A Human Factors Approach to Spam Filtering

Robert Beverly
MIT CSAIL
rbeverly@csail.mit.edu

ABSTRACT

While misclassified spam imposes a burden on end-users, the cost of false positives is much higher. Therefore, significant effort has been spent attempting to conservatively optimize this binary classification decision. While modern email filtering is quite effective, the sheer volume of spam implies that even high precision and high recall filters yield non-zero misclassification, i.e. no classifier is "perfect" against adaptable adversaries. This paper makes explicit recognition of this balancing act and argues for: i) removing the burden of perfect classification from the classifier; ii) separating classification and filtering tasks; and iii) a human factors approach to filtering. We present initial work on SpamGUI, an operational and publicly available embodiment of these ideas.

1. INTRODUCTION

In the war against unsolicited electronic mail, significant effort is expended to provide an acceptable level of protection from junk mail while maintaining a low false-positive rate. The result of many years of research and commercial development is email filtering systems which are highly effective. Various techniques have emerged that are typically employed in combination for maximal effect, including content analysis [7, 11], reputation [12, 13], traffic characterization [3, 10], etc.

Yet, *no classifier is perfect*. Because of the sheer volume of incoming mail [8], even very low false-negative rates result in many unwanted messages making their way into a user's inbox. This filtering "plateau" effect [16], i.e. the difficulty in achieving better than 99.9% accuracy, is a critical weakness in the battle against spam. More distressingly, any non-zero number of false-positives (valid messages incorrectly marked as spam) is very costly to the user. For example, a missed message in business has real impact. Therefore, classifiers often act conservatively in balancing false-positives against false-negatives, allowing suspected spam messages through when the classifier cannot make a strong inference. Making matters worse, the inputs to email filtering, learning, and classifying tasks are adversarial. The means of injecting spam and circumventing existing filtering are continually evolving in response to mitigation efforts.

The human end-recipient is the ultimate arbiter of junk versus valid email. Unfortunately, humans have scarce and valuable attention available for such menial filtering duties. Yet, while human filtering is not possible on a per-message basis, we maintain that it is possible to bring the human back into the filtering loop in an intuitive and non-intrusive manner to better address many of the aforementioned issues.

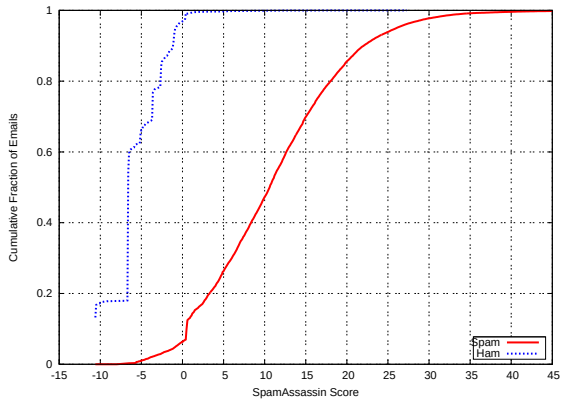
This position paper advocates a new perspective of handling email filtering by explicitly separating the spam classification task from the filtering decision. As a practical embodiment of this idea, we have developed SpamGUI. SpamGUI uses human factors engineering to add novel ways to view an email inbox. We eliminate the notion of separate spam and ham (i.e. valid email) folders in favor of a single inbox. Rather than relying on a sensitive threshold value to place messages into a spam folder, SpamGUI uses the continuous value spam score to sort and visualize all inbox messages. Messages are thereby presented in relation to one another, using color and other techniques, in accordance to their probability of being valid or spam. As a result, messages which are likely spam are outside the user's immediate window view, while ambiguously scored messages are presented in relative order of the classifier's confidence. SpamGUI thus provides the following benefits:

1. Removes the burden of performing "perfect" classification from the classifier. Classifiers need only be "good enough." Sensitive and error-prone score and threshold tuning are unnecessary.
2. Graphically demarcates spam from ham in an intuitive manner. Spam and ham naturally separate from view in relation to the classifier's confidence.
3. Removes the potential for false positives. Messages that would traditionally be near the binary threshold value are not removed from view, and are not easily missed.

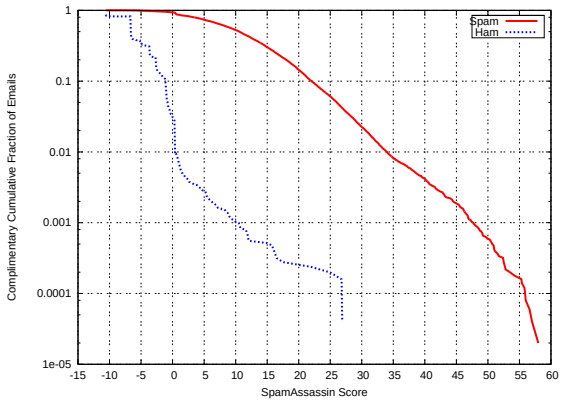
Moreover, our SpamGUI is part of the larger idea that users are inundated by email of all types. Such users would be well-served by mail applications with more intelligent means of inferring and presenting emails in order of importance. SpamGUI is a proof-of-concept implementation of this concept where spam is an instance of a particularly unimportant class of messages.

Our hope is that SpamGUI motivates additional research into the application of human factors engineering and design to spam and email in general. The remainder of this paper reviews related work, motivates our approach using analysis of popular spam corpora, and describes SpamGUI in detail. We conclude by discussing avenues by which we envision this work being extended.

CEAS 2009 - Sixth Conference on Email and Anti-Spam July 16-17, 2009, Mountain View, California USA



(a) Cumulative distribution of message scores.



(b) Complimentary CDF of message scores.

Figure 1: Distribution of SpamAssassin scores as evaluated on the TREC corpus. Any selection of threshold value results in some number of false-positives or false-negatives.

2. FILTERING'S BALANCING ACT

We first motivate SpamGUI by presenting an empirical analysis of the popular SpamAssassin [7] classifier. We analyze SpamAssassin primarily because of its widespread use, public availability and amenability to code inspection. While many commercial and open filtering systems exist, our objective in this section is to demonstrate *the intrinsic difficulty of the classification task*. Thus, the analysis here is meant to be instructive rather than complete.

Spam classifiers perform a variety of tests on incoming email messages, for instance content analysis, reputation inference, header checks, etc. These tests are composed into a single continuous-valued score[1]. Binary classification, whether the message is spam or ham, is typically performed by imposing a threshold value to best separate the two classes. In SpamAssassin, this value is 5. Because the classifier's performance depends critically on the threshold value, considerable effort is spent to generalize the threshold by using training data to properly set each test's relative weight. SpamAssassin, for example, uses a neural network single-perceptron method to optimize test weights according to this threshold value.

How dependent is SpamAssassin on this threshold value? We examine 75,419 email messages from the TREC 2007 [4] dataset. The corpus contains ground-truth labels of 25,220 ham emails and 50,199 spam messages. In our experiment, each message receives a score using a default installation of SpamAssassin. The results are based primarily on message content as we do not enable auxiliary authentication, real-time blacklist or DNS checks (which are not valid for old messages). While SpamAssassin is extensible, our focus here is in illuminating the classification tradeoff.

Clearly, some email inputs are easier to form strong inferences over than others, i.e. messages that are either spam or ham with high probability. More challenging, and the typical source of false-positives, are messages which receive a weak spam score, i.e. those with values close to the threshold. In Figure 1(a), we plot the cumulative distribution of SpamAssassin scores across both spam and ham messages in the TREC corpus.

We are asking "how close to the threshold are typical messages," which is a proxy for the performance dependence of the system on the threshold value. Encouragingly, and not surprisingly given the score optimization process, ham messages are well-separated from the threshold. 99.72% of the ham messages receive a score below the threshold value – a seemingly good result. However, this implies that 0.28%, or approximately 71 valid ham messages, are misclassified.

Approximately 10% of all spam messages are within a value of 1 from the threshold (i.e. 10% of spam messages receive a score between 4 and 6). While 10% appears to be a small fraction, the sheer volume of spam messages implies that the ability of the system to filter these "close" messages does in fact depend strongly on the threshold.

Figure 1(b) provides the same data as a complimentary cumulative distribution in order to intuitively demonstrate the inherent tradeoff classifiers must make. We see that *any value selected for a threshold is a compromise* resulting in some number of false-positives or false-negatives. In particular, to achieve zero false-positives, at a threshold of 27, an unacceptably high number of false-negatives result.

This analysis highlights a well-known result: without perfect classification, classifiers must make a tradeoff between false-positives and false-negatives. While imperfect classifiers are acceptable in most practical situations, imperfect classification in the context of spam presents many uncomfortable difficulties. As an alternative to such strict filtering constraints, we develop SpamGUI.

3. SPAMGUI

SpamGUI is a publicly available [2] extension to the popular Mozilla Thunderbird [9] Mail User Agent (MUA). SpamGUI is an embodiment of our position that the classification of email should be a separate task from the filtering of email. SpamGUI therefore attempts to: i) remove the burden of perfect classification from the classifier; ii) graphically demarcate spam from ham in an intuitive manner; and iii)

[1]Equivalently, each test can be viewed as a weighted coefficient in other schemes.

remove the potential for any false positives.

3.1 Related Work

Electronic mail is a critical Internet resource, with profound societal and economic impact. Early in the development of email, potential weaknesses were recognized [5]. In the face of unprecedented strain on the email architecture, research efforts to date have mainly focused on filtering junk messages. Beyond making the transition to Graphical User Interfaces (GUIs), email clients have remained largely unchanged for decades.

Human-Computer Interaction (HCI) work on email has primarily focused on workflow or task management, attentional factors, cognitive styles, and social clustering [15]. Alharbi and Rigas [1] emphasize the utility in incorporating time as a first-class component of the user's mailbox presentation, while Gruen et al. focus on improving thread presentation and searching in [6]. We are unaware, however, of research in thwarting spam through the use of HCI techniques.

The current state of the art in preventing false-positives is to "quarantine" weakly classified spam messages. In these systems, a notification of the message, along with the subject and sender, is delivered in lieu of the actual message. The user then must explicitly release the message from quarantine if, in fact, it is a valid message. While this strategy helps address false-positives, it only indirectly addresses the underlying issue. It is our contention that HCI can play a significant role in spam classification in future mail systems.

3.2 Design

Modern email systems make spam filtering a strong action: messages classified as likely spam often do not reach the end user's mail agent, e.g. Outlook, Thunderbird, etc. When spam messages are available to the user, the mail server typically places probable spam messages in a separate folder. The volume of spam messages [8] implies that few users are able to cull their spam folder for possibly misclassified false-positives. The spam folder thus effectively implements an "out-of-sight out-of-mind" strategy.

SpamGUI, by contrast, recognizes that humans are the ideal spam filters, but must not be burdened with non-trivial amounts of junk mail. Rather than making the filtering task binary, SpamGUI depicts messages continuously relative to each other on the basis of their spam score. This perspective on spam more naturally matches how the spam classifiers operate: each message has an associated continuous score or confidence value. As a result, messages which are likely spam are readily removed from user view, while ambiguously scored messages are presented in relative order of the classifier's confidence.

Figure 2 depicts an example screenshot of SpamGUI running on the messages received in a single day for a user. The message thread pane colors and sorts messages according to spam score. The colors are fully configurable by specifying color end-points and interpolating a line in RGB space between the colors. Here we use green for spam and red for ham. By providing a means to customize the color range, we address in part the needs of color blind users. Further work, however, is necessary to completely assess the best presentation scheme for the widest range of users.

Note the crucial difference that both spam and ham messages are placed in the same folder. Instead, the message presentation allows the human to intuitively perform filtering *without examining each individual message.* As seen in this example, a natural demarcation "line" between messages the user should focus on emerges.

3.3 Caveats

Some initial caveats became apparent immediately in our deployment. First, the act of retraining traditional classifiers after an error is quite valuable. Often this retraining is achieved simply by moving email between folders. By eliminating the notion of a spam folder, SpamGUI loses this ability. We are investigating the most intuitive means to provide relabeling. More interestingly, we hope to make the user's basic interaction with the mail client a second-order input to the learner. For example, temporal aspects, e.g. particular messages which are read immediately versus those which languish in the inbox, can provide valuable training data. Misclassified messages could similarly be inferred simply on the basis of whether the mail is read, if the user replies to the email, etc.

Second, presenting all emails may be dangerous to certain classes of users, particularly in the context of current phishing attacks. By placing spam messages in a separate folder, the novice user has a harder time falling victim to such attacks. However, HCI design emphasizes never underestimating power of the novice. Our current implementation removes messages that are spam with very high confidence completely from view. We are currently experimenting and evaluating other techniques for intuitive presentation.

Finally, SpamGUI requires users to think about and process their email in a new way. Some users, particularly those accustomed to using their inboxes for task management, may be uncomfortable with spam messages being unfiltered. To address the needs of such users, we are investigating alternate presentation techniques including font sizes, greyed backgrounds, etc.

4. DISCUSSION

It is our contention that the continual war against spam, impossibility of achieving perfect machine learning-based classifiers, and volume of email argue for incorporating approaches similar to SpamGUI in the quest to mitigate spam.

Taken further, user behavior and interaction with the mail client itself are potential inputs to existing learning and classification algorithms, an idea with some early exploration in [14].

While SpamGUI is likely imperfect for all users, we have attempted to make it as customizable as possible while maintaining useful defaults. By publicly releasing SpamGUI as a Mozilla Thunderbird extension, we achieve immediate access to a large user-base. We plan to collect feedback from this community and refine SpamGUI over time in addition to performing controlled user studies.

Computer interfaces are highly personal and notoriously difficult to implement well. Rather than specifying an ideal interface, SpamGUI is meant to be illustrative of the utility of human factors engineering for the spam filtering problem. Thus, our hope is that SpamGUI represents a step forward in understanding how to apply HCI within the email domain.

5. REFERENCES

[1] S. Alharbi and D. Rigas. Graphical browsing of email data: An empirical investigation. In *Proceedings of*

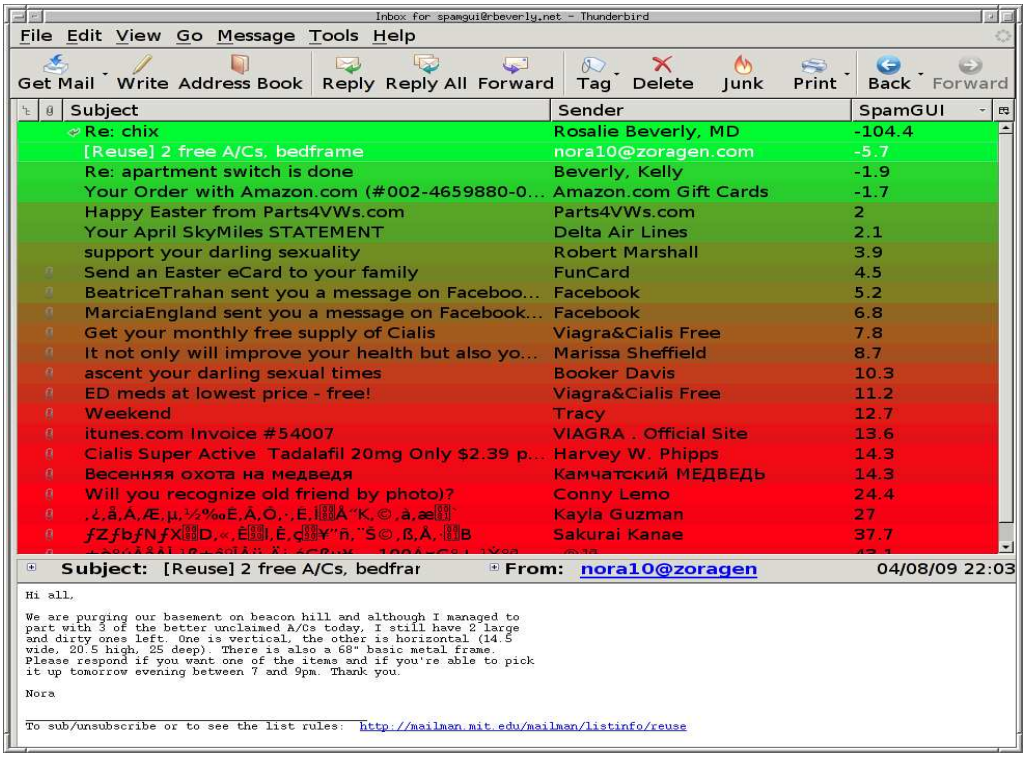


Figure 2: SpamGUI screenshot: messages are presented graphically, here using color gradients and sorting, according to their relative spam score. A natural demarcation "line" between messages the user should focus on emerges. (Figure best viewed in color).

IEEE 5th International Conference on Information Technology, 2008.

[2] R. Beverly. SpamGUI mozilla extension, 2009. `http://www.rbeverly.net/spamgui`.

[3] R. Beverly and K. Sollins. Exploiting transport-level characteristics of spam. In *Proceedings of the Fifth Conference on Email and Anti-Spam (CEAS)*, Aug. 2008.

[4] G. Cormack and T. Lynam. Text retrieval conference (TREC) spam track corpus, 2007. `http://trec.nist.gov/data/spam.html`.

[5] P. J. Denning. Acm president's letter: electronic junk. *Commun. ACM*, 25(3):163–165, 1982.

[6] D. Gruen, S. L. Rohall, S. Minassian, B. Kerr, P. Moody, B. Stachel, M. Wattenberg, and E. Wilcox. Lessons from the remail prototypes. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 152–161, New York, NY, USA, 2004. ACM.

[7] J. Mason. Filtering spam with spamassassin. In *Proceedings of SAGE-IE*, Oct. 2002.

[8] Messaging Anti-Abuse Working Group. Email metrics report, 2007. `http://www.maawg.org/about/EMR`.

[9] Mozilla Foundation. Mozilla thunderbird, 2009. `http://www.mozillamessaging.com`.

[10] A. Ramachandran and N. Feamster. Understanding the network-level behavior of spammers. In *Proceedings of ACM SIGCOMM*, Sept. 2006.

[11] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A bayesian approach to filtering junk e-mail. In *AAAI Workshop on Learning for Text Categorization*, July 1998.

[12] SORBS. Spam and open-relay blocking system, 2007. `http://www.sorbs.net`.

[13] Spamhaus, 2007. `http://www.spamhaus.org/sbl/`.

[14] S. J. Stolfo, S. Hershkop, C.-W. Hu, W.-J. Li, O. Nimeskern, and K. Wang. Behavior-based modeling and its application to email analysis. *ACM Transactions on Internet Technology (TOIT)*, Feb. 2006.

[15] S. Whittaker, V. Bellotti, and P. Moody. Introduction to this special issue on revisiting and reinventing e-mail. *Hum.-Comput. Interact.*, 20(1):1–9, 2005.

[16] W. S. Yerazunis. The spam filtering accuracy plateau at 99.9 percent accuracy and how to get past it. In *MIT Spam Conference*, Jan. 2004.