A Language Model Approach to Spam Filtering

Ben Medlock
Cambridge University Computer Laboratory
William Gates Building
JJ Thomson Avenue
Cambridge, CB3 OFD
benmedlock@dunelm.org.uk

ABSTRACT

We present a classification model for semi-structured documents based on statistical language modelling theory which outperforms extant approaches to spam filtering on the *Ling-Spam* email corpus [1]. We also introduce two variants of a novel discounting technique for higher-order N-gram language models developed in the light of the spam filtering problem.

Categories and Subject Descriptors

I [**Computing Methodologies**]: Document and Text Processing; I.7 [**Document and Text Processing**]: Document Capture; I.7.5 [**Document Capture**]: Document analysis—*document classification, spam filtering*

General Terms

Algorithms, Performance, Theory

Keywords

backing off, generative classification model, interpolation scheme, language model, deleted interpolation estimation, discounting, distribution smoothing, linear interpolation, semi-structured document classification

1. INTRODUCTION

The problem of unsolicited email, or *spam*, is of serious and growing concern in our digital age[1]. This study is concerned with a method of addressing the problem by *filtering* email to remove spam upon receipt, a technique that has recently begun to receive much attention.

2. STATISTICAL SPAM FILTERING

The spam filtering problem can be viewed as an instance of a *text classification problem* [12], on the basis that most email contains some form of identifiable textual content. In fact, the structure of email is richer than that of flat text, with meta-level features such as the fields found in MIME compliant messages (Subject, To, From, Content-Type etc.). In light of this, we present the problem as one of *semi-structured* document classification.

Some of the first published work on statistical spam filtering was carried out by Sahami et al. [10] using a multi-variate Bernoulli NB model; however the training and test sets are small (less than 2000 total messages), and not publicly available, thus rendering the experiments non-repeatable.

Androutsopoulos et al. [1] present the first results for spam filtering on the *LingSpam* corpus, comparing a multinomial NB classifier with a kNN variant. Their implementation of NB outperformed kNN in the reported experiments. Carreras and Marquez [2] build on this work, publishing improved results on the *LingSpam* corpus using boosting decision trees with the *AdaBoost* algorithm.

Drucker, Wu and Vapnik [4] publish results comparing the use of SVM's with various other discriminative classification techniques on the spam filtering problem, with binary-featured SVM's and boosting decision trees performing best overall. Unfortunately the test sets they used are not publicly available.

3. CLASSIFICATION MODEL

3.1 Introduction

The classification model we present can be seen as an extension of extant generative text classification models such as the Naïve Bayesian models presented by McCallum and Nigam [8] and Grobelnik and Mladenic [5]. However, it differs in two significant ways. Firstly it provides a method of capturing local phrasal dependency by incorporating smoothed higher-order N-gram language models[2]. This avoids undesirable violations of the attribute independence assumption when attempting to incorporate higher-order N-grams directly into a naïve Bayesian model. Secondly, it provides a motivated framework for taking advantage of the semi-structured nature of documents such as email messages through the use of multiple language model component interpolation and optimisation techniques.

[1]See research by *MessageLabs* (www.messagelabs.co.uk) and *Ferris* (www.ferris.com).

[2]We use N-grams for efficiency and simplicity; however, the classification model does not preclude the use of more advanced LM's with more effective syntactic/semantic analysis.

Although we only consider application of the proposed classification model to the 2-class spam filtering problem, it is also applicable to the more general N-class semi-structured document classification problem.

3.2 Formal Classification Model

We use the following terminology and definitions:

- *Document*: a domain-specific, discrete item of information (i.e. a single news-story, email message, etc.).
- *Word*: an atomic unit within a document (note that this is broader than the linguistic notion; for instance, a 'smiley' is considered to be a word).
- *Field*: a (possibly recursive) meta-level structure consisting of a label, and a finite number of words (and sub-fields).
- *Class*: a predefined (possibly infinite) set of documents, usually homogeneous with respect to a specified aspect of the information they contain (i.e. topic).
- *Classification*: a mapping of a given document to a given class.

We also make the following assumptions:

1. We assume that a document belongs to exactly one class, though the model can quite easily be extended to account for documents belonging to arbitrarily many classes.
2. We assume that a document is structured into exactly L fields, and that each field is non-recursive, consisting of a finite sequence of words.
3. We assume that the fields within a document are *independent* of each other.
4. We assume that classification is carried out within a single domain, and within that domain, all documents have the same structure.

Given a set of documents $\{D_1, \ldots, D_M\}$ and a set of classes $\{C_1, \ldots, C_N\}$, we seek to discover a set of classifications of the type $D_i \to C_j$ for $i = 1 \ldots M$ where j ranges from $1 \ldots N$ (given assumption 1).

In the classic generative classification framework, the *posterior probability* of each class is estimated for the document in question, and the decision rule formulated from a direct comparison of these probabilities. We adopt this approach, resulting in the following formulation for the decision rule:

$$Decide(D_i \to C_j) \quad \text{where } j = \arg\max_k [P(C_k|D_i)] \tag{1}$$

Under assumptions 2, 3 and 4, we define the posterior probability of a class in terms of a *weighted linear interpolation* of the posterior probabilities of the L fields making up documents in the given domain:

$$P(C_j|D_i) = \sum_{f=1}^{L} \lambda_f \left[P(F_f^{C_j}|F_f^{D_i}) \right] \tag{2}$$

where

F_f^C refers to field f in class C

F_f^D refers to field f in document D

λ_f is the weight given to field f

To maintain the probability model, the sum-to-one property should hold of the weights: $\sum_{f=1}^{L} \lambda_f = 1$. An *interpolation scheme* is used to find the optimal values for the λ's. This is discussed later.

The field-dependent posterior probability is expanded using *Bayes Rule*:

$$P(C_j|D_i) = \sum_{f=1}^{L} \lambda_f \left[\frac{P(F_f^{C_j}) \bullet P(F_f^{D_i}|F_f^{C_j})}{P(F_f^{D_i})} \right] \tag{3}$$

$P(F_f^{C_j})$ is the *prior probability* for field F_f of class C_j. Under the structure uniformity assumption (4) we take all field priors within a given class to be equal to the class prior, i.e. $P(C_j)$.

The denominator, $P(F_f^{D_i})$, is constant with respect to class and thus often ignored in Bayesian classification models; however, the field-level linear interpolation in our model requires true probabilities; thus we retain the denominator. Dividing by $P(F_f^{D_i})$ can also be seen as normalising for unequal field lengths, i.e. scaling the class-conditional probability of a field (dependent on the length of the field by virtue of the fact that probabilistic intersection equates to multiplication) by a value constant with respect to class but multiplicatively proportional to the length of the field.

$P(F_f^{D_i})$ can be expanded to

$$\sum_{k=1}^{N} P(F_f^{C_k}) \bullet P(F_f^{D_i}|F_f^{C_k})$$

which is simply the sum over all classes of the prior times the class-conditional probability for the given field.

$P(F_f^{D_i}|F_f^{C_j})$ is the *language model probability* of field f in document D_i given class C_j. In other words, it is the likelihood that the LM chosen to model field f of class C_j generated the sequence of words composing field f of document D_i (under assumption 2), as represented by the following formula:

$$P(F_f^C|F_f^D) = P(w_{1F_f^D}, \ldots, w_{KF_f^D}|LM_{F_f^C}) \tag{4}$$

The classification model presented does not restrict the type of LM used; however, for our experiments we use *N-gram*

LM's. The N-gram model is based on the assumption that the existence of a word at a given position in a sequence is dependent only on the previous $N-1$ words. Thus the N-gram LM probability for a K-length word sequence can be defined (with allowances for the initial boundary cases) as:

$$P_N(w_1, \ldots, w_K) = \prod_{i=1}^{K} P(w_i | w_{i-N+1}, \ldots, w_{i-1}) \quad (5)$$

This formula is then specialised for $N = 1, 2, 3 \ldots$. We discuss LM construction in the next section.

3.3 LM Construction

In theory, the generation of N-gram statistics from normalised frequency counts is relatively simple. However, the issue of *data sparsity* (especially in higher-order N-grams) raises a number of complications. There is a good deal of literature on such matters, including seminal works by Katz [7], Jelinek and Mercer [6] and more recently Chen and Goodman [3], discussing techniques such as *discounting* and *backing-off* to handle sparsity. We build on this work by introducing two variants of what is to our knowledge a new discounting function for N-grams, based on analysis of the use of LM's for the classification problem within the proposed framework.

We adopt the basic formalisation for discounting and backing-off introduced by [7]. In the bigram case, the formula is as follows:

$$P(w_j|w_i) = \begin{cases} d(freq(w_i, w_j)) \frac{freq(w_i, w_j)}{freq(w_i)} & \text{if } freq(w_i, w_j) > C \\ \alpha(w_i) P(w_J) & otherwise \end{cases} \quad (6)$$

where

$d(r)$ is the discounting function
$\alpha(w)$ is the back-off weight
C is the N-gram cut-off point

For higher-order N-grams the same principles apply, forming a *back-off chain* from higher to lower-order models. The N-gram cut-off point, C, can be seen as the threshold below which we consider the number of occurrences too low to draw robust statistics from. In some cases, we may want to raise the value of C, as low-frequency higher-order N-gram entries may be less reliable than their lower-order counterparts. The discounting function, $d(r)$ is used to remove some of the probability mass from those events that have been observed in the training data, thus making it available to unobserved events. The discounted probability mass is then distributed over lower-order distributions with the back-off weight insuring conformance to the probability model, i.e. $\alpha(w_i) = 1 - \sum \hat{P}(*|w_i)$ where $\hat{P}$ is the *discounted* bigram probability. A small probability must also be assigned to events that remain unobserved at the end of the back-off chain, i.e. unigram entries that have not been seen at all in the training data. We can use this to model the likelihood of encountering unknown words, given a particular class of documents.

Various discounting schemes have been proposed in the literature; we implemented *linear* and *Good-Turing* for our experiments, as well as two variants of a new discounting function, which we will call *confidence discounting*, based on the intuition that the amount of probability mass discounted from a given N-gram entry should be inversely proportional to the *confidence* we have in that particular entry (within certain boundaries), represented by the absolute number of times the entry was observed in the training data. This idea can be formulated as follows:

$$d(r) = \frac{r}{R}(\omega - \phi) + \phi \quad (7)$$

where

$R =$ the number of distinct frequencies
$\phi =$ floor for lowest confidence
$\omega =$ ceiling for highest confidence

The value returned by the function ranges from ϕ to ω. R is an estimate of the highest level of confidence, chosen as the number of distinct N-gram frequencies because of its robustness to outliers. ϕ is chosen to represent the quantity of probability mass retained in the case of least confidence, and ω is chosen to represent the quantity of probability mass retained in the case of highest confidence (i.e. when the N-gram count approaches R). Note that when r exceeds R, an adjustment may need to be made to ensure the function does not return a value greater than one. The function is linear in the space r by $d(r)$.

A non-linear version can be formulated as follows:

$$d(r) = \frac{r(R-1)}{\frac{R}{\omega}(r-1) + \frac{1}{\phi}(R-r)} \quad (8)$$

In both cases, the values of the constants ϕ and ω can either be estimated autonomously from the data, or manually, based on empirical analysis. For our experiments we estimate ϕ from the training data, and use the LM-dependent value $1 - n_3/T$ for ω (where n_3 is the number of N-grams occurring 3 times, and T the total number of words encountered).

The assumption behind the non-linear form (8) is that confidence in a given N-gram should increase significantly after it occurs the first few times, and then continue to increase at a slower rate as it is encountered more and more often. The justification for this assumption is that as if an N-gram has been seen more than a few times (say around 5-10) it is likely to be more than just an erroneous or exceptional case, and our confidence in it should increase rapidly. However, once an N-gram has been seen many times already, seeing it a few more times should not imply such a significant increase in confidence.

Further details of the discounting schemes presented here can be found in [9], along with graphical depictions of their behaviour. In the results section, we consider the performance of the various discounting schemes in practice.

3.4 Interpolation

The purpose of an interpolation scheme is to optimise the weights of two or more interpolated components with respect to their performance under some objective function. The training data is used for this purpose, and a *deleted* interpolation scheme is often used to derive the most accurate estimates. Deleted interpolation estimation involves dividing the training data into a number of sections; one section is held back while the others are used to gather LM statistics. The objective function is then applied to the held-back section, and the optimal weights calculated. This is iterated over all sections, with each being held back in turn. Finally, the optimal weights for each section are averaged to derive the final estimation. More formally:

```
1. Divide training data into $N$ sections $\{S_1, \ldots, S_N\}$.
2. For each section, $i = 1 \ldots N$
        Build LM stats from sections $\{S_1, \ldots, S_N\} \setminus \{S_i\}$
        Estimate optimal weights, $w_i$, by using objective function on $S_i$
3. Derive final weights by averaging $\{w_1, \ldots, w_N\}$
```

In our case, a component is the probability function for a particular model or set of models given some training data. We choose the classification function itself (under the F_1 evaluation metric) as the objective function, which has the advantage of precisely reflecting the nature of the problem. Unfortunately though, the classification function is *non-differentiable*, and therefore optimality of the interpolation weights can only be guaranteed for a small number of components. In our experiments we only optimise two components (see below) so this is not an issue; however if a greater number of fields were available, a method of achieving near-optimality would have to be investigated.

4. DATA

The *LingSpam* corpus is divided into ten sections, which we used for ten-fold cross-validation, in line with previous experiments carried out by Androutsopoulos et al. [1] and others. Within each email message, only two fields are present: *Subject* and *Body.*

5. EXPERIMENTAL METHOD

The classification approach using the model we propose consists of two basic phases: Firstly the language models are constructed from the training data, and secondly the decision rule (equation 1) is used to classify the test data. The results we present were derived from ten-fold cross validation on the *LingSpam* data (each fold consisting of nine training sections and one test section). We experimented with different language model types and present results for unigram and smoothed bigram models using three different discounting schemes:

- GT = Good-Turing/Linear mix
- CN = Non-linear Confidence (8)
- CL = Linear Confidence (7)

The values of the interpolation (λ) weights were estimated using a deleted interpolation scheme, as described above (3.4), and the values for the LM parameters C and ϕ (for the confidence discounting schemes) were estimated from the training data in a similar manner. To save time, we manually estimated the value for the probability of unknown events (see 3.2) as 10^{-8} for unigram LM's and 10^{-7} for bigrams. Given time, optimising these values for specific LM configurations would have been preferable. Our experimental method, then, is as follows:

```
1. Choose LM type
2. For each cross validatory section, $S_i \in \{S_1, \ldots, S_{10}\}$
        Construct training set: $T_i = \{S_1, \ldots, S_{10}\} \setminus \{S_i\}$
        Estimate parameters $C_i$ and $\phi_i$ (confidence discounting only) using $T_i$
        Estimate interpolation weights $W_i$ using deleted interpolation scheme on $T_i$
        Construct language models $LM_i$ from $T_i$ using $C_i$ and $\phi_i$
        Classify $S_i$ using $LM_i$ and $W_i$, yielding results $R_i$
        Calculate evaluation measures on $R_i$, yielding $E_i$
3. Calculate evaluation measure average over $\{E_1, \ldots, E_{10}\}$
```

As an example of the interpolation scheme, Figure 1 (below) shows how the performance of the classifier varies over a single iteration (using section 1 as the unseen test section and section 2 as the weight estimation section) as the weights are varied. It is interesting to note that when the subject weight reaches 1.0 (i.e. there is no contribution from the message body LM's), the unigram classifier performs significantly better than the bigram classifiers, due to the diminutive amount of subject field data and resultant bigram sparsity. A similar pattern can be observed in the greater relative success of the bigram classifiers at recognising genuine email when relying only on the evidence from the message body LM's (subject weight of 0.0). This accords with the fact that there is a great deal more genuine email than spam in the corpus, and thus greater bigram coverage. This evidence suggests that given more data, higher-order N-gram classification yields increasingly accurate results, but conversely, if the data is very sparse, higher-order N-grams become increasingly unreliable. Intuitively, we would expect this to be the case.

6. EVALUATION AND RESULTS

6.1 Evaluation Measures

In our experiments we report precision (p), recall (r) and F_1 for both classes - SPAM and GEN. These measures are defined as follows:

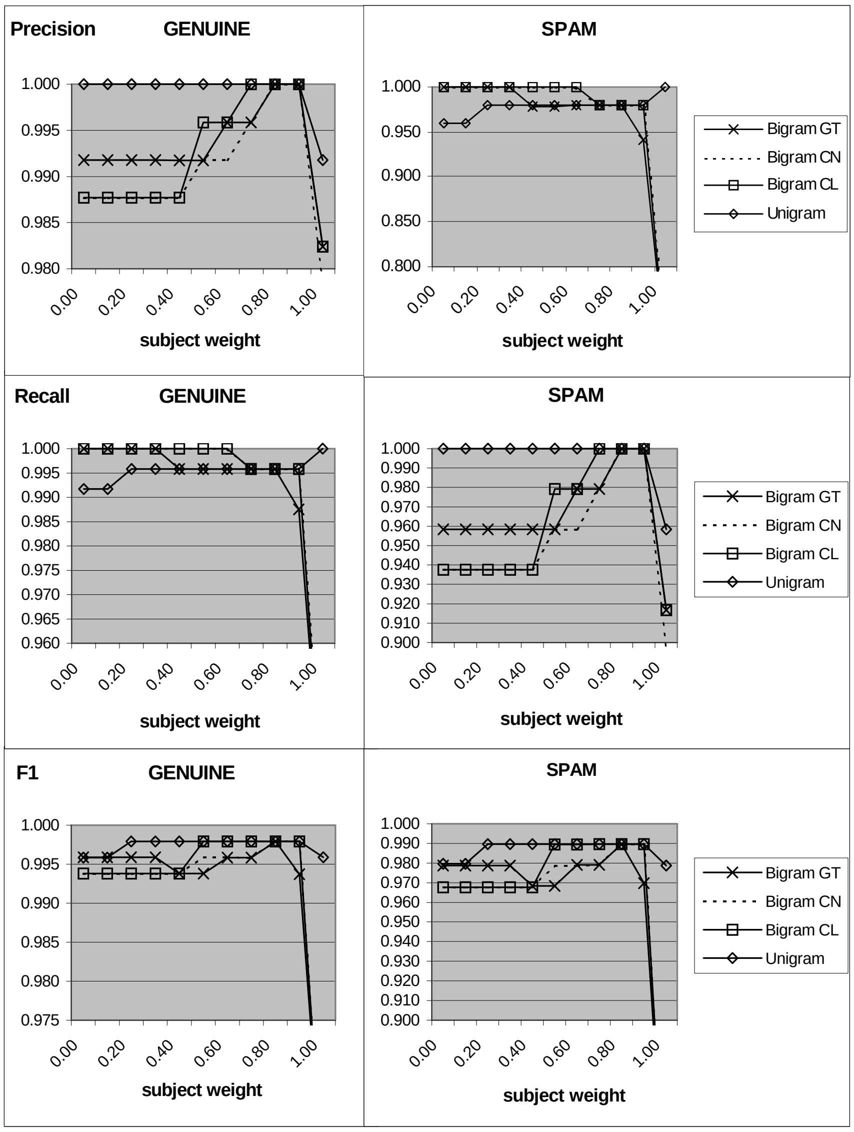

Figure 1: Interpolation scheme - 1 iteration

$$p(C) = \frac{num(D_C \rightarrow C)}{num(* \rightarrow C)} \quad r(C) = \frac{num(D_C \rightarrow C)}{num(D_C)}$$

$$\mathrm{F}_1(C) = \frac{2 \times p(C) \times r(C)}{p(C) + r(C)}$$

where

$$num(D_X \rightarrow Y) = \text{num class } X \text{ docs classified as } Y$$
$$num(D_X) = \text{total number of class } X \text{ docs}$$

Though not strictly necessary in the two-class case, we report both precision and recall for consistency with the extant literature. We do not use the *WAcc* (weighted accuracy) measure [1] because it involves an arbitrary assessment of the cost of misclassification and can only be meaningfully compared to an experimentally specific baseline [9].

6.2 Results

For purposes of comparison, we present results on the *LingSpam* corpus from four other classifiers presented in the literature:

- **NB.** We include the best results reported by Androutsopoulos et al. [1] for the Naïve Bayesian approach, using a lemmatized version of the *LingSpam* corpus and the *mutual information* (MI) metric for feature selection. They find NB to perform optimally in this case with a feature set of around 100 elements.
- **k-NN variant.** From the same paper, we include the best reported results for a variant of the k-nearest neighbour algorithm. As for NB, they perform feature selection based on the MI metric, and achieve optimal results with a smaller feature set of 50 elements.
- **Stacking.** This approach combines NB and k-NN in a

Table 1: Comparative results on *LingSpam* corpus.

	GENUINE			SPAM		
Classifier	Recall	Precision	F1	Recall	Precision	F1
NB	-	-	-	82.35	**99.02**	89.92
kNN	-	-	-	88.60	97.40	92.79
Stacking	-	-	-	91.70	96.50	93.93
TreeBoost	-	-	-	97.92	98.33	98.12
LM (unigram)	99.09	99.51	99.29	97.55	95.51	96.52
LM (bigram GT)	99.65	**99.71**	99.68	**98.53**	98.27	98.40
LM (bigram CN)	99.77	99.67	99.72	98.35	98.84	98.59
LM (bigram CL)	**99.78**	99.67	**99.73**	98.35	98.91	**98.63**

stack of classifiers. Sakkis et al. [11] experiment with various configurations. We include the best reported results from their paper.

- **Boosting Trees.** We include the best results reported by Carreras and Marquez [2] using several variants of the *AdaBoost* algorithm, based on learning and combining *weak rules* in a decision tree structure. They experiment with various tree depths using up to 2500 rules, and report enhanced performance comparative to previous work.

Table 1 displays results obtained using our classifier, alongside those previously published using the above classifiers. The results indicate a clear improvement in performance when the bigram LM classifier is used, with the various discounting strategies performing similarly well.

In addition to the results above, we experimented with trigram LM's, but observed a decrease in performance over bigrams. We strongly suspect that given the relatively small amount of training data and the generally 'noisy' nature of email text (especially in the SPAM case), >2-gram LM's do not generalise well enough to be effective.

7. CONCLUSIONS

We have presented a classification model for semi-structured documents, using motivated smoothing and interpolation of higher-order N-gram language models, which outperforms extant spam filtering techniques on the *LingSpam* corpus. We note that the *LingSpam* corpus is quite small and homogeneous in nature, and that a larger, more varied corpus would more effectively represent general email usage. [9] contains details of such a corpus (*GenSpam*), and a procedure used to protect donor privacy by *anonymising* the data.

Although the classification model we have presented can be seen as an extension of classic generative strategies such as multinomial NB, it is not in fact purely generative, rather it is a *combined model.* In the generative tradition, the decision rule is based on posterior probability modelling and comparison, but the interpolation weights are optimised with respect to the discriminative classification function. There would in fact be a number of advantages gained by using a generative objective function for interpolation such as *perplexity* or *maximium mutual information* (MMI) as they are both differentiable. This would give us an efficient method of guaranteeing optimality with regard to component weights; however, our experiments suggest that neither perplexity nor MMI correlate well with the classification function, resulting in poor performance. This is certainly a future avenue of research though.

On a final note, a more thorough experimental analysis of the novel discounting schemes we present in this paper would be beneficial, as the results from this study are inconclusive in terms of their true performance on a variety of data.

8. ACKNOWLEDGMENTS

I would like to thank my supervisor Dr. Ted Briscoe for his support throughout this work. I would also like to thank Dr. Mark Gales and Dr. Thomas Hain for their valuable input. Finally, I would like to thank Dr. Warren Grieff for his thorough and patient assistance in the preparation of this document.

9. REFERENCES

[1] I. Androutsopoulos, G. Paliouras, V. Karkaletsis, G. Sakkis, C. Spyropoulos, and P. Stamatopoulos. Learning to filter spam email: A comparison of a naive bayesian and a memorybased approach. *Workshop on Machine Learning and Textual Information Access*, 4, 2000.

[2] X. Carreras and L. Marquez. Boosting trees for anti-spam email filtering. *Proceedings of RANLP2001*, pages 58–64, 2001.

[3] S. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. *Proceedings of the 34th Annual Meeting of the ACL*, 1996.

[4] H. Drucker, D. Wu, and V. Vapnik. Support vector machines for spam categorization. *IEEE Trans. On Neural Networks*, 10(5):1048–1054, 1999.

[5] M. Grobelnik and D. Mladenić. Efficient text categorization. 1998.

[6] F. Jelinek and R. Mercer. Interpolated estimation of markov source parameters from sparse data. *Proceedings of the Workshop on Pattern Recognition in Practice*, 1980.

[7] E. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Trans. ASSP*, 35(3), 1987.

[8] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. 1998.

[9] B. Medlock. A generative, adaptive language model approach to spam filtering. Master's thesis, Cambridge University Computer Laboratory, 2003.

[10] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A bayesian approach to filtering junk e-mail. 1998.

[11] G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. Spyropoulos, and P. Stamatopoulos. Stacking classifiers for anti-spam filtering of e-mail. *Proceedings of the 6th Conference on Empirical Methods in Natural Language Processing (EMNLP 2001)*, pages 44–50, 2000.

[12] Y. Yang and X. Liu. A re-examination of text categorization methods. *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*, pages 42–49, 1999.