On-line Spam Filter Fusion

Thomas R. Lynam and Gordon V. Cormack
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario N2L 3G1, Canada
trlynam@uwaterloo.ca, gvcormac@uwaterloo.ca

ABSTRACT

We show that a set of independently developed spam filters may be combined in simple ways to provide substantially better filtering than any of the individual filters. The results of fifty-three spam filters evaluated at the TREC 2005 Spam Track were combined post-hoc so as to simulate the parallel on-line operation of the filters. The combined results were evaluated using the TREC methodology, yielding more than a factor of two improvement over the best filter. The simplest method – averaging the binary classifications returned by the individual filters – yields a remarkably good result. A new method – averaging log-odds estimates based on the scores returned by the individual filters – yields a somewhat better result, and provides input to SVM- and logistic-regression-based stacking methods. The stacking methods appear to provide further improvement, but only for very large corpora. Of the stacking methods, logistic regression yields the better result. Finally, we show that it is possible to select a priori small subsets of the filters that, when combined, still outperform the best individual filter by a substantial margin.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]:information filtering

General Terms: Experimentation, Measurement

Keywords: spam, email, filtering, classification

1. INTRODUCTION

We investigate methods of spam filter fusion – combining the output from separate filters to form a better result. Fusion methods, under a variety of names [12], have been found to achieve varying degrees of benefit for classification and ranked information retrieval applications. Our test setup is different from what is commonly used to evaluate classifiers and information retrieval systems. The input is real email, large-scale, and presented to the filter in chronological order. There is no explicit training set; learning takes place on-line. The filter must return a score as well as a binary classification for each message in turn, after which it is informed of the true classification.

Prior to TREC 2005 [26], we conducted pilot tests using the TREC Spam Filter Evaluation Tool Kit [19], eight open-source filters, and two email corpora containing 55,120 messages in total. These tests supported the primary hypothesis – that naïve fusion improves on the best base filter. The pilot tests also indicated by exhaustive enumeration that subset selection or different score-combining methods might provide further benefit.

After TREC 2005, we conducted tests using the output from fifty-three spam filters run on four corpora within the context of the TREC 2005 Spam Evaluation Track [7]. The fifty-three filters were developed by seventeen independent organizations; the four corpora, totaling 318,482 messages, were derived from independent sources. The principal objective of these tests was to test the primary hypothesis; a secondary objective was to examine the effectiveness of new fusion and subset selection methods.

2. BACKGROUND AND RELATED WORK

We address the problem of on-line content-based spam filtering, an adaptive binary text classification problem[6, 23]. A stream of incoming email messages is presented to the filter, which must label each as spam or ham (not spam). The filter's effectiveness (ineffectiveness) is measured by the proportion of spam and the proportion of ham that it correctly (incorrectly) classifies. As it is difficult to quantify the relative cost of spam and ham misclassification errors, filters typically expose to the user a threshold parameter that may be adjusted to improve one at the expense of the other [18].

Text classification has been studied within the context of information retrieval and machine learning. Spam filtering in particular has been addressed within these contexts; however, the TREC 2005 Spam Evaluation Track provides the first standard test corpora and evaluation tools, and abstracts the problem differently from previously reported efforts. Spam filtering has been the subject of much practical interest; currently, hundreds of commercial and free filters are available. Many rely on content-based classification techniques; others use techniques that are beyond the scope of this evaluation.

Combining the output from multiple tools has been reported to improve information retrieval [20, 21, 2, 25] and classification performance [4, 28, 17, 13, 15]. In information retrieval, a primary concern has been the combination of ranked lists of documents retrieved by different systems. The combination of the results from differently structured

SIGIR'06, August 6–11, 2006, Seattle, Washington, USA.
Copyright 2006 ACM 1-59593-369-7/06/0008 ...$5.00.

queries has also been investigated [3]. These techniques are generally applied to a batch process in which entire ranked lists are combined. The TREC spam filtering approach resembles ranked retrieval in that the *spamminess* score reported by the filter in effect ranks messages, but the ranking is incremental as the scores must be determined one message at a time, without knowledge of future messages.

Ensemble methods [9] have been the subject of much investigation for machine learning in general and for classification in particular. Bagging and boosting combine the results of several weak classifiers, typically employing the same algorithm over perturbed training sets or configuration parameters. Stacking [27], in contrast, uses a meta-learning technique to induce the best combination of stronger classifiers that employ distinct methods. In general, these investigations have employed a batch learning configuration and have been evaluated based on their binary classification effectiveness using separate training and test sets.

Neither naïve fusion nor stacking has been shown conclusively to have substantial benefit in this application. Dzeroski and Zenko state with respect to general text classification, "Typically, much better performance is achieved by stacking as opposed to voting," and "Our empirical evaluation of several recent stacking approaches shows they perform comparably to the best of the individual classifiers selected by cross-validation, but not better."[10] Hull et al., within the context of batch filtering, state, "We have found that simple averaging of probabilities or log odds ratios generates a significantly better ranking of documents," and "We generated [meta] parameter estimates using both linear and logistic regression but failed to reach the standard set by the simple averaging strategies."[13] Sakkis et al. stack Naïve Bayes and k-nearest-neighbor (KNN) classifiers using a KNN meta-classifier over various parameter configurations and observe that the best stacking configuration outperforms the best individual classifier configurations by a small margin: "The results presented here motivate further work in the same direction. In particular, we are interested in combining more classifiers [...] Finally, it would be interesting to compare the performance of stacked generalization to other multi-classifier methods [...] ."[22]

Segal et al. [24] employ a pipeline of purpose-built filters to analyze various aspects of email messages. At the end of the pipeline, if no filter has definitively classified the message, the scores from all filters are combined using linear coefficients computed by a non-linear optimizer, the combination showing improvement over the individual filters.

3. TREC SPAM FILTER EVALUATION

TREC 2005 Corpora

	Ham	Spam	Total
Mr X	9038	40048	49086
S B	6231	775	7006
T M	150685	19516	170201
Full	39399	52790	92189
Aggregate	205253	113129	318482

Table 1: TREC Corpus Statistics

TREC, the Text Retrieval Conference, provides large test collections, uniform scoring procedures, and an annual forum for comparing results for a number of information-retrieval applications. While TREC has previously examined batch and adaptive filtering, spam filter effectiveness was first addressed in TREC 2005.

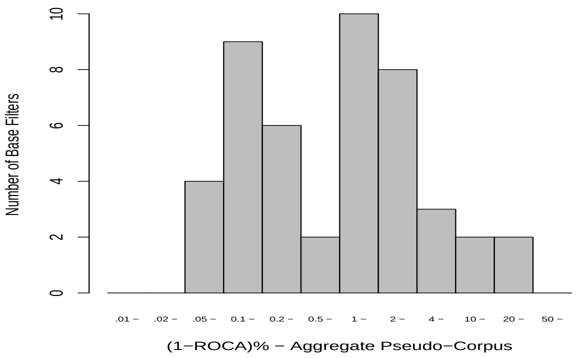


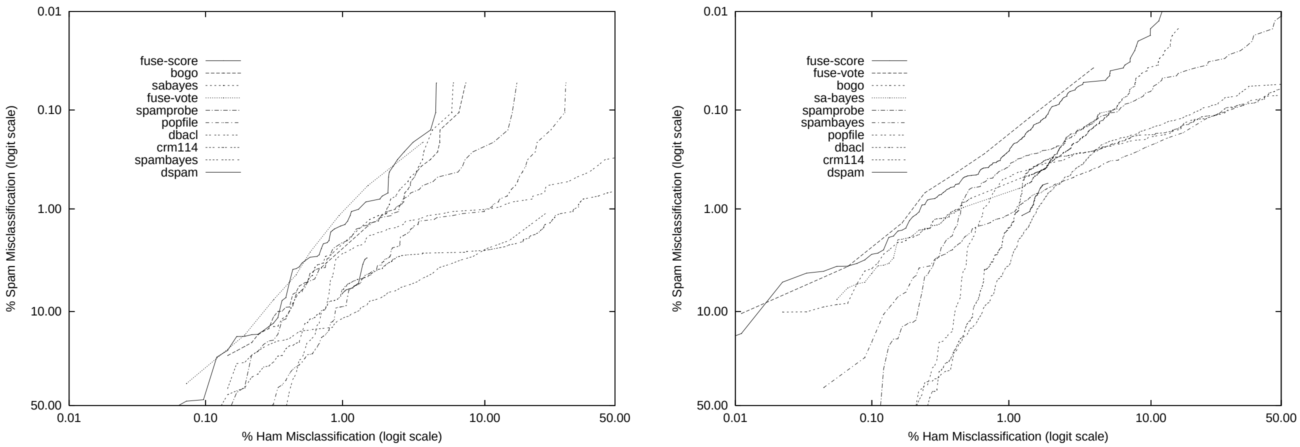
Figure 1: TREC Filter Performance Distribution

The TREC Spam Filter Evaluation Tool Kit, developed for TREC 2005, provides a standardized method for running and evaluating spam filters. Instead of specifying the relative cost of spam and ham errors, the toolkit requires the filter to return a spamminess score that may be compared to an external threshold to yield a binary classification. In addition, the filter must return a binary classification based on some internal threshold chosen by the filter implementor. Receiver Operating Characteristic (ROC) curves provide a mechanism for comparing filters over various possible threshold settings [11]. In addition, the area under the curve (AUC or ROCA) provides a useful summary measure of filter performance. Spam filters typically have extremely low error rates - ROCA = 0.9999 is not uncommon; therefore the toolkit reports 1-ROCA (the area above the curve) as a percentage. That is, ROCA = 0.9999 is reported as (1-ROCA)% = .01. The toolkit also reports (also as percentages) spam misclassification proportion (sm%) at various ham misclassification proportions (hm%). The toolkit provides bootstrap-estimated 95% confidence limits for all ROC measures (cf. [8]).

The toolkit invokes each filter using a command-line interface that presents the messages one at a time to the filter. After the filter returns a classification and score, the true classification is communicated to the filter so that it may learn from the message. The toolkit collects a result file with one line per message containing the filter's output and the true classification. This result file is used as input to the evaluation component of the toolkit, which computes (among others) the following effectiveness indicators: ROC curve, (1-ROCA)%, and sm% at hm% = 0.1.

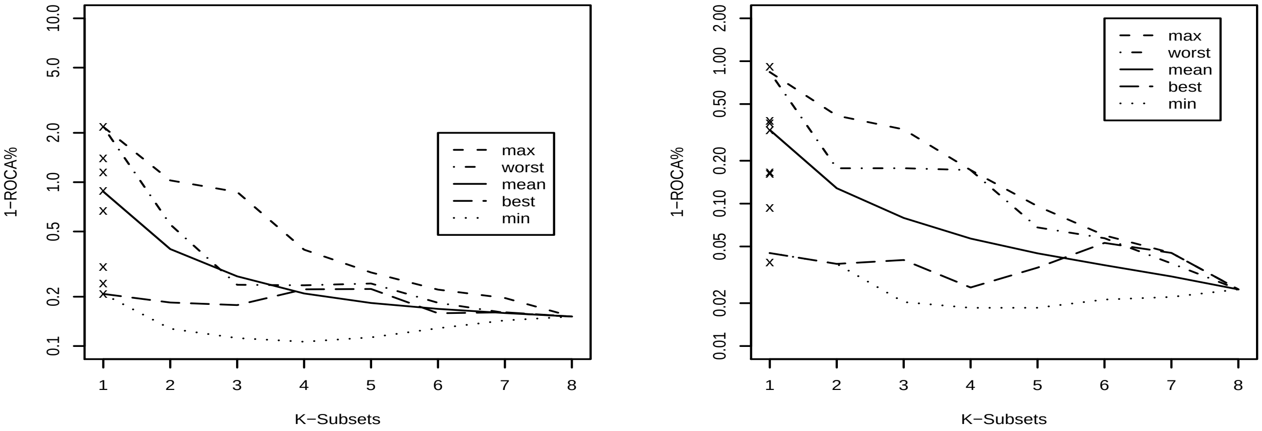
Twelve independent groups participated in the TREC 2005 Spam Track. Each submitted up to four spam filters for evaluation. In addition, variants of five open-source filters were adapted, in consultation with their authors, for evaluation. In total, 53 filters authored by 17 organizations were evaluated[1]. The filters were developed entirely independently from the test corpora and from the authors of this study;

[1] Several filters failed to run on some of the corpora and are excluded from the results on those particular corpora; 46 filters ran successfully on all corpora.



SpamAssassin Corpus **Mr X Corpus**

Figure 2: Pilot Fusion Filters vs. Base Filters



SpamAssassin Corpus **Mr X Corpus**

Figure 3: Pilot Subset Selection

the filters were neither designed nor selected to be amenable to fusion. We used the output from all TREC Spam Track runs as the basis of our main fusion experiments.

Four separately-sourced corpora, ranging in size from 7006 to 170201 messages, were used for evaluation (see table 1). For the purpose of meta-analysis, the results on the four corpora were aggregated and the same summary measures were computed on the aggregate.

Performance among the filters differed dramatically. For example, figure 1, the distribution of (1-ROCA)% of the TREC runs on the aggregate, shows three orders of magnitude difference between the best and the worst. Individual corpus results show similar diversity.

Details of the TREC 2005 filters, corpora and results may be found in the proceedings.[26]

4. PILOT EXPERIMENT

The pilot experiment investigated two naïve fusion methods – voting and normalized score averaging – using eight open-source filters[2] and two test corpora ($n = 6034$[3]; $n = 49086$[4]). We also investigated the potential impact of subset selection by applying the techniques to all 255 non-empty subsets of base filters.

Figure 2 shows superior ROC curves for the two fusion methods, as compared to all of the base filters. But only one curve, normalized score averaging on the larger corpus, nets a significantly better (1-ROCA)% statistic ($p < .02$) than the best base filter.

[2] Bogofilter [bogofilter.sourceforge.net],
CRM114 [crm114.sourceforge.net],
dbacl [dbacl.sourceforge.net],
DSPAM [dspam.sourceforge.net],
POPFile [popfile.sourceforge.net],
SpamAssassin (Bayes filter only) [spamassassin.apache.org],
SpamBayes [spambayes.sourceforge.net],
SpamProbe [spamprobe.sourceforge.net].

[3] SA Corpus [spamassassin.apache.org/publiccorpus].

[4] Mr X Corpus [plg.uwaterloo.ca/˜gvcormac/mrx].

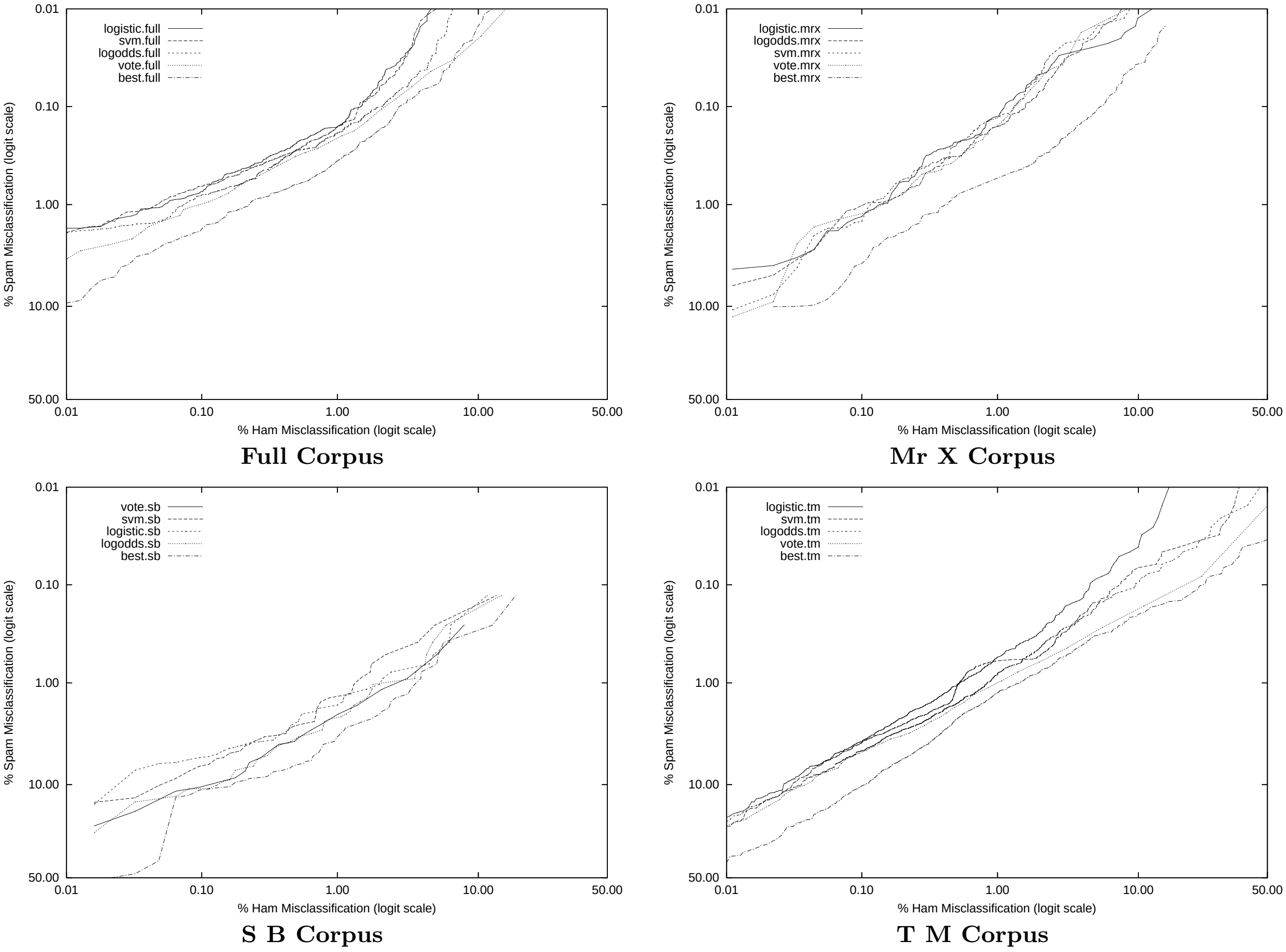


Figure 4: Fusion Filters vs. Best Filter

Figure 3 shows (1-ROCA)% for normalized averaging over k-subsets of the base runs, as a function of k. The curves labeled *max*, *min*, and *mean* are over the (1-ROCA%) scores yielded by all subsets of size k. The curves labeled *best* and *worst* are yielded by selecting post-hoc the base runs that, taken individually, yield the k best and worst (1-ROCA)% statistics. The x symbols on the 1-axis indicate (1-ROCA)% for each of the base runs.

From the pilots we concluded that the naïve combination methods were worthy of further validation. However, we were uncomfortable with normalized averaging as a method for combining scores, as it relies on unwarranted assumptions about the distribution of spamminess scores returned by the base filters. We determined, therefore, to seek to devise a method that relied only on the warranted assumption that each filter would attempt to minimize (1-ROCA)%; that is, to minimize the number of pairs of ham and spam messages in which the ham message yielded the higher spamminess score.

From the k-subset analysis we found reason to hypothesize that subsets of the base filters might be found a priori (as opposed to a posteriori in the pilot) that would yield better performance, or that would yield good performance with less computational expense. And if subsets might be learned, so might other linear and non-linear combinations of the scores.

5. FUSION EXPERIMENT

The primary purpose of our main experiment was to validate the hypothesis that each of the following methods would improve on the best of a collection of separate filters. A secondary purpose was to assess the relative effectiveness of the methods.

Best Filter. As a baseline for comparison, we selected (a posteriori) the filter achieving the best ROC score on each corpus.

Voting. Each base filter's output consists of a binary classification and a spamminess score. Vote fusion uses only the binary classification output of the base filters. The fused filter's spamminess score for a message is the fraction of base filters that classify it as spam – a number between 0 and 1. The fused filter's binary classification is determined relative to some arbitrary constant threshold $0 < t < 1$; a spam classification is returned when $spamminess > t$. The summary statistics that we present are insensitive to our choice of $t = 0.5$.

Log-odds averaging. When a filter reports a spamminess score s_n for the nth message, we estimate L_n, the odds that the message is spam to be

$$L_n = log\, \frac{\mid \{i < n \,|\, s_i \leq s_n \, and\, ith\, message\, is\, spam\} \mid + \epsilon}{\mid \{i < n \,|\, s_i \geq s_n \, and\, ith\, message\, is\, ham\} \mid + \epsilon}\ .$$

Method	$(1-ROCA)\%$	$sm\%@hm\% = .1$
logistic	.007*** (.005-.008)	.73*** (.55-.98)
svm	.008*** (.005-.013)	.65*** (.55-.77)
logodds	.009*** (.007-.011)	.80*** (.65-.98)
vote	.013* (.010-.018)	1.00*** (.82-1.21)
best	.019 (.015-.023)	1.78 (1.42-2.22)

Full Corpus

Method	$(1-ROCA)\%$	$sm\%@hm\% = .1$
logistic	.010*** (.007-.014)	1.32* (.68-2.58)
logodds	.011*** (.007-.016)	1.02** (.53-1.97)
svm	.011*** (.007-.017)	1.48* (.73-2.98)
vote	.014*** (.008-.024)	1.21** (.86-1.71)
best	.045 (.032-.063)	3.90 (1.55-9.50)

Mr X Corpus

Method	$(1-ROCA)\%$	$sm\%@hm\% = .1$
vote	.115** (.071-.184)	10.5 (6.75-15.8)
svm	.155 (.046-.516)	6.71 (3.66-12.0)
logistic	.166 (.057-.483)	5.55 (3.57-8.53)
logodds	.193 (.076-.490)	11.0 (7.01-16.8)
best	.231 (.142-.377)	11.2 (4.38-25.9)

S B Corpus

Method	$(1-ROCA)\%$	$sm\%@hm\% = .1$
logistic	.036*** (.030-.044)	3.89*** (3.43-4.41)
svm	.055*** (.045-.067)	3.97*** (3.50-4.49)
logodds	.061*** (.045-.067)	4.78*** (4.27-5.33)
vote	.095** (.079-.115)	4.91*** (4.45-5.43)
best	.135 (.111-.163)	10.3 (9.16-11.6)

T M Corpus

Method	$(1-ROCA)\%$	$sm\%@hm\% = .1$
logistic	.012*** (.010-.015)	1.20*** (1.07-1.35)
svm	.017*** (.015-.021)	1.29*** (1.16-1.45)
logodds	.020*** (.017-.023)	1.78*** (1.64-1.93)
vote	.028*** (.023-.033)	1.66*** (1.48-1.86)
best	.051 (.044-.058)	3.78 (3.36-4.25)

Aggregate Results

improvement on best: $^*p < .05$, $^{**}p < .005$, $^{***}p < .0005$

Table 2: Fusion Summary Statistics

That is, we simply count the number of prior spam messages with a lower or equal score and the number of prior non-spam messages with a higher or equal score, and take the log of their ratio. The necessary counting can be done in $O(log\, n)$ time with a suitable data structure [5]. The fused spamminess score is the arithmetic mean of the base filters' L_n scores. We set $t = 0$.

SVM. L_i scores were used as features and all prior messages were used as a training set. SVMlight's [14] default kernel and parameters were used. For efficiency reasons, SVMlight was not run after every message; retraining was effected at Fibonacci-like intervals.[5] The SVMlight output was used directly as the fused spamminess score. We set $t = 0$.

Logistic regression. The LR-TRIRLS logistic regression package [16] was used to find weights such that the weighted average of the base filters' L_i scores best predicted the log-odds of the classification of prior messages. This weighted average was used as the spamminess score, and we set $t = 0$. Negative weights were assumed to represent overfitting; an iterative process was used to eliminate them. The filter with the most negative weight was eliminated; regression and elimination were repeated until no negative weights remained. For efficiency reasons, the weights were not recomputed for every message. For the first 100 messages, the weights were fixed at $\frac{1}{f}$, where f is the number of base filters. Thereafter, they were recomputed after every n_j messages where n_1, n_2, n_3, ... forms a Fibonacci-like series.[6]

Figure 4 shows the ROC curves for the four fusion methods and the best filter for each of the four corpora. Table 2 shows the summary statistics for the same runs, with 95% confidence limits and p-values. Each p-value indicates the probability that the statistic's improvement over that of the best filter may be due to chance.

6. SUBSET EXPERIMENT

To select subsets of the base filters, we employed the same elimination process as for logistic-regression stacking. After eliminating the filters corresponding to negative weights, we continued the process – eliminating the filter with the smallest weight – until only k filters remained. These k filters formed the base classifiers for a new fused filter. The resulting filter combines k spamminess scores by multiplying them by their respective weights as determined by the selection process. The subset experiment, unlike fusion, involved a batch process – selection and the computation of weights takes place with respect to a training corpus and the resulting filter is applied to a different test corpus.

To evaluate the subset selection method, we used two corpora – Mr X and S B – as training corpora, and the other two – Full and T M – as test corpora. For each test corpus we computed subsets of size 2, 3, 4, 8, 16, ..., m where m is the largest subset that yields all positive coefficients. Each subset was used in a fusion run on the two test corpora. Tables 3 and 4 show the results of these four sets of runs. All subsets improve on the best run in both measures, significantly so except for the smaller subsets trained on the S B corpus. Performance improves with subset size; performance of the larger subsets is comparable to that of the better fusion methods.

[5] Increasing training set sizes were used to adapt SVM, a batch method, to on-line classification [6]. We used training set sizes of *0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, 10000, 20000, 50000.*

[6] We used increasing training set sizes of *0, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 2100, 4100, 9100, 19100, 39100, 69100, 99100, 129100, 159100.*

Subset	$(1-ROCA)\%$	$sm\%@hm\% = .1$
mrx23	.007*** (.006-.009)	.79*** (.62-.99)
mrx16	.007*** (.006-.009)	.84*** (.69-1.02)
mrx8	.009*** (.007-.011)	.88*** (.71-1.08)
mrx4	.012*** (.009-.015)	1.07*** (.82-1.39)
mrx3	.012*** (.010-.016)	1.15*** (.92-1.44)
mrx2	.016 (.012-.021)	1.31** (1.01-1.68)
best	.019 (.015-.023)	1.78 (1.42-2.22)

Full Corpus

Subset	$(1-ROCA)\%$	$sm\%@hm\% = .1$
mrx23	.047*** (.038-.057)	3.84*** (3.41-4.32)
mrx16	.050*** (.040-.062)	3.99*** (3.56-4.48)
mrx8	.055*** (.041-.072)	4.22*** (3.72-4.79)
mrx4	.084*** (.067-.105)	4.37*** (3.74-5.09)
mrx3	.081*** (.063-.104)	4.20*** (3.66-4.81)
mrx2	.094*** (.075-.118)	4.40*** (3.90-4.96)
best	.135 (.111-.163)	10.3 (9.16-11.6)

T M Corpus

improvement on best: *$p < .05$, **$p < .005$, ***$p < .0005$

Table 3: Mr X-derived Subsets on Full and T M Corpora

Subset	$(1-ROCA)\%$	$sm\%@hm\% = .1$
sb14	.008*** (.007-.010)	1.01*** (.81-1.25)
sb8	.008*** (.007-.010)	1.02*** (.81-1.28)
sb4	.010*** (.008-.012)	1.40* (1.07-1.82)
sb3	.012*** (.010-.015)	1.45* (1.22-1.73)
sb2	.015*** (.012-.018)	1.51 (1.23-1.84)
best	.019 (.015-.023)	1.78 (1.42-2.22)

Full Corpus

Subset	$(1-ROCA)\%$	$sm\%@hm\% = .1$
sb14	.049*** (.041-.059)	5.50*** (4.83-6.27)
sb8	.053*** (.044-.063)	5.78*** (5.01-6.66)
sb4	.058*** (.048-.069)	6.09*** (5.21-7.11)
sb3	.074*** (.061-.089)	7.72*** (6.60-9.00)
sb2	.109** (.087-.136)	8.80*** (7.58-10.18)
best	.135 (.111-.163)	10.3 (9.16-11.6)

T M Corpus

improvement on best: *$p < .05$, **$p < .005$, ***$p < .0005$

Table 4: S B-derived Subsets on Full and T M Corpora

7. ANALYSIS AND DISCUSSION

All fusion methods substantially outperformed the best filter. The lack of significance of results with respect to the S B corpus may be attributed to its size; 775 spam messages are insufficient to distinguish filters at the error rates achieved. It may also be the case that some effects (notably SVM and logistic-regression stacking) increase with corpus size. Voting – simply counting the binary classification outputs of the filters – is remarkably effective, but appears to yield somewhat less improvement than the other filters. On the other hand, we have reason to believe that voting is more stable, and may perform better on short corpora, or on the first several thousand messages of long corpora. One possible reason for this is that voting is better able to take advantage of prior knowledge incorporated into the individual filters; until reliable estimates of the filters' credibility are obtained, simple voting seems to be the safest choice. Nevertheless, given the diversity of performance among the base filters, it is remarkable that a simple vote works so well. Each filter no doubt incorporates several arbitrary parameters set by its authors, not the least important of which is t, the classification threshold. Thus, voting works well due to social behaviour as much as any technical reason.

The log-odds transformation is an essential component of the other techniques – the transformed scores were used directly and also as input to the SVM and logistic regression meta-learning methods. In the pilot experiment we investigated various linear and non-linear combinations of scores. Although the sum of linear-normalized scores worked acceptably well in the pilot, we had no confidence that it would combine well the diverse score distributions found in the TREC runs. Indeed it did not, performing more poorly than simple voting on the Mr X Corpus. Therefore we dropped it from further consideration and did not test it on the other corpora. Since we had used Mr X in the pilot (but with different filters) we used it for testing various parameters and methods, testing only the ones that appeared promising – the ones reported here – on the other corpora. In this sense one may consider the Mr X results to be somewhat "cherry picked" but not the results on the other corpora.

The rationale for the log-odds transformation is as follows. Given a threshold t, messages may be placed in two dichotomous classes: spam messages with spamminess score $s \leq t$, and non-spam messages with $s \geq t$. A new message with spamminess t must necessarily fall into one of these classes. We use the observed size of these classes as an estimate of the odds ratio. That is, the area of the tails of the unnormalized score distributions provides a likelihood ratio multiplied by the prior odds (i.e. the overall odds ratio). We also experimented with using log-likelihood instead of log-odds. Log-likelihood is computed by subtracting log-prior-odds from log-odds; log-prior-odds is easily estimated from the observed spam to non-spam ratio. While log-likelihood makes more "sense" from a probabilistic point of view, it makes no difference to ROC or logistic regression results, and introduces slightly more noise due to the (additional) instability of the log-prior-odds estimate. In addition, we computed positive or negative log likelihood ratios [1] (as appropriate) from the base filters' binary classifications; preliminary testing revealed the average of these works marginally better than voting, but not as well as the average of the log-odds-transformed scores.

Three of the corpora showed better results for log-odds averaging than for voting; two were significant in a 2-tailed test (full, $p < .0002$; mrx, $p < .2$; tm, $p < .0001$), one showed an inferior (sb, $p < .16$) result which we suggest is largely due to chance, but may also be due to the small size of the corpus offering insufficient numbers for accurate log-odds estimates. The aggregate "run", which is not a run at all but an amalgam of the other four, shows that log-odds averaging improves on voting ($p < .0001$).

The log-odds transformed scores were used as input features to SVMlight. We also tried the untransformed scores and the binary classifications as features, with deleterious results. We also tried several combinations of kernels and parameter settings, but found none that yielded better results. We do not claim to have the exhausted space of features, kernels and settings. SVMlight, using default parameters, improves on voting on the same corpora as does log-odds, and shows a significant improvement in the aggregate ($p < .0001$). While SVM's improvement over log-odds is significant only for the aggregate run ($p < .01$), the consistent improvement over the four corpora leads us to believe that it is better.

We found that straightforward logistic regression yielded poor performance, even with very large amounts of training data. We observed, as did Hull [13] in a somewhat different context, that negative coefficients were a near-certain sign of over-fitting[7]. But logistic regression constrained to non-negative results is intractable, so we used the simple heuristic of deleting the filter with the most negative coefficient and repeating until no negative coefficients remained. There is no reason to believe that this is the best approach. For example, we could have used significance rather than magnitude as an elimination criterion. But for efficiency we chose a simplistic technique that appeared to work. We leave it to future research to investigate more sophisticated strategies.

Logistic regression performed the best on all corpora except S. B.; significantly better than the other methods in the aggregate (vote, $p < .0001$; logodds, $p < .0001$; svm, $p < .0001$). S. B.'s discordant result is not significant and may be due to chance. Examination of the ROC curve (figure 2) shows the logistic regression curve apparently superior to the rest, yet the (1-ROCA)% statistic is inferior. Further investigation, and verification of the ROC results with SPSS, shows that an extreme point beyond the scale of the graph accounts for the difference. We note also that $sm\%$ at $hm\% = .1$ shows logistic regression to be superior on the S. B. corpus. While the difference may be due to chance, it is also plausible that stacking methods are superior only on larger corpora, where they have more opportunity to learn.

The stepwise elimination process embodied in the logistic regression approach identifies a subset of the base filters that contribute to the best fusion result. Continuing the elimination process yields smaller subsets which all outperform the best filter; even the subsets of size 2 outperform the best individual filter. Figure 5 indicates the number of distinct Mr X-derived subsets in which each filter participates; the filters are labelled and ordered by their individual performances. We note that the best-performing filter is not a member of any of the subsets – many strong filters are excluded in favour of weaker ones. The S B-derived subsets show the same effect, from which we may infer that inter-filter correlation is a determining factor in subset selection.

The cross-corpus design of the experiments serves to indicate that a subset of filters chosen using one source of email may be expected to yield a fused filter that works well on another.

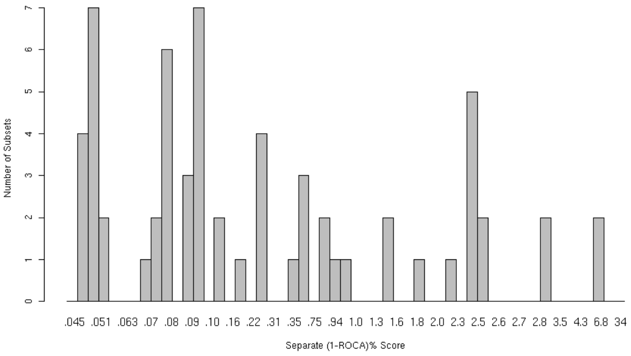


Figure 5: Base Filter Participation in Subsets (by Separate Performance)

8. CONCLUSIONS

The fusion methods presented here produce combined filters that outperform all other tested filters by a substantial margin – more than a factor of two in the standard measures of ROC area and spam misclassification at a 0.1% ham misclassification rate. As such, they are the best filters tested to date on the TREC corpora.

The simplest method – voting based on the binary classifications yielded by the individual filters – yields an ROC curve that is clearly superior to the best filter on each of the corpora. Although voting works well, it lacks appeal because it relies on the arbitrarily-set classification thresholds of the individual filters, and its sensitivity can be adjusted only coarsely by specifying the number of filters that must agree to classify a message as spam. The fifty-three different threshold values afforded by this test were adequate to achieve good ROC results, but we are skeptical as to whether the approach would be practical for a smaller number of filters, unless one had the capability to adjust the individual filters' thresholds.

The score-based methods – log-odds averaging, SVM, and logistic regression – are more appealing in that they use the score and not the threshold setting from each individual filter. The score-based methods appear also to improve on voting, but the incremental improvement is not nearly as dramatic as that of voting over the best individual filter. The ROC curves for these methods don't clearly dominate voting, and the statistics are superior by a significant margin on only the larger corpora. Of these methods, logistic regression (with elimination of filters with negative coefficients) appears to yield the best performance. On the other hand, log-odds averaging is the simplest of the score-based methods, and the other methods take as input the log-odds transformed scores. That is, the log-odds transformation is the essential basis of all the score-based methods.

In practice, it may not be feasible to run 53 separate filters on each incoming email message. Our experiments indicate that it is possible to select a smaller number – roughly half – without compromising performance. Smaller subsets – perhaps only a handful of filters – compromise performance only slightly. Furthermore, it appears that these subsets may be picked a priori, based on a training corpus derived from a distinct source of email.

[7] We say near-certain because the process did in fact discover some valid negative coefficients. Two of the base filters were fusions of other filters, and the regression process yielded a strong negative coefficient for components that were over-represented.

These experiments may be repeated using the TREC public corpus and the open-source filters supplied with the spam evaluation toolkit. The 53 filters tested at TREC include many of the best available filters at the time of writing, as well as several experimental and less-well-performing filters. We advance the hypothesis that as new filters are developed and tested, they too will perform best in combination with other independently-developed filters.

References

[1] Attia, J. Moving beyond sensistivity and specificity: using likelihood ratios to help interpret diagnostic tests. *Australian Prescriber 26*, 5 (2003), 111–113.

[2] Bartell, B. T., Cottrell, G. W., and Belew, R. K. Automatic combination of multiple ranked retrieval systems. In *SIGIR Conference on Research and Development in Information Retrieval* (1994), pp. 173–181.

[3] Belkin, N. J., Kantor, P., Fox, E. A., and Shaw, J. A. Combining the evidence of multiple query representations for information retrieval. In *TREC-2: Proceedings of the second conference on Text retrieval* (Gaithersburg, 1995), NIST, pp. 431–448.

[4] Bennett, P. N., Dumais, S. T., and Horvitz, E. The combination of text classifiers using reliability indicators. *Inf. Retr. 8*, 1 (2005), 67–100.

[5] Bentley, J. L., and Friedman, J. H. Data structures for range searching. *ACM Comput. Surv. 11*, 4 (1979), 397–409.

[6] Cormack, G. V., and Bratko, A. Batch and on-line spam filter evaluation. In *CEAS 2006 – The 3rd Conference on Email and Anti-Spam* (Mountain View, 2006).

[7] Cormack, G. V., and Lynam, T. R. Overview of the TREC 2005 Spam Evaluation Track. In *Fourteenth Text REtrieval Conference (TREC-2005)* (Gaithersburg, MD, 2005), NIST.

[8] Cormack, G. V., and Lynam, T. R. Statistical precision of information retrieval evaluation. In *29th ACM SIGIR Conference on Research and Development on Information Retrieval* (Seattle, 2006).

[9] Dietterich, T. G. Ensemble methods in machine learning. *Lecture Notes in Computer Science 1857* (2000), 1–15.

[10] Dzeroski, S., and Zenko, B. Is combining classifiers with stacking better than selecting the best one? *Mach. Learn. 54*, 3 (2004), 255–273.

[11] Fawcett, T. ROC graphs: Notes and practical considerations for researchers. Tech. Rep. HPL-2003-4, HP Laboratories, 2004.

[12] Gosh, J. Multiclassifier systems: Back to the future. In *Multiple Classifier Systems (MCS2002)* (2002), J. Kittler and F. Roli, Eds., vol. LNCS 2364, pp. 1–15.

[13] Hull, D. A., Pedersen, J. O., and Schutze, H. Method combination for document filtering. In *SIGIR '96: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval* (1996), ACM Press, pp. 279–287.

[14] Joachims, T. Making large-scale support vector machine learning practical. In *Advances in Kernel Methods: Support Vector Machines*, A. S. B. Scholkopf, C. Burges, Ed. MIT Press, Cambridge, MA, 1998.

[15] Kittler, J., Hatef, M., Duin, R. P. W., and Matas, J. On combining classifiers. *IEEE Trans. Pattern Anal. Mach. Intell. 20*, 3 (1998), 226–239.

[16] Komarek, P., and Moore, A. Fast robust logistic regression for large sparse datasets with binary outputs. In *Artificial Intelligence and Statistics* (2003).

[17] Lam, W., and Lai, K.-Y. A meta-learning approach for text categorization. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval* (2001), ACM Press, pp. 303–309.

[18] Lewis, D. D., Schapire, R. E., Callan, J. P., and Papka, R. Training algorithms for linear text classifiers. In *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval* (Zürich, CH, 1996), H.-P. Frei, D. Harman, P. Schäuble, and R. Wilkinson, Eds., ACM Press, New York, US, pp. 298–306.

[19] Lynam, T., and Cormack, G. TREC Spam Filter Evaluation Took Kit. http://plg.uwaterloo.ca/~trlynam/spamjig.

[20] Lynam, T. R., Buckley, C., Clarke, C. L. A., and Cormack, G. V. A multi-system analysis of document and term selection for blind feedback. In *CIKM '04: Thirteenth ACM conference on Information and knowledge management* (2004), pp. 261–269.

[21] Montague, M., and Aslam, J. A. Condorcet fusion for improved retrieval. In *CIKM '02: Eleventh international conference on Information and knowledge management* (2002), pp. 538–548.

[22] Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C. D., and Stamatopoulos, P. Stacking classifiers for anti-spam filtering of e-mail, 2001.

[23] Sebastiani, F. Machine learning in automated text categorization. *ACM Computing Surveys 34*, 1 (2002), 1–47.

[24] Segal, R., Crawford, J., Kephart, J., and Leiba, B. SpamGuru: An enterprise anti-spam filtering system. In *First Conference on Email and Anti-Spam (CEAS)* (2004).

[25] Shaw, J. A., and Fox, E. A. Combination of multiple searches. In *Text REtrieval Conference* (1994).

[26] Voorhees, E. *Fourteenth Text REtrieval Conference (TREC-2005)*. NIST, Gaithersburg, MD, 2005.

[27] Wolpert, D. H. Stacked generalization. *Neural Networks 5* (1992), 241–259.

[28] Zhang, Y. Using Bayesian priors to combine classifiers for adaptive filtering. In *SIGIR '04: The 27th Conference on Research and Development in Information Retrieval* (2004), pp. 345–352.