Robustness of Adaptive Filtering Methods In a Cross-benchmark Evaluation

Yiming Yang, Shinjae Yoo, Jian Zhang, Bryan Kisiel
School of Computer Science, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213, USA

ABSTRACT

This paper reports a cross-benchmark evaluation of regularized logistic regression (LR) and incremental Rocchio for adaptive filtering. Using four corpora from the Topic Detection and Tracking (TDT) forum and the Text Retrieval Conferences (TREC) we evaluated these methods with non-stationary topics at various granularity levels, and measured performance with different utility settings. We found that LR performs strongly and robustly in optimizing T11SU (a TREC utility function) while Rocchio is better for optimizing Ctrk (the TDT tracking cost), a high-recall oriented objective function. Using systematic cross-corpus parameter optimization with both methods, we obtained the best results ever reported on TDT5, TREC10 and TREC11. Relevance feedback on a small portion (0.05~0.2%) of the TDT5 test documents yielded significant performance improvements, measuring up to a 54% reduction in Ctrk and a 20.9% increase in T11SU (with β=0.1), compared to the results of the top-performing system in TDT2004 without relevance feedback information.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: *Information filtering, Relevance feedback, Retrieval models*, *Selection process*; I.5.2 [**Design Methodology**]: *Classifier design and evaluation*

General Terms

Algorithms, Measurement, Performance, Experimentation

Keywords

Adaptive filtering, topic tracking, cross-benchmark evaluations, logistic regression, Rocchio

1. INTRODUCTION

Adaptive filtering (AF) has been a challenging research topic in information retrieval. The task is for the system to make an online topic membership decision (yes or no) for every document, as soon as it arrives, with respect to each pre-defined topic of interest. Starting from 1997 in the Topic Detection and Tracking (TDT) area and 1998 in the Text Retrieval Conferences (TREC), benchmark evaluations have been conducted by NIST under the following conditions[6][7][8][3][4]:

- A very small number (1 to 4) of positive training examples was provided for each topic at the starting point.
- Relevance feedback was available but *only for the system-accepted documents* (with a "yes" decision) in the TREC evaluations for AF.
- Relevance feedback (RF) was not allowed in the TDT evaluations for AF (or *topic tracking* in the TDT terminology) until 2004.
- TDT2004 was the first time that TREC and TDT metrics were jointly used in evaluating AF methods on the same benchmark (the TDT5 corpus) where non-stationary topics dominate.

The above conditions attempt to mimic realistic situations where an AF system would be used. That is, the user would be willing to provide a few positive examples for each topic of interest at the start, and might or might not be able to provide additional labeling on a small portion of incoming documents through relevance feedback. Furthermore, topics of interest might change over time, with new topics appearing and growing, and old topics shrinking and diminishing. These conditions make adaptive filtering a difficult task in statistical learning (online classification), for the following reasons:

1) it is difficult to learn accurate models for prediction based on extremely sparse training data;
2) it is not obvious how to correct the *sampling bias* (i.e., relevance feedback on system-accepted documents only) during the adaptation process;
3) it is not well understood how to effectively tune parameters in AF methods using cross-corpus validation where the validation and evaluation topics do not overlap, and the documents may be from different sources or different epochs.

None of these problems is addressed in the literature of statistical learning for batch classification where all the training data are given at once. The first two problems have been studied in the adaptive filtering literature, including topic profile adaptation using incremental Rocchio, Gaussian-Exponential density models, logistic regression in a Bayesian framework, etc., and threshold optimization strategies using probabilistic

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SIGIR'05, August 15-19, 2005, Salvador, Brazil.
Copyright 2005 ACM 1-59593-034-5/05/0008...$5.00.

calibration or local fitting techniques [1][2][9][10][11][12][13]. Although these works provide valuable insights for understanding the problems and possible solutions, it is difficult to draw conclusions regarding the effectiveness and robustness of current methods because the third problem has not been thoroughly investigated. Addressing the third issue is the main focus in this paper.

We argue that *robustness* is an important measure for evaluating and comparing AF methods. By "robust" we mean consistent and strong performance across benchmark corpora with a systematic method for parameter tuning across multiple corpora. Most AF methods have *pre-specified* parameters that may influence the performance significantly and that must be determined before the test process starts. Available training examples, on the other hand, are often insufficient for tuning the parameters. In TDT5, for example, there is only one labeled training example per topic at the start; parameter optimization on such training data is doomed to be ineffective.

This leaves only one option (assuming tuning on the test set is not an alternative), that is, choosing an external corpus as the validation set. Notice that the validation-set topics often do not overlap with the test-set topics, thus the parameter optimization is performed under the tough condition that *the validation data and the test data may be quite different from each other.* Now the important question is: which methods (if any) are robust under the condition of using cross-corpus validation to tune parameters? Current literature does not offer an answer because no thorough investigation on the robustness of AF methods has been reported.

In this paper we address the above question by conducting a cross-benchmark evaluation with two effective approaches in AF: incremental Rocchio and regularized logistic regression (LR). Rocchio-style classifiers have been popular in AF, with good performance in benchmark evaluations (TREC and TDT) if appropriate parameters were used and if combined with an effective threshold calibration strategy [2][4][7][8][9][11][13]. Logistic regression is a classical method in statistical learning, and one of the best in batch-mode text categorization [15][14]. It was recently evaluated in adaptive filtering and was found to have relatively strong performance (Section 5.1). Furthermore, a recent paper [13] reported that the joint use of Rocchio and LR in a Bayesian framework outperformed the results of using each method alone on the TREC11 corpus. Stimulated by those findings, we decided to include Rocchio and LR in our cross-benchmark evaluation for robustness testing. Specifically, we focus on how much the performance of these methods depends on parameter tuning, what the most influential parameters are in these methods, how difficult (or how easy) to optimize these influential parameters using cross-corpus validation, how strong these methods perform on multiple benchmarks with the systematic tuning of parameters on other corpora, and how efficient these methods are in running AF on large benchmark corpora.

The organization of the paper is as follows: Section 2 introduces the four benchmark corpora (TREC10 and TREC11, TDT3 and TDT5) used in this study. Section 3 analyzes the differences among the TREC and TDT metrics (u*tilities* and *tracking cost*) and the potential implications of those differences. Section 4 outlines the Rocchio and LR approaches to AF, respectively. Section 5 reports the experiments and results. Section 6 concludes the main findings in this study.

2. BENCHMARK CORPORA

We used four benchmark corpora in our study. Table 1 shows the statistics about these data sets.

TREC10 was the evaluation benchmark for adaptive filtering in TREC 2001, consisting of roughly 806,791 Reuters news stories from August 1996 to August 1997 with 84 topic labels (subject categories)[7]. The first two weeks (August 20th to 31st, 1996) of documents is the training set, and the remaining 11 & ½ months (from September 1st, 1996 to August 19th, 1997) is the test set.

TREC11 was the evaluation benchmark for adaptive filtering in TREC 2002, consisting of the same set of documents as those in TREC10 but with a slightly different splitting point for the training and test sets. The TREC11 topics (50) are quite different from those in TREC10; they are queries for retrieval with relevance judgments by NIST assessors [8].

TDT3 was the evaluation benchmark in the TDT2001 dry run[1]. The tracking part of the corpus consists of 71,388 news stories from multiple sources in English and Mandarin (AP, NYT, CNN, ABC, NBC, MSNBC, Xinhua, Zaobao, Voice of America and PRI the World) in the period of October to December 1998. Machine-translated versions of the non-English stories (Xinhua, Zaobao and VOA Mandarin) are provided as well. The splitting point for training-test sets is different for each topic in TDT.

TDT5 was the evaluation benchmark in TDT2004 [4]. The tracking part of the corpus consists of 407,459 news stories in the period of April to September, 2003 from 15 news agents or broadcast sources in English, Arabic and Mandarin, with machine-translated versions of the non-English stories. We only used the English versions of those documents in our experiments for this paper.

The TDT "topics" differ from TREC topics both conceptually and statistically. Instead of generic, ever-lasting subject categories (as those in TREC), TDT topics are defined at a finer level of granularity, for *events* that happen at certain times and locations, and that are "born" and "die", typically associated with a bursty distribution over chronologically ordered news stories. The average size of TDT topics (events) is two orders of magnitude smaller than that of the TREC10 topics. Figure 1 compares the document densities of a TREC topic ("Civil Wars") and two TDT topics ("Gunshot" and "APEC Summit Meeting", respectively) over a 3-month time period, where the area under each curve is normalized to one.

The granularity differences among topics and the corresponding non-stationary distributions make the cross-benchmark evaluation interesting. For example, algorithms favoring large and stable topics may not work well for short-lasting and non-stationary topics, and vice versa. Cross-benchmark evaluations allow us to test this hypothesis and possibly identify the weaknesses in current approaches to adaptive filtering in tracking the drifting trends of topics.

[1] http://www.ldc.upenn.edu/Projects/TDT2001/topics.html

Table 1: **Statistics of benchmark corpora for adaptive filtering evaluations**

Corpus	#Topics	N(tr)	N(ts)	Avg n+ (tr)	Avg n+ (ts)	Max n+ (ts)	Min n+ (ts)	#Topics per doc (ts)
TREC10	84	20,307	783,484	2	9795.3	39,448	38	1.57
TREC11	50	80.664	726,419	3	378.0	597	198	1.12
TDT3	53	18,738*	37,770*	4	79.3	520	1	1.06
TDT5	111	199,419*	207,991*	1	71.3	710	1	1.01

N(tr) is the number of the initial training documents; N(ts) is the number of the test documents;

n+ is the number of positive examples of a predefined topic; * is an average over all the topics.

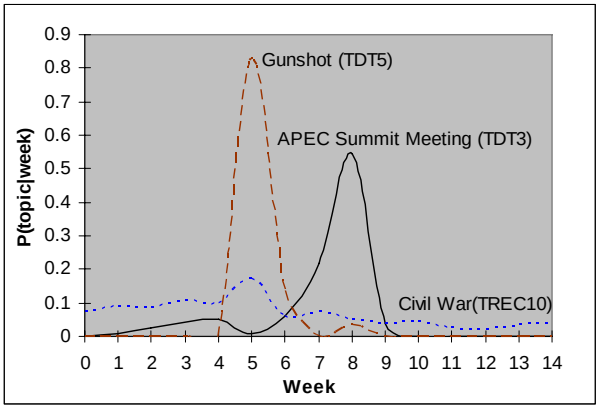

Figure 1: The temporal nature of topics

3. METRICS

To make our results comparable to the literature, we decided to use both TREC-conventional and TDT-conventional metrics in our evaluation.

3.1 TREC11 metrics

Let A, B, C and D be, respectively, the numbers of true positives, false alarms, misses and true negatives for a specific topic, and $N = A + B + C + D$ be the total number of test documents. The TREC-conventional metrics are defined as:

$$\text{Precision} = A/(A+B), \quad \text{Recall} = A/(A+C)$$

$$F_\beta = \frac{(1+\beta^2)A}{A+B+\beta^2(A+C)}$$

$$T11SU_{\beta,\eta} = \frac{\max\left((A-\beta B)/(A+C), \eta\right) - \eta}{1-\eta}$$

where parameters β and η were set to 0.5 and -0.5 respectively in TREC10 (2001) and TREC11 (2002). For evaluating the performance of a system, the performance scores are computed for individual topics first and then averaged over topics (*macro-averaging)*.

3.2 TDT metrics

The TDT-conventional metric for topic tracking is defined as:

$$C_{trk}(T) = w_1 P(T) P_{miss} + w_2 (1 - P(T)) P_{fa}$$

where P(T) is the percentage of documents on topic T, P_{miss} is the miss rate by the system on that topic, P_{fa} is the false alarm rate, and w_1 and w_2 are the costs (pre-specified constants) for a miss and a false alarm, respectively. The TDT benchmark evaluations (since 1997) have used the settings of $w_1 = 1$, $w_2 = 0.1$ and $P(T) = 0.02$ for all topics. For evaluating the performance of a system, *Ctrk* is computed for each topic first and then the resulting scores are averaged for a single measure (the *topic-weighted Ctrk*).

To make the intuition behind this measure transparent, we substitute the terms in the definition of *Ctrk* as follows:

$$P(T) = \frac{A+C}{N}, \quad 1 - P(T) = \frac{B+D}{N},$$

$$P_{miss} = \frac{C}{A+C}, P_{fa} = \frac{B}{B+D},$$

$$C_{trk}(T) = w_1 \cdot \frac{A+C}{N} \cdot \frac{C}{A+C} + w_2 \cdot \frac{B+D}{N} \cdot \frac{B}{B+D}$$

$$= \frac{1}{N} \cdot (w_1 C + w_2 B)$$

Clearly, C_{trk} is the average cost per error on topic *T*, with w_1 and w_2 controlling the penalty ratio for misses vs. false alarms.

In addition to C_{trk}, TDT2004 also employed $T11SU_{\beta=0.1}$ as a utility metric. To distinguish this from the $T11SU_{\beta=0.5}$ in TREC11, we call former TDT5SU in the rest of this paper.

3.3 The correlations and the differences

From an optimization point of view, TDT5SU and T11SU are both *utility* functions while Ctrk is a *cost* function. Our objective is to maximize the former or to minimize the latter on test documents. The differences and correlations among these objective functions can be analyzed through the shared counts of *A, B, C* and D in their definitions. For example, both TDT5SU and T11SU are positively correlated to the values of A and D, and negatively correlated to the values of B and C; the only difference between them is in their penalty ratios for misses vs. false alarms, i.e., 10:1 in TDT5SU and 2:1 in T11SU. The Ctrk function, on the other hand, is positively correlated to the values of C and B, and negatively correlated to the values of A and D; hence, it is negatively correlated to T11SU and TDT5SU.

More importantly, there is a subtle and major difference between Ctrk and the utility functions: T11SU and TDT5SU. That is, Ctrk has a very different penalty ratio for misses vs. false alarms: it favors recall-oriented systems to an extreme. At first glance, one would think that the penalty ratio in Ctrk is 10:1 since $w_1 = 1$ and $w_2 = 0.1$. However, this is not true if $P(T) = 0.02$ is an inaccurate estimate of the on-topic documents on average for the test corpus. Using TDT3 as an example, the true percentage is:

$$P(T) = \frac{n_+}{N} = \frac{79.3}{37770} \approx 0.002$$

where N is the average size of the test sets in TDT3, and n+ is the average number of positive examples per topic in the test sets. Using $\hat{P}(T) = 0.02$ as an (inaccurate) estimate of 0.002 enlarges the intended penalty ratio of 10:1 to 100:1, roughly speaking. To wit:

$$\begin{aligned} C_{trk}(T) &= w_1 \times 0.02 \times P_{miss} + w_2 \times (1 - 0.02) P_{fa} \\ &= w_1 \times \rho \times P(T) P_{miss} + w_2 \cdot (1 - \rho \times P(T)) P_{fa} \\ &= 1 \times 10 \times \frac{C}{N} + 0.1 \times \left(1 - 10 \times \frac{A + C}{N}\right) \frac{B}{B + D} \\ &= 1 \times 10 \times \frac{C}{N} + 0.1 \times \left(1 - 10 \times \frac{79.3}{37770}\right) \frac{B}{(37770 - 79.3)} \\ &\approx 10 \times \frac{C}{N} + 0.1 \times \frac{B}{N} = \frac{1}{N}(10 \times C + 0.1 \times B) \end{aligned}$$

where $\rho = \frac{\hat{P}(T)}{P(T)} = \frac{0.02}{0.002} = 10$ is the factor of enlargement in the estimation of P(T) compared to the truth. Comparing the above result to formula 2, we can see the actual penalty ratio for misses vs. false alarms was 100:1 in the evaluations on TDT3 using Ctrk. Similarly, we can compute the enlargement factor for TDT5 using the statistics in Table 1 as follows:

$$\rho = \frac{\hat{P}(T)}{P(T)} = \frac{0.02}{71.3 / 207{,}991} = 58.3$$

which means the actual penalty ratio for misses vs. false alarms in the evaluation on TDT5 using Ctrk was approximately 583:1.

The implications of the above analysis are rather significant:

- Ctrk defined in the same formula does not necessarily mean the same objective function in evaluation; instead, the optimization criterion depends on the test corpus.
- Systems optimized for Ctrk would not optimize TDT5SU (and T11SU) because the former favors high-recall oriented to an extreme while the latter does not.
- Parameters tuned on one corpus (e.g., TDT3) might not work for an evaluation on another corpus (say, TDT5) unless we account for the previously-unknown subtle dependency of Ctrk on data.
- Results in Ctrk in the past years of TDT evaluations may not be directly comparable to each other because the evaluation collections changed most years and hence the penalty ratio in Ctrk varied.

Although these problems with Ctrk were not originally anticipated, it offered an opportunity to examine the ability of systems in trading off precision for extreme recall. This was a challenging part of the TDT2004 evaluation for AF.

Comparing the metrics in TDT and TREC from a utility or cost optimization point of view is important for understanding the evaluation results of adaptive filtering methods. This is the first time this issue is explicitly analyzed, to our knowledge.

4. METHODS

4.1 Incremental Rocchio for AF

We employed a common version of Rocchio-style classifiers which computes a prototype vector per topic (*T*) as follows:

$$\vec{p}(T) = \alpha \vec{q}(T) + \beta \frac{\sum_{\vec{d} \in D_+(T)} \vec{d}}{|D_+(T)|} - \gamma \frac{\sum_{\vec{d}' \in D_-(T)} \vec{d}'}{|D_-(T)|}$$

The first term on the RHS is the weighted vector representation of topic description whose elements are terms weights. The second term is the weighted centroid of the set $D_+(T)$ of positive training examples, each of which is a vector of within-document term weights. The third term is the weighted centroid of the set $D_-(T)$ of negative training examples which are the nearest neighbors of the positive centroid. The three terms are given pre-specified weights of α, β and γ, controlling the relative influence of these components in the prototype.

The prototype of a topic is updated each time the system makes a "yes" decision on a new document for that topic. If relevance feedback is available (as is the case in TREC adaptive filtering), the new document is added to the pool of either $D_+(T)$ or $D_-(T)$, and the prototype is recomputed accordingly; if relevance feedback is not available (as is the case in TDT event tracking), the system's prediction ("yes") is treated as the truth, and the new document is added to $D_+(T)$ for updating the prototype. Both cases are part of our experiments in this paper (and part of the TDT 2004 evaluations for AF). To distinguish the two, we call the first case simply "Rocchio" and

the second case "PRF Rocchio" where PRF stands for pseudo-relevance feedback.

The predictions on a new document are made by computing the cosine similarity between each topic prototype and the document vector, and then comparing the resulting scores against a threshold:

$$sign(\cos(\vec{p}(T), \vec{d}_{new}) - \theta) = \begin{cases} + (yes) \\ - (no) \end{cases}$$

Threshold calibration in incremental Rocchio is a challenging research topic. Multiple approaches have been developed. The simplest is to use a universal threshold for all topics, tuned on a validation set and fixed during the testing phase. More elaborate methods include probabilistic threshold calibration which converts the non-probabilistic similarity scores to probabilities (i.e., $P(T \mid \vec{d})$) for utility optimization [9][13], and margin-based local regression for risk reduction [11].

It is beyond the scope of this paper to compare all the different ways to adapt Rocchio-style methods for AF. Instead, our focus here is to investigate the robustness of Rocchio-style methods in terms of how much their performance depends on elaborate system tuning, and how difficult (or how easy) it is to get good performance through cross-corpus parameter optimization. Hence, we decided to use a relatively simple version of Rocchio as the baseline, i.e., with a universal threshold tuned on a validation corpus and fixed for all topics in the testing phase. This simple version of Rocchio has been commonly used in the past TDT benchmark evaluations for topic tracking, and had strong performance in the TDT2004 evaluations for adaptive filtering with and without relevance feedback (Section 5.1). Results of more complex variants of Rocchio are also discussed when relevant.

4.2 Logistic Regression for AF

Logistic regression (LR) estimates the posterior probability of a topic given a document using a sigmoid function

$$P(y = 1 \mid \vec{x}, \vec{w}) = 1/(1 + e^{-\vec{w} \cdot \vec{x}})$$

where $\vec{x}$ is the document vector whose elements are term weights, $\vec{w}$ is the vector of regression coefficients, and $y \in \{+1, -1\}$ is the output variable corresponding to "yes" or "no" with respect to a particular topic. Given a training set of labeled documents $D = \{(\vec{x}_1, y_1), \cdots, (\vec{x}_n, y_n)\}$, the standard regression problem is defined as to find the maximum likelihood estimates of the regression coefficients ("the model parameters"):

$$\vec{w}_{ml} = \arg\max_{\vec{w}} \{P(D \mid \vec{w})\} = \arg\max_{\vec{w}} \{\log P(D \mid \vec{w})\}$$
$$= \arg\min_{\vec{w}} \{\textstyle\sum_{i=1}^{n} \log(1 + \exp(-y_i \vec{w} \cdot \vec{x}_i))\}$$

This is a convex optimization problem which can be solved using a standard conjugate gradient algorithm in *O(INF)* time for training per topic, where *I* is the average number of iterations needed for convergence, and *N* and *F* are the number of training documents and number of features respectively [14].

Once the regression coefficients are optimized on the training data, the filtering prediction on each incoming document is made as:

$$sign\left(P(y \mid \vec{x}_{new}, \vec{w}) - \theta_{opt}\right) = \begin{cases} + (yes) \\ - (no) \end{cases}$$

Note that $\vec{w}$ is constantly updated whenever a new relevance judgment is available in the testing phase of AF, while the optimal threshold θ_{opt} is constant, depending only on the pre-defined utility (or cost) function for evaluation. If T11SU is the metric, for example, with the penalty ratio of 2:1 for misses and false alarms (Section 3.1), the optimal threshold for LR is $1/(2+1) = 0.33$ for all topics.

We modified the standard (above) version of LR to allow more flexible optimization criteria as follows:

$$\vec{w}_{map} = \arg\min_{\vec{w}} \left\{ \sum_{i=1}^{n} s(y_i) \log(1 + e^{-y_i \vec{w} \cdot \vec{x}_i}) + \lambda \left\| \vec{w} - \vec{\mu} \right\|^2 \right\}$$

where $s(y_i)$ is taken to be α, β and γ for query, positive and negative documents respectively, which are similar to those in Rocchio, giving different weights to the three kinds of training examples: topic descriptions ("queries"), on-topic documents and off-topic documents. The second term in the objective function is for *regularization*, equivalent to adding a Gaussian prior to the regression coefficients with mean $\vec{\mu}$ and covariance variance matrix $1/2\lambda \cdot \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Tuning λ (≥ 0) is theoretically justified for reducing model complexity ("the effective degree of freedom") and avoiding over-fitting on training data [5]. How to find an effective $\vec{\mu}$ is an open issue for research, depending on the user's belief about the parameter space and the optimal range. The solution of the modified objective function is called the *Maximum A Posteriori (MAP)* estimate, which reduces to the maximum likelihood solution for standard LR if $\lambda = 0$.

5. EVALUATIONS

We report our empirical findings in four parts: the TDT2004 official evaluation results, the cross-corpus parameter optimization results, and the results corresponding to the amounts of relevance feedback.

5.1 TDT2004 benchmark results

The TDT2004 evaluations for adaptive filtering were conducted by NIST in November 2004. Multiple research teams participated and multiple runs from each team were allowed. Ctrk and TDT5SU were used as the metrics. Figure 2 and Figure 3 show the results; the best run from each team was selected with respect to Ctrk or TDT5SU, respectively. Our Rocchio (with adaptive profiles but fixed universal threshold for all topics) had the best result in Ctrk, and our logistic regression had the best result in TDT5SU. All the parameters of our runs were tuned on the TDT3 corpus. Results for other sites are also listed anonymously for comparison.

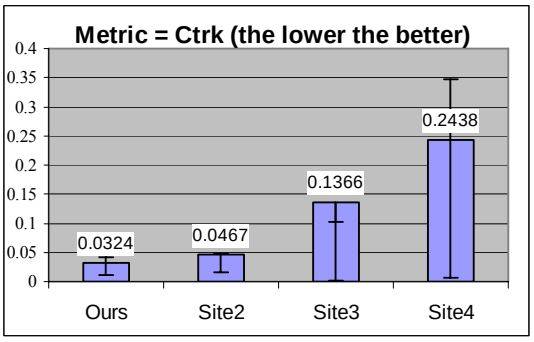

Figure 2: TDT2004 results in Ctrk of systems using true relevance feedback. ("Ours" is the Rocchio method.) We also put the 1st and 3rd quartiles as sticks for each site.[2]

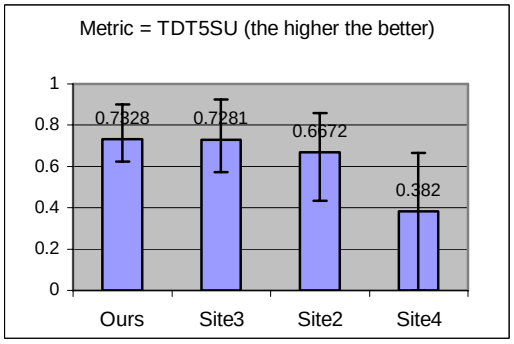

Figure 3:TDT2004 results in TDT5SU of systems using true relevance feedback. ("Ours" is LR with $\vec{\mu} = 0$ and $\lambda = 0.005$).

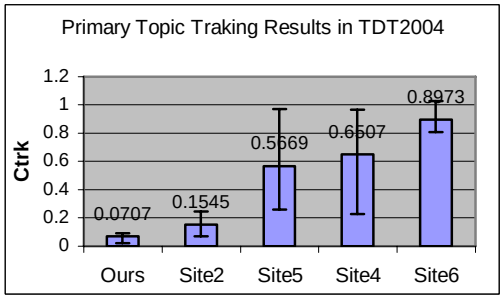

Figure 4:TDT2004 results in Ctrk of systems without using true relevance feedback. ("Ours" is PRF Rocchio.)

Adaptive filtering without using true relevance feedback was also a part of the evaluations. In this case, systems had only one labeled training example per topic during the entire training and testing processes, although unlabeled test documents could be used as soon as predictions on them were made. Such a setting has been conventional for the Topic Tracking task in TDT until 2004. Figure 4 shows the summarized official submissions from each team. Our PRF Rocchio (with a fixed threshold for all the topics) had the best performance.

[2] We use quartiles rather than standard deviations since the former is more resistant to outliers.

5.2 Cross-corpus parameter optimization

How much the strong performance of our systems depends on parameter tuning is an important question.

Both Rocchio and LR have parameters that must be pre-specified before the AF process. The shared parameters include the sample weights α , β and γ , the sample size of the negative training documents (i.e., $D\_(T)$), the term-weighting scheme, and the maximal number of non-zero elements in each document vector. The method-specific parameters include the decision threshold in Rocchio, and $\vec{\mu}$, λ and MI (the maximum number of iterations in training) in LR. Given that we only have one labeled example per topic in the TDT5 training sets, it is impossible to effectively optimize these parameters on the training data, and we had to choose an external corpus for validation. Among the choices of TREC10, TREC11 and TDT3, we chose TDT3 (c.f. Section 2) because it is most similar to TDT5 in terms of the nature of the topics (Section 2). We optimized the parameters of our systems on TDT3, and fixed those parameters in the runs on TDT5 for our submissions to TDT2004. We also tested our methods on TREC10 and TREC11 for further analysis. Since exhaustive testing of all possible parameter settings is computationally intractable, we followed a *step-wise forward chaining* procedure instead: we pre-specified an order of the parameters in a method (Rocchio or LR), and then tuned one parameter at the time while fixing the settings of the remaining parameters. We repeated this procedure for several passes as time allowed.

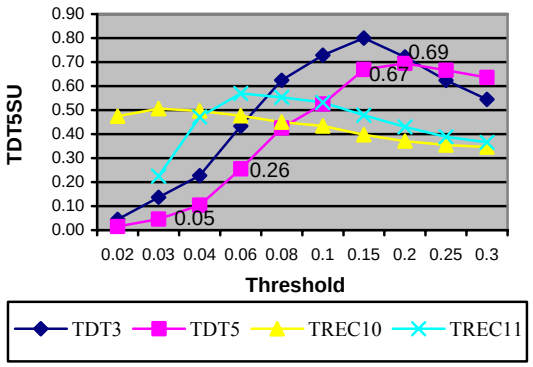

Figure 5: Performance curves of adaptive Rocchio

Figure 5 compares the performance curves in TDT5SU for Rocchio on TDT3, TDT5, TREC10 and TREC11 when the decision threshold varied. These curves peak at different locations: the TDT3-optimal is closest to the TDT5-optimal while the TREC10-optimal and TREC1-optimal are quite far away from the TDT5-optimal. If we were using TREC10 or TREC11 instead of TDT3 as the validation corpus for TDT5, or if the TDT3 corpus were not available, we would have difficulty in obtaining strong performance for Rocchio in TDT2004. The difficulty comes from the ad hoc (non-probabilistic) scores generated by the Rocchio method: the distribution of the scores depends on the corpus, making cross-corpus threshold optimization a tricky problem.

Logistic regression has less difficulty with respect to threshold tuning because it produces probabilistic scores of $\Pr(y = 1 \mid x)$

upon which the optimal threshold can be directly computed if probability estimation is accurate. Given the penalty ratio for misses vs. false alarms as 2:1 in T11SU, 10:1 in TDT5SU and 583:1 in Ctrk (Section 3.3), the corresponding optimal thresholds (t) are 0.33, 0.091 and 0.0017 respectively.

Although the theoretical threshold could be inaccurate, it still suggests the range of near-optimal settings. With these threshold settings in our experiments for LR, we focused on the cross-corpus validation of the Bayesian prior parameters, that is, $\vec{\mu}$ and λ. Table 2 summarizes the results[3]. We measured the performance of the runs on TREC10 and TREC11 using T11SU, and the performance of the runs on TDT3 and TDT5 using TDT5SU. For comparison we also include the best results of Rocchio-based methods on these corpora, which are our own results of Rocchio on TDT3 and TDT5, and the best results reported by NIST for TREC10 and TREC11. From this set of results, we see that LR significantly outperformed Rocchio on all the corpora, even in the runs of standard LR without any tuning, i.e. λ=0. This empirical finding is consistent with a previous report [13] for LR on TREC11 although our results of LR (0.585~0.608 in T11SU) are stronger than the results (0.49 for standard LR and 0.54 for LR using Rocchio prototype as the prior) in that report. More importantly, our cross-benchmark evaluation gives strong evidence for the *robustness* of LR. The robustness, we believe, comes from the probabilistic nature of the system-generated scores. That is, compared to the ad hoc scores in Rocchio, the normalized posterior probabilities make the threshold optimization in LR a much easier problem. Moreover, logistic regression is known to converge towards the Bayes classifier asymptotically while Rocchio classifiers' parameters do not.

Another interesting observation in these results is that the performance of LR did not improve when using a Rocchio prototype as the mean in the prior; instead, the performance decreased in some cases. This observation does not support the previous report by [13], but we are not surprised because we are not convinced that Rocchio prototypes are more accurate than LR models for topics in the early stage of the AF process, and we believe that using a Rocchio prototype as the mean in the Gaussian prior would introduce undesirable bias to LR. We also believe that variance reduction (in the testing phase) should be controlled by the choice of λ (but not $\vec{\mu}$), for which we conducted the experiments as shown in Figure 6.

Table 2: Results of LR with different Bayesian priors

Corpus	TDT3	TDT5	TREC10	TREC11
LR(μ=0,λ=0)	0.7562	0.7737	0.585	0.5715
LR(μ=0,λ=0.01)	**0.8384**	**0.7812**	**0.6077**	**0.5747**
LR(μ=roc*,λ=0.01)	0.8138	0.7811	0.5803	0.5698
Best Rocchio	0.6628	0.6917	0.496[4]	0.475

[3] The LR results (0.77~0.78) on TDT5 in this table are better than our TDT2004 official result (0.73) because parameter optimization has been improved afterwards.

[4] The TREC10-best result (0.496 by Oracle) is only available in T10U which is not directly comparable to the scores in T11SU, just indicative.

*: $\vec{\mu}$ was set to the Rocchio prototype

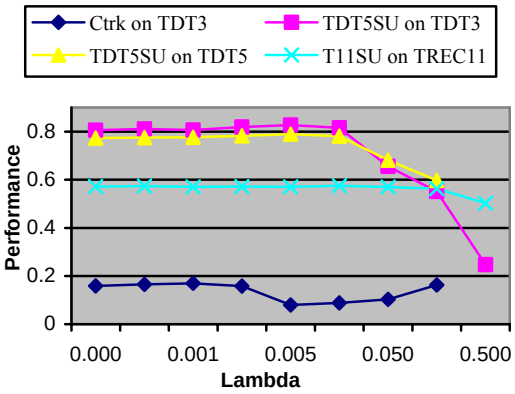

Figure 6: LR with varying lambda.

The performance of LR is summarized with respect to λ tuning on the corpora of TREC10, TREC11 and TDT3. The performance on each corpus was measured using the corresponding metrics, that is, T11SU for the runs on TREC10 and TREC11, and TDT5SU and Ctrk for the runs on TDT3,. In the case of maximizing the utilities, the "safe" interval for λ is between 0 and 0.01, meaning that the performance of regularized LR is stable, the same as or improved slightly over the performance of standard LR. In the case of minimizing Ctrk, the safe range for λ is between 0 and 0.1, and setting λ between 0.005 and 0.05 yielded relatively large improvements over the performance of standard LR because training a model for extremely high recall is statistically more tricky, and hence more regularization is needed. In either case, tuning λ is relatively safe, and easy to do successfully by cross-corpus tuning.

Another influential choice in our experiment settings is term weighting: we examined the choices of binary, TF and TF-IDF (the "ltc" version) schemes. We found TF-IDF most effective for both Rocchio and LR, and used this setting in all our experiments.

5.3 Percentages of labeled data

How much relevance feedback (RF) would be needed during the AF process is a meaningful question in real-world applications. To answer it, we evaluated Rocchio and LR on TDT with the following settings:

- Basic Rocchio, no adaptation at all
- PRF Rocchio, updating topic profiles without using true relevance feedback;
- Adaptive Rocchio, updating topic profiles using relevance feedback on system-accepted documents plus 10 documents randomly sampled from the pool of system-rejected documents;
- LR with $\vec{\mu} = \vec{0}$, $\lambda = 0.01$ and threshold = 0.004;
- All the parameters in Rocchio tuned on TDT3.

Table 3 summarizes the results in *Ctrk*: Adaptive Rocchio with relevance feedback on 0.6% of the test documents reduced the

tracking cost by 54% over the result of the PRF Rocchio, the best system in the TDT2004 evaluation for topic tracking without relevance feedback information. Incremental LR, on the other hand, was weaker but still impressive. Recall that Ctrk is an extremely high-recall oriented metric, causing frequent updating of profiles and hence an efficiency problem in LR. For this reason we set a higher threshold (0.004) instead of the theoretically optimal threshold (0.0017) in LR to avoid an untolerable computation cost. The computation time in machine-hours was 0.33 for the run of adaptive Rocchio and 14 for the run of LR on TDT5 when optimizing Ctrk. Table 4 summarizes the results in TDT5SU; adaptive LR was the winner in this case, with relevance feedback on 0.05% of the test documents improving the utility by 20.9% over the results of PRF Rocchio.

Table 3: AF methods on TDT5 (Performance in Ctrk)

	Base Roc	PRF Roc	Adp Roc	LR
% of RF	0%	0%	0.6%	0.2%
Ctrk	0.076	0.0707	0.0324	0.0382
±%	+7%	(baseline)	-54%	-46%

Table 4: AF methods on TDT5 (Performance in TDT5SU)

	Base Roc	PRF Roc	Adp Roc	LR(λ=.01)
% of RF	0%	0%	0.04%	0.05%
TDT5SU	0.57	0.6452	0.69	0.78
±%	-11.7%	(baseline)	+6.9%	+20.9%

Evidently, both Rocchio and LR are highly effective in adaptive filtering, in terms of using of a small amount of labeled data to significantly improve the model accuracy in statistical learning, which is the main goal of AF.

5.4 Summary of Adaptation Process

After we decided the parameter settings using validation, we perform the adaptive filtering in the following steps for each topic: 1) Train the LR/Rocchio model using the provided positive training examples and 30 randomly sampled negative examples; 2) For each document in the test corpus: we first make a prediction about relevance, and then get relevance feedback for those (predicted) positive documents. 3) Model and IDF statistics will be incrementally updated if we obtain its true relevance feedback.

6. CONCLUDING REMARKS

We presented a cross-benchmark evaluation of incremental Rocchio and incremental LR in adaptive filtering, focusing on their robustness in terms of performance consistency with respect to cross-corpus parameter optimization. Our main conclusions from this study are the following:

- Parameter optimization in AF is an open challenge but has not been thoroughly studied in the past.
- Robustness in cross-corpus parameter tuning is important for evaluation and method comparison.
- We found LR more robust than Rocchio; it had the best results (in T11SU) ever reported on TDT5, TREC10 and TREC11 without extensive tuning.
- We found Rocchio performs strongly when a good validation corpus is available, and a preferred choice when optimizing Ctrk is the objective, favoring recall over precision to an extreme.

For future research we want to study *explicit* modeling of the temporal trends in topic distributions and content drifting.

Acknowledgments

This material is based upon work supported in parts by the National Science Foundation (NSF) under grant IIS-0434035, by the DoD under award 114008-N66001992891808 and by the Defense Advanced Research Project Agency (DARPA) under Contract No. NBCHD030010. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the sponsors.

7. REFERENCES

[1] J. Allan. Incremental relevance feedback for information filtering. In *SIGIR-96*, 1996.

[2] J. Callan. Learning while filtering documents. In *SIGIR-98*, 224-231, 1998.

[3] J. Fiscus and G. Duddington. Topic detection and tracking overview. In *Topic detection and tracking: event-based information organization,* 17—31, 2002.

[4] J. Fiscus and B. Wheatley. Overview of the TDT 2004 Evaluation and Results. In TDT-04, 2004.

[5] T. Hastie, R. Tibshirani and J. Friedman. *Elements of Statistical Learning*. Springer, 2001.

[6] S. Robertson and D. Hull. The TREC-9 filtering track final report. In *TREC-9*, 2000.

[7] S. Robertson and I. Soboroff. The TREC-10 filtering track final report. In *TREC-10*, 2001.

[8] S. Robertson and I. Soboroff. The TREC 2002 filtering track report. In *TREC-11*, 2002.

[9] S. Robertson and S. Walker. Microsoft Cambridge at TREC-9. In *TREC-9*, 2000.

[10] R. Schapire, Y. Singer and A. Singhal. Boosting and Rocchio applied to text filtering. In *SIGIR-98*, 215-223, 1998.

[11] Y. Yang and B. Kisiel. Margin-based local regression for adaptive filtering. In *CIKM-03*, 2003.

[12] Y. Zhang and J. Callan. Maximum likelihood estimation for filtering thresholds. In *SIGIR-01*, 2001.

[13] Y. Zhang. Using Bayesian priors to combine classifiers for adaptive filtering. In *SIGIR-04*, 2004.

[14] J. Zhang and Y. Yang. Robustness of regularized linear classification methods in text categorization. In SIGIR-03: 190-197, 2003.

[15] T. Zhang, F. J. Oles. Text Categorization Based on Regularized Linear Classification Methods. Inf. Retr. 4(1): 5-31 (2001).