Spam Sender Detection with Classification Modeling on Highly Imbalanced Mail Server Behavior Data

Yuchun Tang, Sven Krasser, Dmitri Alperovitch, Paul Judge
Secure Computing Corporation
4800 North Point Parkway, Suite 300
Alpharetta, GA 30022, USA
Email: {ytang, skrasser, dalperovitch, pjudge}@securecomputing.com

***Abstract*— Unsolicited commercial or bulk emails or emails containing viruses pose a great threat to the utility of email communications. A recent solution for filtering is reputation systems that can assign a value of trust to each IP address sending email messages. By analyzing the query patterns of each node utilizing reputation information, reputation systems can calculate a reputation score for each queried IP address. In this research, we explore a behavioral classification approach based on features extracted from such global messaging patterns. Due to the large amount of bad senders, this classification task has to cope with highly imbalanced data. Firstly, for each observed sender, we calculate periodicity properties using a discrete Fourier transform and global breadth information reflecting message volume and recipient distribution. After that, a Granular Support Vector Machine - Boundary Alignment algorithm (GSVM-BA) is implemented to solve the class imbalance problem and compared to cost sensitive learning. Lastly, we determine the performance of support vector machine, C4.5 decision trees, naïve Bayesian decision trees, and multinomial logistic regression classifiers on the resulting data set. The best performance is observed by using GSVM-BA for rebalance and then using SVM for classification.**

I. Introduction

Traditional content filtering anti-spam systems can provide highly accurate detection rates but are usually prohibitively slow and poorly scalable to deploy in high-throughput enterprise and ISP environments. An early approach for efficient filtering of unwanted spam email traffic has been the use of real-time blacklists (RBLs). An RBL is a service containing a list of IPs from which email servers should not accept messages. Typically, RBLs can be queried over the DNS protocol. Whenever a host on the Internet tries to send a message to an email server, the email server can query an RBL to check whether the IP of the sending host is listed. If the IP is listed then the sending host is a known source of malicious traffic and the receiving email server can reject the message. Alternatively, the information that the sender is listed can be combined with a local analysis result to make a final decision.

RBLs generally receive information about malicious senders from spam messages delivered to spamtrap email addresses, manual listings, or user feedback. There are several drawbacks to this approach. To be automatically listed, a particular spam run needs to target a spamtrap email address. All messages that have been delivered to other addresses beforehand cannot be stopped. Manual listings and user feedback require a human in the loop and are inherently non-automatic. This results in a slow reaction time, which is especially problematic since most machines sending spam are zombie machines with only a few hours of sending activity.

One approach to counter these shortcomings is to take more sources of feedback into account. Especially the real-time queries sent over DNS can yield additional insight. Fig. 1 shows the data that can be gathered from such a query. A sending host with IP address Q tries to send a message to a receiving email server. The email server queries an RBL for the IP address of the sending host (Q), which we therefore call the *queried IP*. The query is relayed over DNS to an RBL server, which will see the query packet coming from the IP address S of the DNS server, which we call the *source IP*. In addition, the time T the query has been received is stored. This results in a tuple $< Q, S, T >$ generated by every query.

Ramachandran *et al.* analyze the source IPs querying an RBL [1]. They investigate a dataset of RBL queries to spot exploited machines (zombies) in botnets that are sending queries to test whether members of the same botnet have been blacklisted. Therefore, additional information can be obtained with regard to the maliciousness of the source IP.

Ramachandran *et al.* propose another approach that is able to detect malicious IPs if information on the destination domain is available by looking at the distribution of domains to which a sending IP sends email [2].

However, such classifiers cannot always make a definite decision. Introducing a continuous reputation value in contrast to a discrete yes/no decision allows the user of such a system to define his or her own thresholds. More importantly, this continuous value can be used as a feature in a local classification engine.

One such reputation system is Secure Computing's TrustedSource [3]. TrustedSource operates on a wide range of data sources including proprietary and public data. The lat-

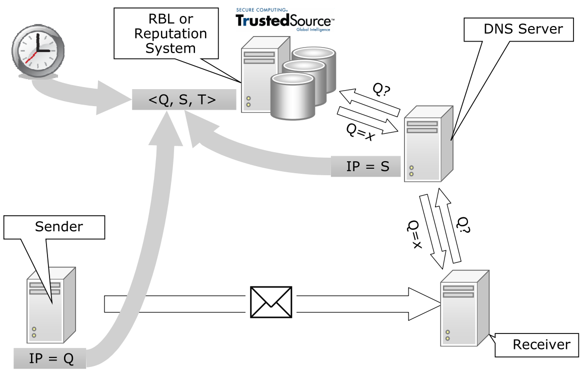


Fig. 1. Data gathered from query.

ter includes public information obtained from DNS records, WHOIS data, or real-time blacklists (RBLs). The proprietary data is gathered by over 5000 IronMail appliances that give a unique view into global enterprise email patterns. Based on the needs of the IronMail administrator, the appliance can share different levels of data and can query for reputation values for different aspects of an email message. Currently, TrustedSource can calculate a reputation value for the IP address a message originated from, for URLs in the message, or for message fingerprints. This research uses queries for IPs as input to detect spam sender IPs among them.

In previous work, we outlined the detection of spam senders based on query patterns [4]. This approach allows gaining additional information on the queried (sending) IP by aggregating query patterns into spectral and breadth features with a focus on spectral features.

In this paper, we focus on the extraction of breadth features. In addition, we compare the performance of GSVM-BA we proposed previously with four standard algorithms.

The rest of the paper is organized as follows. Section II describes the data we use for classification modeling. Section III gives a brief review of research on imbalanced classification. In Section IV, GSVM-BA is presented in detail as a solution to the class imbalance challenge. Section V compares performance between GSVM-BA and cost-sensitive learning with four classification algorithms on IP classification. Finally, Section VI concludes the paper.

II. Query Data Analysis

TrustedSource reputation system provides an RBL-style interface to retrieve reputation values for IP addresses over DNS. While TrustedSource allows other means to be queried that can include additional information such as message fingerprints or email header metadata, we limit this classification problem to $< Q, S, T >$ tuples (or *QST data* in short) since this type of data is the most generic one and applies also to standard RBLs widely used on the Internet today.

A. Sources of Noise

Generally, an IP that is queried has sent a message shortly before the query has been logged (at time T). Note that this assumption does not always hold. IPs can also be queried by hand by humans trying to find information regarding a listing. Also, as noted previously, some zombie machines try to automatically send queries to evade IP-based blocking mechanisms [1]. Zombies could also attempt to actively poison the data feed by submitting bogus queries. Lastly, not all messages result in queries since DNS results can be cached. This can occur even when an RBL server specifically returns low time-to-live (TTL) values with its responses due to DNS servers relaying these responses not honoring the TTL values. Normally, a well designed classifier will be able to still yield useful results in the presence of such noise, but there are a number of techniques to avoid these problems. These include moving away from DNS (not desirable for RBLs since DNS queries are an accepted standard and widely supported), adding a unique string to the query (to prevent caching, needs to be supported by both email server client and RBL server), and adding authentication information (to prevent data poisoning).

B. Feature Extraction

For this research, we are interested in classifying email senders based on their sending behavior. For each sender observed in the data set (i.e. each queried IP), we calculate a number of features based on the distribution of source IPs (S) and timestamps (T). These fall into two general categories: spectral features and breadth features.

1) Spectral Features: Spectral features are based on the distribution of the set of timestamps observed for each queried IP. Currently, we do not take source IP information into account for this set of features. Under the assumptions outlined previously, each timestamp seen from a particular IP Q_i corresponds to a message sent from Q_i. We consider a time interval ΔT that we split up in N equally sized intervals Δt_n where $n = 0 \ldots (N-1)$. The number of timestamps falling into interval Δt_n is denoted c_n, which corresponds to the total number of messages observed from Q_i in that interval. This sequence is then transformed into the frequency domain using a discrete Fourier transform (DFT). Since we do not consider time zones or time shifts, we are only interested in the magnitude of the complex coefficients of the transformed sequence and throw away the phase information. This results in the sequence C_k. C_0 is the constant component that corresponds to the total message count. The message count will be considered as part of the breadth features, and we will use it here only to normalize coefficients yielding a new sequence $C'_k = C_k/C_0$. Furthermore, since the input sequence c_n is real, all output coefficients C_k with $k > N/2$ are redundant due to the symmetry properties of the DFT and can be ignored.

We have chosen $\Delta T = 24\ h$ and $N = 48$ resulting in 30 minute time slots Δt_i. This results in 24 usable raw spectral features, C'_1 to C'_{24}. An evaluation of these raw features indicated that for ham senders there are three distinct groups. C'_1, C'_2 to C'_4, and C'_5 to C'_{24} all lie within specific ranges. We use C'_1, the mean and the standard deviation of C'_2 to C'_4, and the mean the standard deviation of C'_5 to C'_{24} as the first five spectral features. In addition, we add log-scaled versions of C'_1 to C'_{24} to the spectral feature set.

The selection of these features is mostly based on heuristics at this point, and we expect to be able to achieve a vastly improved performance after a careful analysis. Spectral features have been previously covered in our preliminary results presented in [4].

2) Breadth Features: Breadth features are also calculated based on data gathered in a 24 hour time window. In that window, the sending behavior of each queried IP Q is analyzed. First, the numbers of source IPs S querying Q is calculated, which we denote B_{src}. Under the assumptions made this count reflects the number of recipients that Q attempted to send email to. Since S is the IP of the DNS server relaying the query and multiple recipients may either use the same DNS server or one recipient may use multiple DNS servers, this count does not perfectly reflect the amount of recipients. It does however give a good idea about the breadth of recipients across the Internet. For the sake of brevity we refer to the source IP of a query (i.e. the IP address of the DNS server relaying that query) as the *virtual recipient*—keeping in mind that the actual recipient is not exposed in the QST data.

Second, the number of queries for each IP Q (and therefore the number of messages sent by that IP) denoted B_{msgs} is calculated. As previously explained, a major source of noise in this feature is due to caching.

Third, we introduce the notion of a *session*. A session is a 30 minute period in which an IP Q_i sends to a particular virtual recipient S_j. When S_j queries Q_i (i.e. when Q_i sends a message to the virtual recipient S_j) for the first time, a new session is counted. This session is valid for 30 minutes, and all additional queries involving $< Q_i, S_j >$ do not result in new sessions (while queries involving a different virtual recipient in the same time frame will). After 30 minutes the session is closed, and a query will result in a new session. All sessions for Q_i across all virtual recipients are summed up yielding the session count B_{ssn}. Note that this feature mimics the query pattern that would be observed if the relaying DNS server would cache all queries with a 30 minute TTL.

Fourth, we calculate the average number of messages per session per particular source IP, sum all averaged values up, and divide by the total number of sessions. We denote this feature as B_{mpss}.

Fifth, we calculate the number of global sessions. A global session is akin to a regular session but does not distinguish between different virtual recipients and therefore reflects the global time of activity for each queried IP. We denote the global session count as B_{gssn}.

Lastly, we heuristically incorporate two derived features, $B_{\text{s/m}} = \frac{B_{\text{src}}}{B_{\text{mpss}}}$ and $B_{\text{s/s}} = \frac{B_{\text{ssn}}}{B_{\text{src}}}$.

The Signal-to-Noise ratio (S2N), defined as the distance of the arithmetic means of the spam and non-spam (ham) classes divided by the sum of the corresponding standard deviations [5], for each of these five features is presented in Table I. The S2N values show that these breadth features are informative and can be utilized for classification modeling. The B_{gssn} feature achieves the best ratio. The probability densities for the spam and ham classes of this particular feature are shown in Figure 2.

TABLE I
BREADTH FEATURE CHARACTERISTICS

	S2N	μspam	σspam	μ_{ham}	σ_{ham}
B_{src}	0.07	136.14	138.13	161.90	227.93
B_{ssn}	0.16	182.35	243.83	324.35	673.69
B_{gssn}	0.44	12.68	10.27	22.50	12.05
B_{msgs}	0.13	287.34	464.73	617.15	2108.90
B_{mpss}	0.09	3.01	19.08	1.03	3.20
$B_{\text{s/m}}$	0.21	0.65	0.23	0.55	0.25
$B_{\text{s/s}}$	0.24	1.17	0.32	1.78	2.23

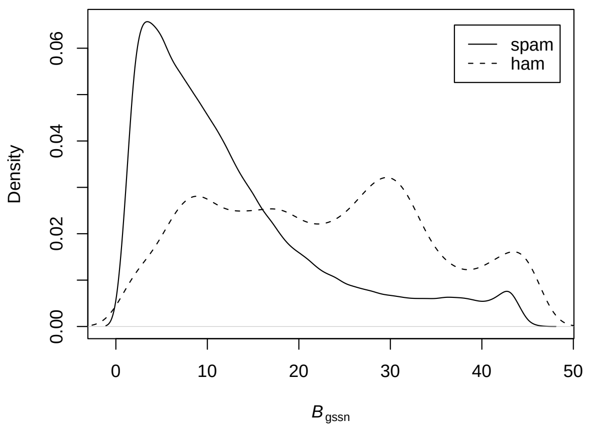


Fig. 2. Probability density for B_{gssn} for spam and ham senders.

III. IMBALANCED CLASSIFICATION

How to build an effective and efficient model on a huge and complex dataset is a major concern of the science of knowledge discovery and data mining. With emergence of new data mining application domains such as messaging security, e-business, and biomedical informatics, more challenges are arising. Among them, highly skewed data distribution has been attracting noticeably increasing interest from the data mining community due to its ubiquitousness and importance [6], [7].

A. Class Imbalance

Class imbalance happens when the distribution on the available dataset is highly skewed. This means that there are significantly more samples from one class than samples from another class for a binary classification problem. Class imbalance is ubiquitous in data mining tasks, such as diagnosing rare medical diseases, credit card fraud detection, intrusion detection for national security, etc.

For spam filtering, each IP is classified as spam or non-spam based on its sending behavioral patterns. This classification is highly imbalanced, as shown in our experiments. As our current target in this research is to detect spam IPs, we define spam IPs as positive and non-spam IPs as negative.

B. Methods for Imbalanced Classification

Many methods have been proposed to handle imbalanced classification, and some good results have been reported [7]. These methods can be categorized into three different kinds: cost-sensitive learning, oversampling the minority class, or undersampling the majority class. Interested readers may refer to [8] for a good survey.

For a real world classification task like spam IP detection, there are usually a large amount of IP samples. These samples need to be classified quickly so that spam messages from those IPs can be blocked in time. However, cost sensitive learning or oversampling usually increases decision complexity and hence slows down classification. On the other hand, undersampling is a promising method to improve classification efficiency. Unfortunately, random undersampling may not generate accurate classifiers because informative majority samples may be removed.

In this paper, granular computing and SVM are utilized for undersampling by keeping informative samples while eliminating irrelevant, redundant, or even noisy samples. After undersampling, data is cleaned and hence a good classifier can be modeled for IP classification both in terms of effectiveness and efficiency.

C. SVM for Imbalanced Classification

SVM approximates the structural risk minimization principle that minimizes an upper bound on the expected risk [9], [10]. Because structural risk is a reasonable trade-off between the training error and the modeling complication, SVM has a great generalization capability. Geometrically, the SVM modeling algorithm works by constructing a separating hyperplane with the maximal margin.

Compared with other standard classifiers, SVM performs better on moderately imbalanced data. The reason is that only Support Vectors (SVs) are used for classification and many majority samples far from the decision boundary can be removed without affecting classification [11]. However, performance of SVM is significantly deteriorated on highly imbalanced data [11], [12]. For this kind of data, it is prone to find the simplest model that best fits the training dataset. Unfortunately, the simplest model is exactly the naïve classifier that identifies all samples as part of the majority class.

SVM is usually much slower than other standard classifiers [13], [14], [15]. The speed of SVM classification depends on the number of SVs. For a new sample X, $K(X, SV)$ is calculated for each SV. Then it is classified by aggregating these kernel values with a bias. To speed up SVM classification, one potential method is to decrease the number of SVs.

IV. THE GSVM-BA ALGORITHM

In this work, the *Granular Support Vector Machines-Boundary Alignment* algorithm (GSVM-BA) is designed and targeted at improving both effectiveness and efficiency for IP classification based on the principles of granular computing.

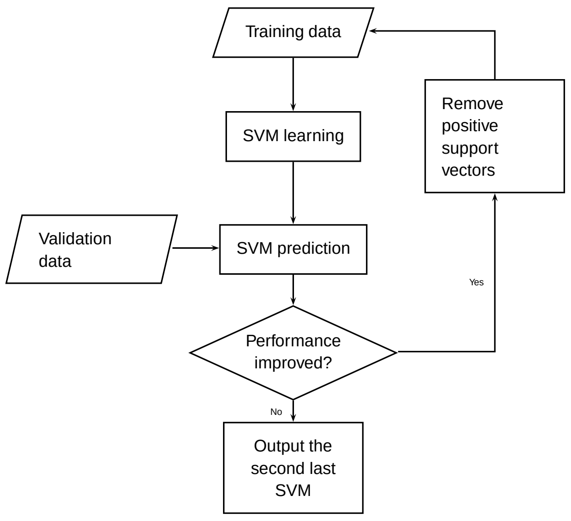


Fig. 4. Flow chart of the GSVM-BA algorithm.

A. Granular Computing and GSVM

Granular computing represents information in the form of some aggregates (called *information granules*) such as subsets, subspaces, classes, or clusters of a universe. It then solves the targeted problem in each information granule [16]. There are two principles in granular computing. The first principle is divide-and-conquer to split a huge problem into a sequence of granules (*granule split*); The second principle is data cleaning to define the suitable size for one granule to comprehend the problem at hand without getting buried in unnecessary details (*granule shrink*). As opposed to traditional data-oriented numeric computing, granular computing is knowledge-oriented [17]. By embedding prior knowledge or prior assumptions into the granulation process for data modeling, better classification can be achieved.

A granular computing-based learning framework called Granular Support Vector Machines (GSVM) was proposed in [18]. GSVM combines the principles from statistical learning theory and granular computing theory in a systematic and formal way. GSVM works by extracting a sequence of information granules with granule split and/or granule shrink, and then building an SVM on some of these granules when necessary. The main potential advantages of GSVM are:

1) GSVM is more sensitive to the inherent data distribution by trading off between local significance of a subset of data and global correlation among different subsets of data, or trading off between information loss and data cleaning. Hence, GSVM may improve the classification performance.
2) GSVM may speed up the modeling process and the classification process by eliminating redundant data locally. As a result, it is more efficient and scalable on huge datasets.

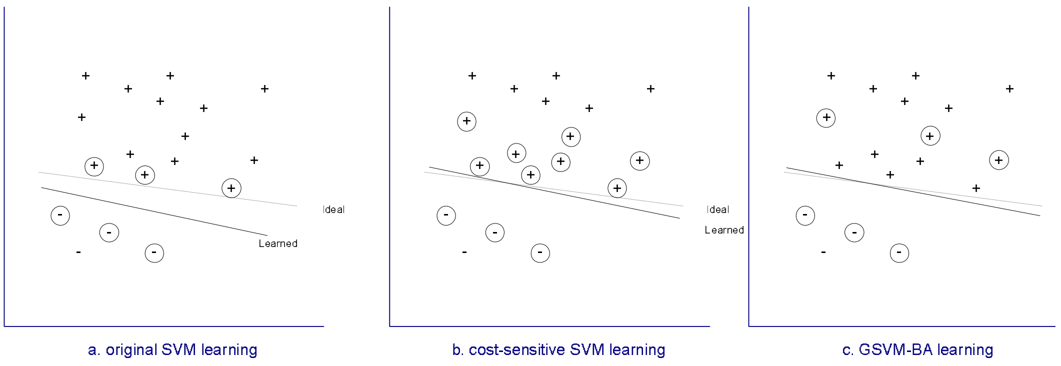


Fig. 3. GSVM-BA can push the boundary back close to the ideal position with fewer SVs.

B. GSVM-BA

Armed with the data cleaning principle of granular computing, GSVM-BA is ideal for spam IP detection on highly imbalanced data derived from QST data. SVM assumes that only SVs are informative to classification and other samples can be safely removed. However, for highly imbalanced classification, the majority class pushes the ideal decision boundary toward the minority class [11], [12]. As demonstrated in Fig. 3(a), positive SVs that are close to the learned boundary may be noisy. Some really informative samples may hide behind them.

To find these informative samples, we can conduct cost-sensitive learning to assign higher penalty values to false positives (FP) than false negatives (FN). This method is named SVM+CS. However, to counter the increased weights on the minority side, SVM+CS increases the number of majority SVs (Fig. 3(b)), and hence slows down the classification process.

In contrast to this, GSVM-BA looks for these informative samples by repetitively removing positive support vectors from the training dataset and rebuilding another SVM. GSVM-BA is knowledge-oriented in that it embeds the *boundary push* assumption (“prior knowledge”)into the modeling process. After an SVM is modeled, a *granule shrink* operation is executed to remove corresponding positive SVs to generate a smaller training dataset on which a new SVM is modeled. This process is repeated to gradually push the boundary back to its ideal location where the optimal classification performance is achieved.

Empirical studies in the next section show that GSVM-BA can compute a better decision boundary with much fewer samples involved in the classification process (Fig. 3(c)). Consequently, classification performance can be improved in terms of both effectiveness and efficiency. Fig. 4 sketches the GSVM-BA algorithm. The first SVM is always the naïve one by default.

V. Experiments

Classification modeling is carried out on a workstation with a Pentium M®CPU at 1.73 GHz and 1 GB of memory. The experiments are targeted at comparing the performance between GSVM-BA undersampling and cost-sensitive learning with different classification algorithms on highly imbalanced IP classification.

A. Data Preparation

We build classifiers on the daily based email server behavioral data, which are retrieved from over 7000 sensors located in 51 countries. We see several millions IPs every day. About 32% IPs can be labeled as spam or ham based on information from the real production system. By running the TrustedSource system over 5 years already, the labeling information is very reliable.

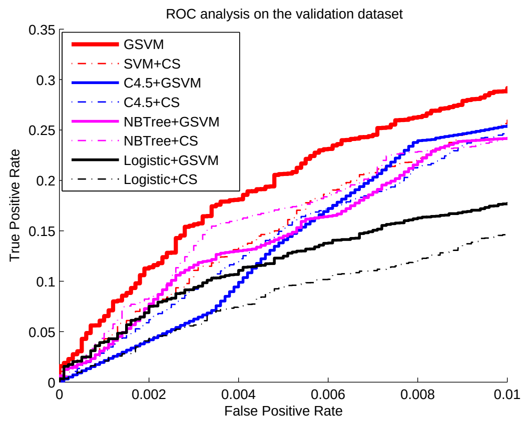
Although our inherent task is to build classifiers on these 32% known IPs to catch spam senders in the remaining 68% unknown IPs, we use QST data from labeled IPs for performance comparison. The QST data gathered on 08/14/2006 is used for training. The QST data gathered between 08/07/2006 and 08/13/2006 is used for validation. Data used for testing has been collected from 08/15/2006 to 09/11/2006. The validation dataset and the testing dataset pass through a stratified random undersampling process so that each of them has similar size to one day’s data. It is convenient for us to estimate classification efficiency in the real production system, in which QST features are retrieved from past 24 hours data. As shown in Table II, the datasets are highly imbalanced with over 93% of the IPs being spam IPs.

TABLE II
QST Data Distribution

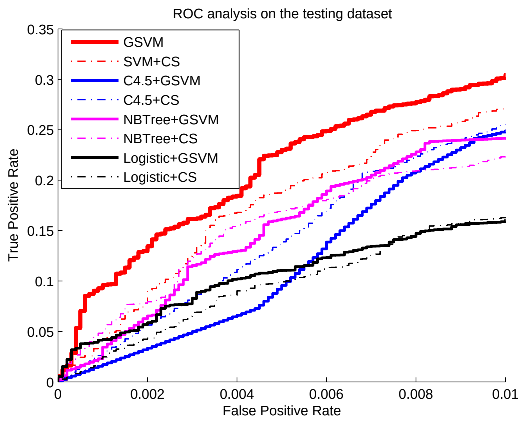
	spam	non-spam
training data	93.96%	6.04%
validation data	93.09%	6.91%
testing data	94.99%	5.01%

The training dataset is normalized so that the value of each input feature falls into the interval of [-1,1]. The validation dataset and the testing dataset are normalized correspondingly.

In the modeling phase, multiple algorithms are applied on the training dataset to build classifiers that are tuned to get the best performance on the validation dataset. The performance of these classifiers on the testing dataset is then reported.



(a) validation performance



(b) testing performance

Fig. 5. ROC analysis.

B. Evaluation Metrics

The performance evaluation is directly related to our spam detection application. To make the classification result reliable, it is required that fpr, as defined in (1), should be less than 1%. This threshold is decided based on the feedback from the technical support team. As they noticed, with fpr under 1%, the FP reports from our customers are controlled at a business satisfied level. In our modeling process, we decide to control the fpr at 0.8% on the validation dataset to be slightly more conservative. With this prerequisite in mind, we are also trying to increase the tpr, which is defined in (2).

$$fpr = FP/(FP + TN) \quad (1)$$

$$tpr = TP/(TP + FN) \quad (2)$$

C. Classification Algorithms

Four classification algorithms are used in this study.

1) The C4.5 algorithm for building a decision tree [19].
2) The NBTree algorithm for building a decision tree with naïve Bayes classifiers at the leaves [20].
3) The Logistic algorithm for building a multinomial logistic regression model with a ridge estimator [21].
4) The SVM algorithm for building a support vector machine [9].

These algorithms represent the state of the art in machine learning as well as the most popular and widely deployed classification solutions.

For the first three algorithms, we choose the Weka software package (available at `http://www.cs.waikato.ac.nz/ml/weka/`). We also choose LIBSVM (available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`) for SVM modeling with the RBF kernel. A cost-sensitive meta-learning strategy is utilized to handle class imbalance between spam IPs and non-spam ones. The misclassification cost for a FN is always 1 while the misclassification cost for a FP is tuned such that the fpr is close to 0.8% on the validation dataset.

As classification with SVM+CS is extremely slow, we designed GSVM-BA. A sequence of *granular shrink* operations is executed to recursively remove positive support vectors and thus to align the classification boundary until the fpr is close to 0.8% on the validation dataset.

For a thorough comparison between GSVM-BA and cost-sensitive learning, we also run C4.5, NBTree, and Logistic algorithms on the training dataset after GSVM-BA undersampling besides running them on the original training dataset.

D. Result Analysis

TABLE III
EFFECTIVENESS COMPARISON

	validation		testing	
	tpr%	fpr%	tpr%	fpr%
SVM+CS	22.51	0.83	25.55	0.88
GSVM-BA	**26.32**	**0.81**	**28.17**	**0.83**
C4.5+CS	20.87	0.79	20.36	0.68
C4.5+GSVM	23.92	0.80	24.13	0.93
NBTree+CS	22.42	0.73	20.61	0.74
NBTree+GSVM	19.39	0.72	19.29	0.61
Logistic+CS	12.20	0.81	13.19	0.72
Logistic+GSVM	16.29	0.81	17.87	1.22

TABLE IV
EFFICIENCY COMPARISON

	#SVs	validation (seconds)	testing (seconds)
SVM+CS	20151	8129	8571
GSVM-BA	**413**	**261**	**241**
C4.5+CS	N/A	13	13
C4.5+GSVM	N/A	13	13
NBTree+CS	N/A	22	20
NBTree+GSVM	N/A	21	19
Logistic+CS	N/A	15	14
Logistic+GSVM	N/A	15	14

Table III compares the effectiveness of the four classification algorithms, combined with the two rebalance techniques. For each combination, the tpr and the fpr are reported both on the validation dataset and on the testing dataset. Modeled on simple QST data, the classifiers demonstrate a significantly high tpr and are therefore effective to catch a large number of spam senders. With the fpr threshold in mind, GSVM-BA is the most effective algorithm with SVM+CS in second place.

Fig. 5 reports ROC analysis [22] results. Only curves with $fpr \leq 1\%$ are plotted so that the comparison is meaningful for the real system. GSVM-BA has the largest area under ROC curve. Moreover, the figures show that GSVM-BA has always the largest tpr for any fpr under 1%. The comparison between the validation performance and the testing performance shows that no overfitting happens.

Although being slower than C4.5, NBTree, or Logistic, GSVM-BA is much faster than SVM+CS for classification as demonstrated in Table IV. The reason is that GSVM-BA extracts much less (only 2%) SVs than SVM+CS. It takes about 4 minutes for classification. Considering the fact that it takes about 30 minutes to retrieve QST data in the real production system, the efficiency improvement is proved to be critical to decrease the response time to detect spam senders.

In our experiments, we observed similar modeling time for GSVM-BA and SVM+CS. For GSVM-BA, most of modeling time is consumed at the first few rounds of *granule shrink* operations. After that, it becomes much faster for SVM modeling to converge because the decision boundary becomes much clearer as the positive SVs are removed.

We do GSVM-BA modeling once every month (offline learning). And then we use the model for classification in the next month (online classification). So classification efficiency is critical while modeling efficiency is not in the real production system.

It has been running for one year and has demonstrated stable performance. For effectiveness, it can catch about 28% spam senders with less than 1% fpr. For efficiency, it only takes about 4 minutes to classify the latest samples.

VI. Conclusion

Four state-of-the-art classification algorithms are utilized to detect malicious email servers on low-informative and highly imbalanced QST data based on global messaging patterns. Based on simple QST records that do not appear to be informative at a first glance, the breadth and spectral features are extracted. Our study demonstrates that these features combined with advanced classification techniques can be reliable for spam detection if combined with methods to deal with class imbalance. SVM with cost-sensitive learning suffers from a long run time due to the large amount of support vectors, which is problematic for detecting malicious senders typically sending only for a few hours. To improve efficiency, GSVM-BA is proposed to handle class imbalance with undersampling by recursively eliminating positive support vectors and rebuilding another SVM. Our study demonstrates that GSVM-BA greatly speeds up the classification process by extracting much less support vectors. GSVM-BA is even more effective than SVM with cost-sensitive learning. At the same FP rate level acceptable for real application, GSVM-BA catches more spam email servers. The resulting classifier contributes to TrustedSource as one source of input and prevents malicious or unwanted messages being delivered to email users.

References

[1] A. Ramachandran, N. Feamster, and D. Dagon, “Revealing botnet membership using dnsbl counter-intelligence,” in *Proc. of 2nd USENIX Steps to Reducing Unwanted Traffic on the Internet*, pp. 49–54, 2006.

[2] A. Ramachandran, N. Feamster, and S. Vempala, “Filtering spam with behavioral blacklisting,” in *Proc. of the 14th ACM Conference on Computer and Communications Security (CCS)*, 2007.

[3] “Secure Computing Corporation, TrustedSource website, http://www.trustedsource.org.”

[4] Y. C. Tang, S. Krasser, P. Judge, and Y.-Q. Zhang, “Fast and effective spam IP detection with granular SVM for spam filtering on highly imbalanced spectral mail server behavior data,” in *Proc. of The 2nd International Conference on Collaborative Computing*, 2006.

[5] T. S. Furey, N. Christianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Hauessler, “Support vector machine classification and validation of cancer tissue samples using microarray expression data.,” *Bioinformatics*, vol. 16, no. 10, pp. 906–914, 2000.

[6] N. Japkowicz and S. Stephen, “The class imbalance problem: A systematic study,” *Intelligent Data Analysis*, vol. 6, no. 5, pp. 429–449, 2002.

[7] N. V. Chawla, N. Japkowicz, and A. Kotcz, “Editorial: special issue on learning from imbalanced data sets,” *SIGKDD Explorations*, vol. 6, no. 1, pp. 1–6, 2004.

[8] G. M. Weiss, “Mining with rarity: a unifying framework,” *SIGKDD Explorations*, vol. 6, no. 1, pp. 7–19, 2004.

[9] V. N. Vapnik, *Statistical Learning Theory*. New York: John Wiley and Sons, 1998.

[10] C. J. C. Burges, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[11] R. Akbani, S. Kwek, and N. Japkowicz, “Applying support vector machines to imbalanced datasets,” in *Proc. of the 15th European Conference on Machine Learning (ECML 2004)*, pp. 39–50, 2004.

[12] G. Wu and E. Y. Chang, “KBA: Kernel boundary alignment considering imbalanced data distribution,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 786–795, 2005.

[13] J. Dong, C. Y. Suen, and A. Krzyzak, “Algorithms of fast SVM evaluation based on subspace projection,” in *Proc. of 2005 IEEE International Joint Conference on Neural Networks, Vol. 2*, pp. 865–870, 2005.

[14] H. Isozaki and H. Kazawa, “Efficient Support Vector Classifiers for Named Entity Recognition,” in *Proc. of the 19th International Conference on Computational Linguistics (COLING’02)*, pp. 390–396, 2002.

[15] B. L. Milenova, J. S. Yarmus, and M. M. Campos, “SVM in oracle database 10g: removing the barriers to widespread adoption of support vector machines,” in *Proc. of the 31st international conference on Very large data bases*, pp. 1152–1163, 2005.

[16] T. Y. Lin, “Data mining and machine oriented modeling: A granular computing approach,” *Applied Intelligence*, vol. 13, no. 2, pp. 113–124, 2000.

[17] A. Bargiela and W. Pedrycz, *Granular Computing: An Introduction*. Kluwer: Kluwer Academic Pub, 2002.

[18] Y. C. Tang, B. Jin, and Y.-Q. Zhang, “Granular support vector machines with association rules mining for protein homology prediction,” *Artificial Intelligence in Medicine*, vol. 35, no. 1-2, pp. 121–134, 2005.

[19] R. J. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.

[20] R. Kohavi, “Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid,” in *Proc. of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 202–207, 1996.

[21] S. le Cessie and J. C. van Houwelingen, “Ridge estimators in logistic regression,” *Applied Statistics*, vol. 41, no. 1, pp. 191–201, 1992.

[22] A. P. Bradley, “The use of the area under the roc curve in the evaluation of machine learning algorithms.,” *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.