Crossroads
The ACM Student Magazine


# Identifying Spam Without Peeking at the Contents

by ***Shlomo Hershkop*** and ***Salvatore J Stolfo***

## Introduction

Spam has evolved from an occasional trickling annoyance to a pervasive flood. According to numerous Internet Service Providers, 60 to 80 percent of all email can be classified as spam. This represents a significant attack on the email infrastructure.

Currently available solutions attempt to filter out spam based on analyzing the contents of the message and calculating a score to indicate the 'spami-ness' of the message. However, users can typically identify their junk email without having to open and read the contents of the specific message. In this article, we outline the general problem, review current options, and propose a new *user-level behavior model* to identify spam messages. We show the performance of this approach and discuss some applications and future directions.

### Background

To understand and address the problem, we must first establish common definitions, explain why the problem exists, discuss limitations of current solutions, and make suggestions of work that can lead to solutions to alleviate the spam problem.

Email is not a permission-based service, yet one may observe and model an individual user's behavior to calculate an expectation of the user's normal email usage. Computer algorithms can be used to learn what types of emails the user opens and reads, and those which he/she immediately discards. For example, electronic bills or annoying forwards from friends might be unwanted, but they are not spam if the user reads them and sometimes responds. Those emails have a clear source marked on the email;

thus, one can easily stop those emails by applying a relatively simple filter rule (such as 'block forwards from user X').

To guide this study, we propose a simple definition of spam: *spam* is all email the user does not want to receive and cannot easily stop receiving. Thus, spam is in the eye of the beholder; the user is the one who decides what is really spam. In this definition, we distinguish spam from email the user would rather have not received but accepts anyway.

Informally, most users agree which messages they would classify as spam. They can identify junk messages without having to open and read the message; why shouldn't computer algorithms be able to do the same? The envelope of the email in combination with user's past behavior should provide sufficient information to decide that an email is unwanted.

Note, that our definition of spam is divorced from the content of the email message, and from the number of recipients. Current spam detection and filtering systems define spam by relying mostly on the contents of the email message and upon the frequency of re-occurrence of the same email message or body text among groups of users. Generally, the current literature defines spam to include unsolicited advertisements [**5**, **12**, **16**, **18**], fraudulent messages [**9**, **13**, **19**], and adult content [**20**, **22**] emails. Although all these definitions may be acceptable, all of these are content-based definitions whose category depends upon the meaning of the message. Our definition does not have this limitation, as it is based on the individual user's behavior.

It is important to note that spam is not only annoying to the individual user, but also represents a security risk and resource drain on the system. Email is a cost effective method of marketing legitimate products or services to millions of users, but it can also be used to conduct scams and confidence schemes to steal information or user identities [**7**, **17**]. By weeding out spam from the email stream, the user is once again empowered to use their email for what it was meant to be: a personal communication tool.

Although all these definitions may be acceptable, all of these are content-based definitions whose category depends upon the meaning of the message. Our definition does not have this limitation, as it is defined based upon a user's behavior.

**Current Solutions**

To address the problem of detecting and filtering email spam, there are four general approaches being explored: preemption, filtering, legislation, and implementation.

The first is *preemptive protection*, putting the burden of protection on the user. This method [**10**] instructs the user to encode their email address in such a way that spammers do not harvest it. Other preemptive techniques [**5**] include choosing unusual email addresses, and using multiple throwaway or 'one-time-use' email accounts. These techniques assume spam as a user's problem. In contrast, a second approach seeks to filter spam email out of the user's inbox. There are three popular methods for filtering out the spam: white lists, black lists, and content-based filtering. A complete survey can be found in [**19**].

Some email systems, such as Hotmail, allow users to specify a list of email addresses to block. Because spammers typically forge their source emails, this list quickly fills up and becomes ineffective at stopping spam messages. In addition, filtering techniques have caused many people's email servers to be added to a black list, for no apparent reason. Furthermore, these lists require frequent updates to keep the blacklists current and thus suffer from lag time vulnerabilities.

Besides technological solutions, legislation been employed to defeat spam. A new federal law makes it illegal or economically less attractive to send out bulk, unsolicited email messages. This approach will succeed in reducing bulk emails from law-abiding mailers, who will thus suffer some economic consequence. However, it is generally agreed that legislation will not stop spam, especially from sources where the law does not reach. For example, hijacked computers, hacked to send out spam, breaking the law is clearly not a deterrent. In fact, recent federal legislation has not resulted in any significant drop in spam volume.

Addressing the limitations of the other approaches, some scholars have called for overhauling the email system with a permission-based protocol. Although a new protocol might solve the problem once-and-for-all, the current open email system is very much entrenched, making it unrealistically hard to implement a new protocol and deploy it across the entire Internet in the foreseeable future.

We have focused our previous work on content-based filtering, which has garnered recent attention in the popular media for being the next "all-inclusive" spam solution

[11]. In this paper, we seek to augment content-based filtering with additional user-behavior models to improve accuracy. Much of the current literature on spam filtering makes an important and fundamental assumption: that content alone from the body of the message can differentiate between normal and spam email messages for all users. We challenge this view and explain how additional information about the user's behavior can improve accuracy and effectiveness of content-based filtering. In addition, we address the evolution of spam designed specifically to trick the content-based filters.

In this paper, to test the power of behavior-based spam identification and filtering, we focus our experiments exclusively on non-content-based analysis using a large set of email data acquired from volunteers. We compare content-based spam detection and purely behavior-based spam detection to reveal the relative power of the techniques.

## Spam Filtering

Content-based filtering has been shown to be accurate with some experiments claiming accuracy as high as 98.8% for certain data sets. Yet, these numbers do not reflect two important realities of the spam economy. The first is that the nature of spam content is dynamic and changes over time; it is constantly being updated to reflect changing social norms. The second is that there is a financial incentive for spammers to circumvent the filters so that their message can be delivered to end users. The filtering models based on content alone do not reflect this reality that spammers are clever enough to change their messaging to avoid having their messages easily filtered.

In general, there are two approaches to do filtering. The first relies on hard-coded rules, which are periodically updated for and by the user [23]. Each email is given a certain amount of spam points based on some rule set. If the email exceeds some threshold score, it is typically quarantined for later deletion by the user. The second approach uses machine learning models, leveraging work done on text classification and natural language processing. A training set of emails is created with both normal and spam emails, and a machine learning technique is chosen to classify the emails. In most cases, a preset classifier automatically classifies emails for the user behind the scenes based on a static learned model. More advanced versions of this technique allows the user to update the learner to give it feedback on how it is doing and continuously train the classifier using new examples.

One of the earliest filters based on machine learning models was proposed in [22]

using Naïve Bayes models to filter spam. For their model, they used the 500 highest frequency words as a binary feature vector, 35 hand-crafted rule phrases, and 20 hand-crafted non-textual features such as the sender's domain. They make some strong assumptions, which most of the literature assumes to be true. The first is that spam can be detected based on textual content alone, just like any other information retrieval task. A second assumption is that they could make broad generalities based on their local corpus of 2500 emails.

Later work [**1**,**2**] introduced the 'bag of words' model, basing the probabilities on tokens found within the email body along with costs associated with either deleting or marking emails as spam.

Our work on spam filtering differs in the fact that it tries to filter out spam using behavior as a model. Each email received over time, by a particular user, forms a larger picture of the individual's email account behavior. Thus, we use statistical features that profile the user's behavior to provide evidence that a received message is indeed one the user would ordinarily filter. This concept of email behavior profiling using machine learning techniques (for security tasks) was first introduced in Columbia's Malicious Email Tracking (MET) and Email Mining Toolkit (EMT) systems [**3**, **24**, **25**]. The behavior models in this paper consist of non-content features which help distinguish spam email from normal email. The features help identify spam without having to parse or tokenize or otherwise interpret the contents of the body of the message.

We have experimental results comparing both content and non-content models to filter out spam messages. Our data consists of a subset of normal and spam emails collected over the year 2002-2003 from accounts on Columbia's Computer Science Department mail server and a single unrelated Hotmail account. In addition, we test the premise that spammers adapt their messaging over time to confound a content-based filtering model that exists at one point in time. Hence, such models tend to fail to remain accurate for very long. We also show results of varying the amount of spam in the training set with results on detection rate, total error rate, and false positive rate.

## Behavior Models

We consider two types of behavior models in this work, one based on typical behavior from a user perspective and one on typical behavior of an email message. Both of these models can be used to detect differences from the past behaviors, which can be

quantified into a probability score.

The user's behavior is modeled with respect to their typical email usage, the frequency and type of messages received and sent, and the typical recipients with whom they exchange messages. Known as a behavior profile, this type of model is computed over some training period to learn how the user behaves within the email account. These behavior models have been shown to be effective at detecting viral email propagations [**24**, **25**, **26**]. The measurements of the user's email behavior include the frequency of inbound/outbound email traffic, the specific times emails arrive and are sent, the "social cliques" of a user, and the user's response rates when replying to specific senders.

For example, looking at the user's inbox, we can learn which recipients are important to the user based on how long it takes for an email response to take place. The faster one responds to an email, the more important the other side is considered to be. If this rate is violated in the current behavior, we can use it as evidence together with other models. See [**24**] for more details.

The second type of behavior is specific to how spam behaves and how it appears in the message folder among normal emails. In general, spam emails can be spotted at a glance because they appear anomalous with respect to the normal set of emails received and opened by the user.

We extract the following features to describe spam behavior for the study in this paper.

- For both receiver and sender of an email
  - Email name: this is the address before the @ sign
  - Domain: this is everything after the @ sign
  - Zone: this is after the @ sign and after the last '.' (e.g.,, com, org, net, etc.)
- Number of recipients on a specific message
- Size of message
- Number of attachments
- Mime type of email
- Number of "re's" in the subject line

Based on only these characteristic features (notice no content features from the body

are used here), we compare the accuracy of the models to the current content-based models in other literature. These features were picked from a larger set because they maximized the information recall score over our dataset.

We compare our results learned from modeling this list of non-content features to models computed using Naïve Bayes and n-gram over the actual body content of the emails. We explain these models in the next sections.

## Text Classifier

Many recent works have applied traditional text classification algorithms directly to the email domain. We have applied two of them here.

### Naïve Bayes

Naïve Bayes classification is a simple, probabilistic method based on Bayes' rule of combining evidence. In the email context, it simply means the probability of spam given a certain word or token can be calculated if you know the probability of unwanted emails and how often this word appears in them, divided by how often the word appears in general in any email body.

Naïve Bayes is known for its robustness to noise (emails which we mislabel in training) and fast learning time, which is linear in the number of samples. In our work, we use a multinomial model to represent the information in documents, which yields the familiar "bag of words" representation. This has been shown to be a better model for email spam detection in [**15**].

### N-gram

An alternative to calculating the 'spami-ness' of the individual tokens or words is to try to find patterns between wanted and unwanted emails. Pattern matching ideas have been applied to the email domain [**21**], author identification [**27**], and plagiarism detection.

An n-gram represents the sequence of any *n* adjacent characters or tokens that appear in a document. We pass an n-character wide window through the entire email body, one character at a time, and count the number of occurrences of each n-gram. This results in a hash table that uses the n-gram as a key and the number of occurrences

as the value for one email; this may be called a *document vector.*

For example if the word is "character" and we have a 4-gram window, we generate, "char" , "hara", "arac", "ract", etc.

Given normal and spam training emails, we use the arithmetic average of the document vectors as the *centroid* for each set. For any test email, we compute the cosine distance [**16**] against the centroid created for the training set. If the cosine distance is 1, then the two documents are deemed identical. The smaller the value of the cosine distance, the more different the two documents are.

Given a new unknown email *x*, we would like to see if it is closer to the spam centroid or non-spam centroid. We can convert x into its n-grams and then calculate how close it is to spam.

The formula for the cosine distance is:

$$D(x,y) \quad \sum_{j=1}^{J} x_j y_j / \left(\sum_{j=1}^{J} x_j^2 \sum_{k=1}^{J} y_k^2\right)^{1/2} \quad \cos\theta_{xy}$$

.

Here J is the total number of possible n-grams appearing in the training set and the test email, x is the document vector for a test email, and y is the centroid for the training set. In addition, $x_j$ represents the frequency of the $j^{th}$ n-gram (the n-grams can be sorted uniquely) occurring in the test email. Similarly $y_k$ represents the frequency of the $k^{th}$ n-gram of the centroid.

The cosine distance is used to classify an email as spam or normal depending upon a threshold setting. Varying the threshold clearly varies performance. See [**8**] for a more detailed analysis and examples.

## Experimental Setup

Many papers compare different machine learning methods using some standard and generally available data sets, among them LingSpam [**1**,**2**] and PU1 [**4**]. Both of these collections are small and do not contain enough different types of spam to be a sufficient test of the types of spam seen in the wild. In addition to their limited size, the emails in the non-spam collection introduce a bias that has not been accurately measured. Furthermore, researchers have not compared time-ordered spam and the

degradation of models over time.

We used 10,000 email messages to test out the ideas presented here. These emails were from a distributed collection of users, which should allow our results to be generalized for the average user's email account.

In these experiments, the corpus of available email was marked by hand; as spam was moved by the subject users to a spam folder, all other folders were considered normal emails. Thus, the user's own behavior in "deleting" spam (via the act of moving) created ground truth that describes *the user's behavior.*

**Time-ordered Spam Experiments**

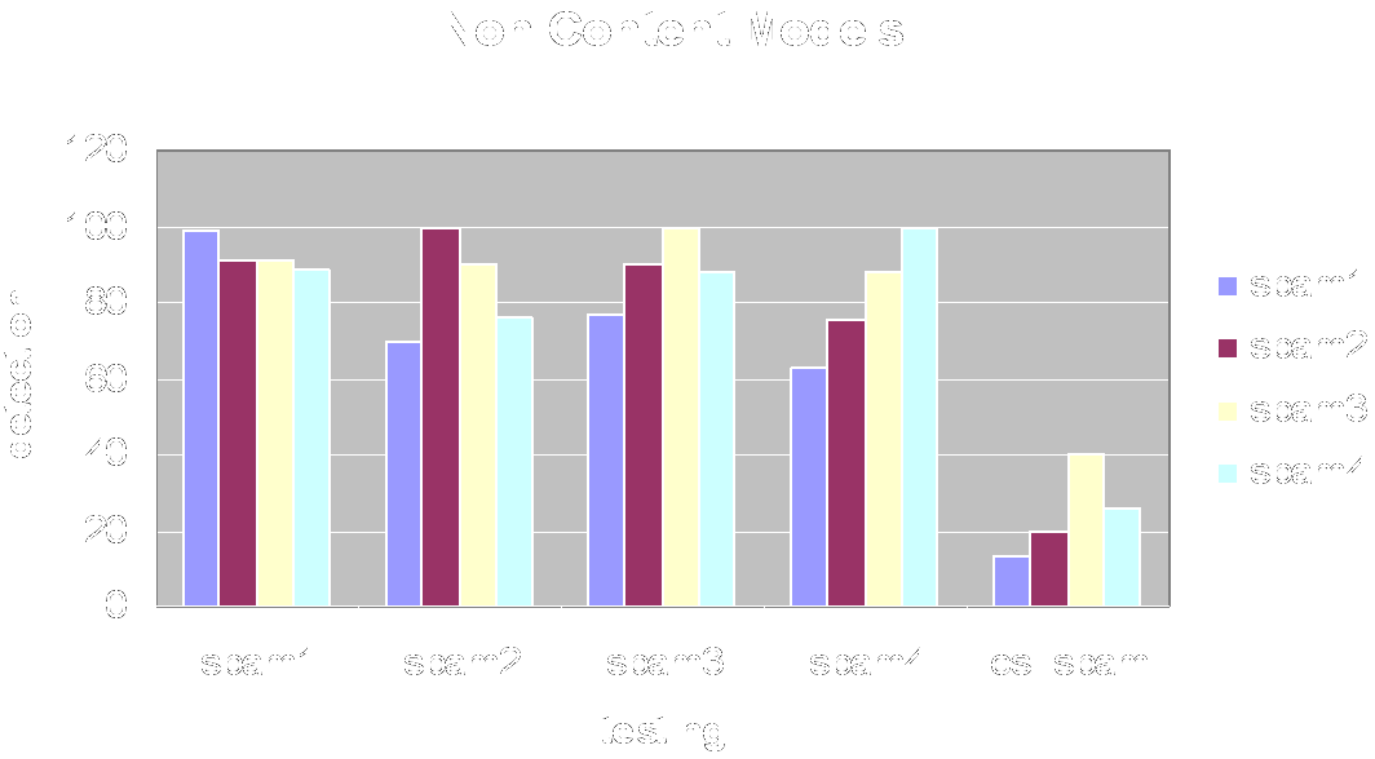


**Figure 1:** Result of training on period X and testing on others with non-content models.

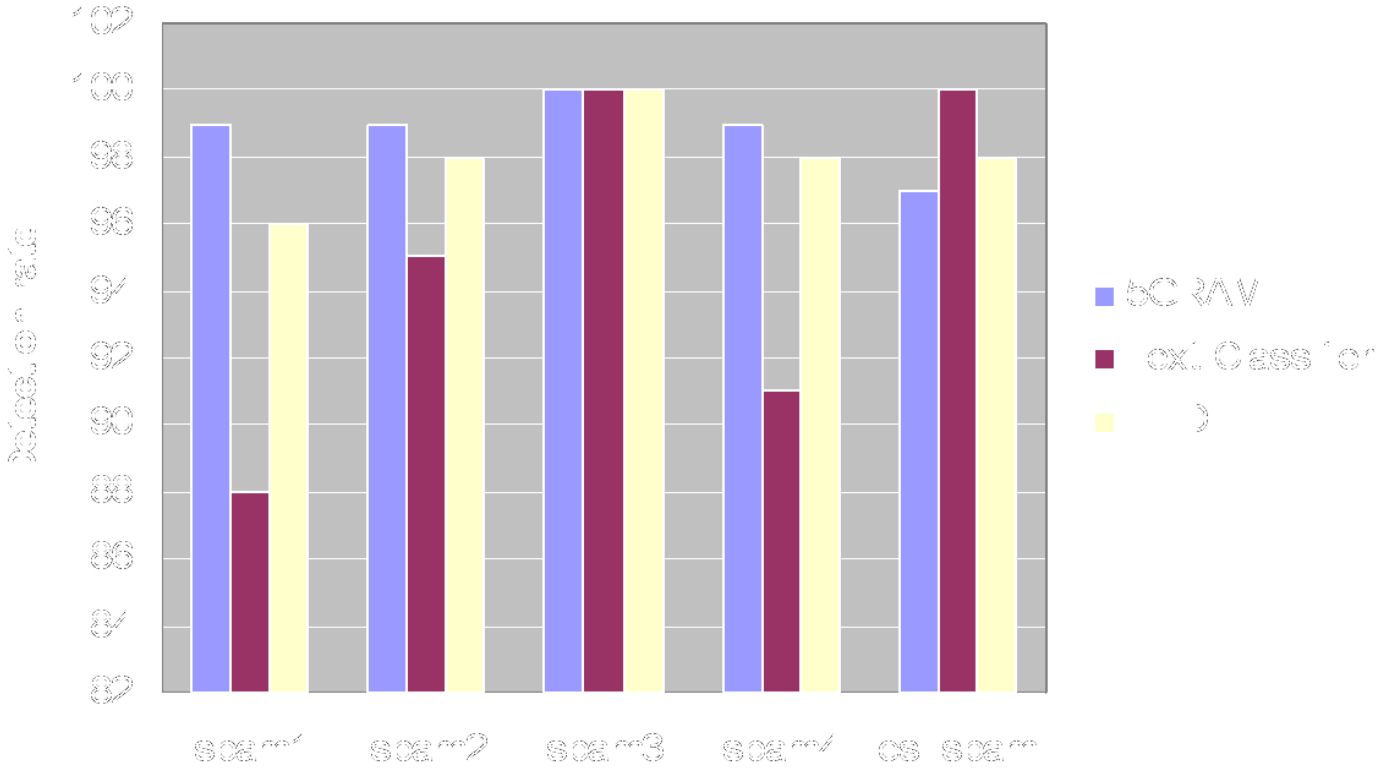

**Figure 2:** Content-based time results.

For the time ordered experiments, the following assumptions were tested. We assume that as time progresses, the models trained on content alone from an earlier time period will become stale. The question is at what point in the future can we accurately detect spam using a trained model, and when does the model's performance begin to degrade as new spam emails appear attempting to thwart detection.

We divided all the spam from 2003 into four periods with about 2000 messages per period measuring the accuracy of each model on past, current, and future spam, delimited by the time periods noted. We compared both content and non-content based models for accuracy.

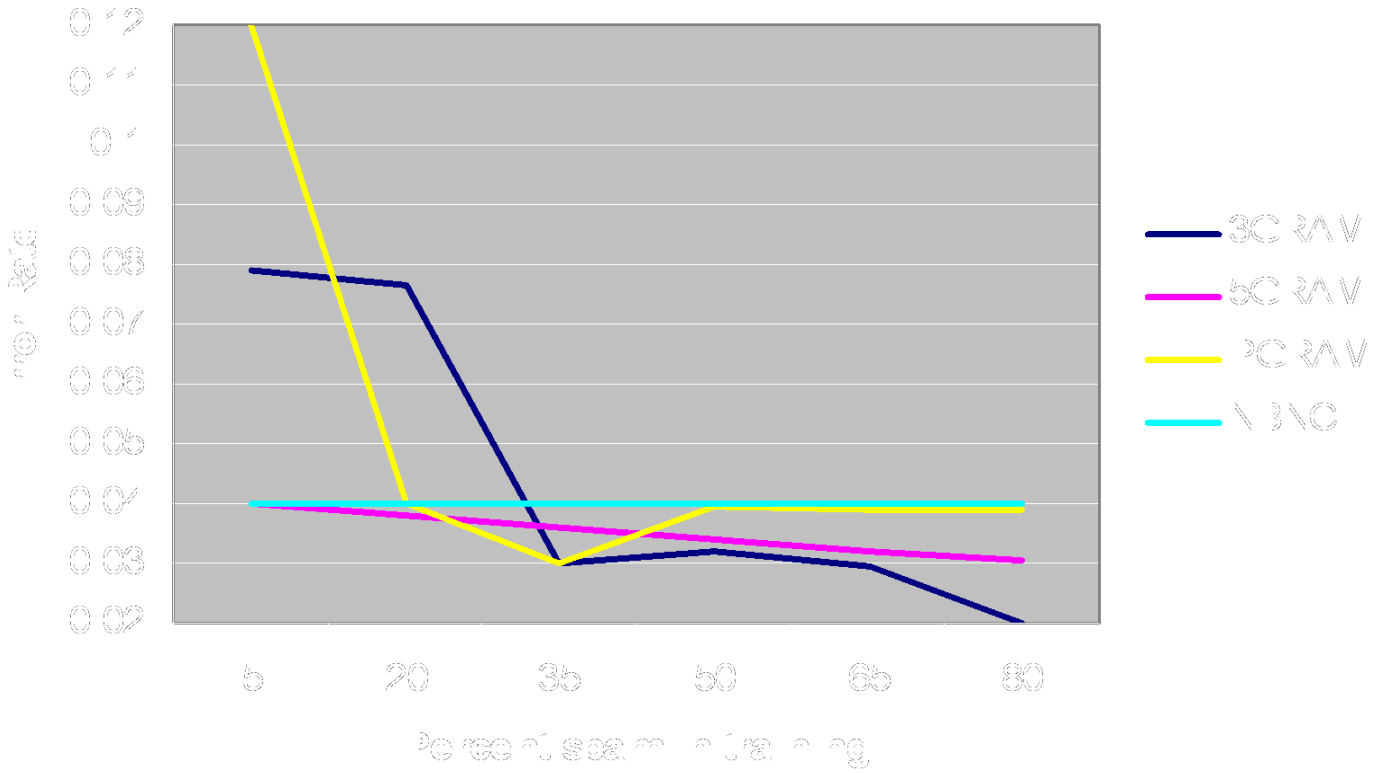

**Figure 3:** Error rate when increasing the percentage of spam in training data.

## Content vs. Non-content Modeling

In order to test our assumption that non-content based filtering works, we ran experiments comparing the detection rate, false positive rate, and error rate of common machine learning methods against the non-content models. We collected 4000 spam messages from an active Hotmail account over the 2003-2004 period, and all emails on the Columbia Computer Science Department server over the same period for the users who volunteered their emails. We used different sets of machine learning methods to measure how well we could filter spam based on different percentages of spam in the training data. The methods used for the content model running over the bodies of the emails were Naïve Bayes over content (**PGRAM**) [**11**], n-gram of five (5GRAM) and n-gram of three (3GRAM). For non-content models we used Naïve Bayes (**NBNC**) over the pre-selected features described above.

We varied the amount of spam in the training set to see how this also affects the accuracy of each model. To our knowledge, none of the current literature explores the effects of training sample size on the accuracy of the model.

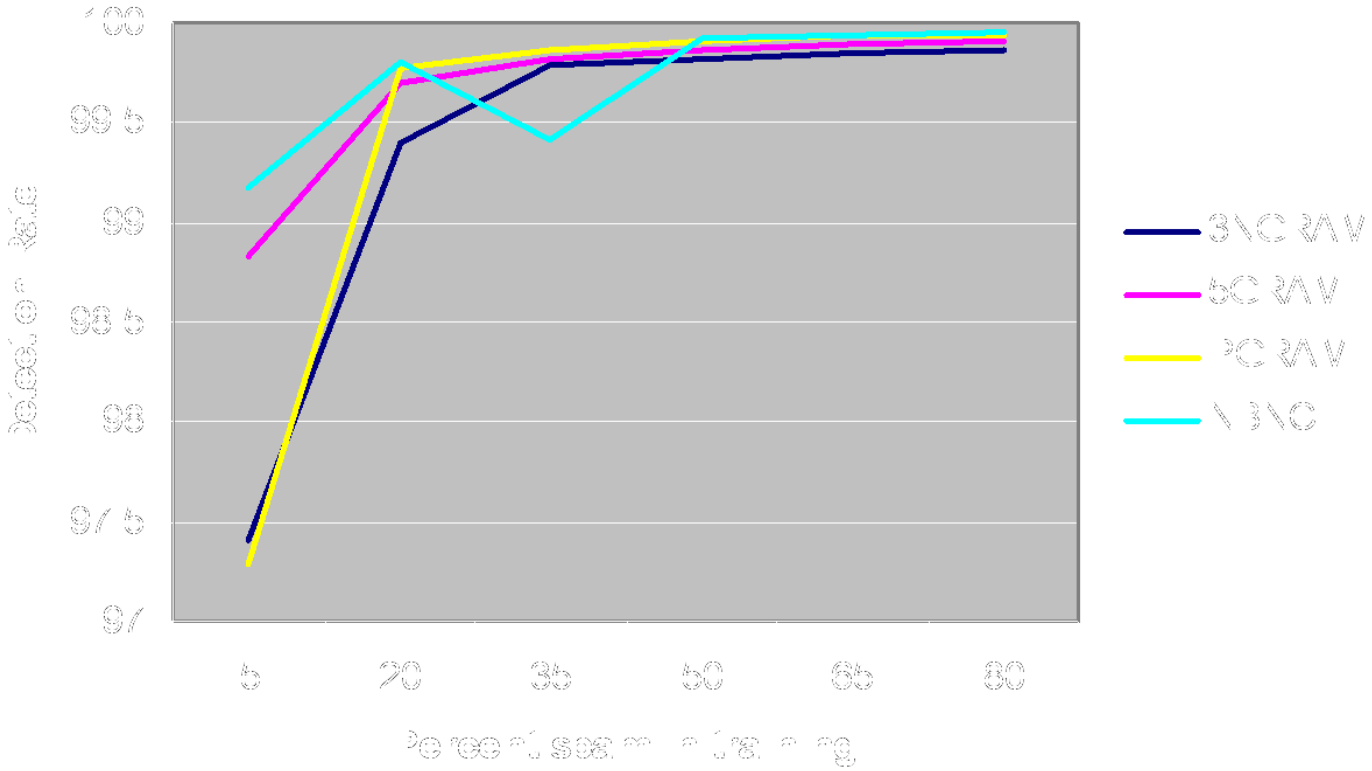

**Figure 4:** Result of detection rate when increasing the amount of spam in training sets.

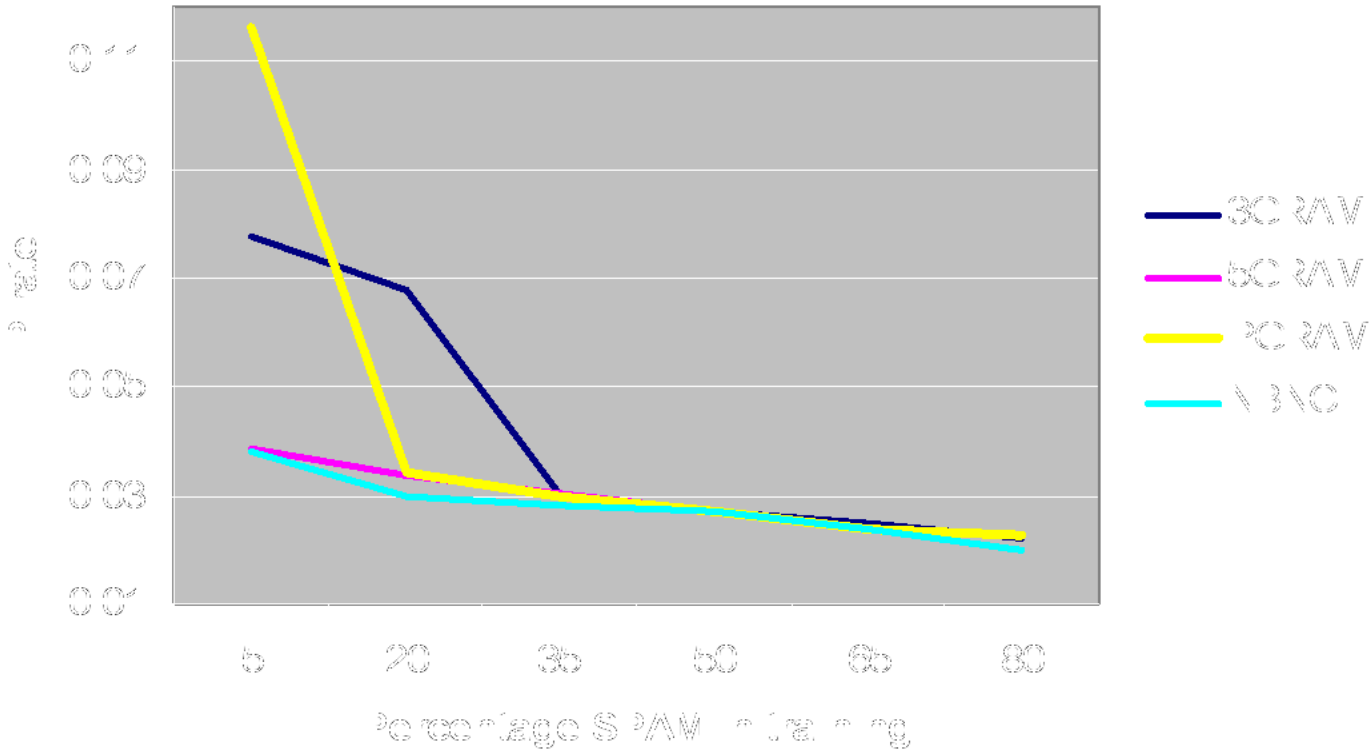

**Figure 5:** FP rate when increasing amount of spam in training sets.

## Results and Discussion

**Figure 1** shows the results of time-ordered experiments. Notice that, for each training period, the accuracy of each model is consistent with the past, but falls off in accuracy rapidly for future emails. Crafty spammers change their emails consistently to avoid detection and filtering. In addition the content-based methods performed about the same as what is displayed in **Figure 2**.

When examining the results generated by the second experiment, comparing different content-based methods and the non-content model, three plots are of interest. The first is the detection rate, which is the total number of unwanted emails detected divided by the total number of spam. This number is an indication of the effectiveness of the model to weed out spam messages. **Figure 4** displays the result with the non-content based outperforming the content-based methods. What is not shown here is that the size of the model and the computation time is about an order of magnitude faster for non-content based models compared to the others. The reason is that the non-content models are based on features mostly in the envelope of the email message, which are available even before the entire message must be processed. Further, there is far more variability in the body of an email which naturally increases the complexity of the modeling of content.

The second number we are interested in is the false positive rate. This represents the number of non-spam emails flagged as spam divided by the total number of non-spam messages. This number measures the amount of normal emails flagged as spam, which should not be so regarded. The ideal goal would be to have a 100% detection rate, and a 0% false positive rate.

The last is the error rate. **Figure 3** shows that the non-content models have a consistently low error rate, which is not based upon the number of spam examples in the training data. The error rate is defined as $\frac{fp + fn}{tp + fp + tn + fn}$. In other words, the error rate is the amount of emails misclassified regardless of the mistakes made by the model.

To summarize, the NBNC model performed with greater than 99% accuracy, while the PGRAM model performed with 97% to 99% accuracy varying as the amount of spam increased. The accuracy of both NGRAM models was between 80-90% overall. The NGRAM performed poorly because spammers have been adapting and adding padding

to the contents of the spam, which is throwing off the n-gram calculations.

We also tested each model on a set of 220 random normal emails, and another 2000 unseen spam messages. These spam messages were chosen because some had made it through a rule-based filter. Here the NGRAM models were consistently identifying 100% of the normal emails but only about 50% of the spam. Both NBNC and PGRAM had about 98% detection for both normal and spam emails. The emails missed by the PGRAM model were those with empty bodies, understandably as it is based on the contents of the email. The emails missed by the NBNC were those that were borderline spam, in other words, certain spam which might look like mailing list emails, and even humans would have difficulty in recognizing them as spam.

## Future work in Spam Detection

The spam problem is usually cast as a *two-class* learning and detection problem, in other words, learning to differentiate between good and spam emails. Some scholars have responded [**6**, **9**, **14**] by suggesting *multi-class* classification. This might not be the best way to approach the problem. For example, any user can usually recognize any other user's spam without knowing their non-spam emails.

We feel a better way would be to think of it as a *one-class* classification problem.

The basic premise would be to use a subset of spam emails to classify unknown messages without using normal messages as part of training. This *unsupervised training* method would be the same across users who have different normal emails but share some common spam emails. It would be interesting to compare current filtering models with non-content models in the context of an unsupervised setting.

The next step in behavioral models is to implement behavior profiles. Although the models presented in this paper show promise, account profiles should allow us to identify anomalous emails with greater accuracy. Future work will address this and explore how often a model needs to be re-computed in a completely automatic fashion. The more autonomous email filters are made, the more widely they will be deployed.

We note that all these techniques do not address a related problem of detecting and dealing with a virus or malicious attachment, which originates from a white-listed source email address. If a friendly user gets infected and starts to send either spam or viruses their behavior profiles would be able to detect an anomalous action. This

advantage of protecting the user's online reputation is not currently possible with the set of features being used by spam filters, as they are specialized for the task of filtering spam.

The makeup of current spam messages reflects the reality that spammers are crafting their messages to make sure they can defeat current content-based filters. Because of mimicry attacks, current filters need to upgrade their capabilities to stay ahead of the spam curve. Behavior models will, we believe, add to the arsenal of techniques to filter spam more effectively.

## References


1

Androutsopoulos, I., Koutsias, J., Chandrinos, K., Paliouras, G., and Spyropoulos, C. An Evaluation of Naïve Bayesian Anti-Spam Filtering. In *Proceedings of the Workshop on Machine Learning in the New Information Age,* 2000. <**http://ww.ics.forth.gr/~potamias/mlnia/paper_2.pdf**>.

2

Androutsopoulos, I., Koutsias, J., Chandrinos, K., and Spyropoulos, C. An Experimental Comparison of Naïve Bayesian and Keyword-based Anti-spam Filtering With Personal Email Messages. In *23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2000, pp. 160-167.

3

Bhattacharyya, M., Hershkop, S., Eskin, E., and Stolfo, S. J. MET: An Experimental System for Malicious Email Tracking. In *New Security Paradigms Workshop (NSPW-2002)*, Virginia Beach, VA, 2002.

4

Carreras, X. and Mrquez, L. Boosting Trees for Anti-spam Email Filtering. In *RANLP-01, 4th International Conference on Recent Advances in Natural Language Processing*, Tzigov Chark, BG, 2001.

5

Center for Democracy & Technology. Why Am I Getting All This Spam, 2003. <**http://www.cdt.org/speech/spam/030319spamreport.pdf**>.

6

Cohen, W. Learning Rules That Classify E-mail. In *Machine Learning in Information Access: AAAI Spring Symposium (SS-96-05)*, 1996, pp. 18-25.

7

Cranor, L. F. and LaMacchia, B. A. SPAM! *Communications of the ACM*, 41(8),

1998, pp. 74-83.

8

Damashek, M. Gauging Similarity With Ngrams: Language Independent Sorting, Categorization and Retrieval of Text. *Science*, *267*(5199), 1995, pp. 843-848.

9

Drucker, H., Wu, D. and Vapnik, V. N. Support Vector Machines for Spam Categorization. *IEEE Transactions on Neural Networks*, 10(5), 1999, pp. 1048-1054.

10

Gabber, E., Jakobsson, M., Matias, Y. and Mayer, A. J. Curbing Junk E-Mail via Secure Classification. In *Proceedings of the Second International Conference on Financial Cryptography*, 1998, pp. 198-213.

11

Graham, P. A Plan For Spam, 2003. <**http://www.paulgraham.com/spam.html**>.

12

Hall, R. J. A Countermeasure to Duplicate-detecting Anti-spam Techniques, ATT Labs Technical Report, 1999.

13

Hidalgo, J. M. G. and Sanz, E. P. Combining Text and Heuristics for Cost-Sensitive Spam Filtering. In *Fourth Conference on Computational Natural Language Learning and of the Second Learning Language in Logic Workshop*, Lisbon, 2000.

14

Itskevitch, J. Automatic Hierarchical E-Mail Classification Using Association Rules, Simon Fraser University Masters Thesis, 2001.

15

John, G. and Langley, P. Estimating Continuous Distributions in Bayesian Classifiers. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, 1995, pp. 338-345.

16

Kolcz, A. and Alspector, J. SVM-based Filtering of E-mail Spam with Content-specific Misclassification Costs. In *Workshop on Text Mining (TextDM'2001)*, San Jose, California, 2001.

17

Krim, J. Spam's Cost To Business Escalates. *Washington Post*, March 13, 2003, A01.

18

Manaco, G., Masciari, E., Ruffolo, M. and Tagarelli, A. Towards an Adaptive Mail Classifier, 2002. <**http://www-dii.ing.unisi.it/aiia2002/paper/APAUT/**

**manco-aiia02.pdf**>.

**19**

Mertz, D. Six Approaches to Eliminating Unwanted E-mail, IBM developerWorks, 2002. <**http://www-106.ibm.com/developerworks/linux/library/l-spamf.html**>.

**20**

Provost, J. Naïve-Bayes vs. Rule-Learning in Classification of Email. Technical Report AI-TR-99-284, University of Texas at Austin, Artificial Intelligence Lab, 1999.

**21**

Rigoutsos, I. and Huynh, T. Chung-Kwei: A Pattern-discovery-based System for the Automatic Identification of Unsolicited E-mail Messages. In *CEAS 2004*, Mountain View, California, 2004.

**22**

Sahami, M., Dumais, S., Heckerman, D. and Horvitz, E. A Bayesian Approach to Filtering Junk E-mail. In *AAAI-98 Workshop on Learning for Text Categorization*, 1998.

**23**

SpamAssassin Project. <**http://spamassassin.apache.org/**>.

**24**

Stolfo, S. J., Hershkop, S., Wang, K., Nimeskern, O., and Hu, C.-W. Behavior Profiling of Email. In *1st NSF/NIJ Symposium on Intelligence & Security Informatics (ISI 2003)*, Tucson, Arizona, 2003.

**25**

Stolfo, S. J., Hershkop, S., Wang, K., Nimeskern, O., and Hu, C.-W. A Behavior-based Approach to Securing Email Systems. *Mathematical Methods, Models and Architectures for Computer Networks Security*, 2003.

**26**

Stolfo, S. J., Li, W.-J., Hershkop, S., Wang, K., Hu, C.-W., and Nimeskern, O. Detecting Viral Propagations Using Email Behavior Profiles. *ACM Transactions on Internet Technology (TOIT)*, May 2004.

**27**

Vel, O. D., Corney, M., Anderson, A. and Mahay, G. Language and Gender Author Cohort Analysis of E-Mail for Computer Forensics. In *DFRWS02*, Syracuse, NY, 2002.

---

**Biography**

Shlomo Hershkop (**shlomo@cs.columbia.edu**) received his BA from Yeshiva College in 1999, and Masters from Columbia University in 2001. He is currently pursuing a PhD at Columbia University. His areas of research are Data Mining, Security, Anomaly Detection, and Email Analysis.

Salvatore J. Stolfo (**sal@cs.columbia.edu**) is Professor of Computer Science at Columbia University. He received his PhD from the NYU Courant Institute in 1979 and has been on the faculty of Columbia ever since. He has published extensively in the areas of Parallel Computing, AI, Knowledge-based Systems, Data Mining, and most recently Computer Security and Intrusion Detection Systems. His most recent research has been devoted to distributed data mining systems with applications to fraud and intrusion detection in network information systems.