YFilter at TREC-9

Yi Zhang Jamie Callan
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15232, USA
{yiz,callan}@cs.cmu.edu

ABSTRACT

We built a filtering system YFILTER this year, which we used for experiments on profile updating and thresholds setting. Our focus is using incremental Rocchio for introducing new query terms and term weighting. Although 1, 0.5, 0.25 is a widely used Rocchio ratio for query expansion based on relevance feedback, we found that the optimal setting for information filtering is corpus and profile dependent. In addition to a new Rocchio ratio, we tested a modified idf measure for term weighting (ydf) that is biased towards words with middle range term frequency.

Keywords
Information Filtering

1. INTRODUCTION

Given an initial description of a profile, a filtering system must sift through a stream of information and deliver the most relevant documents to the profile [19]. Filtering is more like a classification problem than a traditional search problem, because of the threshold, which makes it a binary decision process. Many text classification algorithms, such as SVM, Rocchio, Boosting and Naive Bayes, can be applied to filtering [1, 4, 5, 6, 13, 15...], especially for batch filtering and routing. However, as documents arrive sequentially over time, it is unrealistic to use time-consuming algorithms for online filtering. Our goal is to use a computation and storage efficient algorithm, thus making adaptive filtering possible in a normal environment, such as a PC, while still maintaining reasonably good accuracy. Our research interests caused us to adopt stricter constraints than those imposed by the Filtering task [19]. Our filtering system examines each document just once, accruing a small amount of statistical information in the process. The system does not accumulate or store batches of documents, so it requires only minimal storage and moderate computational resources.

A high performance information filtering system should be effective and efficient. Although this is a general requirement for all engineering systems, this year's 5000 MESH profiles make it a clear requirement. A system must handle a large number of user profiles (such as 5000) and a large volume of information efficiently. Previous experiments on text classification suggest that the performance of most text classification algorithms is relatively similar. We hypothesize this will also be true for information filtering. Our starting point, therefore, is to find an algorithm that can be implemented quickly and then to refine it to perform well. We tried several methods that can be implemented incrementally for profile updating (including Rocchio, mutual information and tf.idf). Based on experiments with TREC-8 and TREC-6 filtering data, we used a refined version of the Rocchio algorithm for our final run on this year's OHSUMED data.

2. SYSTEM DESCRIPTION

In order to achieve our goal, we built a new filtering system, called YFILTER. YFILTER is architecturally similar to InRoute [4] and consists of 3 major modules: YParser, YClipset and YLearner, which we used to support dealing with the input stream, filtering according to the profiles, and learning from relevance feedback, respectively. Different learning algorithms for profile updating can be implemented in the YLearner module.

YFILTER supports both structured Boolean and natural language descriptions of initial profiles. For natural language profiles, this system can automatically update the profile according to user relevance feedback.

YFILTER processes text first by removing useless symbols (such as punctuation, and special characters) and fields (such as the .P field), excluding the 418 highly frequent terms listed in the default INQUERY stop words list [3], and then stemming using the Porter stemmer [16].

3. ALGORITHM FOR TREC-9 FILTERING RUNS

3.1 Initial Profile Setting

In all of our experiments, the initial profile is a list of words from the Title and Description fields of the corresponding TREC topic. The weight of each word is its term frequency in the topic.

for each document d_i

for each profile p_k

if d_i is filtered to p_k and has feedback ***//update profile***

update statistics;

if feedback is relevant

add all words in d_i to p_k's candidate term list

calculate weight of each term in p_k's candidate term list according to the Rocchio formula

sort according to the weight, put the top words in the profile's word list

else threshold-= max(maxstep, score(d_i , p_k)-threshold)/sqrt(1+number of feedbacks of p_k)

else ***//automatically decrease threshold***

if(current delivery ratio φ' <minimum delivery ratio φ)

if(performance(p_k)>0)decrease_step=(Threshold-0.4)* φ

else decrease_step=(Threshold-0.4)*($\varphi - \varphi'$)

threshold=max(threshold-decrease_step, 0.400)

Figure 1: An outline of profile updating algorithm

Given 2 initial relevant documents, we update the profile using the Rocchio algorithm [17], and then use the new profile from these 2 initial documents to score other documents without relevance feedback in OHSUMED87. The initial threshold is set to allow the top $\varphi(M+\delta)$ documents in the training dataset to pass. φ is an estimate of the expected minimum delivery ratio we want to achieve. δ is used to make the initial threshold a little lower, so that we can filter more at the beginning, thus obtain more feedback for learning. We arbitrarily set $\delta = 2$ in our experiment.

3.2 Scoring Method

We used the BM25 tf.idf formula [18] for scoring. Idf is initialized based on OHSUMED87 and updated over time as documents are filtered.

3.3 Profile Updating

YFILTER has profile-specific anytime updating. That is, it updates a profile immediately whenever feedback, positive or negative, is available for that profile (Figure 1).

3.3.1 Threshold Updating

The TREC9 ***T9P*** metric is defined as $R_+ / \max(MinD, (R_+ + N_+)$ [19]. $T9P$ demands a minimum number of documents (MinD) be delivered to each profile. MinD is set at 50 documents over the 4-year test period, thus approximately 1 per month. We set our delivery ratio φ accordingly. Each negative feedback increases the threshold. Otherwise, the threshold is always decreasing according to φ and the current profile's performance. The magnitude of an increase to a threshold is limited by *maxstep* , which is empirically set to 0.005, so that a non-relevant document with an extremely high score will not push the threshold too high. Thus, we can avoid undue influence of an outlier. For measuring the performance of a profile, we arbitrarily used the utility F2 used in TREC-8 [9]. If a profile's F2 utility is positive, we regard it as a good profile, and, therefore decrease its threshold comparatively faster (Figure 1). All of the parameters are set based on TREC8 and TREC6 data. The intuition is to filter more documents for good profiles, while keeping the delivery ratio for bad profiles at least meet the requirement set by the $T9P$ measure.

For the $T9U = \{2 * R_+ - N_+ \ \ if\ (2 * R_+ - N_+) > MinU\}$ measure [19], if we filter by estimated probability of relevance based on the score of the current document only, the linear component of $T9U$ is equivalent to the retrieval rule:

Retrieve if $P(rel \mid score) > 0.33$.

Unfortunately, the sparsity of positive relevance feedback makes it hard to find the optimal threshold for most of the profiles, especially for online searching while doing information filtering. If we set the threshold to the one that yields the best utility over the

accumulated documents, while the current profile is built from the same documents, we expect that the resulting threshold is biased. We prefer the $T9P$ measure to the $T9U$ measure, because when there are no positive utilities, filtering no documents is usually the best strategy. Our profile updating method is $T9P$ oriented, and our submission of a $T9U$ oriented run is just a small change of $T9P$ optimization to make the minimum delivery ratio φ 50 times smaller and to decrease the threshold if the profile's precision is better than average.

3.3.2 Updating Terms and Term Weights

In all of our experiments, each time a positive relevance feedback arrives (including those in the training data), all words in that document are added to the profile's candidate list of terms. Then the weight of each word in the candidate list is calculated according to the incremental Rocchio formula:

$$Rocchio = \alpha \cdot w_q + \beta \cdot w_{rel} - \gamma \cdot w_{non-rel} \quad (1)$$

Where

$w_q(t)$: max(term frequency of word t in original topics, 0.5)

$$w_{rel}(t) = \frac{1}{rel\_set(t)+1} \sum_{d \in rel\_set(t)} tf\_bel_{t,d} * ydf_{t,b} \quad (2)$$

$$ydf_{t,d} = -idf_{t,d} \cdot \log(idf_{t,d}) - (1-idf_{t,d}) \cdot \log(1-idf_{t,d}) \quad (3)$$

$$idf_{t,d} = \kappa \cdot \log((C_d + 0.5)/df_{t,d}) / \log(C_d + 1.0) \quad (4)$$

$$tf\_bel_{t,d} = tf_{t,d} / (tf_{t,d} + 0.5 + 1.5 \cdot (dl_d / avg\_dl_d)) \quad (5)$$

The meanings of the above parameters are:

$tf_{t,d}$: Number of times term t occurs in document d

C_d : Number of documents arrived before document d

dl_d : Length of document d

adv_dl_d : Average length of documents arrived before document d

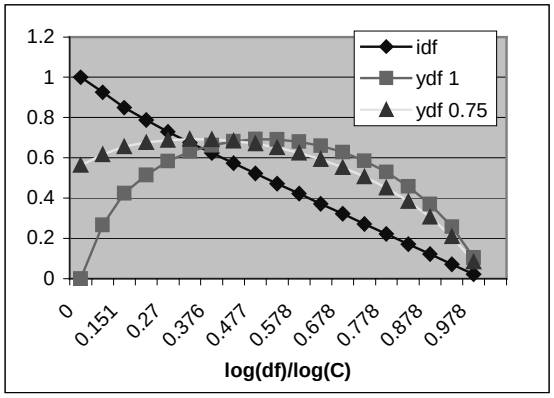
$rel\_set(t)$: Relevant documents after word t is added to the candidate list of the profile

κ : Parameter used to monitoring the query zone

In order to learn faster, we set a bigger β than usual in the relevance feedback formula to emphasize the importance of relevant documents, and changed $w_{rel(t)}$ to emphasize the difference between important words and noisy words.

Some researchers argue that words with middle range term frequencies are more informative than rare words and high frequency words [22], so we introduced a new parameter ydf that favors those words. Ideally ydf is a convex function that reaches its maximum at the optimal query expansion zone. We arbitrarily calculate ydf based on idf using Formula 3 & 4. Setting a different κ can move the query zone. Figure 2 shows the effect of setting κ to 0.75 and to 1.0. We first introduced it while testing mutual information, because mutual information favors rare words [23]. We also found it helpful for Rocchio.

Figure 2: Using ydf to favor words with middle range term frequency: the relationship between document frequency and ydf



In order to avoid adding too many noisy words, especially at the early stage, the number of words added to a profile by its t th updating is at most $7/\gamma$. γ increases as the number of relevant documents increases. We also set the maximum number of words for each profile to 60, because our experiments on query expansion for routing task shows that adding more words than that does not improve the performance. Generally speaking, for the number (N_{kt}) of key words in a profile (P_k) after the t th updating, we have:

$$N_{kt} \le \max(N_{k(t-1)} + 7/\lambda, 60)$$

$\lambda = \sqrt{R_+ + 1}$ if current document that triggers profile updating is relevant, otherwise $7/\lambda$ was set to 0. R_+ is the number of relevant documents that have seen for this profile.

4. ANALYSIS OF RESULT

Topic	Optimized for	Average utility/ precision	Percentage of profiles better than or equal to median	Average median utility
MESH	T9P	0.359	0.50	0.41
MESH SAMPLE	T9P	0.363	0.55	0.40
MESH SAMPLE	T9U	26.7	0.53	34.10
OHSU	T9P	0.267	0.68	0.24
OHSU	T9U	9.3	0.97	-3.75
OHSU	T9P	0.279	0.83	0.24
OHSU	T9U	10.1	0.97	-3.75

Table 1: Submitted runs in TREC-9

4.1 Profile Updating

In order to optimize for the measures used for TREC-9, we set the minimum delivery ratio $\varphi = MinD/(6000 * MinD + 50000)$, where 50000 is the approximate number of documents in OHSUMED87. We have not performed a thorough study of the relationship between φ and the actual delivery ratio based on our algorithm. For the $T9P$ -oriented run, the delivery ratio is about 1/5000 on OHSU topics, and 1/4400 on MESH topics. This is not surprising, because we are expecting a higher delivery ratio for better profiles (Figure 1). So for MESH topics, which contain more relevant documents on average, the actual delivery ratio is higher than our target. But for the same topic, a higher delivery ratio usually results in lower precision.

In our experiments with TREC-6 and TREC-8, we found that a higher β in the Rocchio formula results in higher performance on both corpora. We guessed that this is also true for other corpora if we have enough number of relevance feedback and the positive feedback itself best represents the corresponding topic. As a result we set $(\alpha, \beta, \gamma) = (1, 3.5, 2)$ for the final run on TREC-9. After submitting our results, we did the same experiment on OHSU topics and the first 50 MESH terms to measure the relation between β and precision. The result confirmed our hypothesis (Figure 3), and we did not observe any decrease on precision as β increases. In fact our setting of the Rocchio ratio was a little conservative, although it is already quite different from the widely used $(1, 0.5, 0.25)$ setting.

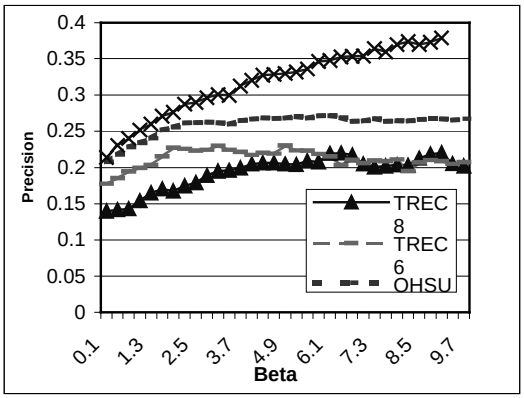


Figure 3: Precision and β setting

4.2 System Performance at Different Stages

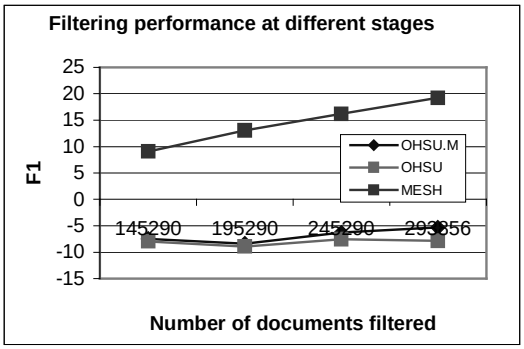


Figure 4: Filtering performances at different stages: F1 Metric

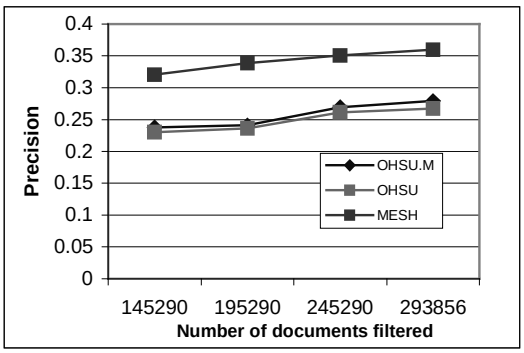


Figure 5: Filtering performances at different stages: Precision Metric

Figure 4 and Figure 5 show our filtering performance using macro average precision and F1 metrics. The horizontal axis does not begin with 0 documents because at the early stage some profiles have no documents filtered to them, and thus the early stage is

not comparable with later stages. It is obvious that filtering performance is improving during the whole process in both measurements. This indicates that the filtering system is learning while filtering. Notice that the performance in Figure 4 and Figure 5 is accumulated performance, so the actual snap shot of performance at difference stages is higher than that was showed here.

4.3 Overall Performance in TREC-9

Our results on OHSU are satisfying, while our results on MESH topics are not so good (Table 1). Possible reasons are:

1. Many parameters are set from experiments on TREC-6 and TREC-8. OHSU topics are more like TREC-6 and TREC-8 topics in query length and in number of relevant documents per profile. For example, we set the maximum number of words for each profile to 60, while this should be different from query to query, based on the number of original query terms. Another example is the Rocchio ratio setting. Further experiments show that a higher β setting, such as 9.7, has a very significant improvement on MESH topics (Figure 3). We believe for better performance, β should be set higher.
2. 7 groups submitted OHSU results, while only 4 groups submitted their MESH results.
3. In order to maintain a certain retrieval rate, we introduced a minimum delivery ratio φ for the threshold setting. But the actual delivery ratio is higher than φ, especially for good profiles, where the goodness is measure by the TREC-8 F2 metric. We have more *good* profiles in the MESH runs, so the actual delivery ratio is higher, thus a lower precision.

4.4 Defects and Explanation

Considering that there are too many non-relevant documents for MESH topics, we did not update the profile every time the system encountered a non-relevant document. While the Rocchio accumulator is inside the profile-updating module, thus the filtering system did not accumulate information for non-relevant documents. The effect is equal to $\gamma = 0$ in Formula 1. In fact we want to use 2 for γ based on our experiments on TREC-6 and TREC-8. After submitting the official result, the bug was fixed, which improved recall from 0.363 to 0.376, and improved precision from 0.267 to 0.271 for OHSU. Further experiments show that when β is set bigger, such as 9, the change of γ has no significant impact. One possible explanation is that non-relevant documents are heterogeneous while relevant documents are homogeneous.

5. CONCLUSIONS

We evaluated our new information filtering system YFILTER by participating in the TREC-9 Adaptive Filtering task. It takes about 10 minutes for YFilter to filter 4 years of MEDLINE dataset for 63 OHSU topics. The experimental results compared favorably with results from other filtering systems. In order to maintain a certain retrieval rate, we introduced a minimum delivery ratio φ for threshold setting, and automatically decreased the profile threshold if its delivery ratio is below that. We found the difference between the actual delivery ratio and φ is reasonable on TREC-6, TREC-8 and TREC-9 data, but further theoretical analysis is needed for a more justifiable threshold setting algorithm. We used incremental Rocchio with a quite different Rocchio ratio setting plus a *ydf* measure that favors middle range term frequency words for adding new terms and term weighting. According to further tests on OHSUMED data after submission of our runs, we find that although our new ratio setting has improved the system performance significantly, it is not optimal for TREC-9. Further experiments show that the optimal Rocchio ratio is corpus and profile dependant. For profiles with more relevance judgments, a higher β is much better, so we think a possible solution is to learn β adaptively. As for the TREC-9 run, we found that Rocchio ratios and the φ setting have a significant influence on system performance. We also believe that the idea of introducing *ydf* will improve performance as well, but our function of mapping from *df* to *ydf* is arbitrary, thus the improvement on TREC-9 runs is not obvious. Further experiments can be focused on finding more principled or more accurate functions for *ydf* calculation.

6. ACKNOWLEDGEMENTS

This material is based on work supported by National Science Foundation grant IIS-9873009 and Air Force Research Laboratory contract F30602-98-C-0110. Any opinions, findings, conclusions or recommendations expressed in this paper are the authors', and do not necessarily reflect those of the sponsors.

7. REFERENCE

[1] J. Allan. 1996. Incremental Relevance Feedback for Information Filtering. *Proceedings of SIGIR 1996*.

[2] T. A.H. Bell, Alistair Moffat, 1996. In *Proc ACM-SIGIR Conference on Research and Development in Information Retrieval*

[3] J. Broglio, J.P. Callan, W.B. Croft, and D.W. Nachbar, 1995. Document retrieval and routing using the INQUERY system. In *Proceeding of Third Text Retrieval Conference (TREC-3),* NIST.

[4] J. Callan. 1996. Document Filtering With Inference Networks. In *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*

[5] J. Callan. 1998. Learning while Filtering. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*

[6] R.E. Schapire, Y. Singer, A. Singhal. 1998 Boosting and Rocchio Applied to Text Filtering. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*

[7] K. Hoashi, K. Matsumoto, N. Inoue, K. Hashimoto. 1999. Experiments on TheTREC-8 Filtering Track. In *Proceeding of Eighth Text Retrieval Conference (TREC-8),* NIST.

[8] K. Hoashi, K. Matsumoto, N. Inoue, K. Hashimoto. 2000. Document Filtering Method Using Non-Relevant Information Profile. *Proceedings of SIGIR 2000*.

[9] D A. Hull, S. E. Robertson. 1999. The TREC-8 Filtering Track Final Report. In *Proceeding of eighth Text Retrieval Conference (TREC-8),* NIST.

[10] D. Hull. 1998. The TREC-7 Filtering Track: Description and Analysis. In *Proceeding of Fourth Text Retrieval Conference (TREC-7),* NIST.

[11] D.A. Hull. 1997. The TREC-6 Filtering Track: Description and Analysis. In *Proceeding of Sixth Text Retrieval Conference (TREC-6),* NIST.

[12] T. Joachims 1997. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. In *Proceedings of International Conference on Machine Learning (ICML).*

[13] Y.H. Kim, S.Y. Hahn, B.T. Zhang, 2000. Text Filtering by Boosting Naive Bayes Classifiers. In *Proceedings of the 23st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*

[14] K.L. Kwok, L. Grunfeld, M. Chan, 1999. TREC-8 Ad-Hoc, Query and Filtering Track Experiments using PIRCS. In *Proceeding of eighth Text Retrieval Conference (TREC-8),* NIST.

[15] R.D. Lyer, D.D. Lewis, R.E. Schapire, Y. Singer, A. Singhal. Boosting for Document Routing. In *Proceeding of ninth International Conference on Information and Knowledge Management.*

[16] M. F. Porter, 1980. An algorithm for suffix stripping.

[17] J. J. Rocchio. 1971. Relevance feedback in information retrieval in The SMART Retrieval System- Experiments in Automatic Document Processing, pages 313-323. Prentice Hall Inc.

[18] S. E. Robertson, S. Walker, M. M. Beaulieu, and M. Gatford, A. Payne, 1995. A. Okapi at TREC-4. In *Proceeding of Fourth Text Retrieval Conference (TREC-4),* NIST.

[19] S. E. Robertson, D. A. Hull 2000. Guidelines for The TREC-9 Filtering Track.

[20] A. Singhal, J. Choi, D. Hindle, D. D. Lewis. 1998. AT&T At TREC-7. In Proceeding of Seventh Text Retrieval Conference (TREC-7), NIST.

[21] S. E. Robertson, S. Walker. 1999. Okapi/Keenbow at TREC-8. In *Proceeding of Eighth Text Retrieval Conference (TREC-8),* NIST.

[22] V. Rijsbergen. 1979 *Information Retrieval*

[23] Y. Yang, J.P. Pedersen 1997. A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*

[24] C. Zhai, P. Jansen, E. Stoica.1998. Threshold Calibration in CLARIT Adaptive Filtering. In *Proceeding of seventh Text Retrieval Conference (TREC-7),* NIST.

[25] C. Zhai, P. Jansen, N. Roma, E. Stoica, D.A. Evans 1999. Optimization in C LARIT Adaptive Filtering. In *Proceeding of eighth Text Retrieval Conference (TREC-8),* NIST.