Fuzzy term proximity with boolean queries at 2006 TREC Terabyte task

Annabelle MERCIER and Michel BEIGBEDER
Ecole Nationale Superieure des Mines de Saint Etienne (ENSM-SE)
158 cours Fauriel 42023 Saint Etienne Cedex 2 FRANCE
mercier@i3s.unice.fr,mbeig@emse.fr

October 24, 2006

Abstract

We report here the results of fuzzy term proximity method applied to Terabyte Task. Fuzzy proxmity main feature is based on the idea that the closer the query terms are in a document, the more relevant this document is. With this principle, we have a high precision method so we complete by these obtained with ZETTAIR search engine default method (dirichlet). Our model is able to deal with Boolean queries, but contrary to the traditional extensions of the basic Boolean IR model, it does not explicitly use a proximity operator because it can not be generalized to nodes. The fuzzy term proximity is controlled with an influence function. Given a query term and a document, the influence function associates to each position in the text a value dependant of the distance of the nearest occurence of this query term. To model proximity, this function is decreasing with distance. Different forms of function can be used: triangular, gaussian etc. For practical reasons only functions with finite support were used. The support of the function is limited by a constant called k. The fuzzy term proximity functions are associated to every leaves of the query tree. Then fuzzy proximities are computed for every nodes with a post-order tree traversal. Given the fuzzy proximities of the sons of a node, its fuzzy proximity is computed, like in the fuzzy IR models, with a mimimum (resp. maximum) combination for conjunctives (resp. disjunctives) nodes. Finally, a fuzzy query proximity value is obtained for each position in this document at the root of the query tree. The score of this document is the integration of the function obtained at the tree root. For the experiments, we modify Lucy (version 0.5.2) to implement our matching function. Two query sets are used for our runs. One set is manually built with the title words (and sometimes some description words). Each of these words is OR'ed with its derivatives like plurals for instance. Then the OR nodes obtained are AND'ed at the tree root. An other automatic query sets is built with an AND of automatically extracted terms from the title field. These two query sets are submitted to our system with two values of k: 50 and 200. The two corresponding query sets with flat queries are also submitted to ZETTAIR search engine.

1 Introduction

In information retrieval domain, systems are founded on three basic ones models: The Boolean model, the vector model and the probabilistic model which were derived within many variations (extended Boolean models, models based on fuzzy sets theory, generalized vector space

model,...) [1]. Though they are all based on weak representations of documents: either sets of terms or bags of terms. In the first case, what the information retrieval system knows about a document is if it contains or not a given term. In the second case, the system knows the number of occurence – *term frequency*, tf – of a given term in each document. So whatever is the order of the terms in the documents, they share the same index representation if they use the same terms. The worthy of note exceptions are most of the Boolean model implementations which propose a NEAR operator [11]. This operator is a kind of AND but with the constraint that the different terms are within a window of size n, where n is an integral value. The set of retrieved documents can be restricted with this operator, for instance, it is possible to discriminate documents about "data structures" and those about "data about concrete structures". The result is an increase in precision of the system [5]. But the Boolean systems that implement a NEAR operator share the same limitation as any basic Boolean system : These systems are not able to rank the retrieved documents because with this model a document *is* or *is not* relevant to a query. In fact, different extensions were proposed to the basic Boolean systems to circumvent this limitation. These extensions represents the documents with some kind of term weights most of the time computed on a tf basis. Then they apply some combining formulas to compute the document score given the term weigths and the query tree. But these extensions are not compatible with the NEAR operator. So some works defined models that attempt to directly score the documents by taking into account the proximity of the query terms within them.

2 Other uses of proximity

Three methods were proposed to score the documents by taking into account some set of intervals containing the query terms. These methods differ in the set of intervalls that are selected in a first step, and then in the formulas used to compute a score for a given interval. The method of Clarke and al. [2] selects the shortest intervals that contains all the query terms (This constraint is relaxed if there are not enough retrieved documents), so the intervals can not be nested. In the methods of Hawking and al. [4], for each query term occurence, the shortest interval containing all the query terms is selected, thus the selected intervals can nest. Rasolofo and al. [8] chose to select intervals only containing *two* terms of the query, but with the additionnal constraint that the interval is shorter than five words. Monz [7] experiments in the domain of question-answering a method based on the minimal interval and a classical cosine measure to take account of term proximity. In an other approach, interval contribution stands for the tf in Okapi formula [10]. Moreover, the passage retrieval methods use indirectly the notion of proximity. In fact, in several methods, document ranking is doing by selecting documents which have passages with high density of query terms that-is-to-say documents where the query terms are closed [12, 3, 6]. The next section presents our method based on term proximity to score the documents.

3 Fuzzy proximity with boolean query based model

To address the problem of scoring the documents by taking into account the relative order of the words in the document, we have defined a new method based on a *fuzzy proximity* between each position in the document text and a query. First, given a document d and a term t, we define a term proximity function $w_{d,t}$. We can use different types of kernel

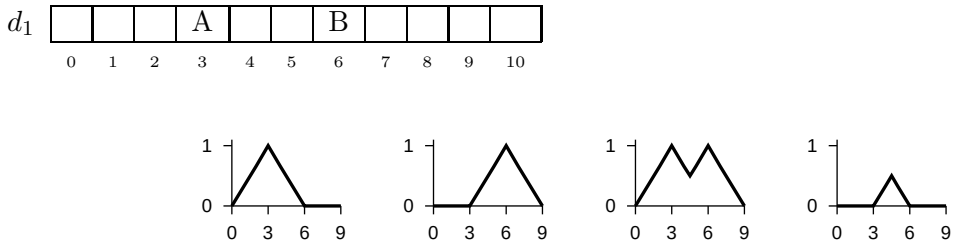

Figure 1: Document 1 – In order, we show w_A^{d1}, w_B^{d1}, $w_{A\,OR\,B}^{d1}$ and $w_{A\,AND\,B}^{d1}$.

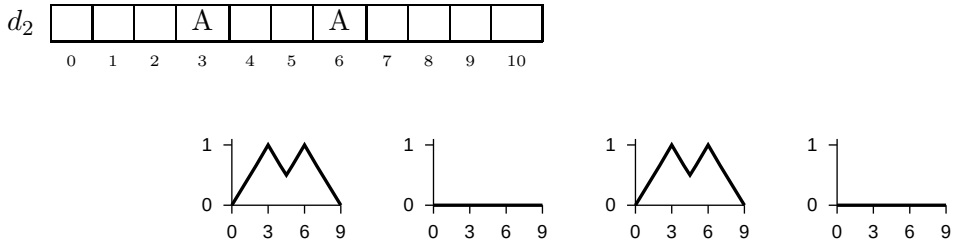

Figure 2: Document 2 – In order, we show w_A^{d2}, w_B^{d2}, $w_{A\,OR\,B}^{d2}$ and $w_{A\,AND\,B}^{d2}$.

(hamming, rectangular, gaussian) for the function but a triangular one is computed. A k constant controls the support of the function and this support represents the extent of each term occurence influence. This function reaches its maximum (value 1) at each occurence of the term t in the document d and linearly decreases on each side down to 0. So for each query term t, we determine the fuzzy proximity at each position of the document d retrieved. When the zone of influence of two terms occurrences overlaps in a document position x the value of the nearest term occurrence is taken so:

$$w_t^d(x) = \max_{i \in Occ(t,d)} f(x-i)$$

where $Occ(t,d)$ is the set of occurrence positions of term t in the document d and f the influence function kernel.

The figures 1 and 2 show the fuzzy proximity function w_A (resp. w_B) for the term A (resp. B) in the document d_0 and d_1.

The query model is that of the classical Boolean model: A tree with terms on the leaves an OR or AND operators on the internal nodes. Given a query q, the term proximity functions located on the query tree leaves are combined in the query tree with usual formulas pertaining to the fuzzy set theory. We compute here the fuzzy proximity of the query. So the fuzzy proximity is computed by :

$$w_{q\,\mathrm{OR}\,q'} = \max(w_q, w_{q'})$$

for a disjunctive node and by

$$w_{q\,\mathrm{AND}\,q'} = \min(w_q, w_{q'}).$$

for a conjunctive node.

So we obtain a function $w_{d,q}$ from the set of positions in the document text to the interval $[0, 1]$. The result of the integration of this function is used as the score of the document :

$$s(q,d) = \sum_{0}^{length(d)+\frac{k}{2}} w_q^d(x)\ \mathrm{d}x,$$

Finally, the computed score $s(q, d)$ depends on fuzzy proximity function and allows to rank document according to query term proximity.

4 Experiments at the Terabyte Track

We carried out experiments on the Terabyte Track TREC 2006 evaluation campaign [1]. In a first step, we use the retrieval tool LUCY that which is based on the Okapi BM-25 information retrieval model [9] to index this collection. This tool is adapted to our method because it keeps in the index the terms positions of the documents. We extend LUCY to compute similarity values according to our fuzzy proximity method. We can index all the terms of the collection with LUCY but the vocabulary map can not be loaded at run time this is the reason why we only index termes of topic file. In a second step, we use the default method of ZETTAIR search engines to obtain documents from a classical method in ordre to improve recall. By this way, we use our method to re-ranking documents, a final script build the run by copying first the fuzzy proximity run and then by adding new documents from dirichlet corresponding run.

For this track, we use the .gov2 collection, which is an image of a part of the Web. We remove all the specific text (HTML tags) and we only keep the text. Finally, we obtain about 40Go distibuted in 26 directories with 1000 compressed files. This treatment took about two weeks.

4.1 Building the queries

Each topic has three parts: `<title>`, `<desc>`, `<narr>`. We built two set of queries for our experiments. Queries are either manually or automatically built from the textual contents of the "title" tags. For topics 701-800, we build only automatics queries and for topics 801 to 850, we add manuel queries.

For automatic built queries, a query is made of terms from the "title" field where the stop words are removed.

For automatic runs, we use only conjonctive queries.

Manual built queries are made of terms from the "title" field and additionnaly terms from the "description" field. Moreover, we add the plurial form of the terms and the terms derivation to compensate the LUCY tool lack of stemming. We thus obtain queries that are conjunction of disjunctions of the different derivations of the terms. But, we restrict the terms to the indexed terms that is to say the words appearing in the topics file, we can not use semantics variation of words because queries were built after indexation time. On the other hand, the evaluation by the ZETTAIR tool uses flat queries that are of different derivations of the terms.

[1] http://trec.nist.gov

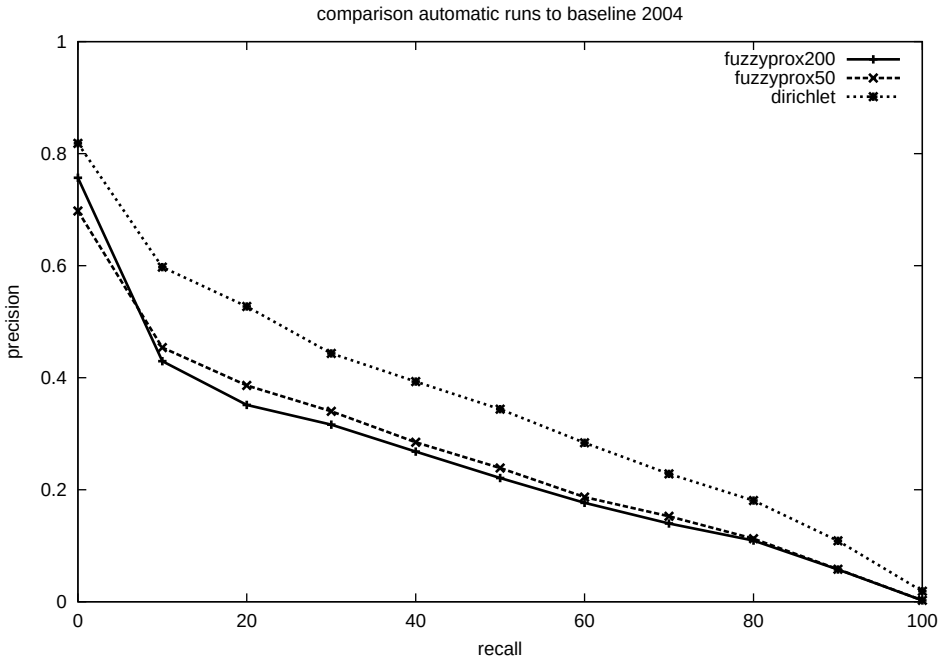

Figure 3: Terabyte 2004 automatic queries and the baseline with Zettair

4.2 Building the result lists

We compare the ZETTAIR default method and our fuzzy method with two different values of k (50, 200). If one of the proximity based method does not retrieve enough documents, then its results list is supplemented by the documents from the Okapi results list that have not yet been retrieved by proximity based methods; the maximum number of documents retrieved is $10,000$.

4.3 Differents runs

In the officials runs, the queries are constructed :

1. automatically with terms conjunction of title field and test with $k = 50$ (run AMRIMtp5006) and $k = 200$ (run AMRIMtp2006),

2. manually with terms of three fields and test with $k = 50$ (run AMRIMtpm5006).

For the runs ZettairTitlePorter and ZettairManPorter, the queries are flat (bag of terms) and these runs provide two baselines produced by using basic ZETTAIR search engine. We get a baseline ZETTAIR search engines for each type of queries :

1. ZettairTitlePorter for automatically built with title field, and,

2. ZettairManPorter for manually built.

The recall precision results are provided in the figure 3 and 4 for the automatic runs and in the figure 5 for manual runs.

We can note that our method gives better result with the largest value of k at the first level of recall but doesn't give better result than 'dirichlet'. We use ZETTAIR with Porter

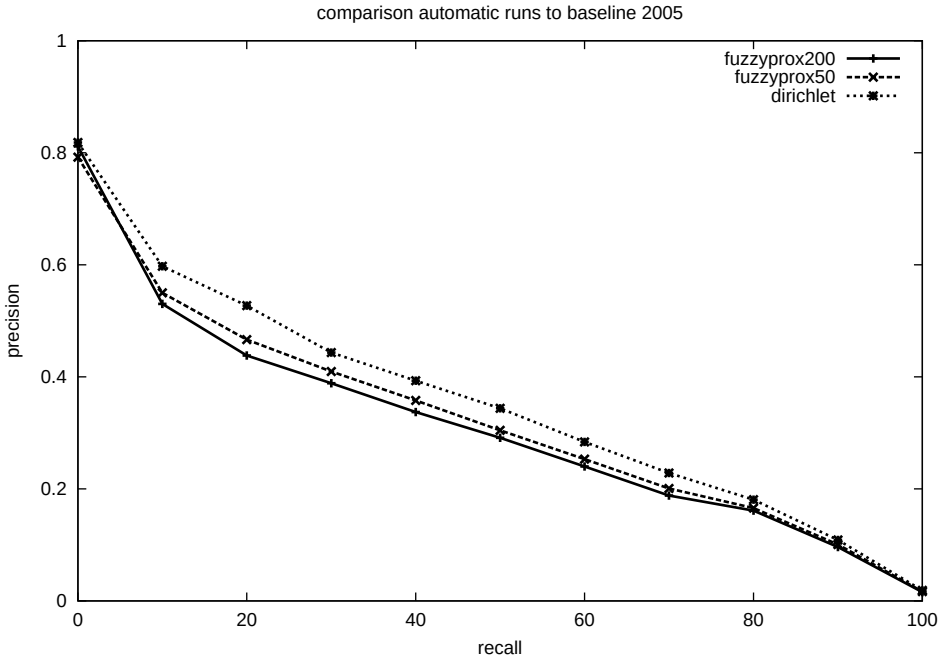

Figure 4: Terabyte 2004 automatic queries and the baseline with Zettair

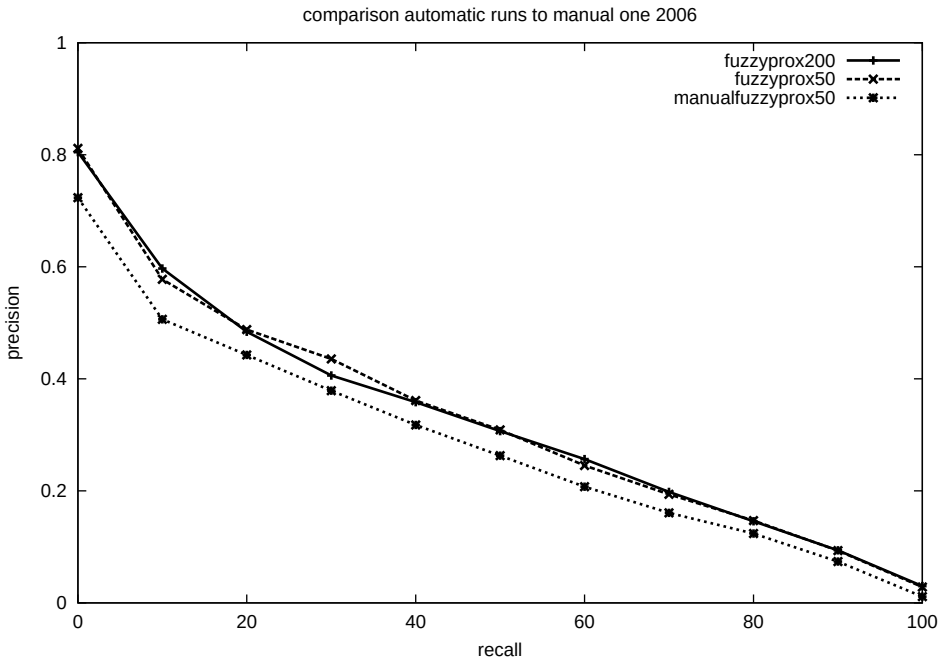

Figure 5: All runs submitted automatic and manual queries (Terabyte 2006)

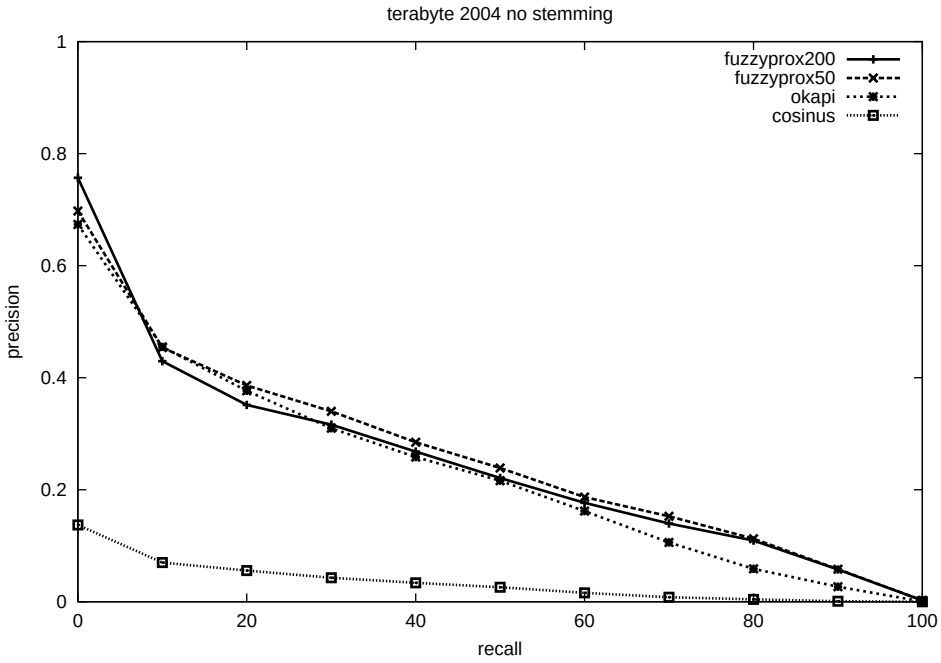

Figure 6: Comparison between Okapi and cosine method with no stemming and runs with automatic queries (Terabyte 2004)

stemming option because it minimize the size of the index but our fuzzy proximity method is implemented in LUCY search engine which haven't stemming treatment. The following results show a comparison between runs with no stemming built with ZETTAIR. As our method do not use stemming and the baseline from ZETTAIR use it, we show in figure 6 and 7 a comparison between our fuzzy proximity method and okapi (resp. cosine) without a stemming treatment on the collection. In this case, we note that our method is better or equal at the Okapi one.

5 Conclusion

We have presented and experimented our information retrieval model which takes into account the position of the term occurences in the document to compute a relevance score on the TREC 2006 Terabyte Track test collection. We notice that the higher the area of influence of term is the better the results are. After the comparison between the runs, we think that the results can be improved by using a stemming step before indexing but our pseudo-stemming by using boolean query can also improve the quality of the response. Consequently, we plan to help the user at the step of query built by propose new words for queries based on a thesaurus or a relevance feedback process.

References

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

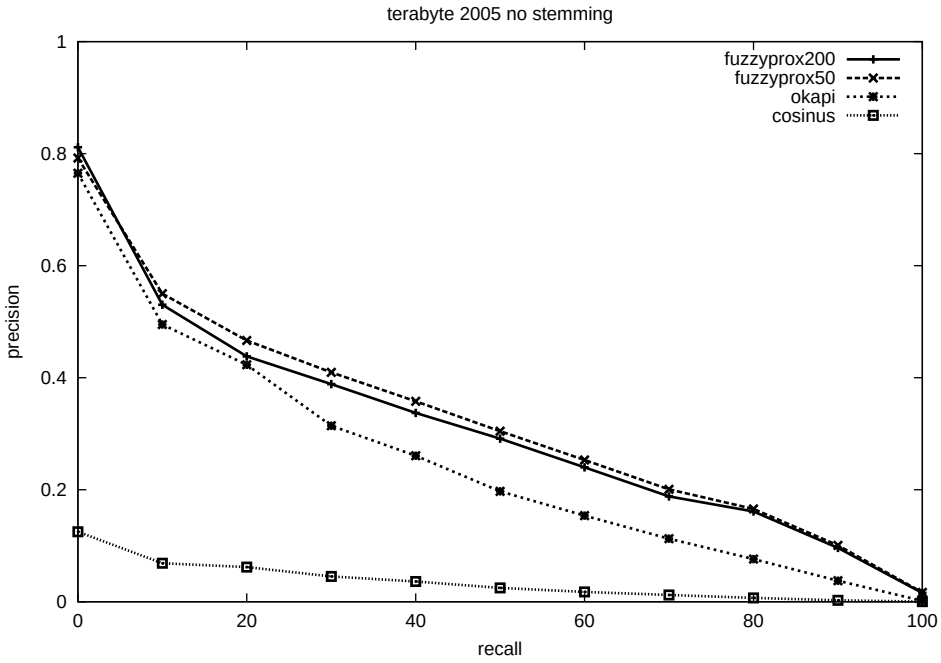

Figure 7: Comparison between Okapi and cosine method with no stemming and runs with automatic queries (Terabyte 2005)

[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(2):291–311, 2000.

[3] Owen de Kretser and Alistair Moffat. Effective document presentation with a locality-based similarity heuristic. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 113–120. ACM, 9 1999.

[4] D. Hawking and P. Thistlewaite. Proximity operators - so near and yet so far. In D. K. Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*, number 500-236. Department of Commerce, National Institute of Standards and Technology, 1995.

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

[6] Koichi Kise, Markus Junker, Andreas Dengel, and Keinosuke Matsumoto. Passage retrieval based on density distributions of terms and its applications to document retrieval and question answering. In Andreas Dengel, Markus Junker, and Anette Weisbecker, editors, *Reading and Learning: Adaptive Content Recognition*, volume 2956 of *Lecture Notes in Computer Science*, pages 306–327. Springer, 2004. No electronic version.

[7] Christof Monz. Minimal span weighting retrieval for question answering. In Rob Gaizauskas, Mark Greenwood, and Mark Hepple, editors, *SIGIR Workshop on Information Retrieval for Question Answering*, pages 23–30, 2004.

[8] Yves Rasolofo and Jacques Savoy. Term proximity scoring for keyword-based retrieval systems. In *25th European Conference on Information Retrieval Research*, number 2633 in LNCS, pages 207–218. Springer, 2003.

[9] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In D. K. Harman, editor, *Overview of the Third Text REtrieval Conference (TREC-3)*, number PB95-216883, pages 109–. Department of Commerce, National Institute of Standards and Technology, 1994.

[10] Wei-Ying Ma Ruihua Song, Ji-Rong Wen. Viewing term proximity from a different perspective. Technical report, 2005.

[11] Gerard Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company, 1983.

[12] Ross Wilkinson. Effective retrieval of structured documents. In *SIGIR 94 proceedings*, pages 311–317. Springer-Verlag New York, 1994.