A Proximity Language Model for Information Retrieval

Jinglei Zhao
iZENEsoft, Inc.
1 PuDong Ave. Shanghai, China
zhao.jinglei@gmail.com

Yeogirl Yun
Wisenut, Inc.
Korea Land Corporation Building, 1009-1
Daechi-dong, Gangnam-gu, Seoul, Korea
yeogirl@gmail.com

ABSTRACT

The proximity of query terms in a document is a very important information to enable ranking models go beyond the "bag of word" assumption in information retrieval. This paper studies the integration of term proximity information into the unigram language modeling. A new proximity language model (PLM) is proposed which views query terms' proximity centrality as the Dirichlet hyper-parameter that weights the parameters of the unigram document language model. Several forms of proximity measure are developed to be used in PLM which could compute a query term's proximate centrality in a specific document. In experiments, the proximity language model is compared with the basic language model and previous works that combine the proximity information with language model using linear score combination. The experiment results show that the proposed model performs better in both top precision and average precision.

Categories and Subject Descriptors

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

General Terms

Algorithms,Experimentation,Theory

Keywords

Information Retrieval,Proximity Model, Proximity Language Model

1. INTRODUCTION

A key task in information retrieval is to rank a collection of documents according to their respective relevance to a user query. Probabilistic models have been successfully applied in document ranking, such as the traditional probabilistic model [23, 13, 24] and stochastic language model [21, 15, 29] etc. By assuming full independence between terms, those models represent query and documents as bag of words and mainly exploit statistics such as term frequency, inverse document frequency and document length in ranking. One shortcoming of those traditional ranking models lies in that they don't consider the proximity of the query terms within a document.

Proximity represents the closeness or compactness of the query terms appearing in a document. The underlying intuition is that the more compact the terms, the more likely that they are topically related, and thus the more possible the document is relevant to the concept represented in the user query. Proximity can be seen as a kind of indirect measure of term dependence. [2] shows that term proximity has a strong influence on the dependence between terms.

Several works have been done towards incorporating proximity factor into existing ranking functions [22, 3, 27, 1]. Those works show that a properly devised proximity measure can improve the effectiveness of probabilistic ranking functions. However, there are two main shortcomings in previous works. One is that the underlying proximity structure of the individual query terms is not exploited. As investigated in [27], two kinds of proximity mechanisms could be distinguished in those approaches, span-based and pair-based. The span-based method rewards a document according to the text span of the overall query terms and doesn't consider the internal structure of those terms. The pair-based method represents a document's proximity score directly via pair-wise term distance and also doesn't take into account the individual term proximity structure. Another weakness of previous works lies in that the proximity factor is combined with probabilistic models in a very intuitive way. Most works just linearly combine a document's overall proximity score externally with relevance score computed by traditional ranking functions at the document level.

Language modeling has become a very promising direction for information retrieval because of its solid theoretical background as well as its empirical good performance. In this paper, we try to integrate term proximity into the unigram language modeling approach. We model the individual query term's proximate centrality as Dirichlet hyper-parameter that weights the corresponding term emission parameter of the multinomial document language model. Thus, we attain an integral ranking formula that effectively boosts the score contribution from terms when the term is proximate to other query terms. This term level incorporation of proximity is mathematically grounded. Further, it performs empirically better than previous intuitive combination of proximity at document level.

SIGIR'09, July 19–23, 2009, Boston, Massachusetts, USA.

The rest of this paper is organized as follows. We introduce the most related works in Section 2. In Section 3, we present the proximity integrated language model. Then,in Section 4, we study several different ways of measuring a query term's proximate centrality. We report the experimental result in Section 5 and finally in Section 6, we conclude our work.

2. RELATED WORKS

2.1 Dependency Modeling

Many efforts have been made to directly incorporate term dependency into probabilistic modeling against the unintuitive "bag of word" assumption. In early attempts, [6, 28] try to incorporate term dependence over Binary Independence Model (BIM). However, those models don't achieve stable improvement over BIM as expected and are rarely used in practice. The reason may lie in that independence assumption in BIM could actually be replaced by a weaker linked dependence assumption [5].

Recently, along with the booming of language modeling in information retrieval, several works are done to integrate term dependence into the language model. [25, 17] propose general language models that combine bigram language model with unigram language model and thus can consider ordered adjacent dependence. [26] introduces biterm language model that also takes into account dependency between unordered adjacent word pairs. [10, 19] put forward dependency language models that consider the relation between terms existed in a dependency tree. [16] advances an exponential language model that allows the consideration of dependency in every subset of query terms. All those works report improvement over the unigram independence language model. However, the main problem of those dependency models lies in that the parameter space becomes very large by direct dependency consideration. This will make the parameter estimation much more difficult to compute and sensitive to data sparse and noise. Those risks may counteract the relatively small benefits one could obtain from direct dependency modeling.

2.2 Phrase Indexing

Another stream of works tries to incorporate bigger unit than word such as phrase or loose phrase in text representation. In such a way, dependence between words can be captured indirectly. [8, 7] test the method of incorporating syntactic and statistical phrases in text indexing. It is shown that statistic phrases perform better than syntactic phrases. However, the improvement of using phrases is not consistent. For several collections, significant improvements in effectiveness are achieved while for other collections, marginal or negative improvements are attained. [18] re-examines the use of phrases for indexing and retrieval. In extracting statistical phrases, all pairs of non-function words that occur contiguously in a number of documents are regarded as phrases. And those phrases are viewed as indexing unit just like simple words. It concludes that the use of phrases do not have a major effect on precision at high ranks if a good basic ranking scheme is used.

As pointed out by [10], the reason for the ineffectiveness of phrase indexing may lie in the following two aspects. First, phrases are different in nature from words, so it may not be appropriate to apply the same weighting schemes for both of them. Second, phrases are likely to be systematically over-scored in the independent model.

2.3 Proximity Modeling

Proximity modeling can be thought of as another indirect way to capture term dependence. In some early works, [11, 14, 4] propose similar proximity models that measure a document's proximity score by the span and density of query terms. The shorter the text span that contains all the query terms, the more relevant the document. Also, the more span instances of query terms in a document, the more relevant the document. However, the above works don't incorporate the proximity measure with an effective probabilistic ranking model.

Recently, several works try to incorporate proximity factor into probabilistic ranking functions. [22, 3] extend BM25 ranking formula [24] with a term-proximity scoring part. The proximity part is a span-based measure that scores every query term pair occurring in a text segment covering all the query terms in analogous form as simple words. The ranking score of the document is the linear combination of the BM25 ranking score with the proximity score. The result shows improvement for top precision, with marginal impact on the average precision.

[27] investigates the span-based measures and also propose several pair-based proximity measures that model proximity in terms of the pair-wise local distance between terms. [27] tries to combine the proximity factor with both BM25 ranking model and the KL divergence language model [15]. The integration of proximity factor is also in an external score combination form as:

$$Rank(Q, D) = KL(q, d) + log(\gamma + exp(-\delta(q, d))) \qquad (1)$$

In which, $KL(q, d)$ is ranking score attained by KL, and $\delta(q, d)$ is a proximity distance measure of the document d with respect to query q. Although in such a simple way, [27] shows that by combining a properly proximity distance measure with the KL language model, comparative result could be attained compared to the more complicated dependence language models such as [16].

Overall, compared to direct dependency modeling and phrase indexing, proximity modeling seems to be an economic while effective way to go beyond the "bag of word" assumption in retrieval. Our work tries to improve the previous works in proximity modeling by the following aspects. We integrate the proximity factor into the unigram language modeling approach in a more systematic and internal way that is more effective than external linear score combination.

3. PROXIMITY LANGUAGE MODEL

3.1 The Unigram Language Model

First, we introduce the traditional multinomial language model for information retrieval. Consider a query q and a document d_l in a collection C. Let $V = \{w_1, w_2, ..., w_{|V|}\}$ denote the vocabulary set, then under the "bag of words" assumption, both the query and the document are represented as vectors of term counts:

$$q = (q_1, q_2, ..., q_{|V|})$$
$$d_l = (d_{l,1}, d_{l,2}, ..., d_{l,|V|})$$

where q_i and $d_{l,i}$ is the frequency of the i^{th} term in query q and document d_l respectively. Now, let θ_l=($\theta_{l,1}$,$\theta_{l,2}$,...,

$\theta_{l,|V|}$) be the parameters of a multinomial generation model for d_l, where $\theta_{l,i}$ means the probability of emission of term w_i in the vocabulary. Then, the probability of generating a particular query q or a document d_l could be given as:

$$p(q|\theta_l) = \frac{n_q!}{\prod_{i=1}^{|V|} q_i!} \prod_{i=1}^{|V|} (\theta_{l,i})^{q_i} \quad (2)$$

$$p(d_l|\theta_l) = \frac{n_l!}{\prod_{i=1}^{|V|} d_{l,i}!} \prod_{i=1}^{|V|} (\theta_{l,i})^{d_{l,i}} \quad (3)$$

where $n_q = \sum_{i=1}^{|V|} q_i$ and $n_l = \sum_{i=1}^{|V|} d_{l,i}$ is the total term count in q and d_l.

For each document d_l, the parameters of θ_l are estimated by viewing the document as the observed data. The relevance of d_l to a query q is then measured by the probability of generating q by the language model estimated from d_l. So, the key element is the estimation of the multinomial parameters in θ_l from a given document d_l. The simplest way is the maximum likelihood estimation.

$$\hat{\theta}_{l,i} = \frac{d_{l,i}}{n_l} \quad (4)$$

To account for rare words and alleviate the data sparseness problem, various smoothing methods are usually applied to the estimation, which commonly interpolate the document estimated language model with collection estimated language models [29].

3.2 Integration with Proximity Information

Now, we discuss how to integrate the proximity information of query terms in the document into the unigram language model. In forming a query, a user wants to use several terms to jointly express a specific concept. Term proximity information expresses the closeness of query terms appeared in a document. The more adjacent terms appear in a document, the more possible that those terms are topically related and functions together to express the concept underlying the user query. So, given two document d_a and d_b, supposing all others being equal while the query terms of q appears more proximate in d_a than in d_b, we believe that d_a should be more relevant to the query than d_b. In other words, if $\hat{\theta}_a$ and $\hat{\theta}_b$ represent the language model estimated from d_a and d_b respectively, we believe that the probability that q is generated from $\hat{\theta}_a$ should be higher than $\hat{\theta}_b$.

To achieve the above expectation, we view the proximity centrality score of a query term w_i as the weight on the term's emission probability $\theta_{l,i}$. That is, the estimates for different term's emission probability should be related and proportional to each other according to each term's proximity centrality score. Here, a query term's proximity centrality means a term's proximity relative to all other query terms in a document which is defined in Section 4. By Bayesian analysis. we could express our knowledge or belief on the uncertainty of the parameters by some prior distribution on it. The conjugate of the distribution where the parameters come from is usually exploited to express the prior belief. The natural conjugate of multinomial distribution is Dirichlet distribution.

Specifically, supposing that B is a proximity centrality computing model and $Prox_B(w_i)$ be the computed proximity centrality of term w_i, we use a Dirichlet prior [9] on θ_l with hyper-level parameters $u = (u_1, u_2, ..., u_{|V|})$

$$p(\theta_l|u) = Z_u \prod_{i=1}^{|V|} (\theta_{l,i})^{u_i - 1} \equiv Dir(\theta_l|u) \quad (5)$$

where $u_i = \lambda Prox_B(w_i)$, and $Z_u = \frac{\Gamma(\Sigma_{i=1}^{|V|})}{\prod_{i=1}^{|V|} \Gamma(u_i)}$ which doesn't depend on the parameter θ_l. Dirichlet is the distribution for probabilities. By using the Dirichlet prior, we express our belief that the estimated probability of the matching words in a document should be correlated in a way that reflects the proximity structure of the terms.

Then, given the document d_l and the proximity computing model, we could get the posterior estimation of θ_l as:

$$p(\theta_l|d, u) = \frac{p(d|\theta_l)p(\theta_l|u)}{p(d|u)} = Z_{u'} \prod_{i=1}^{|V|} (\theta_{l,i})^{d_{l,i}+u_i-1} \quad (6)$$

By the property of natural conjugate distributions, the above equation is also a Dirichlet distribution as $Dir\ (\theta_l|u + d)$ in which $u + d = (d_{l,1} + u_1, ..., d_{l,|V|} + u_{|V|})$. The prior distribution reflects our prior beliefs about the weight of θ_l, while the posterior distribution of θ_l reflects the updated beliefs about θ_l posterior to observing the frequency information of the data d_l. In other words, the posterior distribution is centered at a point that represents a compromise between the prior belief and the data.

Given the posterior distribution, the estimation of the word emission probability can then be noted as:

$$\theta_{l,i}^B = \int_{\theta_l} Dir(\theta_l|u + d)\theta_l d\theta_l = \frac{d_{l,i} + \lambda Prox_B(w_i)}{n_l + \sum_{i=1}^{|V|} \lambda Prox_B(w_i)} \quad (7)$$

In empirical Bayesian analysis, the hyper-parameters u in Dirichlet prior can still be estimated from the data although it conflicts with the intuitive meaning of "prior". Specifically, for each document d_l, a corresponding proximity centrality model B_l will be computed from d_l with respect to the given query according to specified measures introduced in Section 4. Such a B_l is used in equation (7) to compute the word emission probability. In this way, we can see that the proximity information in a document is integrated into the estimated unigram document language model.

In the above estimated document model, the proximity information could be seen as transformed to word count information which is the primary object that unigram language model has the ability to model. From another point of view, we could consider that the "bag of word" representation of document d_l is transformed to a pseudo "bag of word" document representation d_{B_l} given the document's proximity model B_l with respect to a query q. In d_{B_l}, the matching term's frequency is transformed from $d_{l,i}$ to $d_{l,i} + \lambda Prox_B(w_i)$, and the total document length is changed from n_l to $n_l + \sum_{i=1}^{|V|} \lambda Prox_B(w_i)$. Then, the problem of ranking d_l for a query q is changed to ranking d_{B_l}. On such ground, any "bag of word" based language model, e.g. the query likelihood model or the KL divergence model [15], could work on d_{B_l} and thus integrate the proximity information in the document in an internal way.

Now, we give the proximity integrated ranking function in the KL divergence language modeling framework. We further smooth the proximity integrated language model by a collection language model $p(\cdot|C)$ to account for unseen words in the document as in [29]. Specifically, the collection-

based Dirichlet prior ($\mu p(w_1|C)$, $\mu\ p(w_1|C)$, ..., $\mu p(w_i|C)$) is applied for smoothing. We can get the smoothed proximity integrated estimation:

$$\hat{\theta}^B_{l,i} = \frac{d_{l,i} + \lambda Prox_B(w_i) + \mu p(w_i|C)}{n_l + \sum_{i=1}^{|V|} \lambda Prox_B(w_i) + \mu} \quad (8)$$

Then, the ranking function could be stated as:

$$Rank(q, d_l) = -D(\hat{\theta}_q || \hat{\theta}^B_l) \quad (9)$$

By further deduction following standard smoothing scheme, it can be shown the above scoring formula is essentially:

$$Rank(q, d_l) = \sum_{i:d_{l,i}>0, q_i>0} p(w_i|\hat{\theta}_q) log \frac{p_s(w_i|d_l, u)}{\alpha_d p(w_i|C)} + log\alpha_d \quad (10)$$

where $p_s(\cdot|d_l, u)$ is the seen word probability in the document d_l with respect to the proximity model B_l. $\alpha_d p(w_i|C)$ is the probability assigned to the unseen words in d_l.

$$p_s(w_i|d_l, u) = \hat{\theta}^B_{l,i} \quad (11)$$

$$\alpha_d = \frac{\mu}{n_l + \sum_{i=1}^{|V|} \lambda Prox_B(w_i) + \mu} \quad (12)$$

In the above formation, note that the proximity factor mainly functions to adjust the parameters for seen matching terms with respect to a query in a document. It is conceptually very different from the collection based priors used for smoothing which is motivated to weight the unseen words in the document.

4. TERM PROXIMITY MEASURE

A key notion in the above proximity language model is a term's proximate centrality $Prox_B(w_i)$. It represents a term's importance in forming the overall proximity structure in a specific document relative to a given query. For non-query terms, we assume they have a constant score of zero. For a query term, its proximate centrality should be computed according to a proximity measure that reflects the term's closeness to other query term's in the same document. However,most previous woks in proximity modeling compute a document's overall proximity score for a query. There is no well-established proximity measure for computing the proximate centrality or proximity score for a specific individual term. In this section, we develop several measures that could give such term specific centrality score.

4.1 Measuring Proximity via Pair Distance

An intuitive idea to represent a term's proximity is by measuring its distance to other query terms in the document. A short distance to other terms means that the term is in a high proximate area and should be assigned a high proximity score. There are two key points to implement such an idea. The first is how to define a term's distance to other terms in a document. The second is how to devise a proper non-linear function to map such a distance to the proximity score of the term.

First, we define the distance between any pair of terms in a document. It could be measured through their occurring positions in the document. The main difficulty lies in that both terms may have many occurrences in the document. Let $Q = \{Q_1, Q_2, ..., Q_m\}$ be the set of different query terms in a query, $O_{Q_i} = \{o_{i_1}, o_{i_2}, ..., o_{i_n}\}$ be the set of word occurrence positions of the word Q_i in document D. We use $Dis(x, y; D)$ to denote the pairwise distance between any two terms or two term occurrences in D. Following [27], the pairwise term distance is represented as the distance between the closest occurrring positions of the two terms in the document.

$$Dis(Q_i, Q_j; D) = \begin{cases} |D|, \text{ if } |O_{Q_i}| = 0 \text{ or } |O_{Q_j}| = 0 \\ MIN\_DIS, \text{ otherwise} \end{cases} \quad (13)$$

$$MIN\_DIS = min_{o_{i_k} \in O_{Q_i}, o_{j_k} \in O_{Q_j}} \{Dis(o_{i_k}, o_{j_k}; D)\} \quad (14)$$

in which, $|D|$ is the length of document D and $|O_{Q_i}|$ is the number of occurrences of query term Q_i in D. Note that the pair distance measure is symmetric which means $Dis(Q_i, Q_j; D) = Dis(Q_j, Q_i; D)$.

Having the pair-wise distance measure in hand, now we define the function needed to transform the distance into pair-wise term proximity. The transformation function plays a very important role in setting the scale of proportional ratio between proximity score of different matching terms. The following exponential form is used and tested, of which *data* is the input distance score and *para* is the parameter to control the scale of the transformed score.

$$f = para^{-data} \quad (15)$$

Then, the pair-wise term proximity could be noted as:

$$Prox(Q_i, Q_j; D) = f(Dis(Q_i, Q_j; D)) \quad (16)$$

4.2 Computation of Term's Proximate Centrality

Based on the above devices, we are ready to define the term proximity centrality measures we need. We propose and test on the following term proximity centrality measures.

Term Proximity based on Minimum Distance

In this measure, the proximity score of a query term is obtained by applying the proximity transformation function to the term's minimum pair distance with other terms. Such a measure is noted as $P\_MinDist$:

$$P\_MinDist(Q_i) = f(min_{Q_j \neq Q_i, Q_j \in Q} \{Dis(Q_i, Q_j; D)\}) \quad (17)$$

where f is the nonlinear monotonic function defined in equation (15).

Term Proximity based on Average Distance

Instead of relying on the minimum distance, this measure uses the term's average pair distance with other terms to model the term's proximate status. It is noted as $P\_AveDist$:

$$P\_AveDist(Q_i) = f(\frac{1}{n-1} \sum_{Q_j \neq Q_i, Q_j \in Q} Dis(Q_i, Q_j; D)) \quad (18)$$

of which, n is the number of unique query terms appeared in D.

Term Proximity Summed over Pair Proximity

This measure models a term's proximity as the summation over all the pair-wise proximity the term involved. It is

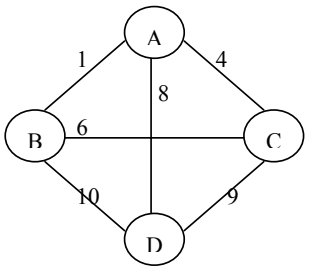

Figure 1: An Example of Term Distance Graph

noted as $P\_SumProx$:

$$P\_SumProx(Q_i) = \sum_{Q_j \neq Q_i, Q_j \in Q} f(Dis(Q_i, Q_j; D)) \quad (19)$$

$P\_SumProx$ first transforms each pair distance to the pair proximity score by non-linear function and then sum over all the pair-wise proximity scores.

We illustrate the features of the above term proximity measures using a simple example. Supposing there are 4 matching terms A, B, C and D, the pair-wise distances between them are illustrated in Fig.1. Now supposing that the proximity transformation function used is $f = 1.5^{-dist}$, Table 1 illustrates the proximity centrality score computed by the three different term proximity measures.

5. EXPERIMENTS

5.1 Data Set and Experimental Setup

We evaluate our model described in previous sections using the following TREC data sets: the popular ad hoc collection AP88 (Associated Press News, 1988), WSJ90-92(Wall Street Journal 1990-92) and WSJ87-92(Wall Street Journal, 1987-92) plus OHSUMED (MedLine Database Abstract, 1987-1991) collection [12] used in TREC-9 filtering track. The statistics of the collections are illustrated in Table 2. While the three ad hoc TREC collections and queries contain polymorphic documents and topics from different domains, the OHSUMED collection is more monomorphic which could be seen as documents from a speific technical domain (medicine). To evaluate the impact of different queries on similar collections, two different TREC topic set, TOPIC 251-300 and TOPIC 151-200 are used for the two overlapping WSJ document collections of different size. For TREC ad hoc topics, only the *title* field of the topic is used. For OHSUMED topics, only the *description* field is used. The OHSUMED query is more verbose than other TREC topics in nature. The relevance assessments for the OHSUMED collection were made by medical librarians and physicians based on the results of interactive searches which are graded as definitely or possibly relevant. We make no distinction between definitely and possibly relevant documents in our test.

Table 1: Term proximity score computed by different measures.

Measure	Prox(A)	Prox(B)	Prox(C)	Prox(D)
P_MinDist	1	1	0.07	0.13
P_AveDist	0.11	0.07	0.04	0.06
P_SumProx	1.17	1.10	0.11	0.23

Table 2: Data Set Statistics

Collection	#Doc.	Len.	Queries	#qrel.
AP88	79919	488	TOPIC251-300	1672
WSJ90-92	74520	514	TOPIC251-300	1064
WSJ87-92	173252	473	TOPIC151-200	3913
OHSUMED	348566	129	Ohsumed topic	3875

All the experiments in this paper are done using Lemur toolkit [20]. All the documents and queries in the above collections are tokenized with a naive tokenizer which views all symbols other than English characters as delimiters. Stemming is applied by using Porter's algorithm. No stop word removing is done at the index time. Instead, a very small stop word list [1] is used for eliminating the most frequent stop words at query time.

Both top precision and average precision are used to evaluate the experiment result. Specifically, we use evaluation metric Pr@5, Pr@10 and MAP which are precision at top 5 documents, precision at top 10 documents and mean average precision respectively.

5.2 Baseline Models

We compare the performance of the proximity integrated language model (noted as PLM) with the basic KL divergence langauge model (noted as LM) and the work for incorporating proximity with langauge model in [27] (noted as LLM). [27] combines the proximity score of the document with relevance score of the language model by external linear combination at the document level as shown in equation (1) in Section 2. In our implementation, the minimum pair distance measure is used for the document's proximity distance $\delta(Q, D)$ which achieves the best performance as shown in [27]. However, such a simple combination doesn't consider the scale and the weight of the proximity score and relevant score, and thus is sensitive to the surface form of the formula for the language model. For example, comparing the following two deduction forms for the KL langauge model:

$$Rank(q,d) = \sum_{i:q_i>0, d_i>0} \frac{q_i}{|q|} \cdot log(1 + \frac{d_i}{\mu \cdot p(w|C)}) + log\frac{\mu}{|d|+\mu} \quad (20)$$

$$Rank(q,d) = \sum_{i:q_i>0, d_i>0} q_i \cdot log(1 + \frac{d_i}{\mu \cdot p(w|C)}) + |q| \cdot log\frac{\mu}{|d|+\mu} \quad (21)$$

Although they are equivalent for ranking, it will be very different when combines directly with a proximity score because of the different scale of the score. We strictly following [27] by using equation (21) as the KL ranking formula in spite of that equation (20) is the more common one.

5.3 Parameter Setting

There are several parameters need to be set in those models. In LM, the most important parameter is the prior collection sample size μ. It is set to 2000 across all the experiments [29]. Such a parameter is also used in LLM and PLM without any further optimization.

In LLM, there is a parameter γ involved in the proximity model as shown in equation (1) in Section 2. We set it by optimizing the performance on the respective test collections

[1] http://www.ranks.nl/resources/stopwords.html

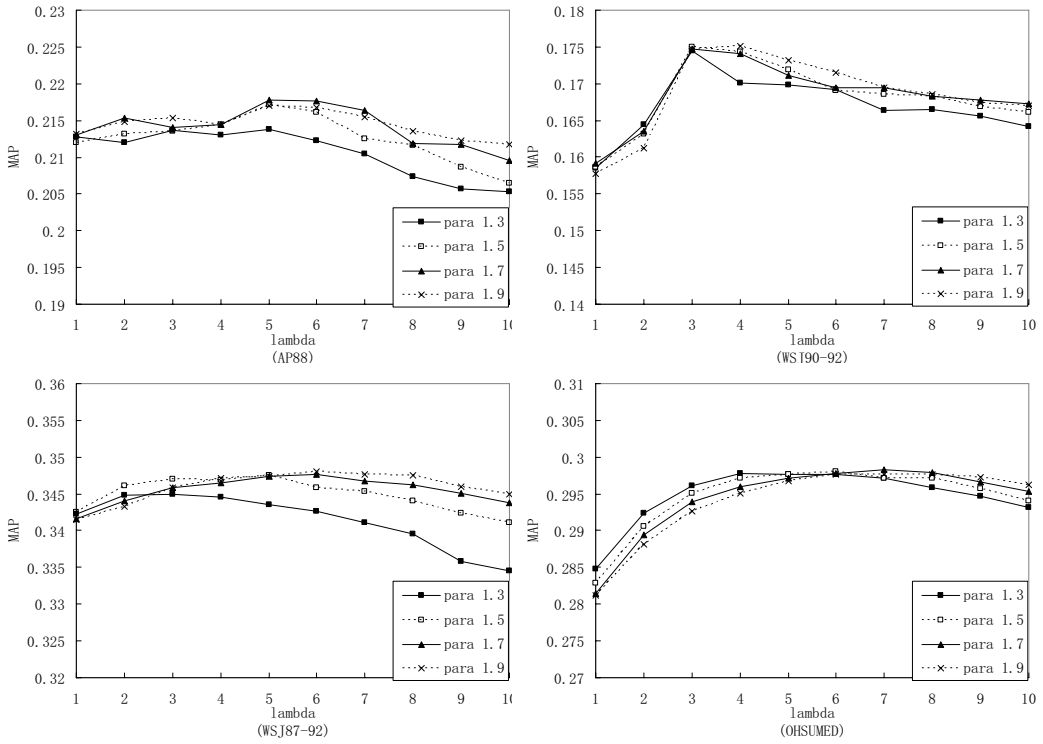

Figure 2: Parameter Sensitivity for PLM using P_MinDist as the centrality measure.

through searching the parameter space

$$\gamma: \ 0.1, 0.2, ..., 1.0$$

In PLM, the most two important parameters are λ and *para*. Parameter λ is the proximity argument for the proximity information as in equation (7) of Section 3. It controls the proportional weight of prior proximity factor relative to the observed word count information in a document. The other parameter *para* is the exponential weight of proximity transition function in equation (15) of Section 4. It controls the numeric scale of different terms' proximity score. In other words, it controls the proportional ratio of proximity score between different query terms in a document. Currently, we set those two parameters by maximizing the empirical performance on the collections through exhaustive searches in the following parameter space.

$$para: \ 1.1, 1.2, ..., 2.0$$
$$\lambda: \ 0.1, 1, 2, 3, ..., 10$$

We are investigating a more efficient approach such as leave-one-out and will include this in our future work.

In Figure 2, we report the parameter sensitivity of PLM on the test collections when using $P\_MinDist$ as the measure of term proximity centrality (Note, the optimal parameter setting for using $P\_AveDist$ and $P\_SumProx$ are a little different from $P\_MinDist$). We can see that except for the case of WSJ90-92, the best performance is achieved when the proximity argument λ is set to about 6. Also, The best λ value is relatively insensitive to different *para* value in the term proximity measure. However, the exponential weight *para* which controls the proportional ratio between different terms does have some influence on the ranking performance. For the three ad hoc collections, a relatively bigger value of *para* of about 1.7 is preferred. While for the very different OHSUMED collection, a smaller value is better.

5.4 Performance Comparison

Table 3 shows the best overall performance achievable by different ranking models. In the table, notation like PLM (P_MinDist) means PLM model which uses $P\_MinDist$ as the term proximity measure.

First, we compare the performance of PLM when applying different term proximity centrality measures. (*) marks the Wilcoxon significance at 0.05 compared with the basic LM. Overall, $P\_SumProx$ and $P\_MinDist$ performs much better than $P\_AveDist$. Also, $P\_SumProx$ performs a little better than $P\_MinDist$, but they are comparable. Please note that the performance reported in Table 3 is when stop words are eliminated from the query as explained in Section 5.1. Things will be a little different when no stop word removing is done. We report the performance in such a case in Section 5.5.

Next, we compare our proximity integrated model PLM with LM and the model that uses external linear score combination (LLM) as described in Section 5.2. LLM improves the basic language model in the three ad hoc collections but failed for OHSUMED collection which has more verbose topics. It actually does some harm to LM. PLM performs much better than the basic language model as well as LLM in terms of both top precision and average precision. PLM performs very well on verbose topics of OHSUMED.

Comparing with the baseline models, the main power of the proximity integrated language model comes from the following aspect. In external linear score combination approach, the combination of proximity score with the relevant score is done in a post-processing manner while in PLM the

Table 3: Performance Comparison of Different Ranking Models

Model/Measure	MAP	PR@5	PR@10	MAP	PR@5	PR@10
	AP88			WSJ90-92		
LM	0.2070	0.3120	0.2620	0.1534	0.2160	0.1860
LLM	0.2123	0.3280	0.2720	0.1589	0.2280	0.1940
PLM(P_MinDist)	0.2178	0.3440	0.2780	0.1751*	0.2320	0.1940
PLM(P_AveDist)	0.2167	0.3360	0.2740	0.1603	0.2320	0.1900
PLM(P_SumProx)	0.2203	0.3440	0.2840	0.1735*	0.2360*	0.2000*
	WSJ87-92			OHSUMED		
LM	0.3351	0.5440	0.4960	0.2704	0.4889	0.4698
LLM	0.3457	0.5640	0.5120	0.2651	0.4857	0.4587
PLM(P_MinDist)	0.3474	0.5680	0.5100	0.2983*	0.5365*	0.5095*
PLM(P_AveDist)	0.3446	0.5600	0.5060	0.2954*	0.5238*	0.5095*
PLM(P_SumProx)	0.3493	0.5720	0.5120	0.2984*	0.5397*	0.5154*

Table 4: Performance Comparison Considering Stop Words in the Query

Model/Measure	MAP	Pr@5	Pr@10
AP88			
LM	0.1537	0.2957	0.2348
LLM	0.1422	0.2609	0.2435
PLM(P_MinDist)	0.1575	0.2870	0.2391
PLM(P_AveDist)	0.1558	0.3030	0.2479
PLM(P_SumProx)	0.1607	0.2957	0.2565
WSJ90-92			
LM	0.1072	0.1652	0.1348
LLM	0.1017	0.1391	0.1217
PLM(P_MinDist)	0.1139	0.1652	0.1391
PLM(P_AveDist)	0.1080	0.1739	0.1348
PLM(P_SumProx)	0.1158	0.1665	0.1522

proximity factor is integrated in a pre-processing manner. For external score combination, the relevant score and proximity score of a document are computed separately for a document. The nature of the two resulted score is totally different and thus could only be combined in a very intuitive way. Most importantly, by using the proximity score at an external document level, the proximity information provided by different terms are mixed and the overall proximity information contained in a document is decreased. On the contrary, in PLM, the proximity information is combined with other document statistics such as term frequency in a distributed way. The proximity information provided by each query term makes its own effort to the matching weight.

5.5 The Influence of Stop Word in Query

As we all know, stop word removing does much help in IR. However, in many practical cases, it is desirable not to remove stop words at all. A good ranking function should also perform well when stop words are considered in the user query. Stop words may have a big influence for proximity modeling. This is because a stop word usually has many occurrences in a document, resulting in a great chance to be proximate with other words in the document. It may make the proximity mechanism at risk to loose its effect. So, for proximity modeling, it is very important to test whether the models could resist to the influence of stop words.

We extract all the queries from TOPIC251-300 that contain at least one word in the used stop word list. This results in totally 23 queries. Then, we test different ranking models on AP88 and WSJ90-92 collection for those queries. Note, it is meant that we don't use any stop word list in either indexing or retrieval phase in this test.

Table 4 shows the performance of various ranking models on the stop word containing queries. From it, we can see that LLM approach fails when stop words are considered in the query. It does harm to the performance of the basic language model on both collections. In such a case, PLM can still improve on the basic language model in some degree.

Comparing different term proximity measures in PLM, the improvement ratio of $P\_MinDist$ on LM drops in some degree. $P\_SumProx$ now performs much more better than it. The problem for $P\_MinDist$ lies in that if there is a stopword like term in the query, say "of", this word may occur very proximate with each content word in the query due to its high occurring frequency. Thus, it will make some query term have a false big proximity score in some cases. Overall, $P\_SumProx$ is a good choice for using in PLM, which performs well on improving the basic language model irrelevant of whether stop word removing is used.

6. CONCLUSIONS

In summary, we have developed a novel way to integrate proximity factor into the unigram language modeling for information retrieval, which views query terms' proximate centrality as Dirichlet hyper-parameters to weight the parameters of the multinomial document language models. This integration method has solid mathematical foundation and shows empirical better performance than previous works. Most importantly, besides simple keyword query, this model also performs well in verbose query and stop word containing query.

However, there is still much work to be done in the future. First, as we mentioned in Section 5.3, we should develop a more efficient way to set the parameters of the PLM model instead of using exhaustive search. Second, in developing PLM, we don't make any effort to normalize different query terms' proximity centrality score. It will be very helpful to study how to normalize it to a given scale or even to probability, and see the effect on ranking result as well as parameter tuning. Finally, it will be very interesting to study how to combine the proximity information with other document information such as prior document strength to further improve the effectiveness of language modeling.

7. ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers for their valuable comments and helpful suggestions.

8. REFERENCES

[1] J. Bai, Y. Chang, H. Cui, Z. Zheng, G. Sun, and X. Li. Investigation of partial query proximity in web search. 2008.

[2] D. Beeferman, A. Berger, and J. Lafferty. A model of lexical attraction and repulsion. *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 373–380, 1997.

[3] S. Buttcher and C. Clarke. Efficiency vs. Effectiveness in Terabyte-Scale Information Retrieval. *Proceedings of the 14th Text Retrieval Conference (Gaithersburg, USA, November 2005)*.

[4] C. Clarke, G. Cormack, and E. Tudhope. Relevance ranking for one to three term queries. *Information Processing and Management*, 36(2):291–311, 2000.

[5] W. Cooper. Some Inconsistencies and Misidentified Modeling Assumptions in Probabilistic Information Retrieval. *ACM Transactions on Information Systems*, 13(1):100–111, 1995.

[6] W. Croft. Boolean queries and term dependencies in probabilistic retrieval models. *Journal of the American Society for Information Science*, 37(2):71–77, 1986.

[7] W. Croft, H. Turtle, and D. Lewis. The use of phrases and structured queries in information retrieval. *Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 32–45, 1991.

[8] J. Fagan. Experiments in Automatic Phrase Indexing For Document Retrieval: A Comparison of Syntactic and Non-Syntactic Methods. 1987.

[9] T. Ferguson. A Bayesian analysis of some nonparametric problems. *Ann. Statist*, 1(2):209–230, 1973.

[10] J. Gao, J. Nie, G. Wu, and G. Cao. Dependence language model for information retrieval. *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 170–177, 2004.

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

[12] W. Hersh, C. Buckley, T. Leone, and D. Hickam. OHSUMED: an interactive retrieval evaluation and new large test collection for research. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–201. Springer-Verlag New York, Inc. New York, NY, USA, 1994.

[13] K. Jones, S. Walker, and S. Robertson. *A Probabilistic Model of Information Retrieval: Development and Status*. University of Cambridge, Computer Laboratory, 1998.

[14] C. LA Clark and G. Cormack. Shortest-Substring Retrieval and Ranking. *ACM Transactions on Information Systems*, 18(1):44–78, 2000.

[15] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 111–119, 2001.

[16] D. Metzler and W. Croft. A Markov random field model for term dependencies. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 472–479, 2005.

[17] D. Miller, T. Leek, and R. Schwartz. A hidden Markov model information retrieval system. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 214–221, 1999.

[18] M. Mitra, C. Buckley, A. Singhal, and C. Cardie. An analysis of statistical and syntactic phrases. *Proceedings of RIAO-97, 5th International Conference "Recherche d'Information Assistee par Ordinateur"*, pages 200–214, 1997.

[19] R. Nallapati and J. Allan. Capturing term dependencies using a language model based on sentence trees. *Proceedings of the eleventh international conference on Information and knowledge management*, pages 383–390, 2002.

[20] P. Ogilvie and J. Callan. Experiments Using the Lemur Toolkit. *NIST Special Publication SP*, pages 103–108, 2002.

[21] J. Ponte and W. Croft. *A language modeling approach to information retrieval*. ACM New York, NY, USA, 1998.

[22] Y. Rasolofo and J. Savoy. Term Proximity Scoring for Keyword-Based Retrieval Systems. *Lecture Notes in Computer Science*, pages 207–218, 2003.

[23] S. Robertson, S. Jones, et al. Relevance Weighting of Search Terms. *Journal of the American Society for Information Science*, 27(3):129–46, 1976.

[24] S. Robertson, S. Walker, and M. Beaulieu. Okapi at TREC-7: automatic ad hoc, filtering, VLC and interactive track. *NIST Special Publication SP*, pages 253–264, 1999.

[25] F. Song and W. Croft. A general language model for information retrieval. *Proceedings of the eighth international conference on Information and knowledge management*, pages 316–321, 1999.

[26] M. Srikanth and R. Srihari. Biterm language models for document retrieval. *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 425–426, 2002.

[27] T. Tao and C. Zhai. An exploration of proximity measures in information retrieval. *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 295–302, 2007.

[28] C. Yu, C. Buckley, K. Lam, and G. Salton. A Generalized Term Dependence Model in Information Retrieval. *Information technology: research and development*, 2(4):129–154, 1983.

[29] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS)*, 22(2):179–214, 2004.