Combining multiple learning strategies for effective cross validation

Yiming Yang YIMING@CS.CMU.EDU
Thomas Ault TOMAULT@CS.CMU.EDU
Thomas Pierce TOMP@CS.CMU.EDU
Language Technologies Institute and Computer Science Department, Newell Simon Hall 3612D, Carnegie Mellon University, Pittsburgh, PA 15213-8213

Abstract

Parameter tuning through cross-validation becomes very difficult when the validation set contains no or only a few examples of the classes in the evaluation set. We address this open challenge by using a combination of classifiers with different performance characteristics to effectively reduce the performance variance on average of the overall system across all classes, including those not seen before. This approach allows us to tune the combination system on available but less-representative validation data and obtain smaller performance degradation of this system on the evaluation data than using a single-method classifier alone. We tested this approach by applying k-Nearest Neighbor, Rocchio and Language Modeling classifiers and their combination to the event tracking problem in the Topic Detection and Tracking (TDT) domain, where new classes (events) are created constantly over time, and representative validation sets for new classes are often difficult to obtain on time. When parameters tuned on an early benchmark TDT corpus were evaluated on a later TDT benchmark corpus with no overlapping events, we observed a 38-65% reduction in tracking cost (a weighted combination of errors) by the combined system over the individual methods evaluated under the same conditions, strongly suggesting the robustness of this approach as a solution for improving cross-class performance consistency of statistical classifiers when standard cross-validation fails due to the lack of representative validation sets.

1. Introduction

The fast growing and dynamically changing information space in the real world (e.g., the World Wide Web) presents new challenges to machine learning or statistical classification applied to practical problems. One of those new challenges is how to effectively handle *living, evolving* and *short-lasting* classes. Topic Detection and Tracking (TDT) is such an application domain where satisfactory answers to this question would have a significant impact. TDT is a new line of research focusing on the online detection and tracking of events reported in news stories from multiple sources of TV/radio broadcasts and newswires.[1] Statistical classification techniques have been applied to solve TDT problems, including unsupervised document clustering for event detection and supervised text categorization for event tracking (Allan et al., 1998; Yang et al., 1999; Carbonell et al., 1999; Yamron et al., 1999; Jin et al., 1999; Schwartz et al., 1997; Carbonell et al., 1999). In this paper, we focus our investigation on event tracking, the task of online identification of news stories reporting previously identified events.

Event tracking can be considered a specific form of text categorization subject to the constraints imposed by the TDT domain. In this domain, we assume that the user is unwilling to make exhaustive relevance judgements but is instead only willing to provide a small number of positive examples which define the single event he or she is interested in. Relevance judgements from other users are also assumed to be unavailable. The system must make an on-line decision for each document in the news stream as it arrives (the user is interested in tracking events as they unfold, not after they have occurred). Finally, the system must adhere to the temporal constraints imposed by news streams; only documents preceding the document being evaluated may be used as training data, and of these, only the positive examples identified by the user are explicitly labelled.

There is a subtle yet important distinction between an *event* and a *topic*. An *event* in the TDT context is *something that occurs at specific place and time associated with some specific actions.* It contrasts with a *topic* in the traditional text categorization sense in that events are localized in space and time. For example, the *EgyptAir-990 crash* is an event, but not a topic, and "airplane accidents" is a topic but not an event. Systems in the TDT domain must be able to distinguish between events, regardless of whether they are part of the same topic or not. Events are typically short in duration and thus only a small portion of any corpus will be about any particular event.

[1] In the TDT domain, "topic" is sometimes used as a synonym for "event". To avoid confusion, we will always use "event" when refering to TDT events.

The dynamic nature of events and the small number of positive training examples per event make it difficult to obtain validation sets that are sufficient for effective parameter tuning using standard cross-validation techniques. In the TDT benchmark evaluations, the number of positive training examples per event was 1 to 4, scattered among a larger collection of unlabelled documents which may or may not contain additional positive examples. Holding back even one of these human-identified examples for validation could have a strong negative impact on the performance of the classifier, and the small number of examples held for validation will probably not offer sufficient statistics for parameter tuning.

Furthermore, the theme in an event often evolves over time, and so the examples available for training and validation are often not representative of later stories for the same event. The validation corpora provided for tuning the parameters of systems in TDT benchmark evaluations typically consisted of news stories up to a few months before the start of the evaluation corpus and covered a period of 3-6 months of data. Those retrospective corpora have few, if any, events in common with the evaluation benchmark corpora. Empirical results in these evaluations (by our group and others) have shown a large performance degradation when systems tuned on these retrospective corpora are evaluated on the benchmark corpus, demonstrating that parameter tuning for event tracking systems is a serious problem.

For effective event tracking, we need a classification system that has consistent performance across both old and new classes, and we need to find a way to tune parameters in such a classification system that does not rely on the availability of sufficient labelled examples of new classes. This is a new problem that has not been explicitly addressed in text categorization. In the rest of this paper, we propose our solution to this problem: using a diverse set of classifiers with different performance characteristics to improve the cross-class performance consistency of the overall system. Intuitively, this strategy is similar to a well-known optimal portfolio management strategy for the stock market: invest in a variety of funds with a history of excellent returns to offset the volatility of any one of them, yielding better returns on average in the long term, and was also inspired by the known practice of combining systems in information retrieval, speech recognition and machine learning to improve overall performance(See section 6).

2. Classification Methods

2.1 Rocchio

Rocchio is an effective method using relevance judgments for query expansion in information retrieval and filtering(Jr., 1971; Salton & Buckley, 1990). Applied to text classification (categorization)(Cohen & Singer, 1996; Lewis et al., 1996; Schapire et al., 1998), it uses a vector to represent each class and document, computes their similarity using the cosine value of these two vectors, and obtains a binary decision by thresholding on this value. The vector representation for a document consists of the weights of terms (words or phrases) in the document. In this study, we use a common version of the TF-IDF scheme(Salton & Buckley, 1990) for term weighting, defined to be

$$w(t,\vec{d}) = \frac{(1+\log_2 tf(t,\vec{d})) \times \log_2(N/n_t)}{\sqrt{\sum_{t\in\vec{d}} w(t,\vec{d})^2}}$$

$w(t,\vec{d})$ is the weight of term t in document $\vec{d}$;

$tf(t,\vec{d})$ is the within-document term frequency (TF);

$\log_2(N/n_t)$ is the Inverted Document Frequency (IDF);

N is the number of documents in the training set;

n_t is the number of training documents in which t occurs.

The vector representation for a class, called the *prototype* or *centroid*, is constructed using a set (R) of positive training examples and a set ($\bar{R}$) of negative training examples of that class. We use a variant of Rocchio where $\bar{R}$ consists of the n top-ranking documents ("the query zone"(Schapire et al., 1998)) retrieved from the negative training examples when using the centroid (vector sum) of the positive training examples as the query. The prototype vector is defined to be:

$$\vec{c}(\gamma,n) = \frac{1}{|R|}\sum_{i}^{|R|}\vec{u}_i \in R - \gamma\frac{1}{n}\sum_{i}^{n}\vec{v}_i \in \bar{R}_n \qquad (1)$$

where $\bar{R}_n$ is the "query zone". The scoring function for test document with respect to the class is:

$$f_{roc}(\vec{x}|\vec{c}) = cos(\vec{x},\vec{c}) \qquad (2)$$

The pre-specified parameters in the Rocchio method are n (the size of the local zone), γ (the weight of the negative centroid) and t, the decision threshold. For stable and long-lasting classes, as is typical in traditional text categorization problems, tuning these parameters on a per-class basis often yields better results than tuning the global parameters for all the classes. However, in event tracking, since the events in available validation sets often have little overlap with the events in later documents and parameters tuned for early events are of little use for later events, we use global values for n, γ and t obtained from tuning on a retrospective validation corpus. Rocchio's performance is sensitive to the choices for the values of these parameters, as we will show in Section 5.

2.2 K-Nearest Neighbor Classification

kNN, an instance-based classification method, has been an effective approach to a broad range of pattern recognition and text classification problems(Dasarathy, 1991; Yang, 1999; Yang & Liu, 1999).

In contrast to Rocchio, which uses a static centroid per class, it uses the training documents "local" to each test document to make its classification decision on that document. We have developed two new variants of kNN to address the potential problems when the number of positive training examples is extremely small – a typical situation in event tracking.[2] These new variants, called kNN.avg1 (Formula 3) and kNN.avg2 (Formula 4), use following scoring functions for each test document:

$$f_{avg1}(\vec{x}|k) = \frac{1}{|P_k|}\sum_{\vec{u}\in P_k} cos(\vec{x},\vec{u}) - \frac{1}{|Q_k|}\sum_{\vec{v}\in Q_k} cos(\vec{x},\vec{v}) \quad (3)$$

$$f_{avg2}(\vec{x}|kp,kn) = \frac{1}{|U_{kp}|}\sum_{\vec{u}\in U_{kp}} cos(\vec{x},\vec{u}) - \frac{1}{|V_{kn}|}\sum_{\vec{v}\in V_{kn}} cos(\vec{x},\vec{v}) \quad (4)$$

where documents $\vec{x}, \vec{u}$ and $\vec{v}$ have the same meaning as for Rocchio; P_k (Q_k) is the set of the positive (negative) instances among the k nearest neighbors of $\vec{x}$ in the training set; U_{kp} consists of the kp nearest neighbors of $\vec{x}$ among the positive documents in the training set; and V_{kn} consists of the kn nearest neighbors of $\vec{x}$ among the negative documents in the training set. Binary decisions are obtained by thresholding on f_{avg1} and f_{avg2}.

kNN.avg1 is similar to the conventional versions of kNN where a weighted sum of the scores from the nearest-neighbors training documents is computed for a class label, but kNN.avg1 computes the average instead of the score sum. This modification allows the parameter k to be large without permitting negative examples to dominate the decisions all the time. kNN.avg2 has even greater differences from conventional kNN than kNN.avg1. Instead of using one zone (k nearest neighbors) local to each test document, it uses two zones (kp positive nearest neighbors and and kn negative nearest neighbors), guaranteeing that the system uses both positive and negative examples to score the test document. Thus the system will not lose any discriminatory power when using small local zones. In summary, these two variants of kNN allow more freedom in parameter tuning without causing significant damage in extreme cases. The two methods have different performance characteristics: one tends to produce high-precision results while the other yields better recall benefits. Their error trade-off curves are sensitive to the choices for the values of k, kp and kn, as we will show in Section 3.

[2]These new variants yielded significant performance improvement over the original kNN in event tracking, making kNN among the two top-ranking systems in the TDT3 official evaluation in December 1999. Details about these methods are reported in separate papers(Yang et al., 2000).

2.3 Language Modeling

Various forms of language modeling (LM) have been applied to TDT, including the KL-divergence based clustering approach by Dragon Systems Corporation, the 2-state hidden Markov models (HMM) by BBN, the exponential LM and the hierarchical LM using deterministic annealing by Carnegie Mellon University (CMU) (Yamron et al., 1999; Walls et al., 1999; Jin et al., 1999; Schwartz et al., 1997; Carbonell et al., 1999). For the study in this paper, we implemented the BBN Topic Spotting (BBN/TS) approach[3] to event tracking as an additional method to Rocchio and kNN, so that we can test our hypothesis about using diverse classifiers to reduce the performance variance of combined system in event tracking.

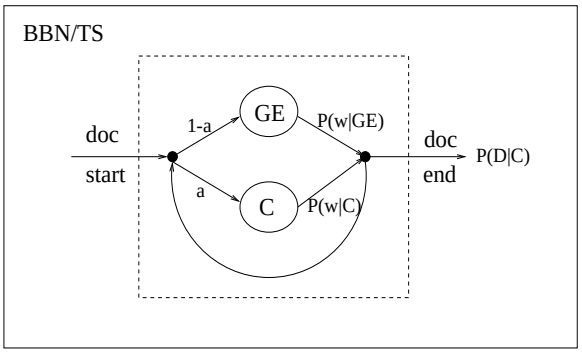

Figure 1: Two state HMM for generating a document on topic C

Described briefly, the BBN/TS method is essentially a naive Bayesian classifier using a specific form of smoothing. Figure 1 illustrates the generation process of an on-topic document by a 2-state hidden Markov model (HMM): the two states correspond to class (event) C and background GE, respectively. Each state has its own word distribution: $P(w_j|C)$ and $P(w_j|GE)$. The smoothed distribution is

$$P'(w_j|C,\lambda) = \lambda P(w_j|C) + (1-\lambda)P(w_j) \quad (5)$$

where λ and $1-\lambda$ are the mixture weights. Both the mixture weights and the two word distributions in the HMM can be learned automatically using a training set of documents and a Expectation Maximization (EM) algorithm. However, BBN/TS chose to fix the mixture weights and $P(w|GE)$, but train the HMM only on $P(w_j|C)$ using a simplified version of EM. Space limitations force us to omit these details, but detailed descriptions can be found in BBN's papers(Walls et al., 1999; Jin et al., 1999; Schwartz et al., 1997).

The scoring function for test document D with respect to class C is defined to be

$$f_{lm}(D|C,\lambda) = \log P(C) + \sum_{w_j\in D}\left[\log\frac{P'(w_j|C)}{P(w_j)}\right] \quad (6)$$

[3]Our results in this paper should not be interpreted as the results by BBN's event tracking systems.

A binary decision is obtained by thresholding on this score. The parameters tuned in this method are the choice of λ and the decision threshold.

2.4 Best Overall Results Generator (BORG)

We hypothesize that by combining the output scores of classifiers whose errors tend to be uncorrelated, the resulting system will have much less cross-collection or cross-event performance variance than those of the individual classifiers. We call the combined system BORG. Ideally, we would like to test BORG with a large number of classifiers, e.g., a few dozen. However, such a large number of systems are not currently available, so we limit our empirical validation to Rocchio, the two variants of kNN and BBN/TS language modeling with a more careful analysis on the behavior of these classifiers and the conditions under which we combine them.

We use the Decision Error Trade-off (DET) curve (Section 3) of the classifiers as the primary means of analyzing the error patterns each produces. Given a validation set, we use the following procedure to generate a BORG system:

1. Run each classifier with different parameter settings, resulting in a set of system-generated scores and a DET curve per run.
2. Select the runs whose DET curves are either globally optimal, or significantly better than other runs in a local region, e.g., for low false alarm (high precision), low miss (high recall), or minimized cost (the weighted sum of misses and false alarms, as defined in Section 3). Allow more than one run to be selected for a classifiers, if needed.
3. Combine the system output of selected runs by first normalizing the scores of each system and then compute the sum of the scores of multiple runs per test document. The normalization formula is
$$x' = \frac{x - \mu}{s.d}$$
where x is the original score, μ is the mean of the scores for the run, and $s.d.$ is their standard deviation. This results in a set of scores of BORG; re-normalize these scores in the same way.
4. Find the optimal threshold for BORG on the validation set. This is BORG's only parameter.

3. Decision Error Trade-Off

In the TDT evaluations, two primary metrics are used to evaluate system performance: "tracking cost," or C_{trk}, which is a weighted combination of miss and false alarm rates (defined below), and the *Decision Error Trade-off (DET)* curve, which is a plot of the normal deviates of miss vs. false alarm rate as the decision threshold is swept over the range of output scores generated by the system. The miss rate, false alarm rate and C_{trk} are computed from the contingency table:

Table 1. Contingency table

	YES is true	NO is true
System-predicted YES	a	b
System-predicted NO	c	d

- Miss Rate $m = \frac{c}{a+c}$ if $a + c > 0$, otherwise undefined;
- False Alarm Rate $f = \frac{b}{b+d}$ if $b + d > 0$, otherwise undefined;
- Tracking cost $C_{trk} = \alpha_1 \frac{b}{n} + \alpha_2 \frac{c}{n}$ where $n = a + b + c + d$.

For the recent TDT evaluations, α_1 and α_2 have been fixed at 0.1 and 1.0 respectively.

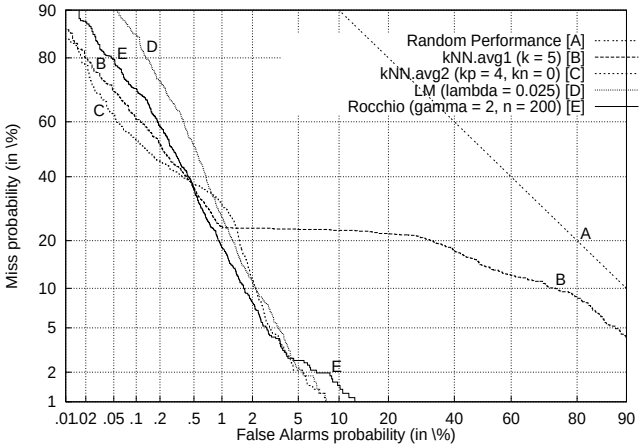

Figure 2: Classifiers in tracking evaluated on TDT1 corpus

An example of the DET curve can be seen in Figure 2. The DET curve is almost a dual of the recall-precision curve found in the text categorization literature. Whereas the recall-precision curve shows the trade-off between two measures of system correctness – recall and precision – as the decision threshold is swept over the range of output scores generated by the system, the DET curve shows the trade off between two error measures – the miss rate and the false alarm rate. It is "almost" a dual form in that instead of plotting the miss and false-alarm scores directly, it plots the normal deviates of those scores, expressed as a percentage (e.g. a normal deviate of x% means that the score at this threshold is higher than x% of the scores at other thresholds), as is illustrated in the DET curve example. Thus, if a system's scores are normally distributed, its DET curve will be a straight line, and a system with random performance will be a straight line with slope -1 passing through the (50%,50%) point. Systems that perform worse than random guessing will have DET curves above this line; those that perform better will have curves below it.

Global performance over all events is reported in the TDT literature using macro-average scores and DET curves almost exclusively. Macro-average scores are

obtained by first computing scores for each event and then averaging those scores over all events, giving each event equal weight in scoring; macro-average DET curves plot macro-average miss vs. false-alarm rates. To keep our results comparable with other results in the TDT literature, we use macro-average C_{trk} and DET curves to present our results. We also use macro-average DET curves to select which systems to combine to form BORG. Figure 2 shows the DET curves of the four "best" tuned methods on the TDT1 corpus. From these curves, we can see that the kNN methods perform best in the high precision (low false-alarm) region, Rocchio performs the best when a more balanced trade-off between recall and precision (misses and false-alarms) is desired, and language modeling has smooth performance over its entire range of decision thresholds. Clearly, all the classifiers perform differently in different regions of the DET space and are thus good candidates to make a BORG classifier.

4. TDT Corpora

We chose the TDT1 corpus of news stories covering July 1, 1994 to June 30, 1995 and the TDT3 dry-run corpus covering January 1, 1998 to June 30, 1998 as our temporally-disjoint corpora. We use both corpora as they are and set the evaluation conditions as close as possible to those used in the TDT1 and TDT3 benchmark evaluations to make our results comparable to the published results on these evaluations.

The TDT1 corpus, developed by the researchers in the TDT Pilot Research Project, was the first benchmark evaluation corpus for TDT research.[4] It consists of 15,863 chronologically-ordered news stories; roughly half of these stories are randomly sampled Reuters articles, and the other half are CNN broadcasts which were manually transcribed by the Journal Graphics Institute (JGI). Twenty five events were manually identified from the memories of the participants for the period covered by this corpus or by scanning through the collection, and then exhaustive relevance judgements were made for each of those events for all stories in the corpus. There are about 43 documents per event on average. Each story was assigned a label of YES (article focusses on event), NO (article does not mention event) or BRIEF[5] (article mentions event in passing) for each of the 25 events. Note that this process resulted in only a subset of the existing events in the corpus; however, for each of those labelled events, complete relevance judgments over all documents are available. For each event and a particular N_t value, the TDT1 corpus was split at the point right after the N_t-th positive example of that event; the stories before that split point were allowed to be used for training, and the remaining stories were used for testing. Fifteen of the 25 events have more than 16 YES stories, the maximum allowable N_t in TDT1, and thus were the ones used for event tracking evaluation. In this paper, we fix N_t at 4 to make the setting consistent with the TDT3 evaluation.

The TDT3 dry-run corpus, developed at LDC, is a larger and richer collection, consisting of 82,084 documents with 100 manually identified events. Each event consists of 350 documents on average and comprises about 0.4% of the corpus. Not all the events have exhaustive relevance judgments over the entire corpus. The documents were collected from 3 newswires, 3 radio programs and 4 television programs; some of these documents are in both English and Mandarin, with a machine-translated (via SYSTRAN at the LDC) version of the Mandarin-language documents also available.[6] The audio sources (TV or radio) were transcribed either manually or by automatic speech recognition. Twenty out of the 100 events were selected for the TDT3 dry-run evaluation on the basis that each event was bi-lingually parallel in the corpus and had more than 4 positive instances in both English and Mandarin. Each tracking system could use up to 4 positive training instances per event as training data for TDT3 dry runs. In the evaluation, the corpus was split immediately after the fourth positive example in the English sources or the fourth positive example in the Mandarin sources, whichever occurred later. The stories before that split point were used for training, and those after for testing, implying that, at the splitting point, there may be more than 4 positive training examples in one of the two languages but only 4 are explicitly labelled.

5. Empirical Validation

Table 2 summarizes the evaluation results of the individual classifiers and BORG under the four conditions: tuned and evaluated on TDT1 ("AA"), tuned on TDT1 and evaluated on TDT3 ("AB"), tuned on TDT3 and evaluated on TDT1 ("BA"), and tuned and evaluated on TDT3 ("BB"). Under the cross-validation condition AB, we obtained a 55% reduction in C_{trk} by using BORG instead of the best single-method classifier (Rocchio with $\gamma = 2$ and $n = 200$) on TDT1. Under the condition BA, the C_{trk} reduction was 13-19% when using BORG instead of the best single-method classifier (kNN.avg1 with $k = 2000$ or kNN.avg2 with $kp = 4$ and $kn = 2000$) on TDT3. Note that difference in performance between BORG and the individual methods is much less on the same-class ("AA" and "BB") evaluation conditions than cross-class ("AB" and "BA") evaluation conditions.

Perhaps the most important point is the small per-

[4] TDT data collections are made available via the Linguistic Data Consortium (LDC) – see www.ldc.upenn.edu/TDT

[5] In the official TDT evaluations, the stories judged as BRIEF were allowed to be used for training but were excluded from testing. We did the same in our evaluations to make our results comparable with the results by others in the TDT benchmark evaluations.

[6] Including Mandarin data is not important for our study; however, we used this corpus as it is to make our results comparable to other TDT benchmark evaluation results.

Table 2. C_{trk} of classifiers evaluated on TDT1 and TDT3 Dry-run Corpora

System (parameters tuned on TDT1)	AA	δ	AB	δ
kNN.avg1 ($k = 5$)	.0033	+34%	.0063	+57%
kNN.avg2 ($kp = 4, kn = 0$)	.0030	+27%	.0076	+65%
Rocchio ($\gamma = -2, n = 200$)	.0022	+3%	.0060	+55%
LM ($\lambda = 0.025$)	.0035	+40%	.0045	+38%
BORG (combining above four)	.0021	-	.0028	-
System (parameters tuned on TDT3)	**BB**	δ	**BA**	δ
kNN.avg1 ($k = 2000$)	.0023	+0%	.0030	+13%
kNN.avg2 ($kp = 4, kn = 2000$)	.0023	+0%	.0032	+19%
Rocchio ($\gamma = -.25, n = 200$)	.0026	+12%	.0033	+21%
LM ($\lambda = 0.25$)	.0040	+43%	.0040	+33%
BORG (combining above four)	.0023	-	.0027	-

AA: tuned on TDT1 and tested on TDT1; AB: tuned on TDT1 and tested on TDT3;
BA: tuned on TDT3 and tested on TDT1; BB: tuned on TDT3 and tested on TDT3;
δ: C_{trk} reduction by using BORG instead of the individual classifier.

formance variance exhibited by BORG between validation ("AA" and "BB") and evaluation ("AB" and "BA") conditions. When tuned on TDT1, only language modeling had a comparably small performance variance (0.0010) to BORG (0.0007) between evaluation on TDT1 ("AA") and TDT3 ("AB"); the other single methods had much larger performance variances than BORG (0.0030-0.0046). When tuned on TDT3, all methods exhibited small performance variances, ranging from 0.0000 (language modeling) and 0.0009 (kNN.avg2) between validation ("BB") and evaluation ("BA") conditions, with BORG having the second-smallest performance variance (0.0004). Although language modeling showed cross-corpus variances comparable to BORG for both the "AA" to "AB" and "BB" to "BA" conditions, it's actual performance was much worse than the other classifiers for all conditions except "AB," where it was second, but still far behind BORG (C_{trk} of 0.0045 vs. 0.0028). Thus, by choosing BORG over the best-performing single method during validation, one is much more likely to "win" during evaluation.

Figure 3 shows the DET curves of all the methods under the AB condition; amazingly, BORG outperformed all the single-method classifiers in the entire DET space. This suggests that the "bad" scores generated by individual classifiers are highly uncorrelated on individual documents, thus the combined scores are globally improved. Figures 4 and 5 compare the DET curves of each method under the conditions of AB and BB, respectively. Evidentially, BORG and LM have much smaller cross-corpus performance variance than the other methods, and BORG is much better than our LM, integrating the good parts of all the single-method classifiers in the DET space.

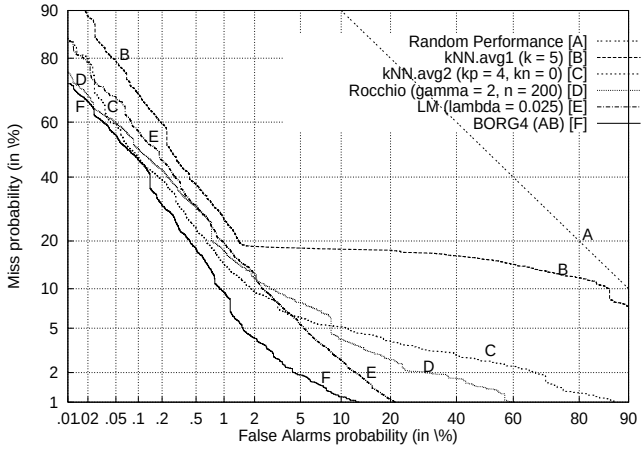

Figure 3: DET curves of classifiers tuned on TDT1 and evaluated on TDT3

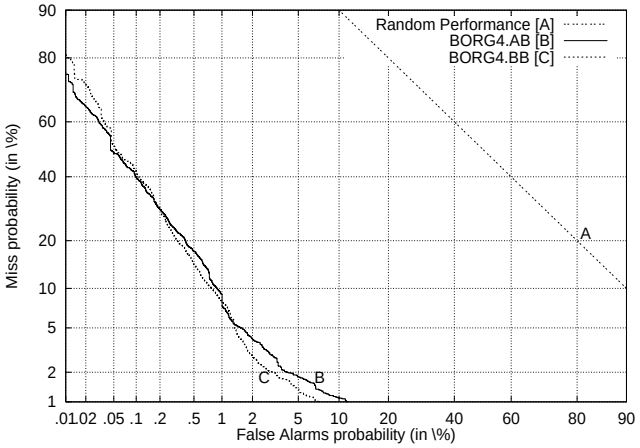

Figure 4: BORG evaluated on TDT3

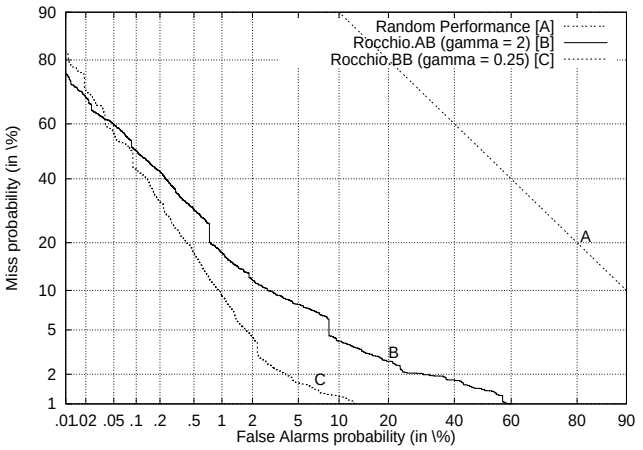

Figure 5:Rocchio evaluated on TDT3

6. Discussion

The benefits of combining multiple systems have been studied in various fields. In information retrieval, performance is often improved when combining systems that exploit different portions or representations of data, multiple schemes of term weighting, or alternate ways of normalization(Katzer et al., 1982; Bartell et al., 1994; Lee, 1995). In speech recognition, word recognition errors are typically reduced when using a dynamic programming algorithm to align the output of multiple recognizers, and then take a majority vote or a linear combination of system-generated scores by these recognizers for a joint prediction of words(Fiscus, 1997). In text categorization, improved effectiveness has been reported when combining different classifiers(Larkey & Croft, 1998; Freitag, 1998), or combining a large number of the same type of classifiers but giving different weights on training examples in a boosting process(Weiss et al., 1999; Schapire et al., 1998). In the TDT area, the work closest our study in this paper is BBN's, who reported significant improvement in event tracking when combining three different language models(Jin et al., 1999), including the one we re-implemented for our study.

While the work mentioned above provided relevant context and motivation for our study, perhaps it is important to point out what is novel in this paper: we identified a non-trivial problem in statistical text categorization that has not been explicitly identified in the previous studies, neither inside nor outside of the TDT domain. This problem is cross-class parameter tuning – using a validation set to tune parameters in a classification system for classes that are different from those in that validation set. Although other researchers in TDT had to implicitly deal with such a problem as a consequence of working in the TDT domain, and some of them (e.g., BBN) have shown improvement in performance of a combination system over their individual classifiers, their evaluations only compared these two approaches (combining the classifiers verses not combining them) using a fixed set of classes. This kind of evaluation does not allow adequate measurement of the cross-class performance variance (or the lack thereof) of classifiers, and therefore is inadequate or insufficient for validating the effect of combining classifiers (although it can be some times indicative) for cross-class parameter tuning. In other words, one may not be able to observe the same phenomena as we presented in our experiments if using only a fixed set of classes to evaluate the combining system. As shown in our results (Table 2), the performance improvement by using BORG over Rocchio was only 3% on the TDT1 validation set (see the AA column in the table), but 55% on the new classes in the TDT3 evaluation set (see the AB column in the same table). These sorts of research findings have not been reported in previous studies.

To summarize, our unique contributions are a novel insight into the problem, an effective solution for this newly identified problem, and the strong empirical evidence to support the effectiveness of our solution in a validation setting which is fundamentally different from the experiments in previous studies in statistical text categorization, including these conducted in the TDT domain.

The main limitation of our experiments so far is that we only used a very simple way to combine classifiers, i.e., giving an equal weight for each classifier in a linear combination. Although the evidence for the power of our approach is already strong, there is room for improvement by finding better ways to identify the performance characteristics of individual classifiers, and by analyzing the properties of their combinations. As for future research, we would like to leverage the rich literature of methods combination in the related areas (information retrieval, machine learning, speech recognition, etc.) to improve the performance of BORG and the theory of cross-class validation.

7. Conclusions

We addressed an open challenge in text classification: parameter optimization for classifiers when representative validation sets for new classes are not available. We proposed a effective solution: combining the output of classifiers with diverse learning strategies and performance characteristics to improve the performance consistency over classes. We tested this approach with the event tracking problem in the TDT domain using two benchmark corpora in which the event sets do not overlap. Using one of these corpora for validation and another for evaluation, we compared the system combining the Rocchio, kNN and LM classification schemes with the performance of the individual classifiers alone and obtained a 38-65% reductions in C_{trk}, the weighted sum of two types of errors.

These results strongly support the robustness of our approach. We believe this is also a powerful solution for problems beyond event tracking, such as categorization of documents (web pages) on the World Wide Web where the category spaces are typically fast growing and changing, or effective training for ex-

tremely rate categories in conventional text categorization problems. We would like to study this approach with a large number of diverse classifiers, including support vector machines, neural networks. etc.

References

Allan, J., Carbonell, J., Doddington, G., Yamron, J., & Yang, Y. (1998). Topic detection and tracking pilot study: Final report. *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop* (pp. 194–218). San Francisco, CA: Morgan Kaufmann Publishers, Inc.

Bartell, B. T., Cottrell, G. W., & Belew, R. K. (1994). Automatic combination of multiple ranked retrieval systems. *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 173–181). New York: The Association for Computing Machinery.

Carbonell, J., Yang, Y., Lafferty, J., D.Brown, R., Pierce, T., & Liu, X. (1999). Cmu report on tdt-2: Segmentation, detection and tracking. *Proceedings of the DARPA Broadcast News Workshop* (pp. 117–120). San Francisco, CA: Morgan Kaufmann Publishers, Inc.

Cohen, W. W., & Singer, Y. (1996). Context-sensitive learning methods for text categorization. *SIGIR '96: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 307-315.

Dasarathy, B. V. (1991). *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. McGraw-Hill Computer Science Series. Las Alamitos, California: IEEE Computer Society Press.

Fiscus, J. (1997). A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover). *IEEE Workshop on Automatic Speech Recognition and Understanding*. Piscataway, NJ: IEEE Signal Processing Society.

Freitag, D. (1998). Multistrategy learning for information extraction. *Proceedings of the Fifteenth International Conference on Machine Learning* (pp. 161–169). San Francisco: Morgan Kaufmann.

Jin, H., Schwartz, R., Sista, S., & Walls, F. (1999). Topic tracking for radio, tv broadcast and newswire. *Proceedings of the DARPA Broadcast News Workshop* (pp. 199–204). San Francisco, CA: Morgan Kaufmann Publishers, Inc.

Jr., J. J. R. (1971). Relevance feedback in information retrieval. *The SMART Retrieval System: Experiments in Automatic Document Retrieval* (pp. 313–323). Englewood Cliffs, New Jersay: Prentice-Hall, Inc.

Katzer, J., McGill, M., Tessier, J., Frankes, W., & Dasupta, P. (1982). A study of the overlap among document representations. *Information Technology: Research and Development* (pp. 261–274).

Larkey, L. S., & Croft, W. B. (1998). Combining classifiers in text categorization. *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 289–297). New York: The Association for Computing Machinery.

Lee, J. H. (1995). Combining multiple evidence from different properties of weighting schemes. *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 180–188). New York: The Association for Computing Machinery.

Lewis, D. D., Schapire, R. E., Callan, J. P., & Papka, R. (1996). Training algorithms for linear text classifiers. *SIGIR '96: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 298-306.

Salton, G., & Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of American Society for Information Sciences, 41*, 288–297.

Schapire, R. E., Singer, Y., & Singhal, A. (1998). Boosting and rocchio applied to text filtering. *21th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)* (pp. 215–223).

Schwartz, R., Imai, T., Nguyen, L., & Makhoul, J. (1997). A maximum likelihood model for topic classification of broadcast news. *Proceedings of Eurospeech* (pp. 1455–1458). Rhodes, Greece.

Walls, F., Jin, H., Sista, S., & Schwartz, R. (1999). Topic detection in broadcast news. *Proceedings of the DARPA Broadcast News Workshop* (pp. 193–198). San Francisco, CA: Morgan Kaufmann Publishers, Inc.

Weiss, S., Apte, C., Damerau, F., Johnson, D., Oles, F., Goets, T., & Hampp, T. (1999). Maximizing text-mining performance. *IEEE Intelligent Systems, Special Issue on Applications of Intelligent Information Retrieval, 14*, 63–69.

Yamron, J., Carp, I., Gillick, L., Lowe, S., & van Mulregt, P. (1999). Topic tracking in a news stream. *Proceedings of the DARPA Broadcast News Workshop* (pp. 133–136). San Francisco, CA: Morgan Kaufmann Publishers, Inc.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval, 1*, 67–88.

Yang, Y., Ault, T., & Pierce, T. (2000). Improving text categorization methods for event tracking. *The 23th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'00)* (p. (accepted)).

Yang, Y., Carbonell, J., Brown, R., Pierce, T., Archibald, B. T., & Liu, X. (1999). Learning approaches for detecting and tracking news events. *IEEE Intelligent Systems, Special Issue on Applications of Intelligent Information Retrieval, 14*, 32–43.

Yang, Y., & Liu, X. (1999). A re-examination of text categorization methods. *The 22th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)* (pp. 42–49).