Adaptive Relevance Feedback in Information Retrieval

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

Relevance Feedback has proven very effective for improving retrieval accuracy. A difficult yet important problem in all relevance feedback methods is how to optimally *balance* the original query and feedback information. In the current feedback methods, the balance parameter is usually set to a fixed value across all the queries and collections. However, due to the difference in queries and feedback documents, this balance parameter should be optimized for each query and each set of feedback documents.

In this paper, we present a learning approach to *adaptively* predict the optimal balance coefficient for each query and each collection. We propose three heuristics to characterize the balance between query and feedback information. Taking these three heuristics as a road map, we explore a number of features and combine them using a regression approach to predict the balance coefficient. Our experiments show that the proposed adaptive relevance feedback is more robust and effective than the regular fixed-coefficient feedback.

Categories and Subject Descriptors

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

General Terms

Algorithms

Keywords

Adaptive relevance feedback, relevance feedback, learning, prediction, language models

1. INTRODUCTION

Relevance Feedback has proven very effective for improving retrieval accuracy [27, 25, 28, 21, 32]. Relevance feedback refers to an interactive process that helps to improve the retrieval performance: when a user submits a query, an information retrieval system would first return an initial set of result documents and then ask the user to judge whether some documents are relevant or not; after that, the system would reformulate the query based on the user's judgments, and return a set of new results. Imagine that in a comprehensive search task [2], the aim of the user is to find as many documents as possible on a given topic; thus both recall and precision should be as high as possible. Relevance feedback, in this case, is a natural way to save users from reformulating queries in a trial-and-error manner.

Relevance feedback has been extensively studied, and most of the methods take it as a supervised learning problem with a special treatment of query [27, 25, 28, 32]. How to handle the relation between the query and feedback documents has been a difficult but important problem: we need to carefully *balance* the original query and feedback information because if we over-trust the feedback information, we may be biased to favor a particular subset of relevant documents, but under-trusting it would not take advantage of feedback. This problem is especially critical when we have only a few feedback documents, because the over-fitting would cause more damage. In existing feedback methods, the balance is usually controlled by some parameter, which is often set to a fixed value across all the queries and collections. However, due to the variations of queries and feedback documents, this balance parameter presumably should be optimized for each query and each set of feedback documents.

As far as we know, how to optimize the balance of the query and feedback information has not been well studied in previous work. In this paper, we study this novel problem and propose an *adaptive relevance feedback* method to dynamically predict an optimal balance coefficient using machine learning. Specifically, we estimate a potentially different feedback coefficient for each query and each set of feedback documents, rather than manually set it to a fixed constant. We hypothesize that the proposed method will do better than the current fixed-coefficient approaches.

Besides, we propose three heuristics to characterize feedback coefficients: (1) discrimination of query: we expect that the more discriminative the query is, the more drifting-tolerant it could be, and thus it would be safe to utilize more feedback information. (2) discrimination of feedback documents: we hypothesize that clearer feedback documents could be trusted more. (3) divergence between query and feedback documents: if the divergence between a query and its feedback documents is large, it possibly means that the query does not represent relevant documents well, thus we may need a larger feedback coefficient. Following these three heuristics, we explored a number of features and combined

CIKM'09, November 2–6, 2009, Hong Kong, China.
Copyright 2009 ACM 978-1-60558-512-3/09/11 ...$10.00.

them using the logistic regression model to predict the feedback coefficient.

We evaluate our algorithm on two representative TREC data sets. The experimental results demonstrate that our proposed adaptive relevance feedback has shown clear improvements over a robust baseline system with a well-tuned feedback coefficient, especially when the training data is noisy. It indicates that, the regular fixed-coefficient feedback only works when the training and testing sets are very similar, while the adaptive relevance feedback method is more noise-tolerant and can adapt to the characteristics of the documents and query effectively.

Our work makes the following contributions: (1) we propose an adaptive relevance feedback algorithm to dynamically handle the balance between query and feedback documents in relevance feedback. (2) we propose three heuristics to characterize the balance between original query and feedback information, based on which, a number of features are explored to dynamically predict the feedback coefficients.

2. RELATED WORK

Relevance feedback has been shown to be effective with different kinds of retrieval models [27, 25, 28, 21, 32, 3]. 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 relevant documents and minimizing its similarity to non-relevant documents [27]. The feedback method in classical probabilistic models is to select expanded terms primarily based on Robertson/Sparck-Jones weight [25]. In the language modeling approaches, relevance feedback can be implemented through estimating a query language model [32, 29] or relevance model [21] through exploiting a set of feedback documents. Recently, a relevance feedback track was organized by TREC to evaluate and compare different relevance feedback algorithms [3].

When there are no "real" relevance judgments available, alternatively, pseudo relevance feedback [4, 26] may be performed, which simply assumes that a small number of top-ranked documents in the initial retrieval results are relevant and then applies relevance feedback. Besides, somewhere in between relevance feedback and pseudo relevance feedback is implicit feedback [19], in which a user's actions in interacting with a system are used to infer the user's information need.

All the above work differs from our study in that they used a fixed parameter to control the balance between original query and feedback information, which often has to be manually set for different queries and feedback documents. Although in Tao and Zhai's work [29], a regularized EM algorithm was proposed to eliminate this balance parameter, it turns out that the weight of the original query and the weight of feedback documents are roughly the same (i.e., the balance parameter is fixed to 0.5) when the algorithm stops. Our study, however, proposes an adaptive relevance feedback approach to dynamically predict a potentially different parameter for different query and feedback documents.

Logistic regression [15] is widely used in data mining and machine learning. In IR, it has been used to learn a retrieval function that can be used to rank documents directly [13, 10, 14, 8, 34, 30]. A main difference of our work from these studies and recent work on learning to rank (e.g.,[18, 5, 6]) is that we leverage state-of-the-art language models for ranking documents and use logistic regression to optimize an important parameter *inside* the language modeling framework. Since current work on learning to rank has mostly relied on the traditional content-based retrieval models to compute features (e.g., BM25 scores, language model scores) and learn optimal ways to combine these features, our work can also be regarded a novel use of learning techniques to improve these features by going "inside" a traditional retrieval model instead of treating such a model as a black box.

Our work uses some features similar to the measures proposed in [11, 31, 7] to characterize queries, documents, as well as their relations (e.g., similarity, distance, and clarity). All these studies used these features to predict the query performance or query difficulty. But our work is to predict the balance between query and feedback information and optimize the performance of relevance feedback.

Selective query expansion is closely related to our work, where the idea is to disable query expansion if query expansion is predicted to be detrimental to retrieval [1, 12, 31, 17]. However, existing studies on selective query expansion only studied whether to do query expansion or not, while we focus on how much weight we can put on expansion information. Presumably, selective query expansion would not work for explicit relevance feedback, since almost all the queries benefit from feedback information.

3. PROBLEM FORMULATION

Given a query Q and a document collection $\mathcal{C}$, a retrieval system returns a ranked list of documents $\mathcal{L}$. Let L_i denote the i-th ranked document in the ranked list.

We assume that a user would be willing to give explicit feedback information. Our goal is to study how to use these feedback judgments, $\mathcal{J} \subseteq \{L_1, ..., L_k\}$, to effectively re-rank the next r unseen documents: $\mathcal{U} = \{L_{k+1}, ..., L_{k+r}\}$ to bring more relevant and novel information to the user.

For any relevance feedback model M, we assume that it has a parameter α to control the balance of query Q and feedback documents J. A general formula of feedback is:

$$M = (1 - \alpha)g_1(Q) + \alpha g_2(J)$$

where $\alpha \in [0, 1]$ is the feedback coefficient, and g_1 and g_2 are two functions that map a query and a set of relevant documents, respectively, into some comparable representations. For example, in the vector space model, queries and documents are represented as vectors of weighted terms, while in language modeling approaches, they are represented as language models.

Our goal is to optimize the feedback coefficient α for different queries and feedback documents. Formally, we assume there is a function B, that can map a query Q and the corresponding feedback documents J to the optimal feedback coefficient (i.e., $\alpha = B(Q, J)$), and hope to learn such a function B using past queries as training data.

In this study, we explore the problem of adaptive relevance feedback in the popular language modeling framework, particularly the KL-divergence retrieval model and mixture-model feedback method, mainly because language models deliver state of the art retrieval performance [24, 20, 33] and the mixture-model based feedback is one of the most effective feedback techniques which outperforms Rocchio feedback [32]. However, our methodology could be applicable to other retrieval and feedback models as well, which we leave as future work.

We now review our baseline language modeling framework and discuss how we incorporate our feedback prediction function into this framework.

3.1 The KL-Divergence Retrieval Model

The KL-divergence retrieval model [20] is a generalization of the query likelihood retrieval method [24] and can support feedback more naturally than the query likelihood method. In this model, queries and documents are all represented by unigram language models, which are essentially word distributions. Assuming that these language models can be appropriately estimated, KL-divergence retrieval model scores a document D with respect to a query Q by computing the negative Kullback-Leibler divergence between the query language model θ_Q and the document language model θ_D:

$$S(Q,D) = -D(\theta_Q||\theta_D) = -\sum_{w \in V} p(w|\theta_Q) \log \frac{p(w|\theta_Q)}{p(w|\theta_D)}$$

where V is the set of words in our vocabulary. Clearly, the retrieval performance of the KL-divergence would depend on how we estimate the document model θ_D and the query model θ_Q. The document model θ_D needs to be smoothed and an effective method is Dirichlet smoothing [33].

The query model intuitively captures what the user is interested in, and thus would affect retrieval accuracy significantly. Without feedback, θ_Q is often estimated as $p(w|\theta_Q) = p(w|Q) = \frac{c(w,Q)}{|Q|}$, where $c(w,Q)$ is the count of word w in the query Q, and $|Q|$ is the total number of words in Q.

3.2 The Mixture Model Feedback Method

The query model described above, however, is not very discriminative because a query is typically extremely short. Several different methods have been proposed to improve the estimation of θ_Q by exploiting documents, especially those documents that are used for relevance feedback or pseudo-relevance feedback [20, 21, 32, 29]. In [32], it was proposed that feedback can be implemented in the KL-divergence retrieval model as updating the query model based on the feedback documents. Specifically, we can define a two-component mixture model (i.e., a fixed background language model $p(w|\mathcal{C})$ estimated using the whole collection and an unknown topic language model to be estimated) and assume that the feedback documents are generated using such a mixture model. Formally, let θ_T be the unknown topic language model and $F \subset \mathcal{C}$ be a set of feedback documents (in explicit relevance feedback, F is comprised of documents that are judged relevant by the user.). The log-likelihood function of the mixture model is:

$$L(F|\theta_T) = \sum_{D \in F} \sum_{w \in V} c(w,D) \log[(1-\lambda)p(w|\theta_T) + \lambda p(w|\mathcal{C})]$$

where $\lambda \in [0,1)$ is a mixture noise parameter which controls the weight of the background model. Given a fixed λ, a standard EM algorithm can then be used to estimate $p(w|\theta_T)$, which is then interpolated with the original query model $p(w|Q)$ to obtain an improved estimation of the query model:

$$p(w|\theta_Q) = (1-\alpha)p(w|Q) + \alpha p(w|\theta_T)$$

where α is the feedback coefficient to be set manually. As in other existing feedback methods [27, 25, 28], the parameter α is generally fixed across all queries and documents. This model has been shown to perform quite well [32, 23] .

In this paper, we will study how to learn a function B to optimize the feedback coefficient α and plug it into the mixture model feedback to improve the retrieval performance. We view this problem as a prediction problem and propose learning methods for solving it. We now present our method.

4. A LEARNING APPROACH TO ADAPTIVE RELEVANCE FEEDBACK

We will first identify features possibly correlated to the feedback coefficient. Then we set it up as a learning problem, which can take past queries as training data to learn a function B to map these features to the feedback coefficient α. Finally, B can be used to predict new α for future queries.

4.1 Heuristics and Features

In this section, we propose three heuristics to predict the feedback coefficient: discrimination of query, discrimination of feedback documents, and divergence between query and feedback documents. The three heuristics capture intrinsic characteristics of the two main components (i.e. query and feedback document set) and the relationship between these two components in a feedback process. We propose a number of features guided by the three heuristics, but our method flexibly allows many other features (e.g., [16]) to be explored in the future work by taking the three heuristics as a road map.

4.1.1 Discrimination of Query

Intuitively, if a query is more discriminative, it could be more drifting-tolerant, and thus it would be safe to utilize more feedback information. So we expect the discrimination of query is correlated with the feedback coefficient. Several measures are proposed to quantify it as follows.

(1) Query Length:

A longer query is generally more discriminative than a short one, so our first feature is query length, formally defined as::

$$|Q| = \sum_{w \in Q} c(w,Q) \qquad (1)$$

where $c(w,Q)$ is the count of term w in Q.

(2) Entropy of Query:

Since the query is often very short, we compute query entropy score based on top-N result documents F' (Note that we have used a slightly different notation F for relevance feedback documents) in the initial retrieval, defined as follows:

$$QEnt\_A = \sum_{w \in F'} -p(w|\theta_{F'}) \log_2 p(w|\theta_{F'}) \qquad (2)$$

where $p(w|\theta_{F'})$ is estimated as $p(w|\theta_{F'}) = \frac{c(w,\mathcal{F}')}{\sum_w c(w,\mathcal{F}')}$.

(3) Clarity of Query:

The "query clarity" has shown to predict query difficulty well [11]. Therefore, we expect it to also be useful for predicting feedback coefficient. In the definition, the clarity of a query is the Kullback-Leibler divergence of the query model from the collection model.

To compute query clarity, we need to estimate the query language model first, which, however, involves again an interpolation between the original query model θ_Q and the pseudo feedback model $\theta_{F'}$, as well as the setting of an interpolation coefficient. To avoid this problem, in this paper, we do not estimate an entropy for the interpolated query model directly; instead, we compute two clarity scores based on θ_Q and $\theta_{F'}$ respectively and use them directly as features in a supervised learning framework, leaving the optimization of their combination to the training process.

To reduce the influence of common terms, $\theta_{F'}$ is smoothed using the collection language model with the Jelinek-Mercer smoothing method ($\lambda = 0.7$) [33]. Following [11], we define relative entropy $QEnt\_R1$ and $QEnt\_R2$ as:

$$QEnt\_R1 = \sum_{w \in Q} p(w|\theta_Q) \log \frac{p(w|\theta_Q)}{p(w|\mathcal{C})} \quad (3)$$

$$QEnt\_R2 = \sum_{w \in F'} p(w|\theta_{F'}) \log \frac{p(w|\theta_{F'})}{p(w|\mathcal{C})} \quad (4)$$

where $p(w|\mathcal{C})$ is the collection language model.

Intuitively, we would like $QEnt\_R1$ to contribute positively to the measure of the discrimination of query, which simply says that a higher $QEnt\_R1$ implies a more discriminative query. However, we can see that $QEnt\_R1$ favors queries that contain high IDF terms; these queries may be extremely specific and over-discriminative. To avoid assigning too high discrimination scores to such queries, we would like query discrimination to increase quickly when $QEnt\_R1$ is small but increase slowly as $QEnt\_R1$ is very large. This heuristic is defined formally as follows:

Let Q_1, Q_2 and Q_3 be three queries, and C_1, C_2 and C_3 be the corresponding query clarity scores computed based on Formula 3. We define $d(Q)$ as the discrimination of query Q. If $C_1 = C_2 - \epsilon$ and $C_2 = C_3 - \epsilon$, where ϵ is a small constant, then $d(Q_2) - d(Q_1) > d(Q_3) - d(Q_2)$.

To capture this heuristic, we propose another measure by taking a *logarithm* transformation on the $QEnt\_R1$ to approximate the true discrimination function $d(Q)$, as:

$$QEnt\_R3 = \log(QEnt\_R1) \quad (5)$$

Additionally, for $QEnt\_R2$, because we adopt a large λ to smooth $\theta_{F'}$, which not only "explains away" common terms but also decreases the strength of topical terms. To compensate the side effect of smoothing, we apply an exponential function to enlarge $QEnt\_R2$ and obtain another feature:

$$QEnt\_R4 = \exp(QEnt\_R2) \quad (6)$$

4.1.2 Discrimination of Feedback Documents

We only utilize documents that are judged relevant by the user for feedback and do not consider negative feedback in this study. Hence, intuitively if feedback documents are more discriminative, it possibly means that they focus more on the relevant topic and far away from noise. Therefore, discriminative feedback documents could be trusted more in the feedback process.

(1) Feedback Length:

We also introduce the number of feedback documents, i.e., feedback length $|F|$, as one feature, which is defined as:

$$|F| = \sum_{d} \delta(d, F) \quad (7)$$

where $\delta(d, F) = 1$ if document $d \in F$; otherwise 0.

(2) Feedback Radius:

To measure if feedback documents are concentrated on similar topics, we follow previous study [7] to measure the broadness of feedback documents. We define *feedback radius* as the average divergence between each document and the centroid of the feedback documents, which can be approximated using the Jensen-Shannon divergence [22] among feedback document models, defined below.

$$FBRadius = \frac{1}{|F|} \sum_{d \in F} \sum_{w \in d} p(w|\theta_d) \log \frac{p(w|\theta_d)}{p(w|\theta_{centroid})} \quad (8)$$

where $p(w|\theta_{centroid}) = \frac{1}{|F|} \sum_{d \in F} p(w|\theta_d)$

(3) Entropy of Feedback Documents:

Feedback length and feedback radius, as described above, capture the discrimination of feedback documents on the document level, whereas the entropy of feedback documents, which measures the term distribution, is on the term level. Similarly to the computation of query entropy, the entropy of feedback document model θ_F is defined as:

$$FBEnt\_A = \sum_{w \in F} -p(w|\theta_F) \log_2 p(w|\theta_F) \quad (9)$$

where $p(w|\theta_F)$ is estimated as $p(w|\theta_F) = \frac{c(w,\mathcal{F})}{\sum_w c(w,\mathcal{F})}$.

(4) Clarity of Feedback Documents:

Similar to query entropy $QEnt\_A$, the computation of feedback entropy $FBEnt\_A$ is also affected severely by common terms. So, we follow the same idea to smooth θ_F using Jelinek-Mercer smoothing method and then compute the "relative entropy of feedback documents" (or clarity of feedback documents) as an alternative feature, which is defined as:

$$FBEnt\_R1 = \sum_{w \in F} p(w|\theta_F) \log \frac{p(w|\theta_F)}{p(w|\mathcal{C})} \quad (10)$$

Similarly, to compensate the side effect of smoothing, we apply an exponential function to it and get another feature:

$$FBEnt\_R2 = \exp(FBEnt\_R1) \quad (11)$$

Furthermore, as we have discussed, another method to "explain away" common terms is to apply a mixture model to separate the topic model from the background model [32], which, although time-consuming, is applicable in our algorithm: since we will compute the topic model for feedback documents in the feedback stage anyway, we can obtain the topic model θ_T here by amortizing the computation cost. The reader can refer to paper [32] for more details. So, we get another feature:

$$FBEnt\_R3 = \sum_{w \in F} p(w|\theta_T) \log \frac{p(w|\theta_T)}{p(w|\mathcal{C})} \quad (12)$$

4.1.3 Divergence between Query and Feedback Documents

The motivation of divergence between query and feedback documents is that, we may rely on feedback more if the query does not represent relevant information well (i.e., the divergence between the query and its feedback documents is large.) Below, we list two measures to quantify it.

(1) Absolute Divergence:

A direct and intuitive way to estimate the divergence is computing the KL-divergence between query model θ_Q and feedback model θ_F.

With respect to θ_Q, usually we interpolate the original query model and the pseudo feedback model to obtain a more accurate query model. However, there are several problems with this method: first, we need to choose an interpolation coefficient; secondly, terms occurring in the original query will dominate the divergence score and the contribution of other related terms would not be rewarded appropriately; thirdly, the probability of common terms (e.g., ‘the’, ‘and’, etc.) in the interpolated query model θ_Q is possibly much less than its counterparts in the feedback model θ_F, which will affect the final score significantly. Based on these observations: we decide to simply use pseudo feedback model $\theta_{F'}$ instead of θ_Q to compute the divergence, which is defined as:

$$QFBDiv\_A = \sum_{w \in F} p(w|\theta_F) \log \frac{p(w|\theta_F)}{p(w|\theta_{F'})} \qquad (13)$$

To prevent zero probability, $\theta_{F'}$ is smoothed using the collection language model as $p(w|\theta_{F'}) = \frac{c(w,F')+\mu p(w|\mathcal{C})}{\sum_w c(w,\mathcal{F}')+\mu}$ where μ is set to 1500. We call this measure “absolute divergence” in contrast to the relative divergence to be discussed below.

(2) Relative Divergence:

A large absolute divergence value does not necessarily mean a bad query. For example, if the divergence between query and irrelevant documents is much larger than that between query and relevant documents, we can say that the query probably represents relevant information well, no matter what is the absolute divergence value.

To address this problem, we propose another feature to capture *relative* divergence. Intuitively, if a large portion of top-ranked documents are judged relevant by the user for feedback, it could indicate that the query represents the feedback documents well, suggesting there is possibly a small divergence between query and relevant documents. Motivated by this intuition, we adopt a *virtual* average precision to measure the relative divergence:

$$QFBDiv\_R = \sum_{d \in F} \frac{prec(r_d)}{K} \qquad (14)$$

where r_d is the rank of document d, e.g., the rank of the first document is 1 and the second one is 2 ...; $prec(r_d)$ is precision of top r_d documents; K is a constant.

4.2 Learning Algorithm

With these heuristically defined features, we hope to use some learning techniques to combine them to generate a score for predicting feedback coefficients. In principle, we may use any of the state-of-the-art learning methods. In this paper, we use the logistic regression model, which appears to handle our problem well: it can take as an input, any value from $-\infty$ to ∞, whereas the output is confined to values between 0 and 1.

Logistic regression models are also called maximum entropy models and are of the form:

$$f(z) = \frac{1}{1 + \exp(-z)}$$

where variable z represents some set of features, while $f(z)$ represents the probability of a particular outcome, given that set of features. We use $f(z)$ as the predicted value for the coefficient α, and interpret it as the probability that we would use only the feedback model (as opposed to the original query model) in the mixture model formed by interpolating the two models. Variable z is a measure of the total contribution of all the features used in the model, which is usually defined as $z = \bar{w}\bar{x}$. Specifically, $\bar{x}$ is a vector of numeric values representing the features, for instance, our features might include query length $|Q|$, the entropy of feedback documents, etc. And $\bar{w}$ represents a set of weights, which indicates the relative weights for each feature. A positive weight means that the corresponding feature increases the probability of the outcome, while a negative weight means that its corresponding feature decreases the probability of that outcome; a large weight means that the feature strongly influences the probability of that outcome, while a near-zero weight means that the feature has little influence on the probability of that outcome.

	Training sets		Testing set
	Terabyte04&05	TREC6&7&8	Terabyte06
topics	701-800	301-450	801-850
#queries	89	133	44
#total_qrel	22471	13692	6169

Table 1: Data set characteristic

Typically, we learn these weights from training data, which is described in section 5.1. Because logistic regression models have a global optimum, the choice of learning algorithm is usually of little importance. In our study, we use the statistical package R [1] to train our model.

Once the weight vector $\bar{w}$ of the equation has been derived for a particular data set (e.g., past queries), in our adaptive relevance feedback, the equation can be used to predict feedback coefficients online for new data sets (i.e., future queries).

5. EXPERIMENTS

5.1 Experiment Design

We used two standard TREC data sets in our study: Terabyte (GOV2) and TREC678 (TREC disk 4 and 5 minus Congressional Record). GOV2 data [9] is the largest test set publicly available for ad hoc retrieval with rich relevance judgments. And we used another large data set TREC678 to help further evaluate the adaptability of our algorithm to different training data. Queries were taken from the “title” field of the corresponding query topics.

Besides document collections and queries, we also need users’ feedback for each query. We chose to simulate feedback documents as follows. First, we employed the Lemur toolkit and Indri search engine [2] to index document collections and initially retrieved a document list for each query using our baseline KL-divergence retrieval model. And then, for each query, we assumed all relevant documents on the first result page (i.e., top-10 results) were judged by users for relevance feedback. We only used documents that were “judged” relevant for feedback and did not consider negative feedback in this study. Since there are a few queries that have no relevant documents in the first result page, we simply removed them from our evaluation data sets. Table 1 shows statistics of the training and testing queries, including the query topics, the number of queries with user “feedback”, and the total number of relevance judgments. All the above processing is the same for our training and testing sets. And the preprocessing of documents and queries is minimum, involving only stemming with the Porter stemmer. No stop words have been removed.

To train the proposed adaptive relevance feedback, as well as the fixed-coefficient relevance feedback, we need to obtain

[1] http://www.r-project.org/

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

the training data first. In our study, we split our queries into two parts: we used queries from Terabyte04 (topics 701-750), Terabyte05 (topics 751-800), TREC6 (topics 301-450), TREC7 (topics 351-400), and TREC8 (topics 401-450) as training queries to simulate the "past" queries, and took topics from Terabyte06 (topics 801-850) as testing queries to simulate queries in the "future". To get the optimal feedback coefficients for training queries, we used the baseline retrieval model with the mixture model feedback to do relevance feedback experiments on training data sets; through trying different feedback coefficients $\alpha \in \{0.0, 0.1, ..., 1.0\}$, we chose the optimal one for each query. For model training, we can train the two feedback methods on any subset of the training queries. The difference between the fixed-coefficient feedback and the adaptive feedback lies mainly in what they can learn from the training data: the former learns a fixed feedback coefficient α that leads to the best mean average precision on the training query set, while the latter learns a prediction model that best fits the training query set.

As for the evaluation, we first excluded the top-10 result documents (including both relevant and irrelevant documents) for all related queries from the collection, and then applied relevance feedback methods to retrieve documents from the *residue* collection. The reason is that we focus on how much the relevance feedback techniques can improve the accuracy of the *unseen* pages from users' perspective. After that, the top-ranked $1,000$ documents for all runs were compared in terms of their mean average precisions (MAP), which we used as our main evaluation metric. Besides, some other performance measures, such as precision at top-30 documents and recall at $1,000$ documents, were also considered in our evaluation. Additionally, to make the performance comparable, throughout our experiments, we fixed Dirichlet smoothing prior to 1500, feedback term count to 100, and mixture noise parameter λ in the mixture model to 0.9, and only left feedback coefficient for tuning.

5.2 Sensitivity of Feedback Coefficient

As we have discussed in Section 3, relevance feedback is usually controlled by an interpolation coefficient α. When $\alpha = 0$, we are only using the original query model (i.e., no feedback), while if $\alpha = 1$, we ignore completely the original query and rely only on the feedback model. To show the sensitivity of α, we plot the MAP of several randomly selected queries (TREC topics 757, 776, and 793) in relevance feedback experiments by varying α from 0 to 1. The results are shown in Figure 1. We can observe that the setting of α can affect the retrieval performance significantly, and the optimal coefficients for different queries could be quite different.

5.3 Feature Analysis and Selection

The objectives of this section are to link the features to feedback coefficients and to determine the effect of each feature on predicting optimal feedback coefficients.

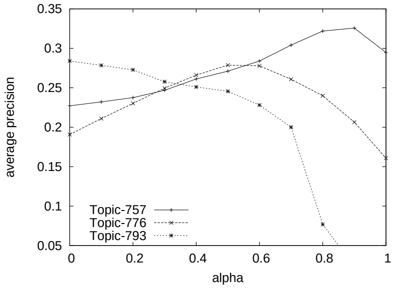

Figure 1: Sensitivity to the feedback coefficient (i.e., α) of several TREC query topics.

We first measure the correlation between features and the optimal feedback coefficient for each query on the Terabyte04&05 data, as reported in Table 2. Among these features, $FBRadius$, $FBEnt\_R2$, and $FBEnt\_R3$ are by far the most correlated factors to the optimal feedback coefficient, which are highlighted. It may mean that the discrimination of feedback documents plays the key role in predicting feedback coefficients. Some other factors, such as $QEnt\_R1$, $QEnt\_R3$, $QEnt\_R4$, $|Q|$, and $QFBDiv\_A$ have a lower, but still substantial effect of prediction, which shows that all the three heuristics proposed may be correlated to feedback coefficients. However, other factors, such as $QEnt\_A$, $QEnt\_R2$, $FBEnt\_A$, $FBEnt\_R1$, $QFBDiv\_R$, and $|F|$, have less effect and are thus discarded in our prediction model.

Next, we try to fit the training data using the remaining features. The assessment of fit is based on significance tests for the regression coefficients. Table 3 shows the significance of these features. By removing $QEnt\_R4$ and $|Q|$ one by one, we finally obtain a set of 6 features whose significance values are all close to or less than 0.01, and this feature set also minimizes the AIC (Akaike's Information Criteria) score. These 6 features are highlighted in Table 3, based on which, the following coefficients are derived from Terabyte04&05 data.

$$\begin{aligned} z = & - 22.69168 + 0.52229 * QFBDiv\_A \\ & - 0.12386 * FBEnt\_R2 + 0.50930 * FBEnt\_R3 \\ & - 0.87825 * QEnt\_R1 + 11.83219 * QEnt\_R3 \\ & - 1.61556 * FBRadius \end{aligned}$$

where we use the absolute value for each feature without any normalization, since there is no predictable value range for each feature. The last row of Table 2 also gives the correlation between the optimal feedback coefficient and z, which shows that the combination of these 6 features could be able to predict the optimal feedback coefficient. Given a new query, we can predict its feedback coefficient directly using the formula: $\alpha = \frac{1}{1+\exp(-z)}$.

From the above formula, we can see that, for two query clarity scores, $QEnt\_R1$ and $QEnt\_R3$, the former is correlated negatively to the feedback coefficient, while the latter shows a positive correlation. As we know, $QEnt\_R3 = \log(QEnt\_R1)$, so it may mean: (1) when query clarity score $QEnt\_R1$ is relatively small, "$11.83219 * QEnt\_R3$" dominates over "$-0.87825 * QEnt\_R1$", and the overall effect of query clarity is positive. This is consistent with our expectation that a more discriminative query is more drifting-tolerant and thus it is safe to use a large feedback coefficient. (2) However, when $QEnt\_R1$ is very large, "$-0.87825 * QEnt\_R1$" will be as important as or even dominate over "$11.83219 * QEnt\_R3$". In this case, the overall effect of query clarity score will be insignificant or even forced to be negative. One possible explanation is that a very large query clarity score may mean that the query is clear enough, and thus would not benefit much from feedback.

Features		Pearson	Spearman
No.	Name		
1	$\|Q\|$	-0.085	-0.118
2	$QEnt\_A$	-0.033	0.019
3	$QEnt\_R1$	0.051	0.162
4	$QEnt\_R2$	-0.126	0.032
5	$QEnt\_R3$	-0.083	0.162
6	$QEnt\_R4$	-0.181	0.032
7	$\|F\|$	-0.066	-0.091
8	**FBRadius**	-0.192	-0.210
9	$FBEnt\_A$	-0.019	-0.070
10	$FBEnt\_R1$	-0.028	0.151
11	**FBEnt_R2**	-0.160	0.151
12	**FBEnt_R3**	0.221	0.235
13	$QFBDiv\_A$	0.071	0.161
14	$QFBDiv\_R$	-0.051	-0.097
–	z	**0.556**	**0.564**

Table 2: Pearson and Spearman correlation coefficients between features and the optimal feedback coefficients on Terabyte training topics.

Features	Signif.	Features	Signif.
FBEnt_R3	$p = 9.13e-05$	**FBRadius**	$p = 0.006680$
FBEnt_R2	$p = 0.000105$	**QEnt_R1**	$p = 0.012837$
QFBDiv_A	$p = 0.002805$	**QEnt_R3**	$p = 0.013210$
$QEnt\_R4$	$p = 0.893582$	$\|Q\|$	$p = 0.856332$

Table 3: Statistical significance of features

The different behaviors of $FBEnt\_R2$ and $FBEnt\_R3$ could be explained as a trade-off between "discriminative" and "extreme" (i.e., too discriminative). When the discrimination value of feedback documents is relatively small, $FBEnt\_R3$ is often larger than $FBEnt\_R2$ due to the use of smoothing in computing $FBEnt\_R2$, and thus "$-0.12386*FBEnt\_R2+0.50930*FBEnt\_R3$" usually appears greater than zero; it may suggest that we can trust feedback more and use a higher feedback coefficient if the topic of feedback documents is more discriminative. However, when the discrimination of feedback documents is very large, $FBEnt\_R2$ would dominate over $FBEnt\_R3$ due to the use of exponential function in $FBEnt\_R2$, and thus "$-0.12386*FBEnt\_R2+0.50930*FBEnt\_R3$" is often less than zero; it possibly means that we do not need a large feedback coefficient if the feedback is too discriminative, since such feedback information can easily drift away from the original query.

The remaining two features $QFBDiv\_A$ and $FBRadius$ work similarly as we discussed in Section 4, so we do not duplicate explanations and analysis here.

To further examine our feature selection process, we train three prediction models on Terabyte04&05 data by using three different sets of features respectively: (i) the 6 most significant features, (ii) the 8 features in Table 3, and (iii) all of the proposed features. They are labeled as "LRM", "LRM+", and "LRM++" respectively. Besides retrieval performance, we also compute the Mean Absolute Error ("Pred. Err") to show how far off the coefficients used in each method and the optimal coefficients. The results are summarized in Table 4. We can observe that "LRM", although using fewer features, performs better than or comparably to "LRM++" and "LRM+". Therefore in all the following experiments, we train our adaptive feedback model using the "LRM" feature set.

We have not done feature analysis and selection on TREC678 and just trained our prediction model on it using the same "LRM" feature set. However, it would be interesting to explore and compare feature behaviors on different data sets in the future work.

–	Pred. Err	MAP	Prec@30	Recall
LRM	0.183	0.360	0.552	4110/6169
LRM+	0.183	0.360	0.552	4111/6169
LRM++	0.200	0.356	0.539	4091/6169

Table 4: Performance comparison of different feature sets.

5.4 Analysis of Computation Efficiency

Once the regression model has been estimated in an offline training process, we can predict feedback coefficients directly for "future" queries. Thus, the computation of the 6 features takes the major time in our approach. We show below all of these features can be computed efficiently for online prediction.

To compute the features, we only need to estimate the following language models based on the maximum likelihood estimation.

- M_1: query model $p(w|\theta_Q)$.
- M_2: language model of top-N (N is set to 50) retrieved documents as a whole, i.e., $p(w|\theta_{F'})$.
- M_3: language model of relevance feedback documents as a whole, i.e., $p(w|\theta_F)$.
- $M_4 = \{\theta_d\}$: language model of each relevance feedback document, i.e., $p(w|\theta_d)$, where $d \in F$; and let $\bar{M}_4 = \frac{1}{|F|}\sum_{d\in F}\theta_d$.

Besides, there are two other models needed:

- M_5: the topic model of relevance feedback documents, i.e., $p(w|\theta_T)$.
- M_6: collection language model $p(w|\theta_C)$.

However we can "borrow" M_5 from the feedback process without any extra effort, and M_6 is computed offline. All the 6 features can be estimated based on the above six language models: $QEnt\_R1 = D(M_1||M_6)$, $QEnt\_R2 = \log(QEnt\_R1)$, $FBEnt\_R3 = D(M_5||M_6)$, $FBEnt\_R2 = \exp(D(M_3{}^*||M_6))$, $FBRadius = \frac{1}{|M_4|}\sum_{\theta_d\in M_4} D(\theta_d||\bar{M}_4)$, and $QFBDiv\_A = D(M_3||M_2{}^*)$, where $D(M_i||M_j)$ represents the KL-divergence between M_i and M_j, and $M_j{}^*$ indicates a smoothed M_j (see Section 4 for details).

5.5 Performance of Adaptive Relevance Feedback

We evaluate the proposed adaptive relevance feedback in three variant cases: (1) The training set is an "ideal" one, that is, the training data and the testing data are in the same domain; (2) The training set is the "toughest" one, which is dominated by the data not in the same domain; (3) we have sufficient training data in the same domain, but it is mixed with "noisy" data.

We choose Terabyte04&05 to simulate an "ideal" training data since Terabyte 04, 05, and 06 share the same document collection and use similar query topics. We train both the adaptive relevance feedback (AdaptFB) method and the fixed-coefficient relevance feedback (FixedFB) method on this data set. In addition, we also introduce another run (labeled as "OptimalFB"), in which we manually set the optimal feedback coefficient for each query; it defines an

–	Pred. Err	MAP	Prec@30	Recall
Baseline	–	0.279	0.448	4033/6169
OptimalFB	0.000	0.389	0.591	4176/6169
FixedFB	0.210	0.357	0.542	4092/6169
AdaptFB	0.183	0.360	0.552	4110/6169

Table 5: Performance comparison of AdaptFB and FixedFB using Terabyte04&05 as training set.

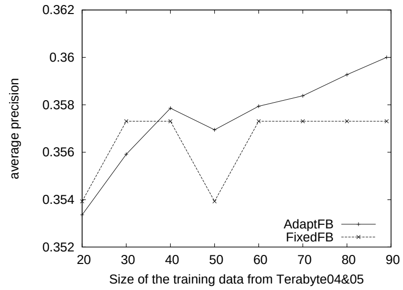

Figure 2: Sensitivity to number of queries for training.

upper bound of adaptive relevance feedback. The results are reported in Table 5. It shows that both AdaptFB and FixedFB outperform the baseline (without any feedback) significantly. Comparing them with OptimalFB, we can see there is still room to improve the retrieval performance by further optimizing feedback coefficients.

In this ideal environment, although AdaptFB is more effective than FixedFB, the improvement is not significant, suggesting that the regular FixedFB is also a reasonable method when we have consistent training data in the same domain. One possible reason could be that we only have 89 training queries in Terabyte04&05, which may be insufficient to train an accurate prediction model. To examine it, we draw the learning curves w.r.t. the number of queries for training in Figure 2. It is interesting to see an increasing trend in the performance of AdaptFB as we increase the number of training queries, while the performance of FixedFB appears to be stable. This observation suggests that AdaptFB would potentially outperform FixedFB more if we have more training data. Further experiments and analysis are needed to understand this better.

Since our learning approach is to minimize the prediction error, rather than to maximize the retrieval performance directly, one interesting question is whether lower prediction errors have enabled AdaptFB to outperform FixedFB. So, we plot the improvement of MAP as a function of the reduction of prediction error in Figure 3. The reduction of prediction error is formally defined as: $|\alpha_o - \alpha_f| - |\alpha_o - \alpha|$, where α_o is the optimal feedback coefficient and α_f is the well-tuned fixed coefficient. Figure 3 shows clearly a strong positive correlation between these two objective functions, which indicates that the improvement of performance is indeed due to the reduction of prediction errors.

In reality, we often do not have, or do not have sufficient training data in the same domain. We start with an extreme situation when our training data are all in a different domain, and we use TREC6&7&8 data to simulate such training data. We then gradually add the Terabyte04 and Terabyte05 data into the training set to mix with the existing TREC6&7&8 data to simulate the scenario when we have more and more training data in the right domain. We compare AdaptFB and FixedFB in Table 6. The results show that AdaptFB yields significantly better results than FixedFB with insufficient training data of the same domain. We further compare the sensitivity of the two methods in Figure 4. It is clearly observed that, with only a few (e.g., 20) training samples from Terabyte topics, AdaptFB has already reached reasonably good performance. However, even after we add all the Terabyte04&05 data into the training set, the performance of FixedFB is still very poor. This shows that AdaptFB picks up test domain characteristics more quickly than FixedFB. One possible explanation is that, FixedFB often over-fits the specified training data; the proposed AdaptFB, however, can really adapt to the characteristics of feedback and query effectively due to the use of feedback and query-specific features.

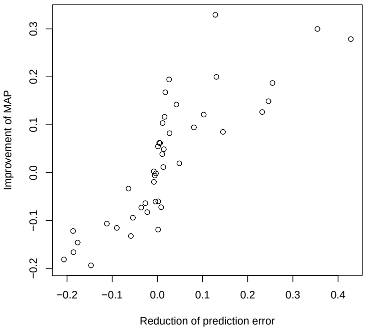

Figure 3: The correlation between the reduction of prediction error and the improvement of MAP.

Training set	Method	MAP	Prec@30	Recall
+0	FixedFB	0.308	0.482	3741/6169
	AdaptFB	0.328^+	0.505^+	$3919/6169^+$
+Terabyte04	FixedFB	0.326	0.497	3851/6169
	AdaptFB	0.350^+	0.536^+	$4021/6169^+$
+Terabyte04&05	FixedFB	0.340	0.517	3919/6169
	AdaptFB	0.354^+	0.545^+	$4041/6169^+$

Table 6: Comparison of AdaptFB and FixedFB when Terabyte04&05 data is gradually added into the existing training set (i.e., TREC6&7&8). $'+'$ means that the corresponding improvements over FixedFB are statistically significant.

In the above experiments, although we added all of the Terabyte04&05 data into the training set finally, the performance of AdaptFB and FixedFB is still not as good as

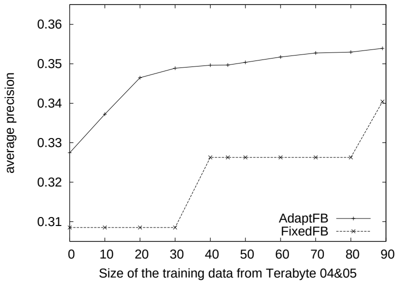

Figure 4: Sensitivity to the insufficiency of training data in the same domain.

Training set	Method	MAP	Prec@30	Recall
+TREC6	FixedFB	0.356	0.542	4063/6169
	AdaptFB	0.360	0.552$^+$	4080/6169
+TREC6&7	FixedFB	0.340	0.517	3919/6169
	AdaptFB	0.356$^+$	0.544$^+$	4053/6169$^+$
+TREC6&7&8	FixedFB	0.340	0.517	3919/6169
	AdaptFB	0.354$^+$	0.545$^+$	4041/6169$^+$

Table 7: Comparison of AdaptFB and FixedFB when TREC6&7&8 data is gradually added into the existing training set (i.e., Terabyte04&05). $'+'$ means that the corresponding improvements over FixedFB are statistically significant.

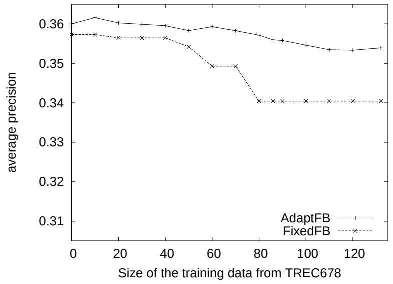

Figure 5: Sensitivity to the noisy training queries (i.e., TREC6&7&8) in the training data.

that in the "ideal" case. It indicates that the "noisy" data (i.e., TREC6&7&8) contributes negatively to the prediction accuracy.

Therefore, we now go to the third question: what is the influence of the noise to two methods if the training set is involved in "noisy" data? In fact, in real Web search environment, it is often hard to obtain a training set of the same domain for every query; thus, we will generally have some "noisy" data in the training set. In order to understand the influence of noise, we keep the current Terabyte04&05 data in our training set, but at the same time we add TREC6, TREC6&7, and TREC6&7&8 data into the training data set respectively to learn new prediction models. The results are reported in Table 7. We can observe that AdaptFB outperforms FixedFB significantly. To understand the sensitivity of two methods better, we further plot the sensitivity curves w.r.t. the noisy data in Figure 5. We can see that the performance of FixedFB decreases dramatically when there is noisy data, but AdaptFB is more stable. Interestingly, it is observed that the performance of AdaptFB is even improved slightly when a little noisy data is introduced. Both Table 7 and Figure 5 show that FixedFB only works when the training and testing sets are very similar, and it would not work well when the training set is noisy; while AdaptFB is robust and effective in both cases. It means that AdaptFB is more noise-tolerant than FixedFB.

To make the comparison between AdaptFB and FixedFB more illustrative, we also plot the average precision of the two methods in Figure 6, where both of the methods are trained on a mixture of Terabyte04&05 data and TREC6&7 data. We can see that AdaptFB outperforms FixedFB for most of the queries with only a few exceptions. In addition, AdptFB appears to work especially effectively for *difficult* queries. Further experiments are needed to study and understand this interesting observation better.

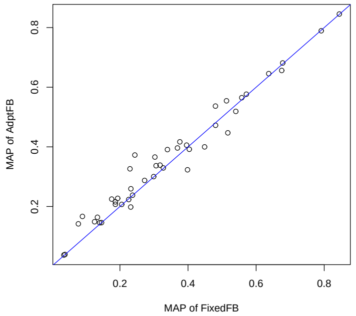

Figure 6: MAP plot for AdaptFB and FixedFB.

6. CONCLUSIONS

In this paper, we proposed an adaptive relevance feedback algorithm to optimize the balance between query and feedback information. First, three heuristics were proposed to characterize the balance between query and feedback information, including discrimination of query, discrimination of feedback documents, and divergence between query and feedback documents. Then, taking these three heuristics as a road map, we explored a number of features and combined them using the logistic regression model to predict the balance coefficient. Finally, we did extensive experiments to evaluate our algorithm from different perspectives; our experiments show that the three heuristics are all very important, each capturing one aspect of information to predict the feedback coefficient, and the proposed adaptive relevance feedback is more robust and effective than the regular fixed-coefficient feedback, especially when the training data is noisy: it suggests that, the regular fixed-coefficient feedback only works when the training and testing sets are very similar; while the adaptive relevance feedback is more noise-tolerant and can adapt to the general characteristics of feedback and query effectively.

There is still room to explore in the future work. First, the proposed method relies on explicit user feedback for training, so it would be interesting to study how to adaptively exploit pseudo and implicit feedback. Secondly, we only evaluated the proposed methodology of adaptively learning feedback coefficient for the mixture-model feedback method, so we also hope to apply our idea to other feedback approaches, e.g., Rocchio feedback, to examine its performance. Thirdly, the training in the proposed method needs target values for the feedback coefficients, and a possibly more natural strategy which would be done in our future work could be to learn the coefficient so as to optimize some measure related to the MAP criterion, e.g., using a learning to rank method, which would remove the need for computing desired coefficient values. Fourthly, we should study more effective and robust features in the future. Fifthly, it would also be interesting to explore and incorporate negative feedback into the proposed adaptive relevance feedback method.

7. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their useful comments. We also thank Wan Chen for her help to 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, and IIS-0713571.

8. REFERENCES

[1] Giambattista Amati, Claudio Carpineto, and Giovanni Romano. Query difficulty, robustness, and selective application of query expansion. In *Proceedings of ECIR '04*, pages 127–137, 2004.

[2] Anne Aula. Query formulation in web information search. In *Proceedings of ICWI '03*, pages 403–410, 2003.

[3] Chris Buckley and Stephen E. Robertson. Relevance feedback track overview: Trec 2008. In *TREC '08*, 2008.

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

[5] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 89–96, 2005.

[6] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of ICML*, pages 129–136, 2007.

[7] David Carmel, Elad Yom-Tov, Adam Darlow, and Dan Pelleg. What makes a query difficult? In *Proceedings of SIGIR '06*, pages 390–397, 2006.

[8] Ben Carterette and Desislava Petkova. Learning a ranking from pairwise preferences. In *Proceedings of SIGIR '06*, pages 629–630, 2006.

[9] Charles Clarke, Nick Craswell, and Ian Soboroff. Overview of the trec 2004 terabyte track. In *Proceedings of TREC '04*, pages 500–261, 2004.

[10] William S. Cooper, Fredric C. Gey, and Daniel P. Dabney. Probabilistic retrieval based on staged logistic regression. In *Proceedings of SIGIR '92*, pages 198–210, 1992.

[11] Stephen Cronen-Townsend, Yun Zhou, and W. Bruce Croft. Predicting query performance. In *Proceedings of SIGIR '02*, pages 299–306, 2002.

[12] Stephen Cronen-Townsend, Yun Zhou, and W. Bruce Croft. A framework for selective query expansion. In *Proceedings of CIKM '04*, pages 236–237, 2004.

[13] Norbert Fuhr and Chris Buckley. A probabilistic learning approach for document indexing. *ACM Transactions on Information Systems (TOIS)*, 9(3):223–248, 1991.

[14] Fredric C. Gey. Inferring probability of relevance using the method of logistic regression. In *Proceedings of SIGIR '94*, pages 222–231, 1994.

[15] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning.* Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[16] Claudia Hauff, Djoerd Hiemstra, and Franciska de Jong. A survey of pre-retrieval query performance predictors. In *Proceedings of CIKM '08*, pages 1419–1420, 2008.

[17] Ben He and Iadh Ounis. Combining fields for query expansion and adaptive query expansion. *Inf. Process. Manage.*, 43(5):1294–1307, 2007.

[18] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM KDD 2002*, pages 133–142, 2002.

[19] Diane Kelly and Jaime Teevan. Implicit feedback for inferring user preference: a bibliography. *SIGIR Forum*, 37(2):18–28, 2003.

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

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

[22] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Trans. Infor. Theory*, 37:145–151, 1991.

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

[24] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR '98*, pages 275–281, 1998.

[25] 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.

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

[27] 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.

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

[29] Tao Tao and ChengXiang Zhai. Regularized estimation of mixture models for robust pseudo-relevance feedback. In *Proceedings of SIGIR '06*, pages 162–169, 2006.

[30] Zuobing Xu and Ram Akella. A bayesian logistic regression model for active relevance feedback. In *Proceedings of SIGIR '08*, pages 227–234, 2008.

[31] Elad Yom-Tov, Shai Fine, David Carmel, and Adam Darlow. Learning to estimate query difficulty: including applications to missing content detection and distributed information retrieval. In *Proceedings of SIGIR '05*, pages 512–519, 2005.

[32] ChengXiang Zhai and John D. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of 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 *Proceedings of SIGIR '01*, pages 334–342, 2001.

[34] Zhaohui Zheng, Keke Chen, Gordon Sun, and Hongyuan Zha. A regression framework for learning ranking functions using relative relevance judgments. In *Proceedings of SIGIR '07*, pages 287–294, 2007.