On-line Supervised Spam Filter Evaluation

GORDON V. CORMACK and THOMAS R. LYNAM
University of Waterloo

Eleven variants of six widely used open-source spam filters are tested on a chronological sequence of 49086 email messages received by an individual from August 2003 through March 2004. Our approach differs from those previously reported in that the test set is large, comprises uncensored raw messages, and is presented to each filter sequentially with incremental feedback. Misclassification rates and Receiver Operating Characteristic Curve measurements are reported, with statistical confidence intervals. Quantitative results indicate that content-based filters can eliminate 98% of spam while incurring 0.1% legitimate email loss. Qualitative results indicate that the risk of loss depends on the nature of the message, and that messages likely to be lost may be those that are less critical. More generally, our methodology has been encapsulated in a free software toolkit, which may used to conduct similar experiments.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*information filtering*; H.4.3 [**Information Systems Applications**]: Communications Applications—*electronic mail*

Additional Key Words and Phrases: spam, email, text classification

1. INTRODUCTION

We report the comparative evaluation, in a realistic controlled environment, of commonly-deployed spam filters applied to a sequence of all email delivered to an individual within a specific time interval. Our study advances the methodology, scale, realism and repeatability of spam filter evaluation. The specific results obtained may be taken as an indicator of the utility of spam filters. As with any scientific study, generalizability of these results depends on the extent that the subjects of our study – the filters and the email sequence – are typical. Such generalizability is established by repeated independent but comparable experiments with different subjects or circumstances. To this end, we have embodied our methods in a free toolkit [Lynam & Cormack [Lynam and Cormack 2005]] to be used in future studies. We also maintain an archive of the email corpus used in this study, and undertake to evaluate, on request, new filters with respect to this corpus.

Our approach is novel in that it closely models real filter usage, presenting to the filter a large sequence of real email messages, one at a time in chronological order, for classification. The same sequence of messages, under exactly the same conditions, is presented to several filters for the purpose of comparative analysis. Measures are computed which, we argue, reflect a filter's effectiveness for its intended purpose; i.e. abating spam while preserving welcome email messages. Statistical confidence

Authors' address: David R. Cheriton School of Computer Science, University of Waterloo, Waterloo ON N2L 3G1, Canada.

intervals, which estimate the extent to which the measured results might be due to chance, are computed for all measures.

Previous studies have used diverse test methods and evaluation measures, rarely including statistical analysis. We contrast these studies with ours and, to enhance inter-study comparability, we recast their results according to our measures, with confidence intervals.

2. ON-LINE SUPERVISED SPAM FILTERING

Unwelcome email is inconvenient, annoying and wasteful. Its volume threatens to overwhelm our ability to recognize welcome messages. An automatic spam filter can mitigate these problems, provided that it acts in a reliable and predictable

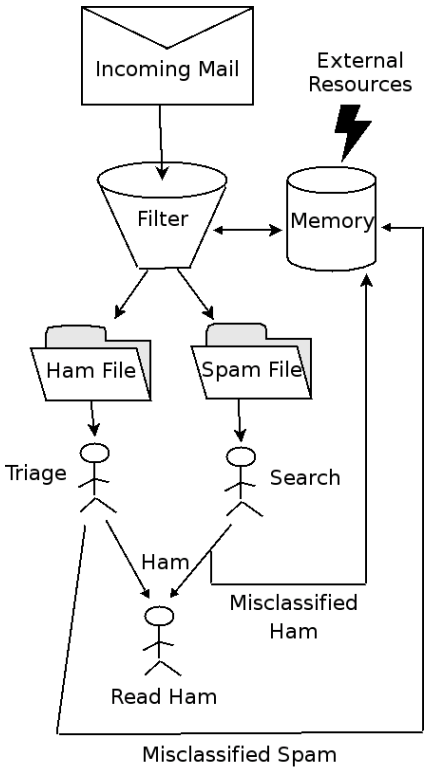

Fig. 1. Spam Filter Usage

manner, eliminates a large proportion of unwelcome email, and poses minimal risk of eliminating welcome email.

Figure 1 models spam filter deployment and use as it relates to an individual email recipient. Messages from an incoming email stream are presented to the spam filter, which classifies each as good email (*ham*) or as indiscriminately sent unwanted email (*spam*). Messages classified as ham are placed in the recipient's mailbox (*ham file*) or quarantined or discarded (placed in the *spam file*). The

recipient, in the normal course of reading the ham file, may notice spam messages (which have been misclassified by the filter) and may provide feedback to the filter noting these errors. From time to time the recipient may search the spam file for ham messages (which have also been misclassified) and may provide feedback on these errors as well. The filter may avail itself of *external resources* such as global statistics, collaborative judgements, network information, black lists, and so on.

A perfect spam filter would avoid *ham misclassification* – incorrectly placing a ham message in the spam file – and *spam misclassification* – incorrectly placing a spam message in the ham file. Ham misclassification poses a serious risk; were the recipient to fail to retrieve the misclassified message from the spam file, it would be lost. The expected cost to the user is some combination of the probability of misclassifying a particular ham message, the likelihood of retrieving it from quarantine, and the value of the message. Spam misclassification, on the other hand, exposes to the recipient a degree of the original inconvenience, annoyance and risk associated with spam. An effective filter should mitigate the effects of spam while maintaining an acceptable risk of loss.

3. STUDY DESIGN

It is difficult to quantify the risks and costs associated with ham and spam misclassification [Fawcett [Fawcett 2003a]; Kolcz & Alspector [Kolcz and Alspector 2001]]. For this reason, we use as our principal effectiveness measures the ham misclassification rate (hm) and the spam misclassification rate (sm), which are simply the fraction of ham messages that are misclassified and the fraction of spam messages that are misclassified. These measures, combined with estimates of the risks and costs external to the filter, allow us to estimate the degree to which a filter, in a particular situation, fulfills its intended purpose.

Because the relative risks and costs associated with ham and spam misclassification may vary from one situation to another, most spam filters, in addition to classifying each email message, have a *threshold* parameter which may be adjusted to decrease hm at the expense of sm, or vice versa. We use Receiver Operating Characteristic Curves (ROC) to assess the impact of this tradeoff, and the area under the curve (AUC) as a summary measure over all possible threshold settings [cf. Fawcett [Fawcett 2003b]; Flach [Flach 2004]; Park et al. [Park et al. 2004]].

Each filter configuration was tested in a laboratory environment simulating the usage characterized by our model. In the interest of repeatability, we made two simplifying assumptions. We assumed[1] that no filter used time-varying external resources, so that email messages captured at one time would be classified the same way later. We idealized the recipient's behaviour by assuming that he or she accurately and immediately reported all misclassified messages to the filter.

We captured all of the email received by one individual over a period of time. These messages were presented, one at a time, in chronological order, to each filter for classification. In addition, we extracted from each filter a *spamminess score*, indicating the filter's estimate of the likelihood that the classified message was

[1]These assumptions apply to this particular study and are not entrenched in our test and evaluation methods, or the toolkit that implements them. The toolkit may therefore be used in other studies that admit external or time-varying resources or less-than-ideal recipient behavior.

spam. Immediately thereafter, a *gold standard* classification for each message was reported to the filter. The filter's classification, the filter's spamminess score, and the gold standard classification were recorded for later analysis. Summary statistics were derived from the results, assuming the final gold standard to be ground truth. Confidence intervals were computed for each measure, under the assumption that the test email sequence sampled an infinite hypothetical population of materially similar email[2].

Human adjudication is a necessary component of gold standard creation. Exhaustive adjudication is tedious and error-prone; therefore we use a bootstrap method to improve both efficiency and accuracy [Cormack & Lynam [Cormack and Lynam 2005a]]. The bootstrap method begins with an initial gold standard G_0. One or more filters is run, using the toolkit and G_0 for feedback. The evaluation component reports all messages for which the filter and G_0 disagree. Each such message is re-adjudicated by the human and, where G_0 is found to be wrong, it is corrected. The result of all corrections is a new standard G_1. This process is repeated to form G_2, and so on, until $G_n = G_{n+1}$.

4. TEST CORPUS

We captured the email received by one individual (X) from August 2003 through March 2004. These 49,086 messages were initially classified in real-time by SpamAssassin 2.60 [[SpamAssassin 2005]] and placed in X's ham and spam files. X regularly examined both files and reported misclassification errors to SpamAssassin. G_0 consisted of the judgements rendered by SpamAssassin, amended to correct all misclassification errors reported by X.

X has had the same userid and domain name for 20 years; variants of X's email address have appeared on the Web, and in newsgroups. X has accounts on several machines which are forwarded to a common spool file, where they are stored permanently in the order received.

X began using a spam filter in 2002 when the proportion of spam in his email began to exceed 20%, causing X to overlook two important messages which arrived amongst bursts of spam. Since August 2003, X has used SpamAssassin 2.60 in a supervised configuration to classify this incoming mail. It was necessary to modify SpamAssassin to incorporate this use, as SpamAssassin was designed to be used primarily in an unsupervised configuration. User feedback was facilitated by two macros added to X's mail client. SpamAssassin records every judgement (rendered automatically and amended to reflect user feedback) in its learning database, so it was possible to recover our preliminary gold standard judgements from this database.

Each trial run is an idealized[3] reproduction of X's behaviour from August 2003 to March 2004, with a different filter in place of SpamAssassin 2.60. The subject filter is presented with each message, with original headers, in the same order as originally delivered. Each filter was encapsulated using three common interface

[2]The notion of *population* has been the subject of historical and current philosophical debate [Lenhard [Lenhard 2006]]. We adopt Fisher's view [[Fisher 1925]] of an infinite hypothetical population.

[3]Idealized in that feedback to the filter is immediate and completely accurate.

procedures: *filterinit, filtereval,* and *filtertrain. filterinit* sets the filter's memory to a clean initial state; *filtereval* is given an email message and returns a pair consisting of a classification and a spamminess score; *filtertrain* is given an email message and the gold standard classification. Some filters require that *filtertrain* be invoked for every message (*train on everything*) while others require that *filtertrain* be invoked only for misclassified messages (*train on error*). We used the method suggested by each filter's documentation, as detailed in the next section.

All the filters were used in the bootstrap construction of G_5, the final gold standard. The net effect is that every message reported as a ham or spam misclassification for any filter has been adjudicated by X.

To facilitate further analysis, we categorized messages – both ham and spam – into *genres* which may predict risk or cost of misclassification. For example, we suggest that individually addressed messages and news digest messages, while both ham, may present different levels of challenge to the filter and also different costs to the recipient, were they to be lost. After the filter tests were complete, each misclassified ham message was examined and assigned one of seven genres that we believed might be associated with the likelihood of misclassification and the importance of the email to the recipient. We also assigned a genre to each of a random sample ($n = 352$) of all incoming ham. Similarly, we assigned one of five different genres to each spam message misclassified by one or more of the four best-performing systems, and also to a random sample of spam messages ($n = 100$) misclassified by each of the other systems. We also assigned a genre to each of a random sample ($n = 142$) of all incoming spam.

5. SUBJECT FILTERS

In February 2004, we selected the current versions of six open-source filters whose deployment had been widely reported on the internet and in the popular press. Although a large number of classification techniques potentially relevant to spam filtering have been reported in the literature, an extensive search of available practical email filters yielded filters that used only a limited number of techniques, which we characterize as hand-coded rule bases, internal or external black lists and white lists, and content-based 'statistical' or 'Bayesian' filters owing their heritage to Graham's *A Plan for Spam* [[Graham 2002; 2004]] with improvements due to Robinson [[Robinson 2004; 2003]].

Other machine learning methods have not, to our knowledge, been deployed in any practical filter amenable to our evaluation [cf. Cormack and Bratko [Cormack and Bratko 2006]]. Studies of machine learning methods typically model spam filtering as an off-line (batch) supervised learning task in which a hard binary classifier is induced on a set of labeled training messages and then used to predict the class (ham or spam) of each of a set of unlabeled test messages. Many of these studies further abstract the messages to feature vectors, eliminating patterns and other information that real filters may use to distinguish ham from spam. Although a number of techniques, such Support Vector Machines, fare very well in these off-line evaluations, we were simply unable to find them deployed in any real filters available for testing.

Of the filters we selected, SpamAssassin [[SpamAssassin 2005]] is a hybrid system

which includes hand-coded spam-detection rules and a statistical learning component. The other filters – Bogofilter [Raymond et al. [Raymond et al. 2004]], CRM114 [Yerazunis [Yerazunis 2004b]], DSPAM [Zdziarski [Zdziarski 2004]], SpamBayes [Peters [Peters 2004]], and SpamProbe [Burton [Burton 2002]] – are all 'pure' statistical learning systems, with only a few tacit rules such as those for tokenization.

Five different configurations of SpamAssassin were tested, in order to evaluate the roles and interactions of its various components. These five configurations were compared with one another, and with the in situ performance of SpamAssassin, which was deployed when the email for the test corpus was collected. The five statistical learning systems were tested and compared as a separate group. One configuration of SpamAssassin – its learning component in isolation – was also included in this group.

In effect, the two groupings constitute separate experiments with separate goals; to evaluate combinations of rule-based and statistical filtering, and to evaluate statistical filters with similar heritage. The only filter in common between the two groups is SpamAssassin's learning component.

5.1 The SpamAssassin Runs

SpamAssassin contains two principal components: a set of static ad hoc rules that identify patterns associated with spam, and a Bayes filter fashioned from Graham's and Robinson's proposals. Each ad hoc rule has a predetermined weight; the weights of features observed in a particular message are summed to yield a combined spamminess score. The Bayes filter, on the other hand, is adaptive – it uses statistics from previously-classified messages to estimate the likelihood that a particular message is spam. This likelihood estimate is converted to a (possibly negative) weight which is added to the ad hoc spamminess score. The overall score is compared to a fixed threshold; the message is classified as spam if the score exceeds the threshold.

We tested several configurations of SpamAssassin 2.63 so as to evaluate the relative contributions of the ad hoc and Bayes components, and to evaluate various training regimens for the Bayes filter.

SA-Supervised. SpamAssassin 2.63 (both components) with the default threshold value of 5.0.

SA-Nolearn. SpamAssassin 2.63 (ad hoc component only) with the default threshold of 5.0.

SA-Bayes. SpamAssassin 2.63 (Bayes component only) with a threshold of 0.0.

SA-Standard. SpamAssassin 2.63 (Standard configuration with no user feedback) with a threshold of 5.0. SpamAssassin is configured by default to be used in a situation, such as a mail server, where misclassification errors go unreported. To this end, it includes an internal mechanism to train the Bayes component automatically, based on the spamminess score rendered by the ad hoc component alone. *filtertrain* is never invoked.

SA-Unsupervised. SpamAssassin 2.63 (Unsupervised automated feedback.) *filtertrain* is invoked after every message, but with SpamAssassin's output classification rather than the gold standard; that is, its own judgement is fed back to itself as if it were the gold standard.

SA-Human. Real-world baseline. These are the initial classification results that we captured from the memory of X's SpamAssassin 2.60 configuration. As such, they represent the classifications rendered in situ by the spam filter, as amended in real time in response to misclassification errors reported by X. These results show the combined effectiveness of spam filter and recipient under real-world conditions.

5.2 Pure Learning Filter Runs

Except as noted, the learning filters were installed using default threshold and tuning parameters. Prior training was not used; *filterinit* initialized the filter's memory to the empty state.

CRM114 (version 20040328-Blame St. Patrick-auto.1). We trained the system only after misclassifications, as suggested in the documentation. We did not use the whitelist or blacklist facilities supplied with CRM114. Filter memory size was set at 10000001 *buckets* for both ham and spam.

DSPAM (version 2.8.3). DSPAM 2.8.3 self-trains on every message it classifies, and annotates the message with a signature that contains information necessary for it to reverse this self-training. We altered our test setup to supply this annotated message, rather than the original, to *filtertrain*. We did not use the *purge* facility, which reduces the size of the statistical table maintained by DSPAM.

Bogofilter (version 0.17.5). Bogofilter is a Bayes filter, like SpamAssassin's, modeled after the proposals by Graham and Robinson. Bogofilter emphasizes simplicity and speed.

SpamProbe (version 0.9h). A C++ Bayes filter inspired by Graham's proposal.

SpamBayes (version 1.061). A Python Bayes filter inspired by the proposals of Graham and Robinson.

SA-Bayes. SpamAssassin 2.63 (Bayes component only). From the SpamAssassin comparison group.

6. ANALYSIS

A contingency table (table I) enumerates the possible outcomes of applying a filter to a mail stream. The primary measures of interest are the *ham misclassification fraction,* $hm = \frac{c}{a+c}$, and the *spam misclassification fraction* $sm = \frac{b}{b+d}$. We also report the (overall) *misclassification fraction,* $m = \frac{b+c}{a+b+c+d}$, because it is equivalent

		Gold Standard: ham	Gold Standard: spam
Filter	ham	a	b
	spam	c	d

Table I. Contingency Table

to *accuracy* ($m = 1 - accuracy$)[4], which is commonly reported.

[4]We quantify misclassifications rather than accuracy so as to avoid presenting nearly equal numbers that represent large differences in performance. Graphical results are displayed using the logistic transformation, $logit(p) = log(\frac{p}{1-p})$, which maps the range $[0:1]$symmetrically to the range $-\infty:\infty$.

The result of applying a filter to a spam message is a dichotomous variable *result* taking on the value *ham* or *spam.* In our primary analysis, we estimate the probability of each outcome as a function of *true*, the true classification, also a dichotomous value taking on the value *ham* or *spam.* In particular, *hm* estimates $Pr(result = \text{spam} \,|\, true = \text{ham})$ and *sm* estimates $Pr(result = \text{ham} \,|\, true = \text{spam})$. *m*, on the other hand, estimates $Pr(result \neq true)$. These estimates are the result of three sets of Bernoulli trials; one for ham messages ($true = \text{ham}$), one for spam messages ($true = \text{spam}$), and one for all messages.

Each set of trials consists of n observations, x of which exhibit the truth value whose probability P is to be estimated. Given P and n, the probability of any particular value of x is determined exactly by the binomial distribution. The cumulative probability over all $t \leq x$ is the sum of $x+1$ discrete binomial probabilities. Since $x \leq n < 50,000$ for each of the three sets, we were able to calculate cumulative probabilities with minimal computational cost.

Given n and x, the maximum likelihood estimate for P is simply $\frac{x}{n}$. 95% confidence limits are computed as follows. When $x = 0$, the lower confidence limit is 0 and the upper confidence limit is the smallest P such that the cumulative binomial probability over all $t \leq x$ (i.e. the probability of $t = x = 0$) is less than 0.05. When $x > 0$, the lower confidence limit is the largest P such that the cumulative binomial probability over all $t \geq x$ is less than 0.025; the upper confidence limit is the smallest P such that the cumulative binomial probability over all $t \leq x$ is less than 0.025. Each P was computed using binary search.

Because all filters are applied to the same messages, we are able to use exact paired tests to evaluate differences that might not be apparent from comparing misclassification proportions. For a pair of filters, A and B, we count each of the four possible pairs of results when A and B are applied to the same message. Table II illustrates the four possible outcomes: a is the number of times that the filters both return the correct classification; d is the number of times they are both incorrect; b is the number of times B is correct but A is incorrect; c is the number of times A is correct but B is incorrect. a and d, the cases of agreement, do not differentiate the systems and may be ignored. b and c, the cases of disagreement, are the cases of interest. The disagreement cases constitute a set of Bernoulli trials

Filter A

Filter B	correct	incorrect
correct	a	b
incorrect	c	d

Table II. Matched-Pair Result Table

with $n = b + c$, $x = b$. Under the null hypothesis (that A and B exhibit the same performance), $P = 0.5$, and $E(x) = \frac{n}{2}$. Any non-zero difference $|x - \frac{n}{2}| > 0$ must be due either to chance or to the falsity of the null hypothesis. p, the chance probability, is computed as the sum of binomial probabilities for all t such that $|t - \frac{n}{2}| \geq |x - \frac{n}{2}|$.

In this study, we test several hypotheses. For those that are amenable to statistical inference we state confidence intervals and declare significant differences based

on the error probability $\alpha = 0.05$. As for any set of statistical inferences, whether from the same or separate studies, we must be aware that some results reported as significant will in fact be due to chance. According to Streiner [[Streiner 1986]]:

> Of course, most statistical analysis uses an α-level of 0.05, which means that there is one chance in 20 that they will conclude there is some difference when there isn't. This also means that of every 20 "significant" differences reported in the literature, one is wrong. Wonder which one it is!

That said, we shall avoid a discussion of the philosophy of statistics and defer to common practice.

A fallacy not admitted by common practice is to perform several hypothesis tests and to report only those yielding a "significant" result. If we perform n tests, we expect about αn of them to show $p < \alpha$, even if the null hypothesis holds in every case. On the other hand, only α of the tests would show $n \cdot p < \alpha$ under the null hypothesis; in other words, the chance of some test showing $n \cdot p < \alpha$ is α. Bonferroni correction captures this effect: when selected from a set of n tests, any test showing $n \cdot p < \alpha$ is significant with (Bonferroni corrected) $p < \alpha$. Bonferroni correction may be applied repeatedly using Holm's stepdown method [[Holm 1979]]: the result with smallest p is selected from the set; if it is significant after Bonferroni correction, the test is removed from the set and the process repeated with the remaining $n - 1$ tests. If the result with the smallest p is not significant, none of the remaining results is considered significant. When we rank the results of n tests, we are in effect performing $\frac{n(n-1)}{2}$ paired tests, which we correct using Holm-Bonferroni stepdown method.

Receiver operating characteristic (ROC) analysis [cf. Fawcett [Fawcett 2003b]; Flach [Flach 2004]; Park et al. [Park et al. 2004]] is used to evaluate the trade-off between ham and spam misclassification probabilities. Using each of the numerical scores returned by a given filter, we conduct a hypothetical run to determine the ham and spam misclassification fractions that would have resulted had that score been used as a threshold. The set of pairs *(hm, 1-sm)* resulting from the hypothetical runs define a monotone non-decreasing function that is plotted as an ROC curve. As a summary measure of the relationship between ham and spam misclassification fractions over all possible thresholds, we present $1 - AUC$, where AUC is the area under the ROC curve. $1 - AUC$ estimates the probability that a random spam message is (incorrectly) given a lower score than a random ham message. AUC estimates and 95% confidence intervals were computed using SPSS 12.

Logistic regression [cf. Agresti [Agresti 1996]] is used to evaluate the effect of the number n of messages processed on the probability P of ham or spam misclassification (i.e. the *learning curve).* P and n are assumed to be related by the formula $logit(P) =_{def} log(\frac{P}{1-P}) = \alpha + n\beta$ (alternatively, $\frac{P}{1-P} = e^{\alpha}e^{n\beta}$) for some α and β. Maximum likelihood estimates for α and β, 95% confidence limits, and p-values (for the null hypothesis that $\beta = 0$) were computed using SPSS 12. $\frac{P}{1-P}$ is the *odds* (as opposed to the probability) of misclassification; i.e. the ratio of incorrect to correct classifications. e^{α} is the *initial odds* when $n = 0$, and $e^{n\beta}$ is the *odds ratio*; for every n messages the odds increase (or decrease) by a factor of $e^{n\beta}$. For

small P, odds and probability are nearly equal, so we may consider $e^{n\beta}$ also to be the *risk ratio;* for every n messages the probability of misclassification changes by this same constant factor.

A piecewise graphical estimate of $logit(P)$ vs. n is juxtaposed with the logistic regression curve as a visual indicator of the appropriateness of the logistic model. Estimates of initial and final misclassification rates, as well as the odds ratio, are tabulated with 95% confidence limits.

Within each genre of ham identified in our post-hoc classification, we estimated the proportion of incoming ham messages and, for each filter, the proportion messages misclassified by that filter. The ratio of these proportions provides an estimate of the relative difficulty that each filter has in classifying messages of different genres, and an estimate of the maximum likely confounding effect due to each particular genre.

7. RESULTS

The test sequence contained 49,086 messages. Our gold standard classified 9,038 (18.4%) as ham and 40,048 (81.6%) as spam. The gold standard was derived from X's initial judgements, amended to correct errors that were observed as the result of disagreements between these judgements and the various runs.

7.1 Classification Performance - SpamAssassin Variants

Table III and figure 2 report the performance of our SpamAssassin runs. *SA-Supervised*, our baseline run, misclassifies 6 of 9,038 ham messages (0.07%) and 605 of 40,048 spam messages (1.51%). Overall, SA-Supervised misclassifies 611 of 49,086 messages (1.24%). The area under the ROC curve, AUC, is 0.9994 which we report as 1-AUC (%) or 0.06.

Filter	Ham Misc. (%)	Spam Misc. (%)	Overall Misc. (%)	1-AUC (%)
SA-Supervised	0.07 (0.02-0.14)	1.51 (1.39-1.63)	1.24 (1.15-1.35)	0.06 (0.04-0.07)
SA-Bayes	0.17 (0.09-0.27)	2.10 (1.96-2.24)	1.74 (1.63-1.86)	0.15 (0.11-0.18)
SA-Nolearn	0.19 (0.11-0.30)	9.49 (9.21-9.78)	7.78 (7.54-8.02)	0.80 (0.74-0.86)
SA-Standard	0.07 (0.02-0.14)	7.49 (7.23-7.75)	6.12 (5.91-6.34)	1.00 (0.93-1.06)
SA-Unsupervised	0.11 (0.05-0.20)	8.11 (7.84-8.38)	6.63 (6.41-6.86)	0.82 (0.76-0.88)
SA-Human	0.09 (0.04-0.18)	1.06 (0.97-1.17)	0.88 (0.80-0.97)	-

Table III. Filter Misclassification - SpamAssassin Variants

The SA-Supervised filter is a committee of two distinct components: *SA-Nolearn,* a static rule-based filter, and *SA-Bayes*, a pure learning filter. Taken separately, each component shows inferior performance to the baseline according to all four measures. We note in particular that SA-Supervised shows 2.5 times fewer ham misclassifications than either SA-Bayes ($p < .004$) or SA-Nolearn ($p < .035$), two-thirds as many spam misclassifications as SA-Bayes ($p \approx 0.000$) and 6 times fewer spam misclassifications than SA-Nolearn ($p \approx 0.000$).

SA-Standard uses SpamAssassin's default configuration: the same static and learning filter, but with the filter trained only on errors, as adjudicated by the difference in results between the learning filter and a separate (more conservative) internal invocation of the static filter. In contrast, SA-Unsupervised trains on every

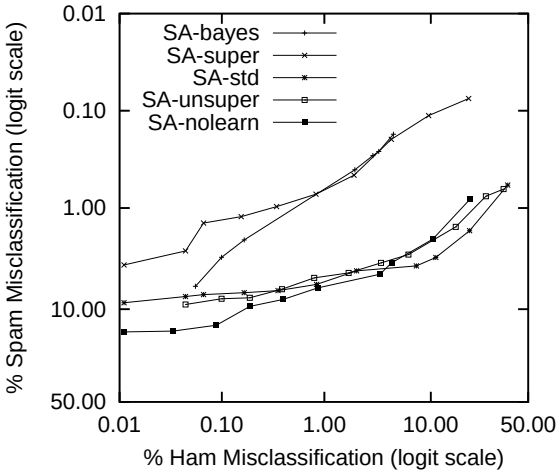

Fig. 2. ROC Curves - SpamAssassin Variants

judgement returned by *filtereval*. Both runs are unsupervised in that they operate autonomously with no human intervention. As with SA-Supervised, both runs show fewer ham and spam misclassifications than either SA-Bayes or SA-Nolearn taken separately. Of the differences in ham misclassifications only the difference between SA-Standard and SA-Nolearn may be interpreted as significant ($p < .035$). All differences in spam misclassification are significant ($p \approx 0.000$).

SA-Human uses essentially the same configuration as SA-Supervised, but the system was supervised by X in real-time. That is, for every misclassification observed by X, the system was retrained and the human-corrected classification was recorded as the result for SA-Human. While SA-Human resulted in two more ham misclassifications than SA-Supervised (i.e. 8 vs. 6) no significant difference can be inferred. SA-Human resulted in two-thirds as many spam misclassifications ($p \approx 0.000$).

We note that ham, spam, and overall misclassification rates rank the six runs in the same order. AUC inverts SA-Standard and SA-Unsupervised, and is inapplicable to SA-Human. Nevertheless, AUC ranking is consistent with the overall effect: that all tested combinations of static and learning filters outperform these individual components in isolation. The ROC curves show that SA-Supervised dominates the other runs, performing better than SA-Bayes when ham misclassification is minimized and as well when spam misclassification is minimized. SA-Supervised and SA-Bayes both dominate the remaining runs. These runs, SA-Nolearn, SA-Standard, and SA-Unsupervised, show ROC curves that intersect many times, indicating that their relative AUC scores are likely to be uninformative.

7.2 Classification Performance - Pure Learning Filters

Table IV and figure 3 show the classification performance of six pure learning filters (including SA-Bayes, the learning component of SpamAssassin, also reported above). For this group of runs we have no baseline, and wish instead to evaluate their relative performance. The columns labeled *ham misclassification* and *spam*

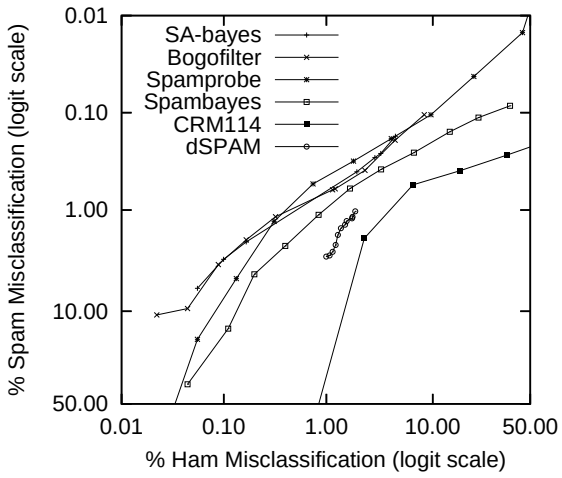

Fig. 3. ROC Curves – Pure Learning Filters

misclassification show nearly opposite effects. Bogofilter offers the least number

Filter	Ham Misc. (%)	Spam Misc. (%)	Overall Misc. (%)	1-AUC (%)
Bogofilter	0.08 (0.03-0.16)	6.63 (6.39-6.88)	5.43 (5.23-5.63)	0.08 (0.05-0.10)
SpamBayes	0.17 (0.09-0.27)	5.86 (5.63-6.10)	4.81 (4.63-5.01)	0.16 (0.12-0.20)
SA-Bayes	0.17 (0.09-0.27)	2.10 (1.96-2.24)	1.74 (1.63-1.86)	0.15 (0.11-0.18)
SpamProbe	0.34 (0.23-0.49)	1.03 (0.93-1.14)	0.90 (0.82-0.99)	0.09 (0.05-0.13)
DSPAM	1.28 (1.06-1.54)	1.98 (1.84-2.12)	1.85 (1.73-1.97)	1.03 (0.90-1.17)
CRM114	3.26 (2.91-3.65)	0.99 (0.90-1.09)	1.41 (1.31-1.52)	1.10 (0.94-1.27)

Table IV. Filter Misclassification - Pure Learning Filters

Ham Misc.	Spam Misc.	AUC
Bogofilter SpamBayes SA-Bayes	CRM114 SpamProbe	Bogofilter SpamProbe SA-Bayes SpamBayes
SpamProbe	DSPAM SA-Bayes	
DSPAM	SpamBayes	DSPAM CRM114
CRM114	Bogofilter	

Table V. Significant divisions ($p < .05$, Bonferroni-Holm corrected)

of ham misclassifications and the greatest number of spam misclassifications, while CRM114 shows the opposite.

To divide the filters into groups separated by significant differences classification performance, we considered ham and spam separately; for each we performed a paired test between every pair of runs. The first two columns in table V summarizes the results of these 30 tests, corrected using Holm's stepdown method. Every pair of runs from different boxes shows a significant different difference ($p < 0.05$),

while every pair in the same box does not. We see that the runs are divided into four groups with respect to ham classification performance, and four (different) groups with respect to spam classification performance. Although ham and spam misclassification performance yields nearly opposite rankings, we note that Bogofilter, SpamBayes, and SA-Bayes are distinguished by their spam performance but not by their ham performance. Similarly, CRM114 and SpamProbe; and also DSPAM and SA-Bayes, are distinguished by their ham performance but not by their spam performance.

Overall Misclassification results are largely reflective of spam misclassification results, and are not analyzed further. The ROC curves show that the curves for Bogofilter, SpamProbe, and SA-Bayes intersect one another in many places throughout the operating range, but SA-Bayes and Bogofilter appear to have a lower spam misclassification proportion when the ham misclassification proportion is low (i.e. less than 0.3%). All three dominate SpamBayes by a narrow margin and dominate DSPAM and CRM114 by substantial margins. AUC scores largely reflect the major differences observable in the curves, but fail to provide a meaningful distinction among Bogofilter, SpamProbe, SA-Bayes, and SpamBayes. The last column of table V shows that the filters may be separated into two groups such that every member of the upper group shows significantly better AUC than every member of the lower group ($p < .05$).

7.3 Effects of Learning on Classification Performance

Table VI summarizes the fraction of spam received by X as a function of the number of messages received. Although the overall spam fraction is 81.6%, logistic regression indicates that this fraction increased from 75.7% to 86.6% (an odds ratio of 2.07, $p < .001$) over the eight months during which our email stream was collected.

Initial Spam %	Final Spam %	Odds Ratio	p
75.7 (75.0, 76.6)	86.6 (86.0, 87.1)	2.07 (2.04, 2.10)	0.00

Table VI. Spam as a fraction of incoming messages

Figure 4 shows a piece-wise approximation of this function juxtaposed with the regression line.

Tables VII and VIII summarize the ham and spam misclassification fractions as functions of the number of messages processed. Each row estimates the initial misclassification proportion, the final misclassification proportion, and the odds ratio between the two. 95% confidence limits and p-values are given for each. Figures 5 and 6 provide graphical representations of these functions.

Of particular interest is the "learning" performance of SA-Nolearn; as this system has no learning component, its performance may be used to gauge any change in 'difficulty' of the spam messages over the eight months. Table VIII shows that SA-Nolearn's spam misclassification fraction increases from 7.73% to 11.37% ($p < .001$), indicating that the nature of spam has changed so as to make it 'more difficult.' Figure 5 confirms this trend, but also shows anomalous spikes in misclassifications centered at about 6,000 and 17,000 messages. SA-Nolearn's ham misclassification fraction shows no significant slope over the eight-month interval.

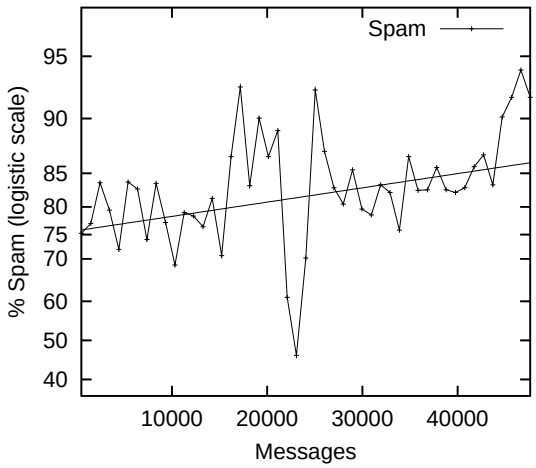

Fig. 4. Spam Growth

In contrast, the learning filters show no apparent degradation in performance over the eight-month interval. All learning filters show a reduction in both ham and spam misclassification fractions as more messages are processed, though not all reductions are large or significant. In particular, confidence intervals for the ham misclassification odds ratio are very large, due to the fact that the curve is fitted to few points – of the order of ten for the better-performing runs. Subject to this caveat, The plotted curves show a good fit between piecewise approximation and logistic regression. Possible exceptions are DSPAM, SA-Standard, and SA-Unsupervised. DSPAM's spam misclassification curve, shown in figure 6, has a piecewise approximation that appears to be more concave than the regression curve. SA-Standard and SA-Unsupervised (figure 5) both indicate substantially lower spam misclassification rates prior to the virus-induced anomaly at message 6,000, followed by consistent improvement notwithstanding the anomaly[5] at message 17,000. We observe that the initial misclassification fraction of a number of systems is substantially better than the final misclassification fraction of others.

We included SA-Human in our analysis, as a real-world foil to our laboratory results. SA-Human's ham misclassification fraction shows a large significant increase with a huge confidence interval [odds ratio 54 $(2, 1222)$], indicating that this measurement is unstable, rather than that X suffered some degeneration in discriminatory ability. Further investigation reveals that the positive odds ratio may be accounted for entirely by three automated (but legitimate) messages received the same day from the same source. SA-Human's apparent decrease in spam misclassification may also be accounted for by the anomalous spike at 17,000 messages.

7.4 Misclassification by Genre

In the course of examining the misclassified messages, we identified several message genres that we suspect might be associated with the filters' performance. Ham

[5]Due to *backscatter*, as defined in section 7.4.

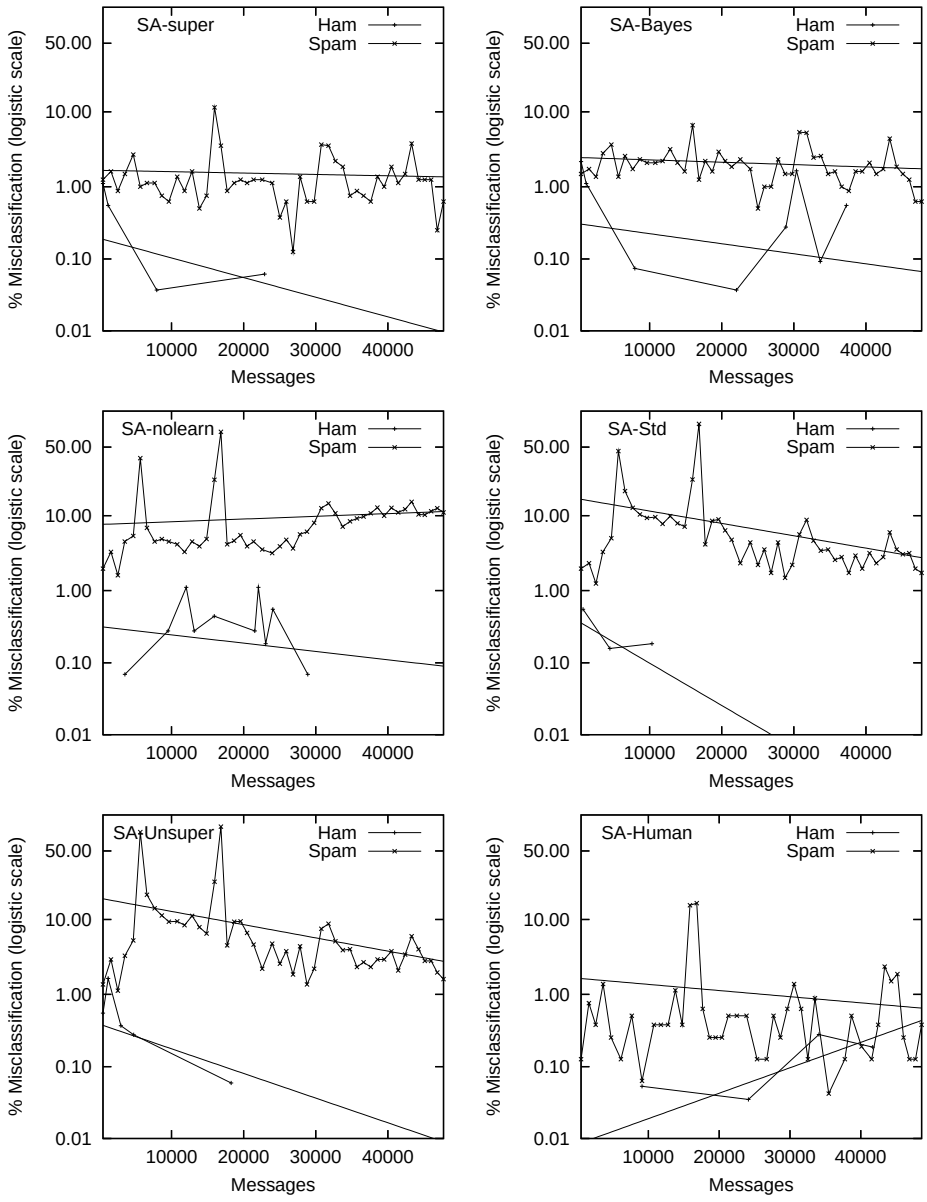

Fig. 5. Learning Curves – SpamAssassin Configurations

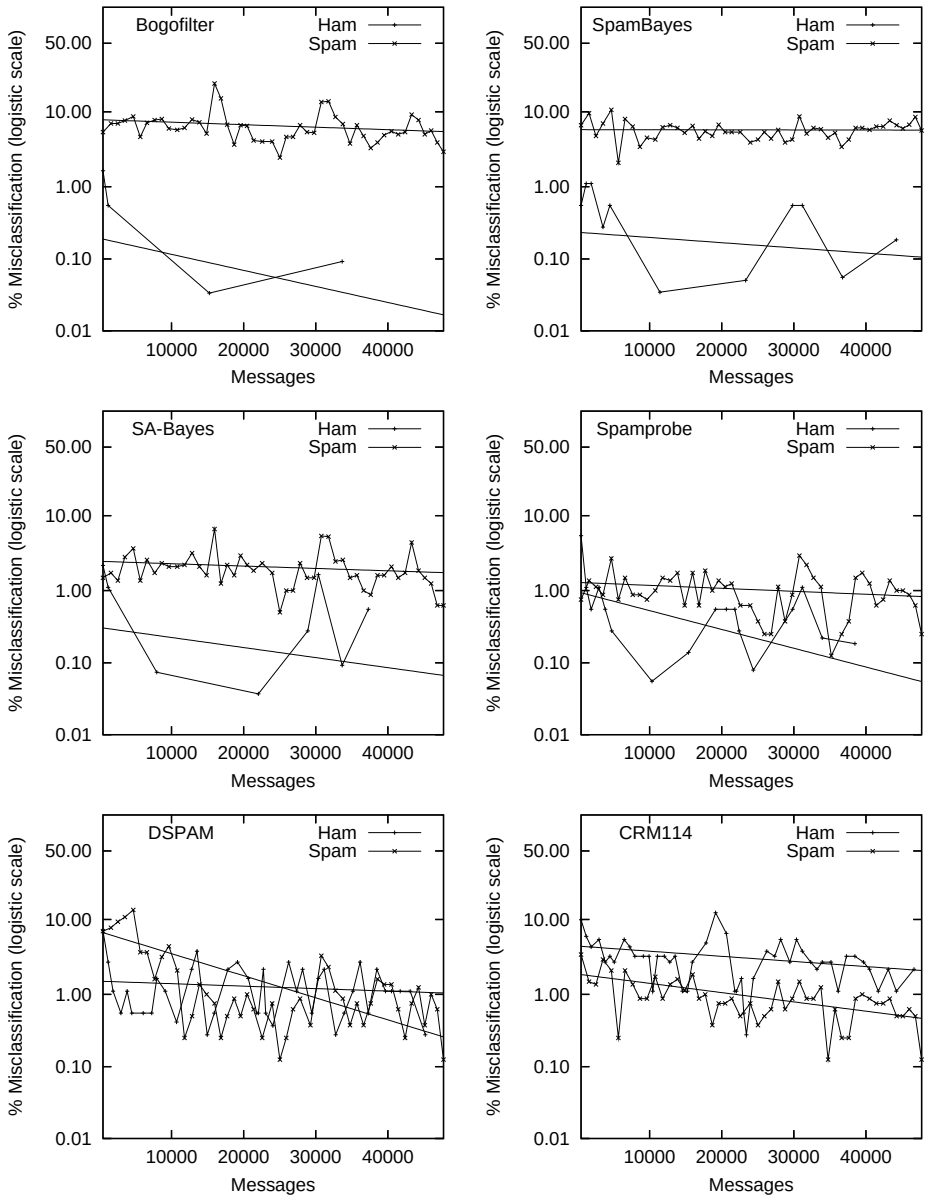

Fig. 6. Learning Curves – Pure Statistical Filters

Filter	Initial Misc. (%)	Final Misc. (%)	Odds Ratio	p
Bogofilter	0.19 (0.06, 0.62)	0.02 (0.00, 0.17)	0.08 (0.00, 1.98)	0.12
CRM114	4.53 (3.71, 5.52)	2.08 (1.56, 2.75)	0.45 (0.29, 0.69)	0.00
DSPAM	1.52 (1.09, 2.12)	1.03 (0.67, 1.58)	0.68 (0.35, 1.33)	0.26
SA-Bayes	0.31 (0.13, 0.72)	0.06 (0.02, 0.26)	0.21 (0.03, 1.52)	0.12
SA-Human	0.01 (0.00, 0.09)	0.45 (0.15, 1.38)	54 (2, 1222)	0.01
SA-Nolearn	0.32 (0.14, 0.71)	0.09 (0.02, 0.31)	0.27 (0.04, 1.72)	0.17
SA-Standard	0.38 (0.12, 1.19)	0.00 (0.00, 0.07)	0.00 (0.00, 0.40)	0.02
SA-Supervised	0.19 (0.06, 0.66)	0.01 (0.00, 0.15)	0.05 (0.00, 1.80)	0.10
SA-Unsupervised	0.39 (0.15, 0.98)	0.01 (0.00, 0.10)	0.02 (0.00, 0.47)	0.01
SpamBayes	0.23 (0.10, 0.58)	0.10 (0.03, 0.37)	0.44 (0.07, 2.96)	0.40
SpamProbe	0.96 (0.56, 1.65)	0.05 (0.01, 0.17)	0.05 (0.01, 0.26)	0.00

Table VII. Ham Learning Performance

Filter	Initial Misc. (%)	Final Misc. (%)	Odds Ratio	p
Bogofilter	7.95 (7.41, 8.53)	5.50 (5.10, 5.94)	0.68 (0.59, 0.77)	0.00
CRM114	1.90 (1.61, 2.24)	0.45 (0.35, 0.57)	0.23 (0.16, 0.33)	0.00
DSPAM	7.02 (6.33, 7.77)	0.23 (0.18, 0.30)	0.03 (0.02, 0.04)	0.00
SA-Bayes	2.51 (2.21, 2.85)	1.74 (1.52, 2.00)	0.69 (0.55, 0.87)	0.00
SA-Human	1.67 (1.40, 1.98)	0.64 (0.52, 0.79)	0.38 (0.27, 0.53)	0.00
SA-Nolearn	7.73 (7.25, 8.25)	11.37 (10.76, 12.02)	1.53 (1.37, 1.72)	0.00
SA-Standard	16.07 (15.22, 16.96)	2.67 (2.43, 2.92)	0.14 (0.13, 0.16)	0.00
SA-Supervised	1.68 (1.44, 1.96)	1.36 (1.16, 1.59)	0.81 (0.61, 1.07)	0.13
SA-Unsupervised	18.03 (17.13, 18.98)	2.67 (2.44, 2.92)	0.12 (0.11, 0.14)	0.00
SpamBayes	5.91 (5.46, 6.39)	5.82 (5.38, 6.29)	0.99 (0.85, 1.14)	0.82
SpamProbe	1.29 (1.08, 1.56)	0.81 (0.67, 1.00)	0.63 (0.45, 0.88)	0.01

Table VIII. Spam Learning Performance

messages were classified into seven genres:

(1) *Advertising.* Messages from companies or organizations having a relationship with the recipient.
(2) *Cold Call.* Messages from individuals with whom X had no prior correspondence or relationship.
(3) *Delivery.* Messages from an email server pertaining to the delivery of an email message.
(4) *List.* Mailing list messages, broadly defined. This genre includes automated mailing lists, service messages from mailing lists, and ad hoc messages consisting of general information copied to a large number of recipients.
(5) *News.* News clipping and digest services to which X is subscribed.
(6) *Personal.* Mail specifically addressed to X by an individual; the equivalent of *first class mail.*
(7) *Transaction.* Responses to electronic internet transactions, such as receipts, travel itineraries, shipping information, passwords, acknowledgements, or status information.

Spam messages were classified into five genres:

(1) *Advertising.* Messages sent indiscriminately to X aimed at acquiring some or all of X's wealth.

(2) *Backscatter.* Delivery messages from a third-party server, rejecting a message not sent by X, but forged to appear to have been sent by X. These messages are deemed to be spam (as opposed to Delivery ham messages) because they are a direct consequence of spam.

(3) *Demographic.* Advertising messages for goods and services of marginal value sent to a specific demographic group to which X belongs.

(4) *Targeted.* Messages addressed to X for no reason other than X's membership in a broad identifiable group (profession, geographic location, appearance on a subject-related web-page, etc.).

(5) *Virus.* Messages that contain malware.

Table IX shows the number of misclassified ham messages, by genre, for each filter. Also shown is an estimate of the proportion of all ham represented by each genre.

Filter	Advertising	Cold Call	Delivery	List	News	Personal	Transaction	Total
SA-Standard	4	2	0	0	0	0	0	6
SA-Super	1	0	0	1	1	0	3	6
Bogofilter	1	0	0	2	1	0	3	7
SA-Human	0	0	0	3	4	0	1	8
SA-Unsuper	5	0	0	1	0	1	3	10
SA-Bayes	1	0	0	4	1	1	8	15
SpamBayes	1	0	2	5	1	3	3	15
SA-Nolearn	1	0	4	0	3	9	0	17
SpamProbe	3	2	4	5	1	8	8	31
DSPAM	15	5	9	28	6	35	18	116
CRM114	7	15	13	78	10	135	37	295
Incoming Ham	0%	1%	17%	13%	14%	51%	4%	9038

Table IX. Ham Misclassification by Genre

Four of the runs have no *personal* misclassifications, a much lower fraction than would be suggested by the fact that this genre comprises 51% of all ham. At the other end of the spectrum, CRM114 misclassifies 135 *personal* ham messages, or about 3% of all such messages. DSPAM also misclassifies a high number of *personal* messages: 35, or about 0.75% of the total.

In general, *advertising, cold call*, and *delivery* messages each represent a small proportion of overall ham and a disproportionately large number of misclassifications. *Personal* messages represent disproportionately few misclassifications, while *transaction, list,* and *news* fall in between.

Table X shows the estimated fraction of misclassified spam messages, by genre, for each filter, as well as the fraction of all spam represented by each genre. The vast majority of spam messages are *advertising,* with *backscatter* representing a mere 1%. Yet nearly as many *backscatter* messages are misclassified. In particular, we note that SA-Human and SA-Super misclassify a fraction of backscatter messages approaching or exceeding 50%. Three-fifths of all of SA-Human's misclassifications are attributable to misclassified *backscatter.* The reason for this is that X was

Filter	Advertising	Backscatter	Demographic	Targeted	Virus	Total
CRM114	72%	8%	12%	4%	4%	397
SA-Human	14%	66%	10%	7%	4%	413
Spamprobe	48%	17%	17%	7%	12%	421
SA-Super	28%	36%	22%	5%	9%	605
DSPAM	58%	8%	17%	3%	14%	791
SA-Bayes	45%	19%	17%	8%	11%	840
SpamBayes	50%	16%	25%	7%	2%	2348
Bogofilter	68%	14%	10%	2%	6%	2656
SA-Standard	17%	29%	5%	0%	49%	2999
SA-Unsupervised	9%	31%	7%	1%	52%	3246
SA-Nolearn	51%	24%	5%	0%	20%	3802
Incoming Spam	92%	1%	0%	0%	8%	40048

Table X. Spam Misclassification by Genre

overwhelmed by the burst of *backscatter* occurring at 17,000, and skipped over many of these messages without recording a judgement[6].

8. OTHER EVALUATION MEASURES

Although widely reported, *accuracy* has little value in evaluating and comparing spam filters[Provost et al. [Provost et al. 1998]]. The consequences of ham and spam misclassification are materially different, while measurements of accuracy conflate them. The computation of accuracy depends directly on the ratio of ham to spam messages in the incoming email, and also on the threshold parameter used by the filter to transform scores into judgements. For a given filter, the problem of optimizing accuracy reduces to the decision-theoretic problem of picking the best threshold [Lewis [Lewis 1995]] for the anticipated ham-to-spam ratio ($hs = \frac{a+c}{b+d}$; a, b, c, d from table I). Tables III and IV include *overall misclassification fraction (1-accuracy)* which reflect influence of the systems' default threshold parameters. Every system in this study, had its threshold been set to optimize accuracy[7], would have yielded an unacceptably high level of ham misclassification (see table XI).

Androutsopolous et al. [[Androutsopoulos et al. 2004]] argue that the relative importance of ham over spam misclassification errors be quantified by a parameter λ used as input to the filter in *cost-sensitive classification* and to the evaluation measure in *cost-sensitive evaluation.* Weighted accuracy is defined as $w=\frac{\lambda a+d}{\lambda a+b+\lambda c+d}$. Further, they suggest *total cost ratio* $TCR = \frac{b+d}{b+\lambda c}$, as a measure to distinguish the weighted accuracy of a filter from that of a simplistic approach that classifies every message as ham. In contrast, our test methods and evaluation measures are agnostic as to the relative importance of ham over spam, and leave the cost-sensitive

[6]X subsequently deployed an ad-hoc filter to identify backscatter messages and to record a judgement automatically.

[7]The results presented here are the result of a hypothetical run for which the optimal threshold was known in advance. Lewis discusses automatic methods of adjusting the threshold so as to optimize error rate (i.e. $1 - accuracy$) and other measures.

Filter	Ham Misc.	Spam Misc.	Overall Misc.
SpamProbe	0.94	0.44	0.54
SA-Super	1.62	0.49	0.69
SA-Bayes	1.97	0.40	0.69
Bogofilter	1.25	0.59	0.71
SpamBayes	1.60	0.61	0.79
DSPAM	1.90	1.03	1.19
CRM114	3.95	0.75	1.34
SA-Nolearn	5.53	2.77	3.28
SA-Standard	2.98	3.88	3.71
SA-Unsuper	10.64	1.67	3.32

Table XI. Effect of Optimizing Accuracy

interpretation of these results to the reader.

Hidalgo [[Hidalgo 2002]] discusses the use of cost-sensitive evaluation to mitigate these difficulties:

> The main problem in the literature on [spam] cost-sensitive categorization is that the [ham-spam cost ratios] used do not correspond to real world conditions, unknown and highly variable. No evidence supports that classifying a legitimate message as [spam] is 9 nor 999 times worse than the opposite mistake.

This criticism – dependence on highly variable external factors, arbitrary filter parameters, and arbitrary evaluation weights – applies to a large class of combined evaluation measures [cf. Sebastiani [Sebastiani 2002]]. To this criticism we add a note of caution with respect to the statistical power of filter evaluations. Ham misclassification rates for good filters are exceptionally low, amounting to only a handful of messages in our sample of nearly 50,000. These rates are even lower when stratified by genre, often yielding 0 occurrences (e.g. four of the runs misclassified *no* personal email messages). The statistical uncertainty due to these small numbers will dominate any weighted score, potentially masking significant differences in spam misclassification rates for filters with comparable ham misclassification rates.

Hidalgo suggests the use of ROC curves, originally from signal detection theory and used extensively in medical testing, as better capturing the important aspects of spam filter performance. In the event that the ROC curve for one filter is uniformly above that of another, we may conclude that there is a parameter setting such that its performance exceeds the other for any combination of external factors and evaluation weights. The area under the ROC curve serves to quantify this difference and, perhaps surprisingly, represents a meaningful quantity: the probability that a random spam message will receive a higher score than a random ham message. In the event that the ROC curves intersect, one may consider the area under only a subset, the *normal operating region.* For a spam filter, this operating region would likely be the fragment of the curve above the range of acceptable ham misclassification fraction values.

Tuttle et al. [[Tuttle et al. 2004]] present spam filter effectiveness using a tabular representation of an ROC curve: hm vs. $(1 - sm)$. Further, they choose 1% hm as a proxy for the normal operating region and report sm at this value. More broadly, ROC-based evaluation for machine learning and information retrieval is of current

interest. We found that ROC analysis provided us with valuable insight to our results, complementing but not obviating distinct ham and spam misclassification analyses. With one inversion (SA-Standard vs. SA-Unsupervised) AUC values agreed with our subjective ranking of the systems. The ROC curves for these two runs intersect; SA-Standard demonstrates superior performance within the normal operating region (small hm) while SA-Unsupervised overtakes it for large hm.

Like the measures described above, *recall, precision,* and *precision-recall* curves evaluate the tension between ham and spam classification performance. Precision and recall originate with information retrieval, in which the objective is to discover relevant documents from a collection. The measures are asymmetric, predicated on the general assumption that there are many fewer relevant than non-relevant documents in the collection. *Recall* is the fraction of all relevant documents retrieved by the system; *precision* is the fraction of retrieved documents that are relevant. Within the context of spam classification, it is necessary to consider either the ham or the spam messages as relevant, and the others as not relevant. This labelling is arbitrary, but must be identified. Ham precision ($hp = \frac{a}{a+b}$) and ham recall ($hr = \frac{a}{a+c}$), in which ham messages are deemed to be relevant, have perhaps the more intuitive meaning within the context of spam filtering. The complementary measures are spam precision ($sp = \frac{d}{c+d}$) and spam recall ($sr = \frac{d}{b+d}$).

Ham recall is the same thing as ham accuracy ($1 - hm$). Spam recall is the same thing as spam accuracy ($1 - sm$). But these two measures are not used as a pair in information retrieval evaluation, which assumes a consistent labelling of relevant and non-relevant documents. Instead, ham precision and ham recall (or spam precision and spam recall) are used together[8]. Ham precision depends on sm but depends also on hr and hs: $hp = \frac{r}{1+r}$ where $r = hs \cdot \frac{hr}{sm}$. r, the ratio of ham to spam delivered to the mail file, is proportional to the incoming ham-spam ratio. Ham precision simply recasts r as a fraction as opposed to a ratio. Thus we conclude that precision and recall, taken as a pair, exhibit the same essential shortcoming as accuracy. *Average precision*, the analog of AUC, is similarly influenced by hs.

The medical diagnostic testing literature [cf. Rothman & Greenland [Rothman and Greenland 1998]] casts the problem as one of testing a population of patients for a particular disease. The test offers a diagnosis of *diseased* or *disease-free.* To apply diagnostic testing metaphors to spam, we (arbitrarily but with some support from connotation) label spam to be diseased and ham to be disease-free. The variables a, b, c, d from table I are known as *true negatives, false negatives, false positives,* and *true positives* respectively. Ham accuracy is *specificity,* while spam accuracy is *sensitivity* [9]. The literature also discusses *negative predictive value* and *positive predictive value.* Negative predictive value is the probability that a random patient, on receiving a negative diagnosis, is really disease-free. Positive predictive value is the probability that a random patient, on receiving a positive diagnosis, is really diseased. Predictive values use Bayesian inference to combine two distinct estimates: specificity (or sensitivity), which is a property of the diagnostic test, and

[8]The information retrieval literature defines *fallout*, which in this context would be the same as sm and therefore equivalent to spam recall. Recent evaluations often report precision and recall; rarely fallout.

[9]To our knowledge, no analog of overall accuracy exists in medical diagnostic testing.

prevalence, which is a property of the population being tested. Negative predictive value is exactly ham precision as described above, while positive predictive value is spam precision.

Precision, like predictive value, is very useful in predicting the in situ performance of a filter. We believe it should, like predictive value, be computed post-hoc by combining separate measurements of filter performance and incoming ham-spam ratio, rather than used as a fundamental measure of filter performance.

DET curves [Martin et al. [Martin et al. 1997]] have been used to measure the performance of filters within the context of document understanding. A DET curve is exactly an ROC curve, plotted on a normal deviate scale. The normal deviate scale resembles the logit scale we used; the former will tend to yield a linear curve when one assumes that the scores of ham and spam messages are normally distributed; the latter assumes binomial distributions.

9. OTHER STUDIES

Direct comparison of results demands that common data (or at least data sampled from a similar population) be used to test different filters, or that common filters be tested on different data, and that common measures be reported. Valid measurements must be based on realistic assumptions and statistically well founded. With these criteria in mind, we explore the commonality among studies, and, where possible from the published data, recast their results in terms of common measures with confidence intervals.

Sahami et al. [[Sahami et al. 1998]] conducted an early study that indicated the utility of Bayesian classifiers for spam filtering. One experiment used a corpus of 1789 actual email messages (11.8% ham; 88.2% spam), split chronologically into 1538 training messages and 251 test messages. Both ham and spam precision/recall curves were calculated. The best-performing system achieved ham recall of 100% and spam recall of 98.3%. From these values and the test sample size we may compute $hm = 0\%\,(0\% - 9.5\%)$ and $sm = 1.7\%\,(0.4\% - 4.6\%)$. A second experiment classified the spam component of a similar corpus into two genres: *pornographic* and *non-pornographic*. The genres were used in an evaluation of ternary classification, but not for a stratified evaluation of binary classification. A third experiment most closely resembles those which we conducted: an individual's email messages were captured over one year, classified manually, and used as training data. The filter was applied to further week's email received by the same individual. The resulting classification table, shown in table XII, demonstrates $hm = 1.7\%\,(0.3\% - 4.9\%)$, $sm = 20\%\,(9.6\% - 34.6\%)$. Sahami et al. further examine the three misclassified

Contingency table			% Ham Misc.	% Spam Misc.	% Misc.
	ham	spam			
ham	174	9	1.7% (0.3%-4.9%)	20% (9.6%-34.6%)	5.41 (2.82-9.25)
spam	3	36			

Table XII. Sahami et al.

ham messages, observing two to be newsletter messages and one to be a personal

message that includes a spam message as an attachment. The test corpus is unavailable for comparative evaluation.

Several studies [e.g. Androutsopoulos et al. [Androutsopoulos et al. 2000; Androutsopoulos et al. 2000]; Drucker et al. [Drucker et al. 1999]; Pampathi et al. [Pampapathi et al. 2005]; Sakkis et al. [Sakkis et al. 2001a]; Zhang et al. [Zhang et al. 2004]] investigate the utility of various machine-learning techniques on spam filtering using a small corpus and ten-fold cross validation [cf. Kohavi [Kohavi 1995]]. The design of their experiments is typical of machine-learning research. A classifier is trained on a fixed set of labeled data, the training set, and then asked to classify another set of similar data, the test set. Ten-fold cross validation [cf. Kohavi [Kohavi 1995]] is used for evaluation. A corpus of size n is divided randomly into 10 subsets of size $\frac{n}{10}$. Each subset is used as the test set with the remaining nine subsets combined for training. The union of the ten sets of results has the same statistical precision as one set of n Bernoulli trials. The validity of cross validation depends on the assumption that the order of messages is unimportant; that the ratio of ham to spam and the characteristics of the ham and spam messages are invariant with time. Consider, for example, a burst of five nearly identical spam messages that arrive in a short time interval. In a real email sequence, the filter might easily be flummoxed by the first of these messages, but learn its characteristics and correctly classify the rest. With ten-fold cross-evaluation, it is nearly certain that each training set will contain several of these messages, so the filter's ability to classify the first-of-a-kind is essentially untested. In general, cross-validation tests the performance of a filter only after a fixed number of training examples; spam filter users seldom have several hundred or thousand labeled examples available for training prior to deployment.

Ling Spam [Androutsopoulos et al. [Androutsopoulos et al. 2000]] is an abstraction of 2412 ham messages from a mailing list and 481 spam messages from an individual recipient. We say abstraction because the messages are stripped of

λ	Contingency table		% Ham Misc.	% Spam Misc.	% Overall Misc.
1	2410	83	0.08 (0.01-0.30)	17.3 (14.0-20.9)	2.94 (2.35-3.62)
	2	398			
9	2410	104	0.08 (0.01-0.30)	21.6 (18.0-25.6)	3.67 (3.01-4.41)
	2	377			
999	2412	168	0 (0-0.12)	34.9 (30.7-39.4)	5.81 (4.98-6.72)
	0	313			

Table XIII. Androutsopoulos et al.

headers and line breaks, converted to lower case, tokenized and stemmed. The filters tested on the Ling Spam corpus were purpose-built to use it to evaluate specific machine-learning techniques. Although the results are reported in terms of spam recall, spam precision and weighted accuracy, it is possible to reconstruct the contingency table from these results. Table XIII, for example, recasts the results of Androutsopoulos et al. [[Androutsopoulos et al. 2000]] in terms of hm and sm. Ling Spam is freely available and has been used in many studies [Androutsopoulos et al. [Androutsopoulos et al. 2000], [Androutsopoulos et al. 2000]; Sakkis et al. [Sakkis et al. 2001b]; Zhang et al. [Zhang et al. 2004]]. We found that real spam

Filter	% Ham Misc.	% Spam Misc.	% Misc.
SpamAssassin	0	95.4	15.9
Bogofilter	0	59.5	9.9
SpamProbe	0	31.3	5.2
CRM114	54.9	11.2	18.5

Table XIV. Real filter results on Ling Spam corpus

filters were in general unable to classify the Ling Spam messages (see table XIV); we are unaware of how to modify either the corpus or the filters so as to use them together in a valid experiment.

Androutsopoulos et al. [[Androutsopoulos et al. 2004]] define four public corpora – PU1, PU2, PU3 and PUA – with a total of 5957 messages (3508 ham and 2449 spam); each corpus abstracts and also obfuscates email from one recipient, so as to preserve privacy. In addition, repeated spam messages and spam messages from regular correspondents – about half the spam and eighty percent of the ham – are discarded in forming the corpus. As for Ling Spam, experiments using these corpora depend on purpose-built filters. One such filter – Filtron – was trained on

Corpus	Contingency table		% Ham Misc.	% Spam Misc.	% Overall Misc.
PU3	2249	90	2.8 (2.1-3.5)	4.9 (4.0-4.9)	3.7 (3.2-4.3)
	64	1736			
Real Email	5057	173	1.0 (0.8-1.3)	10.7 (9.2-12.3)	3.8 (3.3-4.3)
	52	1450			

Table XV. Filtron Results

PU3 and tested on real email received by an individual over seven months. During this interval, 5109 ham and 1623 spam messages were received and classified. Table XV summarizes the results. Neither Filtron nor the real email corpus is available for comparative study.

The public *SpamAssassin* corpus [[SpamAssassin 2004]] consists of 6034 messages – 4149 ham and 1885 spam – gathered from various sources at various times. Although it is not a chronological sequence of messages delivered to a single recipient, the messages contain original headers with minor elision for the sake of privacy. Holden [2004] used the SpamAssassin corpus and ten-fold cross-validation to test fourteen open-source filters, including versions of the six tested here. Holden's results are summarized in table XVI. Holden further tested the filter on one-month's personal email, again using cross-validation; results are shown in table XVII. Holden provides a qualitative description of the misclassified ham messages, observing a preponderance of messages like welcome advertising, news clippings, mailing lists, etc. Many other studies have used the SpamAssassin corpus [Meyer & Whateley [Meyer and Whateley 2004]; Yerazunis [Yerazunis 2004a]; Zhang et al. [Zhang et al. 2004]].

Zhang et al. [[Zhang et al. 2004]] evaluate several learning algorithms on four corpora – Ling Spam, PU1, SpamAssassin, and ZH1. ZH1 is a private corpus of 1633 Chinese messages with headers; 428 ham and 1205 spam. Only the TCR statistic is reported ($\lambda = 9$ and $\lambda = 999$) ; from this statistic it is impossible, in general, to recover sm and hm. In the specific case of $\lambda = 999$ we may deduce

Filter	Contingency table		% Ham Misc.	% Spam Misc.	% Overall Misc.
Annoyance	4147	209	0.07 (0.01-0.21)	11.0 (9.6-12.5)	3.5 (3.1-4.0)
	3	1688			
Antispam	4016	35	3.2 (2.7-3.8)	1.8 (1.3-2.6)	2.8 (2.4-3.2)
	133	1863			
Bayesspam	4033	77	2.9 (2.3-3.4)	4.1 (3.2-5.0)	3.2 (2.8-3.7)
	117	1863			
bmf	4137	78	0.3 (0.2-0.5)	4.1 (3.3-5.1)	1.5 (1.2-1.8)
	18	1819			
Bogofilter	4145	184	0.12 (0.04-0.28)	9.7 (8.4-11.1)	3.1 (2.7-3.6)
	5	1713			
CRM114	4100	58	1.2 (0.9-1.6)	3.1 (2.3-3.9)	1.8 (1.5-2.2)
	50	1839			
dbacl	309	22	92.5 (91.7-93.3)	1.2 (.73-1.75)	63.9 (62.7-65.1)
	3841	1875			
DSPAM	4137	76	0.3 (0.17-0.5)	4.0 (3.2-5.0)	1.5 (1.2-1.8)
	13	1821			
lfile	4087	129	1.5 (1.2-1.9)	6.8 (5.7-8.0)	3.2 (2.7-3.6)
	63	1768			
qsf	4134	160	0.39 (0.22-0.63)	8.4 (7.2-9.8)	2.9 (2.5-3.4)
	16	1737			
SpamAssassin	4144	74	0.14 (0.05-0.31)	3.9 (3.1-4.9)	1.3 (1.1-1.6)
	6	1823			
SpamBayes	4143	83	0.17 (0.07-0.34)	4.4 (3.5-5.4)	1.5 (1.2-1.8)
	7	1814			
SpamOracle	4150	309	0 (0-0.07)	16.3 (14.6-18.0)	5.1 (4.6-5.7)
	0	1588			
SpamProbe	4144	65	0.14 (0.05-0.31)	3.4 (2.7-4.3)	1.2 (0.9-1.5)
	6	1832			

Table XVI. Holden on SpamAssassin Corpus

that the best-ranked classifiers had no ham misclassifications (i.e. $c = 0$) and we may use this deduction combined with corpus statistics to compute hm and sm. TCR for these filters, on the SpamAssassin corpus, was approximately 12. Suppose $c > 0$. Because b and c are whole numbers, we have $b \geq 0$ and $c \geq 1$. We have $TCR = \frac{b+d}{\lambda c+b} \leq \frac{1897}{999+0} = 1.9$, which contradicts the reported value. Therefore $c = 0$, $b \approx 158$, $hm = 0\,(0\% - 0.07\%)$, $sm \approx 8.3\%\,(7.1\% - 9.7\%)$. The confidence interval for hm should be interpreted with caution because Zhang et al. adjusted the threshold parameter θ_{999} so as to optimize TCR, effectively fixing $c = 0$ for the corpus data. For $\lambda = 9$ we are unable to deduce the values of b and c, so in figure XVIII we present as TCR these results, and also Holden's, on the SpamAssassin corpus. We note that this comparison is not entirely valid, as the filters for which the threshold and other parameters were not adjusted to optimize the result (i.e. the filters tested by Holden and, we understand, the Bayes method of Zhang et al.) are at considerable disadvantage.

Tuttle et al. [[Tuttle et al. 2004]] evaluate three common machine-learning algorithms – naive Bayesian classifiers, support vector machines, and boosted decision trees – within the context of an enterprise mail system. They deployed a novel architecture to capture email messages and judgements from several users, keeping this information private and under the control of the users to whom the messages

Filter	Contingency table		% Ham Misc.	% Spam Misc.	% Overall Misc.
Annoyance	144	34	1.4 (0.17-4.9)	0.96 (0.67-1.3)	0.97 (0.68-1.3)
	2	3523			
Antispam	84	2	42.5 (34.4-50.9)	0.06 (0.007-0.2)	1.7 (1.3-2.2)
	62	3555			
Bayesspam	117	189	19.9 (13.7-27.2)	5.3 (4.5-6.1)	5.9 (5.2-6.7)
	29	3668			
bmf	138	25	5.5 (3.4-10.5)	0.7 (0.5-1.0)	0.9 (0.6-1.2)
	8	3532			
Bogofilter	146	169	0 (0-2.0)	4.8 (4.1-5.5)	4.6 (3.9-5.3)
	0	3338			
CRM114	119	19	18.5 (12.6-25.8)	0.5 (0.3-0.8)	1.2 (0.9-1.7)
	27	3538			
dbacl	137	2224	6.2 (2.9-11.4)	62.5 (60.9-64.1)	60.3 (58.1-61.9)
	9	1333			
DSPAM	146	1723	0 (0-2.0)	48.4 (46.8-50.1)	46.5 (44.9-48.2)
	0	1834			
lfile	144	90	1.4 (0.2-4.8)	2.5 (2.0-3.1)	2.5 (2.0-3.0)
	2	3467			
qsf	146	149	0 (0-2.0)	4.1 (3.6-4.9)	4.0 (3.4-4.7)
	0	3408			
SpamAssassin	146	1799	0 (0-2.0)	50.6 (48.9-52.2)	48.6 (47.0-50.2)
	0	1758			
SpamBayes	145	189	0.7 (0.2-3.8)	5.3 (4.6-6.1)	5.1 (4.4-5.9)
	1	3368			
SpamOracle	144	406	1.4 (0.2-4.8)	11.4 (10.4-12.5)	11.0 (10.0-12.1)
	2	3151			
SpamProbe	136	9	6.8 (3.3-12.2)	0.25 (0.12-0.48)	0.51 (0.31-0.80)
	10	3548			

Table XVII. Holden on Personal Email

Holden		Zhang et al.	
Filter	TCR ($\lambda = 9$)	Method	Best TCR (approx, $\lambda = 9$)
Annoyance	8.0	Naive Bayes	1.9
Antispam	1.5	Max. Entropy	15.2
Bayesspam	1.7	Memory Based	7.0
bmf	7.9	SVMlight	12.1
Bogofilter	8.2	Boost Stumps	10.4
CRM114	3.7		
dbacl	0.1		
DSPAM	9.8		
lfile	2.7		
qsf	6.2		
SpamAssassin	14.7		
SpamBayes	12.9		
SpamOracle	6.1		
SpamProbe	15.8		

Table XVIII. Holden and Zhang et al. on SpamAssassin Corpus

belonged. Test runs were "pushed" to the users' corpora, and only statistics were reported back to the central system. Seven users participated in the study, and corpora consisting of up to 800 messages per user were subject to ten-fold cross-validation. Results for each of the seven corpora were computed and the mean of these results was reported. The primary experiment used individual corpora with 400 messages each, approximately 62% spam, and reported piece-wise ROC curves (see table XIX) for $hm \in \{0.0\%, 0.5\%, 1.0\%, 2.0\%, 5.0\%\}$. Other experiments fixed $hm = 1.0\%$ as a proxy for the operating range. The published averages yield insuf-

%Ham Misc.	% Spam Misc.		
	Naive Bayes	SVM	AdaBoost
0.0	5.9	6.2	10.5
0.5	4.1	4.4	8.1
1.0	2.8	3.5	5.6
2.0	2.0	2.2	2.6
5.0	1.1	0.5	1.3

Table XIX. Tuttle

ficient information to compute confidence intervals, but we note the overall sample size of 2800 suggests that they would be comparable in magnitude to those for Sahami et al. and Androutsopoulos et al. Tuttle et al. perform a 2-factor analysis of variance and conclude that there is a significant difference in results among the seven corpora, but not among the three filters.

Kolcz and Alspector [[Kolcz and Alspector 2001]] model the cost of misclassifying various genres of messages. This approach stands in contrast to the cost-sensitive methods discussed above, which assume the cost to be dependent only on whether the message is ham or spam. It also stands in contrast to ours, in which the test method and quantitative evaluation measures assume no particular cost model, and messages genres are treated qualitatively. Kolcz and Alspector assembled a corpus of 11408 messages (6043 ham; 5365 spam) which were labeled according to category; each category was assigned an estimated cost of misclassification. The corpus was split into training and test sets in a 3:1 ratio. Results are reported in terms of TCR and ROC analysis. Although Kolcz and Alspector report their intent to publish the corpus, to our knowledge it is not available.

The methods and tools developed here have been used in at TREC 2005 [Cormack & Lynam [Cormack and Lynam 2005b]] and TREC 2006 [[Cormack 2006]] to evaluate spam filters developed by some twenty independent groups on eight independently-sourced corpora. One of the corpora was the "Mr. X" corpus used in this study. Another was the "Mr. X II" corpus built from email delivered to X from October 2005 through May 2006. The other corpora – three private and three public – were developed using the same methodology. The results reported at TREC for the "Mr X" corpus may be compared directly to those reported here; those based on public corpora and filters may be reproduced independently.

10. CONCLUSIONS

Supervised spam filters are effective tools for attenuating spam. The best-performing filters reduced the volume of incoming spam from about 150 messages per day to

about 2 messages per day. The corresponding risk of mail loss, while minimal, is difficult to quantify. The best-performing filters misclassified a handful of spam messages early in the test suite; none within the second half (25,000 messages). A larger study will be necessary to distinguish the asymptotic probability of ham misclassification from zero.

Most misclassified ham messages are advertising, news digests, mailing list messages, or the results of electronic transactions. From this observation, and the fact that such messages represent a small fraction of incoming mail, we may conclude that the filters find them more difficult to classify. On the other hand, the small number of misclassifications suggests that the filter rapidly learns the characteristics of each advertiser, news service, mailing list, or on-line service from which the recipient wishes to receive messages. We might also conjecture that these misclassifications are more likely to occur soon after subscribing to the particular service (or soon after starting to use the filter), a time at which the user would be more likely to notice, should the message go astray, and retrieve it from the spam file. In contrast, the best filters misclassified no personal messages, and no delivery error messages, which comprise the largest and most critical fraction of ham.

A supervised filter contributes significantly to the effectiveness of SpamAssassin's static component, as measured by both ham and spam misclassification probabilities. Two unsupervised configurations also improved the static component, but by a smaller margin. The supervised filter alone performed better than than the static rules alone, but not as well as the combination of the two.

The choice of threshold parameters dominates the observed differences in performance among the four filters (Bogofilter, SA-Bayes, SpamProbe, SpamBayes) implementing methods derived from Graham's and Robinson's proposals. Each shows a different tradeoff between ham accuracy and spam accuracy. ROC analysis shows that the differences not accountable to threshold setting, if any, are small and observable only when the ham misclassification probability is low (i.e. $hm < 0.1\%$). The other filters (DSPAM, CRM114) show lower performance over all threshold settings.

Ham and spam misclassification proportions should be reported separately. Accuracy, weighted accuracy, and precision should be avoided as primary evaluation measures as they are excessively influenced by threshold parameter setting and the ham-spam ratio of incoming mail. ROC curves provide valuable insight into the tradeoff between ham and spam accuracy. Area under the ROC curve provides a meaningful overall effectiveness measure, but does not replace separate ham and spam misclassification estimates. Each case of ham misclassification should be examined to ascertain its cause and potential impact.

Caution should be exercised in treating ham misclassification as a simple proportion. Extremely large samples would be needed to estimate it with any degree of statistical confidence, and even so, it is not clear what effect differences in proportion would have on the overall probability of catastrophic loss. The use of a filter may mitigate rather than exacerbate this risk, owing to the reduction in classification effort required of the user. We advance the proposition that, at the misclassification rates demonstrated here, the end-to-end risk of loss is dominated by human factors and exceptional events, and is comparable to that of other communication

media.

It has been widely suggested [cf. Graham-Cumming [Graham-Cumming 2006]] that spam senders may be able to adapt so as to defeat statistical spam filters. We are able to observe this adaptation by its effect on the rule-based filter results over time. But we see no evidence that the adaptation compromises the efficacy of on-line statistical filters, either during the eight-month interval of this study, or the interval between this study and a subsequent study which we conducted using email delivered to X more than two years later. [Cormack [Cormack 2006]]

The potential contribution of more sophisticated machine learning techniques to real spam filtering is as-yet unresolved. In artificial environments they appear to be promising, but this promise is yet to be demonstrated in comparison to existing filters that use perhaps more primitive techniques [Cormack and Bratko [Cormack and Bratko 2006]]. The potential contribution of real-time network resources and collaborative methods to spam filtering also has yet to be established. Spam filtering is an adversarial task – the degree to which spam is able to adapt to counter advances in filtering has yet to be studied. While constructing controlled experiments to measure these factors presents a significant logistical challenge, our model and evaluation methods are amenable.

REFERENCES

Agresti, A. 1996. *An Introduction to Categorical Data Analysis*. Wiley, New York.

Androutsopoulos, I., Koutsias, J., Chandrinos, K., Paliouras, G., and Spyropoulos, C. 2000. An evaluation of naive bayesian anti-spam filtering. In *Proceedings of the Workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning (ECML 2000)*, G. Potamias, V. Moustakis, and M. van Someren, Eds. Barcelona, Spain, 9–17. Available: `http://arXiv.org/abs/cs.CL/0006013`.

Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C., and Stamatopoulos, P. 2000. Learning to filter spam e-mail: A comparison of a naive bayesian and a memory-based approach. In *Proceedings of the Workshop on Machine Learning and Textual Information Access, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2000)*, H. Zaragoza, P. Gallinari, , and M. Rajman, Eds. Lyon, France, 1–13. Available: `http://arXiv.org/abs/cs/0009009`.

Androutsopoulos, I., Paliouras, G., and Michelakis, E. 2004. Learning to filter unsolicited commercial e-mail. Tech. Rep. 2004/2, NCSR Demokritos.

Burton, B. 2002. Spamprobe - a fast bayesian spam filter. http://spamprobe.sourceforge.net.

Cormack, G. V. 2006. TREC 2006 Spam Track Overview. In *Fifteenth Text REtrieval Conference (TREC-2005)*. NIST, Gaithersburg, MD.

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

Cormack, G. V. and Lynam, T. R. 2005a. Spam corpus creation for TREC. In *Proc. CEAS 2005 – The Second Conference on Email and Anti-Spam*.

Cormack, G. V. and Lynam, T. R. 2005b. TREC 2005 Spam Track Overview. In *Fourteenth Text REtrieval Conference (TREC-2005)*. NIST, Gaithersburg, MD.

Drucker, H., Vapnik, V., and Wu, D. 1999. Support vector machines for spam categorization. *IEEE Transactions on Neural Networks 10,* 5, 1048–1054.

Fawcett, T. 2003a. 'in vivo' spam filtering: A challenge problem for data mining. *KDD Explorations 5,* 2 (December). Available: `http://www.hpl.hp.com/personal/Tom_Fawcett/papers/spam-KDDexp.pdf`.

FAWCETT, T. 2003b. ROC graphs: Notes and practical considerations for data mining researchers. Tech. rep., HP Laboratories.

FISHER, R. A. 1925. Theory of statistical estimation. *Proceedings of the Cambridge Philosophical Society 22*, 700–725.

FLACH, P. A. 2004. The many faces of roc analysis in machine learning. http://www.cs.bris.ac.uk/~flach/ICML04tutorial/.

GRAHAM, P. 2002. A plan for spam. http://www.paulgraham.com/spam.html.

GRAHAM, P. 2004. Better bayesian filtering. http://www.paulgraham.com/better.html.

GRAHAM-CUMMING, J. 2006. Does Bayesian poising exist? *Virus Bulletin*.

HIDALGO, J. M. G. 2002. Evaluating cost-sensitive unsolicited bulk email categorization. In *Proceedings of SAC-02, 17th ACM Symposium on Applied Computing*. Madrid, ES, 615–620.

HOLM, S. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics 6*, 65–70.

KOHAVI, R. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*. 1137–1145.

KOLCZ, A. AND ALSPECTOR, J. 2001. SVM-based filtering of e-mail spam with content-specific misclassification costs. In *Proceedings of the TextDM'01 Workshop on Text Mining - held at the 2001 IEEE International Conference on Data Mining*. Available: `http://www-ai.ijs.si/DunjaMladenic/TextDM01/papers/Kolcz_TM.pdf`.

LENHARD, J. 2006. Models and statistical inference: The controversy between Fisher and Neyman-Pearson. *British Journal for the Philosophy of Science*.

LEWIS, D. D. 1995. Evaluating and optimizing autonomous text classification systems. In *SIGIR'95, Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Seattle, Washington, USA, July 9-13, 1995 (Special Issue of the SIGIR Forum)*, E. A. Fox, P. Ingwersen, and R. Fidel, Eds. ACM Press, 246–254.

LYNAM, T. AND CORMACK, G. 2005. TREC Spam Filter Evaluation Tool Kit. http://plg.uwaterloo.ca/~trlynam/spamjig.

MARTIN, A., DODDINGTON, G., KAMM, T., ORDOWSKI, M., AND PRZYBOCKI, M. 1997. The DET curve in assessment of detection task performance. In *Proc. Eurospeech '97*. Rhodes, Greece, 1895–1898.

MEYER, T. AND WHATELEY, B. 2004. Spambayes: Effective open-source, bayesian based, email classification system. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*. Available: `http://www.ceas.cc/papers-2004/136.pdf`.

PAMPAPATHI, R. M., MIRKIN, B., AND LEVENE, M. 2005. A suffix tree approach to email filtering. In *UK Workshop on Computational Intelligence*. London, UK.

PARK, S. H., GOO, J. M., AND JO, C.-H. 2004. Receiver Operating Characteristic (ROC) curve: Practical review for radiologists. *Korean Journal of Radiology 5,* 1, 11–18.

PETERS, T. 2004. Spambayes: Bayesian anti-spam classifier in python. http://spambayes.sourceforge.net/.

PROVOST, F., FAWCETT, T., AND KOHAVI, R. 1998. The case against accuracy estimation for comparing induction algorithms. In *Proc. 15th International Conf. on Machine Learning*. Morgan Kaufmann, San Francisco, CA, 445–453.

RAYMOND, E. S., RELSON, D., ANDREE, M., AND LOUIS, G. 2004. Bogofilter. http://bogofilter.sourceforge.net/.

ROBINSON, G. 2003. A statistical approach to the spam problem. *Linux Journal* 107.

ROBINSON, G. 2004. Gary robinson's spam rants. http://radio.weblogs.com/0101454/categories/spam/.

ROTHMAN, K. J. AND GREENLAND, S. 1998. *Modern Epidemiology*. Lippinscott Williams and Wilkins.

SAHAMI, M., DUMAIS, S., HECKERMAN, D., AND HORVITZ, E. 1998. A bayesian approach to filtering junk E-mail. In *Learning for Text Categorization: Papers from the 1998 Workshop*. AAAI Technical Report WS-98-05, Madison, Wisconsin.

Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C. D., and Stamatopoulos, P. 2001a. Stacking classifiers for anti-spam filtering of e-mail. In *Proceedings of EMNLP-01, 6th Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Morristown, US, Pittsburgh, US. Available: `http://www.arxiv.org/abs/cs.CL/0106040`.

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

Sebastiani, F. 2002. Machine learning in automated text categorization. *ACM Computing Surveys 34,* 1, 1–47.

SpamAssassin. 2004. The spamassassin public mail corpus. http://spamassassin.apache.org/publiccorpus.

SpamAssassin. 2005. The apache spamassassin project. http://spamassassin.apache.org.

Streiner, N. 1986. *PDQ Statistics*. B.C. Decker Inc.

Tuttle, A., Milios, E., and Kalyaniwalla, N. 2004. An evaluation of machine learning techniques for enterprise spam filters. Tech. Rep. CS-2004-03, Dalhousie University, Halifax, NS.

Yerazunis, B. 2004a. The spam-filtering plateau at 99.9 In *Proceedings of the Spam Conference*. Available: `http://crm114.sourceforge.net/Plateau_Paper.pdf`.

Yerazunis, W. S. 2004b. CRM114 - the controllable regex mutilator. http://crm114.sourceforge.net/.

Zdziarski, J. A. 2004. The DSPAM project. http://www.nuclearelephant.com/projects/dspam/.

Zhang, L., Zhu, J., and Yao, T. 2004. An evaluation of statistical spam filtering techniques. *ACM Transactions on Asian Language Information Processing (TALIP) 3,* 4, 243–269.