Regular Paper

DBSJ Journal
Vol. 14, Article No. 2 March 2016

Heading-Aware Proximity Measure and Its Application to Web Search

Tomohiro Manabe[♡]
Keishi Tajima[♢]

Proximity of query keyword occurrences is one important evidence which is useful for effective query-biased document scoring. If a query keyword occurs close to another in a document, it suggests high relevance of the document to the query. The simplest way to measure proximity between keyword occurrences is to use distance between them, i.e., difference of their positions. However, most web pages contain hierarchical structure composed of nested logical blocks with their headings, and it affects logical proximity. For example, if a keyword occurs in a block and another occurs in the heading of the block, we should not simply measure their proximity by their distance. This is because a heading describes the topic of the entire corresponding block, and term occurrences in a heading are strongly connected with any term occurrences in its associated block with less regard for the distance between them. Based on these observations, we developed a heading-aware proximity measure and applied it to three existing proximity-aware document scoring methods: MinDist, P6, and Span. We evaluated these existing methods and our modified methods on the data sets from TREC web tracks. The results indicate that our heading-aware proximity measure is better than the simple distance in all cases, and the method combining it with the Span method achieved the best performance.

1 Introduction

Proximity of query keywords is an important factor for effective query-biased document scoring. It has been widely studied in the context of general document search, and many proximity-aware scoring functions have been proposed [9, 19, 20]. Most of them use *distance*, i.e., the difference of their positions, to measure proximity between two keyword occurrences. Less distant occurrences of query keywords in a document suggest more relevance of the document to the query. The distance, and also existing distance-based proximity-aware scoring functions, treat a document as an array of term occurrences. However, most documents have logical structure, and it affects logical proximity between term occurrences. Therefore, we should not measure logical proximity in structured documents by the simple distance.

Hierarchical heading structure is the most prevalent type of logical structure in documents [11]. It consists of nested blocks with their headings. A *heading* is a brief topic description of a logical segment of a document, and a *block* is such a segment associated with a heading. Figure 1 shows an example document with hierarchical heading structure. In this figure, each block (including the entire document) is enclosed by a rectangle and headings are emphasized in italic font.

Hierarchical heading structure strongly affects proximity. Suppose we have documents A, B, and C. In all of them, a query keyword X occurs n-term-occurrences-distant from an occurrence of another query keyword Y. In A, X occurs in a block and Y occurs in the heading of the block. In B, X and Y occur in the non-heading part of one block. In C, X and Y occur in two separate blocks that have no ancestor-descendant relationship. Among these documents, document A must be the most relevant to the query. This is because a heading describes the topic of the entire corresponding block, and term occurrences in a heading are strongly connected with any term occurrences in its associated block with less regard for the distance between them. Document C must be the least relevant to the query because the keywords occur in two separate blocks that may refer to two different topics.

In this paper, we propose a heading-aware proximity measure for scoring documents with heading structure. We measure proximity by using linear functions of distance that have different coefficients for each structural relationship between term occurrences. Our measure can be combined with various existing proximity-aware document scoring functions.

In this paper, we combine our measure with three existing proximity-aware scoring functions, and evaluated these combined methods on the data sets from the Text Retrieval Conference (TREC) web tracks. We compared the official baseline rankings, rankings generated by the existing proximity-aware scoring functions that use the simple distance, and rankings generated by the proximity-aware scoring functions that use our heading-aware proximity measure. The results indicate that our heading-aware proximity measure is better than the simple distance in all cases, and the method combining it with the Span method achieved the best performance.

2 Related Work

One way to score documents taking account of proximity between query keyword occurrences is to count frequency of n-grams ($1 < n$) that contain multiple query keywords as well as frequency of single query keyword [1, 15]. However, this method cannot deal with proximity between distant occurrences because it requires huge index size for large n.

Tao and Zhai [20] proposed five proximity-aware document scoring functions. They concluded that the *MinDist* function, which takes into account only the closest occurrence pair of two keywords, is the most effective for document ranking. Cummins and O'Riordan [9] proposed 10 basic proximity-based functions and produced three new functions by combining these 10 functions and two more basic functions by using genetic programming. Song et al. [19] proposed a method of segmenting a document into spans, and a proximity-aware document scoring function based on numbers of keyword occurrences and all term occurrences in the spans.

To the best of our knowledge, however, no study has proposed proximity measures or scoring functions that consider the properties of heading structure explained in Section 1.

Passage retrieval is another approach to improve docu-

[♡] Student Member Graduate School of Informatics, Kyoto University Research Fellow of Japan Society for the Promotion of Science manabe@dl.kuis.kyoto-u.ac.jp

[♢] Member Graduate School of Informatics, Kyoto University tajima@i.kyoto-u.ac.jp

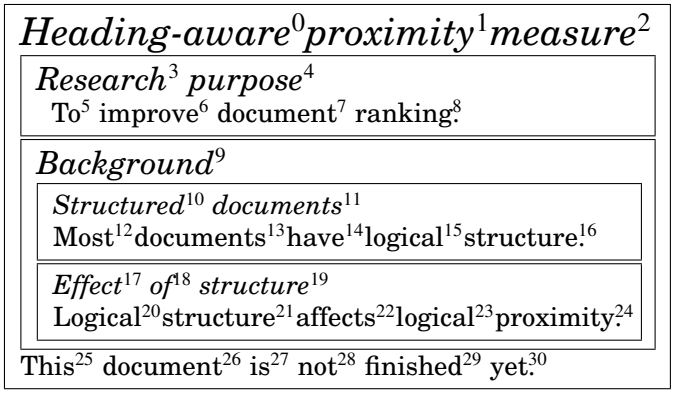

Figure 1: Example document with heading structure. Each rectangle encloses block, each text in italic font is heading, and each superscript number is position of term occurrence.

ment retrieval and is similar to, but not exactly the same as, proximity search [13]. Passage retrieval systems first segment documents into multiple regions, score the regions, then score the documents based on the region-based scores. The first problem in effective passage retrieval is how to extract regional structure of documents reflecting their topic structure [3]. There are many region extraction methods and some have already been applied to passage retrieval [18].

The Vision-based Page Segmentation method (VIPS) [2] is one of the most well-known methods of extracting regional structure in web pages. It detects margins in a page then segments the page into regions split by the margins. To extract the hierarchy, VIPS measures the weights of the margins mainly based on their width then recursively merges the regions split by the lightest margin. The same authors [3] and De Moura et al. [10] proposed passage retrieval methods for web pages based on VIPS. However, their methods do not use headings. To the best of our knowledge, heading-aware passage retrieval has also not been studied sufficiently.

As explained above, heading-aware document retrieval has not been studied sufficiently. An exception is many methods that take into account headings of entire documents, i.e., document titles. For example, BM25F [16], which is a variant of BM25 [17], is a widely used document scoring function. The BM25F function was designed to score documents composed of multiple fields, such as document titles, anchor text of in-links, and body text. However, it does not consider proximity between fields, such as title and body text.

3 Proximity Measures

In this section, we explain the simplest proximity measure, i.e., distance, and propose our heading-aware proximity measure, which we call *heading-aware semi-distance*. Note that they measure proximity between two term occurrences, not two terms, while some proximity-based functions shown later measure proximity between two terms.

3.1 Distance

The simplest measure for proximity between two term occurrences is *distance* between them, that is,

$$dist(o_1, o_2) = |pos(o_1) - pos(o_2)|$$

where o_1 and o_2 are term occurrences in a document and $pos(o)$ denotes the position of o, i.e., a serial number assigned to all term occurrences in the document. For example, in the document shown in Figure 1, $dist$(Heading-aware^0, Structured^{10}) = 10 and $dist$(proximity^{24}, document^{26}) = 2.

3.2 Heading-Aware Semi-distance

As discussed in Section 1, distance cannot represent logical proximity between two term occurrences o_1 and o_2 in documents containing heading structure. To measure proximity between them, we define a new measure, *heading-aware semi-distance*, denoted by *hasd*, as follows:

$$hasd(o_1, o_2) = \begin{cases} dist(o_1, o_2) \cdot a_{hc} + b_{hc} & \text{if } hc(o_1, o_2) = \text{true ;} \\ dist(o_1, o_2) \cdot a_{db} + b_{db} & \text{if } db(o_1, o_2) = \text{true ;} \\ dist(o_1, o_2) & \text{otherwise.} \end{cases}$$

This measure has four parameters, a_{hc}, b_{hc}, a_{db}, and b_{db}. The parameters a_{hc} and a_{db} are weights of simple distance, and b_{hc} and b_{db} are penalties given regardless of the simple distance. Functions hc and db are predicates on structural relationship between o_1 and o_2, which are defined below.

Heading-Content relationship: The predicate $hc(o_1, o_2)$ is true iff $o_1 \neq o_2$ and either o_1 or o_2 occurs in the hierarchical headings of a block in which the other occurs. In other words, $hc(o_1, o_2)$ is true iff the two occurrences are in relationship of a heading word and its corresponding content word. Note that it includes cases where both occurrences are in a heading and also cases where one occurrence is in the heading of a block and the other is in (either the heading or non-heading part of) its descendant block. For example, in Figure 1, hc(Heading-aware^0, measure^2) = true, hc(Heading-aware^0, Structured^{10}) = true, and hc(measure^2, Most^{12}) = true. Also note that $hc(o_1, o_2)$ is false when either o_1 or o_2 occurs in the heading of a block and the other is in the non-heading component of its ancestor block. For example, in Figure 1, hc(structure^{19}, document^{26}) = false. According to our assumption, $a_{hc} < 1$ and $b_{hc} = 0$ so that $dist(o_1, o_2) > hasd(o_1, o_2)$ when $hc(o_1, o_2)$ = true.

Occurrences in Different Blocks: On the other hand, $db(o_1, o_2)$ is true iff $o_1 \neq o_2$, $hc(o_1, o_2)$ is false, and o_1 and o_2 occur in two different blocks. Note that it includes cases where the two blocks have ancestor-descendant relationship. In Figure 1, db(Structured^{10}, Effect^{17}) = true, db(proximity^{24}, document^{26}) = true, db(Heading-aware^0, Structured^{10}) = false, and db(document^{26}, finished^{29}) = false. According to our assumption, $1 < a_{db}$ and $0 \leq b_{db}$, or $1 \leq a_{db}$ and $0 < b_{db}$, so that $dist(o_1, o_2) < hasd(o_1, o_2)$ when $db(o_1, o_2)$ = true.

In this paper, we assume $0 < a_{hc}$, $0 \leq b_{hc}$, $0 < a_{db}$, and $0 \leq b_{db}$ so that *hasd* is a symmetric positive-definite function (or a semimetric). Note that *hasd* does not necessarily satisfy the triangle inequality. For example, in Figure 1, $hasd$(Most^{12}, Structured^{10}) + $hasd$(Structured^{10}, structure^{16}) < $hasd$(Most^{12}, structure^{16}) when we set $a_{hc} < 0.5$ and $b_{hc} = 0$.

4 Proximity-Aware Scoring Methods

In this section, we explain three existing distance-based document scoring methods, which use simple distance. For each of them, we also propose a modified method that uses our heading-aware semi-distance instead of simple distance.

4.1 MinDist

The MinDist proximity-based function is proposed by Tao and Zhai [20]. They also proposed four other functions, but concluded in their paper that MinDist is their best performing function. In their paper, MinDist is defined for a document (or an array of term occurrences) D and a query (or a

set of keywords) Q. However, for conformity with the next scoring function P6, we re-define it for D and a pair of different query keywords, κ_1 and κ_2, without changing their scoring results. The function, denoted by *mindist*, is defined as below:

$$mindist(\kappa_1, \kappa_2, D) = \min_{o_1 \in O(\kappa_1, D), o_2 \in O(\kappa_2, D)} dist(o_1, o_2)$$

where $O(\kappa, D)$ is a set of occurrences of κ in D.

Their proximity-based score π of D for Q is defined by:

$$\pi(Q, D) = \log\left[\alpha + \exp\left\{-\min_{\kappa_1, \kappa_2 \in Q \cap D, \kappa_1 \neq \kappa_2} mindist(\kappa_1, \kappa_2, D)\right\}\right]$$

where α is a free parameter and $Q \cap D$ denotes a set of keywords in Q which also occurs in D.

Their final score of D for Q is the sum of the proximity-based score π and a non-proximity-based score. As the non-proximity-based score, we use a score given by Indri (explained later) with scaling. The final score is $s \cdot indri(Q, D) + \pi(Q, D)$ where s is a free parameter for scaling and $indri(Q, D)$ is the Indri score of D for Q. We call a document ranking method by this score the MinDist method.

4.2 P6

Cummins and O'Riordan developed three proximity functions by combining 12 basic proximity functions (including MinDist) by using genetic programming [9]. On most data sets in their evaluation, one of the functions, denoted by $p6$, achieved the best mean average precision (MAP) scores among them. The function $p6$ is defined as follows:

$$2 \cdot p6 = \left[\left\{3 \cdot \log\left(\frac{10}{mindist}\right) + \log\left(prod + \frac{10}{mindist}\right) + \frac{10}{mindist} + \frac{prod}{sum \cdot qt}\right\} / qt\right] + \frac{prod}{avgdist \cdot mindist}$$

where $qt = |Q \cap D|$ and $p6$, *mindist*, *prod*, *sum*, and *avgdist* are all functions of (κ_1, κ_2, D). The definition of *mindist* is the one shown before. The others are:

$$\begin{aligned} prod(\kappa_1, \kappa_2, D) &= |O(\kappa_1, D)| \cdot |O(\kappa_2, D)| \,, \\ sum(\kappa_1, \kappa_2, D) &= |O(\kappa_1, D)| + |O(\kappa_2, D)| \,, \text{ and} \\ avgdist(\kappa_1, \kappa_2, D) &= \frac{\sum_{o_1 \in O(\kappa_1, D), o_2 \in O(\kappa_2, D)} dist(o_1, o_2)}{prod(\kappa_1, \kappa_2, D)} \,. \end{aligned}$$

Their proximity-based score of D for Q is defined by:

$$S(Q, D) = \sum_{\kappa_1, \kappa_2 \in Q \cap D, \kappa_1 \neq \kappa_2} p6(\kappa_1, \kappa_2, D) \,.$$

Their final score of D for Q is the sum of the proximity-based score and a non-proximity-based score. We again use Indri score with scaling as the non-proximity-based score. The final score is $s \cdot indri(Q, D) + S(Q, D)$. We call a document ranking method by this score the P6 method.

4.3 Expanded Span

Song et al. proposed an efficient method to segment a document into *expanded spans* that never overlap, include as many unique query keywords as possible, and have minimal widths [19]. They also proposed a scoring function f of an expanded span for a query keyword, a scoring function rc of a document for a query keyword, and a BM25-like [17] scoring function of a document for a query.

In this paper, an expanded span is treated as a substring of D and denoted by $E = [o_i, ..., o_j]$. Note that o_i and o_j are occurrences of two different query keywords because of the width-minimality of expanded span. The width of E is:

$$width(E) = \begin{cases} dist(o_i, o_j) + 1 & \text{if } 1 < |E| \text{ and} \\ M & \text{otherwise} \end{cases}$$

where M is a parameter and $0 < M$. The score of a span E for a query keyword κ is:

$$f(\kappa, E) = \begin{cases} \left\{\frac{|Q \cap E|}{width(E)}\right\}^x \cdot |Q \cap E|^y & \text{if } 0 < |O(\kappa, E)| \text{ and} \\ 0 & \text{otherwise} \end{cases}$$

where $|Q \cap E|$ is number of query keywords (not limited to κ) that occur in E, and x and y are free parameters. The score of D for κ is $rc(\kappa, D) = \sum_{E \subset D} f(\kappa, E)$ and the score of D for Q is:

$$\sum_{\kappa \in Q} \frac{(k_1 + 1) \cdot rc(\kappa, D)}{k_1 \cdot \{(1 - b) + b \cdot |D| / avdl\} + rc(\kappa, D)} \log \frac{N - n(\kappa) + 0.5}{n(\kappa) + 0.5}$$

where k_1 and b are parameters, *avdl* is the arithmetic mean of $|D|$, N is the number of all documents, and $n(\kappa)$ is the number of documents that contain κ. We call a document ranking method by this score the Span method.

4.4 Heading-Aware Methods

We can easily combine our heading-aware semi-distance with these three methods MinDist, P6, and Span, by replacing *dist* in them by our *hasd*. We call the heading-aware versions of these three methods, the heading-aware MinDist method (or HA-MinDist), the heading-aware P6 method (or HA-P6), and the heading-aware Span method (or HA-Span).

In HA-Span, a proximity measure for two adjoining query keyword occurrences is also required for the segmentation of a document into expanded spans, and we used our *hasd* also for this segmentation. See their original paper for the details of this segmentation [19].

5 Application to Web Search

In this section, we explain some details of our implementation of the proximity-aware document scoring methods. Note that our heading-aware semi-distance is independent from the features of web pages and that it may be applicable to search for general documents containing heading structure, e.g., books and news articles.

5.1 Heading Structure Extraction

To apply our heading-aware variants of the existing proximity-aware scoring methods to web search, we need to extract hierarchical heading structure in web pages. It is not a trivial task and we used our previously proposed method [11]. Its input is an HTML web page with a DOM tree and the output is DOM node arrays that correspond to headings and blocks in the page. See the original paper for detailed design decisions and accuracy of the extraction method itself.

5.2 Text Content Extraction and Tokenization

To apply the scoring methods to a web page, we also need to extract D, i.e., an array of term occurrences in the page.

For non-heading-aware methods, we extract all DOM text nodes from each web page, join the contents of the nodes in document order into a string, then replace adjoining whitespace characters in the string by a single space. We ignored descendant text nodes of `STYLE` or `SCRIPT` HTML elements. When joining text contents, we insert a space between them.

Finally, we tokenize the string into D by the Treat implementation[1] of the Stanford Penn-Treebank style tokenizer.

For heading-aware methods, we split the string at the borders of headings and blocks. This is because headings and blocks are coherent information unit and a term occurrence must not partially overlap with headings or blocks. After that, we tokenize the strings. The other processes are same as those for non-heading-aware methods.

We score D for Q after tokenizing Q by the same implementation, removing 33 default stop words of Apache Lucene[2] from Q, and stemming all occurrences in D and Q by the Treat implementation of the Porter stemming algorithm [14].

6 Data Sets and Evaluation Measures

For optimization of the parameter values of the methods and also for evaluation of the methods, we used data sets prepared for TREC 2013–2014 web tracks [7, 8]. In this section, we explain the data sets and evaluation measures.

Queries and intents: The data sets have 50 keyword queries (topics) for each year[3]. There are one or more intents (subtopics) behind each query. We split the total 100 queries into training and test sets. Because the queries for each year are biased (for example, there are only six intents marked as navigational behind 2014 queries while 38 behind 2013) we split them based on their topic IDs, not years. We used the even-numbered 50 queries for training and the odd-numbered 50 for test.

Document collection: They have a huge web collection ClueWeb12, which is crawled by the Lemur Project[4] in 2012. Because of our resource limitation, we used its subset ClueWeb12B, which is also a TREC official collection. The collection contains about 52 million web pages.

Baseline scoring and ranking: The 2014 official baseline rankings were generated by default scoring of the Indri search engine (including query expansion based on pseudo-relevance feedback) and Waterloo spam filter[5]. Note that they also include rankings for the 2013 queries. We only used top-200 documents in the rankings because it requires much computation resource to score all the documents in the document collection or even the entire rankings and because we have to repeat it many times for parameter optimization.

Relevance judgment data: We used the TREC official graded relevance judgment data of the documents to the intents[3]. Note that it includes relevance judgment for entire ClueWeb12 and this fact affects the values of *ideal* DCG@20 explained below.

Evaluation measures: We used four TREC official and well-known evaluation measures, namely ERR-IA@20, α-nDCG@20, NRBP, and MAP-IA, for evaluating a document ranking for one or more intents behind a query. We used the TREC official code, `ndeval.c`, to calculate scores of these measures[5]. The definition of ERR@20 is:

$$\text{ERR@20} = \sum_{i=1}^{20} \frac{R(g_i)}{i} \prod_{l=1}^{i-1} \{1 - R(g_l)\} \text{, where}$$

$$R(g_i) = \frac{2^{g_i} - 1}{16}$$

where $g_i \in \{0, 1, 2, 3, 4\}$ is the graded relevance of the ith document in the ranking to an intent. The definition of P@i is the ratio of relevant documents in top-i documents in the ranking. The ith document is considered relevant iff $0 < g_i$. The definition of AP is the arithmetic mean of P@i where ith document is relevant. The *ERR-IA@20* and AP-IA measures are respectively arithmetic mean of ERR@20 and AP scores for all intents (with at least one relevant document) behind the query. The ERR-IA@20 measure is better at measuring effectiveness for navigational intents [4]. The definition of α-*nDCG@20* [5] is:

$$\alpha\text{-nDCG@20} = \frac{\text{DCG@20}}{\textit{ideal}\ \text{DCG@20}} \text{, where}$$

$$\text{DCG@20} = \sum_{i=1}^{20} \frac{G(i)}{\log_2(1+i)} \text{, and}$$

$$G(i) = \sum_{j=1}^{m} \frac{J(i,j)}{2^{C(i,j)}} .$$

The number of intents behind the query is denoted by m, $J(i, j)$ is 1 iff the ith document in the ranking is relevant to the jth intent behind the query, i.e., $0 < g_i$ about j, and is 0 otherwise, $C(i, j)$ is number of documents ranked higher than i and relevant to the jth intent. The definition of *NRBP* is:

$$\text{NRBP} = \frac{m}{4} \sum_{i} \frac{G(i)}{2^{i-1}} .$$

The NRBP measure is better at measuring effectiveness for ambiguous and underspecified queries [6].

To integrate the evaluation scores for multiple queries, we calculated arithmetic mean of the scores. Especially, we call arithmetic mean of AP-IA scores *MAP-IA*.

7 Fine Tuning with Training Data

In this section, we explain the optimization method we used for parameter values of the methods, explain optimized parameter values, and discuss meanings of the values.

7.1 Optimization Method

For the optimization, we used the 50 training queries and adopted the simple *Coordinate Ascent* algorithm [12], which optimize only one parameter at a time. This is because optimization method is not the main topic of this paper. To avoid local maximum, we check three smaller and three larger candidate values than the current value. If the best score among the six candidates is better than the current best score, we adopt the candidate as a new current value then check new six candidates around the value. If not, we optimize the next parameter. Next to the last parameter, we optimize the first parameter again. If we cannot further improve the score by changing any of parameters, we quit the optimization.

The target function is set to MAP-IA. This is because the other measures tend to (almost) ignore lower-ranked documents even though they are important in the middle of the greedy optimization process.

Median of initial values are: $a_{hc} = 0.30$, $b_{hc} = 0$, $a_{db} = 1.00$, $b_{db} = 15$, $\alpha = 1.00$, $s = 1.00$, $M = 45$, $x = 0.25$, $y = 0.30$, $k_1 = 0.40$, and $b = 0.30$. These values are taken from the original papers of the existing methods or reasonably selected by us. Initial

[1] https://github.com/louismullie/treat
[2] https://lucene.apache.org/
[3] http://trec.nist.gov/data/webmain.html
[4] http://www.lemurproject.org/
[5] https://github.com/trec-web/trec-web-2014/

Table 1: Optimized parameter values of our *hasd*.

Method	a_{hc}	b_{hc}	a_{db}	b_{db}
HA-MinDist	.450	0	1.50	3
HA-P6	.600	0	1.70	36
HA-Span	.800	3	.800	30

Table 2: Optimized values of other parameters.

Method	s	α	M	x	y	k_1	b
MinDist [20]	2.83	.420					
HA-MinDist	2.83	.297					
P6 [9]	256						
HA-P6	256						
Span [19]			54	.250	1.35	3.20	.250
HA-Span			27	.250	.800	.800	.350

values are randomly selected from these median values, five smaller values, and five larger values.

When we change α, s, or k_1 values in the optimization, we multiply it by $2^{-\frac{1}{4}}$ or $2^{\frac{1}{4}}$ because they are expected to be larger than 0 and their optimized values may be dozens of times higher than the initial value. When changing a_{hc}, a_{db}, x, y, or b values, we add -0.05 or 0.05 to the value. When changing b_{hc}, b_{db}, or M values, we add -3 or 3 to the value.

Considering the meanings of parameters, we defined the ranges of some parameters and did not check values outside them. Because *hasd* must be larger than 0, we let $0.05 < a_{hc}$, $0 \leq b_{hc}$, $0.05 < a_{db}$, and $0 \leq b_{db}$. We also let $0 \leq b \leq 1$.

The values of *avdl*, N, and $n(\kappa)$ in (HA-)Span were computed by using the ClueWeb12B and the older ClueWeb09B collections for robust estimation. The value of *avdl* was about 1,650 and the value of N was about 103 million.

7.2 Optimized Parameter Values

Table 1 lists the parameter values of our *hasd* measure optimized for each of our heading-aware method. Optimized values of the other parameters are listed in Table 2. They are derived from 64 initial settings for each method.

7.3 Discussion on Parameter Values

First we discuss the parameter values of *hasd* (see Table 1).

Heading-Content relationship: For both the HA-Mindist and HA-P6 methods, optimized values of a_{hc} are 0.6 or smaller and b_{hc} was 0. It means logical distance between a pair of term occurrences in a heading and in its associated block is evaluated to be smaller than the simple distance. These results coincide with our observation explained in Section 1. However, for the HA-Span method, optimized values of a_{hc} was 0.8 while b_{hc} was 3. When $hc(o_1, o_2)$ = true and $a_{hc} = 1$ and $b_{hc} = 0$, *hasd* is equal to *dist* and 0.8 is close to 1 and 3 is close to 0. Therefore, it means that there is no significant difference between *dist* and *hasd*.

Occurrences in Different Blocks: On the other hand, optimized values of a_{db} and b_{db} were inconsistent over three methods. For HA-MinDist, optimized a_{db} was 1.5 but b_{db} was 3. In this setting, term occurrences in separate blocks are logically more than 1.5 times distant than in the simple distance. For HA-P6, a_{db} was 1.7 and b_{db} was 36. In this setting, term occurrences in two separate blocks are logically far more than 1.7 times distant than in the simple distance. For HA-Span, a_{db} was 0.8 and b_{db} was 30. Because the value of M (maximum *hasd* between adjoining term occurrences in the same span), is set to 27, this setting in effect means that term occurrences in different blocks are logically distant enough so that there is no necessity of considering proximity between occurrences in different blocks. All these results coincide with our observation explained in Section 1.

Table 3: Comparison of average evaluation scores.

Method	ERR-IA@20	α-nDCG@20	NRBP	MAP-IA
Baseline	.310	.364	.285	.016
MinDist [20]	.298	.360	.270	.017
HA-MinDist	.335	.392	.304	.018
P6 [9]	.307	.367	.277	.017
HA-P6	.309	.368	.280	.017
Span [19]	.402$*$	.440	.383$*$	.020
HA-Span	**.436**$*$	**.470**$*$	**.418**$*$	**.021**$*$

$*$ statistically significantly different from baseline ($p < 0.05$)

As discussed above, most obtained parameter values of *hasd* follows our assumptions. This fact supports the validity of our idea and definition of *hasd* proximity measure.

Next we discuss the other parameter values (see Table 2). The values of s for P6 and HA-P6 methods were optimized to 256. This fact means the effect of the $p6$ function was quite limited and the resulting rankings were almost same as the baseline. This should be because the $p6$ function itself is already optimized by using genetic programming for a certain data set [9].

The original papers of the MinDist and Span methods show parameter values already optimized for certain data sets. The value of α in the *mindist* function is 0.3 [20] and the values of M, x, y, k_1, and b in the Span method are respectively 45, 0.25, 0.3, 0.4, and 0.3 [19]. However, according to our preliminary evaluation with test data, our optimized parameter values consistently achieved better scores than the parameter values shown in the original papers on all the evaluation measures. Therefore, hereafter in this paper, we use our optimized parameter values. This fact also supports usefulness and robustness of the optimization method we used.

8 Evaluation with Test Data

Last of all, we explain the evaluation result of web page rankings by the methods.

8.1 Evaluation Method

We used the 50 test queries for evaluation. As explained before, we only scored and ranked top-200 documents in the baseline rankings for the queries (total 10,000 tuples of a document, a query, and the baseline score of the document for the query). In other words, we evaluated the methods by re-ranking. All parameters of the methods are set to the values listed in Table 1 and 2.

8.2 Evaluation Result

Table 3 lists the evaluation scores of all pairs of a measure and a method. In this table, each asterisk means statistically significant difference from the baseline ($p < 0.05$) according to Student's paired t-test where each pair consists of the evaluation scores of two methods for a query. Hereafter in this paper, we discuss statistical significance based on the same test procedure.

8.3 Discussion on Evaluation Results

Comparison with baseline scores: First, we compare each score with the baseline score on the same evaluation measure. As shown in Table 3, for 10/12 pairs of one of our heading-aware methods and an evaluation measure, our

methods achieved the better scores than the baseline scores. For 7/12 pairs of an existing method and an evaluation measure, existing methods also improved the baseline scores. However, according to the t-test procedure, only the improvements by the existing Span and our heading-aware Span methods were statistically significant on some of the measures. Especially, our HA-Span method statistically significantly improved the baseline rankings on all the measures. Moreover, our HA-Span method also achieved the best scores among all the methods on all the measures. These facts strongly supports the validity of our assumptions about the effects of heading structure to logical proximity and the effectiveness and robustness of our *hasd* measure and HA-Span method for query-biased document ranking.

Comparison with the existing methods: Next, we compare the scores of our heading-aware methods with their corresponding existing methods on each evaluation measure. As shown in Table 3, all our heading-aware methods consistently achieved the even or better scores than the corresponding existing methods on all the evaluation measures. In details, our heading-aware methods improved the scores of the existing methods in 11/12 cases. This fact supports the effectiveness and robustness of our *hasd* measure. However, because of the small number of queries, the difference between the scores of our methods and the corresponding existing methods were not statistically significant except for ERR-IA@20 and α-nDCG@20 of HA-MinDist and MinDist.

9 Conclusion

In this paper, we first explained our assumptions about the effect of heading structure to logical proximity: The logical distance between a term occurrence in a heading and another term occurrence in the block associated with the heading is shorter than their simple distance, while the logical distance between two term occurrences in two different blocks is longer than their simple distance unless one of them is in a heading and the other is in its associated block. Based on these assumptions, we proposed a heading-aware proximity measure and our heading-aware variants of three existing proximity-aware document scoring methods. We then optimized the parameters of the methods, and finally evaluated all the methods by using TREC data sets. Most of optimized parameter values of the functions and evaluation scores supported the validity of our assumptions. The evaluation results also supported the robustness and usefulness of our *hasd* measure and the heading-aware Span method. Our HA-Span method achieved the best scores among all the methods on all the evaluation measures. Moreover, the score differences from the baseline scores were all statistically significant. These facts also support the effectiveness and robustness of HA-Span for query-biased document ranking.

[Acknowledgments]

This work was supported by JSPS KAKENHI Grant Number 13J06384 and Grant Number 26540163.

[Bibliography]

[1] 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.

[2] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma. VIPS: a vision-based page segmentation algorithm. Technical Report MSR–TR–2003–79, Microsoft, 2003.

[3] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma. Block-based web search. In *SIGIR*, pages 456–463, 2004.

[4] O. Chapelle, D. Metlzer, Y. Zhang, and P. Grinspan. Expected reciprocal rank for graded relevance. In *CIKM*, pages 621–630, 2009.

[5] C. L. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In *SIGIR*, pages 659–666, 2008.

[6] C. L. Clarke, M. Kolla, and O. Vechtomova. An effectiveness measure for ambiguous and underspecified queries. In *ICTIR*, pages 188–199, 2009.

[7] K. Collins-Thompson, P. N. Bennett, F. Diaz, C. Clarke, and E. M. Voorhees. TREC 2013 web track overview. In *TREC*, 2013.

[8] K. Collins-Thompson, C. Macdonald, P. N. Bennett, F. Diaz, and E. M. Voorhees. TREC 2014 web track overview. In *TREC*, 2014.

[9] R. Cummins and C. O'Riordan. Learning in a pairwise term-term proximity framework for information retrieval. In *SIGIR*, pages 251–258, 2009.

[10] E. S. de Moura, D. Fernandes, B. Ribeiro-Neto, A. S. da Silva, and M. A. Gonçalves. Using structural information to improve search in web collections. *JASIST*, 61(12):2503–2513, 2010.

[11] T. Manabe and K. Tajima. Extracting logical hierarchical structure of HTML documents based on headings. *VLDB*, 8(12):1606–1617, 2015.

[12] D. Metzler and W. Bruce Croft. Linear feature-based models for information retrieval. *Inf. Retr.*, 10(3):257–274, 2007.

[13] C. Monz. Minimal span weighting retrieval for question answering. In *IR4QA, SIGIR*, pages 23–30, 2004.

[14] M. F. Porter. An algorithm for suffix stripping. In *Readings in Information Retrieval*, pages 313–316. Morgan Kaufmann Publishers, 1997.

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

[16] S. Robertson, H. Zaragoza, and M. Taylor. Simple BM25 extension to multiple weighted fields. In *CIKM*, pages 42–49, 2004.

[17] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In *SIGIR*, pages 232–241, 1994.

[18] H. Sleiman and R. Corchuelo. A survey on region extractors from web documents. *TKDE*, 25(9):1960–1981, 2013.

[19] R. Song, M. J. Taylor, J.-R. Wen, H.-W. Hon, and Y. Yu. Viewing term proximity from a different perspective. In *ECIR*, pages 346–357, 2008.

[20] T. Tao and C. Zhai. An exploration of proximity measures in information retrieval. In *SIGIR*, pages 295–302, 2007.

Tomohiro MANABE

He is a doctoral student at Graduate School of Informatics, Kyoto University. His research interests include web search and retrieval.

Keishi TAJIMA

He is a professor at Graduate School of Informatics, Kyoto University. His research interests include web retrieval.