An Information Retrieval Model Using the Fuzzy Proximity Degree of Term Occurences

Michel Beigbeder
mbeig@emse.fr

Annabelle Mercier
mercier@emse.fr

École Nationale Supérieure des Mines de Saint-Étienne
158, cours Fauriel
F 42023 SAINT ETIENNE CEDEX 2

ABSTRACT

Based on the idea that the closer the query terms in a document are, the more relevant this document is, we propose a mathematical model of information retrieval based on a fuzzy proximity degree of term occurences. Our model is able to deal with Boolean queries, but contrary to the traditional extensions of the basic Boolean information retrieval model, it does not explicitly use a proximity operator. A single parameter allows to control the proximity degree required. With conjunctive queries, setting this parameter to low values requires a proximity at the phrase level and with high values, the required proximity can continuously be relaxed to the sentence or paragraph levels. We conducted some experiments and present the results.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*retrieval models*

General Terms

Theory, Experimentation

Keywords

Proximity, fuzzy set theory, fuzzy proximity

1. INTRODUCTION

Whereas the information retrieval domain as a computer science research field is as old as computers themselves, and that thousands of experiments were conducted, there is no agreement on a unique model of information retrieval. In every text book about information retrieval, a chapter is dedicated to the models and each of them proposes, at least in its table of content, a taxonomy of the models. While there is not a complete consensus on the classification of these models, the one proposed by Baeza-Yates and Ribeiro-Neto [1] is quite common. These authors consider three *classic models*, namely the *Boolean* model, the *vector* model and the *probabilistic* model. Each of these three models is refined in *i) Set Theoretic* models (*fuzzy* and *extended Boolean*) *ii) Algebraic* models (*Generalized vector*, *Latent Semantic Indexing*, and *Neural Networks*), *iii*) *Probabilistic* models (*Inference Networks* and *Belief Networks*).

In the basic Boolean model, for a given query, *binary* relevance values are assigned to the documents: according to the system, documents *are* or *are not* relevant to the query. One of the strength of this model is that the query model is quite powerful in its expressiveness.

On the other hand, the vector model is based on a simpler query model – which is simply a bag of words – and it takes into account the number of occurences of the query terms in the document to score them with a value in a continuous space, typically either $\mathbb{R}$ or a subset of $\mathbb{R}$. One advantage is the possibility to sort the documents according to their system relevance score values. The benefit is that the documents are presented to the user in decreasing confidence of relevance: much of the information retrieval research is based on this ranking, in particular the systems evaluation.

The ranking capacity of the vector model which is not available in the Boolean model has lead to the introduction of the *extended Boolean* models and the use of *fuzzy sets*. Both of them use some kind of term weighting within the documents, and apply some combining formulas to compute the documents scores, given the term weigths and the query tree. These term weights, also used by the vector model, are usually computed with some kind of $tf \cdot idf$ formulas. The idea behind the tf factor was formulated in the very begining of the information retrieval research by Luhn [9]:

> *"It is here proposed that the frequency of word occurrences in an article furnishes a useful measurement of word significance. It is further proposed that the relative position within a sentence of words having given values of significance furnishes a useful measurement for determining the significance of sentences. The significance factor of a sentence will therefore be based on a combination of these two measurements."*

Actually, only the first aspect – *frequency of word occurrences* – received a great attention and was derived in many variations, one of them being the great variety of flavours

SAC'05 March 13-17, 2005, Santa Fe, New Mexico, USA

of $tf \cdot idf$. In the next section, we will review the works that were dedicated to this second aspect – *relative position [...] of words* – either directly or indirectly. Then we will present mathematical models for the set theoretic models in Sect. 3. Finally, we will present our model which merges the previous ones in Sect. 4 and some experimental results in Sect. 5 before concluding in the last section.

2. PROXIMITY USAGE IN INFORMATION RETRIEVAL

Proximity operators were introduced in Boolean information retrieval with an operator (commonly called NEAR, ADJ-acent, or WINDOW) which is a kind of AND but with the constraint that the different terms are within a window of size n, where n is either a user session wide value, or a value specified in the query for each such operator. Salton et al. [12] present these operators as linked to commercial systems. There were different works on the implementation problems raised by this operator and its consequence on the index itself. Keen [8] presents a study on the performance obtained in terms of Recall-Precision by different softwares that use the NEAR operator.

Though this operator cleanly fits in the basic Boolean model, we do not know any work that attempted to address the important problem of ranking, i.e. its integration in some kind of extended Boolean model.

In the beginning of information retrieval, the tests were conducted on "small collections" where the documents were homogeneous in a given collection, in particular with respect to their length. Then different works have been conducted on *passage retrieval*. These works were motivated by the varying length of documents found in large databases. So this subject can be seen as a kind of proximity usage: what the user expects to see is not one very long document, but rather the most relevant passage(s) of this document. Croft [3] gives a survey of the approaches to passage retrieval. These works can be seen as traditional document retrieval where the documents are not those of the collection, but each document of the collection is first split in passages. The passage either are explicitly delimited in the documents with markers, or are deduced from syntaxic features, or are of fixed length (overlapping or not windows).

Another interpretation of passage retrieval is to try to select the top ranked passages, i.e. those that concentrate many occurences of most of the query terms. So passage retrieval can be interpreted as finding the spots in the collection where the density of the query terms is the highest.

There were some attempts to directly score documents with explicit proximity information. Keen [7] tried some practical ideas but without a mathematical framework. These ideas were implemented over a Boolean system with a proximity operator and this design limited the possibilities.

Independently, Clarke et al. [2] and Hawking et al. [5] conducted very similar experiments: they search the smallest spans in the document text that contain all of the keywords, a score is then assigned to each of these spans (the shorter the span, the higher its score), and finally the score of a document is computed as the sum of the scores of the selected spans that it contains. More recently Rasolofo et al. [11] tried to introduce a correcting term to the score computed with the Okapi method [6].

3. BOOLEAN AND EXTENDED BOOLEAN MODELS

Any information retrieval system uses document representations which are obtained through an indexation of the documents. Most often this indexation is carried out with the extraction of the words found in the documents. In the sequel, we will call T the set of terms appearing in the documents.

3.1 Document and Collection Models

In the most basic Boolean model, a document is a *set* of terms, which can be modelized either as $d \subset T$ or as $d \in \{0,1\}^T$. We will prefer this second form because it is easily extended to the one used in the extended Boolean models based on the fuzzy set theory [10], where: $d \in [0,1]^T$. With such a definition, a document d is a function $d : T \to [0,1]$ and $d(t)$ is the membership degree of the document d to the term t considered as a fuzzy set. In the vector model context, $d(t)$ is usually called the weight of the term t within the document d, and is usually written $w_{d,t}$. Note that in the vector model the range of the function d is likely to be $\mathbb{R}^+$ (and even $\mathbb{R}$), rather than the interval $[0,1]$.

The collection model is a set of documents, so C is a collection iff: $C \subset \{0,1\}^T$ for the basic Boolean model, or $C \subset [0,1]^T$ for any fuzzy model, or $C \subset \mathbb{R}^T$ for the other extended Boolean models and the vector space model.

Given these definitions and notations, we can consider the family of functions $(\mu_t)_{t \in T}$ defined by $\mu_t(d) = d(t)$. When C is a subset of $\{0,1\}^T$, μ_t is the characteristic function of the set of documents containing at least one occurence of the term t. When C is a subset of $[0,1]^T$, μ_t is the membership degree function of the *fuzzy* set of documents containing the term t.

3.2 Query Model

The Boolean query model is a tree where the leaves are (weighted or not) terms, and the internal nodes are either AND or OR operators (with possibly a numerical parameter). In the pure Boolean model, both the weights and the parameters are equal to 1, and thus are not explicitly represented. For instance the query $(A \text{ AND } B) \text{ OR } C$ is represented by the tree drawn on Fig. 1.

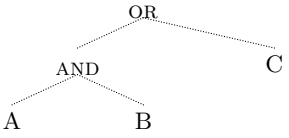

Figure 1: Query tree for $(A$ and $B)$ or C.

We will use the following notations. Q is the query set. An element of Q is either a *leaf node* or an *internal node*. A leaf node is an element $(t, w_{q,t}) \in T \times \mathbb{R}$. An internal node is a triplet $(\text{op}, (q_i)_i, p) \in \{\text{AND}, \text{OR}\} \times \mathcal{P}(Q) \times \mathbb{R}$, where $(q_i)_i$ is a finite subset of Q and p is a numerical parameter for this node. When the terms are not weighted, a leaf is just the term t, and when the operators are not parametrized we will notate them $(\text{op}, (q_i)_i)$.

3.3 Scoring Models

In the basic Boolean model the results are built by taking the union (for the OR operator) and the intersection (for the

AND operator) in the sets of documents containing the query terms. A mathematical formalization of this model consists in considering the matching function μ_q for the query $q \in Q$ which assigns a value in the set $\{0,1\}$ to each document of the collection C: $\mu_q : C \to \{0,1\}$. This function associates the value 1 to the documents that verify the property described by the query tree and 0 to the other documents. With this notation, the set of the documents relevant to the query q, according to the system, is $\mu_q^{-1}(1)$, where μ_q^{-1} denotes the inverse function of μ_q. It is possible to build the function μ_q using the family $(\mu_t)_{t\in T}$ previously introduced.

A query tree leaf with the term t is associated to the function μ_t. The set of documents matching an internal node $(\text{OR}, (q_i)_i)$ is the union of the sets of documents matching some of the $(q_i)_i$, which is expressed by

$$\mu_{(\text{OR},(q_i)_i)}{}^{-1}(1) = \bigcup_i {\mu_{q_i}}^{-1}(1).$$

This result can be obtained with

$$\mu_{(\text{OR},(q_i)_i)} = \max_i \mu_{q_i}.$$

Likewise, the AND operator is associated to the intersection and we write

$$\mu_{(\text{AND},(q_i)_i)} = \min_i \mu_{q_i}.$$

These formulas are recursively applied up to the root of the tree which represents the query q, and they define the function μ_q.

Here, the relevance criterion is binary, so the retrieved documents cannot be ranked. However, the previous formulas match the usual definition of union and intersection in the fuzzy set framework, so they can be used to define the fuzzy set q where μ_q is the membership function $C \to [0,1]$. Other possibilities were developped in this framework, one of them being: $\mu_{(\text{AND},(q_i)_i)} = \prod_i \mu_{q_i}$ and $\mu_{(\text{OR},(q_i)_i)} = 1 - \prod_i (1 - \mu_{q_i})$.

4. OUR MODEL

4.1 Document Model

In our model, we want to represent the documents with the positions of the term occurences. This is not new and it is implemented in any Boolean system which supports any kind of a NEAR operator. An intuitive view of this representation is that a document is a finite suite over $\mathbb{N}$ of length l of term occurences.

Formally, we extend this suite over $\mathbb{Z}$. We introduce an element ϵ that does not belong to T, and we notate: $T^* = T \cup \{\epsilon\}$. We modelize a document d as a sequence of elements belonging to T^*, $d : \mathbb{Z} \to T^*$, with the following condition

$$(\exists\, l \in \mathbb{N})\ (d^{-1}(T) = [0, l-1]).$$

With this notation, for some term t, $d^{-1}(t)$ is the set of the positions in the document d where the term t appears. Moreover $d^{-1}(T)$ is the set of positions where actual terms appear.

A collection is a set of documents. Figure 2 shows an example of a collection of four documents (d_0 to d_3) where the positions valued to ϵ are not represented, and where only two different elements of T, A and B, are represented. In this example, we have for instance: $d_3(2) = A$, $d_3(3) = A$, and $d_3^{-1}(A) = \{2, 3\}$.

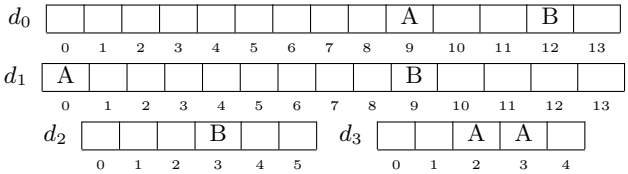

Figure 2: Example of a collection C, A and B are some elements of T.

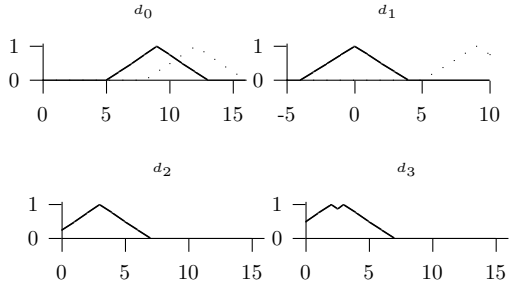

Figure 3: $(\mu_A^d)_{d\in C}$ plotted with plain lines and $(\mu_B^d)_{d\in C}$ plotted with dotted lines for the collection C of Fig. 2 with $k = 4$.

4.2 Local Term Proximity Model

Within the basic boolean model, it is easy to define a NEAR operator at the leaf level. For instance, the query $A \,\text{NEAR}\, 5\, B$ is evaluated to true if it exists one occurence of the term A and one occurence of the term B with less than five other words between them. This is a *binary* proximity value and we want to fuzzify this notion by taking into account the number of words between the two terms occurences.

In fact, we define a notion of proximity in the text which is not between two terms, but between a position in the text of the document and one term. Formally, we define $\mu_t^d : \mathbb{Z} \to [0,1]$ with

$$\mu_t^d(x) = \max_{i \in d^{-1}(t)} (\max(\frac{k - |x - i|}{k}, 0)),$$

where k is some integral parameter which controls to which extent one term occurence spreads its influence. In the following examples illustrated on figures, we will use $k = 4$. The function μ_t^d reaches its maximum (the value 1) where the term t appears and it decreases with a constant slope down to zero on each sides of this maximum. In other terms, this function has a triangular shape at each occurence of the term t. Fig. 3 shows $(\mu_A^d)_{d\in C}$ and $(\mu_B^d)_{d\in C}$ for the collection C shown in Fig. 2.

This function can be interpreted as the membership degree of any text position x in the document d to the fuzzy set (d, t).

4.3 Local Query Proximity Model

We now define a local proximity function between a text position and a query. Our query model is that of the boolean model. The functions μ_t^d defined in the previous section are associated to the leaves of the query tree. We define the local proximity at a given OR node with $\mu_{(\text{OR},(q_i)_i)} = \max_i \mu_{q_i}$, and likewise $\mu_{(\text{AND},(q_i)_i)} = \min_i \mu_{q_i}$. These are the same formulas

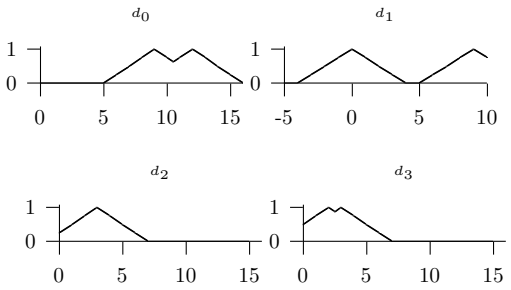

Figure 4: $(\mu^d_{A \text{ or } B})_{d \in C}$ for the collection C of Fig. 2.

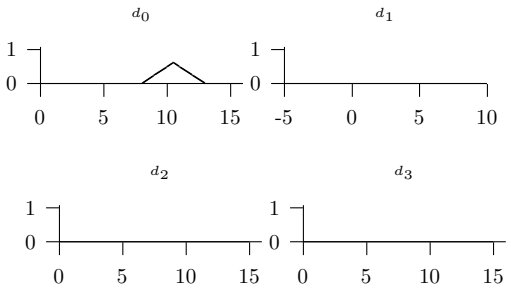

Figure 5: $(\mu^d_{A \text{ and } B})_{d \in C}$ for the collection C of Fig. 2.

as that of section 3.3 but here, given a document d, we are dealing with functions over $\mathbb{Z}$ rather than with numbers.

The recursive application of the previous formula up to the tree root q leads to a local proximity function between a document and the query q. This proximity means that the closer the terms requested by the AND operators are, the higher the value of the function is. Moreover this value is augmented by the closest of the terms requested by an OR operator. This function can be interpreted as the membership degree of any text position x in the document d to the fuzzy set (d, q).

Fig. 4 (resp. Fig. 5) plots $\mu^d_{A \text{ OR } B}$ (resp. $\mu^d_{A \text{ AND } B}$) for the documents of the collection C of Fig. 2. Note that the function $\mu^{d_1}_{A \text{ AND } B}$ is uniformly zero, though the document d_1 contains both the terms A and B because their occurences are too distant.

4.4 Global Document–Query Proximity Model

Within the information retrieval domain, one of the first similarity measure (i.e. an evaluation of the relevance between a document and the query) was the *coordination level*. This measure is used with bag of words queries. It operates by counting the number of occurences of the query terms. The idea, which was carried out further in the vector model, is to accumulate pieces of evidence for relevance. This measure, which we will notate c, also may be interpreted as the number of elements of the set of positions in the text where one of the query term appears

$$c(q, d) = |\cup_{t \in q} d^{-1}(t)|.$$

As the sets given in the union are mutually disjoint, we have

$$c(q, d) = \sum_{t \in q} |d^{-1}(t)|,$$

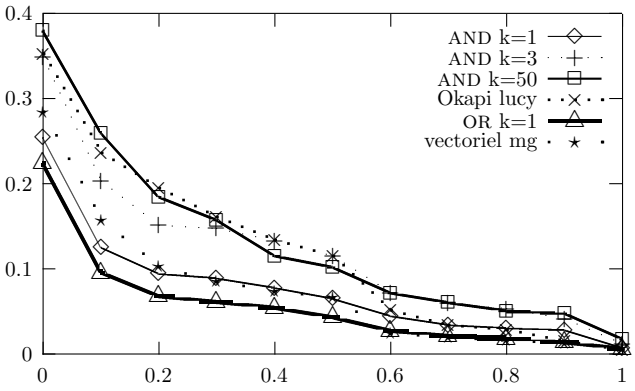

Figure 6: Precision-Recall curves for the vector model with mg, the Okapi method with lucy, our method with an and operator ($k = 3$ and $k = 50$) and our method with an or operator ($k = 1$). The latter case corresponds to the coordination level model.

or

$$c(q, d) = \sum_{x \in \mathbb{Z}} \nu^d_q(x),$$

with $\nu^d_q(x) = 1$ iff $d(x) \in q$ and $\nu^d_q(x) = 0$ otherwise. Please note that here the query is a *set* of terms. Computing the relevance score with the coordination level method is not possible with boolean queries.

In our model, the relevance score of a document d to the query q is computed with

$$s(q, d) = \sum_{x \in \mathbb{Z}} \mu^d_q(x).$$

According to the fuzzy set model, this is the fuzzy number of elements of the set of positions in the document d. Here, $x \mapsto \mu^d_q(x)$ represents the local proximity to the request for the document d while $x \mapsto \nu^d_q(x)$ represents a binary proximity to a (non fuzzy) set of terms.

The relevance score is a positive real number and the documents can be ranked according to their scores.

5. EXPERIMENTS

We have implemented our model by modifying the MG[1] package. To compute our scoring based on the term positions we need to keep them, which led us to build an additional inverted file.

We have used the TREC WT10g[2] collection with the topics 451–500[3] and the corresponding relevance judgements used in TREC9. We have compared the vector model implemented in MG and the Okapi model implemented in LUCY[4] with different variations of our model.

We had to build queries from the topics found in the TREC topics file. Our model can support full boolean queries but the vector and Okapi models cannot. So we have chosen to only build flat queries for these first experiments in order to

[1] http://www.cs.mu.oz.au/mg/

[2] http://es.csiro.au/TRECWeb/wt10g.html

[3] http://trec.nist.gov/data/topics_eng/topics.451-500.gz

[4] http://www.seg.rmit.edu.au/lucy/

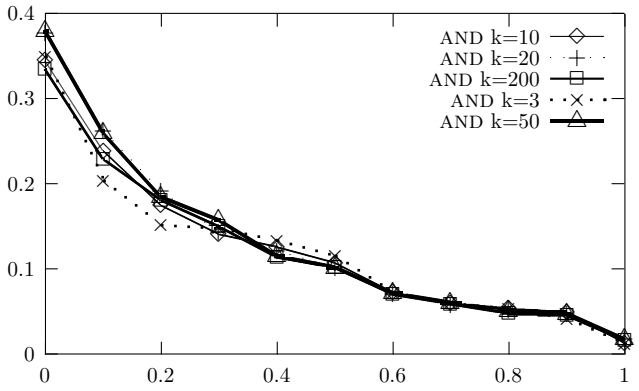

Figure 7: Precision-Recall curves for our method with an and operator and different values of k.

fairly compare our model with the vector and Okapi models. Given a topic, we only used the set of non stopwords found in the `title` field. For the vector model, the query simply is this set of words. In our model, we have used

- an OR operator with this set of words and $k = 1$,
- an AND operator with this set of words and $k = 1$, $k = 3$, $k = 10$, $k = 20$, $k = 50$, $k = 200$.

The parameter k introduced in section 4.2 defines the extent of the influence of each term occurence. The different values tested were chosen to match phrase lengths ($k = 3$), sentence lengths ($k = 20$), and paragraph length ($k = 200$).

With $k = 1$, the influence extent is reduced to the actual position of the term occurence and so the proximity capacity of our model is not really used. With $k = 1$ and the OR operator, we obtain another implementation of the historic coordination level model. The plots in Fig. 6 show that this model obtained the worst results.

With the AND operator and $k = 1$ all the documents are scored to zero and they are ranked in the collection order which is not meaningfull. Though, this model is a little better than the previous one. This is because only the documents that contain all the keywords are ranked and this is quite favourable with large collections like WT10g. This model is worst than the vector model at low recall level ($R < 0.3$) and better at high recall level ($R > 0.5$).

On Fig. 7 we compare the performance of our model with an AND operator when the parameter k is varying. The best performance are obtained for $k = 20$ and $k = 50$. The two curves for these two values are only distinguishable at $R = 0.3$ The worst tested case is for $k = 3$. Augmenting the influence extent does not improve the performance as can be seen with $k = 200$.

When proximity is really used with the AND operator and $k \geq 3$ our model is better than the vector model at every recall levels. Our model compares favourably to the Okapi model both at low and high recall level.

6. CONCLUSION

We have presented and implemented a new information retrieval model which takes into account the position of the term occurences in the document to compute a relevance score. So, one of its major advantages is to integrate the Luhn's proximity idea which has not gained a lot of work.

Different variations can be introduced, most of them could be pushed by controlling the shape of the proximity function introduced in section 4.2. We only used a triangular shape with a maximum set to one, the only parameter whose value was varied in our experiments was the extent of the triangle base which controls the extent of the influence of each term occurence. Other variations could use different combining formulas for the AND and OR operators. In our tests, we only used those based on the basic min and max.

Though our query model is that of the basic Boolean model, we only have tested this model with flat queries because we wanted to have fair comparisons with the vector and the Okapi models which don't support Boolean queries. We think that better results could be obtained by a manual work on the queries as it has to be done for every Boolean based model. Even without this improvement, our model have been showed to obtain quite good results compared to two classic models of information retrieval.

Finally an advantage of our model is that it does not require any collection wide information such as the *idf* values used in the vector model. So it could easily be used if the documents are distributed over a network.

7. REFERENCES

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[2] C. L. A. Clarke, G. V. Cormack, and F. J. Burkowski. Shortest substring ranking (multitext experiments for trec-4). In Harman [4].

[3] W. B. Croft. *Advances in Information Retrieval*, chapter Combining Approaches to Information Retrieval. Kluwer Academic Publishers, 2000.

[4] D. K. Harman, editor. *The Fourth Text REtrieval Conference (TREC-4)*. Department of Commerce, National Institute of Standards and Technology, 1995.

[5] D. Hawking and P. Thistlewaite. Proximity operators - so near and yet so far. In Harman [4].

[6] S. Jones, S. Walker, M. Gatford, and T. Do. Peeling the onion: Okapi system architecture and software design issues. *Journal of documentation*, 53:58–68, 1997.

[7] E. M. Keen. The use of term position devices in ranked output experiments. *The Journal of Documentation*, 47(1):1–22, 1991.

[8] E. M. Keen. Some aspects of proximity searching in text retrieval systems. *Journal of Information Science*, 18:89–98, 1992.

[9] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2:159–168, 1958.

[10] S. Miyamoto. *Fuzzy Sets in information retrieval and cluster analysis*. Kluwer Academic Publishers, 1990.

[11] Y. Rasolofo and J. Savoy. Term proximity scoring for keyword-based retrieval systems. In *25th European Conference on IR Research, ECIR 2003*, number 2633 in LNCS, pages 207–218. Springer, 2003.

[12] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill International, 1983.