CRTER: Using Cross Terms to Enhance Probabilistic Information Retrieval

Jiashu Zhao
Information Retrieval and Knowledge Management Research Lab
Department of Computer Science & Engineering
York University
Toronto, Canada
jessie@cse.yorku.ca

Jimmy Xiangji Huang, Ben He
Information Retrieval and Knowledge Management Research Lab
School of Information Technology
York University
Toronto, Canada
{jhuang, benhe}@yorku.ca

ABSTRACT

Term proximity retrieval rewards a document where the matched query terms occur close to each other. Although term proximity is known to be effective in many Information Retrieval (IR) applications, the within-document distribution of each individual query term and how the query terms associate with each other, are not fully considered. In this paper, we introduce a pseudo term, namely Cross Term, to model term proximity for boosting retrieval performance. An occurrence of a query term is assumed to have an impact towards its neighboring text, which gradually weakens with the increase of the distance to the place of occurrence. We use a shape function to characterize such an impact. A Cross Term occurs when two query terms appear close to each other and their impact shape functions have an intersection. We propose a Cross Term Retrieval (CRTER) model that combines the Cross Terms' information with basic probabilistic weighting models to rank the retrieved documents. Extensive experiments on standard TREC collections illustrate the effectiveness of our proposed CRTER model.

Categories and Subject Descriptors

H.3.3 [**Information Storage & Retrieval**]: Information Search & Retrieval

General Terms

Performance, Experimentation

Keywords

Cross Term, Kernel, BM25, Proximity, Probabilistic IR

1. INTRODUCTION AND MOTIVATION

Most of the traditional Information Retrieval (IR) models are based on the assumption that query terms are independent of each other, where a document is represented as a bag of words. Nevertheless this assumption may not hold in practice. There might be some implied associations among them. For example, given a query "Vancouver Olympics", there exists an association between the query terms. Users are not looking for either other events in Vancouver or the Olympic games in other cities.

- The **Vancouver Olympics** draw to a spectacular close after 17 days of intense competition.
- **Vancouver** athletes ... 2012 London **Olympics**.

Given above two documents with both "Vancouver" and "Olympics" occurring once each, a traditional IR model will give them the same weights. Obviously the former document contains more valuable information for users than the later. The association between "Vancouver" and "Olympics" reflects the distance between them. Therefore, it is necessary to reward the document where the matched query terms appear together/close.

Many researchers have been working on proximity metrics in information retrieval [4, 11, 12]. Whereas, the nature of the associations among query terms still awaits further study. Some proximity approaches only considers adjacency [26, 27], while non-adjacent terms may also have associations. N-gram models [1, 17] consider n word sequences, which expand the radius of matching. It is yet hard to determine N, and complexity grows exponentially with the growth of N. Also, redundant proximity information may be led into the model, which may on the other hand decreases the performance of the probabilistic weighting models. Other proximity based probabilistic weighting models, such as [5, 6], add proximity information into their weighting functions in a heuristic manner. However, their experiments are not conclusive and their retrieval functions are not shown to be effective and robust enough [28].

In this paper, we present a Cross Term Retrieval model, denoted as CRTER, to model the associations among query terms in probabilistic retrieval models. A pseudo term, Cross Term, is introduced for boosting retrieval performance. A Cross Term is generated by two closely occurred query terms.

We assume that an occurrence of a query term has an impact towards its neighbouring text. This impact attenuates when a position is farther away. If we try to characterize this impact with a mathematic function, intuitively the function should satisfy the following properties: Non-negative, Continuous, Symmetric, Monotonic, Identity (See detail in Sec-

SIGIR'11, July 24–28, 2011, Beijing, China.

tion 3). We use kernel functions that have been brought to proximity retrieval [10, 16] to estimate the query term occurrence's impact. In this paper, we investigated three more kernel functions that satisfy the above properties: Quartic Kernel, Epanechnikov Kernel, and Triweight Kernel. The kernel functions are normalized with a minimum value of 0 and a maximum value of 1. When two query terms, q_i and q_j, occur closely in a document, their impact shape functions will have an intersection. We say q_i and q_j are generating terms of a Cross Term $q_{i,j}$, which occurs when q_i and q_j's impact shape functions intersect.

The corresponding impact shape functions' value at this intersection is Cross Term $q_{i,j}$'s occurrence value, which ranges from 0 to 1. The closer two query terms' occurrences are, their generated Cross Term has higher occurrence value, according to the impact shape function's properties. Therefore, the Cross Term's occurrence value indicates to which extent two query terms are correlated.

The challenge we are facing is how to estimate a Cross Term from a document collection, and integrate the Cross Term into a probabilistic weighting model. In a probabilistic weighting model, some variants change from term to term, namely the within-document Cross Term frequency (tf), the number of documents containing the Cross Term (n), and the within-query Cross Term frequency (qtf), respectively. We define the corresponding variants for Cross Terms: $tf(q_{i,j})$, $n(q_{i,j})$, and $qtf(q_{i,j})$. For $tf(q_{i,j})$, as a pseudo term, the traditional counting method of the occurrences in a document does not make much sense, especially when we aim to reward more on the Cross Terms that the generating query terms are closer. Instead of accumulating the number of occurrences, we accumulate a Cross Term's occurrence value in a document as its $tf(q_{i,j})$, which is small when generating query terms are far away, and large when generating query terms are close to each other. Please note that $tf(q_{i,j})$ is always smaller than the number of occurrences of a Cross Term in the document. In order to balance with $tf(q_{i,j})$, we define $n(q_{i,j})$ and $qtf(q_{i,j})$ correspondingly. $n(q_{i,j})$ is the accumulated Cross Term's average value on each document over the collection. In a query, since it is normally short and only contains query terms, we assume that terms in a query are densely distributed. If two terms exist in a query, they are regarded as being adjacent. Then $qtf(q_{i,j})$ is a simplified case of within-document frequency by treating the query as a document. With the defined Cross Term variants, we integrate Cross Terms into traditional BM25 weighting model [25], by treating them as special terms. Cross Terms' weights are computed and linearly combined with query terms' weights.

The remainder of this paper is organized as follows. In Section 2, we discuss the prior related work. In Section 3, we introduce the concept of Cross Term and define its variants. In Section 4, we propose Cross Term Retrieval (CRTER) model utilizing BM25 as basic weighting function. In Section 5, we set up our experimental environment on six TREC collections, and test the proposed CRTER model and compare it with some existing models. In Section 6, we conclude the paper with a discussion of our findings and future work.

2. RELATED WORK

In the 1990s, some early researchers started to investigate the effectiveness of term proximity in Information Retrieval. Allan and Ballesteros [2] indexed phases instead of terms with InQuery [7], and obtained improvements in TREC campaigns. Clarke and Cormack [9] introduced a "NEAR" operator to quantify the proximity of query terms. Hawking and Thistlewaite [12] evaluated "Span" proximity approaches on TREC data sets, which is, the text segments containing all query term instances.

Some studies heuristically integrated word proximity into probabilistic weighting models, such as [5, 6, 24, 28]. However, the nature of query term proximity still awaits further study.

N-gram IR models have been investigated as a proximity approach by researchers for years [1, 17, 18]. They are recognized as having high complexity and may lead to redundant information. Bigram models make a better balance between effectiveness and complexity. Song and Croft [26] proposed a general language model that combines bigram language models with several smoothing techniques including a Good-Turing estimate and corpus-based smoothing of unigram probabilities. The relative contributions of the different models to the query generation probability are determined empirically. Srikanth and Rohini [27] approximated the biterm probabilities using the frequency of occurrence of terms. Biterm language models are similar to bigram language models except that the constraint of order in terms is relaxed. In [3], authors proposed a language modeling approach that incorporates word pairs, without a constraint on adjacency or word order, where word pairs are determined statistical relationships, or lexical affinities, between words. Pickens [23] introduced an approach that uses non-adjacent biterms, but the particular domain, musical documents, requires an emphasis on the order of "words". We also only investigate the proximity between a pair of query terms, whereas, our approach differs from the previous studies that we propose the concept of a pseudo term, Cross Term, generated by two query terms to investigate how two query terms' occurrence change together. Cross Terms are naturally integrated into basic retrieval models as new terms, and therefore incorporate proximity into retrieval process.

Various proximity approaches integrating knowledge from other realms were investigated. [11] introduced the linkage of a query as a hidden variable, which expressed the term dependencies within the query as an acyclic, planar, undirected graph. A method of incorporating term dependence in probabilistic retrieval model was proposed in [8] by adapting Bahadur-Lazarsfeld expansion (BLE), which was originally used in the pattern recognition field. In [4], authors proposed a mathematical model based on a fuzzy proximity degree of term occurrences particularly for boolean queries. A retrieval model based on Markov random field [20] was presented for developing a general framework for modeling term dependencies.

Density functions based on proximity have been adopted to characterize term influence propagation [10, 14, 16, 19, 22]. [10] is early work that proposed to propagate the $tf \cdot idf$ score of each query term to other positions, where triangle, cosine, circle, and arc contribution functions were discussed . The highest accumulated $tf \cdot idf$ score on all the positions is adopted as the document's score. [14] uses hanning (cosine) window function to characterize the density of terms. Lv and Zhai [16] proposed a positional language model that incorporates the term proximity in a model based approach using four term propagation functions: gaussian, triangle, cosine, and circle. We utilize the above term influence prop-

agation functions in measuring Cross Terms, and test more potentially functions.

3. A NEW PSEUDO TERM: CROSS TERM

In this section, formally define the notion of Cross Term and the method of computing a Cross Term. Suppose we have a query, $Q = \{q_1, q_1, ..., q_n\}$ and a document D, where pos is one of the positions of query term q_i in document D. The term q_i will influence the positions between $pos + k$ and $pos - k$.

We assume that the impact of a matching term q_i at position pos can be captured by the value of an impact function $f_i(pos, k)$.

PROPERTY 1. *$f_i(pos, k)$ is the impact function of a query term q_i at position $pos + k$. And it must follow below 5 properties.*

(1) Non-negative: $f_i(pos, k) > 0$, the impact of a term towards its neighborhood is always non-negative(We only focus on the position influence between query terms).

(2) Continuous: $|f_i(pos, k) - f_i(pos, k + 1)|$ is small, i.e., there is a slight difference between two neighboring positions.

(3) Symmetric: $f_i(pos, -k) = f_i(pos, k)$, the term has the same impact towards two equal-distance positions.

(4) Monotonic: $f_i(pos, k) > f_i(pos, k + 1)$, where $k > 0$. The influence would decrease with the increasing of $|k|$.

(5) Identity: $f_i(pos, 0) = 1$, set one as standard influence.

Gaussian Kernel, Circle Kernel, and Triangle Kernel are widely used kernel function and satisfy all of above properties. We will discuss more about various kernel function in Section 3.4. When two query terms are close enough, their impact shape functions will join. Their point of intersection reflects the association between these two query terms.

Definition 1. Given two query terms q_i and q_j, if there exists points of intersections for impact functions $f_i(pos_1, k_1)$ and $f_j(pos_2, k_2)$, where $pos_1 + k_1$ equals to $pos_2 - k_2$, we say that a **Cross Term** occurs, denoted as $q_{i,j}$. We call q_i and q_j ***Generating Terms*** of $q_{i,j}$.

Definition 2. When a Cross Term $q_{i,j}$ occurs, the ***Cross Term's value*** is the impact function's value at the intersection.

Without previous domain knowledge of a query and a collection, we assume that the query terms are identically distributed, i.e., the query terms have the same impact shape functions. A Cross Term always locates in the middle of its generating query terms, and has higher value when the query terms occur closer.

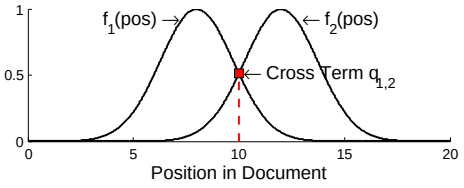


Figure 1: An example of Cross Term

Figure 1 shows a Cross Term's example. Two query terms, q_1 and q_2, located at the 8th position and the 12th position in the document. More intuitively, We adopt Gaussian Kernel as impact shape function. Their impact's shape functions are $f_1(pos_1, 2)$ and $f_2(pos_2, -2)$, accordingly. We can see that two shape functions cross over each other, and there exists an intersection at the 10th position. There is no threshold incorporated in the definitions of Cross Term and its value. For a continuous kernel function, there would always be an intersection if two terms occur in one document.

3.1 Within-Document Cross Term Frequency

Here we want to define the counting method for the frequency. The within-document frequency is the rate at which a term occurs in a document. For a single query term, its frequency in document D equals how many times it occurs in D. But for a Cross Term, to simply count its occurrence might overestimate the frequency. We introduce a new estimation of a Cross Term's within-document frequency. As we can see from Figure 1 that $q_{1.2}$'s value becomes higher if q_1 and q_2 are close and lower if there is a farther distance between q_1 and q_2. Naturally we adopt the Cross Term's value in estimating its within-document frequency.

Suppose the positions of q_i in a document are $\{pos_{1,i}, pos_{2,i}, ..., pos_{tf_i,i}\}$, where tf_i is the term frequency of q_i. Correspondingly, the positions of q_j in the document are $\{pos_{1,j}, pos_{2,j}, ..., pos_{tf_j,j}\}$, where tf_j is the term frequency of q_j. Then the within-document term frequency of $q_{i,j}$ is defined as follows.

Definition 3. The frequency of $q_{i,j}$ in D is the accumulation of $q_{i,j}$'s value.

$$tf\left(q_{i,j}, D\right) = \sum_{k_1=1}^{tf_i} \sum_{k_2=1}^{tf_j} Kernel\left(\frac{1}{2} dist(pos_{k_1,i}, pos_{k_2,j})\right) \quad (1)$$

where tf is the term frequency of $q_{i,j}$ in D, $Kernel(\cdot)$ is the kernel function adopted in query term's impact function, and $dist(\cdot)$ is the distance between two positions

$$dist(pos_{k_1,i}, pos_{k_2,j}) = |pos_{k_1,i} - pos_{k_2,j}|$$

Please note that the frequency of a Cross Term might not be an integer. Meanwhile, various of kernel functions will be studied in 3.4.

3.2 Number of Documents with A Cross Term

To evaluate the number of documents containing a Cross Term $q_{i,j}$, it is not reasonable to simply count how many documents in which $q_{i,j}$ occurs. The contribution from a query term and a Cross Term are different. For a query term q_i, an occurrence means its frequency accumulates 1, and the number of documents containing q_i is

$$n(q_i) = \sum_{D \in Index} \mathbf{1}_{\{qi \in D\}}$$

where $\mathbf{1}_{\{qi \in D\}}$ is an indicator function, which equals to 1 if $q_i \in Doc_i$ and equals to 0 otherwise. On the other hand, an occurrence of a Cross Term adds a value less than 1 to its frequency. A Cross Term's value could be various, and ranges from 0 to 1. Thus the difference of Cross Terms should be shown in its variants.

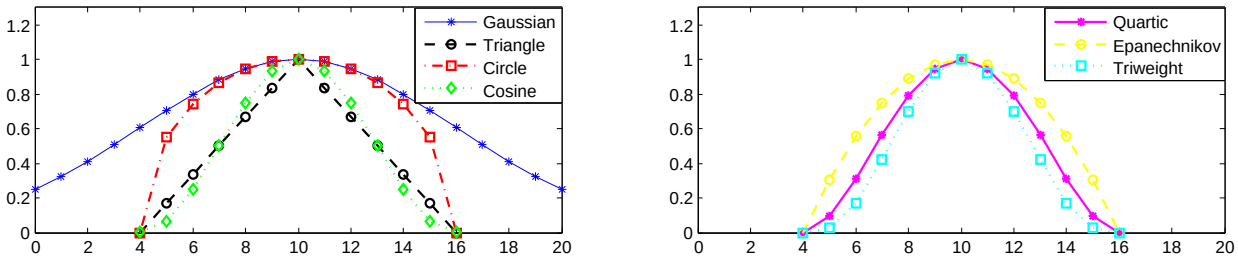


Figure 2: Kernel Functions

Definition 4. The number of documents containing a Cross Term $q_{i,j}$, is the sum of $q_{i,j}$'s average value on each document, shown as follows

$$n(q_{i,j}) = \sum_{D \in Index, Occur(q_{i,j},D) \neq 0} \frac{tf(q_{i,j}, D)}{Occur(q_{i,j}, D)} \quad (2)$$

where $Occur$ is the number of occurrence of $q_{i,j}$, which is

$$Occur(q_{i,j}, D) = \sum_{k_1=1}^{tf_i} \sum_{k_2=1}^{tf_j} \mathbf{1}_{\{Kernel(\frac{1}{2} dist(pos_{k_1,i}, pos_{k_2,j})) \neq 0\}}$$

3.3 Within-Query Cross Term Frequency

To evaluate a Cross Term's within-query term frequency, we can track each positions of q_i and q_j the same way as within-document frequency by the sum of all possible intersections. Moreover, different from in documents, query terms distribute densely in a query, so we can assume that query terms are adjacent to each other.

Definition 5. The within-query frequency of $q_{i,j}$ is

$$qtf(q_{i,j}) = Kernel(\frac{1}{2}) \cdot min\left(qtf(q_i), qtf(q_j)\right) \quad (3)$$

where $qtf(q_i)$ and $qtf(q_j)$ are within query term frequencies of q_i and q_j, and Kernel is the same kernel function utilized in $tf(q_{i,j}, D)$.

3.4 Kernel Functions

We first apply four kernel functions that have been brought into IR applications. Among them, the Gaussian kernel is widely used in statistics and machine learning algorithms such as the Support Vector Machines. Moreover, the Triangle kernel, Circle Kernel, and Cosine Kernel come from basic genomic graphics, which are applied to estimate the proximity-based density distribution for the positional language model [16]. below:

- Gaussian Kernel:

$$Kernel(u) = exp[\frac{-u^2}{2\sigma^2}] \quad (4)$$

- Triangle Kernel:

$$Kernel(u) = (1 - \frac{u}{\sigma}) \cdot \mathbf{1}_{\{u \leq \sigma\}} \quad (5)$$

- Circle Kernel:

$$Kernel(u) = \sqrt{1 - (\frac{u}{\sigma})^2} \cdot \mathbf{1}_{\{u \leq \sigma\}} \quad (6)$$

- Cosine Kernel:

$$Kernel(u) = \frac{1}{2}[1 + cos(\frac{u\pi}{\sigma})] \cdot \mathbf{1}_{\{u \leq \sigma\}} \quad (7)$$

where σ is a normalization parameter, and $\mathbf{1}_{\{u \leq \sigma\}}$ is indicator function, which equals to 1 if $u \leq \sigma$ and equals to 0 otherwise.

Moreover, some other kernel functions satisfy the query term impact function's properties (Property 1). We also introduce them into IR since the kernel functions are not maturely applied in Information Retrieval. They are: Quartic Kernel, Epanechnikov Kernel and Triweight Kernel, shown as the following. The shape of the included kernel functions are shown in Figure 2. The performance of using all the kernel functions will be investigated in the experiments.

- Quartic Kernel:

$$Kernel(u) = (1 - (\frac{u}{\sigma})^2)^2 \cdot \mathbf{1}_{\{u \leq \sigma\}} \quad (8)$$

- Epanechnikov Kernel:

$$Kernel(u) = (1 - (\frac{u}{\sigma})^2) \cdot \mathbf{1}_{\{u \leq \sigma\}} \quad (9)$$

- Triweight Kernel:

$$Kernel(u) = (1 - (\frac{u}{\sigma})^2)^3 \cdot \mathbf{1}_{\{u \leq \sigma\}} \quad (10)$$

4. CROSS TERM RETRIEVAL MODEL

We propose a Cross Term Retrieval (CRTER) model, utilizing the Cross Term as a special term. A new combined weighting model for a document is

$$CRTER(D) = (1-\lambda) \cdot \sum_{1 \leq i \leq K} w(q_i, D) + \lambda \cdot \sum_{1 \leq i < j \leq K} w'(q_{i,j}, D) \quad (11)$$

where w is the weighting function of query terms in Q, w' is the Cross Term $q_{i,j}$'s weight, and λ is a parameter balancing the query terms and Cross Terms. When λ equals to 0, the retrieval model uses query terms only, which is the standard

weighting model. When λ equals to 1, the retrieval model uses Cross Terms only. Since the weights of query terms and Cross Terms are normalized independently, the value of λ reflects the influence of using Cross Terms. In this paper, we use BM25 as the basic weighting model in CRTER. BM25 is a classical weighting function employed by the Okapi system [25]. As shown by previous TREC experimentation, BM25 usually provides very effective retrieval performance on the TREC collections that are used in [29]

4.1 BM25

In BM25, the weight of a search term is assigned based on its within-document term frequency and query term frequency. The corresponding weighting function is as follows.

$$w(q_i, D) = \frac{(k_1+1)*tf}{K+tf} * \frac{(k_3+1)*qtf}{k_3+qtf} * \log\frac{N-n+0.5}{n+0.5} \quad (12)$$

where w is the weight of a query term q_i in a document. The variants in the above formula can be grouped into two categories as follows:

- The first category of variants are query independent. In this category, N is the number of indexed documents in the collection. k_1 and k_3 are tuning constants which depend on the dataset used and possibly on the nature of the queries. K equals to $k_1 * ((1-b) + b * dl/avdl)$, dl is the length of the document, and $avdl$ is the average document length. In our experiments, the values of k_1, k_3 are default to 1.2 and 8, respectively, which is the recommended setting in [25].

- Values of the other group of variants change from query to query. In this category, n is the number of documents containing a specific term. tf is within-document term frequency. qtf is within-query term frequency.

Finally, a document's weight for a query is given by the sum of its weight for each terms in the query,

$$BM25(D) = \sum_{i=1}^{i<K} w(q_i, D) \quad (13)$$

where w is the term weight obtained from Equation 12.

4.2 Algorithm and Time Complexity

In this section, we present the CRTER Algorithm with an analysis on tis time complexity. We show that the time complexity of our proposed algorithm is at the same level of the basic weighting model. Figure 3 is the algorithm for Cross Term Retrieval (CRTER) model. In the rest of this section, we will analyze its time complexity.

Suppose the number of terms in a query Q is $|Q|$, the number of documents in an index is $|Index|$, the average document length is $\overline{|D|}$. The first 6 steps are query processing that the within-query term frequency is computed for both query terms and Cross Terms, and the time complexity of this part is $O(|Q|^2)$. The remaining steps are document processing through the index. Also both query terms and Cross Terms are computed separately. For steps 8 to 12, this is standard document processing except storing term positions as well, and the time complexity of this part is

```
1: for all q_i ∈ Q do
2:   Compute qtf(q_i)
3: end for
4: for all q_i, q_j ∈ Q do
5:   Compute qtf(q_{i,j})
6: end for
7: for all D ∈ Index do
8:   for all q_i ∈ Q do
9:     Compute tf(q_i, D)
10:    Store q_i's positions on D in an array
11:    Compute n(q_i) = n(q_i) + 1
12:  end for
13:  for all q_i, q_j ∈ Q do
14:    for k_1 < tf(q_i, D)&k_2 < tf(q_j, D)
15:      Kernel_temp = Kernel(1/2 |pos_{k_1,i} − pos_{k_2,j}|)
16:      if Kernel_temp ≠ 0
17:        tf(q_{i,j}, D)+ = Kernel_temp
18:        Occur(q_{i,j}, D)+ = 1
19:      end if
20:    end for
21:    Compute n(q_{i,j}) = n(q_{i,j}) + tf(q_{i,j},D)/Occur(q_{i,j},D);
22:  end for
23: end for
24: Compute w(q_i, D)
25: Compute w'(q_{i,j}, D);
26: Compute CRTER(D)
27: Rank documents according to CRTER(D)
```

Figure 3: Algorithm for CRTER

$O(|Q| \cdot \overline{|D|})$. In steps 14 to 20, $tf(q_{i,j}, D)$ and $Occur(q_{i,j}, D)$ are accumulated, and it takes $O(\overline{tf}^2)$ time, where $\overline{tf}$ is average within-document term frequency. Thus computing all Cross Terms' variants on one document takes $O(|Q|^2 \cdot \overline{tf}^2)$. From step 24 to step 27, the weights for query terms and Cross Terms are computed and the documents are ranked according to CRTER. This algorithm for CRTER model has the time complexity of the follows

$$O(|Q|^2 + |Index| \cdot (|Q| \cdot \overline{|D|} + |Q|^2 \cdot \overline{tf}^2)) \quad (14)$$

Generally the number of query terms $|Q|$ in a submitted query is far smaller than the number of documents in index, i.e., $|Q| << |Index|$, therefore the first $|Q|^2$ in Forumla 14 could be eliminated. The use of Cross Terms is represented as $|Q|^2 \cdot \overline{tf}^2$ in Formula 14. The percentage of query terms' within-document frequency over the documents' length is normally very low. Formula 14 will become

$$O(|Index| \cdot (|Q| \cdot \overline{|D|})) \quad (15)$$

when $|Q| \cdot \overline{tf}^2 \leq \overline{|D|}$. This means the time complexity of CRTER is the same as a basic probabilistic weighting model.

5. EXPERIMENTAL RESULTS

5.1 Data Sets and Evaluation Metrics

We conduct a series of experiments on six standard TREC collections shown in Table 1. These collections are diverse

in both sizes and content, which facilitate a thorough evaluation of our proposed CRTER model. The TREC8 contains newswire articles from various sources, such as Financial Times (FT), the Federal Register (FR) etc., which are usually considered as high-quality text data with little noise. AP88-89 contains articles published by Association Press from the year of 1988 to 1989. The WT2G collection is a 2G size crawl of Web documents. The WT10G collection is a medium size crawl of Web documents, which was used in the TREC 9 and 10 Web tracks. It contains 10 Gigabytes of uncompressed data. The .GOV2 collection, which has 426 Gigabytes of uncompressed data, is a crawl from the .gov domain. This collection has been employed in the TREC 14 (2004), 15 (2005) and 16 (2006) Terabyte tracks. The Blog06 collection includes 100,649 blog feeds collected over an 11 week period from December 2005 to February 2006. Following the official TREC settings [21], we index only the permalinks, which are the blog posts and their associated comments. For all test collections used, each term is stemmed using Porter's English stemmer, and standard English stopwords are removed.

Collection Name	# of Docs	Topics	# of Topics
TREC8	528,155	401-450	50
AP88-89	164,597	51-100	50
WT2G	247,491	401-450	50
WT10G	1,692,096	451-550	100
.GOV2	25,178,548	701-850	150
Blog06	3,215,171	851-950	150

Table 1: Overview of the TREC collections used.

Each topic contains three topic fields, namely title, description and narrative. We only use the title topic field that contains very few keywords related to the topic. The title-only queries are usually short which is a realistic snapshot of real user queries in practice. On each collection, we evaluate our proposed model by a 10-fold cross-validation. The test topics associated to each collection are randomly split into ten equal subsets. In each fold, 9 subsets of the test topics are used for training, and the remaining subset is used for testing. The overall retrieval performance is averaged over all 10 test subsets of topics.

We use the TREC official evaluation measures in our experiments, namely the topical MAP on Blog06 [21], and the Mean Average Precision (MAP) on the other five collections [29]. To emphasize on the top retrieved documents, we also include P@5 and P@20 in the evaluation measures. All statistical tests are based on Wilcoxon Matched-pairs Signed-rank test.

5.2 Effectiveness of CRTER

We first investigate how much our proposed Cross Term Retrieval (CRTER) model can boost BM25. We use optimal BM25 as our baseline. The parameter b is set to be 0.35, which is shown to be optimal in our preliminary experiments (See Figure 7). The related experimental results are presented in Table 2. Seven different kernel functions are applied to instantiate the CRTER model, including: Gaussian, Triangle, Circle, Cosine, Quartic, Epanechnikov, and Triweight kernels. All the results are evaluated by MAP, P@5, and P@20. The percentage of how much CRTER outperforms BM25 is also listed. The best result obtained on each collection is marked bold. As shown by the results, our proposed CRTER model outperforms BM25 on all six collections used. The advantage of CRTER over BM25 is especially evident on the .GOV2 and Blog06 Web collections, where statistically significant improvement is observed with all 7 kernel functions used. Moreover, according to the results in Table 2, each kernel function has its advantage on some aspects. There is no single kernel function can outperform others on all the datasets.

5.3 Parameter Sensitivity

An important issue that may affect the robustness of the CRTER model is the sensitivity of its parameters λ (in Equation 11) and σ (in Equation 4-10) to retrieval performance. The parameter λ balances the query terms and the Cross Terms. When λ equals to 0, the retrieval model uses query terms only, which is the standard BM25 weighting model. When λ equals to 1, the retrieval model uses Cross Terms only. Since the weights of query terms and Cross Terms are normalized independently, the value of λ reflects the influence of using Cross Terms. The kernel parameter σ controls the range of a query term's impact. When σ is small, a Cross Term occurs only if its generating term are very close. When σ is large, query terms far away from each other can generate a Cross Term. But the Cross Term's value will be different according to the distance between query terms.

Figure 4 plots the evaluation metrics MAP, P@5, and P@20 obtained by CRTER over λ values ranging from 0 to 1 on WT2G. In addition, a group of different settings of σ are applied, namely $\sigma = 2, 5, 10, 20, 50, 75, 100$. The general tendency on each evaluation metric is similar. As we can see from Figure 4, CRTER's retrieval performance decreases with large λ values. CRTER generally performs well over different datasets when λ locates between 0 and 0.2. Overall, a λ value between 0 and 0.2 is recommended as it is shown to be reliable. In addition, CRTER's retrieval performance increases with large σ values, and tends to be stable when σ is larger than 20.

Figure 5 plots the CRTER model's performance with all kernel functions against kernel parameter σ on TREC8 dataset. It illustrates the effect of kernel parameters in detail. The increment of σ is 1 at the beginning and becomes larger after 10, because the model is more sensitive when σ is smaller. For a given σ, the range of CRTER's performance under different kernel functions is shown as a segment. The CRTER model overall appears to be effective over a wide range of σ values. When σ becomes larger, CRTER's performance normally increases at the beginning, due to the incorporation of term proximity. However, a larger σ value brings noise to the model and therefore decreases CRTER's performance. A σ value between 20 and 25 is recommended to be a reliable setting.

5.4 Robustness and Generalized Performance

The proposed CRTER model's robustness is important to its applications in practice. Ideally, we would like to have a reliable retrieval performance using CRTER on various datasets with its parameters within a stable safe range. This issue is particularly crucial for a given new dataset without training data.

We first fix BM25's parameter b = 0.35, and evaluate the robustness of CRTER provided by an empirical setting, ob-

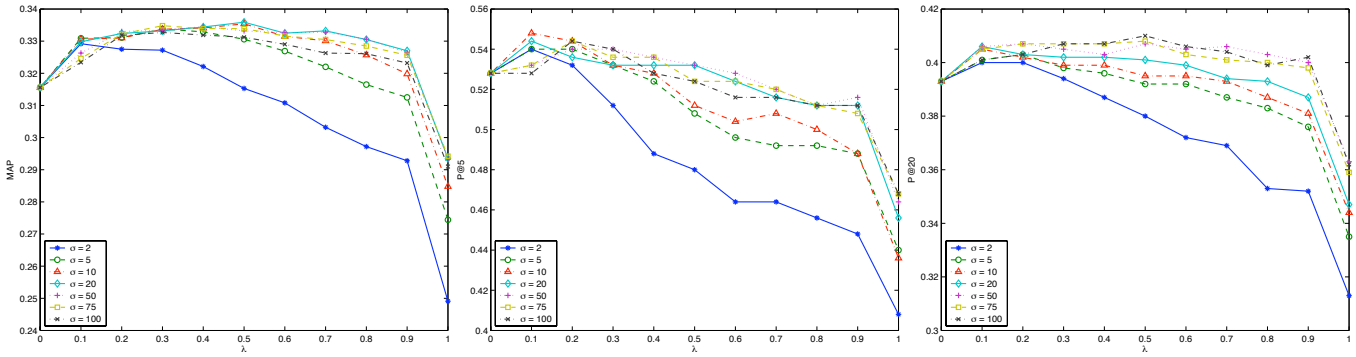
	Eval Metric	TREC8	AP88-89	WT2G	WT10G	.GOV2	Blog06
BM25	MAP	0.2561	0.2710	0.3156	0.2119	0.3039	0.3246
	P@5	0.4920	0.4360	0.5280	0.3800	0.6134	0.6400
	P@20	0.4000	0.3860	0.3930	0.2670	0.5426	0.5997
CRTER Gaussian	MAP	0.2604 (+1.679%)	0.2787 (+2.841%)	0.3354 (+6.274%)	0.2213 (+4.436%)	**0.3322** (+9.312%)	0.3484 (+7.332%)
	P@5	0.5040 (+2.439%)	0.4520 (+3.670%)	**0.5480** (+3.788%)	**0.4080** (+7.368%)	**0.6336** (+3.293%)	0.6440 (+0.625%)
	P@20	0.4190 (+4.750%)	0.3900 (+1.036%)	0.4070 (+3.562%)	0.2775 (+3.933%)	**0.5691** (+4.884%)	0.6147 (+2.501%)
CRTER Triangle	MAP	**0.2606** (+1.757%)	**0.2789** (+2.915%)	**0.3359** (+ 6.432%)	0.2207 (+4.153%)	0.3287 (+8.161%)	0.3494 (+7.640%)
	P@5	0.5040 (+2.439%)	0.4520 (+3.670%)	**0.5480** (+3.788%)	**0.4080** (+7.368%)	**0.6336** (+3.293%)	0.6507 (+1.672%)
	P@20	0.4190 (+4.750%)	0.3890 (+0.777%)	**0.4100** (+4.326%)	0.2775 (+3.933%)	0.5671 (+4.515%)	0.6170 (+2.885%)
CRTER Circle	MAP	0.2599 (+1.484%)	0.2783 (+2.694%)	**0.3359** (+ 6.432%)	**0.2227** (+5.97%)	0.3264 (+7.404%)	0.3487 (+7.425%)
	P@5	0.5040 (+2.439%)	**0.4600** (+5.505%)	0.5440 (+ 3.030%)	0.4060 (+6.842%)	0.6282 (+ 2.413%)	0.6453 (+0.828%)
	P@20	0.4190 (+4.750%)	0.3890 (+0.777%)	0.4080 (+ 3.817%)	**0.2785** (+4.307%)	0.5631 (+3.778%)	0.6137 (+2.335%)
CRTER Cosine	MAP	0.2599 (+1.484%)	**0.2789** (+2.915%)	0.3358 (+ 6.401%)	0.2216 (+4.578%)	0.3289 (+8.226%)	0.3501 (+7.856%)
	P@5	0.5040 (+2.439%)	0.4560 (+4.587%)	0.5440 (+3.030 %)	**0.4080** (+7.368%)	**0.6336** (+3.293%)	0.6533 (+2.078%)
	P@20	0.4190 (+4.750%)	**0.3910** (+1.295%)	0.4090 (+ 4.071%)	0.2765 (+3.558%)	0.5668 (+4.460%)	0.6316 (+5.319%)
CRTER Quartic	MAP	0.2599 (+1.484%)	0.2787 (+2.841%)	0.3352 (+6.210 %)	0.2212 (+4.389%)	0.3258 (+7.206%)	0.3497 (+7.733%)
	P@5	0.5040 (+2.439%)	0.4560 (+4.587%)	0.5440 (+ 3.030%)	**0.4080** (+7.368%)	0.6322 (+3.065%)	0.6533 (+2.078%)
	P@20	0.4170 (+4.250%)	0.3920 (+1.554%)	**0.4100** (+ 4.326%)	0.2780 (+4.120%)	0.5658 (+4.276%)	0.6173 (+2.935%)
CRTER Epanechnikov	MAP	0.2602 (+1.601%)	0.2787 (+2.841%)	0.3343 (+ 5.925%)	0.2217 (+4.625%)	0.3277 (+7.832%)	0.3482 (+7.270%)
	P@5	**0.5080** (+3.252%)	0.4520 (+3.670%)	0.5440 (+ 3.030%)	**0.4080** (+7.368%)	**0.6336** (+3.293%)	0.6467 (+1.047%)
	P@20	**0.4200** (+5.000%)	0.3890 (+0.777%)	0.4070 (+ 3.562%)	0.2785 (+4.307%)	0.5654 (+4.202%)	0.6147 (+2.501%)
CRTER Triweight	MAP	0.2601 (+1.562%)	0.2788 (+2.878%)	0.3356 (+ 6.337%)	0.2225 (+5.002%)	0.3293 (+8.358%)	**0.3506** (+8.010%)
	P@5	**0.5080** (+3.252%)	0.4600 (+5.505%)	0.5440 (+ 3.030%)	0.4060 (+6.842%)	**0.6336** (+3.293%)	**0.6587** (+2.922%)
	P@20	0.4180 (+4.500%)	**0.3910** (+1.295%)	0.4070 (+ 3.562%)	0.2780 (+4.120%)	0.5674 (+4.571%)	**0.6187** (+3.168%)

Table 2: Comparison between BM25 baseline and CRTER with different kernel functions. BM25 parameter b is initialized to be 0.35. All the results are evaluated by MAP, P@5, and P@20. CRTER outperforms BM25 on all collections.

tained from previous observations, namely Triangle Kernel, $\sigma = 25$, and $\lambda = 0.2$. We compare this empirical setting with the following optimization strategies: first, optimize each of kernel function, σ or λ while setting the other parameter to the empirical value; second, optimize all of kernel functions, σ and λ. Figure 6 presents the related results on 4 datasets: TREC8, AP88-89, WT2G and WT10G. From this figure, we can see that the performance obtained by the empirical setting is comparable to the retrieval performance obtained by the optimized parameter settings on all datasets used.

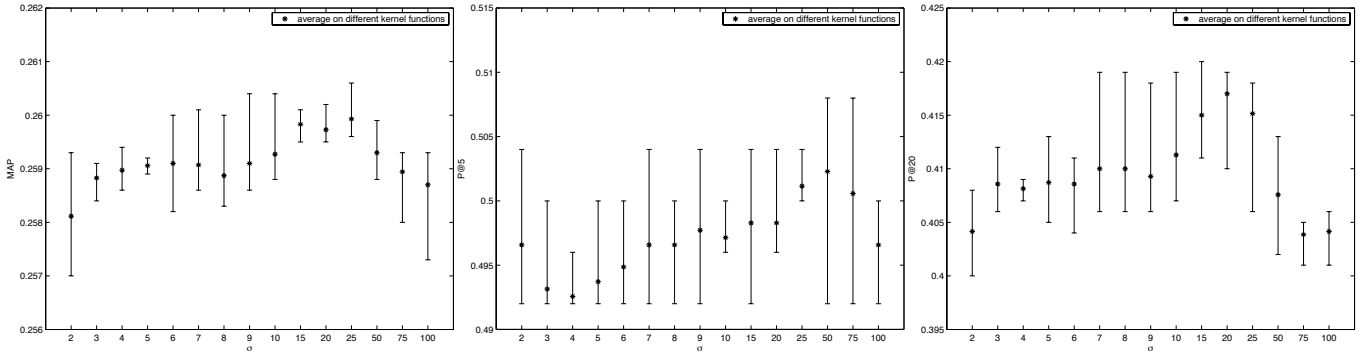
We also test CRTER's performance under the same empirical setting with BM25 over different BM25 parameters on all the six collections. In BM25, b ranges from 0.15 to 0.95. In Figure 7, MAP of BM25 changes over different b, and CRTER can boost BM25 under all b's settings and over all the collections. More specifically, for .GOV2 dataset, we can see that BM25's performance is very sensitive to b, its MAP gradually increases with the incrementment of b at the beginning, and sharply decreases later. CRTER, on the other hand, boosts basic BM25 over different b, and tends to stabilize the retrieval performance.

In general, CRTER performs robustly and has strong gen-



(a) MAP (b) P@5 (c) P@20

Figure 4: Sensitivity to CRTER parameter λ with different kernel parameters on WT2G



(a) MAP (b) P@5 (c) P@20

Figure 5: Sensitivity to CRTER kernel parameter σ on TREC8 (Each segment gives the range of results using different kernel functions under the given kernel parameter σ. The lowest point is the minimal result, the highest point is the maximal result, and the point in the middle is the average result.)

eralized performance. Without much knowledge of a new dataset, a group of parameters are recommended for CRTER: Triangle Kernel, $\sigma = 25$, and $\lambda = 0.2$.

5.5 Comparison with Positional Language Model

We study how the proposed CRTER model performs compared to the state-of-the-art approaches. In particular, its effectiveness is evaluated over the positional language model proposed by Lv et al. [16]. The positional language model (PLM) estimates a language model for every single position in a document. Among various kernel functions tested, the Gaussian kernel is shown to provide the best retrieval performance [16].

We compare CRTER's results with PLM on the datasets used in [16], including AP88-89, WT2G and TREC 8. As illustrated in Table 3, PLM improves the language model baseline (LM) by 1.3%, 2.0%, and 1.1% on TREC8, AP88-89 and WT2G, respectively, while the corresponding improvement over BM25 by CRTER is 1.8%, 2.9% and 6.4%, respectively. Overall, CRTER's retrieval performance is at least comparable to, if not better than PLM in our experiments.

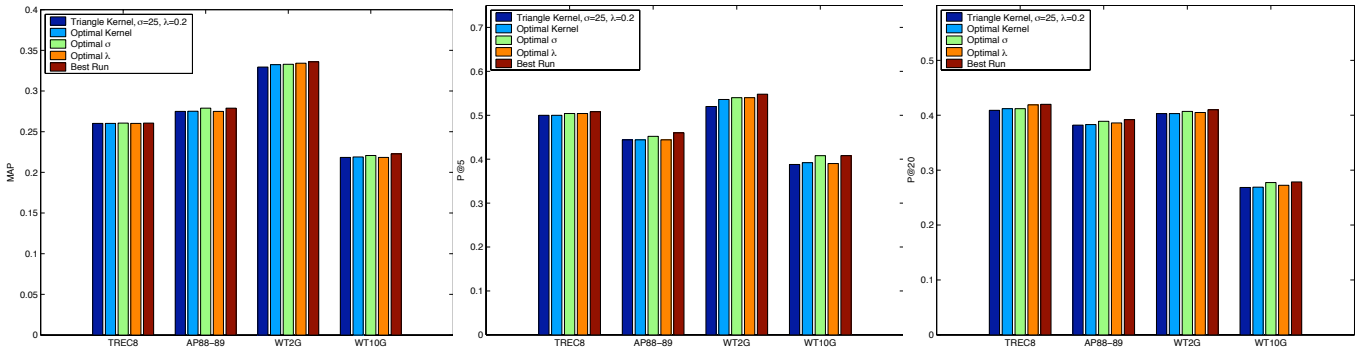
	TREC8	AP88-89	WT2G
LM	0.2518	0.2154	0.3249
PLM	0.255(1.3%)	0.2198(2.0%)	0.3285(1.1%)
BM25	0.2561	0.2710	0.3156
CRTER	0.2606(1.8%)	0.2789(2.9%)	0.3359(6.4%)

Table 3: Direct MAP Comparison with PLM

What is more, another advantage of CRTER model is that it is more flexible that it can be fitted into more basic probabilistic weighting models other than BM25, using defined variants of Cross Term.

5.6 A Case Study

To further analyze the effectiveness of the proposed CRTER model, out of the 550 test topics used in our experiments, we conduct a case study on topic 867 on the Blog06 collection. Using the judged documents, namely the golden standard provided by TREC, we explore the factors that could differentiate relevant documents from non-relevant ones. The statistics about the distribution of the query terms in the relevant and non-relevant documents is shown in Table 4.



(a) MAP (b) P@5 (c) P@20

Figure 6: Robustness of CRTER:Compare CRTER's retrieval performance provided by an empirical setting, namely Triangle Kernel, $\sigma = 25$, and $\lambda = 0.2$ with the following optimization strategies: first, optimize each of kernel function, σ or λ while setting the other parameter to the empirical value; second, optimize all of kernel functions, σ and λ

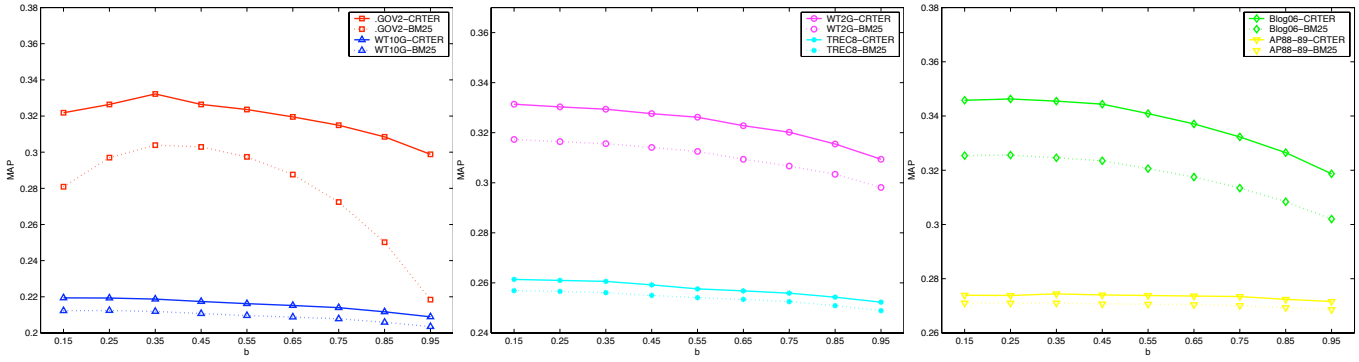


Figure 7: Generalized Performance of CRTER: Compare MAP between CRTER and BM25 with the change of b. (CRTER uses fixed parameter: Triangle Kernel, $\sigma = 25$, $\lambda = 0.2$)

The relevant and non-relevant documents cannot be distinguished from each other by neither their average minimum distance nor the average distance (less than 10). Therefore, traditional proximity-based approaches may not be able to boost the relevant documents of this specific topic. However, the average Cross Term's within document frequency (in CRTER model) of the relevant documents is clearly higher than the non-relevant ones, indicating that our proposed CRTER model can effectively identify the relevant documents from the non-relevant ones for this topic.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we introduce the concept of Cross Term to integrate the context of the query term proximity into IR models. The Cross Term measures the association that two terms have in the textual proximity context. A query term's influence to their neighboring text is approximated by a kernel function, which gradually decreases with the distance to the term. The Cross Term is then defined as the overlapped influence of the two terms. Based on this idea, we propose a Cross Term Retrieval (CRTER) model, where within-document, within-query frequency of a Cross Term and the number of documents that contain the Cross Term are well defined. Through extensive experiments on standard TREC collections with various kernel functions, we show that the proposed model outperforms the BM25 baseline, and is at least comparable to the state-of-the-art Positional Language model. Furthermore, how the setting of the balancing parameter in the CRTER model and the shape parameter of kernel function affect CRTER's effectiveness are discussed. Meanwhile, a group of optimal parameters shows the robustness of the CRTER model on all collections used.

Our proposed concept of Cross Term has various promising future research directions. For example, we can apply the CRTER model to other classical IR models, such as language modeling, and the divergence from randomness models. We can also conduct an in-depth study on the Cross Term's distribution in documents collections, and examine which of the kernel functions fits the best to the actual distribution.

7. ACKNOWLEDGEMENT

This research is supported by the research grant from the

	# of Docs	# of Docs containing both terms	Average min dist. between query terms	Average dist. for dist. <10	Average Cross Term Freq. in Docs
Relevant	702	673	2.34	4.45	7.08
Non-Relevant	867	324	2.32	3.84	1.96

Table 4: A Case Study: Distribution of Terms in Relevant and Non-Relevant Documents. (Topic 867 on the Blog06 Collection)

Natural Sciences & Engineering Research Council (NSERC) of Canada and the Early Researcher Award/ Premier's Research Excellence Award. We thank the four anonymous reviewers for their thorough review on this paper.

8. REFERENCES

[1] F. Ahmed and A. Nurnberger. Evaluation of n-gram conflation approaches for Arabic text retrieval. *Journal of the American Society for Information Science and Technology*, 60(7):1448-1465, 2009.

[2] J. Allan, L. Ballesteros, J.P. Callan, W.B. Croft, and Z. Lu Recent experiments with INQUERY. In *Proc. of the 4th TREC*, pages 49-64, 1995.

[3] C. Alvarez, P. Langlais, and J. Nie. Word pairs in language modeling for information retrieval. In *Proc. 7th International Conference on Computer Assisted Information Retrieval*, pages 686-705. Citeseer, 2004.

[4] M. Beigbeder and A. Mercier. An information retrieval model using the fuzzy proximity degree of term occurences. In *Proc. of the 2005 ACM symposium on Applied computing*, pages 1018-1022. ACM, 2005.

[5] A. Broschart and R. Schenkel. Proximity-aware scoring for XML retrieval. In *Proc. of the 31th ACM SIGIR*, pages 845-846. ACM, 2008.

[6] S. Buttcher, C. Clarke, and B. Lushman. Term proximity scoring for ad-hoc retrieval on very large text collections. In *Proc. of the 29th ACM SIGIR*, page 622-623. ACM, 2006.

[7] J.P. Callan, W.B. Croft, and S.M. Harding The INQUERY retrieval system. *Proceedings of the 3rd International Conference on Database and Expert Systems Applications*, pages 78-83, Citeseer. 1992

[8] B. Cho, C. Lee, and G. Lee. Exploring term dependences in probabilistic information retrieval model. *Information Processing and Management*, 39(4):505-519, 2003.

[9] C. Clarke, G. Cormack, and F. Burkowski. Shortest substring ranking (MultiText experiments for TREC-4). In *Proc. of the 4th TREC*, pages 295-304, 1996.

[10] O. de Kretser and A. Moffat. Effective document presentation with a locality-based similarity heuristic. In *Proc. of the 22th ACM SIGIR*, pages 113-120. ACM, 1999.

[11] J. Gao, J. Nie, G. Wu, and G. Cao. Dependence language model for information retrieval. In *Proc. of the 27th ACM SIGIR*, page 177. ACM, 2004.

[12] D. Hawking and P. Thistlewaite. Proximity operators - So near and yet so far. In *Proc. of the 4th Text Retrieval Conference*, pages 131-143, 1995.

[13] S. Kirkpatric, C. Gelatt, and M. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671-680, 1983.

[14] K. Kise, M. Junker, A. Dengel, and K. Matsumoto. Passage retrieval based on density distributions of terms and its applications to document retrieval and question answering. *Lecture notes in computer science*, pages 306-327. Springer, 2004.

[15] E. Krause. Taxicab Geometry. *Mathematics Teacher*, 66(8):695-706, 1973.

[16] Y. Lv and C. Zhai. Positional language models for information retrieval. In *Proc. of the 32th ACM SIGIR*, pages 299-306. ACM, 2009.

[17] J. Mayfield and P. McNamee. Single n-gram stemming. In *Proc. of the 26th ACM SIGIR*, pages 415-416. ACM, 2003.

[18] P. McNamee and J. Mayfield. Character n-gram tokenization for European language text retrieval. *Information Retrieval*, 7(1):73-97, 2004.

[19] A. Mercier and M. Beigbeder. Fuzzy proximity ranking with boolean queries. *Proc. of TREC*, 2005.

[20] D. Metzler and W. Croft. A Markov random field model for term dependencies. In *Proc. of the 28th ACM SIGIR*, pages 472-479. ACM, 2005.

[21] I. Ounis, M. De Rijke, C. Macdonald, G. Mishne, and I. Soboroff. Overview of the TREC-2006 blog track. *Proc. of TREC*, 2006.

[22] D. Petkova and W. Croft. Proximity-based document representation for named entity retrieval. In *Proc. of the 16th ACM CIKM*, pages 731-740. ACM, 2007.

[23] J. Pickens. A comparison of language modeling and probabilistic text information retrieval approaches to monophonic music retrieval. In *Proc. of ISMIR*. Citeseer, 2000.

[24] Y. Rasolofo and J. Savoy. Term proximity scoring for keyword-based retrieval systems. *Lecture Notes in Computer Science*, pages 207-218. Springer, 2003.

[25] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-4. In *Proc. of the 4th TREC*, pages 73-97, 1996.

[26] F. Song and W. Croft. A general language model for information retrieval. In *Proc. of the 8h ACM CIKM*, pages 316–321. ACM, 1999.

[27] M. Srikanth and R. Srihari. Biterm language models for document retrieval. In *Proc. of the 25th ACM SIGIR*, pages 425-426. ACM, 2002.

[28] T. Tao and C. Zhai. An exploration of proximity measures in information retrieval. In *Proc. of the 30th ACM SIGIR*, pages 295-302. ACM, 2007.

[29] E. Voorhees and D. Harman. *TREC: Experiment and evaluation in information retrieval*. MIT Press, 2005.