Can Network Characteristics Detect Spam Effectively in a Stand-Alone Enterprise?

Tu Ouyang[†], Soumya Ray[†], Michael Rabinovich[†], Mark Allman[‡]

[†]Case Western Reserve University, Cleveland, OH, USA
[‡]International Computer Science Institute, Berkeley, CA, USA

Abstract. Previous work has shown that the network dynamics experienced by both the initial packet and an entire connection carrying an email can be leveraged to classify the email as spam or ham. In the case of packet properties, the prior work has investigated their efficacy based on models of traffic collected from around the world. In this paper, we first revisit the techniques when only using information from a single enterprise's vantage point and find packet properties to be less useful. We also show that adding flow characteristics to a model of packet features adds modest discriminating power, and some flow features' information is captured by packet features.

1 Introduction

Spam email is an ever-present irritant in the modern Internet: it uses scarce server and network resources, costs users' productivity, spreads malware, scams users and recruits bots for all manner of malicious purposes. Hence, judging whether a particular email is spam or ham—i.e., legitimate—is a crucial for operators and users. Many different approaches have been investigated and a handful now enjoy regular use. The most useful techniques to-date have been those that leverage (i) properties of the host sending the email (e.g., IP address black- or grey-lists, domain keys [3], etc.) or (ii) properties of the email messages themselves in the form of filtering in mail servers (e.g., SpamAssassin [2]) or users' mail applications (e.g., Apple Mail).

A new class of techniques has emerged, which attempt to use properties of the network traffic to determine whether a message is spam. Beverly and Sollins [4] used transport-level features (e.g., round-trip time, TCP advertised window sizes) as the basis for predicting whether a particular TCP connection is carrying spam. Hao et al. [10] used mostly lower-than-transport traffic features and in particular found that properties of a single SYN packet from an incoming SMTP connection can effectively identify spam. These "content-blind" techniques are attractive because they leverage properties that are hard to manipulate and can help discard spam quickly and at less computational cost. This previous work raises two sets of pertinent questions:

First, Hao et al. [10] show that single-packet features are effectively detect spam using models developed via a global email reputation service with about 2,500 subscribing institutions which provides for a diverse vantage point. However, do these findings hold for a stand-alone organization that does not subscribe to a global service and hence has a relatively narrow vantage point? Further, while Hao et al. studied their classifiers as a *replacement* for blacklists, most enterprises will likely use such classifiers *in addition to* blacklists. Will packet features still be effective in this case?

Second, Beverly and Sollins [4] find that transport-level features are effective in spam detection while Hao et al. [10] arrive at a similar conclusion regarding single-packet features. Further, in our prior work [13], we observed that though flow features are useful in discriminating ham from spam, examination of the resulting classifiers indicates that in many cases they potentially serve as "proxies" for features that could be computed from a single packet. Given that packet-level features allow one to discard spam more quickly, a key question is whether the more expensive "flow features"—requiring multiple packets—add discriminating power to the packet features. Hao et al. [10] consider a similar question with regard to their packet and message feature sets but do not focus on flow features.

In this paper, we evaluate these questions with a seven month dataset of emails to users at the International Computer Science Institute (ICSI). We develop three key findings. First, neither single-packet features nor flow features by themselves are effective classifiers at the enterprise level. In particular, packet features are much less effective than suggested in [10]. We identify underlying causes, one of which points to fundamental limitations of single packet features for spam detection. Second, while we find that neither single-packet nor flow features are operationally useful by themselves, we find their effectiveness increases when combined, indicating that flow features capture relevant discriminating information beyond packet features. However, even the combination is not as accurate in our setting as reported in prior work. Finally, the above results hold for two methods we used to analyze the data, giving a preliminary indication that these results are independent of the choice of the analysis method and reflect the underlying discriminating power of the features in question.

2 Data and Features

Our dataset includes all incoming email to ICSI from the 11^{th}–18^{th} of each month over 7 months. We work from packet traces with full packet contents. Some of these connections are blocked by the DNS blacklists, for which the SMTP transaction is terminated before email content is transmitted to the monitored servers. We exclude these connections from our analysis, except where noted. We use *Bro* [14] to re-construct the messages and derive some of the features. We additionally use *SpamFlow* [4] and custom tools to derive certain traffic features. The overall characteristics of our data are given in Table 1. A more in-depth description of our methodology is given in [13].

Ground Truth. We cannot manually classify messages in our dataset due to both the scale and the sensitivity of dealing with real users' email. Therefore, we developed an automated procedure to label the messages (as fully developed in [13]). Each message is processed by four content-based spam filters—SpamAssassin [2], SpamProbe [5], SpamBayes [12] and CRM-114 [1]. With the exception of SpamAssassin—which is used in non-learning mode—these tools are trained using the 2007 TREC email corpus [6]. A message is considered spam if any one of the tools flags it as such. Checking all the ham messages in the corpus involving the fourth author as well a 2% sample of email marked as spam by at least one of tools reveals this process yields the correct

	May	**Jun**	**Jul**	**Aug**	**Sep**	**Oct**	**Nov**
Msgs.	279K	302K	317K	292K	300K	223K	249K
Outbound	41K	38K	49K	54K	46K	37K	43K
DNSBL	165K	185K	174K	165K	172K	116K	105K
Unknown	11K	21K	31K	20K	24K	12K	10K
No Msg.	9K	7K	8K	6K	7K	7K	7K
Other	5K	8K	8K	5K	7K	9K	8K
Spam	30K	26K	30K	26K	27K	25K	55K
Ham	18K	18K	18K	15K	17K	17K	21K

Table 1. Data overview: The first row shows the total number of email messages, rows 2–6 show messages removed from the analysis and the last two rows give the number of hams and spams.

classification in 98% of the cases with a false positive rate of 1.23% (standard deviation 0.11%), and a false negative rate of 0.55% (standard deviation 0.07%). We evaluated a majority voting scheme as well, but that procedure was found to be less effective [13].

Packet Features. The upper part of Table 2 lists the single packet features we use. The geoDistance, senderHour, AS-Spamminess, and NeighborDist features are used in [10] although we derive the last three differently, as follows. We do not translate sender's hour into the ratio of ham to spam that were sent during that hour, because sender's hour itself is a numeric feature directly suitable for inclusion in our models. Hao et al. use the AS number directly as a numeric feature in their work. However, AS numbers are individual labels which do not lend themselves to meaningful aggregation in models (e.g., just because ASes 3 and 12 show some common behavior does not mean that ASes 4–11 share that behavior). Further, if treated as discrete values, the number of distinct AS values is problematic for classification methods. So we translate sender's AS number into a numerical value that reflects the prevalence of spam originating in the AS. The value for this feature is derived by using all messages in a training sample to develop a database of the "spamminess" of an AS. If a test message came from an AS that did not occur in the training set, we assign the average spamminess over all ASes as the value of this feature for that message.

To calculate the neighbor distance, NeighborDist, Hao, et al. first split their dataset into 24-hour bins. The NeighborDist is then the average distance to the 20 nearest IPs among preceding senders in the same bin, or among all the neighbors if there are fewer than 20 preceding senders [10, 9]. This procedure is not suitable for our enterprise environment because a one-day bin does not provide enough email to accumulate enough history. Further, since the database is smaller, boundary effects due to insufficient number of neighbors in the beginning of each bin influence the results greatly. This is illustrative of our first contribution (discussed in more detail below): a single edge network's myopic view thwarts development of accurate models. To mitigate this effect, we build IP databases using an entire training sample—consisting of 9/10 of the data for each month, given we use 10-fold cross validation. We then use this database to produce NeighborDist values for the training and test data. Note that because of our procedure, each fold of our experiments uses different databases for the AS-Spamminess and NeighborDist features. We refer to these two features as "database features" below.

Feature	Description
geoDistance^H	The geographical distance between the sender and ICSI, based on the MaxMind GeoIP database [11].
senderHour^H	The hour of packet arrival in sender's timezone.
AS-Spamminess^H	Num. of spams from AS divided by total msgs. from AS in the training set.
NeighborDist^H	Avg. numerical dist. from sender's IP to the nearest 20 IPs of other senders.
OS	OS of remote host as determined by *p0f* tool from SYN packet.
ttl	IP TTL field from SYN received from remote host.
ws	Advertised window size from SYN received from remote host.
3whs^B	Time between the arrival of the SYN from the remote host and arrival of ACK of the SYN/ACK sent by the local host.
fins_local^B	Number of TCP segments with "FIN" bit set sent by the local mail server.
fins_remote^B	Number of TCP segments with "FIN" bit set received from the remote host.
idle^B	Maximum time between two successive packet arrivals from remote host.
jvar^B	The variance of the inter-packet arrival times from the remote host.
pkts_sent / pkts_recvd	Ratio of the number of packets sent by the local host to the number of packets received from the remote host
rsts_local^B	Number of segments with "RST" bit set sent by the local mail server.
rsts_remote^B	Number of segments with "RST" bit set received from remote host.
rttv	Variance of RTT from local mail server to remote host.
rxmt_local^B	Number of retransmissions sent by the local mail server.
rxmt_remote^B	Approximate number of retransmissions sent by the remote host.
bytecount	Number of non-retransmitted) bytes received from the remote host.
throughput	*bytecount* divided by the connection duration.

Table 2. Message features. Features marked with H and B are from [10] and [4], respectively.

Hao et al. also use a feature that requires port scanning the sending IP. This is operationally problematic as it is time consuming and may trigger security alarms on remote hosts. Further, while we have historical packet trace data, we do not have historical port scanning data and mixing current port scans with historical packet traces would be a dubious experimental procedure. While we deleted or modified several single packet features we also added several features: senders' OS, IP's residual TTL, and TCP's advertised window (from prior work [4, 13]).

Flow features. The lower part of Table 2 shows the set of flow features we use to describe messages. This list is not identical to that used in prior work [4] (with common features tagged with a B)). We added several features we believe may help discriminate ham from spam. In addition, we removed three features: *packets* (in each direction), *cwnd0* and *cwndmin*. The number of packets is closely related to the *bytecount* and *pkts_sent/pkts_recvd* features in our list. A more detailed description of these features can be found in [13].

3 Empirical Evaluation

We use two algorithms in our experiments: decision trees [16] (from Weka [18]) and Rulefit [7]. Decision trees use the idea of *recursive partitioning*: at each step, they

	Rulefit			Decision Trees		
Month	**Acc**	**TPR**	**AROC**	**Acc**	**TPR**	**AROC**
May	0.674	0.483	0.944	0.663	0.464	0.783
Jun	0.573	0.280	0.926	0.564	0.266	0.785
Jul	0.555	0.299	0.940	0.563	0.312	0.805
Aug	0.580	0.338	0.940	0.543	0.280	0.773
Sep	0.560	0.279	0.933	0.586	0.322	0.783
Oct	0.609	0.353	0.938	0.640	0.406	0.779
Nov	0.504	0.315	0.904	0.507	0.319	0.660

Table 3. Results for packet features with AS-spamminess. The TPR is reported at 1% FPR.

choose a feature and use it to split the data until a partition only has examples from a single class. Rulefit—used in [10]— constructs a linear classifier that uses the primitive features as well as Boolean tests on feature values as predictors. We perform 10-fold stratified cross validation on each month's data by randomly dividing the trace at the granularity of SMTP sessions into ten folds such that all folds have the same spam/ham ratios as the entire dataset. We train our models using every set of nine folds and test it on the remaining folds. Throughout the paper, we consider spam as the target class; thus our true positive rate (TPR) is the fraction of spam classified as spam and false positive rate (FPR) is the fraction of ham misclassified as spam. We report (averaged over ten folds) accuracy, the area under ROC (AROC) [15], an alternative quality measure, and TPR at a given FPR (0.2% unless stated otherwise), obtained from the ROC graph for the classifier. The ROC, or receiver operating characteristic, graph relates the TPR and FPR as a threshold is varied over the confidence of the predictions.

3.1 Packet-Level Features at the Enterprise

Prior work [10] reports that properties of the first SYN packet from an incoming SMTP connection could be sufficient to filter out 70% of spam with 0.44% false positives (0.2% when adding features beyond single-packet). However, this result was obtained using models derived from 2,500 organizations, in a pre-blacklist setting. Our first question is whether single-packet features could be similarly effective in a stand-alone enterprise mail service using only its own vantage point. To study this, we run decision trees and Rulefit on each month's data, using only the single packet features (in the upper part of Table 2) to describe each message.

We first observed that the unmodified algorithms we used had an average FPR of almost 20% across the months (18.74% for decision trees and 19.83% for Rulefit), and our attempts to reduce FPR by thresholding the confidence degraded the TPR to single-digit percentages. This is clearly not operationally usable. To remedy this, we produce cost sensitive classifiers, trained to penalize FP errors more than FN errors. We use a cost ratio of 175:1 in our experiments for decision trees and 20:1 for Rulefit. The results are shown in Table 3. While the FPR drops significantly compared to the unweighted case, it does not reach 0.2% for decision trees—even when increasing the cost ratio significantly. Analysis of the classifiers reveals that the AS-spamminess feature—from the training data—is chosen as highly predictive in each case. However, it appears to be less predictive on the test data, possibly because even using 90% of our data from a month does not result in a comprehensive database. Thus, even though our results confirm previous findings (e.g., [17]) that AS origin differs for spam and ham, we find the utility of this feature in an enterprise environment limited.

Next, we remove AS-spamminess from the analysis, forcing our classifiers to use other features. The results are shown in the "Rulefit" and "Decision Trees" columns of Table 4. In this case, we are able to obtain an FPR of 0.2%. We further observe that though the classifiers produced achieve a low FPR, their TPR is low as well—as low as 10% and never reaching beyond 36%. This is significantly different from the result in prior work [10], where a TPR of 70% was achieved at FPR=0.44%, as we discussed above (while we omit our full results for 0.44% FPR due to space, Rulefit produced 29% TPR in May and at best 16% in other months; without AS-spamminess, the results improved to 47% in May and 18-31% in other months). Further, the result in [10] was achieved with unweighted Rulefit by purely thresholding confidence [9], while as mentioned earlier, this did not produce usable results in our setting. Thus it generally appears that packet features are significantly less useful in our setting. Further we observe that both decision trees and Rulefit exhibit quite similar results in our experiments. Thus the lack of utility of the single packet features is (at least to some extent) not a function of the learning algorithm they are used with, but purely a consequence of the limited information they convey about the message in this setting.

One might wonder if it is possible to increase the TPR or decrease the FPR further. However, this is difficult with just packet features to describe each message. It is intuitively plausible that looking at a single packet reveals limited information about the message, and one can only construct few features from this information (we use seven). This set of features generally describes the characteristics of a group of hosts sending email. But unless the granularity is extremely fine, such a group will sometimes send ham and sometimes spam. We therefore found many instances where messages were described by identical packet features but had opposing labels, i.e., some were labeled as spam and some as ham. For example, in the May data, 4K messages out of 48K total had a counterpart with the identical feature values but opposite label. This clearly is problematic for any classifier and may increase the FPR and lower the TPR. On the other hand, if granularity is decreased (e.g., imagine using the IP address—with a range of 4 billion values—as a feature), then significantly more data will need to be collected to train a useful classifier. This appears to be a fundamental limit of single packet features for spam detection.

Hao et al.'s work evaluated the utility of packet features in a *pre*-DNS blacklist setting. Although using such a blacklist is natural and common in an enterprise setting, we perform a similar analysis on our data to establish a direct comparison. This experiment follows Hao et. al.'s methodology–using the same classifier (Rulefit) and reports TPR at the same FPR (0.44%)—except it employs the cost-sensitive classifier with cost ratio of 100:1, as lower cost ratios degraded FPR to non-usable levels. In this case, we include messages that were blocked by ICSI's operational DNS blacklist setup ("DNSBL"+"Ham"+"Spam" in Table 1) since their first SYN packet is still available for analysis. We label all messages blocked by the DNS blacklist as spam. In this pre-blacklist experiment we train and evaluate on all messages including those blocked

	Post-Blacklist(Rulefit)			Post-Blacklist(Decision Trees)			Pre-Blacklist (Rulefit)		
Month	**Acc**	**TPR**	**AROC**	**Acc**	**TPR**	**AROC**	**Acc**	**TPR**	**AROC**
May	0.600	0.359	0.909	0.594	0.348	0.768	0.824	0.808	0.967
Jun	0.492	0.139	0.882	0.454	0.072	0.652	0.346	0.290	0.964
Jul	0.478	0.171	0.906	0.438	0.108	0.666	0.467	0.421	0.962
Aug	0.506	0.216	0.905	0.479	0.172	0.660	0.714	0.691	0.969
Sep	0.509	0.189	0.892	0.497	0.169	0.663	0.644	0.612	0.967
Oct	0.515	0.191	0.901	0.488	0.145	0.683	0.520	0.462	0.957
Nov	0.421	0.197	0.884	0.436	0.218	0.826	0.709	0.670	0.943

Table 4. Results for packet features *without* AS-spamminess. The TPR is reported at 0.2% FPR for the post-blacklist and at 0.44% for the pre-blacklist experiments.

by the DNS blacklist (the results are shown in the "Pre-blacklist" column in Table 4), while in all other experiments, we train and evaluate on messages that passed through and have not been filtered out by the DNS blacklist. We observe that the "Pre-blacklist" results vary significantly across months. For some months, our results are comparable to, even exceed, the 70% TPR reported by Hao et al. However, for other months, we find the TPR is far lower (e.g., 29% in June). Such variability reduces the operational utility of packet features for spam detection. While the reason for the variability is unclear, one possibility is that it is due to the enterprise's vantage point. Another possibility is that it is a fundamental property of packet features, and was not observed by Hao et al. since their data was collected over a period of only 14 days.

3.2 Effect of Flow-Level Features

Given the limited accuracy of packet features, we next consider whether adding flow-level features to the message description adds value by making the classifier more accurate. To check this, we compare two settings. First, we use only the flow features in the lower half of Table 2 to classify messages. Then we add the flow features to the packet features (without AS-spamminess) and use the full set in the same way. For these experiments, we use decision trees as the learning algorithm. This is because our prior results show that any patterns we see generalize to Rulefit as well, yet Rulefit is significantly more expensive to run (a cross validation runtime of hours, compared to a few seconds for trees). Further, trees are easier to interpret. The results are shown in Table 5.

Comparing these results to the results in Table 4, we observe that using the flow features by themselves improves TPR at a given FPR as against using the packet features by themselves. (In other experiments, not shown here, we also found that including AS-spamminess in the packet features reversed this trend, so that packet features had a higher TPR at a given FPR; however, as stated above, in that case the comparison was made at 1% FPR since the packet features with AS-spamminess do not achieve 0.2% FPR.) Further, we observe that using all of the features generally achieves a higher TPR at a given FPR as compared to either feature set on their own. This indicates that these two feature sets capture different kinds of information about packet traces.

Even though our results show that flow features can improve spam detection rates in conjunction with packet features in our post-blacklist, enterprise setting, the absolute

	Flow Features			All Features		
Month	**Acc**	**TPR**	**AROC**	**Acc**	**TPR**	**AROC**
May	0.583	0.330	0.709	0.632	0.410	0.759
Jun	0.555	0.244	0.689	0.546	0.230	0.686
Jul	0.547	0.281	0.701	0.564	0.308	0.748
Aug	0.538	0.266	0.691	0.576	0.327	0.754
Sep	0.558	0.270	0.681	0.582	0.309	0.711
Oct	0.488	0.145	0.633	0.529	0.215	0.675
Nov	0.417	0.191	0.823	0.449	0.236	0.788

Table 5. Results for decision trees using flow features (left), all features (without AS-spamminess) (right). TPR is reported at 0.2% FPR.

	Use one feature			Use all but one feature		
Feature	**Accuracy**	**TPR**	**AROC**	**Accuracy**	**TPR**	**AROC**
All single-packet	0.594	0.348	0.768	0.594	0.348	0.768
geoDistance	0.473	0.153	0.667	0.546	0.271	0.746
senderHour	0.378	0(FPR=0)	0.500	0.595	0.35	0.769
NeighborDist	0.378	0(FPR=0)	0.519	0.522	0.233	0.704
OS	0.378	0(FPR=0)	0.500	0.597	0.353	0.770
ws	0.38	0.004	0.622	0.596	0.352	0.768
ttl	0.378	0(FPR=0)	0.500	0.564	0.30	0.721

Table 6. Left: Average accuracy, TPR at 0.2% FPR and AROC when only one single-packet feature is used. Right: Average accuracy, TPR and FPR when one single-packet feature is left out.

TPRs do not reach the 70% found in the setting of prior work [10]. A question is whether an effective pre-filter can be constructed using such a classifier, so that only messages that cannot be classified with high probability are sent to computationally expensive content filters. We are currently investigating this question.

3.3 Utility of Individual Features

Next, we examine which packet and flow features are most useful in discriminating between ham and spam. We consider three situations. First, we look at the accuracy of our classifiers when only a single packet feature is used, and when we leave a single packet feature out from all the packet features (results in Table 6). Next, we start with the full set of packet features and add flow features one at a time to determine the value added by each flow feature. Finally, we look at what happens if we start with the full feature set and leave one flow feature out (results in Table 7).

Table 6 shows *geoDistance* to be the most useful packet feature. The other packet features, when used in isolation, result in zero or near zero TPR. This is because they produce an empty (or nearly empty in the case of *ws*) tree that always predicts "ham" in order to minimize the cost with a high FP cost. Further, only *geoDistance* and *Neighbor-Dist* result in large drops in TPR when they are left out of the feature set. This indicates that though *NeighborDist* is not useful by itself, it has some discriminating power when

	Use one feature			Use All but one features		
Feature	**Accuracy**	**TPR**	**AROC**	**Accuracy**	**TPR**	**AROC**
All packet and flow features	0.632	0.410	0.759	0.632	0.410	0.759
All single-packet features	0.594	0.348	0.768	0.594	0.348	0.768
3whs	0.586	0.336	0.720	0.643	0.427	0.784
fins_local	0.611	0.376	0.772	0.615	0.382	0.729
fins_remote	0.598	0.354	0.770	0.628	0.403	0.759
idle	0.597	0.353	0.770	0.633	0.411	0.759
jvar	0.594	0.348	0.771	0.632	0.410	0.759
pkts_sent/pkts_received	0.599	0.356	0.770	0.618	0.386	0.759
rsts_local	0.595	0.349	0.768	0.632	0.410	0.759
rsts_remote	0.595	0.349	0.770	0.623	0.395	0.788
rttv	0.606	0.367	0.770	0.618	0.386	0.751
rxmt_local	0.596	0.351	0.768	0.629	0.404	0.759
rxmt_remote	0.595	0.350	0.769	0.624	0.396	0.751
bytecount	0.598	0.354	0.768	0.632	0.409	0.759
throughput	0.607	0.369	0.793	0.625	0.398	0.729

Table 7. Left: Average accuracy, TPR at 0.2% FPR and AROC when all single-packet features and only one flow feature are used. Right: Results when one flow feature is left out, and all other flow and single-packet features are used.

used in conjunction with other packet features. For the other packet features, the TPR stays the same or increases slightly when they are dropped, indicating that they do not provide much added value beyond the remaining features. In particular, our results do not show *senderHour* to be useful for spam detection despite previous findings that spammers and non-spammers display different timing properties [8].

From the results in Table 7, we observe that adding any one flow feature to the set of single packet features improves the performance of the classifier, though by a small margin. In particular, *fins_local*, *rttv* and *throughput* provide the greatest additional discriminating power beyond the single packet features. Further, *fins_local*, *rttv* and *pkts_local/pkts_remote* result in the largest drops in performance when they are dropped, indicating that they are also useful in conjunction with the other features (i.e. no other feature captures their information). Some flow features such as *rsts_local* and *3whs* either do not change the TPR or increase it when they are dropped, indicating that they do not provide much added value beyond the remaining features. This contradicts prior results [4, 13] that found that *3whs* was the most useful in discriminating ham from spam, if *only* flow features were used. However, it appears that the information in *3whs* is subsumed by the other features, perhaps by packet features such as *geoDistance*.

4 Summary

This paper addresses two questions: whether an organizational mail server can detect a sizable amount of spam based on the first packet of an incoming connection, and the relative effectiveness of single-packet and flow features in detection. Our primary finding indicates that from an organizational perspective, single-packet features are much less effective than was observed in prior work. “Database” features such as AS-spamminess and sender’s neighborhood density are less effective in this situation, and the limited information conveyed by packet features leads ambiguity and hence non-useful models. Also, adding flow features to packet features improves accuracy, but the net effect is still modest.

Some questions still remain. While we find that network features may not be useful in the enterprise setting, it would be useful to study other such organizations to strengthen our findings. Finally, though network features only filter 20-40% of post-blacklist spam, avoiding content-based processing of these messages may still be a net win for mail servers, which we are quantifying in ongoing work.

Acknowledgments: Robert Beverly provided his SpamFlow tool. Shuang Hao and Nick Feamster clarified the methodology used in [10]. This work was funded in part by NSF grants CNS-0916407, CNS-0831821 and CNS-0433702.

References

1. Crm114 - the controllable regex mutilator. `http://crm114.sourceforge.net/`.
2. SpamAssassin. `http://spamassassin.apache.org/`.
3. E. Allman, J. Callas, M. Delany, M. Libbey, J. Fenton, and M. Thomas. Domain-Based Email Authentication Using Public Keys Advertised in the DNS (DomainKeys), May 2007. RF 4871.
4. R. Beverly and K. Sollins. Exploiting Transport-Level Characteristics of Spam. In *5th Conference on Email and Anti-Spam*, Aug. 2008.
5. B. Burton. SpamProbe. `http://spamprobe.sourceforge.net/`.
6. G. Cormack and T. Lynam. 2007 TREC Public Spam Corpus. `http://plg.uwaterloo.ca/~gvcormac/treccorpus07/`.
7. J. H. Friedman and B. E. Popescu. Predictive learning via rule ensembles. *Annals of Applied Statistics*, 2(3):916–954, 2008.
8. L. H. Gomes, C. Cazita, J. M. Almeida, V. A. F. Almeida, and W. M. Jr. Characterizing a spam traffic. In *IMC*, pages 356–369, 2004.
9. S. Hao and N. Feamster. Personal Communication, 2010.
10. S. Hao, N. A. Syed, N. Feamster, A. G. Gray, and S. Krasser. Detecting spammers with SNARE: Spatio-temporal network-level automatic reputation engine. In *Usenix Security Symp.*, 2009.
11. MaxMind. http://www.maxmind.com.
12. T. Meyer and B. Whateley. SpamBayes: Effective Open-Source, Bayesian Based, Email classification System. In *Proc. First Conference on Email and Anti-Spam*, June 2004.
13. T. Ouyang, S. Ray, M. Allman, and M. Rabinovich. A Large-Scale Empirical Analysis of Email Spam Detection Through Transport-Level Characteristics. Technical Report 10-001, International Computer Science Institute, Jan. 2010.
14. V. Paxson. Bro: A System for Detecting Network Intruders in Real-Time. In *Proceedings of the 7th USENIX Security Symposium*, Jan. 1998.
15. F. Provost, T. Fawcett, and R. Kohavi. The case against accuracy estimation for comparing induction algorithms. In *15th Int. Conf. on Machine Learning*, pages 445–453, 1998.
16. J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
17. A. Ramachandran and N. Feamster. Understanding the Network-Level Behavior of Spammers. In *ACM SIGCOMM*, 2006.
18. I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, second edition, 2005.