How does the performance of spam filters change with decreasing message length?

Adrian Scoică (as2270), Girton College

Total word count (excluding numbers, tables, equations, captions, headings, and references): 4118 words.

1 Problem Description and Project Motivation

Spam e-mails are unsolicited bulk e-mail messages, often with a commercial or phishing end-purpose, which decrease the quality and usability of e-mail services. The seminal work which first attempted to use machine learning techniques for automatically detecting spam messages was described in (Sahami et al., 1998), the paper which cast the problem in the framework of *text categorization*, and opened it up for further research. The Naive Bayes classifiers it proposed were shortly afterwards proven to outperform simple filters based on keyword patterns in (Androutsopoulos et al., 2000), a work which also justified the need for cost-sensitive evaluation metrics, formalized the classification process, and emphasized the distinction between *symmetric* and *asymmetric classification*.

Besides Naive Bayes classifiers, such as the **Multinomial Naive Bayes** (**MNB**), other successful techniques described in the spam filtering literature are **Support Vector Machines** (**SVM**), **Bayesian Logistic Regression** (**BLR**), and techniques based on language modelling. The application of Support Vector Machines to spam filtering was first described in (Drucker et al., 1999), while Bayesian Logistic Regression was first described in (Genkin et al., 2007).

Although varied, many of these approaches rely exclusively on the body of the e-mail messages for solving the classification problem. Evidence of this trend can be seen as early as (Androutsopoulos et al., 2000), where it is claimed that the *language of spam* is sufficiently different from the language of genuine messages to enable training automatic classifiers. However, more recent work acknowledges that e-mails are structured documents, and that performance can be improved by interpolating signals from more than one document field. The **Interpolated Language Model** (**ILM**) classification described in (Medlock, 2006) is notable for achieving state of the art classification accuracy against all three previously-mentioned methods (**MNB**, **SVM**, and **BLR**), by using a generative classification model which interpolates posterior probabilities from the subject field, and from the body field.

When moving away from e-mails and towards less content-rich types of messages, such as tweets or local listings descriptions, classification algorithms have to rely more heavily on non-textual message features. Two of the most recent publications on web-service spam filtering claim that **conventional filtering methods are ineffective**, and propose alternative solutions: (Song et al., 2011) describes spam classification based on social graph metrics, while (Thomas et al., 2011) describes a spam detection system built primarily on location information: URL, IP, DNS, HTTP headers, etc.

Contrary to these recent claims, Cıltık and Güngör empirically stated in (Çıltık and Güngör, 2008), a paper on the applicability of n-gram models to spam filtering, that humans only rely on the title and/or a small fraction of text at the beginning of an e-mail message to recognize whether it is spam or non-spam, theoretically making traditional, n-gram based filters applicable to spam detection in short-text messages.

The purpose of this project was to investigate these two contradictory claims, by reporting on how the performance of *traditional*, text-based spam filters degrades with decreasing test message length, and on whether the classifiers are robust to decreasing training *and* test message length.

2 Project Methodology

The project initially verified the experimental results reported in (Medlock, 2006), which served as the *baseline* for further investigations. We then conducted experiments by varying the size of the training and/or testing data in the corpus through truncation, traced the corresponding changes in performance, and drew the conclusions based on the observed effects.

2.1 Classifiers Used

The classifiers used in the project were the same four classifiers reported in (Medlock, 2006): a **Multinomial Naive Bayes classifier**, a **Support Vector Machine classifier**, a **Bayesian Logistic Regression classifier**, and unigram- and bigram-based **Interpolated Language Model** classifiers. Concrete details on each of the classifiers is given in sections 2.1.1 to 2.1.4.

2.1.1 Multinomial Naive Bayes

Multinomial Naive Bayes is used as the classical baseline for Natural Language Processing (NLP) classification tasks, and its main property is the simplifying assumption that features are independent of one another. Its main advantages are *simplicity* and *computational efficiency*, while its main disadvantage is that it is *susceptible to over-fitting*. The MNB implementation that we used for this project is part of the **scikit-learn** Python library[1], which is developed at INRIA. The library was first released in 2010, under an open source license, and a brief overview of the project is given in (Pedregosa et al., 2011).

Selecting *the most informative features* for representing documents is a key aspect to the performance of a MNB classifier. After carrying out the pre-processing steps described in section 2.2.1 on the corpus data, we are left with a vocabulary of 73261 distinct tokens from among which to select features. We ranked all of the tokens based on the *mutual information metric* (MI), given in equation 1, and kept the top N_{feat} ones. The probabilities were estimated based on counts from the training set. An additional heuristic that we used to improve classification accuracy was to discard from the ranking all features which occur in less than $N_{min} = 3$ training messages, as a measure to counteract the over-fitting disadvantage of MNB classifiers.

$$MI(X, C) = \sum_{x \in \{0,1\}, c \in \{G,S\}} P(X = x, C = c) \log \frac{P(X = x, C = c)}{P(X = x) \cdot P(C = c)} \tag{1}$$

Furthermore, deciding on the *optimal number of features* based on which to build the classifier is just as important as selecting the best features. Early papers on Naive Bayes classifiers, such as (McCallum et al., 1998), suggest that more features provide more discriminative power to the classifier, while later work on optimal feature selection schemes, described in (Rogati and Yang, 2002), indicates the contrary: fewer features provide more generalization power and prevent over-fitting. In order to decide on the optimal number of features to use, we traced the symmetrical and asymmetrical recall curves of the subject and body section-based MNB classifier, depending on the number of features. The graphs are shown in figures 1 and 2 below.

Similarly, the symmetrical and asymmetrical accuracy graphs for the body and subject sections of the test e-mails are given in figures 3 and 4, respectively. All figures indicate that while increasing the number of features results in a performance boost, the performance of the MNB classifier ceils out at around approximately 3000 features, beyond which it starts being affected by over-fitting. All of the results further reported in this paper are based on a number of 3000 **features**.

[1] http://scikit-learn.org/stable/about.html

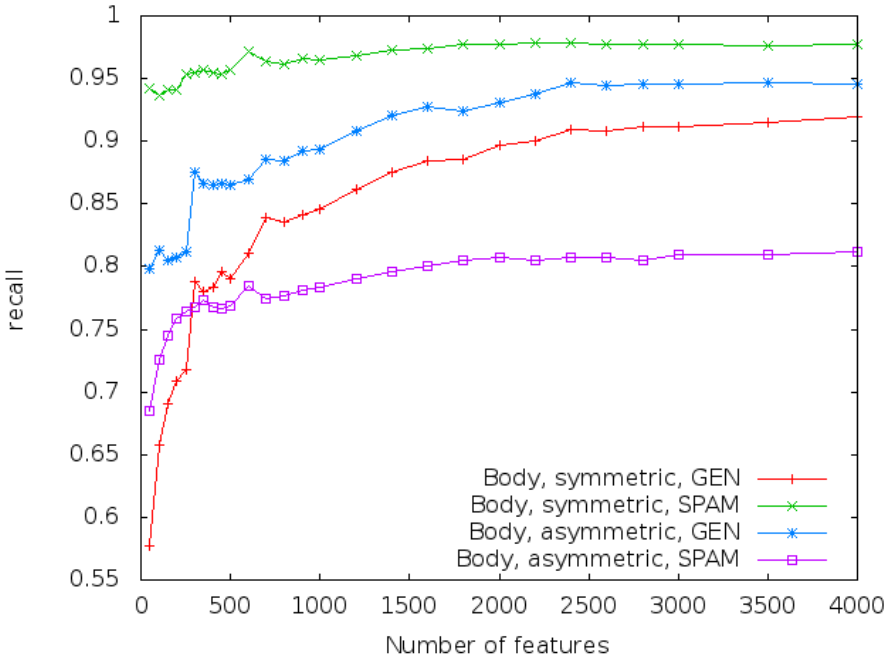


Figure 1: MNB recall on the body of test e-mail messages, depending on the number of features. Asymmetric results are reported for $\lambda = 9$.

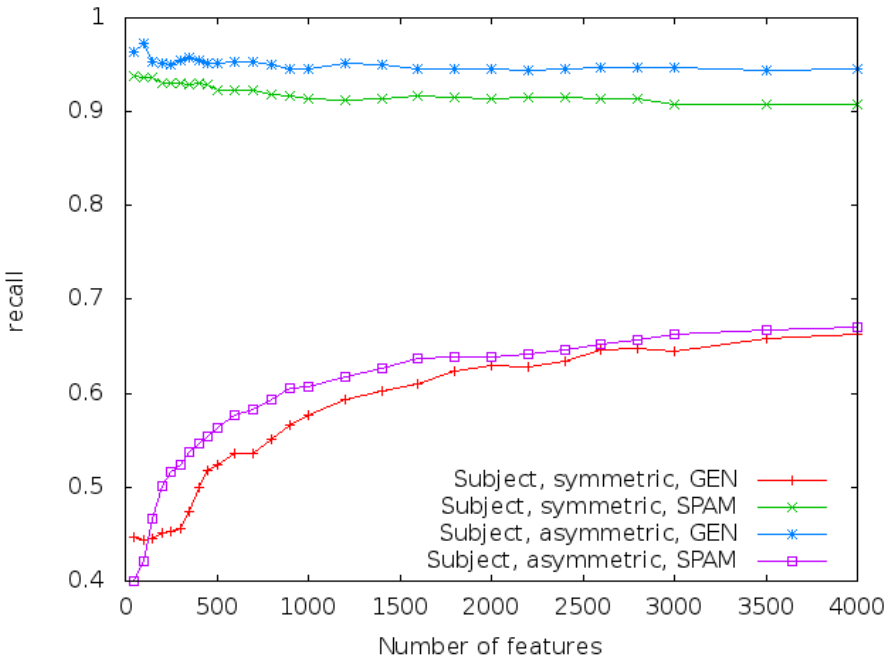


Figure 2: MNB recall on the subject of test e-mail messages, depending on the number of features. Asymmetric results are reported for $\lambda = 9$.

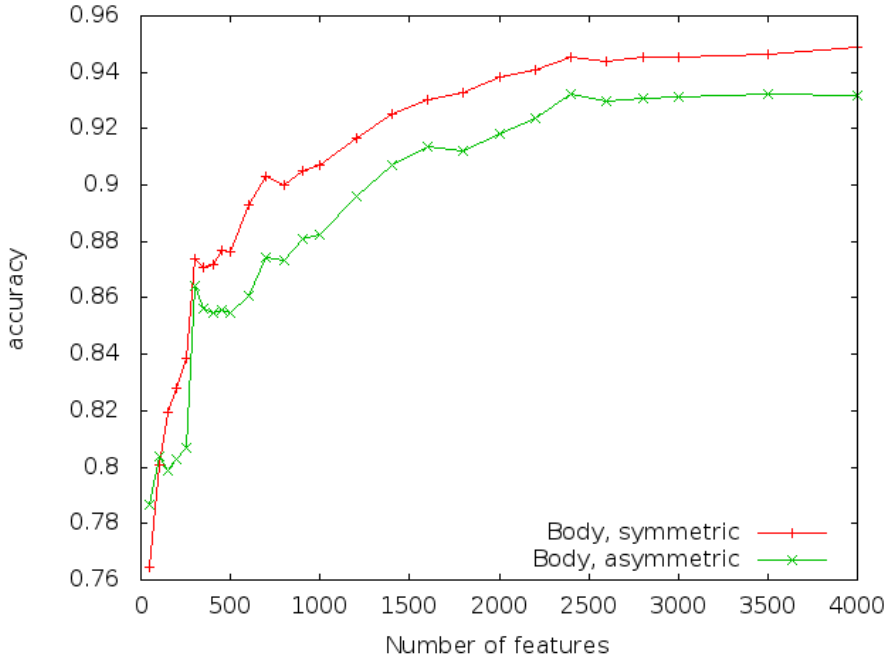


Figure 3: MBN accuracy on the body of test e-mail messages, depending on the number of features. Asymmetric results are reported for $\lambda = 9$.

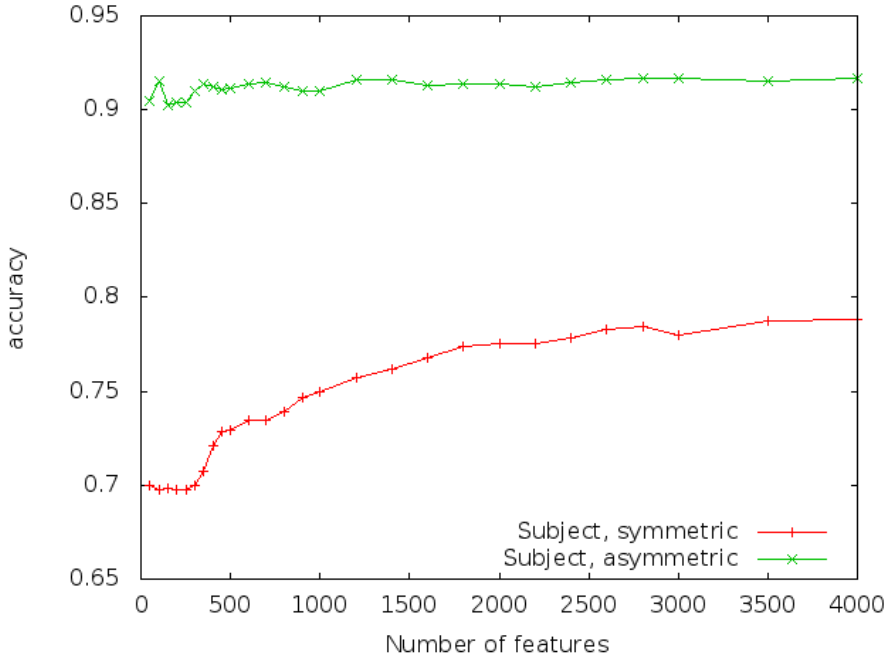


Figure 4: MNB accuracy on the subject of test e-mail messages, depending on the number of features. Asymmetric results are reported for $\lambda = 9$.

2.1.2 Support Vector Machines

Support Vector Machines are a class of models used in machine learning for classifying data which can be represented as a collection of points in a given (high-dimensional) space. Classification is done by identifying a hyperplane which can separate between the two classes of points with as large a margin as possible. In comparison to the Multinomial Naive Bayes classifiers previously discussed, Support Vector Machines are more robust to over-fitting.

The SVM implementation that we used for running the experiments in this project was SVM^{light}, which is written in C and distributed freely on the official page maintained by Thorsten Joachims[2]. (Joachims, 1999) gives a brief description of the algorithms used, and their implementation details.

$$idf(term, Corpus) = \log \frac{|Corpus|}{|1 + \{D \in Corpus : term \in D\}|} \tag{2}$$

SVM^{light} requires the input data to be converted to normalized $tf * idf$ vectors. We computed the term frequencies and inverse document frequencies based on the training section of the GenSpam corpus, using raw frequencies for tf, and the denominator-adjusted formula for idf. The modified idf is described in equation 2, and its purpose is to avoid getting undefined values on words in the test set which are not present in the training set. As reported in (Medlock, 2006), we used $C = 1$ for the regularization parameter, and the linear kernel function option.

When run in classification mode, the output is the value of the decision function on each of the instances in the test set, with the sign of this value indicating the predicted class of the instance. For asymmetric classification, we adjusted the cost-factor parameter of the classifier to $\lambda = 9$, to specify how many times training errors on positive examples outweigh errors on negative examples.

2.1.3 Bayesian Logistic Regression

Bayesian Logistic Regression is a discriminative approach to classification. It is a regression technique that uses the prior selected for the feature weights to avoid problems related to over-fitting. The BLR implementation used for running the experiments in this project is BBR, which is written in C++ and distributed on the project's official page[3]. An in-depth explanation of the theoretical foundations on which BBR is built is given in (Genkin et al., 2007).

BBR shares the same input format with SVM^{light}, and we used the same formulae as those described in section 2.1.2 to compute the normalized $tf * idf$ input vectors for the training and test data. We used a Gaussian prior distribution and the **--autosearch** parameter for optimizing prior variance via 10-fold cross-validation, as mentioned in (Medlock, 2006). Unlike SVM^{light}, BBR outputs the probability for each of the instances in the test set to belong to the positive class (which we arbitrarily chose to represent the $spam$ class). For asymmetric classification, BBR allows for manually setting the classification threshold via the **--probthres** parameter during both training and classifying.

[2] http://svmlight.joachims.org/

[3] http://www.bayesianregression.org/bbr.html

2.1.4 Interpolated Language Models

Interpolated Language Models are a generative classification technique which exploits the structure of documents by interpolating the estimated posterior probabilities for various document nodes to produce the *class* posterior probability. We only considered two e-mail document nodes in our experiments, following the advice given in (Medlock, 2006): the subject line, and the body text. The rewriting of the weighted linear interpolation for the two-node case is given in equation 3. Note that the interpolation only has one free parameter, λ_{body}, since $\lambda_{body} + \lambda subj = 1$.

$$P(C|D) = \lambda_{body} P(C|body(D)) + (1 - \lambda_{body}) P(C|subj(D)) \tag{3}$$

$$P(C|comp(D)) = \frac{P(C) \cdot P(comp(D)|C)}{\sum_{C \in \{G,S\}} P(C) \cdot P(comp(D)|C)}, comp \in \{subj, body\} \tag{4}$$

We computed the individual posterior probabilities, $P(C|body(D))$ and $P(C|subj(D))$, using the Bayes rule given in equation 4, where the denominator of the fraction has been expanded according to the law of total probability. The language model probabilities, $P(comp(D)|C)$, were computed using unigram- and bigram-language models built from the training section of the GenSpam corpus for each of the two classes (genuine and spam e-mail messages), which estimate the probability of a string of tokens based on the assumption that any given token in the sequence is dependent only on the previous one (for unigram language models) or two (for bigram language models) tokens in the string. We didn't test *n*-gram models of higher orders, as the original paper reports that data sparsity results in poor results with trigrams. The specialization of the formula for unigrams is given in equation 5, while the specialization for bigrams is given in equation 6.

$$P_{unigram}(t_1, t_2, ...t_{len(comp(D))}) = \prod_{i=1}^{len(comp(D))} P(t_i) \tag{5}$$

$$P_{bigram}(t_1, t_2, ...t_{len(comp(D))}) = \prod_{i=2}^{len(comp(D))} P(t_i|t_{i-1}) \tag{6}$$

The language models providing the probabilities in equations 5 and 6 are built from the training section of the GenSpam corpus. For both the unigram and bigram models, we discarded single occurrences from the corpus. The floor value for unseen words in the unigram case was set to 10^{-8} for the genuine message LM, and to $1.2 \cdot 10^{-8}$ for the spam message LM, as indicated in the original paper. Similarly, in the case of bigrams, the probability of unseen words was set to 10^{-8} for genuine messages and 10^{-7} for spam messages. The discounting scheme we used for building the bigram model was Good-Turing, and the n-gram cutoff points used were 2 occurrences for the body field of spam training messages, and 1 occurrence for all other language models.

Using the language models to estimate probabilities, the symmetric classification decision for a given document D is made according to equation 7 below (the asymmetric decision is similar, but the argmax is weighted). In standard Bayesian classifiers, the $argmax$ function allows us to disregard the denominator in equation 4, but since interpolation requires true probabilities, we had to include it in the computations. This was especially problematic when working with *log-transformed probabilities* and the log-transformed versions of the formulae 4 to 6, because the denominator in equation 4 is a sum, and using $\log(a + b) = \log(e^{\log(a)} + e^{\log(b)})$ causes underflow which makes the decision impossible in most of the cases. To solve this problem, it was necessary to use a high-precision arithmetic library in our implementation.

$$Decision(D) = \arg\max_{c} \{P(C = c|D)\} \tag{7}$$

2.2 Data Corpus

The results reported in the project were computed based on the **GenSpam** corpus, to enable direct comparison with (Medlock, 2006). GenSpam was published in 2003, it reflects the features of typical spam messages from the 2002-2003 time period, and it is freely available for download from Ben Medlock's web page[4]. According to a later literature survey (Guzella and Caminhas, 2009), GenSpam is also the mixed spam-and-legitimate message dataset with the highest spam-to-legitimate message ratio: 3.38, in comparison to 0.16 for **LingSpam**, 0.54 for **SpamAssassin** and 0.77 for **PU1**, which were the three most popular datasets for the task at the time of the survey. This skewed ratio ultimately proved to impact negatively on the performance of the SVM classifier.

2.2.1 GenSpam Description and Pre-Processing

GenSpam is composed of three parts:

- a **training set**, comprising of 8158 genuine e-mail messages, and 30088 spam e-mail messages;
- an **adaption set**, comprising of 300 genuine e-mail messages, and 300 spam e-mail messages;
- a **test set**, comprising of 754 genuine e-mail messages, and 797 spam e-mail messages;

Note that for the training set, the counts are different from those reported in (Medlock, 2006), indicating that there may have been minor changes to the data sets. Furthermore, for the purpose of the experiments described in this report, we discarded the adaption set, added <ROOT> / </ROOT> elements at the beginning and end of the data files to make them into valid XML, and converted all of the messages from their native character encoding to UTF-8 encoding. Further textual pre-processing on the raw data included escapting the "&" sign in the XML entities used as placeholders for named-entities, numbers, and URLs during the anonymization process.

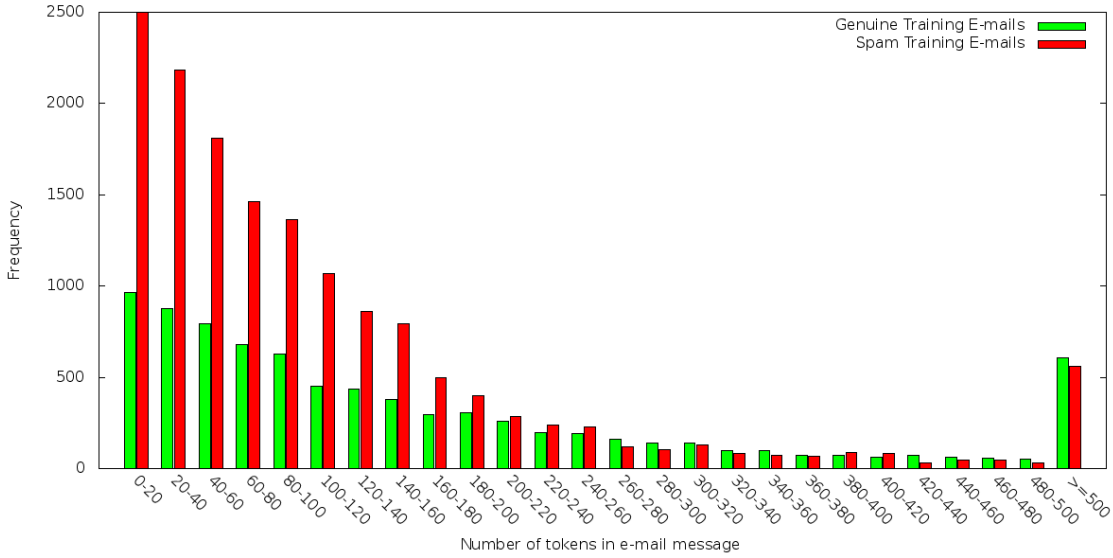
In the XML structure used for representing the e-mails in the corpus, a distinction is being made between normal, top-level, e-mail text (enclosed between <TEXT_NORMAL> and </TEXT_NORMAL> tags), and quoted text (enclosed between <TEXT_EMBEDDED> and </TEXT_EMBEDDED> tags). For the purpose of our experiments, I stripped the tags and treated all of the text as plain body text.

As instructed in (Medlock, 2006), we removed from the data all tokens longer than 15 characters, as well as tokens containing non-letters, while preserving the original casing. We verified that converting tokens to lowercase resulted in a decrease in classifier performance, as claimed in the original paper. No other pre-processing steps were carried out on the data.

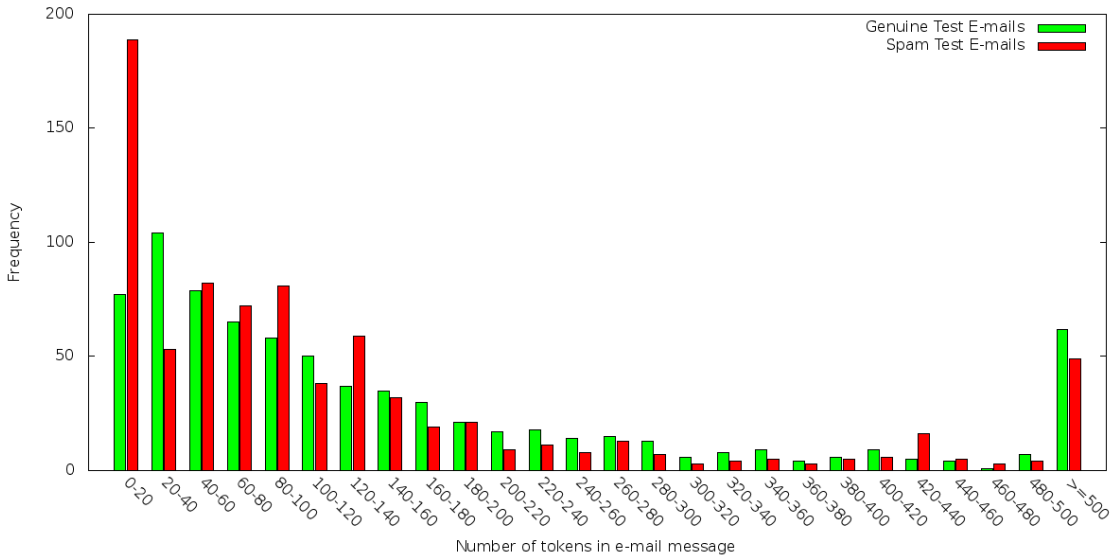
As described in section 2.3.2, the experimental method involves gradually truncating the length of the messages to a fixed number of tokens. In order to understand the effect that the truncation will have on the overall dataset, it is important to profile the dataset beforehand. While we know that the GenSpam **training set has a highly skewed spam-to-genuine e-mail ratio** (negatively impacting the performance of SVM), figure 5a reveals that the frequencies are roughly the same for e-mails of 200 words or longer, and that the shorter the messages, the greater the proportion of spam in the data set. Although clipped on figure 5a, it is also worth noting that the majority (17438 out of 30088) of spam e-mails are under 20 tokens long.

While the number of spam e-mails in the test set that are under 20 tokens long is still greater than the number of genuine messages in the same length group, it can be said that **the test data is balanced between spam and genuine messages**, and that it is not dominated by very short messages. These properties guarantee that truncation introduces an *incremental variation* in the test data set.

[4] http://www.benmedlock.co.uk/genspam.html



(a) Training set. The bar for spam e-mails shorter or equal in length to 20 tokens is clipped at 2500, as the frequency count for these e-mails is 17438.



(b) Test set.

Figure 5: Histogram of the GenSpam e-mails, by length.

2.3 Evaluation

2.3.1 Evaluation Metrics

Similarly to the results reported in (Medlock, 2006), we measured performance based on two types of evaluation schemes: **symmetric classification**, and **asymmetric classification**. The following notations will be used in the equations describing the evaluation metrics thoughout this section:

- Let D be an e-mail document;
- Let C be a document class, with two possible values: $genuine$, and $spam$, abbreviated G and S;
- Let N_G and N_S be the total number of genuine and spam e-mail documents in the test set, respectively;
- Let $n_{C_1 \to C_2}$, where $C_1, C_2 \in \{G, S\}$, be the number of e-mail documents in class C_1 that are classified by the filter into class C_2;
- Let λ be the ratio of the cost of a $G \to S$ misclassification to the cost of a $S \to G$ misclassification;

Symmetric Classification

In the case of **symmetric classification**, erroneously blocking a legitimate message as spam is considered to be as bad as letting a spam message pass through the filter ($\lambda = 1$). The *classification criterion* for labelling a message as spam is given in equation 8.

$$\frac{P(C = S|D)}{P(C = G|D)} > 1 \tag{8}$$

Genuine message recall (Rec_G) and spam recall (Rec_S) are given by equations 9, while the accuracy and error rate of the classifier are given by equations 10.

$$Rec_G = \frac{n_{G \to G}}{n_{G \to S} + n_{G \to G}} \quad Rec_S = \frac{n_{S \to S}}{n_{S \to S} + n_{S \to G}} \tag{9}$$

$$Acc = \frac{n_{G \to G} + n_{S \to S}}{N_G + N_S} \quad Err = \frac{n_{G \to S} + n_{S \to G}}{N_G + N_S} \tag{10}$$

We also report the Total Cost Ratio (TCR), introduced in (Androutsopoulos et al., 2000) and given in equation 11, which indicates how much time a given spam filter saves in comparison to the baseline of using no spam filter.

$$TCR = \frac{Err^b}{Err} = \frac{N_S}{n_{G \to S} + n_{S \to G}} \tag{11}$$

Asymmetric Classification

In the case of **asymmetric classification**, erroneously blocking a legitimate message as spam is considered to be more expensive than letting a spam message pass through the filter (we used $\lambda = 9$). The *classification criterion* for labelling a message as spam is given in equation 12.

$$\frac{P(C = S|D)}{P(C = G|D)} > \lambda \tag{12}$$

The asymmetric accuracy and error rate of the classifier are given by equations 13, while the asymmetric TCR is given in equation 14.

$$WAcc = \frac{\lambda \cdot n_{G \to G} + n_{S \to S}}{\lambda \cdot N_G + N_S} \quad WErr = \frac{\lambda \cdot n_{G \to S} + n_{S \to G}}{\lambda \cdot N_G + N_S} \tag{13}$$

$$WTCR = \frac{WErr^b}{WErr} = \frac{N_S}{\lambda \cdot n_{G \to S} + n_{S \to G}} \tag{14}$$

2.3.2 Results

Before measuring the performance of the classifiers on the truncated e-mail data, we reproduced the original results published in (Medlock, 2006), for both symmetric and asymmetric classification. This provided us with the baseline measurements on one hand, and also served as a correctness validation step on the other hand. The results are summarized in table 1 below.

Type of results	Classifier	GEN recall	SPAM recall	accuracy	TCR
Symmetric ($\lambda = 1$)	MNB	0.9111	0.9774	0.9451	8.87
	SVM	0.9389	0.9824	0.9613	12.56
	BLR	0.9270	0.9811	0.9548	10.77
	ILM Unigram	1.0000	0.9598	**0.9793**	**23.56**
	ILM Bigram	0.9973	0.9372	0.9664	14.50
Asymmetric ($\lambda = 9$)	MNB	0.9456	0.8092	0.9312	1.44
	SVM	0.9787	0.8469	0.9110	5.46
	BLR	0.9814	0.7628	0.9802	0.25
	ILM Unigram	1.0000	0.8494	**0.9841**	**6.28**
	ILM Bigram	0.9986	0.8067	0.9785	4.62

Table 1: *GenSpam Test* set results (the best results for each dataset are in bold)

The numbers for symmetric data are comparable to those reported by Medlock: the MNB classifier has the worst performance, the ILM models have the best performance, and SVM and BLR are situated in the middle. However, there is one notable difference: while (Medlock, 2006) reports the **ILM Bigram** model to have the best performance in terms of accuracy, with the unigram model slightly behind it, our experiments indicate the opposite: that the **ILM Unigram** model is slightly better than the bigram one. One possible explanation for this discrepancy is the minor difference between the version of the GenSpam dataset that I used and the version of the GenSpam dataset cited in the original paper, which was mentioned in section 2.2.1. In addition to the metrics reported by Medlock, we also computed the **Total Cost Ratio**, which proves that all five filters perform better than the no-filter null hypothesis.

The asymmetric evaluation scheme that we used is different from the one used in the original paper, which allowed no more than one in every 200 genuine messages to be misclassified. The reason we decided to use the equations introduced in (Androutsopoulos et al., 2000), and described in section 2.3.1, is because they allowed us to study the effect of e-mail truncation on the recall of both genuine and spam e-mail messages *independently* of each other.

After computing the baseline values for the full-length test e-mails, we repeated the measurements for gradually truncated versions, starting from a maximum size of 300 tokens, and decreasing the length with a step of 20 tokens. Additionally, we measured the performance for 10 and 5 tokens, in order to capture *borderline behaviour*. As figures 6 and 7 indicate, the pattern of peformance degradation is the same for both symmetric and asymmetric metrics. All five classifiers considered are robust to test message truncation down to 60 or 80 tokens, below which there is a steep drop in recall, accuracy and TCR. These observations verify the empirical claim made in (Guzella and Caminhas, 2009).

The performance of the MNB classifier seems to go up as the length of the test messages falls below 100 tokens in one instance (the spam recall under the symmetric metric, in figure 6b). The classifier's tendency to label all short test e-mail as spam under symmetric classification can be explained by the disproportionately large number of spam e-mails in the training data, when compared to the number of genuine e-mails.

The ILM models consistently outperform the other classifiers under both symmetric and asymmetric classification, verifying the claims made in (Medlock, 2006). Furthermore, the ILM classifiers have also proven to be the most robust to test message truncation of the five classifiers. The explanation for their comparatively small performance loss lies in the interpolation process itself: as the body of the truncated messages becomes shorter and less in-

formative, the optimal interpolation weight is observed to decrease accordingly (by as much as 20% in the case of unigram models under symmetric evaluation), allowing a stronger signal from the subject line of the e-mail to compensate.

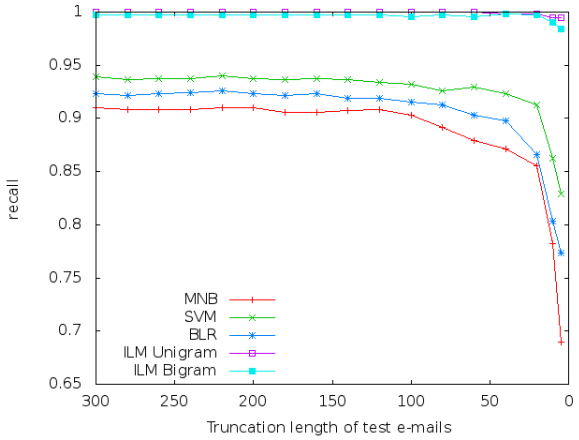
Type of results	Classifier	GEN recall	SPAM recall	accuracy	TCR
Symmetric ($\lambda = 1$)	MNB	0.6445	0.9084	0.7801	2.21
	SVM	0.0000	1.0000	0.5138	1.00
	BLR	0.6737	0.9322	0.8065	2.51
	ILM Unigram	0.9867	0.8883	**0.9361**	**7.61**
	ILM Bigram	0.9734	0.7641	0.8658	3.65
Asymmetric ($\lambda = 9$)	MNB	0.9469	0.6624	0.9170	1.19
	SVM	0.0000	1.0000	0.1051	0.11
	BLR	0.8766	0.7678	0.8652	0.73
	ILM Unigram	1.0000	0.8670	**0.9869**	**7.11**
	ILM Bigram	0.9946	0.5445	0.9473	1.88

Table 2: Results on the subject lines of the *GenSpam Test* set (the best results for each dataset are in bold)

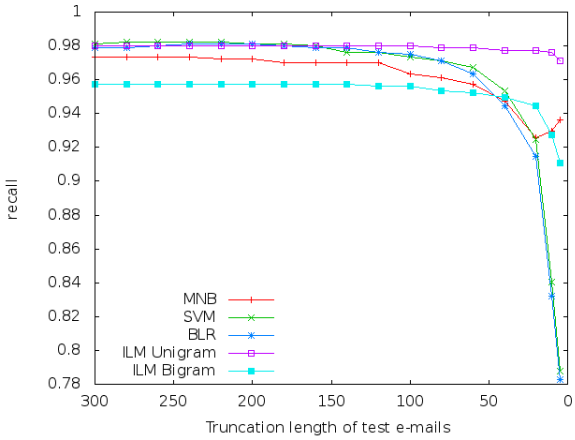
Following this observation, we also measured the classifier performance on the subject lines of the e-mails alone, yielding the results in table 2 below. Again, the ILM Unigram model demonstrated the best performance across the board, with the ILM Bigram model following in second place. The other models, however, do not perform competitively when classifying on the basis of the subject line alone, indicating that they are less effective when trained on low amounts of data.

To test this hypothesis, we ran the same set of experiments, but this time we truncated both the training e-mails, and the test e-mails. This experiment aimed to reveal the effectiveness of the four types of classifiers in detecting spam among short text messages. The results are summarized in figures 8 and 9.

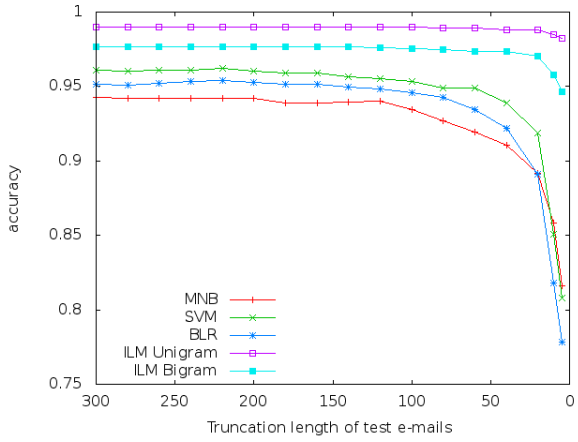
What we discovered was that, especially under symmetric classification, the ILM models still perform better than the other models in all metrics except spam recall (figure 8b). The performance degradation pattern under truncated training and test data is the same as the performance degradation pattern under truncated test data alone, with 100 tokens as the limit for truncation before the metrics start dropping. Although the 100-token truncation limit is also valid for asymmetric classification, the drop in performance is steeper under asymmetric metrics. These results verify the claims made in (Song et al., 2011) and (Thomas et al., 2011) regarding the inapplicability of classifcal spam filtering models to web services, which are poorer in text.



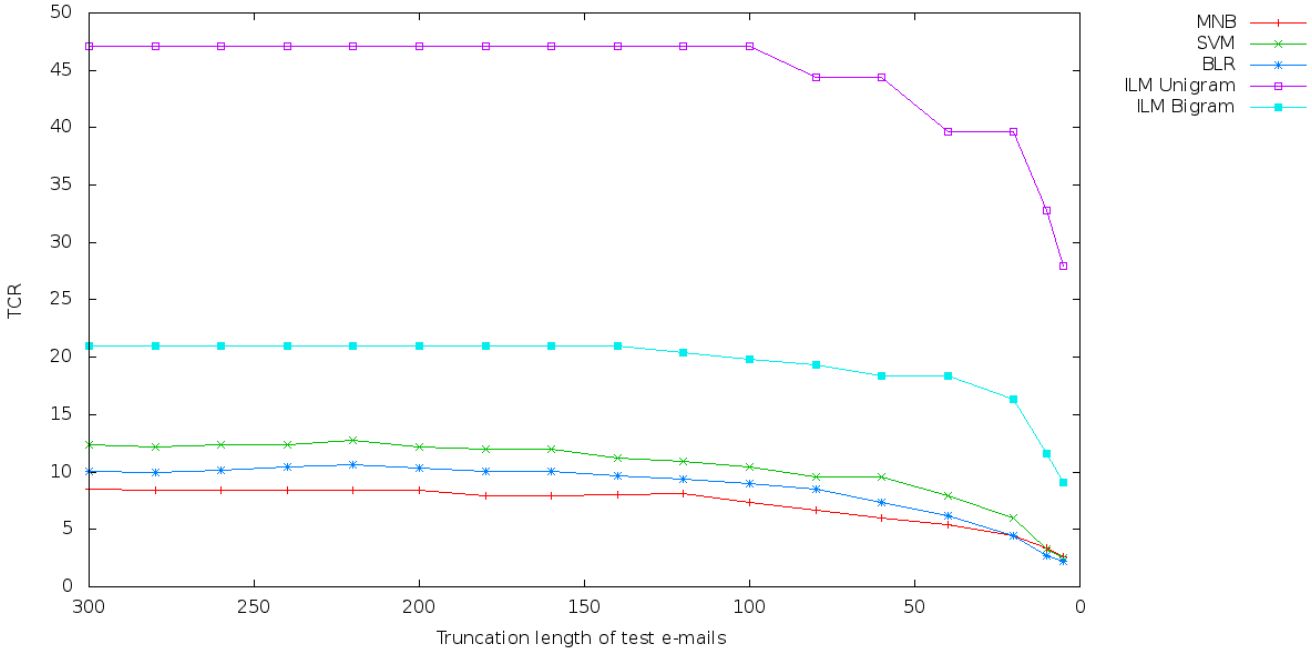
(a) Genuine e-mail recall.



(b) Spam e-mail recall.

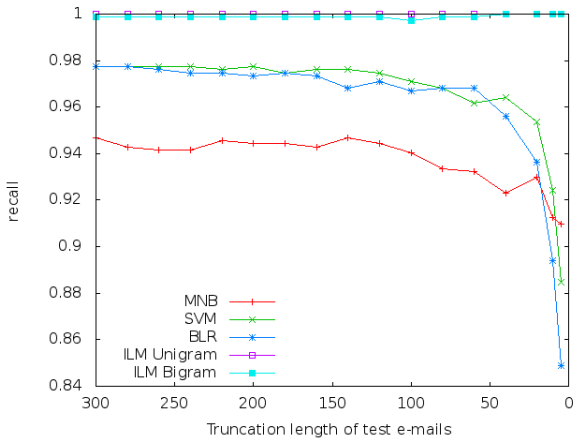


(c) Accuracy.

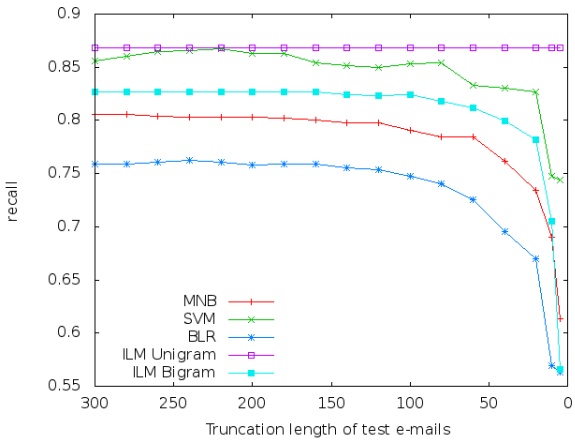


(d) Total Cost Ratio.

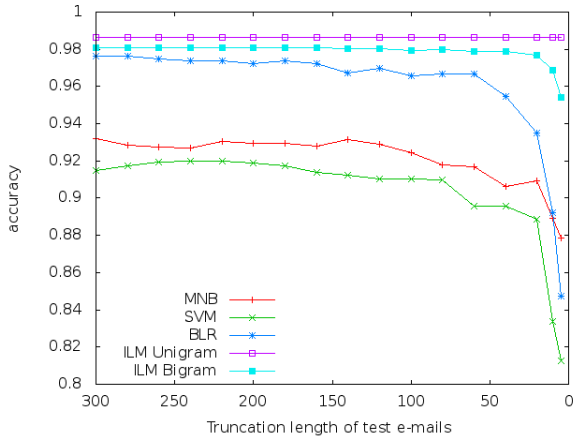
Figure 6: Performance degradation under symmetric classification, when gradually truncating the e-mails in the test data set alone.



(a) Genuine e-mail recall.

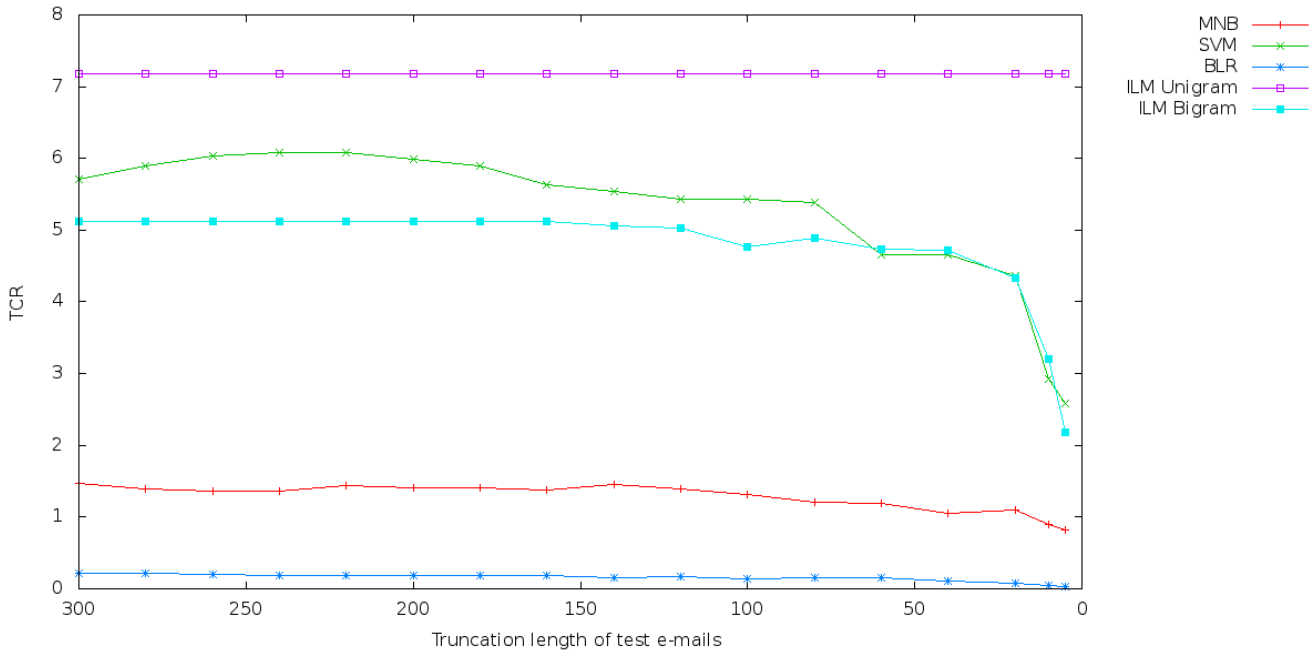


(b) Spam e-mail recall.



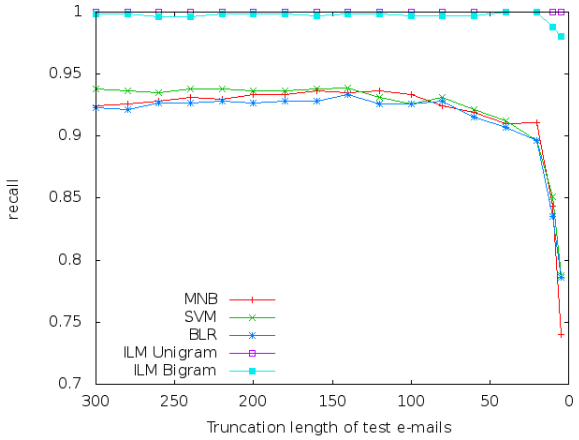
z

(c) Accuracy.

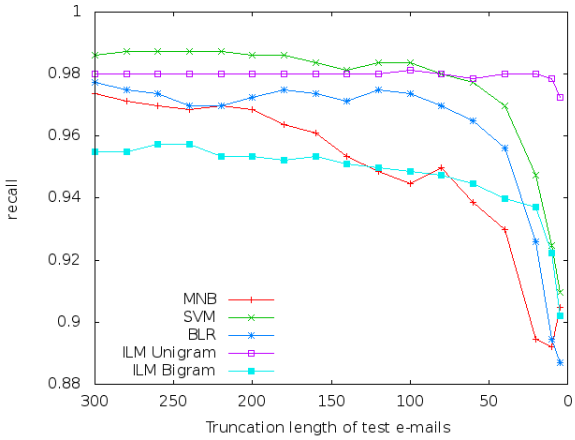


(d) Total Cost Ratio.

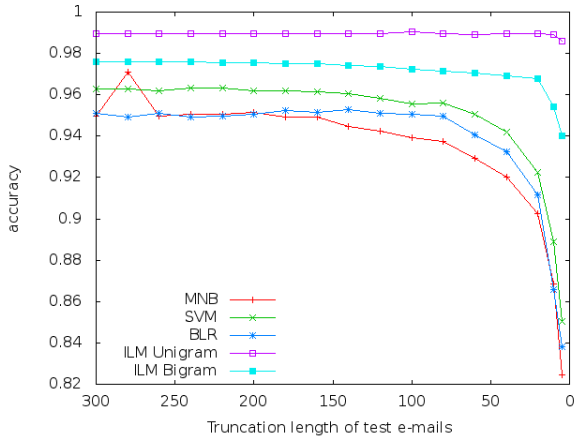
Figure 7: Performance degradation under asymmetric classification, when gradually truncating the e-mails in the test data set alone.



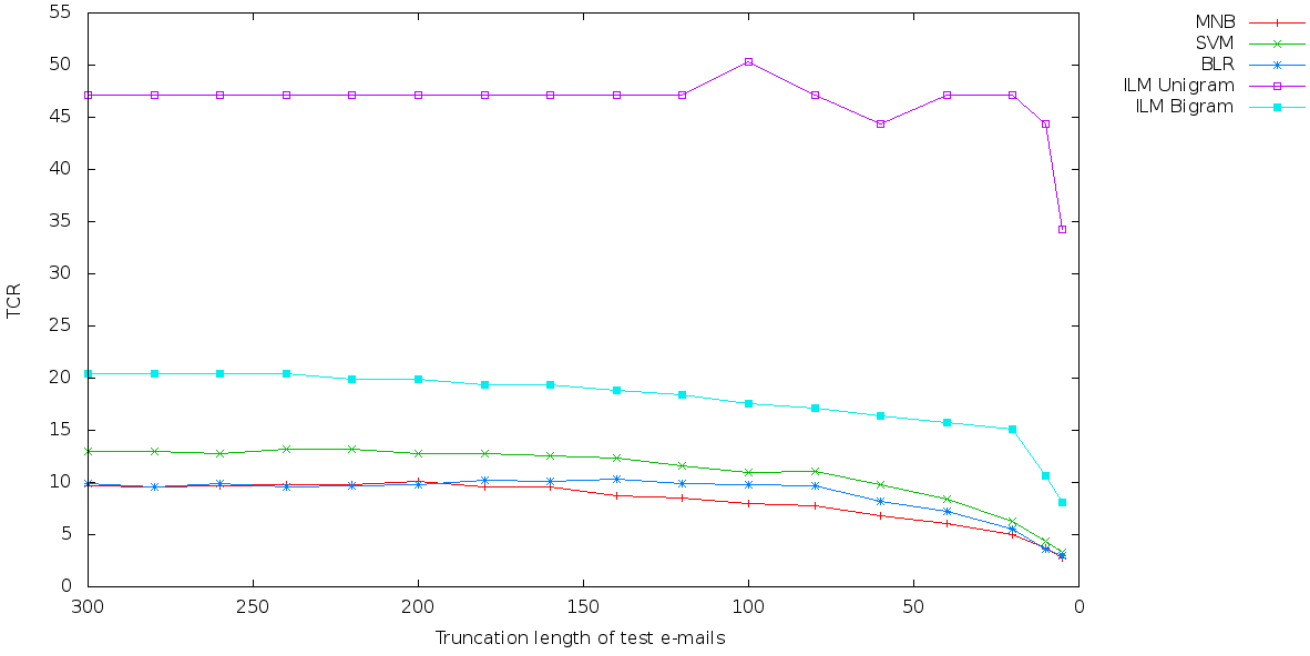
(a) Genuine e-mail recall.



(b) Spam e-mail recall.

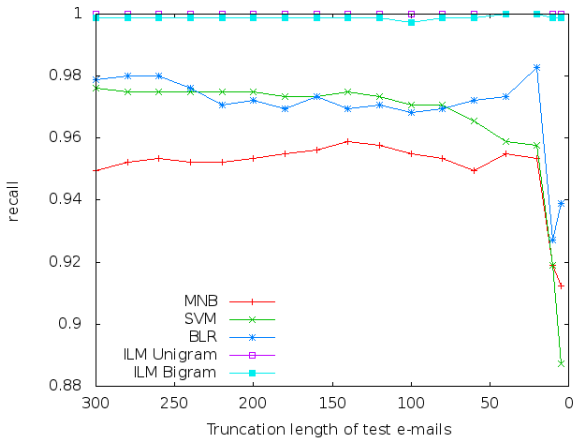


(c) Accuracy.

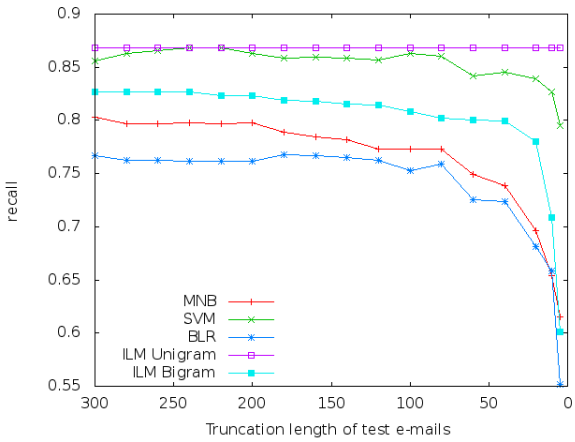


(d) Total Cost Ratio.

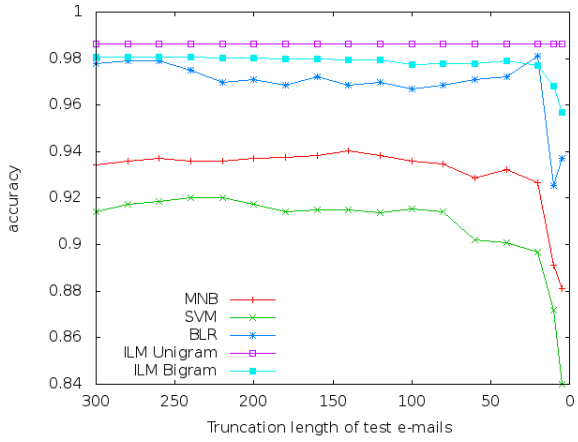
Figure 8: Performance degradation under symmetric classification, when gradually truncating the e-mails in both the training and the test data set.



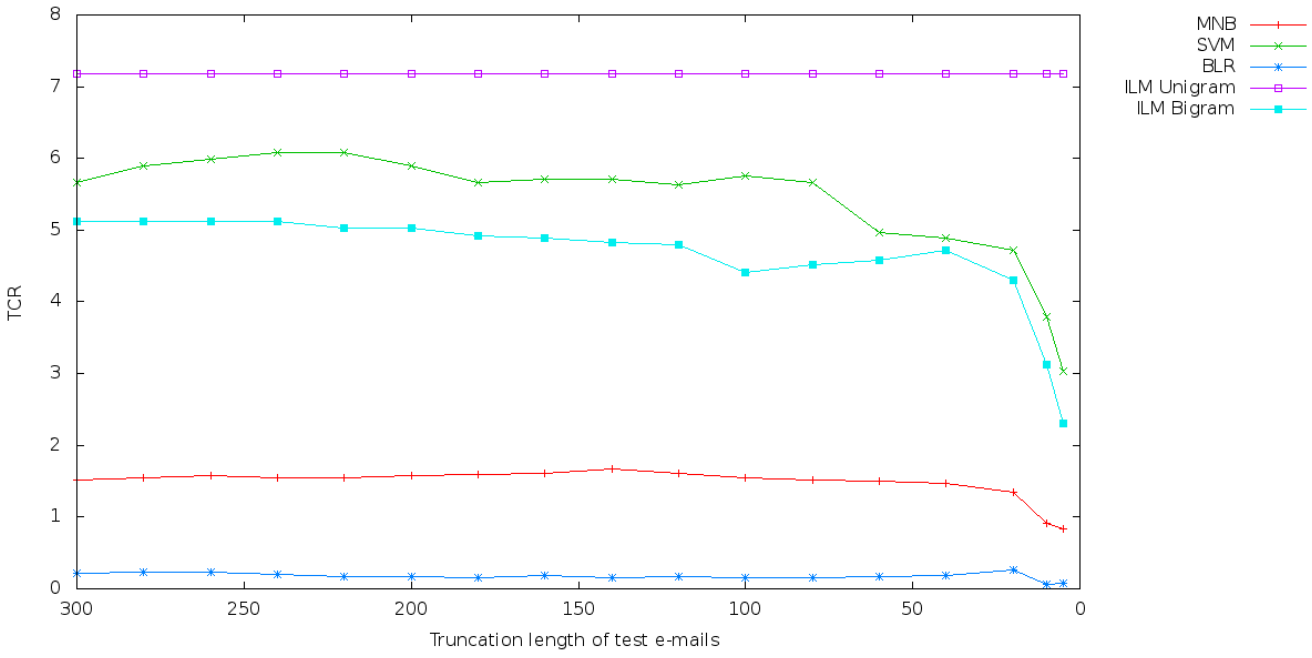
(a) Genuine e-mail recall.



(b) Spam e-mail recall.



(c) Accuracy.



(d) Total Cost Ratio.

Figure 9: Performance degradation under asymmetric classification, when gradually truncating the e-mails in both the training and the test data set.

3 Conclusions

In this report, we have presented the experimental methodology required to test (a) whether the empirical observation that humans only rely on the title of an e-mail and the first tokens of its body to differentiate between genuine and spam messages can be implemented in automated spam filters without loss of performance, and (b) how the performance of four traditional spam filters degrades when applied to short messages.

The experiments carried out on the gradually-truncated test set of e-mails in the GenSpam corpus revealed that three of the classical spam filters (MNB, SVM and BLR) were robust to decreasing test message length down to approximately 100 tokens, while the performance of the ILM models remained unaffected for test message lengths down to 20 tokens, and declined only slightly after that. These experiments *confirm* that the empirical claim made in (Guzella and Caminhas, 2009) is valid. They also show that ILM models are generally more robust on short test messages than other generative/discriminative models, owing to the added degrees of freedom gained through the interpolation weights.

The experiments carried out on the gradually-truncated training and test set of e-mails revealed that the performance of the four classifiers is also robust in the case of training text lengths down to approximately 100 tokens, beyond which we can see a rapid decline. As the amount of text present in current web services is often much smaller than 100 tokens (eg. 140 characters for a tweet, which based on an average word-length of 5 characters in the English language, can be approximated to 23 words with spaces in between), the experiments *confirm* that traditional text-based spam classifiers are not effective in these media.

Last but not least, our experiments verify that spam classifiers based on ILM Unigram and Bigram models outperform MNB, SVM and BLR classifiers across the board, as indicated in (Medlock, 2006).

References

Ion Androutsopoulos, John Koutsias, Konstantinos V Chandrinos, and Constantine D Spyropoulos. An experimental comparison of naive bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 160–167. ACM, 2000.

Ali Çıltık and Tunga Güngör. Time-efficient spam e-mail filtering using n-gram models. *Pattern Recognition Letters*, 29(1):19–33, 2008.

Harris Drucker, Donghui Wu, and Vladimir N Vapnik. Support vector machines for spam categorization. *Neural Networks, IEEE Transactions on*, 10(5):1048–1054, 1999.

Alexander Genkin, David D Lewis, and David Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304, 2007.

Thiago S Guzella and Walmir M Caminhas. A review of machine learning approaches to spam filtering. *Expert Systems with Applications*, 36(7):10206–10222, 2009.

Thorsten Joachims. Making large scale svm learning practical. 1999.

Andrew McCallum, Kamal Nigam, et al. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer, 1998.

Ben Medlock. An adaptive, semi-structured language model approach to spam filtering on a new corpus. In *Third Conference on Email and Anti-Spam*, 2006.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss,

V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Monica Rogati and Yiming Yang. High-performing feature selection for text classification. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 659–661. ACM, 2002.

Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 workshop*, volume 62, pages 98–105, 1998.

Jonghyuk Song, Sangho Lee, and Jong Kim. Spam filtering in twitter using sender-receiver relationship. In *Recent Advances in Intrusion Detection*, pages 301–317. Springer, 2011.

Kurt Thomas, Chris Grier, Justin Ma, Vern Paxson, and Dawn Song. Design and evaluation of a real-time url spam filtering service. In *Security and Privacy (SP), 2011 IEEE Symposium on*, pages 447–462. IEEE, 2011.