Integrating Boolean and Vector Models of Information Retrieval with Passage Retrieval

Michel Beigbeder – *École des Mines de Saint-Etienne*
158 cours Fauriel – F 42023 Saint-Étienne Cedex 2 – France

Abstract

In the context of information retrieval, we propose here to merge in a single mathematical framework: the Boolean model, the vector space model, and passage retrieval in a single mathematical framework based on signal theory. In this framework, we define the weight $w_{d,t}$ of the term t in the document d not as a number, but as a function.

1 Introduction

The *Boolean* model and the *vector* model are the most classic models of information retrieval (IR) [1]. Another approach to IR appeared with the growing heterogeneity of the collections of documents, particularly in the size of the documents: *Passage retrieval* is concerned with retrieving passages that concentrate many occurences of most of the terms. These different IR models don't rely on a single mathematical modelization. We present here a mathematical model that merge the Boolean model and its fuzzy extension with the vector model and is tied to the passage retrieval model ideas.

In the basic Boolean model, given a 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 then 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 IR 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 both the *extended Boolean* models and the use of *fuzzy sets*. Both of them use some kind of term weights within the documents, and applies some combining formulas to compute the document score given the term weigths and the query tree. These term weights, also used by the vector model, usually are computed with some kind of *tf* · *idf* formulas. The *tf* factor (*Term Frequency*), proposed by Luhn [6], received great attention in the IR field. Another idea proposed by Luhn was that *"relative position [. . .] of words"* should be taken into account. *Proximity* of query term occurences is a way to consider their relative position.

Proximity Usage in Information Retrieval Proximity operators were introduced in Boolean information retrieval with an operator (commonly called NEAR, or 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 integral value, or an integral value specified in the query for each such operator [8]. Keen [5] studied 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 about any work that attempted to address the important problem of ranking, i.e. its integration in some kind of extended Boolean model.

Passage retrieval can be seen as a kind of proximity usage: what the user expects to see is not one too long document, but rather the most relevant passage(s) of this document. These works can be seen as traditional document retrieval where the documents are not those of the collection, but each document of the collection first is splitted in passages. The passages are either explicitly delimited in the documents with markers, or they are deduced from syntaxic features, or they 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 concentrates many occurences of most of the query terms.

There were some attempts to directly score documents with explicit proximity information. Keen [4] 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. [3] conducted very similar experiments: they search the smallest spans in the document text that contains all 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 one document is the sum of the scores of the selected spans that it contains. More recently Rasolofo et al. [7] tried to modify the score computed with the Okapi method with a term taking into account every intervals containing any couple of terms.

2 Boolean, Extended Boolean, and Vector Models

In the sequel, we will call T the set of terms appearing in the documents. In ordinary language, it is the *vocabulary* used by the collections of documents to be processed, it also could be called the *dictionnary*.

Document Model. In the most basic Boolean model, a document is a *set* of terms, which can be modelized as $d \in \{0,1\}^T$. It is easily extended to the one used in the models based on the fuzzy set theory where $d \in [0,1]^T$, or to that of the extended Boolean models where $d \in \mathbb{R}^T$. With such a definition, d is a function

$d : T \rightarrow [0,1]$ (or $d : T \rightarrow \mathbb{R}$), and $d(t)$ is the weight of the term t within the document d, more usually written $w_{d,t}$. A collection is a set of documents.

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 an optional numerical parameter). In the pure Boolean model, both the weights and the parameters are equal to 1, and are not explicitly represented.

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.

Scoring Model. The scoring function has to compute $s(q, d)$. The definition of this function is recursive, just like the query model. It is easily defined on the leaves of the query tree: $s(q, d) = w_{q,t} \cdot w_{d,t}$ if $q = (t, w_{q,t})$; and it is recursively defined for the nodes with formulas that combine the scores of each node's sons. The simplest formulas are $s((\text{OR}, (q_i)_i), d) = max_i s(q_i, d)$ and $s((\text{AND}, (q_i)_i), d) = min_i s(q_i, d)$. The key point in these formulas is that if more and more son nodes (either term leaves or sub-query nodes) are added to a given OR operator node, the score of this node cannot be lower. Reciprocally, if more and more son nodes are added to a given AND operator node, the score of this node cannot be higher.

Vector Model. In this model documents and queries are both some functions $T \rightarrow \mathbb{R}$, i.e. elements of $\mathbb{R}^T$. The scoring model is either an inner product: $s(q, d) = \sum_t q(t) \cdot d(t)$ or a cosinus: $s(q, d) = \frac{\sum_t q(t) \cdot d(t)}{\sqrt{\sum_t q(t)^2} \cdot \sqrt{\sum_t d(t)^2}}$. As stated before, the $d(t)$ and $q(t)$ are derived with some kind of $tf \cdot idf$ scheme. $d(t)$ and $q(t)$ are usually called $w_{d,t}$ and $w_{q,t}$ respectively.

3 Our Model

In our model, we want to represent the documents with the position of the term occurences. 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 terms belonging to T^*: $d : \mathbb{Z} \rightarrow T^*$ with the following condition: $(\exists\, l \in \mathbb{N})$ $(d^{-1}(T) = [0, l-1])$. An intuitive view of this definition is that a document is a finite suite over $\mathbb{N}$ of length l of term occurences, the suite is extended over $\mathbb{Z}$ with the value ϵ. Figure 1 shows an example of a collection of documents where the ϵ values are not represented.

We now define for some document d and some term t in the vocabulary T, the function d_t which has Dirac pulses where the term t occurs in the document d. The formal definition is: $d_t = \sum_{n \in d^{-1}(t)} \delta_n$ where $\delta_n : \mathbb{R} \rightarrow \{0, \infty\}$ is defined by $\delta_n(x) = \infty$ iff $x = n$ and $\delta_n(x) = 0$ otherwise, with $\int_{-\infty}^{+\infty} \delta_n(x)\, dx = 1$.

Given a family of *window* functions over T, $(g_t)_{t \in T}$ i.e. functions $g_t : \mathbb{R} \rightarrow \mathbb{R}$ with finite support. We consider the convolutions $d_t * g_t$. The functions g_t captures

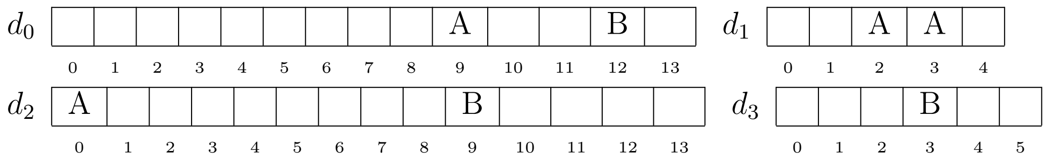

Figure 1: Example of a collection C, A and B are some elements of T.

the *influence* of each occurence of the term t within its neighbouring. Different window functions can be used: first, it is possible to choose different families of window functions (Hamming, Hanning, Gaussian, etc.), secondly, given a family, it is possible to choose some parameters of the function depending on the term t. Of course, it is possible to choose a single window function, g, and to use $g_t = g$ for each $t \in T$. Figure 2 shows the resulting $(d_t * g)_{d \in C}$ for the terms A and B where g has a triangular shape and a support of $[-4, +4]$.

Term Scoring Model. An interpretation of the sum in the vector scoring model is an accumulation of pieces of relevance evidences. Mathematical integration is able to capture such an idea by computing the surface below some curve. With the definitions we used, the more there are occurences of some term t in a document d, the more there are some peaks (or other shape, depending on the window function) in $d_t * g_t$. So, the mathematical integration of the function $d_t * g_t$ is able to capture the *tf* behaviour. We will call $s_v(d,t) = \int_{-\infty}^{+\infty} d_t * g_t(x)\,dx$.

Vector Like Scoring Model. If we use for g_t a rectangular function with support $[-\frac{1}{2}, +\frac{1}{2}]$ and with a height equal to the *idf* of the term t, we have

$$s_v(d,t) = \int_{-\infty}^{+\infty} d_t * g_t(x)\,dx = \mathrm{Card}(d^{-1}(t)) \cdot \mathit{idf}(t) = \mathit{tf}(d,t) \cdot \mathit{idf}(t).$$

Now, if we consider a query q as a *set* of terms $q \subset T$, under the assumption that each term of the query equally contributes to the relevance score of the document within a sum, we have $s_v(d,q) = \sum_{t \in q} s_v(d,t)$. More generally, if we consider a query q as a function $q : T \longrightarrow \mathbb{R}$, we define

$$s_v(d,q) = \sum_{t \in q} (q(t) \cdot s_v(d,t)), \qquad \text{so we have} \qquad s_v(d,q) = \sum_{t \in q} (q(t) \cdot \mathit{tf}(d,t) \cdot \mathit{idf}(t)).$$

The latter formula is similar to the traditionnal inner product of the vector space model. We can also interpret the previous equation as:

$$s_v(d,q) = \sum_{t \in q} (q(t) \cdot s_v(d,t)) = \sum_{t \in q} (q(t) \cdot \int_{-\infty}^{+\infty} d_t * g_t(x)\,dx) = \int_{-\infty}^{+\infty} \sum_{t \in q} (q(t) \cdot d_t * g_t(x))\,dx.$$

Here, the emphasis is on the integral. The interpretation is to integrate a function $x \mapsto w_{d,q}(x)$ over $\mathbb{R}$. This function represents the local relevance of the document d to the query q. In the vector space model, the function $w_{d,q}$ is $\sum_{t \in q} q(t) \cdot d_t * g_t$, and it appears as a linear combination of the $(d_t * g_t)_{t \in T}$.

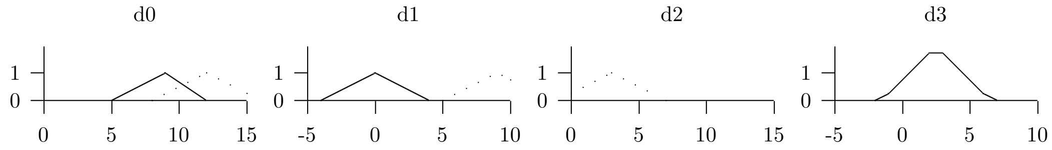

Figure 2: $(d_A * g)_{d \in C}$ plotted with plain lines and $(d_B * g)_{d \in C}$ plotted with dotted lines for the collection C of Fig. 1.

Merging the Boolean and the Vector Model. We want to merge the Boolean query model and the previous scoring model in the same framework. At each leaf $(t, w_{q,t})$ of the query tree, we associate the different function $d_t * g_t$. Note that g_t can depend on the value of $w_{q,t}$.

We have to deal with the internal nodes of the tree. If q is a node in the query tree which is either an OR or an AND operator, and the set of sons of the node q is $(q_i)_i$, we define $w_{d,(\mathrm{OR},(q_i)_i)} : \mathbb{R} \to \mathbb{R}$ with $w_{d,(\mathrm{OR},(q_i)_i)}(x) = max_i\, w_{d,q_i}(x)$, and, similarly $w_{d,(\mathrm{AND},(q_i)_i)}(x) = min_i\, w_{d,q_i}(x)$. Applying recursively these formulas up to the top-level node gives $w_{d,q} : \mathbb{R} \to \mathbb{R}$.

Finally, we define the relevance score value of the document d to the query q

$$s(d,q) = \int_{-\infty}^{+\infty} w_{d,q}(x)\, dx.$$

About the query $q = (\mathrm{OR}, \{A, B\})$, we can see on Fig. 2 that the relevance score value of d_0 is higher than that of d_1. With a vector model which does not take into account the position of the occurences of the terms, the two documents d_0 and d_1 would get the same score because they have the same number of occurences of A and B (and for some scoring, the same length too).

About the query $q = (\mathrm{AND}, \{A, B\})$, it is easy to see that if the two terms A and B were closer in a document d than in the document d_0 of the collection C, the resulting peak in the graph of $w_{d,q}$ would be larger and higher. So the relevance score value of the document d would be higher than that of document d_0. Moreover the most these two terms appear together and the closest their occurences, the higher is the score of the document. So with this mathematical framework, we have captured the two ideas evoqued by Luhn: *"frequency of word occurrences"* and *"relative position [...] of words"*.

Passage Retrieval Model. To evaluate the score of a passage with this model, the integration should be done on every passage of interest, this does not pose any problem other than the complexity directly linked to the number of passages to rank. Selecting a passage consists in defining two bounds x_0 and x_1, the relevance score value of the passage is defined as: $s(d[x_0, x_1], q) = \int_{x_0}^{x_1} w_{d,q}(x)\, dx$.

4 Conclusion

We have presented a mathematical framework that unifies the Boolean model and the vector model. This model can take into account a lot of variation of these two basic models: fuzzy sets, weighting schemes, extended Boolean models, etc. and they appear as particular cases of our model. Both in the conjonctive and disjonctive queries, the position of the occurences of the query terms are taken into account: the closer the occurences, the higher the score in the conjonctive case; the contrary in the disjonctive case.

The ideas are not far from the signal theory, as we introduce a *local relevance weight* with a function $w_{d,t}(x)$ where x is a position in the text. This function is a generalization of the traditionnal $w_{d,t}$ which is a single number. This could open a new field of information retrieval where other sound mathematical theories were applied with success: linear algebra, probability theory for instance. One of its major advantages is to integrate the Luhn's proximity idea which has not drawn a lot of work.

References

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

[2] Charles L. A. Clarke, Gordon V. Cormack, and Elizabeth A. Tudhope. Relevance ranking for one to three term queries. *Information Processing and Management*, 36:291–311, 2000.

[3] D. Hawking and P. Thistlewaite. Proximity operators - so near and yet so far. In D. K. Harman, editor, *TREC-4 proceedings.* NIST, 1995.

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

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

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

[7] Y. Rasolofo and J. Savoy. Term proximity scoring for keyword-based retrieval systems. In *ECIR 2003 proceedings*, number 2633 in LNCS, pages 207–218. Springer, 2003.

[8] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill International, 1983.