

An Analysis of Learned Proximity Functions

Ronan Cummins
Dept. of Computing Science
University of Glasgow
Glasgow, Scotland
ronanc@dcs.gla.ac.uk

Colm O’Riordan
Dept. of Information
Technology
National University of Ireland
Galway, Ireland
colmor@it.nuigalway.ie

Mounia Lalmas
Dept. of Computing Science
University of Glasgow
Glasgow, Scotland
mounia@acm.org

ABSTRACT

A lot of recent work has shown that the proximity of terms can be exploited to improve the performance of information retrieval systems. We review a recent approach that uses an intuitive framework to incorporate proximity functions into vector based information retrieval systems.

More importantly, we present several proximity functions that were learned within this framework and show that they adhere to previously developed constraints regarding the shape of a good proximity function. Finally, we include results of all of the learned functions on unseen test data that shows the consistency of the learning approach used.

Categories and Subject Descriptors

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

General Terms

Experimentation, Performance

Keywords

Information Retrieval, Learning to Rank, Proximity

1. INTRODUCTION

Much recent work has shown that the proximity between terms is useful for improving retrieval quality in various information retrieval models [3, 5, 6]. Recent work has also used a machine learning approach to incorporate proximity functions into a general framework [1]. Within this framework a number of proximity functions have been learned using a symbolic machine learning approach (genetic programming).

Indeed, machine learning approaches applied to tasks in information retrieval have become more and more widespread. Support vector machines, neural networks, decision trees, gradient descent and genetic programming have all been applied to problems in information retrieval. While many of these artificial learning techniques show improved performance over traditional benchmarks for certain tasks, many lack a theoretical reason as to their comparative, if not superior, performance. To date, there has been a lack of analysis

regarding the output of many of these machine learning approaches. It is important that solutions from these many and varied machine learning approaches can be analysed so that theories underpinning the research in question can be further refined. The contributions of this paper are three-fold:

- In section 2 we review a general approach to incorporating proximity into any standard retrieval function and show that one of the main benchmark proximity approaches is a special case of this framework.
- In section 3 we analyse several proximity functions produced from a machine learning process and show that they adhere to theoretically sound properties regarding proximity.
- Finally, in section 4 we present results of these learned functions on unseen test data and show that all of the functions outperform a standard benchmark approach.

2. PROXIMITY RETRIEVAL MODEL

Vector based information retrieval systems are still some of the most common and efficient in use. Thus, a framework that intuitively incorporates proximity fully into these models is quite useful. In [1], a framework for incorporating information about the proximity between *all* query-terms into a retrieval model has been developed. This approach uses the following representation of a document D that matches a query $Q = \{t_1, t_2, t_3\}$:

$$D = \begin{pmatrix} w(t_1) & p(t_2, t_1) & p(t_3, t_1) \\ p(t_1, t_2) & w(t_2) & p(t_3, t_2) \\ p(t_1, t_3) & p(t_2, t_3) & w(t_3) \end{pmatrix}$$

where $w()$ is a traditional *tf · idf* type weighting and $p()$ is some proximity function. This *tf · idf* score can be derived from a vector space model (e.g. pivoted document length normalisation [2]), probabilistic model (e.g. *BM25* [2]) or even a language model (e.g. using Dirichlet Priors [2]), as all of these models can be realised using a vector based framework.

The score, $R(Q, D)$, of this document is then simply some aggregation of the values in the document matrix. The following formulation [1] is used:

$$R(Q, D) = \sum_{i \in Q \cap D} \sum_{j \in Q \cap D} \begin{cases} w(i) & \forall i = j \\ p(i, j) & \forall i \neq j \end{cases}$$

Thus, $R(Q, D)$ is the final score of a document in relation to a query Q . By ignoring the proximity function in this framework, that is, by setting $p(t_i, t_j) = 0 \forall i \neq j$, the standard ranking function is recovered. Using the sample document above and assuming a *BM25* weight for the initial terms, document D can be scored as follows:

$$R(Q, D) = BM25(Q, D) + 2 \cdot (p(t_1, t_2) + p(t_1, t_3) + p(t_2, t_3))$$

as $p(t_1, t_2) = p(t_2, t_1)$ in this framework. Thus, we can see that by simply extending any vector based model, the proximity of query-terms within a document can be incorporated.

However, there have been other approaches that incorporate proximity into retrieval functions. A common benchmark approach is that of Tao and Zhai [5]. Using document D as an example and assuming that t_1 and t_2 are the closest pair of query-terms in document D , Tao’s approach ($TR(Q, D)$) scores a document as follows:

$$\begin{aligned} TR(Q, D) &= BM25(Q, D) + tao(t_1, t_2) \\ &= BM25(Q, D) + 2 \cdot (tao(t_1, t_2)/2) \end{aligned}$$

where $tao(t_1, t_2) = \log(\alpha + \exp(\pi(t_1, t_2, D)))$. In this formulation, α is a tuning parameter and $\pi(t_1, t_2, D)$ is the distance between t_1 and t_2 in D .

By comparing the way in which $R(Q, D)$ and $TR(Q, D)$ both score the sample document D , we can see that they are only equivalent when there are two matching terms (i.e. t_1 and t_2). We can see that $R(Q, D)$ is a more complete view of proximity than that taken by a Tao’s approach. By viewing the scoring of the sample document D as an aggregation of all the entries in the matrix for D , we can see that $tao(t_1, t_2)$ can approximate *only* two of the larger valued non-diagonal scores in D (i.e. one pair of terms), and therefore it ignores the rest of the non-diagonal entries. This can be seen more easily in Figure 1. The darkest cells indicate the original weight of a term and are common to the score of a document in all approaches, i.e. non-proximity (*BM25*) and proximity ($R(Q, D)$ and $TR(Q, D)$) approaches. The lighter shaded cells are used in the proximity approaches identified here, i.e. $R(Q, D)$ and $TR(Q, D)$. Finally, the non-shaded cells are used in only one of the proximity approach outlined (i.e. $R(Q, D)$).

w₁₁	p₂₁	p₃₁	p₄₁
p₁₂	w₂₂	p₃₂	p₄₂
p₁₃	p₂₃	w₃₃	p₄₃
p₁₄	p₂₄	p₃₄	w₄₄

Figure 1: Graphical View of Proximity Framework

So, if t_1 and t_2 are the closest pair of terms in the document, the $tao(t_1, t_2)$ function can approximate $2 \times p(t_1, t_2)$, but will ignore $2 \times p(t_1, t_3)$ and $2 \times p(t_2, t_3)$ and the remaining non-shaded cells. For queries of length 2, the frameworks are the same, but for longer queries Tao’s approach will ignore the proximity between many more terms. Therefore, we

would expect the more general approach ($T(Q, D)$) to work well on longer queries. It is in this regard that the framework re-introduced here is a generalisation of that developed by Tao and Zhai [5] as it deals with queries of any length. Now that a complete framework has been outlined, the next step is to investigate the properties of a proximity function ($p(t_i, t_j)$) for all matching query-terms in a document.

3. PROXIMITY FUNCTION ANALYSIS

Now that we have outlined how a proximity function can be fully incorporated into any standard ranking function, we now wish to look at actual instantiations of these proximity functions. Developing proximity functions is not a trivial task [5] and much research into modelling proximity functions and proximity kernels [4, 3] has been conducted recently. In particular, previous work [5] has developed two constraints relating to proximity and it is argued that these constraints help model proximity correctly. These constraints state that (1) the score of a proximity function should decrease as the distance between the two terms increases and (2) that the score should drop sharply at first and become nearly constant as the distance between terms increases (i.e. a convex shape). Although these constraints have been outlined, much research has continued that aims to discover other useful functional shapes regarding proximity. A number of different proximity-based kernel functions continue to be explored by researchers [4, 3] (e.g. Gaussian, triangle, cosine, circle and discrete passage based kernels have all been adopted). Indeed, many of these kernel functions impose a different shape on a proximity function than those modelled by the constraints developed by Tao and Zhai.

A proximity function is typically comprised of one or more measures of the distances between a pair of terms in a document. An extensive list of proximity measures has previously been introduced [1]. Table 1 shows a number of proximity measures that can capture the distances between all occurrences of a pair of terms (t_i and t_j) in a document.

Previous work [1] has developed a number of proximity functions (i.e. $p(t_i, t_j)$) that can be used in the framework outlined in section 2) using a machine learning approach. In that work, a machine learning approach is used to combine multiple measures of proximity (i.e. all those from Table 1 and a number of other scaling factors) with a number of pre-defined functions to produce a number of proximity functions. These proximity functions were produced using a genetic programming approach that optimised the MAP (mean average precision) over a number of queries on some training data. Genetic programming is a biologically inspired survival of the fittest stochastic algorithm. Randomly created solutions undergo guided perturbations based on their level of fitness (i.e. performance on some training data). The interested reader is directed to [1].

The functions produced from that data-driven approach are outlined in Table 2 (labelled $p_1()$ to $p_6()$). We present all of the learned proximity functions and also analyse all of the functions produced from that work. On inspection of all of the six functions, the only explicit proximity measures that appear in each function are min_dist_{ij} and avg_dist_{ij} . Therefore, we have re-written the functions letting $x = min_dist_{t_i t_j}$ and $y = avg_dist_{t_i t_j}$. Various different normalisation measures and scaling measures also appear in these functions, but they do not affect the explicit distances that separate

Table 1: Proximity Measures between two terms (t_i and t_j) in a document D

$min_dist_{t_i,t_j}$	the minimum distance between any occurrence of t_i and t_j in D
$diff_avg_pos_{t_i,t_j}$	the difference in the average position of t_i and t_j in D
$avg_dist_{t_i,t_j}$	the average distance between all occurrences of t_i and t_j in D
$min_avg_dist_{t_i,t_j}$	the average of the minimum distance between an occurrences of t_i and t_j in D
$match_dist_{t_i,t_j}$	the average of the distances of uniquely paired occurrences of t_i and t_j in D
$max_dist_{t_i,t_j}$	the maximum distance between any occurrence of t_i and t_j in D

two pairs of terms. Indeed, both of these measures (x and y) appear in five of the six proximity functions produced.

In the learned functions (Table 2), c_1 and c_2 are constants for this analysis (they are actually the *sum* and *product* respectively of the term-frequencies of the pairs of terms) and can be considered scaling factors. The c_3 factor is the number of query-terms that match the document, while the c_4 factor is the portion of the document that contains all occurrences of all query-terms.

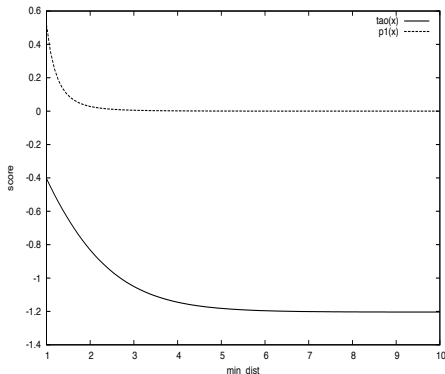


Figure 2: Curve for $tao()$ and $p_1()$

We now analyse these functions by varying x and y for typical values for both measures. In Figure 2, we plot the Tao function [5] ($tao()$) and $p_1()$ as they both contain only $min_dist_{t_i,t_j}$ (i.e. x) as an explicit proximity measure. We can see that the learned function $p_1()$ adheres to both proximity constraints as (1) as the distance measure increases the proximity function decreases and (2) the shape of this decrease is convex. Tao’s function was designed to adhere to the constraints, while the $p_1()$ was learned on training data. Furthermore, in Figures 3 and 4 we can see that all the learned proximity functions adhere to both proximity constraints for *both* measures of proximity contained within ($x = min_dist_{t_i,t_j}$ and $y = avg_dist_{t_i,t_j}$). This is quite interesting as it shows that a useful proximity function can often contain different measures of proximity, and furthermore, it shows the constraints for proximity apply to each of these different measures of proximity. These can be viewed as different dimensions of proximity being governed by the same constraints. Therefore, if a proximity function contains many different measures of proximity, each measure of proximity should also adhere to the constraints. Moreover, there may be many more than two proximity measures contained within any particular proximity function. This is an interesting finding as no other approaches to date have included multiple measures relating to proximity in their document scoring functions. Furthermore, very little or no analysis

of machine learning approaches to IR problems have been analysed to date.

As all these functions were produced from a data-driven approach, where the shape of the function is determined mainly by performance, the analysis of these learned functions reinforces the correctness of the constraints outlined by Tao and Zhai [5]. Furthermore, although much research has looked at different kernel functions for proximity, it would seem that only a few of these are suitable in a retrieval setting (i.e. those that adhere to the constraints). Indeed, some research has suggested that the Gaussian kernel [3] is one of the better performing proximity kernels. It is interesting that the Gaussian kernel has a similar shape to the learned functions presented here.

4. PERFORMANCE OF PROXIMITY FUNCTIONS

In this section, we include empirical results using all of the learned proximity functions to show that they perform well on a variety of different test collections. For the unseen test data, the FBIS (with topics 301-450), LATIMES (with topics 351-450), FR (with topics 251-450) collections from TREC disks 4 and 5 were used as test collections. The OHSUMED collection and its topics was also used. Stemming and stop-word removal was employed.

Table 3 shows the performance (MAP) on test data (\dagger shows statistical significance over the underlying function at the 0.05 level). The table shows the performance of all functions with respect to the original ranking function (labelled $S(Q, D)$). On average, all these learned functions outperform the underlying function on test data. This confirms the generality of the machine learning approach adopted. The worst performing learned proximity functions are also comparable to Tao’s baseline proximity approach (i.e. $S(Q, D) + tao()$). Furthermore, we can see that Tao’s approach performs poorer on some collections that use the longer (title and description) queries, which was mentioned in our discussions in section 2. Although many of the learned proximity functions are not statistical significant compared to the underlying function, the results do suggest a consistent trend among the learned functions.

5. CONCLUSION

We have shown that the framework outlined in [1] is a more general, yet related, framework to that shown in previous work [5]. More interestingly, we have shown that the constraints, previously developed, correctly model the best performing proximity functions produced from a learning approach. Finally, we include the results of all six proximity functions on test data and that show the consistency of the learning process adopted.

Table 2: Proximity Functions $p_1()$ to $p_6()$

$p_1() = (((((2 \cdot c_1 + (c_1/x)) + (c_2/((c_1 + c_2)/x))/x^2)/c_2)/x)/(c_1 \cdot x)$
$p_2() = \log(10/x) + 5 \cdot (c_2/y) + \sqrt{10/x}$
$p_3() = ((\sqrt{(100 \cdot (c_2/y))/x}/x^2) + (0.5/x^2))$
$p_4() = ((\exp(\sqrt{\sqrt{((c_2/y) \cdot (10/x)) + 1)/0.5}})/x^2)$
$p_5() = ((((((\frac{\log(c_4)}{x^2} + \frac{10}{c_1}) \cdot x) - 0.5)/(x^2)) + (((\log(0.5)) + (\frac{c_2}{y}))/0.5))/x) - 0.5$
$p_6() = ((3 \cdot \log(\frac{10}{x})) + \log(c_2 + \frac{10}{x})) + \frac{10}{x} + \frac{prod}{c_1 \cdot c_3}/c_3) + \frac{c_2}{y \cdot x}$

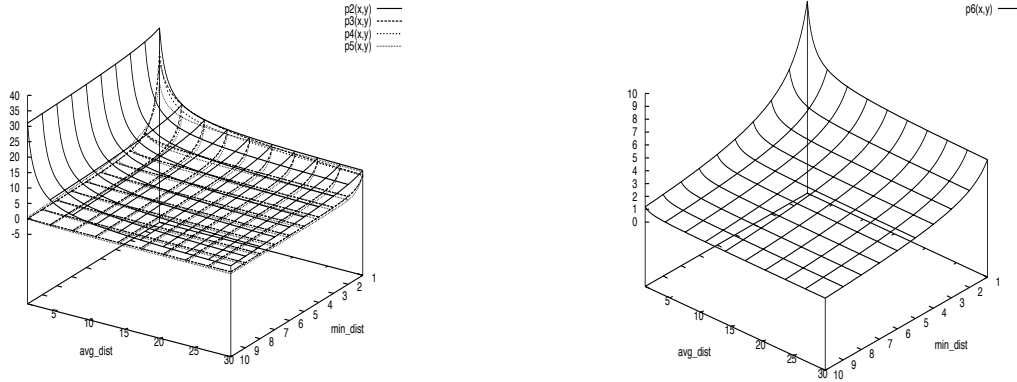


Figure 3: Surface for functions $p_2()$ to $p_5()$ (left) and function $p_6()$ (right)

Table 3: MAP on Test Data

	title only queries			title and description queries			
	LA	FBIS	FR	LA	FBIS	FR	OHSUMED
$S(Q, D)$	0.2099	0.2666	0.2772	0.2293	0.2920	0.2792	0.3314
$S(Q, D)$ with $tao()$	0.2169	0.2678 †	0.2615	0.2306 †	0.2686	0.2290	0.3327 †
$S(Q, D)$ with $p_1()$	0.2162	0.2781 †	0.2786	0.2367	0.2980	0.2842	0.3334
$S(Q, D)$ with $p_2()$	0.2152	0.2746	0.2820	0.2394 †	0.3001	0.2848	0.3367
$S(Q, D)$ with $p_3()$	0.2146	0.2740	0.2758	0.2425	0.2896	0.2738	0.3281
$S(Q, D)$ with $p_4()$	0.2135	0.2713	0.2813	0.2377	0.2958	0.2823	0.3282
$S(Q, D)$ with $p_5()$	0.2208	0.2797 †	0.2788	0.2363	0.3009 †	0.2826	0.3290
$S(Q, D)$ with $p_6()$	0.2233	0.2770 †	0.2834 †	0.2420 †	0.2979	0.2901 †	0.3371 †

6. ACKNOWLEDGMENTS

Ronan Cummins is funded by the Irish Research Council for Science, Engineering and Technology, co-funded by Marie Curie Actions under FP7. Mounia Lalmas is currently funded by Microsoft Research/Royal Academy of Engineering.

7. REFERENCES

- [1] Ronan Cummins and Colm O’Riordan. Learning in a pairwise term-term proximity framework for information retrieval. In *SIGIR 2009*, pages 251–258, Boston, MA, USA, 2009. ACM.
- [2] Hui Fang and ChengXiang Zhai. An exploration of axiomatic approaches to information retrieval. In *SIGIR ’05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 480–487. ACM Press, 2005.
- [3] Yuanhua Lv and ChengXiang Zhai. Positional language models for information retrieval. In *SIGIR 2009*, pages 299–306, Boston, MA, USA, 2009. ACM.
- [4] Desislava Petkova and W. Bruce Croft. Proximity-based document representation for named entity retrieval. In *CIKM ’07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 731–740, New York, NY, USA, 2007. ACM.
- [5] Tao Tao and ChengXiang Zhai. An exploration of proximity measures in information retrieval. In *SIGIR 2007*, pages 295–302, Amsterdam, The Netherlands, 2007. ACM.
- [6] Jinglei Zhao and Yeogirl Yun. A proximity language model for information retrieval. In *SIGIR ’09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 291–298, New York, NY, USA, 2009. ACM.