Positional Relevance Model for Pseudo-Relevance Feedback

Yuanhua Lv
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
ylv2@uiuc.edu

ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
czhai@cs.uiuc.edu

ABSTRACT

Pseudo-relevance feedback is an effective technique for improving retrieval results. Traditional feedback algorithms use a whole feedback document as a unit to extract words for query expansion, which is not optimal as a document may cover several different topics and thus contain much irrelevant information. In this paper, we study how to effectively select from feedback documents those words that are focused on the query topic based on positions of terms in feedback documents. We propose a positional relevance model (PRM) to address this problem in a unified probabilistic way. The proposed PRM is an extension of the relevance model to exploit term positions and proximity so as to assign more weights to words closer to query words based on the intuition that words closer to query words are more likely to be related to the query topic. We develop two methods to estimate PRM based on different sampling processes. Experiment results on two large retrieval datasets show that the proposed PRM is effective and robust for pseudo-relevance feedback, significantly outperforming the relevance model in both document-based feedback and passage-based feedback.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Retrieval models, Relevance feedback, Query formulation

General Terms

Algorithms

Keywords

Positional relevance model, pseudo-relevance feedback, positional language model, proximity, passage-based feedback, query expansion

1. INTRODUCTION

Pseudo-relevance feedback (or blind feedback) is an important general technique for improving retrieval accuracy [26, 24, 27, 3, 25, 16, 32]. The basic idea of pseudo-relevance feedback is to assume that a small number of top-ranked documents in the initial retrieval results are relevant and select from these documents related terms to the query to improve the query representation through query expansion, which generally leads to improvement of retrieval performance.

Most existing feedback algorithms (e.g., [26, 24, 27, 3, 25, 16, 32]) use a whole feedback document as a unit for selecting expansion terms, which, however, is non-optimal when the content of a document is incoherent (i.e., covering several different topics) and thus may contain much irrelevant information as often happens in Web search. The existence of multiple topics and irrelevant information would lead to a noisy feedback model as potentially harmful terms from non-relevant topics may be picked up to include in the feedback model. As a result, the use of pseudo feedback may not improve or even decrease the retrieval performance. Thus a critical challenge in improving all feedback methods is to effectively select from feedback documents those terms that are most likely relevant to the query topic.

In this paper, we solve this challenge by exploiting the position and proximity information of terms as cues to assess if a term is related to the query topic. Since topically related content is usually grouped together in text documents, terms closer to the occurrences of query words are, in general, more likely relevant to the query topic, thus a good feedback model should intuitively place higher weights on such terms.

Based on this intuition, we propose a novel positional relevance model (PRM) to incorporate the cues of term positions and term proximity in a probabilistic feedback model based on statistical language modeling. The key idea is to extend the relevance model [16] to aggregate the associations between a term and query words at the position level via the positional language model (PLM) [19]. An important advantage of estimating a relevance model based on PLM is that it can model the "relevant positions" in a feedback document with probabilistic models so as to assign more weights to terms at more relevant positions in a principled way, thus leading naturally to selection of expansion terms more likely relevant to the query topic.

Since PRM estimates a relevance model at the level of term positions, it incorporates individual term positions *directly* into a probabilistic model. This is in contrast with virtually all the existing pseudo feedback techniques which have only made use of term statistics at the document level [26, 24, 27, 3, 25, 16, 32, 18], or at the best, at the level of passages [2, 30, 17, 21] without distinguishing every different position.

Analogously to the two methods proposed for estimating the relevance model [16], we also derive two methods for estimating PRM, leading to two different ways to aggregate term information based on positions. We evaluate the proposed PRM on two large TREC datasets. Experimental results demonstrate that PRM is effective in exploiting term proximity for pseudo feedback and significantly outperforms the relevance model in both document-based feedback and passage-based feedback.

2. RELATED WORK

2.1 Pseudo-Relevance Feedback

Pseudo-relevance feedback has been shown to be effective with various retrieval models [26, 24, 27, 3, 25, 16, 32, 18].

In the vector space model, feedback is usually done by using the Rocchio algorithm, which forms a new query vector by maximizing its similarity to pseudo-relevant documents [26]. The feedback method in classical probabilistic models is to select expansion terms primarily based on Robertson/Sparck-Jones weight [24].

Several query expansion techniques have been developed in the language modeling framework, including, e.g., the mixture-model feedback method [32] and the relevance model [16]. The basic idea is to use feedback documents to estimate a better query language model. Both the mixture model and relevance model have been shown to be very effective, but the relevance model appears to be more robust [18].

In the mixture-model feedback [32], the words in feedback documents are assumed to be drawn from two models: (1) background model and (2) topic model. The mixture-model feedback finds the topic model that best describes the feedback documents by separating the topic model from the background model. The topic model is then interpolated with the original query model to form the expanded query.

Much like mixture-model feedback, the relevance model also estimates an improved query language model. Given a query Q, a relevance model is a multinomial distribution $P(w|Q)$ that encodes the likelihood of each term w given the query as evidence. To estimate the relevance model, the authors first compute the joint probability of observing a word together with the query words in each feedback document and then aggregate the evidence by summing over all the documents. It essentially uses the query likelihood $P(Q|D)$ as the weight for document D and takes an average of the probability of word w given by each document language model.

All these pseudo feedback algorithms use a whole feedback document as a unit, and thus term position and proximity evidences are largely ignored. Our work is an extension of the relevance model to estimate a feedback model based on individual term positions.

2.2 Passage Feedback

There have been several studies to exploit passage-level evidence of documents for feedback, e.g., [2, 30, 17], which can potentially address the heterogeneous topical structure of documents to some degree. However, these approaches usually take a traditional feedback model as a *black box* to handle sub-document units as if they were regular documents. For example, Liu and Croft's work [17] estimates a relevance model based on the best matching passage of each feedback document, where fixed-length arbitrary passages that resemble overlapped windows but with an arbitrary starting point [12] can often be used due to its effectiveness and efficiency [12, 17]. A limitation of this approach is that term positions are not directly incorporated into the feedback model. As we will show later in the paper, the proposed PRM outperforms such a passage feedback approach.

Some other approaches, e.g., [31, 4], make use of visual cues or eye tracker to improve passage feedback for web search: on the server side of a search engine, documents can be decomposed into topically different components via visual cues [31], while on the client side of users, gaze-based attention feedback [4] can go down to the sub-document level by exploiting evidence about which document parts the user looks at. However, such approaches face the same problems as general passage feedback without being able to model each individual position.

In fact, these passage-based or sub-document level feedback models are orthogonal to the proposed PRM in the sense that PRM can be applied to passages to model proximity inside a passages in the same way as it can be applied to whole documents. Moreover, the underlying positional language model [19], which can capture passage-level evidence in a *soft* way in model estimation, has been shown to work better than imposing a "hard" boundary of passages.

Recently, Metzler and Croft's work on Latent Concept Expansion [21] also indirectly captures term position and proximity evidence through the use of appropriate passages. Their work provides a more general model which is complementary with our ideas in that we can use PRM as an effective feature defined on their graph, so our PRM scores can then be combined with other features explored in [21] to further improve its performance.

2.3 Term Proximity Heuristic in IR

The term proximity heuristic, which rewards a document where the matched query terms occur close to each other, has been previously studied in [13, 14, 11, 9, 23, 20, 5, 6, 28, 19, 10, 34]. Keen's work [13, 14] is among the earliest efforts, in which, a "NEAR" operator was introduced to address proximity in Boolean retrieval model. The shortest interval containing a match set was first used as a measure of proximity in [9, 11]. Recent work has attempted to heuristically incorporate proximity into an existing retrieval model (often through score combinations) [22, 23, 5, 6, 28]. For example, a variety of proximity measures were proposed and explored in [28]. Another work [10] used a learning approach to combine various proximity measures to obtain an effective proximity-based retrieval function. Recently in our previous work [19], we proposed a positional language model (PLM) for information retrieval, which not only captured term proximity information but also covered passage retrieval in a unified language modeling approach. However, in all these studies, term proximity has been solely used for ranking documents in response to a given query rather than improving pseudo feedback.

There has been relatively little work done in the area of formally modeling term proximity heuristic in the context of pseudo feedback. However, there have been several attempts to simply combine term proximity with other feedback heuristics to select good expansion terms. In [29], several distance functions were evaluated for selecting query expansion terms from windows or passages surrounding query term occurrences; however, no improvement was observed as compared to existing feedback methods. Cao et al. [7] used a supervised method to classify whether an individual expansion term is good or not, in which term proximity is one of their features. Their method only loosely combined term proximity with traditional feedback heuristics; in contrast, we incorporate term position and proximity into a probabilistic feedback model with more meaningful parameters.

3. POSITIONAL RELEVANCE MODEL

In this section, we describe the proposed positional relevance model (PRM) which incorporates term position information into the estimation of feedback models so that we can naturally reward terms close to query terms in the feedback documents and avoid including irrelevant terms in the feedback model.

The proposed PRM can be regarded as an extension to the relevance model (RM) [16]. We thus first give a brief introduction to the relevance model.

3.1 Relevance Models

As a pseudo feedback method, the relevance model [16] has proven to be not only effective, but also robust in a recent study [18]. The basic idea is to use the query likelihood score of a feedback document as the weight and estimate a query language model (for feedback) based on weighted aggregation of term counts in the feedback documents.

Formally, let $Q = \{q_1, q_2, \cdots, q_m\}$ be a query and Θ represent the set of smoothed document models for the pseudo feedback documents. One of the most robust variants of the relevance model (RM1) [18] is computed as follows [16]:

$$P(w|Q) \propto \sum_{\theta_D \in \Theta} P(w|\theta_D) P(\theta_D) \prod_{i=1}^{m} P(q_i|\theta_D) \tag{1}$$

where $p(\theta_D)$ is a prior on documents and is often assumed to be uniform without any additional prior knowledge about document D. Thus, the estimated relevance model is essentially a weighted combination of individual feedback document model with the query likelihood score of a document as its weight.

After the relevance model is estimated, the estimated $P(w|Q)$ can then be interpolated with the original query model θ_Q to improve performance [1].

$$P(w|\theta'_Q) = (1-\alpha) P(w|\theta_Q) + \alpha P(w|Q) \tag{2}$$

where α is a parameter to control the amount of feedback. In the rest of the paper, we will refer to this instantiation of relevance model as **RM3**.

3.2 Positional Relevance Models

In the relevance model, the count of a term is computed over an entire feedback document. The main idea of the proposed positional relevance model (PRM) is to further distinguish different positions of a term and discount the occurrences of a term at positions that are far away from a query term in a feedback document.

Similarly to RM, a PRM is also a multinomial distribution $P(w|Q)$ that attempts to capture the probability that term w is seen in a relevant document. However, PRM goes beyond RM to estimate the conditional probability $P(w|Q)$ in terms of the joint probability of observing w with the query

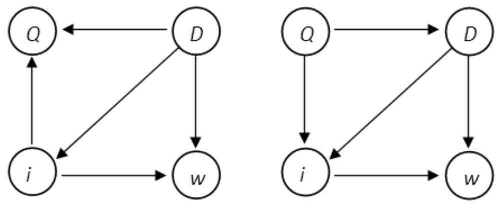


Figure 1: Dependence networks for two methods, i.e., method 1 (left) and method 2 (right), of estimating positional relevance models.

Q at every position in every feedback document. Formally,

$$P(w|Q) = \frac{P(w,Q)}{P(Q)} \propto P(w,Q) = \sum_{D \in \mathcal{F}} \sum_{i=1}^{|D|} P(w,Q,D,i) \tag{3}$$

where i indicates a position in document D, and F is the set of feedback documents (assumed to be relevant).

The challenge now lies in estimating the joint probability $P(w,Q,D,i)$. Inspired by the two estimation methods proposed in [16] for estimating relevance models, we derive two methods similarly for estimating $P(w,Q,D,i)$. The first method assumes that w is sampled in the same way as Q, while the second method assumes that w and Q are sampled using two different mechanisms.

3.2.1 Method 1: i.i.d. sampling

In this method, we first compute the joint probability of observing a word together with the query words at each position and then aggregate the evidence by summing over all the possible positions. Specifically, we factor the joint probability $P(w,Q,D,i)$ for each pseudo-relevant document D as follows:

$$P(w,Q,D,i) = P(D)P(i|D)P(w,Q|D,i) \tag{4}$$

Intuitively, we have assumed a generative model in which we would first pick a document according to $P(D)$, then choose a position i in document D with probability $P(i|D)$, and finally generate word w and query Q conditioned on D and i, with probability $P(w,Q|D,i)$.

$P(D)$ can be interpreted as a document prior and set to a uniform distribution with no prior knowledge about document D. While it is possible to estimate $P(i|D)$ based on document structures, here we assume that every position is equally likely, i.e., $P(i|D) = \frac{1}{|D|}$. Improving the estimation of $p(D)$ and $P(i|D)$ would be an interesting future work. An illustration of the dependencies between the variables involved in the derivation is shown on Figure 1 (left side).

After making these assumptions and a further assumption that the generation of word w and that of query Q are independent, we have

$$P(w,Q,D,i) \propto \frac{P(w,Q|D,i)}{|D|} = \frac{P(Q|D,i)P(w|D,i)}{|D|} \tag{5}$$

Plugging Equation 5 into Equation 3, we obtain the following estimate of the PRM:

$$P(w|Q) \propto P(w,Q) \propto \sum_{D \in \mathcal{F}} \sum_{i=1}^{|D|} \frac{P(Q|D,i)P(w|D,i)}{|D|} \tag{6}$$

In the above equation, $P(w|D,i)$ is the probability of sampling word w at position i in document D. To improve the efficiency of PRM, we simplify $P(w|D,i)$ as:

$$P(w|D,i) = \begin{cases} 1.0 & \text{if } w \text{ occurs at position } i \text{ in } D \\ 0.0 & \text{otherwise} \end{cases} \qquad (7)$$

The term $P(Q|D,i)$ in Equation 6 is the key component in estimating the positional relevance model. It is the query likelihood at position i of document D, and we will discuss how to estimate it based on the positional language model [19] in Section 3.2.3. Additionally, there is a third term $|D|$ in the equation, which penalizes long documents to prevent them from dominating the feedback model (long documents naturally have more positions).

Thus, Equation 6 essentially combines all terms in feedback documents by assigning different weights to each term: (1) $P(Q|D,i)$ serves as a relevance-based weight for each position in each document so that a position with many query terms nearby would have a higher weight. Thus as an intra-document weight, $P(Q|D,i)$ can measure the relative weights of positions within a document: a position closer to query words would more likely generate the query, and as a result a term that occurs at this position would naturally receive a higher weight. (2) $|D|$ comes into the formula because of the assumption about uniform distribution over all the positions in a document and can be interpreted as an inter-document weight: it penalizes a long document which is reasonable since a longer document by nature has more positions and more occurrences of terms to contribute.

3.2.2 Method 2: conditional sampling

In this method, we consider the following different way to decompose the joint probability distribution:

$$P(w,Q,D,i) = P(Q)P(D|Q)P(i|Q,D)P(w|D,i) \qquad (8)$$

The assumed generative model is as follows. We first pick a query according to some prior $P(Q)$. We then generate a document D with probability $P(D|Q)$. Finally, we select a position i in D with probability $P(i|Q,D)$ and generate word w according to $P(w|D,i)$. An illustration of this sampling process is given on the right side of Figure 1.

For the purpose of estimating $P(w|Q)$, we can clearly ignore the term $P(Q)$ as it is a query-specific constant. Using Bayes Rule and assuming both $P(D)$ and $P(i|D)$ to be uniform (as we have assumed in the first estimation method), we have

$$P(D|Q) = \frac{P(Q|D)P(D)}{\sum_{D\in F} P(Q|D)P(D)} = \frac{P(Q|D)}{\sum_{D\in F} P(Q|D)} \qquad (9)$$

$$P(i|D,Q) = \frac{P(Q|D,i)P(i|D)}{\sum_{i=1}^{|D|} P(Q|D,i)P(i|D)} = \frac{P(Q|D,i)}{\sum_{i=1}^{|D|} P(Q|D,i)} \qquad (10)$$

Plugging Equations 8, 9, and 10 into Equation 3, we obtain the following estimate of PRM:

$$P(w|Q) \propto \sum_{D\in\mathcal{F}} \sum_{i=1}^{|D|} \frac{P(Q|D)}{\sum_{D\in F} P(Q|D)} \frac{P(Q|D,i)}{\sum_{i=1}^{|D|} P(Q|D,i)} P(w|D,i) \qquad (11)$$

where $P(Q|D)$ is the query likelihood score of document D, which can be computed using either the positional language models [19] or the standard document language model. In our experiments, we use the latter, i.e., $P(Q|D) = \prod_{j=1}^{m} P(q_j|D)$.

As in the first estimation method, we compute $P(w|D,i)$ using Equation 7.

Similarly to the first estimation method, this second estimate of PRM also essentially combines all terms in feedback documents by assigning different weights to each term: the first weighting term in Equation 11 is seen to be the normalized query likelihood score of the document, which assigns more weights to documents that are more likely to be relevant, while the second weighting term is the normalized query likelihood of each *positional* language model, which assigns more weights to terms that are closer to query words.

Compared to the first estimation method, the document length normalizer $|D|$ is missing, but a comparable effect is now achieved by normalizing the query likelihood of each positional language model $P(Q|D,i)$. Indeed, the effect of intra-document weighting and inter-document weighting can now be seen even more clearly, i.e., the normalized $P(Q|D)$ can be interpreted as the inter-document weight favoring a document matching the query well, while the normalized $P(Q|D,i)$ clearly achieves intra-document weighting to place more weight on terms closer to query terms in document D.

3.2.3 More Estimation Details

This section provides the final estimation details for our positional relevance model (Equation 6 and 11), i.e., how to estimate $P(Q|D,i)$. We adapt the positional language model [19] to do that.

The key idea of PLM is to estimate a language model for each position of a document. Specifically, we let each word at each position of a document propagate the evidence of its occurrence to all other positions in the document so that positions closer to the word would get more share of the evidence than those far away. The PLM at each position can then be estimated based on all the propagated counts of all the words to the position as if all the words had appeared actually at the position with discounted counts. This new family of language models is intended to capture the content of the document at a position level, which is roughly like a "soft passage" centered at this position but can potentially cover all the words in the document with less weight on words far away from the position.

Formally, the PLM at position i of document D can be estimated as:

$$P(w|D,i) = \frac{c'(w,i)}{\sum_{w'\in V} c'(w',i)} \qquad (12)$$

where $c'(w,i)$ is the total propagated count of term w at position i from the occurrences of w in all the positions. Following [19], we estimate $c'(w,i)$ using the Gaussian kernel function:

$$c'(w,i) = \sum_{j=1}^{|D|} c(w,j) \exp\left[\frac{-(i-j)^2}{2\sigma^2}\right] \qquad (13)$$

where i and j are absolute positions of the corresponding terms in the document, and $|D|$ is the length of the document; $c(w,j)$ is the *real* count of term w at position j. With the approximation method proposed in [19], the following estimation of $P(w|D,i)$ is obtained:

$$P(w|D,i) = \frac{c'(w,i)}{\sqrt{2\pi\sigma^2}\cdot\left[\Phi\left(\frac{|D|-i}{\sigma}\right) - \Phi\left(\frac{1-i}{\sigma}\right)\right]} \qquad (14)$$

where $\Phi(\cdot)$ is the cumulative normal distribution and the denominator is essentially the length of the "soft" passage centered at position i.

However, there is one issue with the above estimation: the length of "soft" passages around the *boundaries* of a document would be smaller than that in the *middle* of the document; as a result, boundary positions tend to unfairly receive more weights. This may not raise problems in PLMs for retrieval [19], but it is a more serious concern for PRM, where the relative weights of terms are more important. So we decide to use a fixed length for all "soft" passages in feedback documents to estimate their corresponding positional language models as follows:

$$P(w|D,i) = \frac{c'(w,i)}{\sqrt{2\pi\sigma^2}} \tag{15}$$

This strategy has shown to be better than the original implementation in [19] for estimating PRM.

The distribution $P(\cdot|D,i)$ needs to be smoothed. Now that all "soft" passages have equal length, we use Jelinek-Mercer smoothing method to smooth PLM, which is shown to work as well as the Dirichlet prior smoothing method and is relatively insensitive to the setting of σ in our experiments.

$$P_\lambda(w|D,i) = (1-\lambda)P(w|D,i) + \lambda P(w|C) \tag{16}$$

where $\lambda \in [0,1]$ is a smoothing parameter and $p(w|C)$ is the collection language model. Now we can compute the positional query likelihood score $P(Q|D,i)$ for position i.

$$P(Q|D,i) = \prod_{j=1}^{m} P_\lambda(q_j|D,i) \tag{17}$$

Plugging Equation 17 into Equations 6 and 11, we would be able to compute the two estimation methods directly. Interestingly, if we set $\lambda = 1$ or $\sigma = \infty$, Method 2 will degenerate to the general relevance model (see Equation 1).

The computation of positional query likelihood is the most time-consuming part in estimating PRM. Fortunately, there is no serious efficiency concern even with an unoptimized implementation. The reason is because we only need to traverse each position of a document twice: during the first pass, the positions of query terms are recorded; in the second, we compute a positional query likelihood for each position directly based on the position information of query terms collected in the first pass. Therefore, the efficiency is comparable to the estimation of the relevance model.

Finally, the estimated positional relevance model $P(w|Q)$ will also be interpolated with the original query model θ_Q using Equation 2 to improve performance with a similar parameter α to that used in the mixture-model feedback [32] and RM3 [1].

4. EXPERIMENTS

4.1 Experimental Setup

We used two standard TREC datasets in our study: Terabyte (i.e., the Gov2 collection) and ClueWeb09 Category B. They represent two very large web text collections in English. Queries were taken from the title field of the TREC topics. We used the Lemur toolkit (version 4.10) and Indri search engine (version 2.10) [1] to implement our algorithms. For both datasets, the preprocessing of documents and queries included stemming with the Porter stemmer and stopwords removing using a total of 418 stopwords from the standard InQuery stoplist. Table 1 shows some basic statistics about the datasets.

	Terabyte05	Terabyte06	ClueWeb09 Cat. B
queries	751-800	801-850	20001-21000
#qry(with qrel)	50	49	358
#documents	25, 205, 179		50, 220, 423
mean(dl)	931		875

Table 1: Document set characteristic

We evaluated seven methods. (1) The basic retrieval model is the KL-divergence retrieval model [15], and we chose the Dirichlet smoothing method [33] for smoothing document language models, where the smoothing parameter μ was set empirically to 1500. This method was labeled as "NoFB". (2) The baseline pseudo feedback method is the relevance model "RM3" described in Section 3.1 [1], which is one of the most effective and robust pseudo feedback methods under language modeling framework [18]. (3) Another baseline pseudo feedback method is a standard passage-based feedback model, labeled as "RM3-p", which estimates the RM3 relevance model based on the best matching passage of each feedback document [17]. (4) We have two variations of PRM, i.e., "PRM1" and "PRM2", which are based on the two estimation methods described in Section 3.2, respectively. (5) In addition, we also used PRM1 and PRM2 for passage feedback in a way as RM3-p does. Specifically, we first computed a PLM for each position of the document, and then we estimate a PRM based on a passage of size 2σ centered at the position with the maximum positional query likelihood score (see Equation 17). These two runs are labeled as "PRM1-p" and "PRM2-p" respectively.

There are several parameters in these pseudo feedback algorithms. We fixed the number of feedback documents to 20 and the number of terms in feedback model to 30. Other parameters, including the feedback interpolation coefficient α, the two additional parameters σ and λ in PRM, the passage size, and the passage smoothing parameter in RM3-p, were all tuned on Terabyte05 dataset.

We used Terabyte06 and ClueWeb09 for testing. The top-ranked 1000 documents for all runs were compared in terms of their mean average precisions (MAP) (for Terabyte06) or eMAP [8] (for ClueWeb09). In addition, other performance measures, such as Pr@10, Pr@30 and Pr@100 for Terabyte06 and eP@10, eP@30 and eP@100 for ClueWeb09, were also considered in our evaluation.

4.2 Feedback Effect

We first examine the overall retrieval precision of the pseudo feedback models for document-based feedback. The results are summarized in Table 2, where the best result for each row is highlighted. As we see, both PRM1 and PRM2 significantly outperform the basic KL-divergence retrieval model in terms of MAP. In addition, PRM1 and PRM2 are also significantly better than RM3 across data sets. For example, the relative improvements of PRM1 over NoFB are 9.0% on Terabyte06 and 13.5% on ClueWeb09 in terms of average precision, which are much larger than the corresponding improvements achieved by RM3 (only 2.8% and 5.9% respectively). RM3 improves Pr@10 over NoFB in neither dataset; however both PRM1 and PRM2 often improve Pr@10, though not significantly. Besides, comparing PRM1

[1]http://www.lemurproject.org/

Collection	Metric	NoFB	RM3	PRM1	PRM2
Terabyte06	MAP	0.3047	0.3131	**0.3322**$^{*+}$	0.3319^{*+}
	Pr@10	0.5367	0.5041	0.5306	**0.5490**$^{+}$
	Pr@30	0.4653	0.4660	**0.4884**$^{+}$	0.4871^{+}
	Pr@100	0.3547	0.3576	0.3671^{*+}	**0.3741**$^{*+}$
ClueWeb09	eMAP	0.0713	0.0755	**0.0809**$^{*+}$	0.0786^{*+}
	eP@10	0.2371	0.2307	**0.2418**$^{+}$	0.2377^{+}
	eP@30	0.2433	0.2486	**0.2536**$^{*+}$	0.2525^{*+}
	eP@100	0.2216	0.2283	**0.2356**$^{*+}$	0.2325^{*+}

Table 2: Comparison of different pseudo feedback models for document-based feedback. '*' and '+' mean the corresponding improvements over NoFB and RM3 are significant respectively.

Collection	RM3-p	PRM1	PRM2	PRM1-p	PRM2-p
Terabyte06	0.3077	0.3322*	0.3319*	0.3331*	0.3290*
ClueWeb09	0.0781	0.0809*	0.0786	0.0800*	0.0798*

Table 3: MAP/eMAP comparison of passage-based feedback methods. '*' means the corresponding improvement over RM3-p is significant.

and PRM2, we find that PRM1 is slightly more effective than PRM2.

We are also interested in evaluating if a heuristic passage-based feedback (i.e., RM3-p) can work as well as PRM, since both PRM1 and PRM2 essentially can be regarded as achieving a soft effect of passage feedback. Moreover, we can also use PRM1 and PRM2 for "hard" passage feedback in a way as RM3-p does, which leads to PRM1-p and PRM2-p respectively. So we further compare the average precision of PRM1, PRM2, PRM1-p, PRM2-p, and RM3-p in Table 3. From the table, it is clear that PRM1, PRM2, PRM1-p and PRM2-p all outperform RM3-p significantly in most cases, suggesting that our model does not only have sound statistical foundation but also works effectively. In addition, we also observe that RM3-p behaves quite differently in two datasets: it beats RM3 on ClueWeb09 but loses to RM3 on Terabyte06. However, all the four variations of PRM perform better than RM3 consistently. Finally, it is also interesting to see that PRM1 and PRM2 work similarly to PRM1-p and PRM2-p respectively, which may mean that PRM1 and PRM2 have already achieved successfully an effect of passage-based feedback by assigning weights to different positions, so it does not bring too much additional benefit to apply PRM to passages explicitly.

Next we examine the robustness to the parameter setting in PRM on the Terabyte06 collection.

4.3 Robustness Analysis

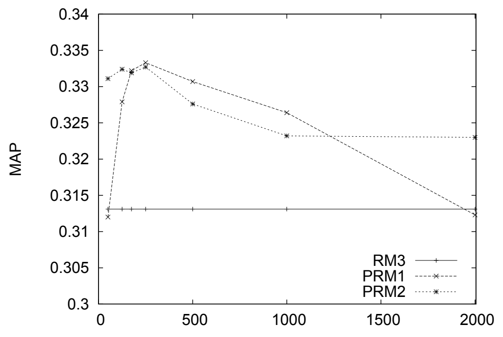
In PRM1 and PRM2, there is a parameter σ inherited from the positional language model to control the propagation range, which would influence the effect of term position and term proximity. Specifically, if we increase σ to infinity, the effect of term position and proximity will be disabled. However, if we decrease this parameter to a finite value, term position and proximity will play an important role in PRM. We fix other parameters to their default values as trained on Terabyte05 and focus on understanding how σ affects the retrieval performance of PRM1 and PRM2. From Figure 2 (left), we can see that, as long as σ is in the range of [100, 1000], both PRM1 and PRM2 outperform RM3 clearly. Indeed, by setting σ around 200, we can often obtain the optimal performance for both PRM1 and PRM2. This result confirms the observation in previous work [19]. In addition, comparing PRM1 and PRM2, PRM2 seems to be less sensitive to σ.

Next, the positional language model is smoothed using Jelinek-Mercer method to estimate PRM. The smoothing is controlled by a parameter λ. When $\lambda = 0$, we are using the pure positional language model, while if $\lambda = 1$, we completely ignore the position and proximity evidence so that every position will receive the same weight. Again, we fix other parameters and show in Figure 2 (right) how the average precision changes under different λ. The experiment results indicate that when λ is set to around 0.1, both PRM1 and PRM2 achieve their optimal performance. However, PRM1 and PRM2 always outperform RM3 with $\lambda < 1$. Comparing PRM1 and PRM2, we see again that PRM2 seems to be more robust.

Recall that we interpolate the feedback model with the original query model. The interpolation is controlled by a coefficient α. When $\alpha = 0$, we are only using the original query model (i.e., no feedback), while if $\alpha = 1.0$, we completely ignore the original query model and use only the estimated feedback model. We fix other parameters and show in Figure 3 (left) how the average precision changes according to the value of α. We can see that both PRM1 and PRM2 are clearly better than RM3 with different α values. And the optimal α for all the methods seems to be in a range around 0.5. Besides, it is also interesting to observe that the pure feedback model results ($\alpha = 1.0$) of PRM1 and PRM2 are much better than that of RM3, suggesting that the positional relevance model can lead to a more accurate query model. Finally, comparing PRM1 and PRM2, the former seems to be slightly more effective.

We further compare the robustness of different methods w.r.t. the number of feedback documents. We change the number of feedback documents from 1 to 200. The MAP results are shown in Figure 3 (middle). We notice that PRM1 and PRM2 are more robust to the number of feedback documents as compared to RM3. It is also interesting to see there is almost no performance decrease of PRM1 and PRM2 even when we set the parameter to 200, suggesting that the proposed positional relevance model works better in tolerating noisy information. Moreover, with only 1 feedback document, PRM1 and PRM2 have already been able to outperform RM3, no matter how many feedback documents RM3 uses, which may indicate that our methods can identify good feedback terms more accurately by assigning position-dependent weights.

Additionally, we also compare the sensitivity of different methods to the number of expansion terms in Figure 3 (right). We vary the number of terms from 5 to 100, and observe that both PRM1 and PRM2 can achieve a very effective performance with only 10 expansion terms, while



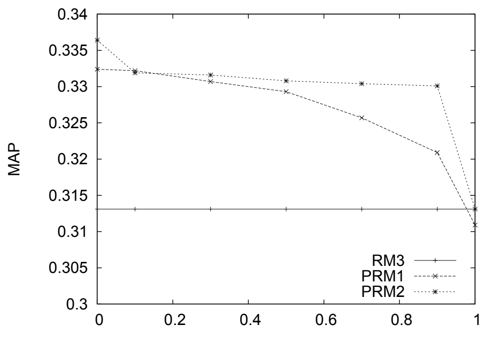
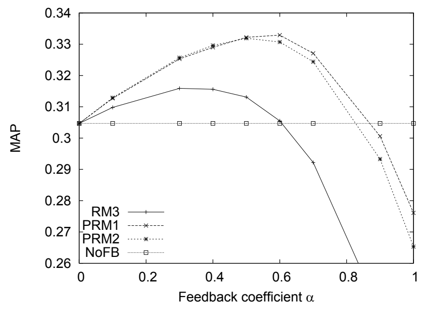
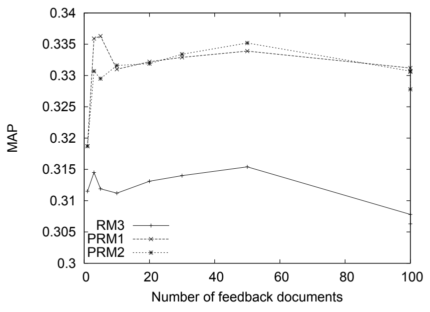


Figure 2: Sensitivity to the propagation range σ (left) and the smoothing parameter λ (right) of PRM.





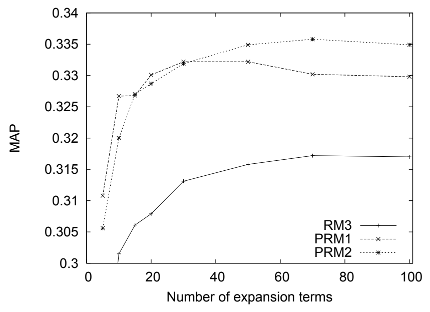
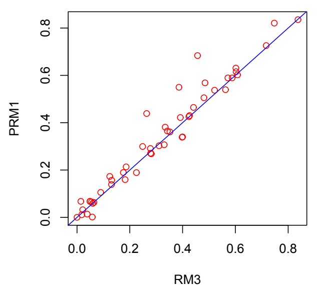


Figure 3: Sensitivity to the feedback interpolation coefficient α (left), the number of feedback documents (middle), and the number of expansion terms (right) of different pseudo feedback methods



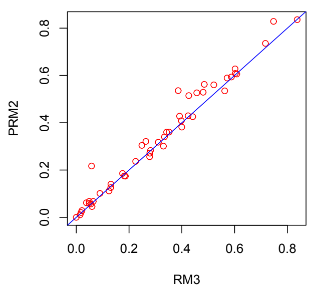


Figure 4: MAP Plot of PRM1 (left) and PRM2 (right) as compared to RM3 on Terabyte06

RM3 needs 70 terms, but even so, its performance is still not as good as our methods with 10 terms. This would be another advantage of our methods since fewer expansion terms mean higher efficiency, which is very important for retrieval systems.

To further see the robustness of our methods on individual queries, we plot the MAP of PRM1 versus RM3 and PRM2 versus RM3 on Terabyte06 in Figure 4. It is interesting that the proposed methods, particularly PRM2, are quite robust; they improve most of the queries clearly with only a small number of queries decreased slightly.

5. CONCLUSIONS

We proposed a novel positional relevance model (PRM) for pseudo-relevance feedback. The PRM exploits term *position* and *proximity* evidence to assign more weights to words closer to query words based on the intuition that words closer to query words are more likely to be consistent with the query topic. Specifically, PRM generalizes the relevance model to aggregate the associations between a word and query words at the position-level in a probabilistic way. We also developed two methods to estimate the PRM based on different generative models.

Experiment results on two large web data sets show that the proposed PRM is quite effective and robust and performs significantly better than the state of the art relevance model in both document-based feedback and passage-based feedback. Compared to the relevance model, the proposed models are also less sensitive to the setting of various parameters, such as feedback coefficient, number of feedback documents, and number of expansion terms. Comparing the two estimation methods of PRM, the first method (PRM1) appears to be more effective, while the second (PRM2) tends to be more robust. Both methods achieve its optimal retrieval performance when setting the σ value in a range around 200 and λ to around 0.1.

There are many interesting future research directions to explore. One of the most interesting directions is to further study whether setting a term-specific and/or query-specific σ can further improve performance. Another interesting direction is to study how to optimize σ automatically based on the layout of web pages. Improving the estimate of other components in PRM (e.g., the probability of choosing a position in a document) would also be interesting.

6. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their useful comments. We also thank Wan Chen for helping improve the English in this paper. This material is based upon work supported by the National Science Foundation under Grant

Numbers IIS-0347933, IIS-0713581, IIS-0713571, and CNS-0834709.

7. REFERENCES

[1] Nasreen Abdul-Jaleel, James Allan, W. Bruce Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Donald Metzler, Mark D. Smucker, Trevor Strohman, Howard Turtle, and Courtney Wade. Umass at trec 2004: Novelty and hard. In *TREC '04*, 2004.

[2] James Allan. Relevance feedback with too much data. In *SIGIR '95*, pages 337–343, 1995.

[3] Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. Automatic query expansion using smart: Trec 3. In *TREC '94*, pages 69–80, 1994.

[4] Georg Buscher, Andreas Dengel, and Ludger van Elst. Query expansion using gaze-based feedback on the subdocument level. In *SIGIR '08*, pages 387–394, 2008.

[5] Stefan Buttcher and Charles L. A. Clarke. Efficiency vs. effectiveness in terabyte-scale information retrieval. In *TREC '05*, 2005.

[6] Stefan Buttcher, Charles L. A. Clarke, and Brad Lushman. Term proximity scoring for ad-hoc retrieval on very large text collections. In *SIGIR '06*, pages 621–622, 2006.

[7] Guihong Cao, Jian-Yun Nie, Jianfeng Gao, and Stephen Robertson. Selecting good expansion terms for pseudo-relevance feedback. In *SIGIR*, pages 243–250, 2008.

[8] Ben Carterette, James Allan, and Ramesh Sitaraman. Minimal test collections for retrieval evaluation. In *SIGIR '06*, pages 268–275, 2006.

[9] Charles L. A. Clarke, Gordon V. Cormack, and Forbes J. Burkowski. Shortest substring ranking (multitext experiments for trec-4). In *TREC '95*, pages 295–304, 1995.

[10] Ronan Cummins and Colm O'Riordan. Learning in a pairwise term-term proximity framework for information retrieval. In *SIGIR '09*, pages 251–258, 2009.

[11] David Hawking and Paul B. Thistlewaite. Proximity operators - so near and yet so far. In *TREC '95*, pages 500–236, 1995.

[12] Marcin Kaszkiel and Justin Zobel. Effective ranking with arbitrary passages. *Journal of the American Society for Information Science and Technology*, 52(4):344–364, 2001.

[13] E. Michael Keen. The use of term position devices in ranked output experiments. *The Journal of Documentation*, 47(1):1–22, 1991.

[14] E. Michael Keen. Some aspects of proximity searching in text retrieval systems. *Journal of Information Science*, 18(2):89–98, 1992.

[15] John D. Lafferty and Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. In *SIGIR '01*, pages 111–119, 2001.

[16] Victor Lavrenko and W. Bruce Croft. Relevance-based language models. In *SIGIR '01*, pages 120–127, 2001.

[17] Xiaoyong Liu and W. Bruce Croft. Passage retrieval based on language models. In *CIKM '02*, pages 375–382, 2002.

[18] Yuanhua Lv and ChengXiang Zhai. A comparative study of methods for estimating query language models with pseudo feedback. In *CIKM '09*, pages 1895–1898, 2009.

[19] Yuanhua Lv and ChengXiang Zhai. Positional language models for information retrieval. In *SIGIR '09*, pages 299–306, 2009.

[20] Donald Metzler and W. Bruce Croft. A markov random field model for term dependencies. In *SIGIR '05*, pages 472–479, 2005.

[21] Donald Metzler and W. Bruce Croft. Latent concept expansion using markov random fields. In *SIGIR '07*, pages 311–318, 2007.

[22] Christof Monz. Minimal span weighting retrieval for question answering. In Rob Gaizauskas, Mark Greenwood, and Mark Hepple, editors, *SIGIR Workshop on Information Retrieval for Question Answering*, pages 23–30, 2004.

[23] Yves Rasolofo and Jacques Savoy. Term proximity scoring for keyword-based retrieval systems. In *ECIR '03*, pages 207–218, 2003.

[24] Stephen E. Robertson and Karen Sparck Jones. Relevance weighting of search terms. *Journal of the American Society of Information Science*, 27(3):129–146, 1976.

[25] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at trec-3. In *TREC '94*, pages 109–126, 1994.

[26] J. J. Rocchio. Relevance feedback in information retrieval. In *In The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–323. Prentice-Hall Inc., 1971.

[27] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society of Information Science*, 41(4):288–297, 1990.

[28] Tao Tao and ChengXiang Zhai. An exploration of proximity measures in information retrieval. In *SIGIR '07*, pages 295–302, 2007.

[29] Olga Vechtomova and Ying Wang. A study of the effect of term proximity on query expansion. *Journal of Information Science*, 32(4):324–333, August 2006.

[30] Jinxi Xu and W. Bruce Croft. Query expansion using local and global document analysis. In *SIGIR '96*, pages 4–11, 1996.

[31] Shipeng Yu, Deng Cai, Ji-Rong Wen, and Wei-Ying Ma. Improving pseudo-relevance feedback in web information retrieval using web page segmentation. In *WWW '03*, pages 11–18, 2003.

[32] ChengXiang Zhai and John D. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *CIKM '01*, pages 403–410, 2001.

[33] ChengXiang Zhai and John D. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR '01*, pages 334–342, 2001.

[34] Jinglei Zhao and Yeogirl Yun. A proximity language model for information retrieval. In *SIGIR '09*, pages 291–298, 2009.