自适应信息过滤中使用少量正例进行阈值优化*

夏迎炬+, 黄萱菁, 胡 恬, 吴立德

(复旦大学 计算机科学系,上海 200433)

Threshold Optimization with a Small Number of Samples in Adaptive Information Filtering

XIA Ying-Ju+, HUANG Xuan-Jing, HU Tian, WU Li-De

(Department of Computer Science, Fudan University, Shanghai 200433, China)

+ Corresponding author: E-mail: yingjuxia@yahoo.com.cn; {ldwu,xjhuang}@fudan.edu.cn

http://www.cs.fudan.edu.cn

Received 2002-06-01; Accepted 2002-09-04

Xia YJ, Huang XJ, Hu T, Wu LD. Threshold optimization with a small number of samples in adaptive information filtering. *Journal of Software*, 2003,14(10):1697~1705.

http://www.jos.org.cn/1000-9825/14/1697.htm

Abstract: One special challenge in adaptive information filtering is the problem of extremely sparse data. So it is very important to learn optimal threshold while filtering the input textual stream. In this paper, an algorithm is presented for the threshold optimization. The algorithm learns fast by using few positive samples. Moreover, most of the quantities the algorithm requires can be updated incrementally, so its memory and computational power requirements are low. It also has the merits of effective, robust, and practically useful. Fudan University's adaptive text filtering system used this algorithm for the first time and came in third in all runs of TREC10. Its T10U and T10F are 0.215 and 0.414 respectively.

Key words: adaptive information filtering; vector space model; threshold optimization; delivery ratio; relevance feedback

摘 要: 自适应信息过滤中一个大的挑战在于其数据稀疏问题.因此,在对输入的文本流进行过滤的同时学习最优阈值非常重要.提出了一种新颖的阈值优化算法.该算法可以通过少量的正例进行快速的学习,所需数据的获得具有增量性,故而其计算量及所需的存储空间很小.此外,该算法还具有高效、健壮、实用性强等优点.在第

* Supported by the Supported by the National Natural Science Foundation of China under Grant Nos.69873011, 69935010, 60103014 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant Nos.2002AA142090, 2001AA114120 (国家高技术研究发展计划(863))

XIA Ying-Ju was born in 1972. He is a Ph.D. candidate at the Dept. of Computer Science, Fudan University. His current research interests include the information filtering and the information retrieval. **HUANG Xuan-Jing** was born in 1972. She is an associate professor at the Dept. of Computer Science, Fudan University. Her current research interests include the information filtering, the information retrieval and the text categorization. **HU Tian** was born in 1978. He is a master student at the Dept. of Computer Science, Fudan University. His current research interests include the information filtering and the information retrieval. **WU Li-De** was born in 1937. He is a professor and doctoral supervisor at the Dept. of Computer Science, Fudan University. His current research interests include the natural language processing and the image processing.

10 届国际文本检索会议(TREC10)上,复旦大学的自适应信息过滤系统使用了该阈值优化算法,并取得了第 3 名的成绩.其 T10U 和 T10F 分别达到了 0.215 和 0.414.

关键词: 自适应信息过滤;向量空间模型;阈值优化;检出率;相关反馈
中图法分类号: TP391 文献标识码: A

1 Introduction

In an adaptive information filtering environment, a system starts with only a user profile and a very small number of positive examples (relevant documents); it must begin filtering documents without any other prior information; each retrieved document is immediately judged for relevance, and this information can be used by the system to adaptively update the filtering profile and threshold.

Generally speaking, there are three basic problems in adaptive information filtering: initialization, decision and learning. The initialization means building a filtering profile and setting an initial threshold based on user profile and a very small number of positive examples. The decision means making the decision to accept or reject a document according to its relevance score and the threshold. Learning includes learning filtering profile and dissemination threshold from relevance judgments during filtering.

For the extremely spare data, machine learning plays an important role in adaptive information filtering. Most the prior research on adaptive information filtering, however, has been focused on learning filtering profile. Among the filtering systems based on vector space model[1], the common approach to learning filtering profiles is to use an incremental version of the Rocchio algorithm[2~5]. So to speak, profile learning has been well motivated. But threshold learning received little attention until recently. It is in part because that when dissemination could be delayed, methods that sort documents by their sores and disseminate the top N document are more effective than methods using threshold. But in a practical environment, the adaptive information filtering system can't delay the dissemination decisions; it is necessary to find a dissemination threshold for each profile.

Intuitively, the greater the amount of relevant judgments, the better the performance of the system. Buckley et al. experimentally verified that the recall-precision effectiveness is roughly proportional to the log of the number of relevant documents[6]. However, it is not feasible in practice to have so many relevance judgments as provided by TREC10's[7] adaptive filtering task, which the number of average relevant documents per topic is about 9796. On the other hand, Sebastinai[8] has shown that too many positive documents can cause over-fitting. This brings up the question: can we get high system performance by using as few positive samples as possible? Using few positive samples also has special meanings in practice since we can only get a small number of positive samples in most cases.

In this paper, we focus on the profile-modifying algorithm using a small number of positive samples. Section 2 describes the role of threshold optimization in adaptive information filtering and methods used in threshold optimization. Section 3 summarizes the framework of our adaptive filtering system that has participated in the TREC10's adaptive filtering task. Section 4 describes the threshold-adjusting algorithm used in our adaptive information filtering system. Finally, Section 5 and Section 6 give the experiment results and some conclusions from them.

2 Threshold Optimization in Adaptive Information Filtering

The problem of threshold optimization can be described as finding a threshold that maximizes a utility metric in which each relevant document is associated with some reward and each irrelevant document is associated with some penalty. Most of the recent research on threshold optimization is based on either heuristics or regression

methods[2,3,5,9~12]. For example, Fudan University's adaptive filtering system used precession and delivery ratio to set the optimal threshold[9]. CLARITECH used a heuristic that allowed the threshold to vary between a lower bound utility value (θ_{zero}) and an upper bound optimal value (θ_{opt})[2]. Microsoft Cambridge used logistic regression to set the optimal threshold[10]. KUN assumed a Gaussian distribution for the scores of relevant documents and an exponential distribution for the scores of irrelevant documents, and then found the threshold that maximized a given linear utility measure. KUN's approach was very effective and achieved the best result for utility oriented runs in the TREC9 filtering track evaluation[5].

One potential weakness with the KUN approach, as Zhang[11] has pointed out, is that it assumes that the training data accurately represents the distribution of relevant and irrelevant document scores. This assumption is not true in an adaptive filtering environment, because relevance information is obtained only for documents that are actually disseminated. Relevance information is not available for documents that have scores below the threshold, so the training data is inherently biased. Based on the same assumption on the distribution of training data as KUN, Zhang[11] has presented a new algorithm that explicitly models the sampling bias, and uses the maximum likelihood principle to find unbiased estimates of the parameters. It jointly estimates the parameter of the two density distributions and the ratio of the relevant documents in the corpus. Experiments have been done to show this algorithm's effectiveness.

The MLE (maximum likelihood estimation) method presented by Zhang[11], however, has many drawbacks. For example, the algorithm introduced a minimum delivery ratio to avoid disseminate too few documents. It assumed that the distribution of relevant documents in the textual stream is uniform (or their relative density is approximately constant). However, in general it is a false assumption. Take TREC10's test data for example, the number of each topic's relevant documents ranges from 38 to 39448 (Fig.1). Further more, even the average of relevant documents can't be known in advantage. Another drawback of MLE algorithm is that it is not sensitive to the threshold adjusting effect of each adjusting interval. So the adjusting may be delayed and in some extreme conditions, may be disastrous. This may be the reason why they introduce the minimum delivery ratio. But the delivery ratio can work well only in the environment that the distribution of relevant documents in the textual stream is approximately uniform. Moreover the resulting equation does not have analytical solutions; therefore it has to be solved numerically, which causes the computation of this algorithm a little heavy.

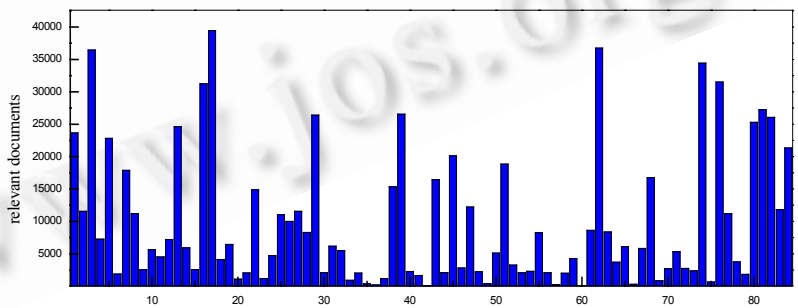

Fig.1 Number of relevant documents of TREC10' topics

In Section 4, we will present our threshold optimization algorithm, which discards the parameter of delivery ratio and consider the interval adjusting effective. Our method uses a heuristic and doesn't assume any distribution of the training data. Most of the quantities the algorithm requires can be updated incrementally, so its memory and computational power requirements are low. Moreover, our algorithm learns fast using few positive samples, which is every important when transferred into commercial products. Since the threshold optimization algorithm is

attached to whole adaptive information filtering system and has a close relationship with the profile learning. In the following sections, we first introduce our adaptive information system briefly and then describe our threshold optimization algorithm in details.

3 Adaptive Information Filtering system Architecture

Our adaptive filtering system consists of two components: training and adaptive filtering. The task of training stage is to get the initial filtering profile and set the initial threshold. In adaptive filtering, the main task is to modify the profile and threshold adaptively. We adopt the vector space model to present topics and documents, so each topic and document is transferred into feature vector.

3.1 Training architecture

Figure2 shows the architecture of the training stage. At first, feature vectors are extracted from positive and pseudo-positive documents. The pseudo-positive documents are those that have high similarity with the topic but haven't been labeled as positive documents in the training set. The pseudo-positive documents can be gotten by several ways. In TREC9, We got pseudo-positive documents by pseudo feedback[9]. In TREC10, since the corpus is the "Reuters Corpus, Volume 1, English language, 1996-08-20 to 1997-08-19" and the hierarchy of the Reuters category codes is also provided[7], we got the pseudo-positive documents by using the hierarchy of categories: a topic's pseudo-positive documents are those that have the same high-level categories provided by the training set.

To get the feature vectors, we first remove stop-words and do morphological analysis on remaining words; then, we compute the logarithm mutual information between words and topics[13]; the formula is shown in Eq.(3.1).

$$\log MI(w_i, T_j) = \log\left(P(w_i \mid T_j) \Big/ P(w_i)\right) \tag{3.1}$$

where w_i is the *i-th* word and T_j is the *j-th* topic. Higher logarithm Mutual Information means w_i and T_j are more relevant. $P(w_i|T_j)$ and $P(w_i)$ are estimated by maximal likelihood method.

For each topic, we select those words with logarithm Mutual Information higher than 3.0 and occuring more than once in the relevant documents. Logarithm Mutual Information is not only used as the selection criterion, but also as the weight of feature words.

After gotten the feature vectors of positive and pseudo-positive documents, we merged them into the initial profile. The initial profile is the weighted sum of positive and pseudo-positive feature vectors. Then we should set initial threshold for each topic. The initial threshold is set based on the similarity of each document in the training set. The similarity between a profile and training document is computed by the cosine formula[1]:

$$Sim(d_i, p_j) = \mathrm{Cos}\theta = \sum_k d_{ik} \bullet p_{jk} \Big/ \sqrt{(\sum_k d_{ik}^2)(\sum_k p_{jk}^2)} \tag{3.2}$$

where p_j is the profile vector of the *j-th* topic and d_i is the vector representation of the *i-th* document. d_{ik} is the weight of the *k-th* word in d_i. *dik* is computed as Eq.(3.3).

$$d_{ik} = 1 + \log(tf_{ik} \bullet avdl / dl) \tag{3.3}$$

where tf_{ik} is the term frequency of the *k-th* word in the *i-th* document, the *dl* is the document length counted by tokens in the document after morphological processing and stop-words removal, the *avdl* is the average document length gotten from training set. According to the similarities of training documents, each initial threshold is set to get the best system performance.

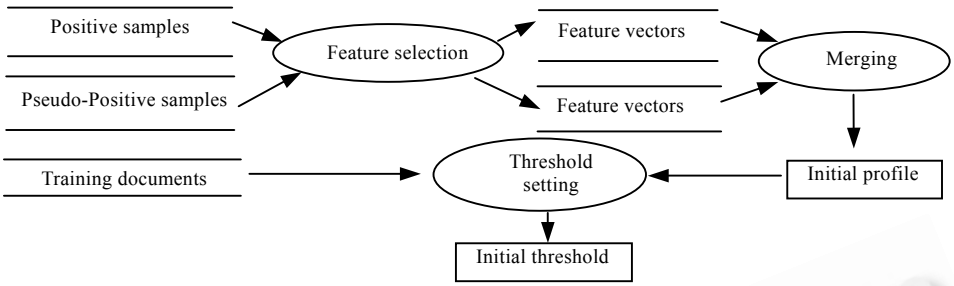

Fig.2 Architecture of the training stage

3.2 Adaptive filtering architecture

Adaptive filtering is very important stage in adaptive information filtering system. The system begins with the initial profile and threshold gotten from training stage. While filtering the input documents, the system updates the topic profile and threshold adaptively by using all kinds of information the system can get, such as the user's feedback, vectors of input documents.

Figure 3 shows the architecture of adaptive filtering. When a document arrives, system calculates its similarity with the topic profile. If its similarity is higher than the current threshold, then the system retrieves this document and gets the user's relevance judgment. If the document is really relevant to the topic, it will be considered as positive sample, otherwise negative sample. The vectors of positive and negative samples will be used to modify the topic's profile, which is shown in Equation 3.4.

$$p'_j = p_j + \alpha \bullet p_j(pos) + \beta \bullet p_j(neg) \tag{3.4}$$

where p'_j is the topic's profile after modification; p_j is the topic's profile before modification; $p_j(pos)$ is vector of positive samples gotten at this updating interval while $p_j(neg)$ is vector of negative samples; α and β are the weight of positive and negative vectors respectively.

Another main step in adaptive filtering is threshold adjusting, which will be discussed in details in Section 4.

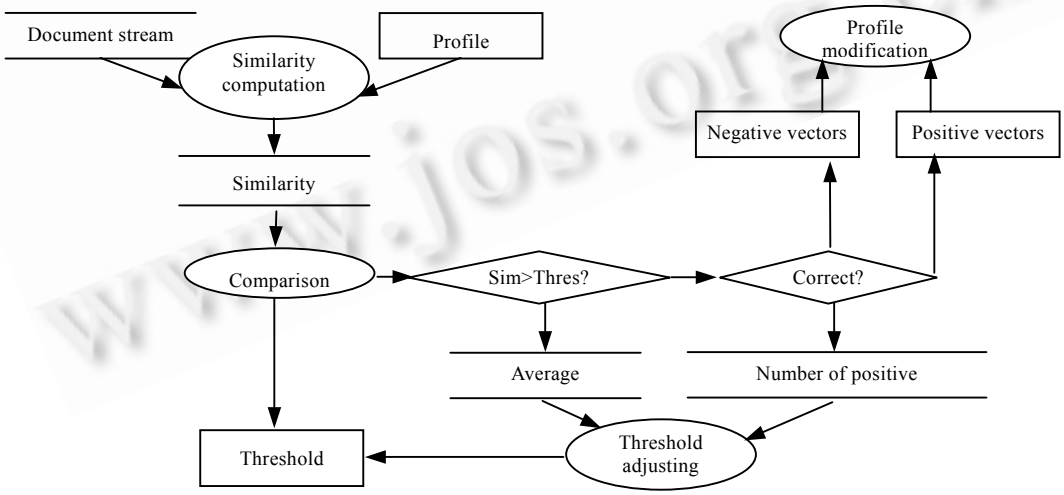

Fig.3 Architecture of the adaptive filtering

4 Threshold Optimization Algorithm

The precise definition for our threshold optimization algorithm uses the following notations:

t: the document's No., since the documents are processed by temporal order, t also can be considered as time.

$n(t)$: the number of documents processed theretofore.

$n_R(t)$: the relevant documents retrieved theretofore.

$n_N(t)$: the irrelevant documents retrieved theretofore

$T(t)$: the threshold at time t

$S^-(t_k,t_{k+1})$: the average similarity of the documents that have been rejected by the system in (t_k,t_{k+1}) interval

$P(t_k,t_{k+1})$: the precision of system in (t_k,t_{k+1}) interval, here

$$P(t_k,t_{k+1}) = \frac{n_R(t_{k+1}) - n_R(t_k)}{n(t_{k+1}) - n(t_k)}$$

Intuitively, we should promote the threshold if the precision is too low and demote the threshold if few documents have been retrieved. We use the $S^-(t_k,t_{k+1})$ and $P(t_k,t_{k+1})$ to decide either promote or demote the threshold. The magnitude of adjusting, namely the adjusting step, should be decided simultaneously. We used a heuristic that allowed the adjusting step to vary between zero and an upper bound adjusting step (the maximum step can be taken). The Architecture of the threshold adjusting is shown on Fig.4 and the algorithm of adjusting threshold is shown below:

IF $p(t_k,t_{k+1}) \le EP(t_{k+1})$

$T(t_{k+1}) = T(t_k) + \alpha(t_{k+1}) \bullet (1 - T(t_k))$

ELSE

IF $S^-(t_k,t_{k+1}) < T(t_k) * r$

$T(t_{k+1}) = T(t_k) \bullet D_1 + S^-(t_k,t_{k+1}) \bullet D_2$

ELSE

$T(t_{k+1}) = (1 - \beta(t_{k+1})) \bullet T(t_k)$

where $\alpha(t_{k+1})$ is the coefficient for promoting the threshold and $\beta(t_{k+1})$ is the coefficient for lowering the threshold, they also can be considered as function of $n_R(t)$. In our experiment, we adopt linear function of $n_R(t)$, which are shown in Eqs.(4.1) and (4.2).

$$\alpha(t_k) = \begin{cases} \alpha_0 \bullet (\mu - n_R(t_k))/\mu & n_R(t_k) \le \mu \\ 0, & n_R(t_k) > \mu \end{cases}, \tag{4.1}$$

$$\beta(t_k) = \begin{cases} \beta_0 \bullet (\mu - n_R(t_k))/\mu & n_R(t_k) \le \mu \\ 0, & n_R(t_k) > \mu \end{cases}, \tag{4.2}$$

here, α_0 is the initial promoting coefficient and β_0 is the initial demoting coefficient. The parameter μ indicates the maximum number of positive documents should be used to adjust the threshold and modify the profile. Here we set α_0=0.02 and μ=0.1 and μ=300. From the Equation, we can see that as time elapsing, $n_R(t_k)$ will grow gradually so the $\alpha(t_k)$ and $\beta(t_k)$ will decrease gradually. This reflects the trend that the system will became better and better and the adjusting step will became smaller and smaller accordingly.

The parameter r indicates that in the condition that $S^-(t_{k+1}-t_k)$ is lower than $T(t_{k+1})*r$, the threshold should be demoted with the coefficient D_1 and D_2. In our experiment, we set r=0.1, D_1=0.8, D_2=0.2.

By '$EP(t_{k+1})$', we mean that the precision which we hope the system should reach at time t_{k+1}. At first, we treat this parameter as constant and try several different values to see the performance of system, but the results are not satisfied. Notice that it is rigorous to hope the system reach the final expectant precision at beginning of filtering;

we adopt a gradual-ascent function. The function is shown in Eq.(4.3).

$$EP(t_{k+1}) = P_0 + (P_{final} - P_0) \bullet n_R(t_{k+1}) \big/ \mu \tag{4.3}$$

where P_0 and P_{final} is the precision we hope the system to reach at beginning and end of the filtering.

The value $n_R(t_{k+1})-n_R(t_k)$ controls the frequency that the system adjusts the threshold. Small value of it means that the system adjusts the threshold more frequently. In our adaptive filtering system, we set $n_R(t_{k+1})-n_R(t_k)$ at 1, which means that the system adjusts the threshold once it retrieves a positive sample.

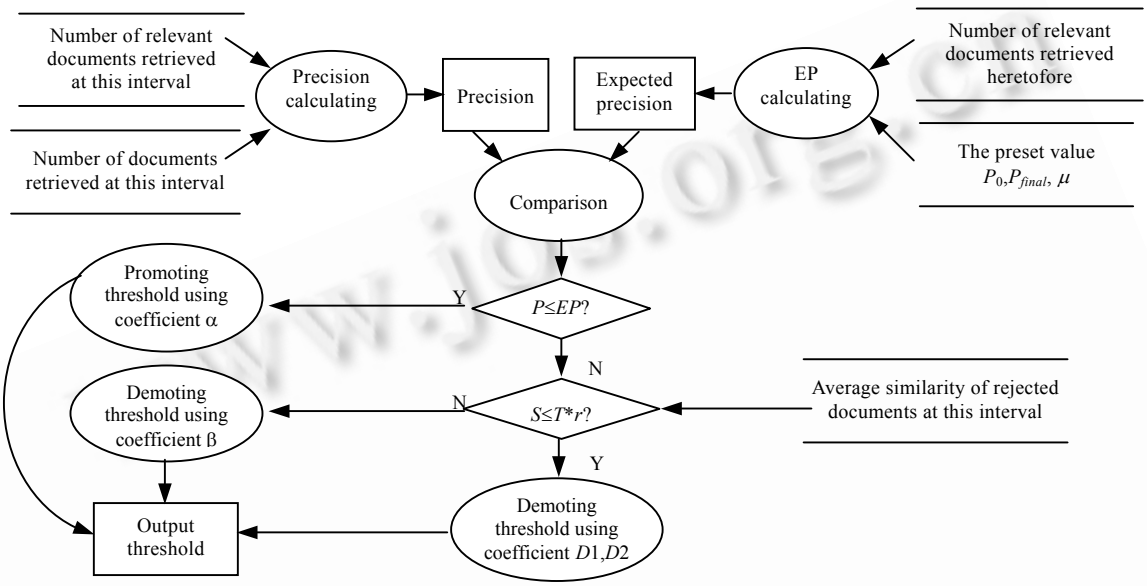

Fig.4 Architecture of the threshold adjusting

In the algorithm, we examine the precision of adjusting interval and the average similarity of rejected documents to decide how to adjust the threshold. The system's performance became better and better while more and more documents have been processed, so the adjusting step descends gradually; and as a result, the threshold approximates the optimal threshold gradually. All the data needed in the algorithm are these values gotten from the previous adjusting interval, such as $n(t)$, $n_R(t)$, $n_N(t)$, $T(t)$, $S^-(t_k,t_{k+1})$ and $P(t_k,t_{k+1})$. System adjusts the threshold according to these data, so it is sensitive to the adjusting interval. Moreover, the $S^-(t_k,t_{k+1})$ can be calculated incrementally, the algorithm's memory and computational power requirements are very low.

5 Experiment Results

The measures used to evaluate the system are T10SU and T10F[7]. The T10SU was a linear utility function that rewarded systems two points for retrieving a relevant document and penalized systems one point for retrieving an irrelevant document. To compute average utility over topics, the utility score for each topic was first scaled between an upper and lower bound (2*number-relevant and -100 respectively). The T10F was a version of van Rijsbergen's F measure with β=0.5[14].

Table 1 shows the experiment results of our adaptive filtering system in the TREC10. Four evaluation criteria are calculated, including T10SU, T10F, Set Precision and Set Recall. Underlined value means that the run is optimized for the corresponding criterion. The last three columns give the number of topics in which our runs perform better, equal and worse than median ones according to the criteria.

Table 1 Adaptive filtering results

Run	T10SU	T10F	Set precision	Set recall	Comparison with median		
					>	=	<
FDUT10AF1	0.215	0.404	0.505	0.330	64	5	15
FDUT10AF4	0.213	0.414	0.493	0.363	71	4	10

To investigate the effect of different μ on the system's performance and running time, we have done a serial of experiments with different μ and recorded T10SU, T10F and running time at each point. Figures 5 and 6 show these results (All the experiments in this paper have been done on a 733 MHz Intel Pentium III processor with 128 MB of memory). From Fig.5, we can see that the system's T10SU and T10F rose rapidly while μ<40; under the condition that 40<μ<100, the two measure still rising but slowly; when 100<μ<300, they rising more slowly. The system performance drops a little while μ>300. Figure 6 shows system's running time at each μ. As the μ increases, running time increases rapidly and becomes intolerable while μ>1000. Figures 5 and 6 also show that the system approaches the optimal performance whileμ is about 100 and uses much less run time.

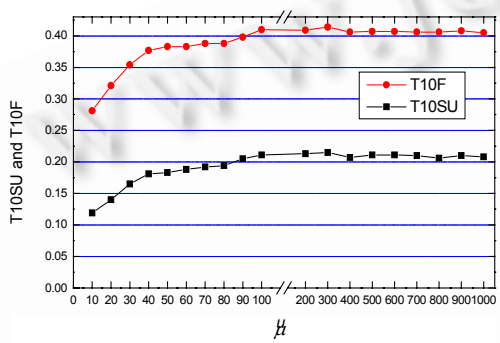

Fig.5 Measures vs. μ

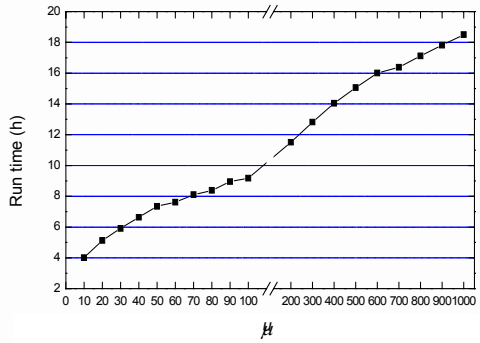

Fig.6 Run time vs. μ

Experiments have also been done to compare the effectiveness of different methods; the results are shown in Table2.

Table 2 Experiment results of different methods

Run	T10SU	T10F	Set precision	Set recall
Non-Adjusting	0.066	0.165	0.610	0.071
MLE method	0.143	0.336	0.493	0.386
μ=30	0.162	0.339	0.511	0.217
μ=40	0.182	0.365	0.481	0.271
μ=50	0.184	0.365	0.456	0.304
μ=60	0.184	0.370	0.501	0.250
μ=100	0.203	0.386	0.456	0.348
μ=200	0.213	0.409	0.488	0.354

Where 'Non-Adjusting' is the method that never updates the threshold at all, MLE method is the method discussed in Section 2. The remains are the results of our threshold optimization algorithm with different μ. From Table 2, we can see that the effect of threshold optimization is significant. Threshold learning actually compensate for the initial inaccuracies. Without threshold learning, the initial threshold settings may lead to disastrous performance. Also, we can see from Table2 that our threshold optimization method outperforms the MLE method at μ=30. With biggerμ, our method performances better. This is partly because that the MLE method uses the delivery ratio which can't reflect the real relevant documents distribution of the TREC10's corpus (see Fig.1). Under the

condition that μ=30, it took our method about 6 hours to filter all the documents in test collection while the MLE method about 16 hours. So our method also outperforms the MLE on computation time. Attention should be paid to that due to too many factors (e.g., scoring), it is hard to compare the different threshold learning methods. The performance of each threshold learning method will vary with different environments. Here we compare our threshold optimization algorithm with modified version of the threshold-optimization algorithm that achieved the best result for utility oriented runs in the TREC9 Filtering Track evaluation is merely want to inspect our method's effectiveness.

6 Conclusions and Future Work

Threshold adjusting is every important in adaptive filtering system. In this paper, we present a novel threshold optimization algorithm used in our adaptive information filtering system. This algorithm can achieve high system performance by using few positive samples. The average relevant documents of each topic in TREC10's test corpus are about 9796. We used less than 1/30 of the relevant documents and got high score. Another feature of this algorithm is fast adjusting. The system can use even less positive samples with a little drop in effectiveness and require much less running time. These are every important when transfer this technology into commercial products for that there are usually few samples that users can provide and the maximum run time users can tolerate is limited.

Our algorithm only uses the precision of each adjusting interval and the average similarity of the rejected documents (which can be gotten accurately) to do the optimization. Since the measure T10F is tradeoff between precision and recall and the T10SU also has a close relationship with precision, in a sense, optimizing precision means optimizing T10F and T10SU. So this algorithm can be used to optimize T10F and T10SU indirectly and experiments have also shown its effectiveness. In the future, we will focus on the problem of how to adjust threshold with different utility functions and directly.

References:

[1] Salton G. Developments in automatic text retrieval. Science, 1991,253:974~979.
[2] Zhai C, Jansen P, Roma N, Stoica E, Evans DA. Optimization in CLARIT adaptive filtering. In: Voorhees EM, Harman DK, eds. Proceedings of the 8th Text Retrieval Conference. 1999. 253~258.
[3] Zhang Y, Callan J. Yfilter at TREC9. In: Voorhees EM, Harman DK, eds. Proceedings of the 9th Text Retrieval Conference. Gaithersburg. 2000. 154~161.
[4] Allan J. Incremental relevance feedback for information filtering. In: Frei HP, Harman D, Schäuble P, Wilkinson R, eds. Proceedings of the 19th annual international ACM SIGIR conference on Research and Development in Information Retrieval 1996. Zurich, Switzerland, 1996. 270~278.
[5] Arampatzis A, Beney J, Koster CHA, van der Weide TP. KUN on the TREC9 filtering track: Incrementality, decay, and theshold optimization for adaptive filtering systems. In: Voorhees EM, Harman DK, eds. Proceedings of the 9th Text Retrieval Conference. Gaithersburg, 2000. 87~109.
[6] Bucldey C, Salton G, Allan J. The effect of adding relevance information in a relevance feedback environment. In: Croft WB, van Rijsbergen CJ, eds. Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval. Dublin, ACM/Springer, 1994. 292~300.
[7] Voorhees EM, *et al*. Overview of TREC 2001. In: Voorhees EM, Harman DK, eds. Proceedings of the 9th Text Retrieval Conference. Gaithersburg, 2001. 1~12.
[8] Sebastiani F. Machine learning in automated text categorization. ACM Computing Surveys, 2002,34(1):1~47.
[9] Wu LD, *et al*. FDU at TREC-9: CLIR, filtering and QA tasks. In: Voorhees EM, Harman DK, eds. Proceedings of the 9th Text Retrieval Conference. Gaithersburg, 2000. 202~219.
[10] Robertson SE, Walker S. Microsoft cambridge at TREC9: Filtering track. In: Voorhees EM, Harman DK, eds. Proceedings of the 9th Text Retrieval Conference. Gaithersburg, 2001. 117~131.
[11] Zhang Y, Callan J. Maximum likelihood estimation for filtering thresholds. In: Croft WB, Harper DJ, Kraft DH, Zobel J, eds. SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New Orleans, 2001. 294~302.
[12] Arampatzis A, van Hameren A. The score-distributional threshold optimization for adaptive binary classification tasks. In: Croft WB, Harper DJ, Kraft DH, Zobel J, eds. SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New Orleans, 2001. 285~293.
[13] Lewis DD, Ringuette M. A comparison of two learning algorithms for text categorization. In: Dengel A, Pedersen JO, eds. Proceedings of the 3rd Annual Symposium on Document Analysis and Information Retrieval. Las Vegas, 1994. 81~93.
[14] van Rijsbergen CJ. Information Retrieval. 2 ed. Butterworths, 1979. 102~131.