Set-Based Model: A New Approach for Information Retrieval

Bruno Pôssas
Universidade Federal de Minas Gerais
30161-970 Belo Horizonte-MG, Brazil
bavep@dcc.ufmg.br

Nivio Ziviani
Universidade Federal de Minas Gerais
30161-970 Belo Horizonte-MG, Brazil
nivio@dcc.ufmg.br

Wagner Meira Jr.
Universidade Federal de Minas Gerais
30161-970 Belo Horizonte-MG, Brazil
meira@dcc.ufmg.br

Berthier Ribeiro-Neto
Universidade Federal de Minas Gerais
30161-970 Belo Horizonte-MG, Brazil
berthier@dcc.ufmg.br

ABSTRACT

The objective of this paper is to present a new technique for computing term weights for index terms, which leads to a new ranking mechanism, referred to as set-based model. The components in our model are no longer terms, but termsets. The novelty is that we compute term weights using a data mining technique called association rules, which is time efficient and yet yields nice improvements in retrieval effectiveness. The set-based model function for computing the similarity between a document and a query considers the termset frequency in the document and its scarcity in the document collection. Experimental results show that our model improves the average precision of the answer set for all three collections evaluated. For the TReC-3 collection, our set-based model led to a gain, relative to the standard vector space model, of 37% in average precision curves and of 57% in average precision for the top 10 documents. Like the vector space model, the set-based model has time complexity that is linear in the number of documents in the collection.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval—*Retrieval models*; H.2.8 [**Information Systems**]: Database Applications—*Data mining*; F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic—*Set theory*

General Terms

Algorithms, Measurement, Performance, Experimentation

Keywords

Information retrieval models, closed association rule mining, weighting index term co-occurrences, data mining, set theory

SIGIR'02, August 11-15, 2002, Tampere, Finland.

1. INTRODUCTION

In the area of information retrieval, one popular model to locate answers that humans judge to be relevant is the vector space model. In the vector space model query terms and documents are represented as weighted vectors in a vector space, and the answers to a user query are the documents whose representative vectors have highest similarity to the query vector. The computation of similarities is based on weights assigned to the index terms that compose document and query vectors. Much of the success of the model is due to the long-term research efforts of Salton and his associates [18, 19].

The term weights can be calculated in many different ways and finding good term weights is a continuing challenge. The best known term weighting schemes use weights that are function of (i) the number of times an index term occurs in a document and (ii) the number of documents of the collection an index term occurs. Such term-weighting strategies are called $tf \times idf$ (term frequency times inverse document frequency) schemes [4, 22].

In this paper we propose a new model for computing index term weights, based on set theory, and for ranking documents. This model is referred to as *set-based model*. The set-based model uses a term weighting scheme based on association rules theory [1]. Association rules are interesting because they provide all the elements of the $tf \times idf$ scheme in an algorithmically efficient and parameterized approach. Also, they naturally provide for quantification of representative term co-occurrence patterns, something that is not present in the $tf \times idf$ scheme.

We evaluated and validated the set-based model through experimentation using three test collections. We compared our model against the vector space model and the generalized vector space model [24]. Our experimental results show that the set-based model yields higher retrieval performance, which is significantly superior for all the collections considered. For the TReC-3 collection [8], containing 3,225 megabytes of text and approximately 1 million documents, the set-based model yields average precision curves that are 37% higher than the vector space model. When only the first 10 answers are considered, this margin goes up to 57%. For the Wall Street Journal collection, containing 509 megabytes of text and 173,252 documents, the set-based model, in comparison with the generalized vector space model, is approximately 22% superior in average precision and approximately 16% superior in precision at the first 10 documents in the answer.

The set-based model is also competitive in terms of computing performance. Both the vector space and the set-based models are linear on the number of documents in the collection. The vector space model is $O(tN)$, where t is the size of the vocabulary collection and N is the number of documents in the collection. The set-based model is $O(cN)$, where c is the number of closed termsets, a value that is $O(2^{|q|})$, where $|q|$ is the number of terms in the query. These are worst case measures and the average measures for the constants involved are much smaller. For the TReC-3 collection, using the 151-200 range of topics, where average number of terms in the query is 22.43, the increase in the average response time for the set-based model over the vector space model is approximately 80%. These are relevant facts from both theoretical and practical points of view, considering that we introduced a new and more complex model for weighting index terms, which will often improve the quality of the results.

The remaining of the paper is organized as follows. In the next section we discuss some related work. Section 3 describes the method for computing co-occurrences among query terms based on a variant of association rules. The set-based model is presented in the Section 4. In section 5 we describe the reference collections and the experimental results comparing the vector-space model, the generalized vector-space model, and our model. Finally, we present some conclusions and future work in Section 6.

2. RELATED WORK

The vector space model was proposed by Salton [18, 15], and different weighting schemes were presented [19, 25, 16]. In the vector space model, index terms are assumed to be mutually independent. The independence assumption leads to a linear weighting function which, although not necessarily realistic, is ease to compute. Simple term weighting was used early on by Salton and Lesk [18]. Spark Jones introduced the *idf* factor [21], and Salton and Yang verified its effectiveness for improving retrieval [19].

Different approaches to account for co-occurrence among index terms during the information retrieval process have been proposed. Raghavan and Yu [13] use a statistical analysis of a set of queries in relation to relevant and non-relevant document sets to establish positive and negative correlations among index terms. The work by van Rijsbergen [14] constructs a probabilistic model incorporating dependence among index terms, and experimental results were later presented in a companion paper [9]. The extent to which two index terms depend on one another is derived from the distribution of co-occurrences in the whole collection or in the relevant and non-relevant sets, leading to a non-linear weighting function. As shown in [17], the resulting formula for computing the dependency factors in [14, 9] does not seem computationally feasible, even for a relatively small number of index terms. The work in [6] presents a study of the necessity of term dependence in a query space for weighted based retrieval.

The work in [24, 23] presents an interesting approach to compute index term correlations based on automatic indexing schemes, defining a new information retrieval model called generalized vector space model. The work shows that index term vectors can be explicitly represented in a 2^t-dimensional vector space such that index term correlations can be incorporated into the vector space model in a straightforward manner. It represents index term vectors from a basic set of orthogonal vectors called min-terms, where each min-term represents a pattern of index term co-occurrence inside documents. The vector representing an index term k_i is obtained by adding all min-terms related to k_i. The model is not computationally feasible for moderately large collections because there are 2^t possible min-terms, where t is the number of terms in the collection vocabulary. Extensions of the generalized vector space model were used in [3, 11].

Our work differs from the above related works in the following aspects. First, determination of index term weights is derived directly from association rules theory, which naturally considers representative patterns of term co-occurrence. Second, experimental results show significant and consistent improvements in average precision curves in comparison to the standard vector space model and to the generalized vector space model. This improvement in average precision does not always occur in the generalized vector space model because exhaustive application of term co-occurrences to all documents in the collection might eventually hurt the overall effectiveness and performance. Third, the set-based model algorithm has computational costs that are linear on the number of documents in the collection. Thus, the implementation of our model is practical and efficient, with processing times close to the times to process the standard vector space model.

The work in [5] introduces a theoretical framework for the association rules mining based on a boolean model of information retrieval known as Boolean Retrieval Model. Our work differs from that presented in [5] in the following way. In our work we use association rules to define a new information retrieval model that provides not only the main term weights and assumptions of the $tf \times idf$ scheme, but also provides for quantification of term co-occurrence.

3. CLOSED TERMSETS

In this section we introduce the concept of closed termsets as a basis for computing term weights. Further, we analyze how closed termsets can be generated from a document inverted list and briefly analyze its computational costs.

3.1 Concepts

Let $T = \{\mathrm{k}_1, \mathrm{k}_2, ..., \mathrm{k}_t\}$ be the vocabulary of a collection of documents D, that is, the set of t unique terms that may appear in a document from D. There is a total ordering among the vocabulary terms, which is based on the lexicographical order of terms, so that $\mathrm{k}_i < \mathrm{k}_{i+1}$, for $1 \leq i \leq t-1$. We define an n-termset s as an ordered set of n unique terms, such that $s \subseteq T$ and the order among terms follows the aforementioned total ordering. Let $S = \{\mathrm{s}_1, \mathrm{s}_2, ..., \mathrm{s}_{2^t}\}$ be the vocabulary-set of a collection of documents D, that is, the set of 2^t unique termsets that may appear in a document from D. Each document j from D is characterized by a vector in the space of the termsets.

With each termset s_i, $1 \leq i \leq 2^t$, we associate an inverted list ls_i composed of identifiers of the documents containing that termset. We also define the frequency ds_i of a termset s_i as the number of occurrences of s_i in D, that is, the number of documents where $s_i \subseteq d_j$ and $d_j \in D$, $1 \leq j \leq N$. The frequency ds_i of a termset s_i is the length of its associated inverted list ($\mid ls_i \mid$).

Definition 1. A termset s_i is a frequent termset if its frequency ds_i is greater than or equal to a given threshold, which is known as *support* in the scope of association rules [1], and refered as minimal frequency in this work.

Definition 2. A closed termset cs_i is a frequent termset that is the largest termset among the termsets that are subsets of cs_i and occur in the same set of documents. That is, given a set $\mathcal{D} \subseteq D$ of documents and the set $\mathcal{S}_{\mathcal{D}} \subseteq S$ of termsets that occur in all documents from $\mathcal{D}$ and only in these, a closed termset cs_i satisfies the property that $\nexists s_j \in \mathcal{S}_{\mathcal{D}} | cs_j \subset s_j$.

Definition 3. A maximal termset ms_i is a frequent termset that is not a subset of any other frequent termset. That is, given the set $\mathcal{S_F} \subseteq S$ of frequent termsets that occur in all documents from $\mathcal{D}$, a maximal termset ms_i satisfies the property that $\nexists s_j \in \mathcal{S_F} | ms_i \subset s_j$.

For sake of information retrieval, closed termsets are interesting because they represent a reduction on the computational complexity and on the amount of data that has to be analyzed, without loosing information, since all termsets in a closure are represented by the respective closed termset.

It is proven that the set of maximal termsets associated with a document collection are the minimum amount of information necessary to derive all frequent termsets associated with a collection [26]. However, they do not embed the same amount of information that closed termsets do, since the occurrence pattern of each termset is discarded, making difficult its use for finding interesting term co-occurrences.

Documents (D)

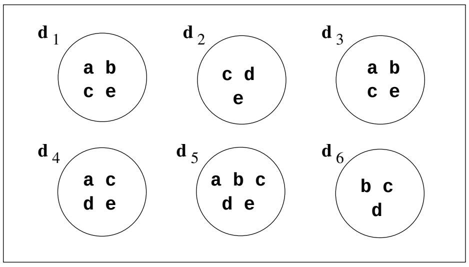

Figure 1: Sample document collection

Next we present a simple example to illustrate the concepts of frequent, closed and maximal termsets. Consider a vocabulary of five terms $T = \{a, b, c, d, e\}$, and a collection D of six documents d_j, $1 \leq j \leq 6$, that contain these terms, i.e., $D = \{(a, b, c, e)$, (c, d, e), (a, b, c, e), (a, c, d, e), (a, b, c, d, e), $(b, c, d)\}$, as depicted in Figure 1. Table 1 shows all frequent and closed termsets for the sample document collection and their respective frequencies. The minimum frequency used for selecting the frequent termsets is equal to 50%, that is, a termset is frequent if it appears in at least three of the six documents. Notice that the number of frequent termsets, although potentially very large, is usually small in natural language texts. This number is a function of the minimum frequency and document collection characteristics. It is possible to vary the number of frequent termsets by changing the minimum frequency employed. Regarding the closed termsets, even in this small example, we can see that the number of closed termsets is significantly smaller than the number of frequent termsets. In our example, there are 20 frequent termsets and just 8 closed termsets, and 2 maximal termsets.

A major advantage of using closed termsets instead of frequent termsets is that they can be generated very efficiently, as proved in [26]. So far, there is just an upper bound on the number of frequent termsets, which is $O(nr2^l)$, where n is the number of documents in the collection, r is the number of maximal frequent termsets, and l is the length of the largest frequent termset. In practice, since the query processing is driven by the query, the largest frequent termset is bounded by the number of terms in the query. Since the number of closed termsets is at most equal to the number of frequent termsets, we may also use this bound as an upper limit of the number of closed termsets. As we show in Section 5, in practice the number of closed termsets is significantly smaller than the number of frequent termsets. For our example, the upper bound of the number of frequent termsets is $Onr2^l = 6 * 2 * 2^4 = 96$. However, its total number was 20, which is significantly smaller than its upper bound. Regarding the closed termsets, we can see that its number, 8, is smaller than the number of frequent termsets.

Table 1: Frequent, closed, and maximal termsets for the sample document collection

Frequency (ds)	Frequent Termsets	Closed Termsets	Maximal Termsets
100% (6)	c	c	
83% (5)	e,ce	ce	
67% (4)	a,ac,ae,ace	ace	
67% (4)	cd	cd	
67% (4)	b,bc	bc	
67% (4)	d,cd	cd	
50% (3)	ab,abc,abe be,bce,abce	acbe	abce
50% (3)	de,cde	cde	cde

3.2 Algorithm Description

Determining closed termsets is a problem very similar to mining association rules and the algorithms employed for the latter are our starting point [1]. Our approach is based on an efficient algorithm for association rule mining, called CHARM [26], which has been adapted to handle terms and documents instead of items and transactions, respectively.

The algorithm for generating termsets is basically an incremental enumeration algorithm that employs a very powerful pruning strategy. More specifically, we start by verifying whether the frequency of 1-termsets is above a given threshold. This approach is based on the fact that, if a term is not frequent, none of the higher-order termsets that include it will be. These frequent 1-termsets will also be defined as closed. We then start grouping frequent termsets and verifying two criteria that are used to determine whether these new termsets are closed or not.

We determine $n + 1$ termsets from n termsets, as follows. The process starts by determining all pairs of termsets (s_i, s_j), $1 \leq i, j \leq 2^t$, that have the same first $n - 1$ terms. A new termset is determined by the union of these sets $(s_i \cup s_j)$, creating an $n + 1$ termset s_{new}. The list of documents where s_{new} appears can be easily determined by intersecting the lists of the two concatenated sets $(l_{new} = l_i \cap l_j)$. After this, the first criterion that verifies whether the new termset is frequent, and was introduced by the original Apriori algorithm [2], is employed. An n-termset may be frequent only if all of its $n - 1$-termsets are also frequent. For instance, we may enumerate the 4-termset ({a b c e}) based on the termsets {a b c} and {a b e} and we apply the frequency criterion by verifying whether the other 3-termsets of the new termset (in this case {a b e} and {b c e}) are also frequent. Next, the inverted list associated with the new termset is generated and it is verified whether its length is above threshold.

The second criterion checks whether a frequent termset f being verified is closed. A termset f is defined as closed if there is no closed termset that is also a closure of f. In this case, all current closed termsets are tested for having f as its closure, being discarded if such condition holds. Otherwise, f is discarded, since its closure is already determined. In our sample collection, for example, the 1-termset $\{a\}$ is initially defined as closed. However, when we combine the 1-termsets {a} and {c} into {a c}, we discard {a}, since they occur in the same set of documents. The same happens when we enumerate {a c e} and find out that {a c} and {a c e} occur in the same documents, allowing us to discard {a c}.

In the next section we describe how we use closed termsets for retrieving documents efficiently.

4. SET-BASED MODEL

In our set-based model, a document is described by a set of closed termsets, extracted from the document itself. With each closed termset we associate a pair of weights representing (a) its importance in each document and (b) its importance in the whole document collection. In a similar way, a query is described by a set of closed termsets with a weight representing the importance of each closed termset in the query.

The algebraic representation of the set of closed termsets for both documents and queries correspond to vectors in a 2^t-dimensional Euclidean space, where t is equal to the number of unique index terms in the document collection. However, in practice, we use an efficient algorithm to obtain closed termsets which is based on association rules theory, thus reducing the number of closed termsets to a number that is linear on the number of documents in the collection (see Section 3 and Section 5.2).

4.1 Termset Weights

A successful ranking algorithm based on index terms weighting has to take into account the following three criteria. First, it has to consider the number of times that an index term occurs in a document. Second, emphasis must be given to scarce terms, and this can be accomplished by reducing term weights for terms that appear in many documents. Third, long documents with many terms will naturally be favored by the ranking process because they are likely to contain more of any given list of query terms, and usually a normalization factor is introduced to discount the contribution of long documents.

Term weights can be calculated in many different ways [19, 25, 16]. For the first two problems pointed out above, the best known term weighting schemes use weights that are function of (i) $tf_{i,j}$, the number of times that an index term i occurs in a document j and (ii) df_i, the number of documents that an index term i occurs in the whole document collection. Such term-weighting strategy is called $tf \times idf$ schemes. The expression for idf_i represents the importance of term i in the collection, it assigns a high weight to terms which are encountered in a small number of documents in the collection, supposing that rare terms have high discriminating value.

In the set-based model, the association rules scheme naturally provides for quantification of representative patterns of term co-occurrences, something that is not present in the $tf \times idf$ approach. To determine the weights associated with the closed termsets in a document or in a query, we also use the number of occurrences of a closed termset in a document, in a query, and in the whole collection. Formally, the weight of a closed termset i in a document j is defined as:

$$w_{i,j} = sf_{i,j} \times ids_i = sf_{i,j} \times \log \frac{N}{ds_i} \quad (1)$$

where N is the number of documents in the collection, $sf_{i,j}$ is the number of occurrences of the closed termset i in document j (obtained from the index), and ids_i is the inverted frequency of occurrence of the closed termset in the collection. $sf_{i,j}$ generalizes $tf_{i,j}$ in the sense that it counts the number of times that the closed termset s_i appears in document j. The component ids_i also carries the same semantics of idf_i, but accounting for the cardinality of the closed termsets as follows. High-order closed termsets usually have low frequency, resulting in large inverted frequencies. Thus, this strategy assigns large weights to closed termsets that appear in small number of documents, that is, rare closed termsets result in greater weights.

4.2 Similarity Calculation

Since documents and queries are represented as vectors, we assign a similarity measure to every document containing any of the query closed termsets, defined as the normalized scalar product between the set of document vectors $\vec{d_j}$, $1 \leq j \leq N$, and the query vector $\vec{q}$. The similarity between a document d_j and a query q is defined as:

$$sim(q, d_j) = \frac{\vec{d_j} \bullet \vec{q}}{|\vec{d_j}| \times |\vec{q}|} = \frac{\sum_{s \in C_q} w_{s,j} \times w_{s,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}},$$

where $w_{s,j}$ is the weight associated with the closed termset s in document d_j, $w_{s,q}$ is the weight associated with the closed termset s in query q, C_q is the set of all closed termsets such that all $s \subseteq q$, $w_{i,j}$ is the weight of term i in document d_j, and $w_{i,q}$ is the weight of term i in query q. We observe that the normalization (i.e., the factors in the denominator) is done using the terms (instead of closed termsets) that compose the query and document vectors. This is important to reduce computational costs because computing the norm of a document d using closed termsets might be prohibitively costly. It solves the third problem pointed out in Section 4.1 because it accomplishes the effect of penalizing large documents, the major objective of normalization by document norms. Further, it allows efficient computation. Our experimental results corroborate the validity of adopting this alternative form of normalization.

4.3 Query Mechanisms

In this section we describe a simple algorithm for implementing a search mechanism based on the set-based model and the indexing information.

For large document collections, the cost to evaluate the similarity measure is potentially high, requiring the reading and processing of whole inverted lists for each query closed termset. This task may have a high cost since some query terms might occur in a large number of documents in the collection. One of the most effective techniques to compute an approximation of the similarity measure without modifying the final ranking is presented in [12]. It uses thresholds for early recognition of which documents are likely to be highly ranked to reduce costs. In the set-based model we employ a variant of this pruning scheme adapted to handle closed termsets instead of terms.

The steps performed by the set-based model to the calculation of the similarity metrics are equivalent to the standard vector space model. First we create the data structures that are used for calculating the document similarities among all closed termsets s_i of a document d. Then, for each query term, we retrieve its inverted list, and determine the first frequent termsets, i.e., the frequent termsets of size equal to 1, applying the minimal frequency threshold. The next step is the enumeration of all closed termsets based on the 1-termsets, as presented in Section 3.2. After enumerating all closed termsets, we employ the pruning scheme presented by Persin [12] to the evaluation of its inverted lists, allowing to discard closed termsets that will not influence the final ranking very early. After evaluating the closed termsets, we normalize the document similarities by dividing each document similarity by the respective document's norm. The final step is to select the k largest similarities and return the corresponding documents.

5. EXPERIMENTAL RESULTS

In this section we describe the experimental results for the evaluation of the set-based model (SBM) in terms of both effectiveness and computational efficiency. Our evaluation is based on a comparison to the standard vector space model (VSM) and to the generalized vector-space model (GVSM). We first present the experimental framework and the reference collections employed, and then discuss the retrieval performance and the computational efficiency of each model.

In our evaluation we use three reference collections, CFC [20], TReC-3 [8] and WSJ, respectively. The WSJ collection is a subcollection of TReC-3, and is used in our experiments because the full TReC-3 collection is too large for running the generalized vector space model. Table 2 presents the main features of these collections, which comprise not only the documents, but also a set of example queries and the relevant responses for each query, as selected by experts. We quantify the retrieval effectiveness of the various approaches through standard measures of average recall and precision. The computational efficiency is evaluated through the query response time, that is, the processing time to select and rank the documents for each query.

Table 2: Characteristics of the reference collections

Characteristics	Collection		
	CFC	WSJ	TReC-3
Number of Documents	1,240	173,252	1,078,166
Number of Distinct Terms	2,105	230,902	1,016,709
Number of Queries	100	300	300
Average Terms per Query	3.82	18.88	22.43
Average Relevants per Query	29.04	235.99	286.89
Size (MB)	1.9	509	3,225

The experiments were performed on a Linux-based PC with a Intel Pentium III 1.0 GHz processor and 1.5 GBytes RAM. Next we present the results obtained for the three collections, which are characterized in Table 2.

5.1 Retrieval Performance

We start our evaluation by verifying the precision-recall curves for each model when applied to each of the three collections. Each curve quantifies the precision as a function of the percentage of documents retrieved (recall). The GVSM could not be evaluated for the TReC-3 collection, because of the cost of the min-term building phase, which is exponential in the size of the vocabulary, making the computational cost of the associated experiments not feasible. The results presented for SBM are the best for a range of minimal frequencies from 1 to 30 documents (see Figure 5).

As we can see in Figures 2, 3, and 4, SBM yields better precision than both VSM and GVSM, regardless of the collection and of the recall level. Further, the gains increase with the size of the collection, because large collections allow computing a more representative set of closed termsets.

We confirm such observations by verifying the overall average precision achieved for each model in each collection, which is presented in Table 3. We also present, in the same table, the gains provided by SBM over the other two models in terms of overall precision. The gains ranged from 8.54% to 36.86%, increasing with the size of the collection and being consistently greater for the SBM. We should note that SBM provides significant gains over GVSM, although both exploit term co-occurrences. We believe that SBM outperforms GVSM because it considers just some interesting co-occurrence patterns that are uncovered by the closed termsets.

The set-based model provides even greater gains when we verify the average precision for the top 10 documents retrieved. This metric estimates the user "satisfaction" while using search engines, for instance. The results are presented in Table 4. We can see that the gains range from 15.62 to 56.80% and the highest gain is associated to the largest collection (TReC-3), showing that the SBM is specially powerful for handling large amounts of information. Again, although the gains over the GVSM are smaller, they are still very significative, showing that SBM not only improves the overall precision, but also ranks more relevant documents at the top.

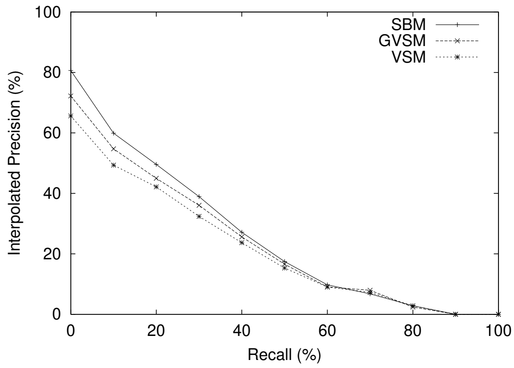

Figure 2: Recall-precision for CFC

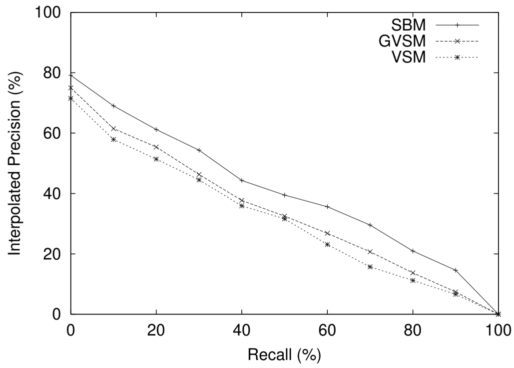

Figure 3: Recall-precision for WSJ

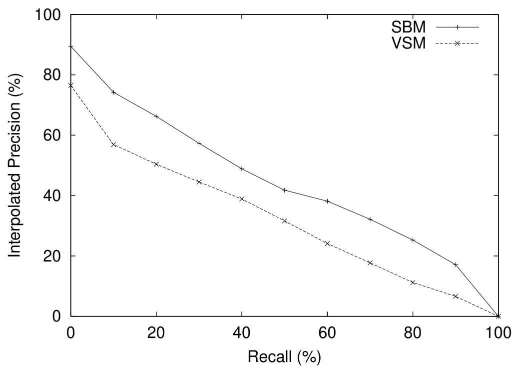

Figure 4: Recall-precision for TReC-3

Table 3: Average precision curves and gain provided by the SBM model

Collection	Average Precision (%)			SBM Gain(%)	
	VSM	*GVSM*	*SBM*	*VSM*	*GVSM*
CFC	22.42	24.47	26.56	18.47	8.54
WSJ	31.76	34.27	41.78	31.55	21.91
TReC-3	32.58	*	44.59	36.86	*

* The GVSM could not be evaluated for the TReC-3 collection due to the exponential cost of the min-term build phase

Table 4: Average precision curves for top 10 documents and gains provided by the SBM model

Collection	Average Precision at 10 (%)			SBM Gain(%)	
	VSM	*GVSM*	*SBM*	*VSM*	*GVSM*
CFC	10.97	12.93	16.02	46.03	23.90
WSJ	12.71	16.58	19.17	50.82	15.62
TReC-3	13.66	*	21.42	56.80	*

* The GVSM could not be evaluated for the TReC-3 collection due to the exponential cost of the min-term build phase

One of the key features of the set-based model is the possibility of controlling the minimal frequency threshold of a termset. It is interesting to note that the variation of the minimal frequency allows us to exploit the trade-off between precision and the number of termsets taken into consideration. We verified how variations in the minimal frequency threshold employed affect the precision achieved for the TReC-3 collection. We performed a set of SBM executions where the minimal frequency threshold is fixed for all queries and varies from 1 to 30 documents. The results are illustrated in Figure 5 and we show that the SBM precision is significantly affected by the minimal frequency employed. Initially, an increase in the minimal frequency results in better precision, achieving a maximum precision for minimal frequencies between 8 and 10 documents. When we increase the minimal frequency beyond 10, we observe that the precision decreases until it reaches almost 0. This behavior can be explained as follows. First, an increase in the minimal frequency causes irrelevant termsets to be discarded, resulting in a better precision. When the frequency gets above 10, then relevant termsets start to be discarded, leading to a reduction in terms of precision.

In summary, our set-based model (SBM) is the first information retrieval model that exploits term correlations effectively and provides significant gains in terms of precision, regardless of the size of the collection and of the size of the vocabulary. In the next section we discuss the computational costs associated with SBM.

5.2 Computational Efficiency

In this section we compare our model to the standard vector space model and the generalized vector space model regarding the query response time, in order to evaluate its feasibility in terms of computational costs. This is important because one major limitation of existing models that account for term correlations is their computational cost. Several of these models cannot be applied to large or even mid-size collections since their costs increase exponentially with the vocabulary size.

We start our evaluation by verifying the theoretical upper bounds

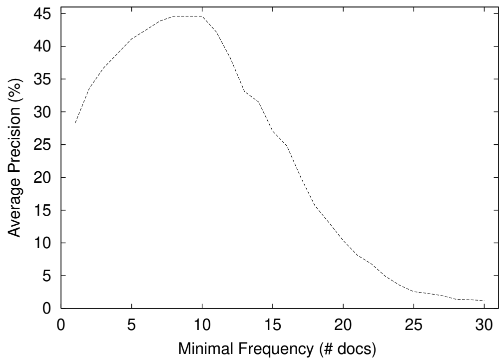

Figure 5: Impact of varying minimal frequency threshold on average precision for TReC-3

on the number of operations performed by VSM and SBM. The complexity of both models is linear with respect to the number of documents in the collection. Formally, the upper bound on the number of operations performed for satisfying a query in the vector space model is $O(|q| \times N)$, where $|q|$ is the number of terms in the query and N is the number of documents in the collection. The worst case scenario for the vector space model is a query comprising the whole vocabulary t ($|q| = t$), which results in a merge of all inverted lists in the collection. As discussed in Section 3, the complexity of the set-based model is $O(r2^l N)$, where r is the number of maximal termsets and l is the largest termset associated with the collection for a given minimal frequency.

The comparison between t and $r2^l$ allows quantifying the worst case cost increase of the SBM model, but the latter is dependent on the document collection and minimal frequency employed. We calculated r and l for each collection employing a 90% confidence interval and they are presented in Table 5. For the SBM model, the upper bound is $29.84 \times 2^{12.13} \times number - of - documents$ operations.

Table 5: Number of maximal termsets and largest frequent termset

Collection	r	l
CFC	3.38 ± 0.27	3.28 ± 0.56
WSJ	19.01 ± 1.15	11.89 ± 1.88
TReC-3	29.84 ± 1.99	12.13 ± 2.39

In fact, for SBM, we use the number of closed termsets, whose worst case is $r2^l$. We determined the average number of closed termsets and the average list sizes while using SBM and the results show that the average case scenario is much better than the worst case just presented, as we can see in Table 6. Considering the TReC-3 collection, the average number of closed termsets is 4,081.25, giving an average upper bound of 4,081.25 × 162.06 operations for the SBM model, where the number 162.06 is the average size of the inverted lists. The number 4,081.25 of closed termsets is approximately 2^l, where $l = 12.13$ from Table 5, Considering the TReC-3 collection, the average number of terms per query is 22.43, resulting in an average upper bound of 22.43 × 162.06 operations for the VSM model.

To validate our analysis, we compared the response time for the models and collections considered, which are summarized in Table 7. We also calculated the increase of the response time of GVSM and SBM when compared to VSM. We observe that SBM

Table 6: Average number of closed termsets and inverted list size

Collection	Closed Termsets	Inverted List Size
CFC	3.14	7.22
WSJ	3,217.28	140.10
TReC-3	4,081.25	162.06

results in a response time increase from 8.7 to 80.1% for the reference collections evaluated when compared to VSM. The results confirm the complexity analysis performed, indicating that the time for processing the lists dominates the processing costs for both VSM and SBM, and the SBM is comparable to VSM in terms of computational costs. As expected, the increase for GVSM is much greater, ranging from 243.5% to 469.9%.

Table 7: Average response time and response time increase

Collection	Average Response Time (s)			Increase(%)	
	VSM	*GVSM*	*SBM*	*GVSM*	*SBM*
CFC	0.0023	0.0056	0.0025	243.5	8.7
WSJ	0.4286	2.0143	0.6296	469.9	46.9
TReC-3	1.2732	*	2.2930	*	80.1

* The GVSM could not be evaluated for the TReC-3 collection due to the exponential cost of the min-term build phase

We identify two main reasons for the relatively small increase in execution time for SBM. First, there is a small number of query related termsets in the reference collections. As a consequence, the inverted lists associated tend to be small (as presented in Table 6) and are usually manipulated in main memory in our implementation of SBM. Second, we employ pruning techniques that discard irrelevant termsets early in the computation, as described in Section 4.2.

As mentioned, we may control the number of closed termsets, and thus the query processing time, by varying the minimal frequency threshold. We varied the minimal frequency used in SBM and measured the response time. Figure 6 shows the response time as a function of the minimal frequency employed, where we see that the increase in the minimal frequency causes significant reductions on the response time, at a possible penalty in average precision curves.

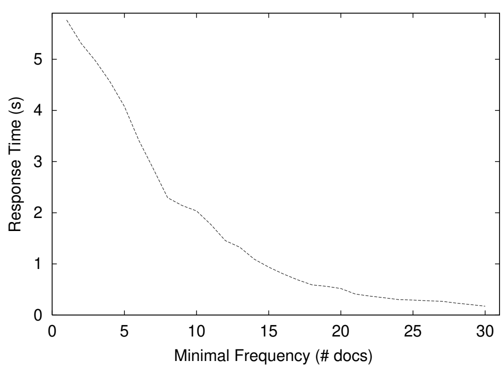

Figure 6: Impact of minimal frequency on response time for TReC-3

6. CONCLUSIONS

We presented a new approach to compute term weights, which is based on the theory of association rules. We showed that it is possible to significantly improve retrieval effectiveness, while keeping extra computational costs small. The computation of frequent termsets enumerated by an algorithm to generate association rules lead to a direct extension of the vector space model. The formal framework we adopted allowed naturally considering relevant patterns of term co-occurrence, with minimal changes to the standard vector space model formulation.

We evaluated and validated our proposed extension for the vector space model using three test collections. We showed through curves of recall versus precision that the new model presents results that are superior for all the collections considered and that the additional computational costs are acceptable. In fact, the computational costs measured allow the use of our set-based model with general large document collections.

Recent experimentation [10, 7] indicates that measurements involving the distance between term occurrences may provide good relevance assessments. For future work we will extend the set-based model to account for the proximity information about query terms in documents. We will also investigate the behavior of our model when applied to larger document collections and for collections formed by Web documents.

7. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD International Conference Management of Data*, pages 207-216, Washington, D.C., May 1993.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *The 20th International Conference on Very Large Data Bases*, pages 487-499, Santiago, Chile, September 1994.

[3] A. H. Alsaffar, J. S. Deogun, V. V. Raghavan, and H. Sever. Enhancing concept-based retrieval based on minimal term sets. In *Journal of Intelligent Information Systems*, volume 14(2–3), pages 155–173, 2000.

[4] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[5] P. Bollmann-Sdorra, A. Hafez, and V. V. Raghavan. A theoretical framework for association mining based on the boolean retrieval model. In *Data Warehousing and Knowledge Discovery: Third International Conference*, pages 21–30, Munich, Germany, September 2001.

[6] P. Bollmann-Sdorra and V. V. Raghavan. On the necessity of term dependence in a query space for weighted retrieval. In *Journal of the American Society for Information Science*, volume 49(13), pages 1161–1168, November 1998.

[7] C. L. A. Clarke, G. V. Cormack, and F. J. Burkowski. Shortest substring ranking. In *In Fourth Text REtrieval Conference (TREC-4)*, pages 295–304, Gaithersburg, Maryland, USA, November 1995.

[8] D. Harman. Overview of the third text retrieval conference (trec 3). In *The Third Text Retrieval Conference*, 1993.

[9] D. J. Harper and C. J. V. Rijsbergen. An evaluation of feedback in document retrieval using co-occurrence data. In *Journal of Documentation*, volume 34, pages 189–216, 1978.

[10] D. Hawking and P. Thistlewaite. Proximity operators - so near and yet so far. In *In Fourth Text REtrieval Conference (TREC-4)*, pages 131–144, Gaithersburg, Maryland, USA, November 1995.

[11] M. Kim, A. H. Alsaffar, J. S. Deogun, and V. V. Raghavan. On modeling of concept based retrieval in generalized vector spaces. In *Proceedings International Symposium on Methods of Intelligent Systems*, pages 453–462, Charlote, N.C., USA, October 2000.

[12] M. Persin, J. Zobel, and R. Sacks-Davis. Filtered document retrieval with frequency-sorted indexes. In *Journal of the American Society of Information Science*, pages 749–764, 1996.

[13] V. V. Raghavan and C. T. Yu. Experiments on the determination of the relationships between terms. In *ACM Transactions on Databases Systems*, volume 4, pages 240–260, 1979.

[14] C. J. V. Rijsbergen. A theoretical basis for the use of co-occurrence data in information retrieval. In *Journal of Documentation*, volume 33, pages 106–119, 1977.

[15] G. Salton. *The SMART retrieval system – Experiments in automatic document processing*. Prentice Hall Inc., Englewood Cliffs, NJ, 1971.

[16] G. Salton and C. Buckley. Term-weighting approaches in automatic retrieval. In *Information Processing and Management*, volume 24(5), pages 513–523, 1988.

[17] G. Salton, C. Buckley, and C. T. Yu. An evaluation of term dependencies models in information retrieval. In *The 5th ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 151-173, 1982.

[18] G. Salton and M. E. Lesk. Computer evaluation of indexing and text processing. In *Journal of the ACM*, volume 15(1), pages 8–36, January 1968.

[19] G. Salton and C. S. Yang. On the specification of term values in automatic indexing. In *Journal of Documentation*, volume 29, pages 351-372, 1973.

[20] W. M. Shaw, J. B. Wood, R. E. Wood, and H. R. Tibbo. The cystic fibrosis database: Content and research opportunities. In *Library and Information Science Research*, volume 13), pages 347–366, 1991.

[21] J. K. Sparck. A statistical interpretation of term specificity and its application in retrieval. In *Journal of Documentation*, volume 28(1), pages 11-21, 1972.

[22] I. Witten, A. Moffat, and T. Bell. *Managing Gigabytes. Compressing and Indexing Documents and Images*. Morgan Kaufmann Publishers, 1999.

[23] S. K. M. Wong, W. Ziarko, V. V. Raghavan, and P. C. N. Wong. On modeling of information retrieval concepts in vector spaces. In *The ACM Transactions on Databases Systems*, volume 12(2), pages 299–321, June 1987.

[24] S. K. M. Wong, W. Ziarko, and P. C. N. Wong. Generalized vector space model in information retrieval. In *The 8th ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 18–25, New York, USA, 1985.

[25] C. T. Yu and G. Salton. Precision weighting – an effective automatic indexing method. In *Journal of the ACM*, volume 23(1), pages 76–88, January 1976.

[26] M. J. Zaki. Generating non-redundant association rules. In *6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 34-43, Boston, MA, USA, August 2000.