Relevance Feedback Exploiting Query-Specific Document Manifolds

Chang Wang*, Emine Yilmaz†,‡, Martin Szummer‡

wangchan@us.ibm.com, eminey@microsoft.com eyilmaz@ku.edu.tr ,szummer@microsoft.com

* IBM T. J. Watson Research Lab, New York, USA
† Koc University, Istanbul, Turkey
‡ Microsoft Research, Cambridge, UK

ABSTRACT

We incorporate relevance feedback into a learning to rank framework by exploiting query-specific document similarities. Given a few judged feedback documents and many retrieved but unjudged documents for a query, we learn a function that adjusts the initial ranking score of each document. Scores are fit so that documents with similar term content get similar scores, and scores of judged documents are close to their labels. By such smoothing along the manifold of retrieved documents, we avoid overfitting, and can therefore learn a detailed query-specific scoring function with several dozen term weights.

Categories and Subject Descriptors: H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval
General Terms: Algorithms
Keywords: Relevance Feedback, Manifold Learning

1. INTRODUCTION

Relevance feedback has been shown to be an effective way of improving accuracy in interactive information retrieval. Hence, many different algorithms that can use explicit or implicit feedback have been proposed in the literature.

Relevance feedback for probabilistic retrieval models works by changing the estimation of model parameters using the information provided by the relevant documents [10]. In the case of language modeling, on the other hand, feedback documents are used to alter the estimate of the query language model [7] or the relevance model [8]. Rocchio's algorithm incorporates relevance feedback information into the vector space model [11]. It is based on constructing a centroid vector from the feedback documents and moving the original query vector towards this centroid vector.

Recently, several machine learning techniques have been proposed to solve relevance feedback problems. Among them, global ranking using continuous conditional random fields (C-CRF) [9] is the closest to our work. C-CRF [9] exploits a ranking model defined as a function on all the judged documents with respect to the query. To infer the parameters of the ranking models, C-CRF also considers similarity between documents based on the contents. There are two main differences between C-CRF and our work. First, C-CRF does not make use of unjudged documents in learning. This could lead to overfitting problems, since the number of the judged documents in the case of relevance feedback is typically small. Second, C-CRF uses raw document contents to compute similarity between documents, while we are using query-specific content from each document for similarity computation.

During the past few years, there has been significant change in retrieval systems. Most modern search engines are now based on the learning to rank approach, which involves calculating a rich and diverse set of features that capture some aspect of relevance of the submitted query and a document. They then learn a combination of these features based on training data [3, 4]. Most relevance feedback algorithms are based on expanding the query by adding weighted terms from the feedback documents [11]. Even though learning to rank approaches are now popular for ranking, few relevance feedback algorithms follow that paradigm [6].

The challenge in applying the learning to rank paradigm stems from the fact that the number of given (judged) relevance feedback documents is very small, typically ten or less. At the same time the number of possible expansion terms is very large, on the order of vocabulary size, yielding a very large space of potential expansions to be considered. This unfavorable proportion of a small number of judged relevance feedback documents to the huge expansion space makes the problem difficult. An attempt at applying learning to rank methods involves taking an initial base ranker and adapting it by training it with the relevance feedback documents for the query; however, when this is done naively, it leads to overfitting (to the labelled documents).

In this paper, we incorporate relevance feedback into a learning to rank framework by exploiting query-specific document similarities. Given a few judged feedback documents and many retrieved but unjudged documents for a query, we learn a function that adjusts the initial ranking score of each document. Scores are regularized so that documents with similar term contents get similar scores, and scores of judged

documents are close to their labels. By such smoothing along the manifold of retrieved documents, we avoid overfitting, and can therefore learn a detailed query-specific scoring function. Using data from TREC and OHSUMED collections, we show that our algorithm produces significantly better results than relevance feedback methods trained only on judged feedback documents.

2. THE ALGORITHM

For each query: $A = \{a_1, \cdots, a_m\}$ represents the set of retrieved documents, where a_i is the original representation (using features like BM25, tf-idf, etc) of document i. $B = \{b_1, \cdots, b_l\}$, represents the judgement, where b_i is a_i's label ($l << m$). For example, we can use '+1' to represent "relevant", '-1' to represent "non-relevant". B is a $1 \times l$ matrix. The initial ranking model F_0 provides a ranking score for each document. $X = \{x_1, \cdots, x_m\}$ is a $t \times m$ matrix, where x_i is the new representation of document i. The desired adjustment $Y = \{y_1, \cdots, y_l\} = \{b_1 - F_0(a_1), \cdots, b_l - F_0(a_l)\}$, Y is a $1 \times l$ matrix.

The problem is formalized as follows: given a document set $X = \{x_1, \cdots, x_m\}$ represented over words, and the desired adjustment $Y = \{y_1, \cdots, y_l\}$ for the judged documents $\{x_1, \cdots, x_l\}$, where $l << m$, we want to construct a mapping function f to project any document x_i to a new space, where $f^T x_i$ matches x_i's desired adjustment y_i. In addition, we also want f to preserve the document manifold topology, such that similar documents get similar adjustments.

2.1 Notation

Notations used in the algorithm are as follows: W is a weight matrix, where $W_{i,j} = e^{-\|x_i - x_j\|^2}$ models the similarity of x_i and x_j. $\|x_i - x_j\|$ stands for the Euclidean distance between x_i and x_j in the vector space. D is a diagonal matrix: $D_{i,i} = \sum_j W_{i,j}$. $\mathcal{L} = D^{-0.5}(D - W)D^{-0.5}$ is the normalized graph Laplacian matrix corresponding to W. I is an $l \times l$ identity matrix. $U = \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix}$ is an $m \times m$ matrix. μ is a weight scalar. "Gap" vector $V = [y_1, \cdots y_l, 0, \cdots, 0]$ is a $1 \times m$ matrix. $()^+$ represents pseudo inverse.

2.2 The Cost Function

Solution to the problem in Section 2.1 is given by the mapping function f to minimize the following cost function:

$$\begin{aligned} & C(f) \\ = & \sum_{i \le l} (f^T x_i - (b_i - F_0(a_i)))^2 + \mu \sum_{i,j} (f^T x_i - f^T x_j)^2 W_{i,j}. \\ = & \sum_{i \le l} (f^T x_i - y_i)^2 + \mu \sum_{i,j} (f^T x_i - f^T x_j)^2 W_{i,j}. \end{aligned}$$

The first term of $C(f)$ is based on judged documents, and penalizes the difference between the mapped result of x_i and its desired adjustment y_i. The second term does not take label information into account. It guarantees that the neighborhood relationship (defined by document contents) within X will be preserved in the mapping. When x_i and x_j are similar, the corresponding $W_{i,j}$ is big. If f maps x_i and x_j to different positions, f will be penalized. The second term is useful to bound the mapping function f and prevents overfitting from happening. Here μ is the weight of the second term. When $\mu = 0$, the model disregards the unlabelled data, and data manifold topology is not respected.

This formulation is similar to score regularization [6], except that (i) we learn a parametric function of x rather than a nonparametric set of regularized scores, and (ii) the function represents an adjustment (an error) between the judgements and the initial ranking model F_0, instead of a function that directly approximates the judgements. The latter appears to be more difficult, and did not work as well in initial testing.

2.3 Dictionary Construction

Our algorithm makes use of query specific document similarities. The similarities are computed using a separate dictionary that depends on the contents of the feedback documents for each query. When we construct query specific dictionaries, we follow an idea that is commonly used in query expansion. In the first step, we retrieve all non-stop words from relevant judged documents. For each word, we sum up its term frequencies in all relevant judged documents resulting in $score^+$ and its term frequencies in all non-relevant judged documents resulting in $score^-$. We sort the terms using $score^+ - score^-$ following descending order. The top t words will be used in our query specific dictionary. We also add the query words to the dictionary if they are not already there.

2.4 The Main Algorithm and its Explanation

The main algorithm to re-rank documents for each query is given in Figure 1. For each judged document, our query-level re-ranker provides an adjustment for the initial ranking model to close the gap between the true label and the initial ranking score. The adjustment has to be learned from the judged documents. However, given the fact that the number of the judged documents is always small, the adjustment could overfit to the judged documents. To solve this problem, we add a regularizer that guarantees that documents (including both labeled and unlabeled documents) with similar contents get similar adjustments.

A question that naturally arises is how to define the query-specific similarity of documents. We know that the documents are represented by features like BM25, tf-idf in the initial ranking model. Such features only describe the relationship between query and document. They are not good to compute similarity between documents. A more reasonable way to compute document similarity is to use document contents. In this scenario, similarity between documents also depends on the given query as the similarities are computed only using the words in the query specific dictionaries. For different queries, the contributions of the same word could be quite different. Our query-level re-ranker uses the contents of the judged documents to create a dictionary that only has useful words for that query. When we compute similarity using document contents, we only consider the words in that dictionary.

2.5 Advantages

Our algorithm offers the following advantages:

(1) The algorithm exploits unlabeled data in the form of document contents. This allows it to learn a more complex retrieval function that weights individual terms. Overfitting is controlled by using the unlabeled data for manifold regularization.

(2) The algorithm relies on manifold learning, a rich model that does not rely on any strong distributional assumptions

1. **Represent each document using query-specific document contents: $X = \{x_1, \cdots, x_m\}$, where x_i is defined by t features, where the j^{th} component is the term frequency of the j^{th} word in the query specific dictionary in document i.**
2. **Construct graph Laplacian matrix $\mathcal{L}$ modeling the document manifold.**
3. **Construct the difference vector $V = [y_1, \cdots y_l, 0, \cdots, 0]$: For $i \in \{1, \cdots l\}$, compute the difference y_i between the base ranking score $F_0(a_i)$ and label b_i.**
4. **New ranking model $f = (X(U + \mu\mathcal{L})X^T)^+ XUV^T$.**
5. **Ranking score for document i is $F_0(a_i) + f^T x_i$.**

Figure 1: Relevance feedback ranker algorithm.

about the data. In contrast, other algorithms make more limiting assumptions about the data, such as Gaussianity.

(3) Only two parameters need to be manually set: μ controls how much we like to focus on manifold topology preservation. For simplicity, we use the same value for μ in all experiments. t is the maximum size of the query specific dictionary. We set it to 200 in all the tests.

(4) Closed form solution: unlike most approaches in this area, our algorithm (like [6]) provides a closed form solution of the result. The solution is global optimal regarding the cost function $C(f)$.

3. JUSTIFICATION

Our main algorithm is based on manifold regularization framework [2], which goes beyond regular regression models in that it applies constraints to those coefficients, such that the topology of the given data manifold will also be respected. A manifold is a mathematical space that on a small enough scale resembles the Euclidean space, but the global structure of a manifold may be more complicated. Manifold topology represents how instances (including both labeled and unlabeld instances) in the space is connected. Preserving manifold topology can be understood as a constraint that neighbors in the manifold space will be modulated in a similar way. Computing the optimal weights in a regression model and preserving manifold topology are conflicting objectives, manifold regularization provides a way to ideally balance the two goals.

Theorem 1: $f = (X(U + \mu\mathcal{L})X^T)^+ XUV^T$ minimizes the cost function $C(f)$.

Proof:

Given the input X, we want to find the optimal mapping function f such that $C(f)$ is minimized:

$$f = \arg\min_f C(f).$$

It is easy to verify that

$$\sum_{i \le l} (f^T x_i - y_i)^2 = f^T XUX^T f - 2f^T XUV^T + VUV^T.$$

We can also verify that

$$\mu \sum_{i,j} (f^T x_i - f^T x_j)^2 W_{i,j} = \mu f^T X\mathcal{L}X^T f.$$

$$C(f) = (f^T XUX^T f - 2f^T XUV^T + VUV^T) + \mu f^T X\mathcal{L}X^T f.$$

Using the Lagrange multiplier trick, we have

$$XUX^T f + \mu X\mathcal{L}X^T f = XUV^T.$$

This implies that

$$X(U + \mu\mathcal{L})X^T f = XUV^T.$$

So $f = (X(U + \mu\mathcal{L})X^T)^+ XUV^T$. $\square$

Interestingly, the solution to f includes $\mathcal{L}$, the normalized graph Laplacian matrix [5], which models data manifold topology. Recently, spectral graph theory combined with classical differential geometry and global analysis on manifolds forms the theoretical basis for "Laplacian" techniques for function approximation and learning on graphs and manifolds, using the eigenfunctions of a Laplace operator naturally defined on the data manifold to reveal hidden structure.

4. EXPERIMENTAL RESULTS

In our experiments, we use data from two different collections: OHSUMED and TREC 6–8 ad-hoc. The OHSUMED collection contains 106 queries in total, and each query contains about 150 judged documents on average, with an average 20% of them being relevant. The TREC collection employs a subset of Letor 3 features, as in [1]. This dataset contains 150 queries, roughly 1,500 documents are retrieved for each query, and roughly 7% of them are relevant. Both datasets are split into 5 folds, where each fold contains training, validation and testing sets. In all the experiments through this paper, we set $\mu = 100$.

In a standard relevance feedback task, there is an initial base ranking, which is shown to a user and the user gives feedback on some documents in this initial ranking. Our base ranker is a linear regression model. One way of using relevance feedback information is to use the labels of the feedback documents for a query to train a separate ranking model for that query that fits the difference from the base ranker. We call this approach the Query-level Greedy feedback Ranker (QGR). Another way is Positive-Negative feedback Ranker (PNR), which is an active relevance feedback approach [12]. The PNR model was shown to outperform standard relevance feedback algorithms such as Rocchio and language modelling based relevance feedback. Hence, we mainly compare our algorithm only with the PNR model.

For each of the 5 folds in the datasets, we run our base ranking algorithm on the queries on these sets. We then assume that the feedback documents are the top k documents retrieved by the base ranker (we tried $k = 5$ and 10 in this paper). We employ average precision (AP) to compare the quality of the manifold model with the base ranking model, the QGR model, and the PNR model. Our approach consistently outperforms the other methods, and the differences are statistically significant using a sign-rank test (Figure 2). For the TREC dataset, the results of the query greedy ranker are not included in the plot as the performance of the algorithm is much worse than the other three algorithms. Compared to PNR, which is implemented in the same algorithmic framework except using non-query-specific document contents to compute document similarity, manifold model achieves better results in all experiment settings. This is a strong indicator to show that the query-specific

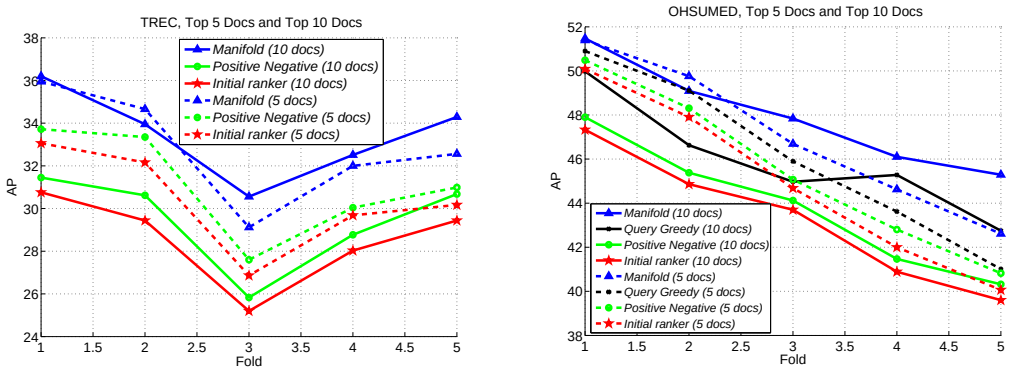

Figure 2: Improvements over the 5 folds for (left) TREC and (right) OHSUMED datasets when the feedback documents are the top 5 and 10 documents in the base ranking. Performances of two initial rankers ($k = 5, 10$) are different, because we throw away queries with no relevant (positive) feedback documents.

Table 1: An example to illustrate the behaviors of manifold model and query-level greedy model

Doc ID	1	2	3	4	5	6	7	8	9	10
Label	1	1	1	-1	1	-1	-1	-1	-1	1
Judged? 1:Yes 0:No	1	1	1	1	1	0	0	0	0	0
Initial Ranking Score	-0.6338	-0.2916	0.2835	-0.2635	0.1482	-1.0036	-1.0369	-0.9072	-0.7359	-0.3886
Greedy Re-ranking Score	1	1	1	-1	1	**-9.6383**	-1.5661	**164.2987**	-0.2086	-0.9875
Manifold Re-ranking Score	0.993	0.223	0.8615	0.2423	0.6801	-0.5156	-0.5491	-0.4094	-0.2456	0.1498

document contents can provide extra valuable information to query-level feedback rankers. The QGR model can be thought as a special case of a manifold model, where manifold topology is not respected. In these experiments, the greedy feedback ranker does not return satisfying results. The reason for this is that the greedy model can easily overfit for the feedback data.

In Table 1, we compare greedy model and manifold model in a real example. When initial ranking scores, judgements of 5 documents and a large amount of unjudged documents (only 5 are shown) are given, both models make adjustment to the original ranking scores. The greedy approach does not respect the manifold topology, so it assigns perfect re-ranking scores to the judged documents. But the ranking scores of the unjudged documents (documents with label -1) could be dramatically changed (see the big changes in the table). We carried out another experiment (not included in the figure) of query-level feedback ranker without using the base ranker; this yielded poor results. The reason is that the number of judged documents for each query is too small to construct a good feedback ranker (without a base ranker) even when manifold topology is respected. Base rankers are valuable, since they provide prior knowledge to help the feedback rankers when the feedback is limited.

5. CONCLUSIONS

We present a novel algorithm to incorporate relevance feedback into a learning to rank framework by exploiting query-specific document contents. Given a few judged feedback documents and many retrieved but unjudged documents for a query, our algorithm provides an optimal solution to adjust the base ranking score of each document, such that scores of judged documents are close to their labels and documents with similar term contents get similar adjustments.

6. REFERENCES

[1] J. A. Aslam, E. Kanoulas, V. Pavlu, S. Savev, and E. Yilmaz. Document selection methodologies for efficient and effective learning-to-rank. In *SIGIR*, 2009.

[2] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. 2006.

[3] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96, New York, NY, USA, 2005. ACM.

[4] C. J. C. Burges, R. Ragno, and Q. V. Le. Learning to rank with nonsmooth cost functions. In *NIPS*, pages 193–200, 2006.

[5] F. Chung. *Spectral graph theory*. 1997.

[6] F. D. Diaz. Improving relevance feedback in language modeling with score regularization. In *SIGIR*, 2008.

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

[8] V. Lavrenko and W. B. Croft. Relevance based language models. In *Proceedings of the 24th ACM SIGIR conference on Research and development in information retrieval*, pages 120–127, New York, NY, USA, 2001. ACM.

[9] T. Qin, T. Liu, X. Zhang, D. Wang, and H. Li. Global ranking using continuous conditional random fields. In *NIPS*, 2008.

[10] S. E. Robertson and K. Sparck Jones. Relevance weighting of search terms. pages 143–160, 1988.

[11] J. Rocchio. *Relevance Feedback in Information Retrieval*, pages 313–323. 1971.

[12] Z. Xu and R. Akella. A bayesian logistic regression model for active relevance feedback. In *SIGIR*, 2008.