical fields. Then a structured document $\mathbf{d}$ can be written as a vector of text-fields:

$$\mathbf{d} = (\bar{d}[1], \bar{d}[2], ..., \bar{d}[f], ...\bar{d}[K])$$

For example, $\bar{d}[1]$ would represent the title of document $\mathbf{d}$, $\bar{d}[2]$ the abstract, etc.

Each field $\bar{d}[f]$ may be seen as a vector of term frequencies $(d[f]_j)_{j=1..V}$ similarly to the unstructured document above. $\mathbf{d}$ is thus a matrix (a vector of vectors), and we note that any field may be empty for a particular document. We will refer to the collection of structured documents as $\mathbf{C}$. Finally, in order to weight fields differently, we define the field weight vector $\mathbf{v} \in \mathcal{R}^K$. For practical reasons indicated later, and without loss of generality, we choose to set one field weight (normally body text) to 1.

When scoring a structured document against a query, we now want to take into account not only its contents and the collection but also the field structure and the relative weight vector $\mathbf{v}$. Our problem is therefore the following:

- how to extend a standard ranking function $W(\bar{d}, q, C)$ into a new function $W(\mathbf{d}, q, \mathbf{C}, \mathbf{v})$.

We assume that query terms may match any number of fields – in other words, the words in the different fields may be drawn from the same vocabulary (even if their statistical characteristics are different). Thus for example titles and abstracts and body text may contain similar words, although the distribution of (say) stopwords in titles, or even what constitutes a stopword in the title context, may differ from that in body text.

2.2 Field Score Combination

How do we apply a ranking function such as (2), designed for unstructured documents, to structured ones?

A trivial way to proceed would be to merge all document fields into a non-structured form, by mapping documents as follows:

$$d := \bar{d}[1] + ... + \bar{d}[K]$$

but of course this would not achieve our aim of exploiting structure, nor could we apply the relative weightings $\mathbf{v}$.

Another approach is to treat each field type as a separate collection of (unstructured) documents. If we do so we can apply our standard ranking function (2) to each collection separately:

$$W(\bar{d}[f], q, C) = \sum_j w_j(\bar{d}[f], C) \cdot q_j \quad (3)$$

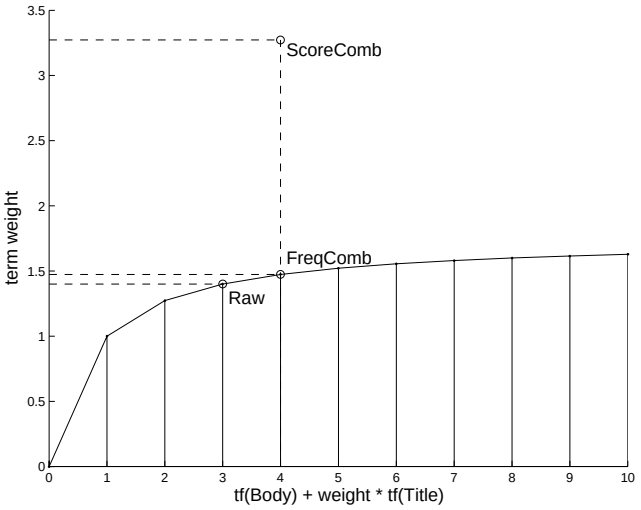
and then form a linear combination of these scores using the field weights:

$$W_1(\mathbf{d}, q, \mathbf{C}, \mathbf{v}) := \sum_{f=1}^{K} v_f \cdot W(\bar{d}[f], q, C) \quad (4)$$

This is what we refer to as a **(linear) combination of scores**, and it has been extensively used to score structured documents. We believe that this approach, while appealingly simple, presents some significant problems which we explore in the next section.

As discussed in the introduction, other approaches have been proposed, but tend to be considerably more complex.

Figure 1: *tf* component of term weight



For example, within the language modelling framework, it is possible to develop a separate language model for each field and then mix them [15]. Such methods may clearly have better formal foundation, but lack the simplicity of the linear combination of scores.

2.3 Issues with the linear combination of scores

Nonlinear TF

Most modern weighting functions based on term frequencies (*tf*) are nonlinear in this parameter. This is desirable because of the statistical dependence of term occurrences: the information gained on observing a term the first time is greater than the information gained on subsequently seeing the same term. In the BM25 formulation, the term weight saturates after a few occurrences, as shown by the curve in Figure 1

However, the linear combination of scores in (4) breaks this relation. For example, consider a document with a query word once in the title and twice in the body: the unstructured term weight would the point marked *Raw* in the Figure 1. Now suppose that we want to weight the title 2 and the body 1. This should boost the weight of this term somewhat; but the linear combination of scores would give us the much higher value *ScoreComb* in the Figure 1. Thus a document matching a single query term over several fields could score much higher than a document matching several query terms in one field only.

Choosing Collection Statistics

Many weighting functions make use of statistics taken from the whole collection; the obvious example is the document frequency of the term (number of documents in which it occurs), used in *idf* formulae. The obvious way to define such statistics for (3) is with reference to the specified field only. In other words, the document frequency of term j in (3) would be taken as the number of documents in which term j occurs in field f. If f is a short field (such as title) this statistic may be quite unstable. There seems no very natural way of sharing data across fields in the linear combination of scores method.

Equal field weights

We consider a degenerate case of field weighting. If we set all the weights v_f to one, we might reasonably expect to revert to the unstructured case (equivalent to merging all fields). However, this is not the case with a non-linear *tf* function and (4), since:

$$W(\bar{d}, q, C) \neq \sum_f W(\bar{d}[f], q, C)$$

Instead, we get a score that is very hard to interpret and that will no longer satisfy whatever good properties the original ranking function was designed for. For this reason, setting weights becomes a hard and counter-intuitive problem.

Document length

Another parameter that features in many weighting functions is document length. Again, if (3) is being used, the obvious way to calculate document length is with reference to the specified field only. The original argument for the form of document length normalisation used in BM25 [18] was based on the verbosity of the author and the scope of the document. In this case, it is not clear whether the same arguments would apply to the different fields, or whether the whole document length should be used.

Function parameters

Many weighting functions have tuning parameters which need to be set following some experimental evaluation. In the case of BM25 the two main parameters concerned are k_1 and b. k_1 controls the non-linear *tf* effect, while b controls the document length normalisation.

A method using (3) would require such tuning parameters to be set for every field. Probably any method using a vector $\mathbf{v}$ of field weights also requires these weights to be set empirically also ($K-1$ of them since one can be set arbitrarily to 1). Thus the total number of tuning parameters to be set in the case of BM25 is $2K + (K-1) = 3K-1$. This could lead to a very substantial number of combinatorial experiments.

3. PROPOSED METHOD: LINEAR COMBINATION OF TFS

After discussing the difficulties with score combination, we analyse the proposed alternative: term frequency combination.

Our intention is to modify standard ranking functions so that they may exploit multiple weighted fields, while satisfying the following requirements:

preserve term frequency non-linearity which has been repeatedly shown to improve retrieval performance.

give a simple interpretation to collection statistics and to document length incorporating field weights.

revert to the unstructured case when field weights are set to 1.

We propose to simply combine the term frequencies of the different fields by forming a linear combination weighted by the corresponding field weights:

$$\mathbf{d}' := \sum_{f=1}^{K} v_f \cdot \bar{d}[f] \tag{5}$$

and $\mathbf{C}'$ is the new collection of documents (note that $\mathbf{d}'$ and $\mathbf{C}'$ both depend on the particular field weight vector $\mathbf{v}$). We then score the documents using the resulting term frequencies:

$$W_2(\mathbf{d}, q, \mathbf{C}, \mathbf{v}) := W(\mathbf{d}', q, \mathbf{C}')$$

In this scenario, the term weighting and scoring functions are applied only once to each document.

Completing Figure 1, we mark by *FreqComb* the point resulting from the weighted sum of term frequencies ($2 + 2 * 1 = 4$). We can see that the resulting term weight is boosted slightly, while term dependence is maintained. The resulting boost is sufficiently small so that matching several terms remains more significant than matching the same term on several fields.

Note that this method is equivalent to mapping the structured collection into a new non-structured collection with modified term frequencies which combine the original term frequencies in the different fields, weighted. We can envisage this mapping as follows: suppose the two fields are title and body, and we wish to weight the title field by 5 (body weight 1), then we simply replace each document by the same document but with the title repeated 5 times, and then merged with the body. Because this new collection is not structured it can be ranked in the usual manner, and the resulting ranks will preserve the desired ranking properties on the original collection.

3.1 Theoretical basis and BM25

The following argument gives some justification for the proposed method as applied with the BM25 ranking function. As originally developed, BM25 is based on a 2-Poisson model of term frequencies in documents [18]. This can be seen as an elementary form of unigram language model, with the model parameter for a given term in any document depending on a single binary hidden variable known as 'eliteness'. Thus for each term, the collection of documents is split into two classes, elite and non-elite (hence the two Poisson distributions).

We can also look at this situation the other way round: for any given *document*, the *terms* are classified into elite and non-elite. The elite terms are those representing topics that the author wants to talk about. The simple language model assumes that each word-position in the document is filled independently with a term from the language, but that the individual term probabilities depend on whether the term is elite or not. We can see this as a general language model corresponding to non-eliteness for all terms, but with selected terms (the elite ones) having their occurrence probabilities boosted. (No attempt is made in this approach to normalise the term probabilities to ensure that they sum to one.)

Suppose now the author is writing a title or abstract. Because he has many fewer term-positions to fill in these fields than in the body, he will boost the probabilities of the elite terms even more. Thus each term occurrence in such a field can be taken as stronger evidence of the eliteness of that term in the document than an occurrence in the body. However, given many occurrences in the body, we are already fairly confident of eliteness, so the title occurrence should not add much more. (In relation to a given term, eliteness is a property of the document, not of the field.)

This argument provides a qualitative motivation for the proposed field-weighting method. Formally speaking, we might regard the extra boosting of elite term probabilities in such fields as title and abstract as a prior distribution on these quantities. As with the original BM25 development, we do not complete a formal development here, but the argument is sufficient to justify further experimental investigation.

3.2 Properties of the method

We may discuss the proposed linear combination of *tf*s, in respect of the issues raised above for the linear combination of scores.

nonlinear *tf* Since the nonlinear transformation is applied only once to each term in a document, it preserves the usual properties that it would have if we simply merged the fields.

collection statistics These are taken very simply, to agree with what would be obtained if we actually modified the documents in the way suggested. Thus the document frequency of a term is what it would be if we simply merged the fields.

equal field weights If we set all field weights to 1, then the new method reduces to exactly what we would get if we simply merged the fields. Thus its limiting behaviour is as one would expect. This is not quite the case if all field weights are set to a constant not equal to 1; however, in the case of BM25, this is equivalent to adjusting the k_1 tuning parameter.

document length There are various different ways of counting document length. The simplest is to count the number of words (tokens) in the document, considering only those words that are indexed. Thus the length of the document is the sum of the term frequencies. This definition applies naturally to the modified documents of $\mathbf{C}'$: we simply sum the modified term frequencies. Again, this is consistent with what would be obtained for the modified documents.

function parameters Here we have, in the case of BM25, a single pair of tuning parameters for the term weighting function, plus the field weights, giving $2+(K-1) = K+1$ parameters.

3.3 k_1 and *tf*

The k_1 parameter of BM25, as indicated, controls the nonlinear *tf* function. As our method changes the *tf*s substantially, we may also expect it to change the optimal value of k_1. We can get some idea of this effect from the following argument. If we were to use the linear combination of frequencies with all field weights the same but not equal to 1 ($v_f = v \neq 1$, equivalent to multiplying all *tf*s by v), then we could obtain exactly the same results as the unweighted case by also multiplying k_1 by v. It follows that optimal k_1 for this case is v times optimal k_1 for the unweighted case. Optimal b would be unchanged.

This suggests that we might look to average *tf* to guide us in how to change k_1. That is, we might optimise k_1 and b for the unweighted case to k_1^* and b^* and then use these values for the weighted case:

$$b = b^* \quad and \quad k_1 = k_1^* \frac{atf_{\,weighted}}{atf_{\,unweighted}}$$

where *atf* is the average term frequency.

The consequence of this argument is that we can probably avoid having to re-optimise the tuning parameters for different field weights. Under these conditions, the linear combination of frequencies method requires substantially less optimisation than the linear combination of scores. Some experimental results below support this argument.

3.4 Generality of the approach

This approach was developed for ranking with the BM25 [18] algorithm. Insofar as the proposed method defines a mapping from a structured document to a new non-structured one, the method can be applied to any ranking function for non-structured documents. However, the benefits of doing so will differ from function to function.

We believe that should bring similar advantages to any term weighting function that is nonlinear in tf. Document length calculations may be affected in different ways, depending on the function used.

Note also that the technique can in principle be applied at indexing or at search time, depending on other factors. A standard inverted-file structure, containing *either* full position information *or* term-frequency information by field could be used, with all field weighting performed at search time; this would also require that the usual document-length table be replaced by a document-field-length table. Under such a scheme, the field weights need not be determined at indexing time but could be flexible. Alternatively, fixed field weights could be used to generate an index containing the weighted term frequencies.

The range of structures dealt with is however limited. So far we have only discussed non-hierarchical and non-repeatable fields. This is ideal for fields such as Title and Body. In the next section a simple approach to deal with repeatable but non-hierarchical fields, such as anchor text or list items. However we are far from the complexity needed to deal properly with full structured IR problems such as those found in XML retrieval.

3.5 Anchor text

Anchor text presents some problems in addition to those already identified.

Anchor text is the text associated with a link in a source document, which is assumed to describe the target document. As is common practice, we extract (copy) it from its source and join it to the target. So one point of difference from (say) title and body is that it is not written by the author of the target, but by the author of the source (and presumably forms part of the text of the source). Another point is that it forms a *repeatable* field in the target – there may be any number of anchor texts joined to a given target document. In practice, in crawls of web collections, it is common that most documents have few incoming links but a few have very large numbers. In these cases the volume of anchor text may swamp the remainder of the document. Finally, we may not want to treat all incoming links as having the same weight – some source documents may be more authoritative than others.

A discussion of possible ways of dealing with anchor text in particular or repeatable fields in general would detract from the main focus of this paper. For the experiments reported below, we have taken the simplest possible approach. That is, we have merged all anchor text snippets from incoming links into a single field in the target document. We have not attempted to avoid the swamping effect, nor to distinguish between different sources.

Despite this simplicity, our method worked surprisingly well.

4. EMPIRICAL EVALUATION

We have hypothesized that score combination suffers from drawbacks that could be corrected by the proposed frequency combination method. Since the main advantage of the proposed system is to take account of dependence in the occurrences of a given term across fields, we expect the new algorithm to perform best on collections with many field types. Indeed the algorithm was originally designed for corporate collections containing from 5 to 30 field types (from Microsoft Office documents and others), including different section headings, a variety of list items, annotation fields, authoring and editing information fields, etc.

Unfortunately no such collections are publicly available[1] for evaluation. For this reason we were limited to standard evaluated collections and the few field types they expose (basically Title, Body and, in the case of web collections, Anchor). The results obtained, even with the small number of fields available, are already illustrative of the dangers of score combination. We expect the advantages of our approach to grow as the complexity of available evaluated collections grows.

4.1 Evaluation Method

We will use the Reuters vol I collection [17] and the 2002 TREC Web-Track crawl of the .gov domain [2] (which we will refer to as dotGov) for evaluation. These two collections expose only two types of text fields: Title and Body Text, and for dotGov also the Anchor text is available. The queries we use with Reuters are the titles of the 50 TREC–2002 filtering topics (the assessor topics), treated as adhoc queries, and for dotGov we use the titles of the 50 TREC–2002 (Web-Track) Topic Distillation task queries. These are two collections which are very different in many ways, and the two query sets represent very different tasks.

As optimization method we simply evaluate the performance of a system on a successively smaller grid over the set of parameters being optimized, until an adequate minimum-step value is reached. As a performance measure we used Precision at rank=10 (Prec@10) [2] since this measure is more meaningfully than Average Precision for the web task and is known to be correlated to it in any case. This is also the measure reported in the results; trends were similar on other measures considered (RPrec, reciprocal rank and mean average precision).

For score combinations (referred to as *ScoreComb*), we first optimise k_1 and b on each collection and each field separately, considering as a collection only the fields being scored. This requires $K = 3$ optimizations of two parameters[2]

For term frequency combinations (referred to as *FreqComb*), we optimise k_1 and b using weights of 1 for available

[1]Except perhaps for the INEX collection [3], which deals with full XML queries

[2]The resulting optimal values for k_1 and b in dotGov are respectively for Title: 0.75 and 0.95; for Body: 0.75 and 0.95; and for Anchor 0.25 and 1. For Reuters, they are for Title: 0 and 0.3; and for Body 1 and 0.2.

Figure 2: Title and Body Fields

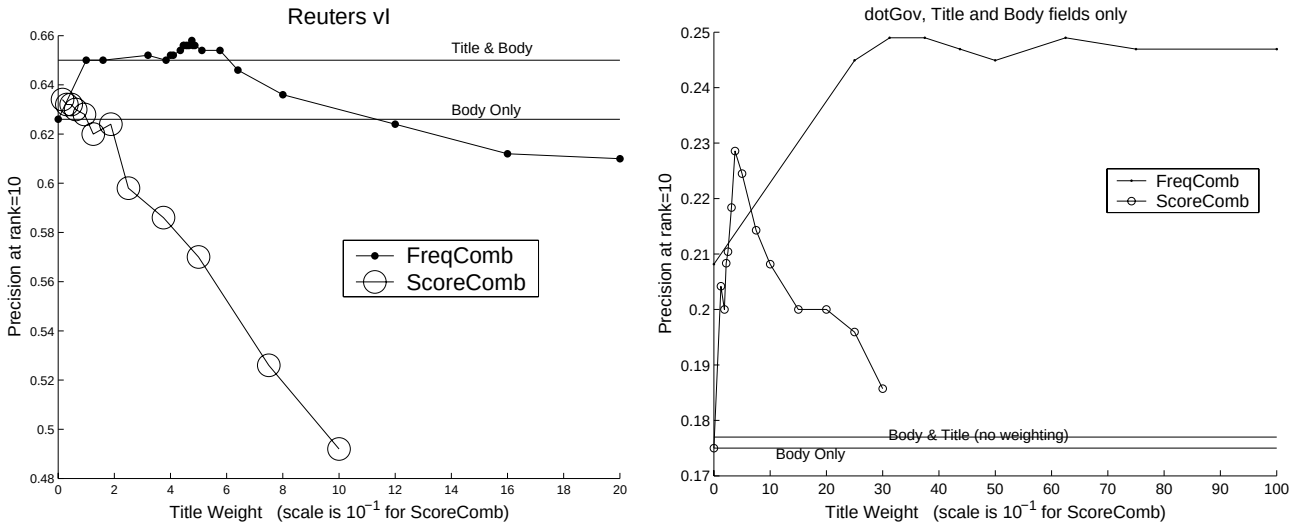
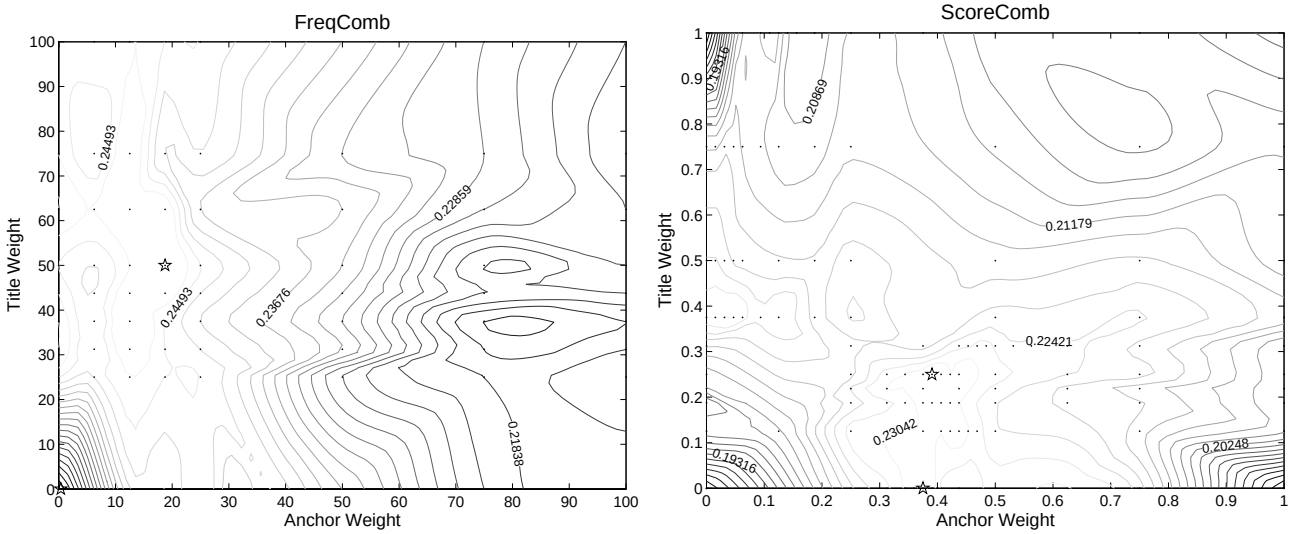


Figure 3: Title, Body & Anchor Fields (Prec@10 contour extrapolated from the points in grey; ⋆ indicates maximum)



fields[3]. This requires a single optimization of two parameters. As discussed in 2.3 we do not need to re-optimise k_1 as the field weights change, since the new parameter can be reliably derived from the weights and the collection statistics, and so this is done without any cost.

Field weights $\mathbf{v}$ are optimized in the same way for both ScoreComb and FreqComb: we set the Body weight to 1 and optimise the remaining weights. The optimisation cost is the same for both methods: a single optimization of $K-1$ parameters. The choice of the initial field weight does not affect the final results (since the combination is linear for ScoreComb, and for FreqComb the non-linearity is scaled by k_1 which is adjusted according to the weights).

Having a reasonable idea of the optimal weights can greatly speed the optimization process. In the case of FreqComb we will see that these weights follow our intuition about the relative importance of the fields: title has a higher than anchor and both have a weight several times that of the body. On the other hand, ScoreComb weights are not so intuitive: Title and Anchor weights are one or two orders of magnitude smaller than Body.

Both tests and optimizations are carried out on the same document set; therefore our results will be over–fitted to some degree. We did not attempt to mitigate this by cross-validation or by changing training and test collections. Our objective in this paper is to point out the dangers of score-combination and the advantages of frequency combination, and we feel these results achieve this objective despite the level of over-fitting expected from our optimization procedure.

4.2 Body and Text

Let us first consider the Title and Body fields only. Figure 2 shows the results for the FreqComb method and the ScoreComb method for different values of the Title field weight. We also indicate the performance of using only the body field (marked *Body Only*) and using body and title together without weighting (marked *Title & Body*). Note that for display purposes we scaled by 10 the weights for ScoreComb, so the range of weights explored is between 0 and 3. Let us first discuss the results on Reuters (Figure 2 left). FreqComb, the proposed field weighting approach behaves as expected. For a weight value of 1 we recover the result obtained without any field weighting. Then performance increases with higher weights for Title, reaching its maximum around 8 and then decreases slowly. Title weights between 1 and 5.5 improve over the non-weighted results, although so slightly that we do not believe this to be significant (the maximum Prec@10 gain on this collection is under 0.01, or 1.5% relative improvement).

ScoreComb, the score combination approach however is greatly hurt by exploiting field combination. Setting the Title to 0.001 increases very slightly the performance over Body only, but any other weight setting decreases it. More importantly, performance never reaches the baseline, for any weight settings. Trying to exploit fields actually decreased performance in this case. Note also the danger of setting Title weight to 1 (10 in the x-axis of figure 3): the Prec@10 performance would decrease 24% (0.16 points) below the baseline!

This is exactly the effect we tried to mitigate with the proposed field weighting approach. It seems to us that using a weighting scheme that does not guarantee (for some setting of the weights) at least as good performance as the baseline can be quite dangerous. Furthermore the scale of the weights with the proposed approach is quite intuitive (e.g. the title has 5 times the weight of the Body), unlike the scale for ScoreComb. Note that for score combination the scale of the weightings does not matter (only their ratio matters), so lowering the initial body weight from 1 to a small value would have no effect on this result, the optimal body weight would always be orders of magnitude greater than the title weight.

Another lesson to be drawn from this example is that it is not so clear that document structure does constitute a source of information for relevance; in many cases exploiting structure may constitute a distraction instead of an improvement. We expect this to worsen when dealing with a wider variety of field types.

Carrying out the same experiment on the dotGov task (ignoring all Anchor) we observe similar results. The main difference is that both methods produce noticeable improvements over the baseline. However, the FreqComb method produces noticeable improvements again over the ScoreComb method. The scale of the weights for ScoreComb is again counter-intuitive: performance decreases for Title weights greater than 0.5.

4.3 Anchor Text

The third field type readily available on current evaluated collections is anchor-text. Anchor-text, as we discussed in section 3.5, is not a standard type of field for several reasons, and might be treated in various ways. However, our focus is not concerned with the particular problem of anchor-text scoring, but rather with studying the effects of score combination and frequency combination on this type of field.

Figure 3.5 shows the Prec@10 contour plots when using FreqComb (left) and ScoreComb (right) for different title and anchor weightings (keeping Body weight to 1). Contours were inferred from the points obtained during the grid minimisation procedure employed to find the maximum Prec@10 weights (shown as small grey dots). The highest value is indicated with a star.

From Figure 3 right we can see that the baseline for this task using only body and title text is 0.177. So in this case both methods outperform this baseline for appropriate weight ranges. Best performance for FreqComb is reached at Prec@10=0.25 (41% relative improvement) for Title and Anchor weights of 50 and 20 respectively. Note that for Title and Anchor weights equal to 1 we obtain less than 1% relative improvement over using only body and title, so field weighting is crucial for this task.

These results were obtained with the method of adjusting k_1 suggested in section 3.3. We have also run some experiments starting with fixed k_1 but then re-optimising k_1 and b after setting the field weights. These give slightly worse results. This confirms that the argument of section 3.3 does work well, and that no further optimisation is required.

For ScoreComb we must choose weights smaller than 1 to improve over the baseline. The maximum is obtained at Prec@10=0.235 (32% relative improvement) for Title and

[3]The resulting optimal values for $k1$ and b in dotGov are, without anchor, 1.0 and 0.85; and with anchor 2.0 and 0.85. For Reuters, the resulting k_1 and b are 0.9 and 0.2 respectively.

Anchor weight near 0.2 and 0.4 respectively. Note that the high contour lines intersect the x axis at the bottom, meaning that changes in the Title weights from 0 to 0.3 have no effect on the performance. In fact the same optimal performance of 0.235 can be reached for Title weight 0 and the Anchor weight to 0.375 (indicated by a second star in the figure). This tells us that the Title information is not being appropriately exploited by the ScoreComb method.

5. CONCLUSION AND FUTURE WORK

The common current method of weighting fields of a document by means of a linear combination of scores is problematic. We have demonstrated both theoretically and empirically some of the issues, and proposed an alternative, the linear combination of term frequencies. The heart of the problem relates to the behaviour of term weighting functions, also in common use, that are non-linear in tf. A linear combination of term frequencies prior to weighting appears to resolve many of the issues, and has added advantages of simplicity and of potentially reducing the effort required to optimise the tuning parameters of a ranking function. The method was primarily intended for use with the BM25 function but is probably useable with several other document ranking functions.

Experiments were conducted on two existing test collections. These were not ideal for our purpose, because of the limited variety of fields they contain (title and body, and in one case anchor text) – the method was intended to help in situations where a larger number of fields have been identified. Also its applicability to anchor text was at least arguable. Nevertheless, the method outperformed the linear combination of scores with and without anchor text, and the optimal field weights obtained for the new method were much more intuitively understandable. We expect this method to show even greater benefits with collections with many fields.

We believe that the linear combination of scores should be rejected in cases where the components that contribute to the score can be combined across fields at an earlier stage.

6. REFERENCES

[1] David Carmel, Yoelle S. Maarek, Matan Mandelbrod, Yosi Mass, and Aya Soffer. Searching xml documents via xml fragments. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 151–158. ACM Press, 2003.

[2] Nick Craswell and David Hawking. Overview of the TREC-2002 Web track. In *TREC 2002*, 2003.

[3] INEX. Initiative for the evaluation of XML retrieval (INEX), `http://inex.is.informatik.uni-duisburg.de:2003`.

[4] Evangelos Kotsakis. Structured information retrieval in XML documents. In *SAC 2002*, volume 1-58113-445-2/02/03, Madrid, Spain, 2002. ACM.

[5] Mounia Lalmas. Uniform representation of content and structure for structured document retrieval. Technical report, Queen Mary and Westfield College, University of London, 2000.

[6] S H Myaeng, D-H Jang, M-S Kim, and Z-C Zhoo. A flexible model for retrieval of SGML documents. In W B Croft, A Moffat, C J van Rijsbergen, R Wilkinson, and J Zobel, editors, *SIGIR'98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 138–145. ACM Press, 1998.

[7] N.Craswell, D.Hawking, A.McLean, T.Upstill, R.Wilkinson and M.Wu. TREC 12 web track at CSIRO. In *TREC 2003*, 2004.

[8] Hongbo Xu, Zhifeng Yang, Bin Wang, Bin Liu, Jun Cheng, Yue Liu, Zhe Yang, Xueqi Cheng and Shuo Bai TREC-11 experiments at CAS-ICT: Filtering and Web. In *TREC 2002*, 2003.

[9] Lide Wu, Xuanjing Huang, Junyu Niu, Yingju Xia, Zhe Feng and Yaqian Zhou. FDU at TREC2002: Filtering, Q&A, Web and Video tasks. In *TREC 2002*, 2003.

[10] Einat Amitay, David Carmel, Adam Darlow, Ronny Lempel and Aya Soffer. Topic distillation with knowledge agents. In *TREC 2002*, 2003.

[11] Abdur Chowdhury, Mohammed Aljlayl, Eric Jensen, Steve Beitzel, David Grossman and Ophir Frieder. Linear combinations based on document structure and varied stemming for Arabic retrieval. In *TREC 2002*, 2003.

[12] Nie Yu, Ji Donghong and Yang Lingpeng. LIT at TREC-2002: Web track. In *TREC 2002*, 2003.

[13] Shuang Liu, Clement Yu and Wensheng Wu. UIC at TREC-2002: Web track. In *TREC 2002*, 2003.

[14] Jacques Savoy and Yves Rasolofo. Report on TREC-11 experiment: Arabic, Named Page and Topic Distillation searches. In *TREC 2002*, 2003.

[15] Paul Ogilvie and Jamie Callan. Combining document representations for known item search. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003)*, 2003.

[16] Benjamin Piwowarski and Patrick Gallinari. A machine learning model for information retrieval with structured documents. In Petra Perner, editor, *Machine Learning and Data Mining in Pattern Recognition (MLDM'03)*, pages 425–438, Leipzig, Germany, July 2003. Springer Verlag.

[17] ReutersI. Reuters corpus volume 1, `http://about.reuters.com/researchandstandards/corpus/index.asp`.

[18] S E Robertson and S Walker. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In W B Croft and C J van Rijsbergen, editors, *SIGIR '94: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 345–354. Springer-Verlag, 1994.

[19] Ross Wilkinson. Effective retrieval of structured documents. In *Research and Development in Information Retrieval*, pages 311–317, 1994.