A Study of Term Proximity and Document Weighting Normalization in Pseudo Relevance Feedback – UIUC at TREC 2009 Million Query Track

Yuanhua Lv[1], Jing He[2], V.G.Vinod Vydiswaran[1], Kavita Ganesan[1], ChengXiang Zhai[1]
Department of Computer Science, University of Illinois at Urbana-Champaign [1]
School of Electronic Engineering and Computer Science, Peking University [2]

ABSTRACT

In this paper, we report our experiments in the TREC 2009 Million Query Track. Our first line of study is on proximity-based feedback, in which we propose a *positional relevance model* (PRM) to exploit term proximity evidence so as to assign more weights to expansion words that are closer to query words in feedback documents. The second line of study is to improve the weighting of feedback documents in the relevance model by using a regression-based method to approximate the probability of relevance (and thus the name RegRM). In the third line of study, we test a supervised approach for query classification. Besides, we also evaluate a selective pseudo feedback strategy which stops pseudo feedback for precision-oriented queries and only uses it for recall-oriented ones.

The proposed PRM has shown clear improvements over the relevance model for pseudo feedback, suggesting that capturing the term proximity heuristic appropriately could lead to a better feedback model. RegRM performs as well as relevance model, but no noticeable improvement is observed. Unfortunately, the proposed query classification methods appear to not work well. The results also show that the proposed selective pseudo feedback may not work well, since precision-oriented queries can also benefit from pseudo feedback, though not as much as recall-oriented queries.

1. INTRODUCTION

We took the opportunity of participating in TREC 2009 Million Query Track to study some novel pseudo relevance feedback algorithms on the large Web data set ClueWeb09.

Most existing feedback algorithms, e.g., [7, 6, 8, 3, 9, 4], used a whole feedback document as a unit, without distinguishing the *position* of each word for term weighting. However, it is often the case that only some part of a document is useful for query expansion. This problem is especially critical for Web search, because the content of a web page is often incoherent and covers several different topics. It motivates us to select expansion terms through assigning appropriate weights to expansion terms by incorporating term position and proximity information into the feedback model, based on the intuition that words closer to query words in feedback documents often appear to be more relevant to the query.

Besides term weighting, *document weighting* also plays an important role in pseudo feedback as shown in our recent work [4], in which we found one variation of relevance model (i.e., RM3) [3, 1] most robust due to its use of the query likelihood as a weight for each pseudo-relevant document. However, our pilot experiments also indicate that the query likelihood score does not reflect the *probability of relevance* well, suggesting there is still room to improve the estimation of the relevance model with better document weighting.

To address these two issues, we extend the relevance model [3] to incorporate term position and proximity information to improve term weighting, and to normalize the query likelihood score to enhance document weighting. Specifically, in the first line of study, we propose a *positional relevance model* (PRM), which extends our previous work [5] to exploit the evidence of term proximity in a probabilistic model for term weighting so as to assign more weights to words closer to query words in feedback documents; In our second line of study, which is an extension to our recent work [4], we develop a regression-based method to normalize document weighting (i.e., query likelihoods) for the relevance model to make it better reflect the probability of relevance, and thus the name regularized relevance model (RegRM).

Besides the two directions discussed above, we also explore query classification in our experiments. We design a number of features which are combined using a logistic classifier for query classification. Based on the results of query classification, we also evaluate a selective pseudo feedback strategy which stops pseudo feedback for precision-oriented queries and only uses it for recall-oriented ones.

The results show that the PRM improves over the relevance model clearly and consistently for pseudo feedback, suggesting that the term proximity heuristic is useful for feedback. RegRM performs as well as relevance model, but no noticeable improvement is observed. Unfortunately, the proposed query classification methods appear to not work well. The results also show that selective pseudo feedback may not work well if the decision only depends on whether a query is recall-oriented or precision-oriented.

2. POSITIONAL RELEVANCE MODEL

2.1 Positional Language Model

The key idea of positional language model (PLM) [5] is to estimate a language model for each position of a document. Specifically, we let each word at each position of a document to 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 the propagated counts of all the words to the position as if they 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 each position, 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:

$$p(w|D,i) = \frac{c'(w,i)}{\sum_{w' \in V} c'(w',i)} \quad (1)$$

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 [5], $c'(w,i)$ is estimated using the Gaussian kernel function:

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

where i and j are absolute positions of the corresponding terms in the document, and N is the length of the document; $c(w,j)$ is the *actual* count of term w at position j.

The PLM $P(\cdot|D,i)$ needs to be smoothed. We use Jelinek-Mercer smoothing method to smooth the PLM, which is shown to work as well as Dirichlet prior smoothing and is relatively insensitive to the setting of smoothing parameter in our experiments.

$$p_\lambda(w|D,i) = (1-\lambda)p(w|D,i) + \lambda p(w|B) \quad (3)$$

where λ is a smoothing parameter and $p(w|B)$ is the background language model estimated using the whole collection.

2.2 Positional Relevance Model

Positional relevance model (PRM) extends PLM for feedback. PRM estimates the conditional probability $P(w|Q)$ in terms of the joint probability of observing w with the query Q at every position in every feedback document. Formally,

$$\begin{aligned} P(w|Q) &\propto P(w,Q) \\ &= \sum_{D \in \mathcal{F}} \sum_{i=1}^{|D|} P(w,Q,D,i) \end{aligned} \quad (4)$$

where i indicates a position of document D, and F is the set of pseudo-relevant documents. We propose two methods for estimating this joint probability.

First method: given each feedback document D, we choose a position i with a probability $P(i|D)$, and then generate word w and query Q conditioned on D and i, i.e., $P(w,Q|D,i)$. Mathematically, we get the following derivation of the joint probability:

$$P(w,Q,D,i) = P(D)P(i|D)P(w,Q|D,i) \quad (5)$$

where, $p(D)$ is a general prior on documents and is often assumed to be uniform. We also do not put any prior on positions without extra knowledge, and thus we can compute $P(i|D)$ as $\frac{1}{|D|}$. We will further assume that the query Q and the word w in feedback documents are sampled identically and independently from a unigram distribution $P(\cdot|D,i)$. We get the following final estimate for the joint probability of w and Q:

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

Second method: we fix a value of Q according to some prior $P(Q)$, and then assume the following generating process: a document D is first drawn conditioned on Q, and a position i of document D is then drawn dependent on D and Q, followed by the generation of word w. Mathematically, we get the following derivation of the joint probability:

$$P(w,Q,D,i) \propto P(D|Q)P(i|Q,D)P(w|Q,D,i) \quad (7)$$

In the above equation, it is natural to assume that the sampling of word w is only conditioned on the position i and the document D and is independent of any query, and thus we obtain: $P(w|Q,D,i) = P(w|D,i)$. We also adopt uniform prior for both documents and positions. After rewriting $P(D|Q)$ and $P(i|Q,D)$ by Bayes rule, we get the following final estimation for the joint probability of w and Q:

$$P(w,Q) \propto \sum_{D \in \mathcal{F}} \sum_{i=1}^{|D|} \frac{P(Q|D)}{\sum_{D \in F} P(Q|D)} \frac{P(Q|D,i)}{\sum_{i=1}^{|D|} P(Q|D,i)} P(w|D,i) \quad (8)$$

In the above two estimation methods, for the efficiency reason, we simplify $P(w|D,i)$ as:

$$P(w|D,i) = \begin{cases} 1.0 & \text{if } D[i] == w \\ 0.0 & \text{otherwise} \end{cases} \quad (9)$$

Note that $P(Q|D,i)$ is the key component in equations 6 and 8. It is the query likelihood of the positional language model at position i of document D.

3. DOCUMENT WEIGHTING NORMALIZATION WITH REGRESSION

In our recent work [4], we have found that the relevance model [3] works more robust than some other feedback models mainly due to its use of the query likelihood as document weighting. However, our pilot experiments show that the query likelihood scores do not reflect the probability of relevance well.

The distribution of relevance can be approximated as follows:

$$p(i|\theta_{rel}) = \frac{\sum_j \delta(Q_j,i)}{\sum_i \sum_j \delta(Q_j,i)} \quad (10)$$

where Q_j is a training query. $\delta(Q_j,i)$ equals to 1 if the ith retrieved document of Q_j is relevant; 0 otherwise. Here the approximated distribution of relevance is essentially the distribution of relevant documents in different ranking positions. Similarly, we can also obtain the distribution of query likelihood scores.

$$p(i|\theta_{QL}) = \frac{1}{M} \sum_j^M \left[\frac{QL(Q_j,i)}{\sum_{k=1}^{|F|} QL(Q_j,k)}\right] \quad (11)$$

where $QL(Q_j,i)$ is the query likelihood score of the ith result document of Q_j, and M is the number of training queries.

Through visualization or power transformation, we can find an approximate "logarithm" relation between the two distributions, that is,

$$\begin{aligned} p(i|\theta_{rel}) &\approx a \cdot \log p(i|\theta_{QL}) + b \\ &\propto \log p(i|\theta_{QL}) + c \end{aligned} \quad (12)$$

Run	Features	All(%)	P(%)	R(%)
uiuc09Adpt	1,2,3,4,5,8,9,10	57.6	43.4	68.2
uiuc09ReqQL	1,2,3,4,5,6,9,10,11	54.9	53.5	55.9
uiuc09KL	1,2,3,4,5,6,7,8,9,10	56.7	31.0	76.1

Table 2: Features used in the runs

where $c = \frac{b}{a}$. We thus hypothesize that in order to get a better approximation of the distribution of relevance, we can normalize the query likelihood by learning parameter c through a simple linear regression method. And then, we can use the normalized query likelihood to replace the raw query likelihood in the relevance model. We call this method regularized relevance model (RegRM) in this paper.

4. QUERY CLASSIFICATION

For the query classification task, we classify a query as either recall-oriented or precision-oriented. We use logistic classification to learn the classification models, which are trained on the TREC 04 Web track dataset. Although in this training set, queries are classified as homepage query (HP), named page query (NP), or topic distillation query (TD), both homepage queries and named page queries can be regarded as precision-oriented queries since their information needs can often be satisfied by one Web page, while topic distillation queries can be considered as recall-oriented queries.

We train two binary classifiers: the first one is to classify navigational queries from informational queries, while the second one is to classify precision-oriented informational queries from recall-oriented informational queries. During the classification process, the first classifier is used to classify all queries, after that, those queries that are judged to be informational queries are further classified into precision-oriented or recall-oriented categories by using the second classifier.

There are three types of features used in our work: pre-retrieval, post-retrieval and search engine generated features. Pre-retrieval features can be computed easily from the query string and document collection statistics; post-retrieval features can only be computed after a ranked list of documents has been retrieved; search engine generated features are computed based on the top ranked documents from Google for the corresponding query. Table 1 lists all the features we used.

We submitted three runs, each of which used a different set of features to train the corresponding classification model. Table 2 shows feature used in these three runs, as well as the classification performance of each run. The overall accuracy, the accuracy of precision-oriented queries, and the accuracy of recall-oriented queries are listed in columns 3, 4 and 5, respectively.

It shows that our methods appear to not work well. For the three runs we submitted, neither term dependency (with ordered/unordered terms) features nor search engine generated features help improve the performance. There are several possible reasons. First, the features may have not been normalized well. Second, the characteristics of precision-oriented queries may not be captured sufficiently (e.g., in our training data, we only take homepage and named page queries that have single-page answers as precision-oriented queries, but actually a precision-oriented query may also be answered by a combination of several pages.), leading to a worse performance on precision-oriented queries.

5. EXPERIMENTS

5.1 Experimental Setup

We use the Lemur toolkit (version 4.10) and Indri search engine (version 2.10) [1] in our experiments. For all data sets, the preprocessing of documents and queries involves stemming with the Porter stemmer and stopwords removing using a total of 418 stopwords from the standard InQuery stoplist.

Our baseline retrieval model is the KL-divergence retrieval model [2], and we choose the popular Dirichlet smoothing method [10] for smoothing document language models, where the smoothing parameter μ is set empirically to $1,500$ in our experiments.

We also fix the number of feedback documents to 20 and the number of terms in feedback model to 30, while the feedback interpolation coefficient for all feedback algorithms is set to 0.4 for WT2g (with query topics 401-450) and 0.5 for ClueWeb09 and Terabyte06 (with query topics 801-850), respectively.

There are two additional parameters σ and λ in the positional relevance model, we fix them to 100 and 0.3 respectively, which have been shown to perform robustly in both our preliminary and official experiments. Besides, the linear regression model used for normalizing query likelihood scores is trained on Terabyte06.

5.2 Experiment Results

We submitted five runs for the Million Query Track to test the proposed pseudo feedback algorithms, including uiuc09KL, uiuc09RegQL, uiuc09MProx, uiuc09GProx, and uiuc09Adpt. For all these runs, we just experimented with the first $1,000$ queries. These runs are described in Table 3. Some runs are also involved in the query classification task as shown in Table 2.

In Table 4, we compare the performance of our official runs. Although the evaluation results of mtc and statMAPs are often inconsistent with each other, we can still see that in general, uiuc09GProx > uiuc09MProx > uiuc09RegQL > uiuc09Adpt. However, the results of statMAPs show surprisingly that all the pseudo feedback algorithms lose to the baseline KL-divergence retrieval model, even though the results of mtc show otherwise.

Overall, both variations of PRM worked well: they perform comparably or better than the relevance model (i.e., RM3). Another run, uiuc09RegQL, performs similarly to the relevance model, but no noticeable improvement has been observed; one possible reason is that queries are different, and we should consider their characteristics to adaptively normalize document weighting.

Unfortunately, uiuc09Adpt fails to improve the performance. To examine if the poor performance is due to the low precision of the query classification, we generated another adaptive retrieval run Adpt+. This new run dynamically chooses uiuc09GProx/uiuc09KL for recall/precision-oriented queries based on the ground truth of query classification, but when a query is not covered by the ground truth, Adpt+ is backed off to uiuc09Adpt. The results in Table 4 show that Adpt+ is slightly better than uiuc09Adpt,

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

pre-retrieval	1. maximal/average normalized df
	2. maximal/average normalized cf
	3. point mutual information between pairs of terms
	4. query length
	5. query likelihood
	6. maximal/average normalized df for n ordered terms from the query
	7. maximal/average normalized df for n any terms from the query
post-retrieval	8. normalized maximal score difference between d_i and d_{i+1}
	9. common document number in top three documents by content retrieval and multiple fields(anchor, url, content) retrieval
	10. is it a homepage document retrieved in top three ranks by multiple field retrieval
search engine	11. Is the top retrieved document from Google

Table 1: Features for Query Classification

Official runs	uiuc09KL	Baseline KL-divergence without feedback
	uiuc09RegQL	Document weighting normalization with regression
	uiuc09MProx	Positional relevance model (i.i.d. sampling) + soft passage
	uiuc09GProx	Positional relevance model (conditional sampling) + soft passage
	uiuc09Adpt	Adaptively choose uiuc09GProx/uiuc09KL for recall/precision-oriented queries
Additional runs	RM3	Baseline relevance model [3, 1]
	PRM1	Positional relevance model (i.i.d. sampling) + fixed-length passage
	PRM2	Positional relevance model (conditional sampling) + fixed-length passage

Table 3: Description of Runs

suggesting that recall-oriented queries indeed benefit a little more from feedback than precision-oriented queries. To further examine if it can improve performance by stopping pseudo feedback for precision-oriented queries, we generated an "ideal" run Adpt++, which does not execute pseudo feedback for precision-oriented queries according to the ground truth but does pseudo feedback for all other queries (including recall-oriented queries and unknown-type queries). It turns out that Adpt++ still loses to uiuc09GProx, which means that precision-oriented queries could also benefit from pseudo feedback, though not as much as recall-oriented queries. Overall, selective pseudo feedback may not work well if the decision only depends on whether a query is recall-oriented or precision-oriented.

In uiuc09MProx and uiuc09GProx, we employ our original implementation of PLM in [5] directly. However, one concern with that implementation is that the length of "soft" passages around the *boundary* of a document would be smaller than that in the *middle* of the document; as a result, the boundary positions tend to receive more weights. This may not raise problems in PLMs for retrieval [5], but it could hurt PRM, where the relative weights of terms are more important. So, we decided to generate two additional runs PRM1 and PRM2 (see Table 3), in which we use a fixed-length (the length of the very middle "soft" passage) for all "soft" passages in the document to estimate their corresponding positional language models.

The results of the additional runs are also presented in Table 4. It is interesting to see that PRM1 and PRM2 are much better than uiuc09MProx and uiuc09GProx respectively. Overall, PRM1 performs the best on the ClueWeb09 data set, and it outperforms the regular relevance model by more than 5% in most of the cases; PRM2 also performs better than the regular relevance model. It suggests that capturing term proximity appropriately could lead to a better feedback model.

In addition, we also compare PRM1 and PRM2 with the regular relevance feedback and the baseline KL-divergence retrieval model on two other TREC data sets, i.e., WT2g and Terabyte06, using traditional evaluation criteria, such as MAP and precision at top-k documents. The results of comparison confirm our observation that both PRM1 and PRM2 outperform RM3, though PRM1 is slightly worse than PRM2, which is inconsistent with the observations on ClueWeb09 data set. More experiments are needed to understand which variation of the PRM is better.

6. CONCLUSIONS

In summary, we first studied proximity-based feedback, for which we propose a *positional relevance model* to exploit term proximity evidence so as to assign more weights to words closer to query words in pseudo-relevant documents; the second line of study was to improve the weighting of feedback documents in the relevance model by using a regression-based method to approximate the probability of relevance; thirdly, we also tested a supervised approach to query classification; in addition, we also evaluated a selective pseudo feedback strategy which skips pseudo feedback for precision-oriented queries and only uses it for recall-oriented ones.

The proposed PRM has been shown to clearly outperform the relevance model for pseudo feedback, suggesting that capturing term proximity appropriately could lead to a better feedback model; the relevance model with regression-based document weighting (RegRM) performs as well as relevance model, but no noticeable improvement is observed; however, the proposed query classification methods appear to not work well. The results also show that selective pseudo feedback may not work well if the decision only depends on whether a query is recall-oriented or precision-oriented.

Evaluation Metric		Official runs					Additional runs			
		uiuc09KL	uiuc09RegQL	uiuc09MProx	uiuc09GProx	uiuc09Adpt	RM3	PRM1	PRM2	Adpt+
mtc.base	eMAP	0.0713	0.0741	0.0750	**0.0762**	0.0720	0.0755	**0.0797**	0.0767	0.0730
	eP10	**0.2371**	0.2277	0.2268	0.2312	0.2258	0.2307	**0.2374**	0.2307	0.2284
	eP30	0.2433	**0.2476**	0.2463	0.2458	0.2457	0.2486	**0.2510**	0.2495	0.2429
	eP100	0.2216	0.2256	**0.2271**	0.2265	0.2251	0.2283	**0.2342**	0.2302	0.2243
mtc.reuse	eMAP	0.0686	0.0743	0.0750	**0.0754**	0.0732	0.0744	**0.0800**	0.0742	0.0737
	eP10	0.2581	0.2608	0.2583	**0.2666**	0.2596	0.2576	**0.2770**	0.2580	0.2692
	eP30	0.2592	0.2639	0.2641	**0.2648**	0.2626	0.2629	**0.2722**	0.2625	0.2633
	eP100	0.2310	0.2374	**0.2386**	0.2383	0.2377	0.2384	**0.2485**	0.2383	0.2389
statMAPs .base	statMAP	**0.2252**	0.2122	0.2091	0.2182	0.2169	0.2158	**0.2266**	0.2213	0.2177
	sMPC10	**0.2454**	0.2433	0.2428	0.2420	0.2297	**0.2454**	0.2412	0.2363	0.2291
	sMPC30	0.2829	0.2875	0.2819	0.2848	**0.2906**	0.2860	**0.2922**	0.2896	0.2760
	sMPC100	0.2650	0.2574	0.2534	**0.2701**	0.2492	0.2575	0.2633	0.2679	**0.2697**
statMAPs .reuse	statMAP	**0.2274**	0.2112	0.2044	0.2087	0.2108	0.2145	0.2196	0.2177	**0.2206**
	sMPC10	**0.3335**	0.3030	0.2910	0.3015	0.2901	0.2873	**0.3250**	0.3158	0.3021
	sMPC30	0.3055	**0.3095**	0.3040	0.2955	0.3054	**0.3220**	0.3180	0.3101	0.3010
	sMPC100	**0.2684**	0.2414	0.2395	0.2408	0.2567	0.2351	0.2530	0.2544	**0.2547**

Table 4: Comparison of different runs, including 5 official runs and 3 additional runs, on the ClueWeb09 data set. "sMPC" is an abbreviation of statMPC. The best results of official runs and additional are highlighted separately.

Collection	Metric	KL	RM3	PRM1	PRM2
WT2g	MAP	0.2869	0.3164	0.3254	**0.3276**
	Pr@10	0.4340	0.4720	0.4800	**0.4860**
	Pr@30	0.3153	0.3427	0.3460	**0.3507**
	Pr@100	0.1928	0.2094	**0.2122**	0.2106
Terabyte06	MAP	0.3047	0.3132	0.3214	**0.3223**
	Pr@10	**0.5367**	0.5102	0.5041	0.5245
	Pr@30	0.4653	0.4680	**0.4776**	**0.4776**
	Pr@100	0.3547	0.3571	0.3575	**0.3663**

Table 5: Comparison of different methods in the preliminary experiments on WT2g and Terabyte06 data sets. The best result for each row is highlighted.

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

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

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

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

[6] 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.

[7] 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.

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

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

[10] 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.