Simple Language Models for Spam Detection

Egidio Terra
Faculty of Informatics
PUC/RS - Brazil

Abstract

For this year's Spam track we used classifiers based on language models. These models are used to compute the log-likelihood for each individual message and then classify them as either ham or spam. Different data sets were used to train these language models. Our approach is simple, we initially create simple unigram language models and smooth the probabilities of unseen tokens by means of the expected likelihood estimator with a small discount probability tuned in a training corpus.

1 Introduction

The statistical approaches for spam filtering are often Bayesian and assume a multivariate Bernoulli distribution for tokens based on the number of messages they appear[9, 6, 1, 4, 8]. Each word in each corpus, ham and spam, is assigned a conditional probability for each given class, i.e. $P(w|C = ham)$ and $P(w|C = spam)$. For a new message M, the goal is to compute $P(C = ham|M)$ and $P(C = spam|M)$, which will then be used to decide to what class the message belongs. Robinson [8] proposes the use of the χ^2 distribution confidence intervals to decide if the message is spam or ham. In the most common approach — the naive bayes — every word is considered independent from each other, i.e. for a given class C, $P(C|M) = \prod_{w \in M} P(C|w)$. To convert $P(w|C)$ into the desired $P(C|w)$, the bayes' formula is used: $P(C|w) = [P(w|C)P(C)]/P(w)$.

Language Models on the other hand normally take a multinomial approach within a single distribution for all tokens, regardless of how many messages they occur. One benefit of the multinomial approach is the number of available discounting and smoothing methods to handle unseen tokens, whereas some Bayesian approaches often handle this situation heuristically (e.g. unseen tokens are assigned a constant probability regardless of other tokens' probabilities).

Our submissions to this year's TREC are based on language models. While spam classification can be seen as a text categorization with only two classes, it has some distinct characteristics from normal text, among others: a) e-mail massages have some structure; b) spam is written to look like a legit message; c) the arrival order of messages maybe important; d) spam messages mutate to avoid filters. It is our intention to evaluate the language models in this context since it has yielded good results in other text classification tasks, outperforming state-of-art classifiers such as SVMs in many of them [7].

Section 2 gives a brief introduction to the language models used in our submissions. The following section, 3, describes the training data we used for our models and section 4 presents our

results and discussion.

2 Simple Language Models

Language models are generative with some order approximation to the language [10]. We start described the general form of the model:

$$P(M) = P(w_1) \prod_{i=2..n} P(w_i|w_1..w_i - 1) \quad (1)$$

where $w_1..w_i - 1$ is called the history. The effect of history is more important in nearby neighbors, i.e. the outcome of the current word is more influenced by recently occurring ones. This is explored in the high-order approximations and it is equivalent to the Markov assumptions, where older history is not taken into account

$$P(M) = \prod_{i=1..n} P(w_i|w_{i-k+1}..w_{i-1}) \quad (2)$$

for a k-order approximation.

A first-order word approximation (unigram model) is simply

$$P(M) = \prod_{i=1..n} P(w_i) \quad (3)$$

where $|M| = n$ and $P(w_i)$ are observed from actual text. Equation 3 is also used to described the zero-order approximation. Shannon makes the distinction between zero and first based only on the estimates of $P(w_i)$. In the zero-order, the estimates are sampled from an uniform distribution.

The model is called generative since we can, by sampling with replacement, generate sequences that will have similar distribution to the model. When used in text classification, a model can be built for each class and the most likely class is chosen [7]. In the case of a dichotomous test, such as the one found in spam filtering, there are other alternatives. Dunning proposes the use of log-likelihood test to validate the hypothesis that the classes are distinguishable [3]. For that he explored the fact that a log-likelihood is asymptotically χ^2 distributed and as such we can use confidence intervals from the distribution to decide if a message is spam or not.

Another important aspect in language models is the sparseness problem. Specially in higher-order approximations, the number of potential combinations is high. For a vocabulary V, containing $v = |V|$ distinct words, a k-order approximation will have v^k combinations. Even for very large corpora the number of possible combinations is greater than then number of seen ones, and even for modest values of k because the vocabulary tend to be large. For this problem several approaches exists [2, 5], the so-called smoothing and discounting techniques. The basic idea is to reserve some probability mass for the unseen events in the training data. For simplicity, we have used the expected likelihood estimate:

$$P_{ele}(w_i) = \frac{f(w_i) + \lambda}{N + B\lambda} \quad (4)$$

where N is the size of the corpus (tokens) and B is the number of unique words (types). The frequency of a word is given by $f(w_i)$. The value of λ can be trained and if its value is set to 1 then it is equivalent to the Laplace's law [5].

3 Training

For each class, ham and spam, we created distinct unigram language models. For every new message the likelihood is computed in both ham

Corpus	% Ham Miscl.	% Spam Miscl.	1-ROCA%
Full	3.41	5.10	2.08
mrx	4.93	2.26	1.91
sb	1.44	24.13	1.41
tm	1.77	27.34	2.92

Table 1: Results for run pucrs0

Corpus	% Ham Miscl.	% Spam Miscl.	1-ROCA%
Full	3.57	5.33	2.18
mrx	5.97	2.72	3.08
sb	1.03	17.29	1.58
tm	1.80	27.87	2.71

Table 2: Results for run pucrs1

and spam language models and the log-likelihood odds is used to classify the message. In two submissions, pucrs0 and pucrs1, we use the result of the equation 5 to classify the message. If the outcome is higher than a threshold the message is considered spam, otherwise it is ham.

$$\begin{aligned} \log LL &= \log \frac{P_s(M)}{P_h(M)} \\ &= \sum_{w \in M} \log P_s(w) - \log P_h(w) \end{aligned} \tag{5}$$

where $P_s(w)$ is the probability of generating the word w according to the spam language model and $P_h(w)$ the analogous for the ham language model.

We use a train-on-everything approach, for all incoming messages we update the corresponding language model. In the first run, labeled pucrs0, the token frequencies from the incoming messages are accumulated to those originated from the training data set, in this case the spamassassin corpus. The same approach was used in the second run, labeled pucrs1, but, instead of the spamassassin corpus, we used messages from a spam archive and for ham messages we used documents extracted from the AQUAINT corpus.

In the third run, pucrs2, the token frequencies from the incoming messages are kept separate from those originated from the training

Corpus	% Ham Miscl.	% Spam Miscl.	1-ROCA%
Full	3.35	5.00	1.97
mrx	6.07	2.77	3.45
sb	2.47	40.90	5.44
tm	2.17	20.73	3.69

Table 3: Results for run pucrs2

data, once again the spamassassin corpus. As result, four distinct models are kept: incoming ham (P_h), incoming spam (P_s), background ham (P_{hb}) and background spam (P_{sb}). The two pairs of language models are then linearly interpolated, as depicted in equation 6. The mixture parameters are not fixed and, after a certain number of messages are collected, the weights assigned to background probabilities are reduced linearly. The initial weights λ_s for spam language models and λ_h for the ham language models are trained on spamassassin corpus. These weights are reduced as the number of incoming messages increases.

$$\begin{aligned} \log LL &= \log \frac{\lambda_s P_s(M) + (1-\lambda_s) P_{sb}(M)}{\lambda_h P_h(M) + (1-\lambda_h) P_{hb}(M)} \\ &= \sum_{w \in M} \log[\lambda_s P_s(M) + (1-\lambda_s) P_{sb}(M)] \\ &\quad - \log[\lambda_h P_h(M) + (1-\lambda_h) P_{hb}(M)] \end{aligned} \tag{6}$$

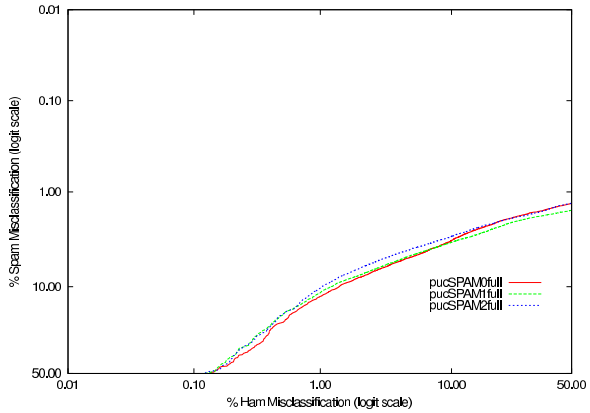

Figure 1: ROC for the Full corpus

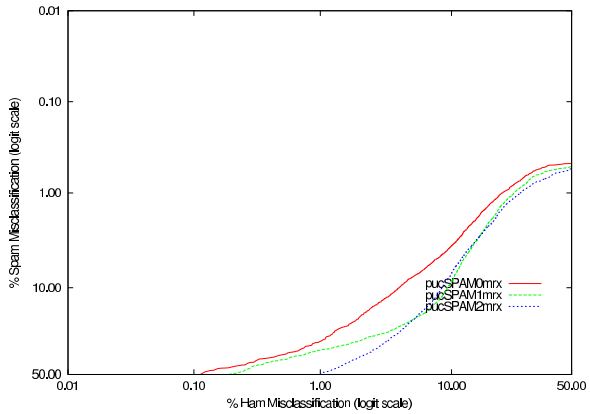

Figure 2: ROC for the mrx corpus

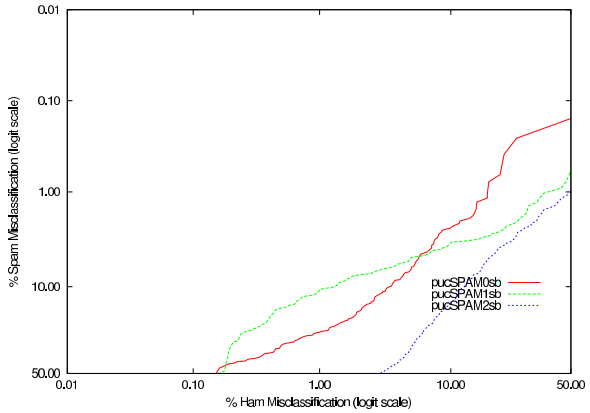

Figure 3: ROC for the sb corpus

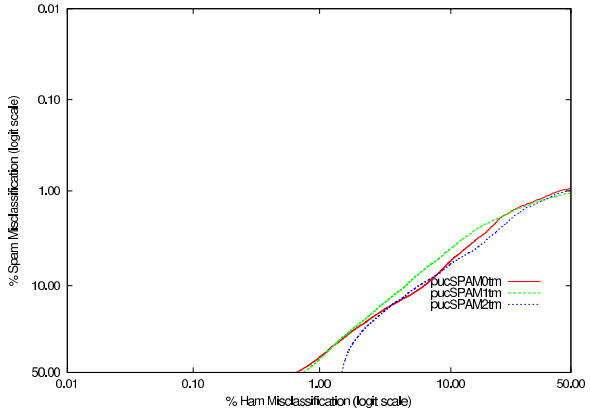

Figure 4: ROC for the tm corpus

To tune the decision process we use a threshold for the likelihood odds and actively change its value based on the misclassifications performed so far. Since ham misclassifications is considered to be worse than spam misclassification, the threshold tuning is more aggressive when the former occur. In our case, this tuning is particularly necessary when the classifiers have not being exposed to enough data to create the model. In spam detection this can be the case when adapting the filter to a new user that does not have enough training data. This tuning process has an effect on the number of messages correctly or incorrectly classified but does not make a difference on the area under the ROC curve.

4 Results and Discussion

For our run pucrs0, the results are shown in the table 1 for the several corpora used. Table 2 presents the results for pucrs1 and table 3 contain the results for pucrs2. The overall misclassification rates are high and on two corpora, *sb* and *tm*, the spam misclassification is very high. These two corpora have number of spam messages small compared to the total number of messages: only around 11% of messages on *sb* and *tm* corpora is spam. In these two corpora the classification thresholds are biased to ham scores, i.e., the threshold to classify a message as spam is high. The two other corpora are more balanced, only 18% of messages in mrx corpus are

ham and 42% of messages in the full corpus is ham. This poor performance indicates that the current tweaking of the threshold is not robust for corpus with unbalanced number of messages on the two classes. The area under the ROC is a more robust measure to determine the usefulness of the scores assigned by the classifiers.

The learning rates are also influenced by the active threshold tuning. Unfortunately, setting the threshold is a quick change but running the filters on the private data is not yet possible if ever.

The results for pucrs2 are worse than the other two runs, as shown in the ROC graphs in Figures 1,2,3, and 4. With the exception of the full corpus, pucrs2 performs statistically worse the pucrs0 and pucrs1 in terms of area under the ROC. Only in the full corpus the performance is similar. This suggests that either the mixture with a background model is ineffective or that the initial weight and decaying rate need to be trained for each corpus.

We have tried Robinson's chi-square Bayesian, adopted in many popular filters such as Bogofilter and Spamassassing, in our framework and the results are not better than the language models. This suggests that the high misclassification rates may be also due to the preprocessing of the messages. Our tokenization process is too simple, we use standard stopword list and all headers are ignored with the exception of the `to:`, `subject:` and `from:` entries. No special handling is made for attachments and multipart messages, all data is considered to be text. All words with less than three characters is discarded. Some regular patterns are used to split URL's and e-mail addresses.

Acknowledgments

We thank the University of Waterloo for letting us use their computer resources.

References

[1] I. Androutsopoulos, J. Koutsias, K.V. Chandrinos, G. Paliouras, and C.D. Spyropoulus. 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)*, 2000.

[2] S. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. Technical report, Harvard University, 1998.

[3] Ted Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1), 1993.

[4] Paul Graham. A plan for spam. http://www.paulgraham.com/spam.html.

[5] Christopher D. Manning and Hinrich Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[6] Patrick Pantel and Dekang Lin. Spamcop– a spam classification and organization program. In *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[7] Fuchun Peng, Dale Schuurmans, and Shaojun Wang. Language and task independent text categorization with simple language models. In *Proceedings of the 2003 Human Language Technology Conference of*

the North American Chapter of the Association for Computational Linguistics, 2003.

[8] Gary Robinson. A statistical approach to the spam problem. *Linux Journal*, (107), 2003.

[9] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk e-mail. In *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[10] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, July and October 1948.