Proximity-Aware Scoring for XML Retrieval

Andreas Broschart
Max-Planck-Institut fuer Informatik
Saarbruecken, Germany
abrosch@mpi-inf.mpg.de

Ralf Schenkel
Max-Planck-Institut fuer Informatik
Saarbruecken, Germany
schenkel@mpi-inf.mpg.de

ABSTRACT

Proximity-aware scoring functions lead to significant effectiveness improvements for text retrieval. For XML IR, we can sometimes enhance the retrieval quality by exploiting knowledge about the document structure combined with established text IR methods. This paper introduces modified proximity scores that take the document structure into account and demonstrates the effect for the INEX benchmark.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Retrieval Models

General Terms: Algorithms, Experimentation

Keywords: proximity scoring, XML retrieval

1. INTRODUCTION

Term proximity has been a common means to improve effectiveness for text retrieval, passage retrieval, and question answering, and several proximity scoring functions have been developed in recent years (for example, [2, 3, 6, 7]). For XML retrieval, however, proximity scoring has not been similarly successful. To the best of our knowledge, there is only a single existing proposal for proximity-aware XML scoring [1] that computes, for each position in an element, a fuzzy score for the query, and then computes the overall score for the element by summing the scores of all positions and normalizing by the element's length.

We propose a proximity score for content-only queries on XML data that extends the existing proximity score by Büttcher et al. [2], taking into account the document structure when computing the distance of term occurrences.

2. PROXIMITY SCORING FOR XML

To compute a proximity score for an element e with respect to a query $q = \{t_1 \dots t_n\}$ with multiple terms, we first compute a linear representation of e's content that takes into account e's position in the document, and then apply a variant of the proximity score by Büttcher et al. [2] on that linearization.

Figure 1 shows an example for the linearization process. We start with the sequence of terms in the element's content.

SIGIR'08, July 20–24, 2008, Singapore.
ACM 978-1-60558-164-4/08/07.

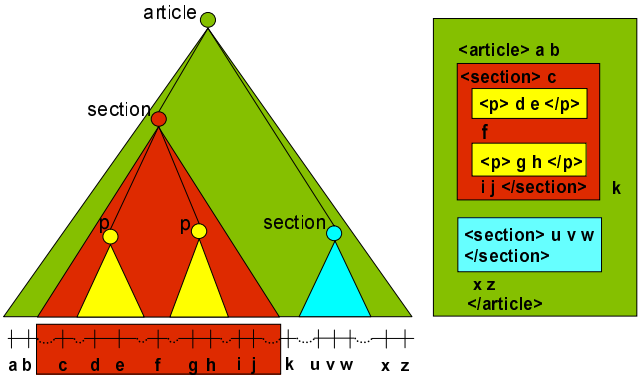

Figure 1: An XML document and its linearization

Now, as different elements often discuss different topics or different aspects of a topic, we aim at giving a higher weight to terms that occur together in the same element than to terms occurring close together, but in different elements. To reflect this in the linearization, we introduce virtual gaps at the borders of certain elements, whose sizes depend on the element's tag (or, more generally, on the tags of the path from the document's root to the element). In the example, gaps of `section` elements may be larger than those of `p` (paragraph) elements, because the content of two adjacent `p` elements within the same `section` element may be considered related, whereas the content of two adjacent `section` elements could be less related. Some elements (like those used purely for layout purposes such as `bold` or for navigational purposes such as `link`) may get a zero gap size. The best choice for gaps depends on the collection. Gap sizes are currently chosen manually; an automated selection of gap sizes is subject to future work.

Based on the linearization, we apply the proximity scoring model of Büttcher et al. [2] for each element in the collection to find the best matches for a query $q = \{t_1, \dots, t_n\}$ with multiple terms. This model linearly combines, for each query term, a BM25 content score and a BM25-style proximity score into a proximity-aware score. Note that unlike the original, we compute these scores for elements, not for documents, so the query-independent term weights in the formulas are inverse *element* frequencies $ief(t) = \frac{N - ef(t) + 0.5}{ef(t) + 1}$, where N is the number of elements in the collection and $ef(t)$ is the number of elements that contain the term t. Similarly, average and actual lengths are computed for elements.

To compute the proximity part of the score, Büttcher et al. first compute an accumulated interim score $acc(t_i)$ for each query term t_i that depends on the distance of this term's

occurrences in the element to other, adjacent query term occurrences. Formally, for each adjacent occurrence of a term t_j at distance d to an occurrence of t_i, $acc(t_i)$ grows by $ief(t_j)/d$. The proximity part of an element's score is then computed by plugging the *acc* values into a BM25-style scoring function:

$$score_{prox}(e,q) = \sum_{t \in q} min\{1, ief(t)\} \frac{acc(t) \cdot (k_1+1)}{acc(t)+K}$$

where, $K = k \cdot [(1-b) + b \cdot \frac{|e|}{avgel}]$ (analogously to the BM25 formula) and b, k_1, and k are configurable parameters that are set to $b = 0.5$ and $k = k_1 = 1.2$, respectively. The overall score is then the sum of the BM25 score and the proximity score:

$$score(e,q) = score_{\text{BM25}}(e,q) + score_{prox}(e,q)$$

3. EXPERIMENTAL EVALUATION

In order to evaluate our methods, we used the standard INEX benchmark, namely the INEX Wikipedia collection [4] with the 111 assessed content-only topics from the INEX AdHoc task 2006. Following the methodology of the INEX Focused Task, we computed, for each topic, a list of the 100 best non-overlapping elements with highest scores and evaluated them with the interpolated Precision metric used at INEX 2007 [5][1].

3.1 Results for Document-Level Retrieval

For our first experiment, we evaluated how good our proximity-aware scoring is at determining documents with relevant content. We limited the elements returned to `article` elements, corresponding to complete Wikipedia articles, and considered different gap sizes, where we report (1) gaps of size 0 for all elements, (2) gaps of size 5 for `section` and 3 for `p` elements, and (3) gaps of size 30 for `section` and `p` elements. Additionally, we report results without proximity (i.e., only the BM25 score) as baseline. Our implementation first computed the 100 best results for the BM25 baseline and then additionally computed the different proximity scores for these results, reranking the result list.

metric	baseline	(1)	(2)	(3)
iP[0.01]	0.6828	0.6814	0.6870	0.6871
iP[0.05]	0.5741	0.5783	0.5829	0.5849
iP[0.1]	0.5386	0.5438	0.5486	0.5486
MAiP	0.2595	0.2640	**0.2653**	**0.2655**

Table 1: Results for Document-Level Retrieval

Table 1 shows the results of these experiments. It is evident that proximity scoring in general can help to improve precision, and that introducing gaps gives an additional improvement. The MAiP values with introduced gaps turned out to be significantly better than the baseline for the Wilcoxon signed rank (WSR) test at a confidence level of at least 95% (printed in bold). However, our approaches could not demonstrate significant improvements as to the baseline for the iP values at the presented recall levels at a confidence level of at least 90% in WSR and paired t-tests.

3.2 Results for Element-Level Retrieval

We now evaluated the performance of proximity-aware scoring for element-level retrieval, where we limit the set of elements to be returned to those with `article`, `body`, `section`, `p`, `normallist`, and `item` tags for efficiency; initial experiments with all tags gave similar results. As we need to remove overlap, we first computed the best 200 results for the BM25 baseline, for which we then computed the proximity scores, resorted the list according to the new scores, and removed overlap (if two elements overlapped, we kept the element with highest score).

metric	baseline	(1)	(2)	(3)
iP[0.01]	0.6122	0.6081	0.6069	0.6075
iP[0.05]	0.4749	0.4859	0.4788	0.4779
iP[0.1]	0.3672	**0.3771**	*0.3712*	*0.3714*
MAiP	0.1281	**0.1338**	0.1315	*0.1315*

Table 2: Results for Element-Level Retrieval

Table 2 shows the results of these experiments. Again, compared to the baseline, proximity scoring in general helps to improve precision, but surprisingly, introducing gaps did not increase the positive effect, but actually reduced performance. WSR tests confirmed improvements of iP[0.1] and MAiP values for proximity scoring over the baseline at a confidence level of at least 95%, whereas for iP[0.1] gap-enhanced approaches could only succeed at a confidence level of at least 90% (printed in italics). Gap-enhanced approaches proved to deliver significantly better MAiP values than the baseline but were not as good as the mere proximity-based, gap-free approach (underlined stands for fail in WSR test and success in paired t-test at a confidence level of at least 90%).

4. CONCLUSIONS AND FUTURE WORK

This paper presented a structure-aware proximity score for XML retrieval that helps to improve the retrieval effectiveness of gap-free approaches for article-level retrieval, but does not show a similar effect for element retrieval. Our future work will further investigate this issue. Additionally, we will focus on automatic methods to determine good gap sizes for elements, determining characteristics for queries where proximity boosts performance, and extending proximity scoring to queries with structural constraints.

5. REFERENCES

[1] M. Beigbeder. ENSM-SE at INEX 2007: Scoring with proximity. In *Preproceedings of the 6th INEX Workshop*, pages 53–55, 2007.

[2] S. Büttcher, C. L. A. Clarke, and B. Lushman. Term proximity scoring for ad-hoc retrieval on very large text collections. In *SIGIR*, pages 621–622, 2006.

[3] O. de Kretser and A. Moffat. Effective document presentation with a locality-based similarity heuristic. In *SIGIR*, pages 113–120, 1999.

[4] L. Denoyer and P. Gallinari. The wikipedia XML corpus. In N. Fuhr, M. Lalmas, and A. Trotman, editors, *INEX*, volume 4518 of *Lecture Notes in Computer Science*, pages 12–19. Springer, 2006.

[5] J. Kamps, J. Pehcevski, G. Kazai, M. Lalmas, and S. Robertson. INEX 2007 evaluation measures. In *Preproceedings of the 6th INEX Workshop*, pages 23–32, 2007.

[6] Y. Rasolofo and J. Savoy. Term proximity scoring for keyword-based retrieval systems. In *ECIR*, pages 207–218, 2003.

[7] R. Song et al. Viewing term proximity from a different perspective. Technical Report MSR-TR-2005-69, Microsoft Research Asia, May 2005.

[1]Due to a bug reported for the original INEX implementation, we used a Java-based reimplementation of the metric.