Spam Filtering for Short Messages

Gordon V. Cormack
School of Computer Science
University of Waterloo
Waterloo, Ontario N2L 3G1,
CANADA
gvcormac@uwaterloo.ca

José María Gómez Hidalgo
Universidad Europea de Madrid
Villaviciosa de Odón
28670 Madrid, SPAIN
jmgomez@uem.es

Enrique Puertas Sánz
Universidad Europea de Madrid
Villaviciosa de Odón
28670 Madrid, SPAIN
enrique.puertas@uem.es

ABSTRACT

We consider the problem of content-based spam filtering for short text messages that arise in three contexts: mobile (SMS) communication, blog comments, and email summary information such as might be displayed by a low-bandwidth client. Short messages often consist of only a few words, and therefore present a challenge to traditional bag-of-words based spam filters. Using three corpora of short messages and message fields derived from real SMS, blog, and spam messages, we evaluate feature-based and compression-model-based spam filters. We observe that bag-of-words filters can be improved substantially using different features, while compression-model filters perform quite well as-is. We conclude that content filtering for short messages is surprisingly effective.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering

General Terms

Experimentation, Measurement

Keywords

SMS, blog, spam, email. filtering, classification

1. INTRODUCTION

While the purpose of IR is to deliver data to the user that meets his or her information need, the purpose of spam is the opposite: to deliver data to the user *contrary* his or her need. The purpose of a spam filter is to mitigate the effect of spam, thus improving delivery of relevant data to the user. Although public interest and much research has, to date, been directed primarily at the scourge of email spam [10], many other media are now afflicted. Spam is now prevalent in instant messages, mobile (SMS) messages, web logs and bulletin boards. Often these messages are very short. All the spam needs to deliver its *payload* is to communicate a web URL, telephone number, stock market symbol, or some odd word (e.g. *Gouranga*) that can be found with a search engine.

We are concerned here with content filtering for short messages and short message fragments that range in length from just a few characters (perhaps one word, such as "OK", or a short phrase) to several hundred characters. Content filtering has been shown to be remarkably effective for whole email messages which are typically an order of magnitude larger; our purpose is to examine the transferability of successful email filtering techniques to very short messages. The fundamental uncertainty underlying this question is "are there enough features in short spam messages to distinguish them from non-spam messages?"

The following sections describe the compilation of three independent corpora consisting of mobile (SMS) messages, blog comments and email message fragments. Experiments are conducted to evaluate the effectiveness of state-of-the-art email spam filters, without modification, on the corpora. Further experiments are conducted to investigate the effect of lexical feature expansion – pre-processing the messages to include tokens based on juxtaposed or near-juxtaposed words and characters – on feature-based filters.

2. MOTIVATION

A number of situations arise in which messages to be filtered are short by nature, or their first bytes must be taken to perform filtering. We consider several of such situations:

- *Short Message System filtering.* SMS spam is now prevalent in Singapore and Japan and it will undoubtedly spread throughout the world. The computational power of third generation cell phones and other devices as PDAs is increasing, making increasingly possible to perform spam filtering at the devices, leading to better personalization and effectiveness.

- *Blog Comments.* Among the Web 2.0 tools, blogs are by far the most popular, so that they have been credited with doubling the number of Web servers over the last year. Also, they are the focus of several kinds of spam, including splog (the whole blog is used to promote a product or service), comment spam (comments promoting services with no relation to the blog topic), and traceback spam (spam that takes advantage of the traceback ping feature of popular blogs to get links from them). We focus blog comment spam, as comments are typically short by nature.

- *Filtering email spam at the router.* It is wise to filter spam as soon as it reaches the LAN, and its entry point is the router, which has access only to the first packet.

- *Screening the quarantine folder.* Most spam filters keep a quarantine folder for each user, where spam messages are stored for a time, allowing the users to check it for false positives (legitimate messages classified as spam). Using only the summary of the message (To, From and Subject fields, and perhaps a line or two of the Body field), the user has to quickly decide on the nature of the message. And using exactly that information, a system may rank the messages according to its spamminess score, leading the user to the most likely legitimate messages.

- *Viewing email summaries at a low-bandwidth device.* As in SMS, the messages can be filtered at the PDA or third generation cell phone by using the summaries downloaded from the POP server, that typically include the To, From, Subject, Date, and a configurable part of the Body.

In all these situations, either the messages are short in nature, or the part used for filtering is much shorter than the average email message. It is worth considering which of the currently available content-based filtering techniques may transfer from full-length or long messages to short messages, which may include specific slang, new features like short phone numbers, and in which the feature space is intuitively larger and sparser than in longer messages.

3. BACKGROUND

The problem of spam filtering may be viewed as text classification, typically modeled as a supervised learning task in which a binary classifier is induced on a set of labelled training messages and then used to predict the class of each of a set of unlabeled test messages [24]. Many approaches further abstract each message to a bag of words or a vector of features derived from the message. Cormack and Lynam [6], and Cormack and Bratko [**?**] provide a comparative review of available email spam filter methods, published results and evaluation methods. The TREC spam filter evaluation tracks [8, 5] represent the most comprehensive email spam filter evaluations to date. From these results we observe:

- Practical "Bayesian" spam filters perform quite well. Bogofilter [20] is consistently one of the strongest-performing filters, in its default configuration.

- Feature engineering is a very important consideration. Orthogonal Sparse Bigrams, as embodied in the OSBF-Lua spam filter [1], showed the best performance at TREC 2006 [5].

- Well-regarded machine learning approaches do not necessarily yield better spam filters. However, methods like logistic regression [12] and support vector machines [23], with suitable tuning and feature selection, are competitive.

- Compression models, which differ from traditional machine learning techniques in that they treat messages as bit strings rather than bags of features, yield superior spam filters [2].

The problem of spam classification for short messages has been considered by Healy et al. [13], who compare Knn, SVM and Naive Bayes classifiers on two private corpora consisting of SMS messages and hotel comment forms. Messages were represented as bag-of-words augmented by some statistical features (e.g. the frequency of upper case letters in the text). They conclude that SVM and Naive Bayes substantially outperformed Knn, contrary to their previous results for full email messages. Other published results, in contrast, show SVM and Naive Bayes to outperform Knn on full email messages [7] as well. Mishne and Carmel [19] created a corpus of 1024 blog comment messages and use the content of the original blog posting to predict which of the comments is spam. We used these messages to form our blog corpus; while the results may be compared we note that Mishne and Carmel derive their classifier from extrinsic data (the blog posting) while we use the comments themselves.

More generally, the problem of short text classification has been considered by Zelikovitz et al. [26] who use extrinsic information and transductive learning to bear. Many approaches to spam filtering also use extrinsic information [11]; their consideration is beyond the scope of this research.

Two of us (Gomez Hidalgo et al. [14]) previously reported the effects of feature engineering on the application of standard classifiers to SMS messages. Preliminary results of the work detailed here have been the subject of a poster presentation [4]. The current presentation adds extensive results and analysis on a substantially enlarged SMS corpus, as well as new email and blog corpora which we make available for comparative evaluation.

4. EXPERIMENTAL METHODOLOGY

4.1 Approaches tested

4.1.1 Base filters

We selected for evaluation five spam filtering approaches that compare favourably with others in the literature:

- Bogofilter, a widely deployed open source spam filter. Although dubbed 'Bayesian' Bogofilter employs a novel method for combining features that has come to be known as χ^2 [21]. Bogofilter's features consist of words consisting of alphanumeric sequences of characters in the text. Words appearing in specific header fields are distinguished from those appearing elsewhere. For example, the word "platypus" would be treated as "Subject:platypus" were it to occur in the subject field of the message, "From:platypus" were it to occur in the from field, and so on. Feature selection is effected separately on each message; the 300 features most strongly indicating each class are chosen. The literature shows Bogofilter's performance to be competitive with the best in every evaluation [8, 7, 9, 15].

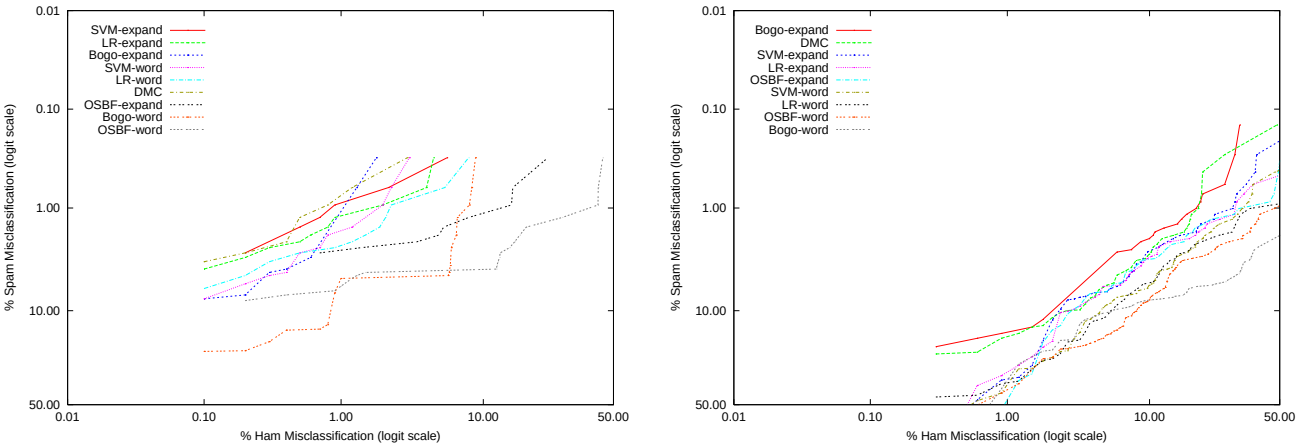
- OSBF-Lua, a new open source spam filter [1] that demonstrated outstanding performance at TREC 2006. OSBF-Lua uses orthogonal sparse bigrams [25], which constructs features from pairs of colocated words, as well as sophisticated training techniques tailored specifically to on-line classification. Orthogonal sparse bigrams appear to improve the performance of many feature-based text classifiers, including those tested here.

Filter method	Filter features	Corpus English SMS
Bogofilter	words	0.4686 (0.3151 - 0.6965)
Bogofilter	expanded	0.0722 (0.0321 - 0.1624)
DMC	-	0.1488 (0.0258 - 0.8514)
LR	words	0.1221 (0.0642 - 0.2321)
LR	expanded	0.0645 (0.0260 - 0.1599)
OSBF-Lua	default	1.0910 (0.6001 - 1.9753)
OSBF-Lua	expanded	0.4187 (0.1481 - 1.1780)
SVM	words	0.0806 (0.0357 - 0.1820)
SVM	expanded	**0.0502** (0.0160 - 0.1575)

Table 1: SMS Corpus Results [1-AUC (%)]

Filter method	Filter features	Blog comment field header	Blog comment field text	Blog comment field header+text
Bogofilter	words	3.4813 (2.6592 - 4.5456)	8.0864 (6.5610 - 9.9287)	3.9596 (2.9515 - 5.2934)
Bogofilter	expanded	1.6793 (1.1415 - 2.4641)	6.7797 (5.4410 - 8.4185)	**1.0673** (0.6566 - 1.7304)
DMC	-	2.6429 (1.8598 - 3.7433)	6.4171 (5.1742 - 7.9336)	1.2732 (0.7691 - 2.1005)
LR	words	3.9492 (2.9137 - 5.3324)	6.5656 (5.1357 - 8.3584)	2.7330 (1.9029 - 3.9109)
LR	expanded	**1.6031** (1.0581 - 2.4220)	**4.7971** (3.6784 - 6.2340)	2.0823 (1.3369 - 3.2298)
OSBF-Lua	default	10.9879 (9.1968 - 13.0775)	9.3913 (7.4740 - 11.7381)	3.4939 (2.5307 - 4.8056)
OSBF-Lua	expanded	2.6342 (1.7675 - 3.9090)	4.6813 (3.3441 - 6.5171)	2.3409 (1.4501 - 3.7579)
SVM	words	4.7170 (3.4101 - 6.4910)	7.1153 (5.5456 - 9.0866)	2.5724 (1.7947 - 3.6746)
SVM	expanded	1.9034 (1.1376 - 3.1684)	5.1392 (3.9794 - 6.6137)	1.9665 (1.1728 - 3.2796)

Table 2: Blog Corpus Results [1-AUC (%)]



SMS messages

Blog comment head+text

Figure 1: ROC Curves – SMS and Blog

- DMC (Dynamic Markov Compression), a spam filter based on data compression models [2] that demonstrates outstanding performance on TREC and other datasets. DMC constructs two variable-order Markov models – one for spam and one for non-spam – and classifies a message according to which of the models predicts it best. DMC treats each message as a string of bits which it models and predicts in sequence, one at a time. As such, it embodies no notion of features or feature engineering.

- SVM (Support Vector Machine), one of the most popular and best performing machine learning methods for classification. SVM finds a hyperplane on a small number of dimensions (features) that separates spam from non-spam, according to a number of optimization parameters. One such parameter is the regularization parameter C, which we set to 100, in accordance with Sculley's recommendation [23]. All other parameters were the default settings of SVMlight [16], which we used for our experiments. SVM is a feature-based classifier, and as such, expects each message to be represented as a vector rather than as text. Our default configuration used words – strings of adjacent alphanumeric characters – as binary features; that is, the value of the feature was 1 if the word was present in the message; 0 otherwise. Words using letters with distinct case were considered distinct.

- LR (Logistic Regression). Although less in vogue than

SVM, logistic regression is a well established and capable classification mechanism. LR computes the coefficients of a linear function to estimate the probability (actually log-odds) that a message is spam. We used LR-TRIRLS [17], open-source logistic regression implementation. We used the same word-based features as for SVM.

4.1.2 Lexical feature expansion

Because many of the messages to be classified are only a few words long, we had reason to believe that feature-based classifiers – that is, all except DMC – would suffer due to insufficient input. We conjectured that the same sort of patterns harnessed by DMC – that is, inter-word and intra-word dependencies – might be harnessed by the feature-based methods (see, for example, Sculley [22]). Furthermore, since the messages were short, there was little risk of exceeding time or memory limits by an aggressive expansion of the number of features.

Prior to our experiments, we chose the following features for our expanded feature set:

- Words, the original default features, were included in the expanded set.
- Orthogonal sparse word bigrams, words separated by 3 or fewer words (i.e. within 5 word positions of one another) were concatenated, along with an indicator of the difference in word positions. For example, "the quick brown fox" would be have OSB features "the(0)quick", "the(1)brown", "the(2)fox", "quick(0)brown", "quick(1)fox" and "brown(0)fox".
- Character bigrams. For example, "the quick" would have character bigram features of "th", "he", "e ", " q", "qu", "ui", "ic" and "ck".
- Character trigrams. For example, "the quick" would have character trigram features of "the", "he ", "e q", " qu", "qui", "uic" and "ick".

In evaluating the SVM and LR classifiers, it was a simple matter to substitute the expanded feature set in place of the word-based set. Altering the feature sets for Bogofilter and OSBF-Lua was more problematic, as these filters take text (email messages) as input and we were loathe to modify the filters themselves. In the case of Bogofilter, we could verify using dump utilities that its feature recognition method was word-based as described above. It was therefore a simple manner to generate text as a sequence of nonsense words, with each word corresponding to a feature in our expanded set. These Bogofilter scanned as words and therefore used as features in its learning algorithm.

We were not sure at all whether or not the same technique would be effective for OSBF-Lua, which uses word-based features but also its own OSB features. Nevertheless, we applied OSBF-Lua to exactly the same sequence of nonsense text that was given as input to Bogofilter.

DMC does not use features and therefore it was not evaluated on the expanded feature set. In total, we applied nine combinations of filter and feature selection to each of the test collections:

1. Bogofilter applied to the raw text so as to use its own word-based features.
2. Bogofilter applied to synthetic text so as to use the expanded feature set.
3. DMC applied to the raw text.
4. Logistic regression applied to the word-based binary feature set.
5. Logistic regression applied to the expanded feature set.
6. OSBF-Lua applied to the raw text so as to use its own word-OSB features.
7. OSBF-Lua applied to the synthetic text.
8. SVM applied to the word-based binary feature set.
9. SVM applied to the expanded feature set.

4.2 Test collections

In this work, we have used an expanded SMS Spam Test collection from our previous work, an email spam test collection from the TREC Spam Filtering Track, and a Blog Comment Spam corpus, all of them described in the following sections. Examples of messages in each test collection are shown in the table 6.

4.2.1 SMS Test Collections

We have built a collection of English SMS messages, including 1002 legitimate messages randomly extracted from the NUS SMS Corpus and the Jon Stevenson Corpus, and 322 SMS spam messages collected from the Grumbletext mobile spam site. The average number of words per message is 15.72, and the average length of a word is 4.44 characters long.

This collection is based on one used in a previous work [14], augmented with 290 new spam messages. It is far bigger than those employed in the related work [13]. This collection is available on request and will be published in due course.

4.2.2 Blog Comment Spam Corpus

For the Blog Comment Spam experiments, we have used the corpus due to Mishne and Carmel [19] [1]. This corpus was built by collecting 50 random blog posts, along with the 1024 comments posted to them; all pages containing a mix of spam and ham (legitimate) comments. The duplicate and near-duplicate comments have been removed, being the number of comments per post between 3 and 96. The authors have manually classified the comments finding 332 to be *legitimate* comments, some of them containing links to related pages and some containing no links; and the other 692 comments being link-spam comments. We split the messages in the corpus into three subcollections:

- The *head* subcollection consists of the header information for the comment; typically a sequence number and the sender's handle and URL or email address.
- The *text* subcollection consists of the text of the message itself.
- The *head+text* subcollection consists of both these fields concatenated together.

[1] available at http://ilps.science.uva.nl/Resources/blogspam/

Filter method	features	Email field From:	Subject:	To:
Bogofilter	words	5.1588 (4.6702 - 5.6954)	15.3653 (14.3347 - 16.4557)	10.8934 (10.1701 - 11.6616)
Bogofilter	expanded	4.5767 (4.0918 - 5.1160)	9.5142 (8.7616 - 10.3241)	10.5598 (9.6997 - 11.4865)
DMC	-	**3.1494** (2.7315 - 3.6290)	**7.1300** (6.5065 - 7.8082)	8.7438 (8.0966 - 9.4374)
LR	words	8.6281 (7.9471 - 9.3614)	16.7609 (15.6556 - 17.9277)	16.9702 (15.8177 - 18.1885)
LR	expanded	3.7561 (3.2947 - 4.2793)	7.8975 (7.3295 - 8.5055)	6.4333 (5.8405 - 7.0818)
OSBF-Lua	default	18.4129 (17.3669 - 19.5070)	18.4357 (17.3531 - 19.5699)	15.8124 (14.8803 - 16.7914)
OSBF-Lua	expanded	3.2874 (2.9112 - 3.7104)	8.4221 (7.7404 - 9.1579)	**7.0672** (6.4083 - 7.7882)
SVM	words	4.0952 (3.6030 - 4.6513)	10.8001 (9.9336 - 11.7323)	16.2741 (15.0478 - 17.5797)
SVM	expanded	3.3418 (2.9289 - 3.8106)	10.7285 (9.9236 - 11.5902)	8.3182 (7.5395 - 9.1693)

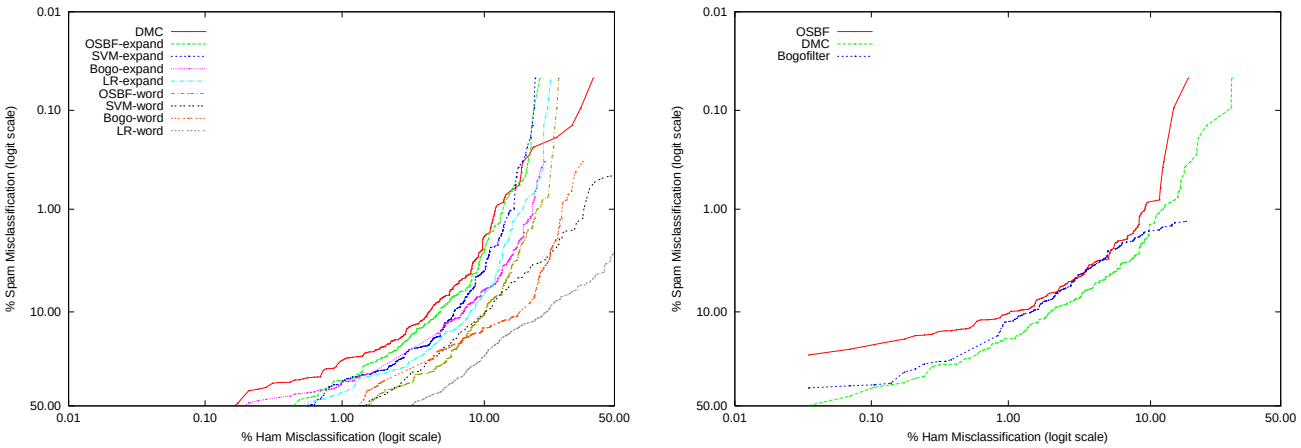
Table 3: Email Corpus Results [1-AUC (%)] – Part I

Filter method	features	Email field from+subject+to	200-byte body prefix	from+subject+to+prefix
Bogofilter	words	4.4353 (3.9397 - 4.9901)	6.4291 (5.8231 - 7.0933)	3.4769 (2.9886 - 4.0417)
Bogofilter	expanded	2.3644 (2.0328 - 2.7485)	5.8266 (5.1569 - 6.5773)	2.0444 (1.6963 - 2.4621)
DMC	-	**1.3604** (1.1577 - 1.5980)	**2.7854** (2.4078 - 3.2202)	**0.9041** (0.7493 - 1.0906)
LR	words	8.3496 (7.6101 - 9.1537)	6.3397 (5.6763 - 7.0747)	3.5156 (3.1057 - 3.9773)
LR	expanded	2.5058 (2.2057 - 2.8456)	3.9118 (3.5346 - 4.3275)	1.5854 (1.3657 - 1.8399)
OSBF-Lua	default	3.5610 (3.1366 - 4.0405)	5.6753 (5.1580 - 6.2412)	2.2847 (2.0083 - 2.5982)
OSBF-Lua	expanded	1.6752 (1.4436 - 1.9432)	4.4860 (3.9952 - 5.0341)	1.2893 (1.0922 - 1.5213)
SVM	words	4.0450 (3.5581 - 4.5953)	8.3954 (7.6497 - 9.2066)	2.5440 (2.2301 - 2.9009)
SVM	expanded	2.0136 (1.7353 - 2.3356)	6.8312 (6.1733 - 7.5535)	1.2423 (1.0771 - 1.4326)

Table 4: Email Corpus Results [1-AUC (%)] – Part II

Filter	Entire message (raw)
Bogofilter	1.2697 (0.9901 - 1.6270)
DMC	0.8756 (0.7239 - 1.0587)
OSBF-Lua	**0.4953** (0.3980 - 0.6162)

Table 5: Email Corpus Results [1-AUC (%)] – Long messages



Email from+subject+to Email raw

Figure 2: ROC Curves – Email

4.2.3 *TREC Spam Corpus*

In these experiments, we have also used the Spam Corpus provided for the TREC 2005 Spam Filtering Track. As the messages in this corpus are in chronological order, and that this order affects filtering performance [7], we extracted ten *chronological* splits from the corpus. Each split consists of 1500 consecutive messages in the corpus, with the first 1000 used as the training set and the following 500 used as the test set. These splits model real spam filter usage, in which the classifier must be constructed from messages seen before those to be classified.

Instead of using the full text of the messages, and addressing the scenarios described in the motivation, we have built several test collections by using only selected fields of the

```
Urgent! call 09061749602 from Landline. Your complimentary 4*
Tenerife Holiday or 10,000 cash await collection SAE TCs BOX
528 HP20 1YF 150ppm 18+

Hey! do u fancy meetin me at 4 at cha  hav a lil beverage on
me. if not txt or ring me and we can meet up l8r. quite tired
got in at 3 v.pist ;) love Pete x x x
```

```
Posted by: <a target="_blank" title="http://www.march--madness.biz"
href="http://blog.makalumedia.com/mt-comments.cgi?__mode=red;id=10327">
march madness</a> at March 16 2005 12:56 PM 1950 march madness march,
madness betting march madness odds march madness gambling march madness
..

Posted by: <a href="mailto:plemer@nildram.co.uk">peter lemer</a>
at February 11 2005 10:13 AM I have grown with Palm Desktop for,
a whole now ( not least because of my Palm handheld),andall my,
..
```

Table 6: An example of an spam and a ham message in the SMS and the Blog Comments test collections.

messages.

- The *From:* subcollection includes only the value of the From field, and leads to an average of 2.77 words per message.
- The *To:* subcollection includes what the To field contains, and each message has an average of 1.95 words.
- The *Subject:* subcollection, with the information in the field Subject, and an average of 4.72 words per message
- The *fst* subcollection, containing the above three fields concatenated.
- The *200b* subcollection, containing the first 200 bytes of the body of each message excluding header and MIME sub-header information. The average number of words per message is 24.58.
- The *fst200* subcolllection, consisting of the From:, Subject:, To:, and 200b fields concatenated together.

The rationale for these choices is to examine the performance of filters on a set of natural prefixes such as might be seen by a router or shown in a quarantine summary or on a low-bandwidth device.

For comparison, we also ran three of the filters – those previously configured for TREC evaluation – on the full text of the same email messages.

4.3 Evaluation methodology

We used the TREC Spam Filter Evaluation Toolkit[2] and the associated evaluation methodology [6]. However, the the toolkit assumes that messages will be evaluated on-line in chronological order, an assumption that cannot be met in our experiments. First, the SMS and Blog collections are undated, so it is impossible to know the correct chronological order. Second, these corpora are quite small – of the order of 1000 messages – and so it would be difficult to achieve statistically precise results from a single on-line evaluation. Therefore, for these corpora, use the "delayed feedback" facility of the toolkit to perform 10-fold cross validation (cf. [7]). On the other hand, the email collection is chronologically ordered and contains ample (92,100) examples. Yet SVM and LR classifiers require adaptation for use in an on-line setting [12, 23], which we wished to avoid in the interests of a standard comparison. Furthermore, we wished the results to be comparable to those for SMS and Blog messages. So we chose ten different chronological splits of the corpus. Our chronological splits each consisted of 1000 training messages and 500 test messages, with no message appearing in two or more splits. In each split all training messages predated all test messages.

The TREC toolkit performs ROC analysis on the classification results and computes several summary statistics as well as an ROC curve which indicates the tradeoff that may be achieved by trading off false positives against false negatives. Although no scalar value completely characterizes the performance of a spam filter, due to space constraints we choose one – ROC area under the curve (AUC), the primary measure used at TREC. Following TREC, we report 1-AUC as a percentage (i.e. AUC = 0.982 is reported as 1-AUC(%) = 1.8. AUC values with 95% confidence intervals are reported[3] as calculated by the toolkit. Space precludes us from presenting ROC curves for all evaluation runs; selected examples are presented to demonstrate general agreement between AUC and filter performance throughout its operating range.

5. RESULTS AND DISCUSSION

Tables 1 through 4 report 1-AUC(%) summary results for all combinations of spam filter, feature engineering and message collection. (Recall that smaller numbers indicate better performance.) Table 5 shows, for comparison with the state of the art, show the result of running the three on-

[2]available at http://plg.uwaterloo.ca/~trlynam/spamjig/

[3]to the same ridiculously large number of significant figures

line filters on the entire text of each message. Figures 1 show 2 the ROC curves corresponding to one column selected from each of these tables.

On the SMS corpus (table 1, figure 1, left panel) using default features, SVM (0.0806) slightly outperforms LR (0.1221) and DMC (0.1488), and substantially outperforms Bogofilter (0.4686) and OSBF-Lua (1.0910). Feature expansion dramatically improves Bogofilter (0.0722) and substantially improves SVM (0.0502), LR (0.0806) and OSBF-Lua (0.4187), although OSBF-Lua remains uncompetitive. DMC is not subject to feature expansion.

Results on the Blog corpus (table 2; figure 1 right panel) show substantial improvement due to feature expansion for all methods on all subcollections. Except for OSBF-Lua's default configuration, all methods did a better job of classifying the header information than the text of the message. This result is surprising as the header information is short, but perhaps structured in such a way that it is more difficult to disguise its "spamminess." Most filters performed substantially better on head+text than on either separately, notable exceptions being SVM and LR with expanded features. One possible explanation is that SVM and LR are discriminative rather than generative filters, and therefore less able to harness heterogeneous indicators of spamminess. Overall, LR is the best performer on the individual fields while Bogofilter is best for the combination.

The TREC corpus (tables 3 and 4; figure 2, left panel) demonstrates once again that feature expansion improves all filters on all subcollections. DMC, however, shows the best performance on the From and Subject fields, as well as both combinations, From+Subject+To and From+Subject+To+prefix. On the To field, DMC falls to fourth place after OSBF-Lua, SVM and LR. All filters show dramatically better performance on the combined fields than on the individual fields themselves. It is perhaps not surprising that the 1-AUC numbers are quite high for these very terse header fields; more surprising is that the combination of these fields yields a classifier that outperforms the same method applied to 200 bytes of the message text.

The results for OSBF-Lua and Bogofilter on the full email messages (table 5; figure 2, right panel) are much better than those for the fragments and combination of fragments in our short corpora; more particularly so when compared to their default feature configuration. DMC also shows improved performance on the full messages, but insubstantially (and certainly not significantly) so. This observation is consistent with the fact that DMC, in its default configuration, uses only the first 2500 bytes of the raw message! Overall, the results – even for the full text of email messages – are substantially inferior to those reported elsewhere for a superset of this data [7]. The most likely difference is in training set size; our corpus has a training set size of 1000 messages while the full corpus has over 90,000.

6. CONCLUSIONS

Short messages contain an insufficient number of words to properly support bag of words or word bigram based spam classifiers. Their performance is improved markedy by expanding the set of features to include orthogonal sparse word bigrams and also to include character bigrams and trigrams. DMC – a compression-model-based classifier – does not rely on explicit featurization and performs well on short messages and message fragments. Overall, performance on selected fields or prefixes of messages is quite good; further analysis is required to determine to which aspects of the various messages the classifiers are sensitive. We have no reason to believe we have yet found the optimal set of features, or the optimal method of combining the results of multiple classifiers on multiple fields of the message. The results of Bratko et al. [3] or Lynam et al. [18] may be useful in this regard.

7. REFERENCES

[1] Assis, F. OSBF-Lua - A text classification module for Lua – the importance of the training method. In *Fifteenth Text REtrieval Conference (TREC-2006)* (Gaithersburg, MD, 2006), NIST.

[2] Bratko, A., Cormack, G. V., Filipič, B., Lynam, T. R., and Zupan, B. Spam filtering using statistical data compression models. *Journal of Machine Learning Research 7*, Dec (2006), 2673–2698.

[3] Bratko, A., and Filipič, B. Exploiting structural information for semi-structured document categorization. *Information Processing & Management 42*, 3 (2006), 679–694.

[4] Cormack, G., Hidalgo, J. M. G., and Snz, E. P. Feature engineering for mobile (SMS) spam filtering. In *30th ACM SIGIR Conference on Research and Development on Information Retrieval* (Amsterdam, 2007).

[5] Cormack, G. V. TREC 2006 Spam Evaluation Track overview. In *Fifteenth Text REtrieval Conference (TREC-2006)* (Gaithersburg, MD, 2006), NIST.

[6] Cormack, G. V. On-line supervised spam filter evaluation. *ACM Transactions on Information Systems 25*, 3 (July 2007).

[7] Cormack, G. V., and Bratko, A. Batch and online spam filter comparison. In *Conference on Email and Anti-Spam, CEAS 2006* (Mountain View, CA, July 2006).

[8] 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.

[9] Cormack, G. V., and Lynam, T. R. On-line supervised spam filter evaluation. http://plg.uwaterloo.ca/~gvcormac/spamcormack, 2006.

[10] Goodman, J., Cormack, G. V., and Heckerman, D. Spam and the ongoing battle for the inbox. *Commun. ACM 50*, 2 (2007), 24–33.

[11] Goodman, J., Heckerman, D., and Rounthwaite, R. Stopping spam. *Scientific American 292*, 4 (April 2005), 42–88.

[12] Goodman, J., and tau Yih, W. Online discriminative spam filter training. In *The Third Conference on Email and Anti-Spam* (Mountain View, CA, 2006).

[13] Healy, M., Delany, S., and Zamolotskikh, A. An assessment of case-based reasoning for short text message classification. In *Procs. of 16th Irish Conference on Artificial Intelligence and Cognitive Science, (AICS-05)* (2005), N. Creaney, Ed., pp. 257–266.

[14] Hidalgo, J. M. G., Bringas, G. C., Sanz, E. P., and Garcia, F. C. Content based SMS spam

filtering. In *DocEng '06: Proceedings of the 2006 ACM Symposium on Document Engineering* (New York, NY, USA, 2006), ACM Press, pp. 107–114.

[15] Holden, S. Spam filters. http://freshmeat.net/articles/view/964/, 2004.

[16] Joachims, T. Making large-scale support vector machine learning practical. In *Advances in Kernel Methods: Support Vector Machines*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT Press, 1998.

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

[18] Lynam, T. R., and Cormack, G. V. On-line spam filter fusion. In *29th ACM SIGIR Conference on Research and Development on Information Retrieval* (Seattle, 2006).

[19] Mishne, G., and Carmel, D. Blocking blog spam with language model disagreement, 2005.

[20] Raymond, E. S., Relson, D., Andree, M., and Louis, G. Bogofilter. http://bogofilter.sourceforge.net/, 2004.

[21] Robinson, G. A statistical approach to the spam problem. *Linux Journal 107* (March 2003), 3.

[22] Sculley, D., and Brodley, C. E. Compression and machine learning: A new perspective on feature space vectors. In *Data Compression Conference (DCC 06)* (Snowbird, 2006), pp. 332–341.

[23] Sculley, D., and Wachman, G. M. Relaxed online support vector machines. In *30th ACM SIGIR Conference on Research and Development on Information Retrieval* (Amsterdam, 2007).

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

[25] Siefkes, C., Assis, F., Chhabra, S., and Yerazunis, W. S. Combining winnow and orthogonal sparse bigrams for incremental spam filtering. In *PKDD* (2004), J.-F. Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi, Eds., vol. 3202 of *Lecture Notes in Computer Science*, Springer, pp. 410–421.

[26] Zelikovitz, S., and Hirsh, H. Improving short text classification using unlabeled background knowledge. In *Proceedings of ICML-00, 17th International Conference on Machine Learning* (Stanford, US, 2000), P. Langley, Ed., Morgan Kaufmann Publishers, San Francisco, US, pp. 1183–1190.