Int J Digit Libr (2000) 3: 131–139

A probabilistic justification for using tf×idf term weighting in information retrieval

Djoerd Hiemstra

Centre for Telematics and Information Technology, University of Twente, The Netherlands;
E-mail: hiemstra@ctit.utwente.nl

Received: 17 December 1998/Revised: 31 May 1999

Abstract. This paper presents a new probabilistic model of information retrieval. The most important modeling assumption made is that documents and queries are defined by an ordered sequence of single terms. This assumption is not made in well-known existing models of information retrieval, but is essential in the field of statistical natural language processing. Advances already made in statistical natural language processing will be used in this paper to formulate a probabilistic justification for using tf×idf term weighting. The paper shows that the new probabilistic interpretation of tf×idf term weighting might lead to better understanding of statistical ranking mechanisms, for example by explaining how they relate to coordination level ranking. A pilot experiment on the TREC collection shows that the linguistically motivated weighting algorithm outperforms the popular BM25 weighting algorithm.

Key words: Information retrieval theory – Statistical information retrieval – Statistical natural language processing

1 Introduction

There are three basic processes an information retrieval system has to support: the representation of documents, the representation of a user request, and the comparison of these two representations. In *text* retrieval the documents and the user request are expressed in natural language. Although text retrieval has had by far the most attention in the information retrieval community, so far the success of natural language processing techniques has been limited. Most of the effort in the field of text retrieval has been put in the development of statistical retrieval models like the vector space model (proposed by Salton et al. [16]), the classical probabilistic model (proposed by Robertson and Spark Jones [12]) and more recently the inference network model (proposed by Croft and Turtle [3]).

The application of natural language processing techniques in combination with these models has solid but limited impact on the performance of text retrieval[1] [19]. The research does however provide little insight to the question *how* to use natural language processing. Natural language processing modules are usually considered as preprocessing steps, that is, they are not included in the model itself. This paper attempts to formulate a model that captures statistical information retrieval and statistical natural language processing into one unifying framework, an approach that others are also beginning to investigate [9, 11]. It is the model itself that explicitly defines how documents and queries should be analysed. This seems a rather trivial requirement, but we claim that this is not the general idea behind the existing models for information retrieval. The (implicit) assumption made by these retrieval models is that some procedure, either manual or automatic, is used to assign index terms to documents. It is the result of this procedure that can be reflected by the model, not the procedure itself.

This paper is organised as follows. In Sect. 2 the basics of the linguistically motivated retrieval model are presented. Section 3 gives a new probabilistic interpretation of tf×idf term weighting by using estimation procedures developed in the field of statistical natural language processing. Section 4 presents a number of experiments, one pilot experiment on the relatively outdated Cranfield collection and two additional experiments on the TREC ad hoc and TREC-CLIR collection. Finally, Sect. 5 presents

[1] Retrieval performance is usually measured in terms of precision (the fraction of the retrieved documents that is actually relevant) and recall (the fraction of the relevant documents that is actually retrieved).

conclusions and plans for future work. These plans include the development of a model for phrases and the development of a model for cross-language information retrieval. An early version of this paper was presented at the Second European Conference on Digital Libraries (ECDL) [5].

2 The basic retrieval model

This paper defines a linguistically motivated model of full text information retrieval. The most important modeling assumption we make is that documents and queries are defined by an ordered sequence of words or terms.[2] This assumption is usually not made in information retrieval. In the models mentioned in the introduction, documents and queries are modeled as unordered collections of terms or concepts. In the field of statistical natural language processing the word order assumption is essential for many applications, for instance part-of-speech tagging, speech recognition and parsing. By making the 'ordered sequence of terms assumption' we will be able to use advances already made in statistical natural language processing. In this section we will define the framework that will be used in the subsequent sections to give a probabilistic interpretation of tf×idf term weighting.

2.1 An informal description: drawing query terms from a document

Before we describe the new retrieval model mathematically, this section gives an informal description of the underlying ideas.

The main goal of an information retrieval system is to find those documents in a document collection that are relevant to a query. A full text retrieval system compares the words in the query with the words in each document to rank the documents. Documents that are likely to be relevant should be ranked at the top and documents that are unlikely to be relevant should be ranked at the bottom of the ranked list. A mathematical model of information retrieval formally defines *how* the system should perform this ranking, usually based on intuitions or metaphors from some well-understood branch of mathematics. For example Salton's vector space model is based on intuitions from geometry: documents and queries are vectors in a high-dimensional space and documents are ranked by the cosine of the angle that separates the document vector and the query vector. The Robertson/Sparck-Jones probabilistic model is based on the intuition that a system can learn from the distribution of terms over relevant and non-relevant documents which documents are probably relevant to a query.

[2] In the linguistically motivated model *terms* and *words* are equivalent, both expressions will be used in this paper. A classical index term that consists of more than one word will be called a *phrase*.

We will use probability theory in a different way here by using a metaphor that is very similar to the "sampling coloured balls from urns" examples that are often used in introductory statistics courses [10]. Instead of drawing balls at random with replacement from an urn, we will consider the process of drawing words at random with replacement from a document. Suppose someone selects one document in the document collection; draws at random, one at a time, with replacement ten words from this document and hands those ten words (the query terms) over to the system. The system now can make an educated guess as from which document the words came from, by calculating for each document the probability that the ten words were sampled from it and by ranking the documents accordingly. This metaphor for information retrieval was introduced by Ponte and Croft [11]. The intuition behind it is that users have a reasonable idea of which terms are likely to occur in documents of interest and will choose query terms accordingly.

The metaphor is a very powerful one as it can be extended in various ways. Because of the sequential nature of the sampling process, it can be extended to model phrases as done by Miller, Leek and Schwartz [9]. It can be extended to Boolean queries by treating the sampling process as an AND-query and allowing that each draw is specified by a disjunction of more than one term. For example, the probability of first drawing the term *information* **and** then drawing either the term *retrieval* **or** the term *filtering* from a document can be calculated by the model introduced in this paper without any additional modeling assumptions. Furthermore, it can be extended with additional statistical processes to model differences between the vocabulary of the query and the vocabulary of the documents. For instance, for cross-language retrieval, statistical translation can be added to the process of sampling terms from a document: e.g., first an English word is sampled from the document, and then this word is translated to Dutch with some probability that can be estimated from a parallel corpus. Evaluations of Boolean queries and statistical translation are described in [6, 7]. In this paper we will focus on the basics of the new retrieval model by defining it mathematically and by stating how it relates to existing tf×idf term weighting algorithms. We will deal with the mathematical details of the extensions of the model in future publications.

2.2 The sample space

We assume that a collection consists of a finite number of textual documents. The documents are written in a language that exists of a finite number of words or terms.

Definition 1. Let P be a probability function on the joint sample space $\Omega_D \times \Omega_T$. Let Ω_D be a discrete sample space that contains a finite number of points d such that each d refers to an actual document in the document collection. Let D be discrete random variable over Ω_D. Let

Ω_T be a discrete sample space that contains a finite number of points t such that each t refers to an actual term that is used to represent the documents. Let T be a discrete random variable over Ω_T.

In other words, the random variable D refers to a document id and the random variable T refers to an index term.

2.3 Modeling documents and queries

Queries will be modeled as compound events. A compound event is an event that consists of two or more single events, as when a die is tossed twice or three cards are drawn one at a time from a deck [10]. The single events that define the compound event are the query terms. In general the probability of a compound event does depend on the order of the single events. For example a query of length n is modeled by an ordered sequence on n single terms $T_1, T_2, \cdots, T_n$. Given a document id D the probability of the ordered sequence will be defined by $P(T_1, T_2, \cdots, T_n|D)$. Most practical models for information retrieval assume independence between index terms. Assuming conditional independence of terms given a document id leads to the following model.

$$P(T_1, T_2, \cdots, T_n|D) = \prod_{i=1}^{n} P(T_i|D) \tag{1}$$

Note that the assumption of independence between query terms does not contradict the assumption that terms in queries have a particular order. The independence assumption merely states that every possible order of terms has the same probability. It is made to illustrate that a simple version of the linguistically motivated model is very similar to existing information retrieval models.

2.4 The matching process

Equation (1) can be used directly to rank documents given a query $T_1, T_2, \cdots, T_n$. It might however be interesting to rewrite (1) to a probability measure that explicitly ranks documents given a query: $P(D|T_1, T_2, \cdots, T_n)$. This measure can be related to (1) by applying Bayes' rule.

$$P(D|T_1, T_2, \cdots, T_n) = P(D)\frac{P(T_1, T_2, \cdots, T_n|D)}{P(T_1, T_2, \cdots, T_n)} \tag{2}$$

$$= P(D)\frac{\prod_{i=1}^{n} P(T_i|D)}{P(T_1, T_2, \cdots, T_n)} \tag{3}$$

Equation (2) is the direct result of applying Bayes' rule. Filling in the independence assumption of (1) leads to (3). It seems tempting to make the assumption that terms are also independent if they are not conditioned on a document D. This will however lead to an inconsistency of the model (e.g., see Cooper's paper on modeling assumptions for the classical probabilistic retrieval model [2]). Since $\sum_d P(D = d|T_1, T_2, \cdots, T_n) = 1$ we can scale the formula using a constant C such that $\frac{1}{C} = \sum_d P(D = d \cap T_1, T_2, \cdots, T_n)$.

$$P(D|T_1, T_2, \cdots, T_n) = C\ P(D) \prod_{i=1}^{n} P(T_i|D) \tag{4}$$

Equation (4) defines the ranking formula of the linguistic motivated probabilistic retrieval model if we assume term independence.

3 Estimating the probabilities

The process of probability estimation defines how probabilities should be estimated from the frequency of terms in the actual document collection. We will look at the estimating process by drawing a parallel to statistical natural language processing and corpus linguistics.

3.1 Viewing documents as language samples

The general idea is the following. Each document contains a small sample of natural language. For each document the retrieval system should build a little statistical language model $P(T|D)$ where T is a single event. Such a language model might indicate that the author of that document used a certain word 5 out of 1000 times; it might indicate that the author used a certain syntactic construction like a phrase 5 out of 1000 times; or ultimately indicate that the author used a certain logical semantic structure 5 out of 1000 times.

One of the main problems in statistical natural language processing and corpus linguistics is the problem of sparse data. If the sample that is used to estimate the parameters of a language model is small, then many possible language events never take place in the actual data. Suppose for example that an author wrote a document about *information retrieval* without using the words *keyword* and *crocodile*. The reason that the author did not mention the word *keyword* is probably different from the reason for not mentioning the word *crocodile*. If we were able to ask an expert in the field of *information retrieval* to estimate probabilities for the terms *keyword* and *crocodile* he/she might for example indicate that the chance that the term *keyword* occurred is one in a thousand terms and the chance that the term *crocodile* occurred is much lower: one in a million. If we however base the probabilities on the frequency of terms in the actual document then the probability estimates of low frequent and medium frequent terms will be unreliable. A full text information retrieval system based on these frequencies cannot make a difference between words that were not used 'by chance', like the word *keyword*, and words that were not used because they are 'not part of the vocabulary of the

subject', like the word *crocodile*. Furthermore there is always a small chance that completely off the subject words occur like the word *crocodile* in this paper.

We believe that the sparse data problem is exactly the reason that it is hard for information retrieval systems to obtain high recall values without degrading values for precision. Many solutions to the sparse data problem were proposed in the field of statistical natural language processing (e.g., see [8] for an overview). We will use the combination of estimators by linear interpolation to estimate parameters of the probability measure $P(T|D)$.

3.2 Estimating probabilities from sparse data

Perhaps the most straightforward way to estimate probabilities from frequency information is maximum likelihood estimation [10]. A maximum likelihood estimate makes the probability of observed events as high as possible and assigns zero probability to unseen events. This makes the maximum likelihood estimate unsuitable for directly estimating $P(T|D)$. One way of removing the zero probabilities is to mix the maximum likelihood model of $P(T|D)$ with a model that suffers less from sparseness like the marginal $P(T)$. It is possible to make a linear combination of both probability estimates so that the result is another probability function. This method is called linear interpolation:

$$P_{li}(T|D) = \alpha_1 P_{mle}(T) + \alpha_2 P_{mle}(T|D)\,, \quad (0 < \alpha_1, \alpha_2 < 1 \quad \text{and} \quad \alpha_1 + \alpha_2 = 1) \tag{5}$$

The weights α_1 and α_2 might be set by hand, in which case we would choose them in such a way that $\alpha_1 P_{mle}(T = t)$ is smaller than $\alpha_2 P_{mle}(T = t|D)$ for each term t. This will give terms that did not appear in the document a much smaller probability than terms that did appear in the document. In general one wants to find the combination of weights that works the best, for example by optimising them on a test collection consisting of documents, queries and corresponding relevance judgements.

Table 1 lists the frequencies that are used to estimate the probabilities of the model. Two frequencies are particularly important, the term frequency and the document frequency. The term frequency of a term is defined by the number of times a term appears in a document and can be viewed at as local or document specific information. Given a specific document many terms will have a frequency of zero, so the term frequency suffers from sparseness. The document frequency of a term is defined by the number of documents in which a term appears and can be viewed as global information. (Sometimes document frequency is referred to as collection frequency.) The document frequency of a term will never be zero, because by definition 1, terms that do not appear in any document will not be included in the model. The sparseness problem can be avoided by estimating $P(T|D)$ as a linear combination of a probability model based on document frequency and a probability model based on term frequency as in (7):

Table 1. Frequency information

N	the number of documents in the collection
$tf(t,d)$	*term frequency*: the number of times the term t appears in the document d.
$df(t)$	*document frequency*: the number of documents in which the term t appears.

$$P(D = d) = \frac{1}{N} \tag{6}$$

$$P(T_i = t_i|D = d) = \alpha_1 \frac{df(t_i)}{\sum_t df(t)} + \alpha_2 \frac{tf(t_i,d)}{\sum_t tf(t,d)} \tag{7}$$

Note that term frequency and document frequency are not derived from the same distribution. Although the term frequency can also be used to compute global information of a term by summing over all possible documents, this information will usually not be the same as the document frequency of a term, more formally: $df(t) \neq \sum_d tf(t,d)$.

Equations (4) and (7) define the ranking algorithm. The formula bears some resemblance with the ranking formula used by Miller, Leek and Schwartz [9]. They showed that the model can be interpreted as a two-state hidden Markov model in which α_1 and α_2 define the state transitions.

3.3 Relation to tf×idf

The use of term frequency and document frequency to rank documents was extensively studied, especially by Salton et al., for the vector space model [15]. Following considerations of the term discrimination model [17], they argued that terms appearing in documents should be weighted proportional to the term frequency and inversely proportional to the document frequency. Weighting schemes that follow this approach are called *tf×idf* (term frequency × inverse document frequency) weighting schemes. The combination of tf×idf weights and document length normalisation gave the best retrieval results on several test collections, but they were not able to justify their approach by probability theory (which is not a prerequisite for using it in the vector space model anyway):

> *... The term discrimination model has been criticised because it does not exhibit well substantiated theoretical properties. This in contrast with the probabilistic model of information retrieval ...*

The lack of theoretical justification of tf×idf weights did not keep developers of the probabilistic model and the inference network model from using them. Robertson et al. [13] justified the use of term frequency in the probabilistic model by approximating a ranking formula that is

based on the combination of the probabilistic model and the 2-Poisson model. There is however a more plausible probabilistic justification of tf×idf weighting which can be justified by the linear interpolation estimator of (7). This can be shown by rewriting. Multiplying the ranking formula defined by (4), (6) and (7) with values that are the same for each document will not affect the final ranking, so we can multiply the ranking formula by $df(t)$ and α_1 as follows:

$$
\begin{aligned}
&P(D=d|T_1=t_1,\cdots,T_n=t_n) \propto \\
&\propto \prod_{i=1}^{n}(\alpha_1\frac{df(t_i)}{\sum_t df(t)}+\alpha_2\frac{tf(t_i,d)}{\sum_t tf(t,d)}) \qquad \text{[by (4),(6) and (7)]} \\
&\propto \prod_{i=1}^{n}(\alpha_1+\alpha_2\frac{tf(t_i,d)\sum_t df(t)}{\sum_t tf(t,d)\cdot df(t_i)}) \qquad [\times \textstyle\prod_{i=1}^{n}\frac{\sum_t df(t)}{df(t_i)}] \\
&\propto \prod_{i=1}^{n}(1+\frac{tf(t_i,d)}{df(t_i)}\cdot\frac{1}{\sum_t tf(t,d)}\cdot\frac{\alpha_2\sum_t df(t)}{\alpha_1}) \, [\times \textstyle\prod_{i=1}^{n}\frac{1}{\alpha_1}]
\end{aligned}
$$

The resulting formula can directly be interpreted as a tf×idf weighting algorithm with document length normalisation, because:

$\frac{\alpha_2\sum_t df(t)}{\alpha_1}$	is constant for any document d and term t
$\frac{tf(t_i,d)}{df(t_i)}$	is the tf×idf weight of the term t_i in the document d
$\frac{1}{\sum_t tf(t,d)}$	is the inverse length of the document d

Any monotonic transformation of the document ranking function will produce the same ranking of the documents. Instead of the product of weights we could therefore also rank the documents by the sum of logarithmic weights.

$$\propto \sum_{i=1}^{n}\log(1+\frac{tf(t_i,d)}{df(t_i)}\cdot\frac{1}{\sum_t tf(t,d)}\cdot\frac{\alpha_2\sum_t df(t)}{\alpha_1}) \text{[log. tr.]}$$

Robertson and Sparck-Jones [12] call the resulting formula a presence weighting scheme (as opposed to a presence/absence weighting scheme) because the formula assigns a zero weight to terms that are not present in a document. Presence weighting schemes can be implemented using the vector product formula as introduced by Salton et al. [15]. The query weights of the vector product formula can be used to account for multiple occurrences of the same term in the query. The resulting vector product version of the ranking formula is displayed in Table 2.

On first glance the constant $\alpha_2\sum_t df(t)/\alpha_1$ seems to have little impact on the final ranking. But in fact, different values of α_1 and α_2 will lead to different document rankings. In Sect. 3.5 we will show some effects of different values of α_1 and α_2 on the ranking of documents, especially for short queries.

As said above, the weighting algorithm can, by the definition of Salton et al., be interpreted as a tf×idf weight with document length normalisation. However, the $\sum_t tf(t,d)$ (that is: the document length) in the denominator of the document term weight in Table 2 is the result of the requirement that probabilities have to sum up to one and not the results of document length normalisation. Document length normalisation *is* assumed by (6). In fact we might assume that longer documents are more likely to be relevant by using the prior probability of (8).

Table 2. Vector product version of weighting algorithm

$$
\begin{aligned}
&\text{similarity}(Q,D)=\sum_{k=1}^{l} w_{qk}\cdot w_{dk} \\
&w_{qk}=tf(t_k,q) \\
&w_{dk}=\log(1+\frac{tf(t_k,d)}{df(t_k)\sum_t tf(t,d)}\cdot\frac{\alpha_2\sum_t df(t)}{\alpha_1})
\end{aligned}
$$

$$P(D=d)=\frac{\sum_t tf(t,d)}{\sum_t\sum_d tf(t,d)} \tag{8}$$

This results in a weighting algorithm that cannot be rewritten into the vector product normal form. It can however be implemented fairly easily by initialising similarities to $\log(\sum_t tf(t,d))$ instead of to zero when processing the query. This version of the weighting algorithm was used in the experiments of Sect. 4.

3.4 A new informal definition of tf×idf weighting

Equation (7) gives rise to a new informal definition of tf×idf weighting. Giving an informal definition after the formal definition seems a bit useless, but we believe that it will help to understand what exactly makes tf×idf weighting successful. The classical definition of tf×idf weighting can be formulated as follows:

Definition 2. The weight of a term that appears in a document should increase with the term frequency of the term in the document and decrease with the document frequency of the term. Terms that do not appear in a document should all get the same weights (zero weights).

An alternative definition is based on (7). If we assume that $\alpha_1 df(t_i)/\sum_t df(t)$ is much smaller than $\alpha_2 tf(t_i,d)/\sum_t tf(t,d)$ then it can be formulated as follows:

Definition 3. The weight of a term that appears in a document should increase with the term frequency of the term in the document. The weight of a term that does not appear in a document should increase with the document frequency of the term.

An example may clarify the implications of both definitions. Suppose the user formulates the query *information retrieval* and there is no document in the collection

in which the terms *information* and *retrieval* both appear. Furthermore, suppose that the term *information* is much more common, i.e., has a higher document frequency than the term *retrieval*. Now the system will rank documents containing k occurrences of the term *retrieval* above documents containing k occurrences of the term *information*. The classical explanation would be as follows:

Explanation 1. The query term *retrieval* matches better with documents containing *retrieval* than the query term *information* matches with documents containing *information*, because *retrieval* has a higher inverse document frequency than *information*.

The alternative explanation would be that:

Explanation 2. The query term *information* matches better with documents not containing *information* than the query term *retrieval* matches documents not containing *retrieval* because *information* has a higher document frequency than *retrieval*.

There is no a priori reason to prefer one explanation above the other. However, the idea of definition 3, that global information is only used to weight terms of which there is no local information, might lead to better understanding of probabilistic term weighting in text retrieval.

3.5 The problem of non-coordination level ranking

There is a well-known problem with statistical information retrieval systems that use tf×idf weighting: sometimes documents containing n query terms are ranked higher than documents containing $n+1$ query terms. We will call this problem the *problem of non-coordination level ranking* in which the *coordination level* refers to the number of distinct query terms contained in a document. A coordination level ranking procedure will always rank documents containing $n+1$ query terms above documents containing n query terms even if the top documents have little evidence for the presence of $n+1$ query terms and lower-ranked documents have a lot of evidence for the presence of n terms.

According to studies of user preferences and evaluations on test collections the problem of non-coordination level ranking becomes particularly apparent if short queries are used [14]. In a lot of practical situations short queries are the rule rather than the exception, especially in situations where there is no or little user training like with Web-based search engines. For some research groups, the importance of coordination level is the reason for developing ranking methods that are based on the lexical distance of search terms in documents instead of on document frequency of terms [1, 4]. However, as pointed out by experiments of Wilkinson et al. [21], some tf×idf measures (e.g., like the measure proposed by Robertson et al. [13]) are more like coordination level ranking than others (e.g., like the measure proposed by Salton et al. [15]). Wilkinson et al., showed that weighting measures that are more like coordination level ranking perform better on the TREC collection, especially if short queries are used.

Following the results of Wilkinson et al., it might be useful to investigate what exactly makes a weighting measure "like" coordination level ranking. The following example may provide some insight. First of all, suppose we use the following ranking formula which can be derived from the probability ranking function in a similar way as is shown in Sect. 3.3.

$$P(D=d|T_1=t_1,\cdots,T_n=t_n) \propto \prod_{i=1}^{n}\left(\frac{\alpha_1}{\alpha_2\sum_t df(t)}+\frac{tf(t_i,d)}{df(t_i)}\cdot\frac{1}{\sum_t tf(t,d)}\right) \tag{9}$$

Now suppose the user enters a small query of only two terms a and b. As in the previous example a might be the term *information* and b might be the term *retrieval*. Furthermore, suppose that the document d_1 contains a lot of evidence for term a and no evidence for the term b; and that document d_2 contains little evidence of both terms. It can be shown that document d_1 will have a lot of evidence for a and none for b if $tf(a,d_1)$ is high, $tf(b,d_1)=0$ and the length of d_1 is short. Document d_2 contains little evidence of a and b if $tf(a,d_2)=tf(b,d_2)=1$ and if d_2 is a long document. Now the following equation defines the requirement for coordination level ranking, that is, the similarity of document d_1 to the query $a\ b$ should be smaller than the similarity of document d_2 to the query. The left hand side of the equation contains the similarity of the query compared to document d_1 and the right hand side contains the similarity of the query to document d_2. We will use short notations of the document length and the constant of (9): $l(d)=\sum_t tf(t,d)$ and $c=\alpha_1/\alpha_2\sum_t df(t)$.

$$\begin{aligned}
(c+\frac{tf(a,d_1)}{df(a)l(d_1)})(c+0) &< (c+\frac{1}{df(a)l(d_2)})(c+\frac{1}{df(b)l(d_2)})\\
c^2+\frac{c\,tf(a,d_1)}{df(a)l(d_1)} &< c^2+\frac{c}{df(a)l(d_2)}+\frac{c}{df(b)l(d_2)}\\
&\quad+\frac{1}{df(a)df(b)l(d_2)^2}\\
\frac{c\,tf(a,d_1)}{df(a)l(d_1)}-\frac{c}{df(a)l(d_2)}-\frac{c}{df(b)l(d_2)} &< \frac{1}{df(a)df(b)l(d_2)^2}\\
&\vdots\\
c &< \frac{l(d_1)}{l(d_2)(tf(a,d_1)df(b)l(d_2)-df(b)l(d_1)-df(a)l(d_1))}
\end{aligned} \tag{10}$$

Equation (10) shows that we can rewrite the requirement for coordination level ranking as a requirement for the constant c. If c is small enough then the problem of non-coordination level ranking will never occur. By changing the value of c the ranking formula can be adapted to

different applications. If we are developing a Web-based search engine we might choose a relatively small value for c, but if we are developing a search engine for evaluation in TREC [20] we might choose a higher value of c. For the Web-based search engine we might define a collection specific lower bound of c by keeping track of the collection extrema like maximum term frequency and document frequency ($maxtf$ and $maxdf$) and maximum and minimum document lengths ($maxl$ and $minl$). If we fill in these extrema and the definition $c = \alpha_1/\alpha_2 \sum_t df(t)$ in ((10)) then the lower bound will be defined as follows on the ratio between α_1 and α_2.

$$\frac{\alpha_1}{\alpha_2} < \frac{minl \cdot \sum_t df(t)}{maxl \cdot (maxtf \cdot maxdf \cdot maxl - maxdf \cdot minl - minl)} \tag{11}$$

Equation (11) defines a ranking formula that always produces coordination level ranking for queries of two words. For longer queries the bound will be lower and for queries with unrestricted length only $\alpha_1 = 0$ will guarantee coordination level ranking.

3.6 A plausible explanation of non-coordination level ranking

The arguments in the previous section showed the following. The smaller the value of the constant c, the more the ranking formula will behave like coordination level ranking. It is good to note that most tf×idf measures defined for the existing models of information retrieval include constants for which the arguments introduced above also hold (for instance the "+0.5" in the Robertson/Sparck Jones formula [12, 13]). However, the classical definition of tf× idf weighting (definition 2) does not give a plausible explanation of why and when non-coordination level ranking does happen. Using the new definition 3 and the fact that c is defined by the ratio α_1/α_2 we can give the following explanation of non-coordination level ranking when tf×idf weights are used.

Explanation 3. Non-coordination level ranking occurs if query terms that do *not* appear in a document are weighted too high compared to query terms that *do* appear in a document.

According to definition 3 terms that do not appear in a document are weighted proportional to the document frequency. If we choose a relatively high value for the constant α_1 then query terms that do not appear in a document will be weighted too high, possibly causing non-coordination level ranking.

4 Experimental results

This section briefly describes the results of a number of experiments with the linguistically motivated term weighting algorithm. A pilot experiment uses the relatively outdated Cranfield collection as reported earlier in [5]. Additional experiments using the TREC collection indicate that the new weighting algorithm outperforms the popular Cornell version of BM25.

4.1 The Cranfield collection

The Cranfield collection is a small collection (1398 documents) with a relatively large number of queries (255 queries). In the experiment we implemented a linguistically motivated probabilistic retrieval engine and a standard vector engine. Both engines used the same tokenisation and stemming of the words in the documents. As a test collection we used the Cranfield collection which was also used extensively in early experiments with the vector space model [15]. Table 3 lists the non-interpolated average precision averaged over 225 queries of the Cranfield collection for different values of α_1 and α_2.

Table 3. Experimental results on the Cranfield collection

weight		avg. precision
$\alpha_1 = 0.05$	$\alpha_2 = 0.95$	0.3832
$\alpha_1 = 0.2$	$\alpha_2 = 0.8$	0.4076
$\alpha_1 = 0.35$	$\alpha_2 = 0.65$	0.4198
$\alpha_1 = 0.5$	$\alpha_2 = 0.5$	0.4257
$\alpha_1 = 0.65$	$\alpha_2 = 0.35$	0.4305
$\alpha_1 = 0.8$	$\alpha_2 = 0.2$	0.4357
$\alpha_1 = 0.95$	$\alpha_2 = 0.05$	0.4247

To evaluate how our weighting scheme performs relative to other tf× idf weighting schemes with document length normalisation we implemented the vector space model with tfc.nfx weighting as proposed by Salton and Buckley [15]. The non-interpolated average precision averaged over 225 queries of this system was 0.4032 on the Cranfield collection.[3] The linguistically motivated system performs better for quite a wide range of different values of α_1 and α_2. The best performance in terms of average precision is approximately at $\alpha_1 = 0.85$.

4.2 Coordination level ranking

Cranfield has the following collection extrema: The smallest document is 18 words long, the longest 354 words. The maximum term frequency is 28 and the maximum document frequency 729. Following the arguments of Sect. 3.5 it is possible to calculate a lower bound on the ratio between α_1 and α_2 that will define coordination level ranking given a query of length 2. This leads

[3] Salton and Buckley [15] report a 3-point interpolated average precision of 0.3841. Our version of their system reaches a 3-point interpolated average precision of 0.4204 which is probably due to the use of a stemmer.

to a lower bound of 0.000525 on the ratio between α_1 and α_2 which corresponds roughly to $\alpha_1 = 0.0005$ and $\alpha_2 = 0.9995$. Although correct, the lower bound introduced by (11) is obviously not very useful for identifying proper values for α_1 and α_2. There are several reasons that might explain why the system performs optimally for much higher values of α_1:

1. Coordination level ranking does not lead to good average precision on the Cranfield collection.
2. The system does produce coordination level ranking, but the bound on the ratio between α_1 and α_2 is too low to be of any use.
3. The system does produce coordination level ranking, but the bound is not useful because the collection does not have very small queries (the average query length is about 9.5 words).

Additional experiments have to point out which reason or reasons actually explain the experimental results the best.

4.3 The TREC collection

Since the existence of the TExt Retrieval Conference (TREC), experiments have resulted in weighting algorithms that perform much better than tfc.nfx weighting. According to Voorhees and Harman [20], todays most popular weighting algorithm is the Cornell implementation of the Okapi BM25 algorithm [13, 18].

Table 4 displays the version of the BM25 weighting algorithm that was used for comparison. Table 5 lists the evaluation results in terms of average precision of the new weighting algorithm compared to the results of the Cornell/BM25 algorithm. We used two modern test collections provided by NIST via TREC. One is the ad hoc collection consisting of articles from the LA Times, the Financial Times, the Federal Register and Foreign Broadcast Information Service.[4] The other collection uses the English documents and English topics 1-24 of the TREC Cross-Language Information Retrieval (CLIR) collection consisting of AP Newswire articles. On both collections we used $\alpha_1 = 0.85$.

Table 4. Cornell version of BM25 weighting algorithm

$$\text{similarity}(Q, D) = \sum_{k=1}^{l} w_{qk} \cdot w_{dk}$$

$$w_{qk} = tf(t_k, q)$$

$$w_{dk} = \frac{tf(t_k, d) \cdot \log(\frac{N - df(t_k) + 0.5}{df(t_k) + 0.5})}{2 \cdot (0.25 + 0.75 \frac{\sum_t tf(t,d)}{\sum_{t,d} tf(t,d)/N}) + tf(t_k, d)}$$

Both weighting algorithms perform approximately equally well on Cranfield, but on the collections of more realistic size the linguistically motivated weighting algorithm outperforms the Cornell/BM25 weighting algorithm. The performance gain of 17% on the ad hoc collection is spectacular. Experiments using the ad hoc topics 351–400 show a significant, though less spectacular, improvement [7].

Table 5. Performance on three test collections

	average precision		
collection	BM25	new algorithm	% diff.
Cranfield	0.4386	0.4374	−0.3
TREC CLIR	0.3652	0.3723	+1.9
TREC ad-hoc	0.2508	0.2925	+16.6

Maybe more important than the results in terms of average precision is the the fact that the best value of the unknown parameter α_1 of the linguistically motivated weighting algorithm is stable across different test collections. On all three collections the best performance of the linguistically motivated weighting algorithm lies at approximately $\alpha_1 = 0.85$. For $\alpha_1 = 0.8$ and $\alpha_1 = 0.9$ the average precision on the English CLIR collection is respectively 0.3653 and 0.3714 and on the ad hoc collection respectively 0.2920 and 0.2884.

5 Conclusion and future plans

This paper has presented the linguistically motivated probabilistic model of information retrieval. Using estimation by linear interpolation which is often used in the field of statistical natural language processing we have been able to present a probabilistic interpretation of tf×idf term weighting. We have shown that this new interpretation leads to better understanding of the behaviour of tf×idf ranking. Experiments with the TREC collection show that a retrieval system that uses the new model outperforms the same system using the Cornell/BM25 weighting algorithm.

This paper has not presented the linguistically motivated model of information retrieval in its full strength. Although we claim that the most important modeling assumption of the model is that documents and queries are defined by an ordered sequence of terms, the assumption is not essential for the claims made in this paper. In future papers we will investigate two major information retrieval issues that require natural language processing techniques. The first issue is the use of phrases in information retrieval. The second issue is the problem of cross-language information retrieval.

Acknowledgements. The work reported in this paper is funded in part by the Dutch Telematics Institute project DRUID. The reported TREC evaluations were funded in part by the EU projects Twenty-One (IE 2108) and Pop-Eye (LE 4234).

I would like to thank the following people for their support. Franciska de Jong, Paul van der Vet and Wilbert Kallenberg for

[4] We left out the Congressional Records because NIST decided not to include this subcollection in TREC-7.

general advice, and David Hawking of the Australian National University for his advice on coordination level ranking. Special thanks go to Wessel Kraaij of TNO-TPD for running the TREC ad hoc experiments.

References

1. Clarke, C.L.A., Cormack, G.V., Tudhope, E.A.: Relevance ranking for one to three term queries. In: Proc. RIAO '97, 1997, pp. 388–400
2. Cooper, W.S.: Some inconsistencies and misidentified modeling assumptions in probabilistic information retrieval. ACM Trans. Information Systems 13:100–111, 1995
3. Croft, W.B., Turtle, H.R.: Text retrieval and inference. In: Jacobs, P. (ed.): Text-based Intelligent Systems. Lawrence Erlbaum, 1992, pp. 127–156
4. Hawking, D., Thistlewaite, P.: Relevance weighting using distance between term occurrences. Technical Report TR-CS-96-08, The Australian National University, 1996 `http://cs.anu.edu.au/techreports`
5. Hiemstra, D.: A linguistically motivated probabilistic model of information retrieval. In: Nicolaou, C., Stephanidis, C. (eds.): Proc. 2nd European Conference on Research and Advanced Technology for Digital Libraries (ECDL-2), 1998, pp. 569–584
6. Hiemstra, D., de Jong, F.M.G.: Cross-language retrieval in Twenty-One: using one, some or all possible translations? In: Proc. 14th Twente Workshop on Language Technology (TWLT-14), 1998, pp. 19–26
7. Hiemstra, D., Kraaij, W.: Twenty-One at TREC-7: Ad-hoc and cross-language track. In: Proc. 7th Text Retrieval Conference (TREC-7). NIST Special Publications, 1999
8. Manning, C., Schütze, H. (eds.): Statistical Natural Language Processing: Theory and Practice (draft). 1998. `http://www.sultry.arts.su.edu.au/manning/courses/statnlp`
9. Miller, D.R.H., Leek, T., Schwartz, R.M.: BBN at TREC-7: using Hidden Markov Models for information retrieval. In: Proc. 7th Text Retrieval Conference, TREC-7. NIST Special Publications, 1999
10. Mood, A.M., Graybill, F.A. (eds.): Introduction to the Theory of Statistics. Second edition. McGraw-Hill, 1963
11. Ponte, J.M., Croft, W.B.: A language modeling approach to information retrieval. In: Proc. 21st ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98), 1998
12. Robertson, S.E., Sparck Jones, K.: Relevance weighting of search terms. J. American Society Information Science 27:129–146, 1976
13. Robertson, S.E., Walker, S.: Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In: Proc. 17th ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '94), 1994, pp. 232–241
14. Rose, D.E., Stevens, C.: V-twin: A lightweight engine for interactive use. In: Voorhees, E.M., Harman, D.K. (eds.): Proc. 5th Text Retrieval Conference TREC-5, NIST Special Publication 500-238, 1997, pp. 279–290
15. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing & Management 24(5):513–523, 1988
16. Salton, G., McGill, M.J. (eds.): Introduction to Modern Information Retrieval. McGraw-Hill, 1983
17. Salton, G., Yang, C.S.: On the specification of term values in automatic indexing. J. Documentation 29(4):351–372, 1973
18. Singhal, A., Salton, G., Mitra, M., Buckley, C.: Document length normalization. Technical Report TR95-1529, Cornell University, 1995. `http://cs-tr.cs.cornell.edu/`
19. Strzalkowski, T., Sparck Jones, K.: NLP track at TREC-5. In: Voorhees, E.M., Harman, D.K. (eds): Proc. 5th Text Retrieval Conference TREC-5, NIST Special Publication 500-238, 1997, pp. 97–101
20. Voorhees, E.M., Harman, D.K.: Overview of the 6th text retrieval conference. In: Proc. 6th Text Retrieval Conference TREC-6, NIST Special Publication 500-240, 1998, pp. 1–24
21. Wilkinson, R., Zobel, J., Sacks-Davis, R.: Similarity measures for short queries. In: Harman, D.K. (ed.): Proc. 4th Text Retrieval Conference TREC-4, NIST Special Publication 500-236, 1996, pp. 277–286