A Study of Adaptive Relevance Feedback – UIUC TREC-2008 Relevance Feedback Experiments

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

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

ABSTRACT

In this paper, we report our experiments in the TREC 2008 Relevance Feedback Track. Our main goal is to study a novel problem in feedback, i.e., *optimization of the balance of the query and feedback information*. Intuitively, 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. In the current 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 difference in queries and feedback documents, this balance parameter should be optimized for each query and each set of feedback documents.

To address this problem, we present a learning approach to adaptively predict the balance coefficient (i.e., feedback coefficient). First, three heuristics are proposed to characterize the relationships between feedback coefficient and other measures, including discrimination of query, discrimination of feedback documents, and divergence between the query and the feedback documents. Then, taking these three heuristics as a road map, we explore a number of features and combine them using a logistic regression model to predict the feedback coefficient. Experiments show that our adaptive relevance feedback is more robust and effective than the regular fixed-coefficient relevance feedback.

1. INTRODUCTION

Among many techniques for improving the accuracy of ad hoc information retrieval, relevance feedback is arguably one of the most effective techniques and has been shown to be effective with variety of retrieval models [7, 6, 8, 4, 10]. In the vector space model, feedback is usually done with the Rocchio algorithm, which forms a new query vector by maximizing its similarity to relevant documents and minimizing its similarity to non-relevant documents [7]. The feedback method in classical probabilistic models is to select expanded terms primarily based on Robertson/Sparck-Jones weight [6]. In the recently proposed language modeling approaches, relevance feedback can be implemented through estimating a query language model [3, 10] or relevance model [4] through exploiting a set of feedback documents.

All these existing methods show that combining feedback information with the original query typically improves the performance. However, we need to carefully *balance* the 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. In the current 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 difference in 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. Thus, in our work, we study this novel problem in relevance feedback and propose an adaptive feedback method which predicts a dynamic balance coefficient by using a learning approach. 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.

We explore a number of features potentially correlated with the feedback coefficient and classified them into three categories: (1) discrimination of query: we expect that the "clearer" (i.e., more discriminative) the query is, the less feedback we need. (2) discrimination of feedback documents: we hypothesize that clearer feedback documents can be trusted more. (3) divergence between the query and the feedback documents: if the divergence between a query and its feedback documents is large, it 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 them using a logistic regression model [2] to predict the feedback coefficient.

Through preliminary experiments, we observe that, although a well-tuned fixed coefficient is not optimal for many queries, it provides a "safe" coefficient range. Compared with it, our predicted value is sometimes too extreme and thus "risky." So we also experimented with some strategies to smooth our prediction using the safe fixed coefficient value. We hypothesize that, with smoothing, our adaptive relevance feedback method would be more robust.

In our experiments, the basic retrieval method is the KL-divergence retrieval model [3] with the Dirichlet smoothing method [9] plus a generative mixture model feedback method [10], which adopts our predicted feedback coefficient. Our proposed method has shown clear improvements

in our experiments over a robust fixed-coefficient relevance feedback method; it is also observed that most features that we explore help predict the feedback coefficient. Through further analysis, we find that our adaptive relevance feedback approach is still robust and effective even if training and testing data sets are inconsistent.

In the rest of this paper, we will first introduce our basic retrieval method in Section 2. After that, we will present the adaptive relevance feedback method in Section 3. In Section 4, we will describe how to smooth the prediction value to make relevance feedback more robust. We report our experimental results in Section 5 and conclude our work in Section 6.

2. RETRIEVAL METHOD

To make our algorithm clear, we break down the relevance feedback task into four steps: initial retrieval, adaptive feedback coefficient prediction, coefficient smoothing, and query language model updating. At the retrieval step, we adopt the KL-divergence retrieval model with Dirichlet smoothing method to do an initial retrieval, based on which, a couple of features are explored to predict the feedback coefficient using the logistic regression model. After that, several strategies are applied to smooth our prediction value using a tuned fixed coefficient. Finally, the smoothed value is plugged into the mixture model relevance feedback method to update the query language model.

In this section, we present our basic retrieval approaches, the KL-divergence retrieval model and the mixture model feedback method.

2.1 The KL-Divergence Retrieval Model

The KL-divergence retrieval model [3] is a generalization of the query likelihood retrieval method proposed in [5] and can support feedback more naturally than the query likelihood method. In this model, all the queries and documents are represented by unigram language models, which are essentially word distributions. Assuming that these language models can be appropriately estimated, the 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 as follows:

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

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 [9]:

$$p(w|\theta_D) = \frac{c(w,D)+\mu p(w|\mathcal{C})}{|D|+\mu}$$

where $p(w|\mathcal{C})$ is the collection language model and is estimated with $p(w|\mathcal{C}) = \frac{c(w,\mathcal{C})}{\sum_w c(w,\mathcal{C})}$, and μ is a smoothing parameter and is usually set empirically. Across all of our experiments, we used the Dirichlet prior smoothing method for estimating document language models.

The query model intuitively captures what the user is interested in, 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 the query.

2.2 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 [3, 4, 10]. In [10], 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 $\mathcal{F} \subset \mathcal{C}$ be a set of feedback documents. The log-likelihood function of the mixture model is:

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

where λ is a mixture noise parameter which controls the weight of the background model. Given a fixed λ ($\lambda = 0.9$ in our experiments), a standard EM algorithm can then be used to estimate parameters $p(w|\theta_T)$, 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. Similarly to other existing feedback methods [7, 6, 8], the parameter α in this formula is generally fixed across all queries and documents.

However, due to the difference in queries and feedback documents, the coefficient α, which indicates the balance between query and feedback, should be optimized for each query and each set of feedback documents. This motivates us to study how to optimize the balance of the query and feedback information. We view this problem as a prediction problem and propose a learning approach to solve it. Although we explore this idea in the context of the mixture model feedback method in this paper, it could be applicable to other feedback methods as well. We now present our method.

3. FEEDBACK COEFFICIENT PREDICTION

3.1 Heuristics and Features

In this work, we investigate three heuristics to predict the feedback coefficient: discrimination of query, discrimination of feedback documents, and divergence between the query and the 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 argue, and then show experimentally in Section 5, that the three heuristics all play important roles in predicting the feedback coefficient. Possibly, many other features can be explored by taking the three heuristics as a road map.

3.1.1 Discrimination of Query

Intuitively, if the query itself is discriminative enough, we do not need to rely heavily on feedback documents. Hence, we expect the discrimination of query is correlated with the feedback coefficient. Several measures are proposed to quantify it.

(1) Query Length: Intuitively, for two queries Q_1 and Q_2, if Q_1 is longer than Q_2 (i.e. Q_1 has more terms than Q_2), Q_1 is usually discriminative than Q_2. Therefore, the query length could be a characteristic of the discrimination of a query. To capture this intuition, we introduce query length $|Q|$ as our first feature. Formally, it is defined as:

$$|Q|\text{: the number of terms in query } Q.$$

(2) Entropy of Query: It is known that more entropy means more randomness and less discrimination. Therefore, we could adopt such a concept to measure how discriminative a query is. To compute the entropy, 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 document model $\theta_{F'}$ as well as the setting of a feedback coefficient. (Note that we have used a slightly different notation θ_F for relevance feedback document model, and throughout this paper we estimate pseudo feedback models by using the top 50 documents.) To avoid this problem, in this paper, we do not estimate an entropy for the interpolated query model directly, instead, two entropy scores respectively for θ_Q and $\theta_{F'}$ are computed, allowing the training system to weigh them, which is expected to get an appropriate approximation. Assume that each query term only appears once in a query, the entropy of θ_Q is defined as:

$$QEnt\_A1 = \sum_{w \in Q} -p(w|\theta_Q) \log_2 p(w|\theta_Q) = \log_2 |Q|$$

Where θ_Q is estimated as $p(w|\theta_Q) = \frac{c(w,Q)}{|Q|} = \frac{1}{|Q|}$. We can see that $QEnt\_A1$ is a negative logarithm transformation of query length $|Q|$. Thus in effect, we just have another query length feature.

Similarly, we defined the entropy of $\theta_{F'}$ as follows:

$$QEnt\_A2 = \sum_{w \in F'} -p(w|\theta_{F'}) \log_2 p(w|\theta_{F'})$$

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

(3) Relative Entropy of Query: In the definition above, query entropy is affected significantly by common terms (e.g., 'the', 'and', ...). This problem can be addressed by using a mixture model to separate the topic model from the background model [10]. However, they both are quite time-consuming. So, we adopt a similar idea of "relative entropy of query" as proposed in [1] to compute the query clarity score, which measures the coherence of the language usage in query language models as compared to the collection model. The "query clarity" has been shown an intrinsic feature of queries and has an important impact on the retrieval performance [1]. Therefore, we expect that it can also predict the feedback coefficient.

In the definition, the clarity of a query is the Kullback-Leibler divergence of the query model from the collection model. Similar to the computation of query entropy, an important role in this definition is the estimation of a query model. To avoid it, we use the same strategy by computing two clarity scores for θ_Q and $\theta_{F'}$ respectively.

To further reduce the effect of common terms, $\theta_{F'}$ is smoothed with the collection language model using Jelinek-Mercer smoothing method with a λ of 0.7[9]. Following [1], we define relative entropy $QEnt\_R1$ and $QEnt\_R2$ as follows:

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

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

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

3.1.2 Discrimination of Feedback Documents

Intuitively, if feedback documents are more discriminative, it means that they focus more on the relevant topic and far away from noise. Therefore, discriminative feedback documents can be trusted more in the feedback process.

(1) Feedback Length: For a query Q and two possible relevant judgment sets F_1 and F_2, if F_1 has more documents than F_2, usually F_1 contains more intensive relevant information than F_2; thus, F_1 could be discriminative than F_2 in describing relevant information. Therefore, the number of feedback documents, which we define as feedback length, can be taken as a characteristic of the discrimination of feedback document set. Formally, feedback length $|F|$ is defined as follows:

$$|F|\text{: the number of documents in } F.$$

(2) Entropy of Feedback Documents: Feedback length, as described above, captures 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. Usually, more entropy means a more random term distribution; thus it is not clear which topic the feedback documents talk about. Similarly to the computation of query entropy, the entropy of feedback model θ_F is defined as:

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

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

(3) Relative Entropy of Feedback Documents: Similar to Query Entropy $QEnt\_A2$, the computation of feedback document 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" as an alternative feature, which is defined as follows:

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

3.1.3 Divergence between Query and Feedback Documents

The motivation of divergence between query and feedback documents is that, we have to 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 divergence between query model θ_Q and feedback model θ_F using the KL-divergence formula. It is clear that θ_F is easily estimated using the Maximum Likelihood estimator: $p(w|\theta_F) = \frac{c(w,\mathcal{F})}{\sum_w c(w,\mathcal{F})}$.

We simply use pseudo feedback document model $\theta_{F'}$ instead of θ_Q to compute the divergence, which is defined below:

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

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 divergence "absolute divergence" in contrast to the relative divergence to be defined below.

(2) Relative Divergence:

With the above absolute divergence, it is often difficult to say that a large divergence value means a bad query, because the absolute divergence only relies on θ_F and θ_Q but does not take other useful factors into consideration, e.g., the divergence between query model and negative feedback model. In fact, if the divergence between query and negative feedback is much larger than that between query and positive feedback documents, we can say that the query represents relevant information well, no matter what is the absolute divergence value.

To address this problem, we propose another feature to capture a relative divergence. Considering a scenario: in a searching process, if document D is judged as a relevant document but its *rank* in the result document list is very low, it also shows that the query does not represent the feedback documents well. Hence, intuitively, the rank of a document also measures the divergence between query and feedback documents, and such a measure seems more comparable among different queries. Because there are sometimes more than one feedback documents, we adopt an *average rank* in our predicting system, as follows:

$$QFBDiv\_R1 = \sum_{d \in F} \frac{r_d}{|F|}$$

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 ...; $|F|$ is feedback length as described before.

In the formula above, a large $QFBDiv\_R1$ value means a low rank. Intuitively, we would like $QFBDiv\_R1$ to positively contribute to the measure of the divergence between query and feedback documents, which simply says that a higher $QFBDiv\_R1$ implies a larger divergence. However, we would like the contribution from a rank measure to drop quickly when the $QFBDiv\_R1$ is low and become nearly constant as it becomes higher. The rationale of this heuristic is the following: a low rank of feedback documents (i.e., large $QFBDiv\_R1$) often implies large divergence between query and feedback documents, thus we should take consideration of such a measure when computing the divergence; however, when $QFBDiv\_R1$ is very large (i.e., the feedback documents are ranked very low), the contribution of rank should not be so sensitive to the difference in ranks as when it is small. The heuristic suggests a concave curve for $QFBDiv\_R1$ and the query-feedback divergence as shown in Figure 1. To capture such a heuristic, we propose another measure by taking a *logarithm* transformation on the average rank to approximate the divergence $div(Q, F)$, as follows:

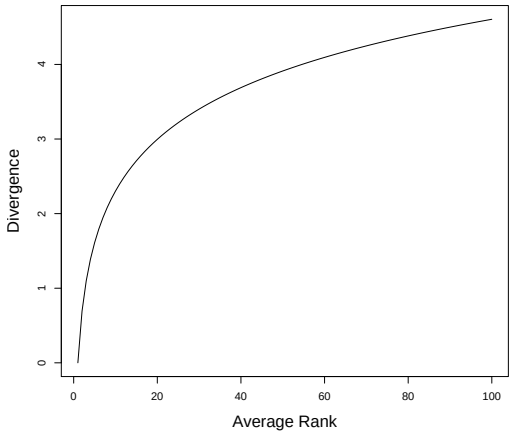

Figure 1: Approximate relation between the query-feedback divergence and the average rank.

$$QFBDiv\_R2 = \log \sum_{d \in F} \frac{r_d}{|F|}$$

3.2 Learning Algorithm

Based on our heuristics and features, we hope to utilize some learning technique to obtain equations which predict feedback coefficients.

We propose to use the logistic regression model [2], which appears to model our problem well: it can take any value from negative infinity to positive infinity as an input, whereas the output is confined to values between 0 and 1.

Logistic regression models are also called maximum entropy models in some communities. In particular, logistic regression models are of the form:

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

where the variable z represents the exposure to some set of features, while $f(z)$ represents the probability of a particular outcome, given that set of features. The 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 indicate 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.

Typically, we learn these weights using training data. For our problem, the training data would consist of feature values along with the corresponding optimal feedback coefficient. To construct such a training data set, we exhaust the

feedback coefficient space for each query-feedback pair to find its optimal coefficient (more details are given in Section 5), where each query-feedback pair together with its optimal coefficient form our training data. 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 have been derived for a particular data set (training data), these weights may be used to predict feedback coefficients for new queries.

4. FEEDBACK COEFFICIENT SMOOTHING

One advantage of our adaptive feedback algorithm is that we can naturally incorporate many features as evidence to improve our estimation of the feedback coefficient. However, through preliminary experiments, we observe that, although a fixed coefficient may not be optimal for many queries, it provides a "safe" coefficient range; compared with it, our predicted dynamic value, though optimized based on many features, is sometimes too extreme and thus "risky". To address this problem, we experimented with some strategies to smooth our prediction using the safe fixed coefficient value. We hypothesize that, with smoothing, our adaptive relevance feedback method would be more robust.

Formally, let α_f be the fixed feedback coefficient and α_d be the dynamically predicted coefficient. Our task here is to estimate a more robust feedback coefficient, which we denote by α_c, obtained by smoothing α_d using α_f. We now describe several different smoothing strategies.

(1) Linear Interpolation: Our first idea is to linearly interpolate the two feedback coefficients (i.e., α_f and α_d) to obtain the final coefficient α_c, which is defined below.

$$\alpha_c = (1-\beta)\alpha_f + \beta\alpha_d$$

where $\beta \in [0,1]$ is a parameter to control the weight on each coefficient. If $\beta = 0$, α_c simplifies to α_f; If $\beta = 1$, α_c simplifies to α_d; otherwise α_c is between α_f and α_d. In our experiments, β is experimentally set to 0.5.

(2) Range Normalization: Suppose there is a safe range for feedback coefficient $[\alpha_f - \delta_l, \alpha_f + \delta_r]$, and we would like to restrict α_c in the safe range. If we have some prior knowledge about δ_l and δ_r, we can obtain the safe range easily. However, most of the time, we do not have such knowledge and have to approximate δ_l and δ_r. In this work, they are empirically approximated as: $\delta_l = \gamma(\alpha_f - 0)$ and $\delta_r = \gamma(1-\alpha_f)$, where we use a γ to indicate the breadth of safe range. Based on these assumptions, we propose to use the following formula to obtain the final feedback coefficient.

$$\alpha_c = 2\delta\alpha_d + \alpha_f - \delta$$

where, if $\alpha_d < \alpha_f$, then $\delta = \delta_l$; otherwise, $\delta = \delta_r$. And it is clear that, α_c is restricted to $[\alpha_f - \delta_l, \alpha_f + \delta_r]$. In our experiments, γ is experimentally set to 0.5.

(3) Pivoted Interpolation: Intuitively, if the feedback coefficient is not very large, the performance of relevance feedback will be at least as good as that of the original query; however, if we use a large coefficient, we have a relative higher possibility to hurt the retrieval performance. To strike a balance between exploration and exploitation, we suggest a preservative strategy to take advantage of the predicted coefficient but at the same time not to be involved in too much risk. Specifically, if $\alpha_d < \alpha_f$, we use α_d as the feedback coefficient; otherwise, we use α_f. Formally, this *Pivoted Interpolation* is defined as follows.

$$\alpha_c = (1-\beta)\alpha_f + \beta\alpha_d$$

where if $\alpha_d < \alpha_f$, $\beta = 1$; otherwise $\beta = 0$.

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

	A	B	C	D	E
Retrieval Model	LM + Dirichlet ($\mu = 1500$)				
FB Model	–	Relevance Model 2			
FB Term Count	–	30			50
Docs per Query	2500				
Stopword	No	319 common words			

Table 1: Parameters in Constructing Workingset. The row "Docs per Query" means that we use top 2500 documents of each query to construct working sets

Smoothing	Mixture Noise	FB Term Count	others
$\mu = 1500$	0.9	100	default

Table 2: Parameters in Relevance Feedback

5. EXPERIMENT RESULTS

5.1 Data Preprocessing

We employ the Lemur toolkit (version 4.5) and Indri search engine (version 2.5) [2] in our experiments. Below, we describe how we pre-process the data collection and how to prepare the training data.

First, due to the large size of the GOV2 data collection, we decide to do the retrieval experiments on a subset of the collection, instead of the whole 426G data. To make the retrieval experiments on our *working sets* equally to that on the whole data set, we use Indri to construct 5 working sets, each for one task. Specifically, we first build an index on the whole Gov2 collection, then we retrieve result documents for each query (for task B-E, we do relevance feedback based on the corresponding positive judgments), and finally these result documents are extracted to construct our working sets. The related parameters we adopt to construct working sets is shown in Table 1, other parameters are the same as Indri's default setting.

Furthermore, we pre-compute a global collection model θ_C using the whole data set, and in the following experiments, whenever we need to access a local collection language model (i.e., language model of a specified working set) to smooth document models, we just use θ_C instead. To very if our working sets work appropriately, we try some retrieval experiments on them and observe that the retrieval performance is almost the same to that on the whole data set.

After constructing working sets, we build a separate index for each working set. Throughout this paper, when building any index, we only stem words using the Porter algorithm, without any other preprocessing.

To train our adaptive relevance feedback model, we need some training data. In our study, we use the Terabyte topics (701-850, excluding those included in this year's test set) as the training queries. There are 100 topics in total, of which

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

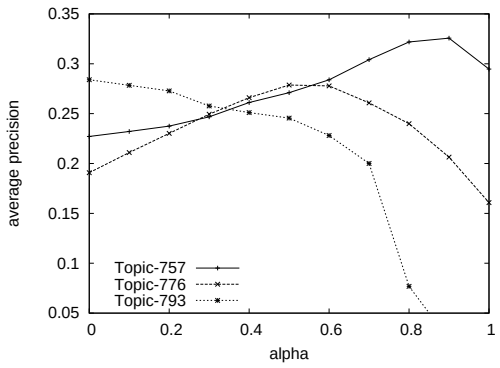

Figure 2: The sensitivity to feedback coefficient of some Terabyte query topics.

one topic has no relevant documents and another fails to return any relevant documents. So, finally, we adopt 98 topics as training data. Then, for each query, we randomly select some top relevant documents to simulate "judgments". Specifically, for each query, with a probability of 0.3 we will select only 1 relevant document for feedback, and with probabilities of 0.4, 0.1, 0.1 and 0.1 we will select 3, 4, 5 and 6 relevant documents respectively. The distribution over these different numbers of relevant documents is heuristically fixed to $\{0.3, 0.4, 0.1, 0.1, 0.1\}$ to approximate the numbers of relevant documents used for feedback in the official tasks B, C, and D.

After that, we also construct a working set for training data using the same parameter setting as task C. Finally, with Lemur toolkit, we adopt the KL-Divergence retrieval model with mixture model feedback to do relevance feedback experiments (related parameters are shown in Table 2); through trying different feedback coefficients (0.0, 0.1, ..., 1.0), we get the optimal coefficient value for each query. The 4th column of Table 3 gives some examples of the optimal coefficients. We use this data set to learn the prediction model.

5.2 Sensitivity of Feedback Coefficient

As we have discussed in Section 2, relevance feedback is controlled by a coefficient α. When $\alpha = 0$, we are only using the original query model (i.e., no feedback), while if $\alpha = 1$, we ignore completely the original query and rely only on the feedback model. To show the sensitivity of α, we plot the MAP of some queries (Terabyte topics 757, 776, and 793) in relevance feedback experiments by varying α from 0 to 1. The results are shown in Figure 2. We can observe that the setting of feedback coefficient α can affect the performance significantly, and the optimal coefficients for different queries could be quite different.

5.3 Preliminary Experiment

We find that Relative Query Entropy $QEnt\_R1$, Relative Entropy of Feedback Document $FBEnt\_R$, Absolute Divergence $QFBDiv\_A$, and Relative Divergence $QFBDiv\_R2$ have the most significant influence on the ability to predict accurate feedback coefficients. Thus, we keep the above four features in our final prediction model, and the following coefficients are then derived from our training algorithm.

	Adaptive	Fixed	Optimal
785	**0.3525**	0.6	0.4
787	**0.4629**	0.6	0.2
789	0.4654	**0.6**	0.7
791	0.6799	**0.6**	0.3
793	0.6006	**0.6**	0.0
795	**0.8717**	0.6	0.9
797	**0.4897**	0.6	0.4
799	0.3533	**0.6**	0.5
801	0.5830	**0.6**	0.8
805	**0.7812**	0.6	1.0
807	**0.5145**	0.6	0.5
809	**0.4262**	0.6	0.5
810	**0.7063**	0.6	0.8
811	**0.5002**	0.6	0.5
813	**0.5616**	0.6	0.5
815	**0.6266**	0.6	0.8
816	**0.4972**	0.6	0.1
817	0.3873	**0.6**	0.7
819	**0.4527**	0.6	0.2
821	**0.4929**	0.6	0.5

Table 3: Samples of Prediction Values

$$\begin{aligned} z_0 = & - 0.93265 + 0.09890 * QEnt\_R1 \\ & - 1.45937 * FBEnt\_R + 0.28350 * QFBDiv\_A \\ & + 0.32427 * QFBDiv\_R2 \end{aligned}$$

Where we use the absolute value of each feature. Then, we can predict feedback directly as below:

$$\alpha = f(z_0) = \frac{1}{1 + \exp(-z_0)}$$

From the above formulas, we can see clearly that $QFBDiv\_A$ and $QFBDiv\_R2$ play the similar roles as we discussed in Section 3. However, Relative Query Entropy $QEnt\_R1$ increases the feedback coefficient, which means that, if a query is more discriminative, we can use a higher feedback coefficient. It is in contrast to our initial expectation. One explanation is that, a more discriminative query (i.e., with a high clarity score) is more drifting-tolerant, and thus it is safe to use a large feedback coefficient in this case. Also $FBEnt\_R$ is negatively correlated to the feedback coefficient α, and it is also in contrast to our intuition. One possible explanation is that, we do not need a large feedback coefficient if the feedback is too discriminative, since a discriminative feedback can easily drift the original query away.

With the prediction formula, we can compute potentially different feedback coefficients for different queries. Note that, the four features are all computed efficiently, because we only use the Maximum Likelihood method to estimate related language models.

We evaluate our method on the training data by using 10-fold cross validation. Some examples of our prediction values are shown in the second column of Table 3.

We compare our adaptive feedback method with our baseline system, i.e., the fixed feedback coefficient approach (where we use a fixed coefficient 0.6, since it brings the best performance). Before evaluation, the judged documents are removed from the results, and the result document list is then restricted to only contain the top 1000 documents. Besides, we also compute the Mean Absolute Error (MAError) to indicate how far off the coefficients used in the two

	MAError	MAP	Recall
Fixed	0.2173	0.3243	12070/18649
Adaptive	0.1824	0.3341*	12365/18649
Improvement	16.1%	3.0%	2.4%
upper-bound	0	0.3553	12696/18649

Table 4: Performance Comparison on Training Data

	MAError	MAP	Recall
All Features	0.1824	**0.3341**	12365/18649
No $FBEnt\_R$	0.1879	0.3320	12308/18649
No $QEnt\_R1$	0.1905	0.3294	12295/18649
No $QFBDiv\_R2$	0.1950	0.3307	12409/18649
No $QFBDiv\_A$	0.1842	0.3330	12316/18649

Table 5: Contributions of Features on Training Data. "No XXX" means removing feature XXX.

methods and the optimal coefficients. The comparison of performances is shown in Table 4, which indicates that our approach outperforms the fixed coefficient approach clearly.

Next, we perform several experiments to show the contributions of individual features. Table 5 shows the results, where every time one feature is removed singly. Below we give the derived formula to predict coefficient without $QEnt\_R1$ as an example. From Table 5, we can see that each feature plays an important role.

$$\begin{aligned} z_1 = \; & + \; 0.01156 - 0.77456 * FBEnt\_R \\ & + \; 0.20068 * QFBDiv\_A \\ & + \; 0.35399 * QFBDiv\_R2 \end{aligned}$$

Also, we design some experiments to evaluate the proposed smoothing methods (Section 4). The results are shown in Table 6. It indicates that Range Normalization (Norm) $\geq$ Linear Interpolation (Linear) $\geq$ Pivot Interpolation (Pivot), however, no smoothing method outperforms our basic adaptive feedback method. Maybe it is because we did not tune the parameters of these smoothing methods, which will be further studied in the future work.

5.4 Official Run Results

We submitted two runs for each task, in which various techniques we designed are applied. These runs are described in Table 7. UIUC.B1, UIUC.C1, UIUC.D1 and UIUC.E1 use the proposed adaptive relevance feedback plus the range normalization smoothing method; UIUC.B2 and UIUC.C2 also adopt our adaptive feedback but respectively use pivot and linear interpolation as the smoothing method; UIUC.D2 only tests the adaptive feedback without any smoothing method; in UIUC.E2, we add a pseudo relevance feedback on top of the adaptive relevance feedback. Table 8 shows the performance of these official runs.

After our official runs were submitted we discovered that

	NoSmooth	Linear	Norm	Pivot
All Features	**0.3341**	0.3316	0.3323	0.3265
No FBEnt_R	0.3320	0.3301	0.3308	0.3263
No QEnt_R1	0.3294	0.3295	0.3304	0.3244
No QFBDiv_R2	0.3307	0.3294	0.3299	0.3254
No QFBDiv_A	0.3330	0.3308	0.3310	0.3262

Table 6: Comparison of Smoothing Methods on Training Data

RunID	Description
UIUC.A1	No Relevance Feedback
UIUC.B1	Adaptive + Norm
UIUC.B2	Adaptive + Pivot
UIUC.C1	Adaptive + Norm
UIUC.C2	Adaptive + Linear
UIUC.D1	Adaptive + Norm
UIUC.D2	Adaptive
UIUC.E1	Adaptive + Norm
UIUC.E2	Adaptive + PseudoFB

Table 7: Description of Runs

RunID	MAP	MTC	StatAP
UIUC.A1	0.1240	0.0460	0.2127
UIUC.B1	0.1868	0.0650	0.2886
UIUC.B2	0.1770	0.0641	0.2833
UIUC.C1	0.1971	0.0714	0.3192
UIUC.C2	0.1971	0.0714	0.3192
UIUC.D1	0.2078	0.0722	0.3397
UIUC.D2	0.2079	0.0712	0.3327
UIUC.E1	0.2118	0.0673	0.3284
UIUC.E2	0.1744	0.0583	0.2681

Table 8: Official Results of Runs

our implementation of the range normalization was not quite accurate and we had left out the query feedback documents divergence feature $QFBDiv\_A$. So, we decided to re-compute our runs. Note that we did not change anything related to our algorithm but just the implementation. We also generated some additional runs to compare different techniques we proposed. Table 9 and 10 show the performance of these runs.

Comparing to our preliminary experimental results, there are two significant changes: feature *QFBDiv_R2* hurts the performance; our smoothing methods, especially the *Range Normalization*, improve the performance. From Table 9, we can see that *"No QFBDiv_R2"* outperforms *"All Features"* all the time, and that *"No QFBDiv_R2"* always beats the baseline system; with Range Normalization, the performances of all methods are improved.

One possible explanation of the two significant changes is that, the training data and the testing data are quite inconsistent. In our training data, we use top relevant documents to simulate judged documents, because in real world, users would like to judge top documents; however, in the testing data, the judged documents are often ranked very low (or even do not occur in the top 2500 result documents). Our feature *QFBDiv_R2* measures the *rank* of feedback documents and thus is very sensitive across two data sets. However, on the other hand, it also shows that our prediction

Task B				
	NoSmooth	Linear	Norm	Pivot
Baseline	0.1889	–	–	–
UIUC.B1	–	–	0.1868	–
UIUC.B2	–	–	–	0.1770
All Features	0.1781	0.1870	0.1892	0.1771
No FBEnt_R	0.1719	0.1837	0.1874	0.1730
No QEnt_R1	0.1718	0.1820	0.1847	0.1724
No QFBDiv_R2	0.1892	0.1915	**0.1948**	0.1826
No QFBDiv_A	0.1760	0.1862	0.1891	0.1753

Table 9: Performance Comparison on Task-B

Task C				
	NoSmooth	Linear	Norm	Pivot
Baseline	0.1948	–	–	–
UIUC.C1	–	–	0.1971	–
UIUC.C2	–	0.1971	–	–
All Features	0.1919	0.1955	0.1973	0.1904
No QFBDiv_R2	0.1968	0.1976	**0.1994**	0.1893
Task D				
Baseline	0.2057	–	–	–
UIUC.D1	–	–	0.2078	–
UIUC.D2	0.2079	–	–	–
All Features	0.2071	0.2090	0.2100	0.2039
No QFBDiv_R2	0.2075	0.2073	**0.2118**	0.2006
Task E				
Baseline	0.2182	–	–	–
UIUC.E1	–	–	0.2118	–
All Features	0.1797	0.2123	0.2110	0.2182
No QFBDiv_R2	0.2192	**0.2193**	0.2184	0.2182

Table 10: Performance on Task C, D and E

model without *QFBDiv_R2* is very robust, which even works well on such a different data set; although the parameter of our Range Normalization smoothing method is not tuned, it still helps a lot when testing and training data are not consistent.

Another reason to explain the changes may be the sparseness of our training data. We only utilize 98 training queries to train 4 features, which have already led to a robust relevance feedback method. It would be interesting to see whether a large number of queries would lead to a more effective logistic regression approach to predict feedback coefficients.

6. CONCLUSIONS

In summary, we studied a novel problem in feedback, i.e., optimization of the balance of the query and feedback information, in this year's relevance feedback task, and proposed an adaptive relevance feedback approach to dynamically predict feedback coefficient. Our experiment results show that the our proposed method is robust and effective, which outperforms the fixed-coefficient relevance feedback, even when training and testing data sets are not consistent.

Besides, we also designed three smoothing strategies to smooth our predicted coefficients to make them more robust. Among the three smoothing methods, Range Normalization is the most effective one; smoothing our prediction value can make it more robust, especially when training and testing data sets are inconsistent.

Among our features, we find that Relative Query Entropy $QEnt\_R1$, Relative Entropy of Feedback Document $FBEnt\_R$, Absolute Divergence $QFBDiv\_A$, and Relative Divergence $QFBDiv\_R2$ are the most effective ones. However, $QFBDiv\_R2$ is very sensitive to the data set and should be used carefully.

There is still much room to explore in the future work. We should study more effective and robust features in the future. And also, we hope to apply our method to other feedback models, e.g. Rocchio Feedback, to show its performance. In addition, it will be interesting to explore how to adaptively predict feedback coefficients in pseudo and implicit relevance feedback models.

7. ACKNOWLEDGMENT

This material is based upon work supported by the National Science Foundation under Grant Numbers IIS-0347933, IIS-0713581, and FIBR-0425852.

8. REFERENCES

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

[2] 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.

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

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

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

[6] Stephen E. Robertson and Karen Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for 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 for Information Science*, 41(4):288–297, 1990.

[9] 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.

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