A Collaborative Approach for Spam Detection

Pedro Sousa, Artur Machado, Miguel Rocha
Dept. of Informatics
University of Minho
Braga, Portugal
pns@di.uminho.pt, ajpcm1@gmail.com, mrocha@di.uminho.pt

Paulo Cortez
Dept. of Information Systems
University of Minho
Guimares, Portugal
pcortez@dsi.uminho.pt

Miguel Rio
Dept. of Electronic & Electrical Engineering, UCL
London, UK
m.rio@ee.ucl.ac.uk

***Abstract*—Electronic mail is nowadays one of the most important Internet networking services. However, there are still many challenges that should be faced in order to provide a better e-mail service quality, such as the growing dissemination of unsolicited e-mail (spam) over the Internet. This work aims to foster new research efforts giving ground to the development of novel collaborative approaches to deal with spam proliferation. Using the proposed system, which is able to complement other anti-spam solutions, end-users are allowed to share and combine spam filters in a flexible way, increasing the accuracy and resilience levels of anti-spam techniques.**

***Keywords*-Electronic mail; Spam; Filtering; Collaboration**

I. Introduction

Electronic mail (e-mail) is a commonly used service for user communications in the Internet. With the advent and the growing popularity of e-mail, unsolicited e-mail (spam) also emerged very quickly and currently accounts for approximately 90% of all e-mail messages [1], i.e., over 120 billion of these messages are sent each day [4]. The cost of sending these e-mails is very close to zero being easy to reach a high number of potential consumers [3], since criminal organizations have access to millions of infected computers (known as botnets) [2], which might be used for spam proliferation. In this context, spam consumes resources, time spent reading unwanted messages, bandwidth, CPU, disk, being also used to spread malicious content.

Currently, there are two major approaches to fight spam [5][10]: Collaborative Filtering (CF) and Content-Based Filtering (CBF). The CF approach is based on sharing information about spam messages, while CBF uses a Data Mining (DM) classifier (e.g., Naive Bayes) that learns to discriminate spam from specific message characteristics (e.g., common spam words). As an example, the CF based approach might use information about spam messages that can be based on blacklists [2] containing IP addresses of known spam senders or fingerprints extracted from spam messages [6]. Current research on spam CBF relies mainly on improving individual classifier performance, by a better preprocessing [5] or enhancement of the learning algorithm [7]. The two approaches, CF and CBF, can also be combined to achieve more reliable methods. For example, a blacklist is often used at a server level to tag a large number of spam, while the remaining spam can be detected later by using a personalized CBF at the client level. Both CBF and CF techniques have drawbacks. CF often suffers from sparsity of data (e.g., users may classify few messages) and the first-rater problem (e.g., an e-mail cannot be classified unless a user has rated it before). Moreover, people have personal views of what is spam and CF often discards this issue [8]. On the other hand, in CBF, poor performances may be achieved by new users, since CBF requires several representative training examples. CBF is also vulnerable to dictionary or focused attacks, where the adversary can exploit DM models by contaminating the training set (e.g., by sending spam with a large amount of normal words) [9].

In this work, we propose and test a novel distributed collaborative approach able to be used both in a stand-alone perspective or complementing other anti-spam solutions. The collaborative perspective of the proposed system has novel characteristics when compared with traditional CF approaches. Here, the collaborative perspective relies on sharing the filtering models of the users, i.e., rather than exchanging information about some spam messages (e.g., fingerprints), these collaborating entities will share information about what each local filter has learned. The aim is to foster mutual relationships, where each user is interested in improving filtering at a personal level and the Internet is used to gather collaborators for that purpose.

The remainder of the paper is organized as follows: Section II discusses some guidelines for the deployment of a collaborative spam detection system; Section III describes the developed prototype; Section IV shows the experiments and the results and, finally, Section V draws the conclusions.

II. Deployment Guidelines

The concept of using several collaborative spam detection filters presents some advantages over the use of a single instance. As simple examples, if the individual local filter was trained with a reduced set of training cases, or was influenced by erroneous user classification, it is expected that a poor classification could be reached for some e-mail messages. Additionally, as spam techniques are constantly changing, it is possible that individual filters might be not sufficiently trained to deal with such volatile scenarios,

resulting in erroneous classification. So, the solution of using an appropriate set of collaborative filters is advantageous to deal with unsolicited email, being expectable that it improves the robustness and resilience levels of anti-spam tools. While sharing models is less sensitive than exchanging e-mail messages, there are still privacy issues to be considered. For instance, if user A has access to the filter of user B, then A may feed a given token into the model and thus know with some probability that such token was classified by B as spam or ham. Furthermore, the collaborative system might be implemented using distinct communication paradigms: a centralized server or a Peer-to-Peer (P2P) like application.

Under the first option, users may register into a centralized and secure service. This could be implemented by large companies or e-mail providers (e.g., Gmail), where all e-mails are stored at a given server. When a new e-mail is received, the system would activate the appropriate filters from the company or email users and then compute the final spam probability for the message. A method able to select appropriate filters for a given email user (also avoiding the use of possible malicious filters) will be presented and discussed later. For scalability, user profiles could be defined (e.g., country, profession) and clustering algorithms could be used to group users with similar interests. Another variant would be the definition of social networks, where users could choose their "friends". In such case, the collaborative exchange would only occur using filters from the social network. These systems could, for example, be implemented by social networking websites (e.g., Facebook). Alternatively, when the messages are stored locally at the client side, the server would be responsible for a blind exchange of the filters. At a given time, users may donate filters to the server, which will store them without any owner labels, to preserve privacy. Users may also automatically fetch anonymous filters from the server, scheduled by the e-mail client application, when the current user performance decays or when notified by the server that new filters are available. This can also be accomplished asynchronously by an explicit request of the user. To exchange the filters, a standard format should be adopted, such as the Predictive Model Markup Language (PMML) [11], which is compatible with a large number of DM tools. It should be noted that exchanging filters requires less communication costs and a filter built from a millions of e-mails can be described by a few hundreds or thousands of bytes [12].

As an alternative to centralized solutions, the use of P2P-like distribution schemes can be adopted, where all peers donate, store and fetch filters among each other. This could be implemented as a trusted and secure plug-in of an e-mail client, such as *Thunderbird*. The filter sharing process among the peers could work similarly to the explained in the secure server scheme (i.e., secure transfers and PMML). A selection of P2P nodes based on "friends" can also be applied, by selecting to which peers the software can connect, allowing the social network to take form in the P2P overlay.

In all of the above solutions, it is possible to define collaborative groups based on user profiles. In such cases, and even taking into account the use of solutions for anonymous filter exchange and storage, the degree of privacy is also dependent on the knowledge that each user has about who belongs to the collaborative group. Obviously, the degree of privacy is increased if each individual is not aware of the group composition. However, for some specific scenarios, it may also be attractive that the group composition can be assessed by all the participants in a given group. In this context, it is important to note that even in such scenarios it might be very difficult to "guess" who created each particular model, as in the envisaged scenarios it is expected that there will be typically a large number of users that dynamically may join or leave the group. Furthermore, additional security related issues can be discussed. The centralized model is vulnerable to DoS attacks, while P2P systems may be affected by sybil attacks. To prevent DoS, the server could employ a resilient statefull packet filter firewall [13]. Under the sybil assault, spammers could create multiple fake identities in order to populate the system with contaminated filters. Yet, as it will be shown in Section III, it is possible to weight the relevance of each shared filter and, by this way, ignore malicious filters.

III. Prototype of a Centralized Architecture

A centralized prototype was implemented using a *Thunderbird* plug-in also allowing that users register and choose friends (http://fiambre.dsi.uminho.pt/sf). The system is described in three modules: *i)* classification/filtering of messages; *ii)* distribution of local filters and *iii)* user interface.

A. Classification and Filtering of Messages

In this module, for filtering purposes, only textual content of e-mail messages will be addressed. While different algorithms can be adopted for spam filtering, such as Support Vector Machines (SVM) [3], we will use the simpler Naive Bayes (NB), which is widely adopted by anti-spam filtering tools [5]. However, this module should be able to simultaneously use and combine, in an intelligent way, several filters that were trained and shared by other collaborative users, instead of using only a local filter. This was implemented as an extension to the *Thunderbird* e-mail client. For that purpose, a previously existent extension entitled *Thunderbayes*, implementing the Naive Bayes algorithm, was further extended to accommodate the devised filtering module. The *Thunderbayes* extension allows to compare the characteristics of a given message with a specific training file, returning a value with the probability of such message being spam. The training file is mainly composed by words (or other features) belonging to messages received and previously classified by the user. Under the *Thunderbayes* implementation, the *Naive Bayes* algorithm runs in a proxy

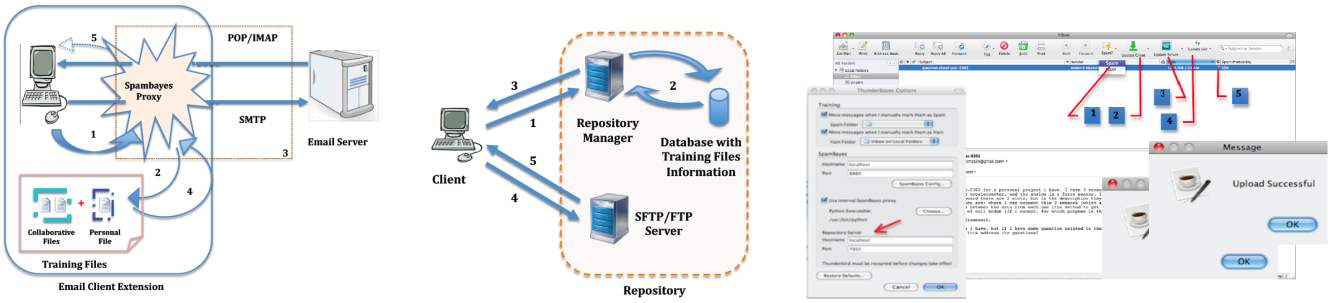

Figure 1. a) Classification and Filtering of Messages; b) Training Files Distribution Strategy; c) The user interface for the devised spam detection system.

entity, which acts as an intermediary between the user and the e-mail server. The default behavior of the *Thunderbayes* extension was modified to allow the use of multiple filters, in addition to the local filter of the user. Figure 1 a) illustrates the process of classification and filtering of e-mail messages in the devised architecture. In the proposed solution, the user is still able to perform a manual classification (step 1 of Fig. 1 a)) of e-mail messages as *ham* or *spam* and to correct such classification, whenever needed. Based on this classification, a local training file is built. This local filter will be used in conjunction with other external filters provided by other users. For that purpose, the user will be able to import (step 2 of Fig. 1 a)) other training files that were previously shared with the collaborative group (process explained in Section III-B). In the user client, when receiving a given e-mail (e.g., using POP as in step 3 of Fig. 1 a)), the proxy analyzes and classifies the message, according to the results provided by the corresponding filter training files (step 4 of Fig. 1 a)). Here, a spam probability should be calculated taking into account the *Naive Bayes* algorithm, which depends, in this case, on the content of several training files. Thus, a spam probability should be evaluated for each training file and all combined to achieve the final probability. In the current prototype, two alternatives solutions are available:

(i) Arithmetic Mean ($M1$) - the final spam probability is the arithmetic mean of all the individual probabilities.

(ii) Pearson Coefficient Correlation ($M2$) - for each imported training file, an individual weight (coefficient) is calculated using the Pearson coefficient correlation.

The objective of the $M2$ strategy is to determine, for a given user, which filters are more important to be considered for the evaluation of the final spam probabilities of the messages. For that purpose, the performance of each imported filter will be evaluated using the past messages previously classified by the user as *ham* or *spam*. The classification results from each imported filter will then be compared with the user's classification performed before for all the received messages. Based on such comparison, a Pearson coefficient correlation value is obtained for each filter. So, when receiving a new message the probabilities returned by each filter will be multiplied by the corresponding coefficient as given by Eq. 1, with $Pr_f(M)$ being the final spam probability of message M, n the number of training files considered, $Pr_i(M)$ the probability returned by the filter i and α_i the correlation coefficient of filter i.

$$Pr_f(M) = \frac{\sum_{i=1}^{n} \alpha_i * Pr_i(M)}{\sum_{i=1}^{n} \alpha_i} \quad (1)$$

The extension will then compute the final spam probability and such value is returned to the client for the final decision of considering the message *spam* or *ham* (step 5 of Figure 1 a)). A threshold $D \in [0, 1]$ is usually considered to assist this decision, i.e., if $Pr_f(M) > D$ then message is considered as *spam*. The method $M2$ has several advantages if selected to assist the classification criteria, as it is expected to be a more accurate method to select which appropriated filters should be considered for the final spam probability, and be an effective way to deal with contaminated filters that malicious users may upload to the group.

B. Distribution of Training Files

The repository of the training files might be organized in distinct ways (e.g., the definition of clusters of training files according to user profiles, personal interests, professions or origin) taking also advantage of the growing organization of Internet users in social networks. Independently of the solutions that might be adopted to rule the storage of training filters, a brief technical description of the centralized distribution module used in the prototype is now given. Figure 1 b) illustrates the main components of this architecture module. Two distinct servers are considered: the repository manager, implemented using the *Java* language, and a common file server adopting file transfer protocols such as FTP or SFTP. The user starts by authenticating itself in the repository manager informing the manager about the required operation, e.g., download or upload of training files (step 1 of Fig. 1 b)). In this communication step, the XML language was adopted to formulate the request. After verifying the user authentication credentials, and in the case of a download request, the repository manager checks in the

database all the training files available to the user. In the case of an upload, the repository manager generates an available file name to identify the file to be uploaded. It is important to note that, to preserve privacy, the selected name should not contain any information related with the contributing user. This interaction between the repository manager and the database is identified as step 2 in Fig. 1 b). After defining a list of available files to download or for uploaded, the manager is now able to construct an XML file to be returned to the user (step 3 of Fig. 1 b)). This file contains a file list and the identification of the ftp/sftp server for file transfer purposes, along with the authentication related data. The client will then connect the file server, starting an interactive process of file upload and/or download (steps 4 and 5 of Fig. 1 b)). The new collection of filters are then available to be combined by the previously mentioned methods ($M1, M2$).

C. User Interface

Figure 1 c) shows the user interface of the prototyped anti-spam service, which is based on a common *Thunderbird* window, with some buttons and fields highlighted with numbers from 1 to 5. The button 1 and field 5 are already defined by the *ThunderBayes* extension, with the first one being the user button for training purposes, allowing the user to manually classify the messages. The area identified by the number 5 shows, in percentage, the spam classification result, with the presented value resulting from the collaborative classification explained before. Buttons 2 to 4 were added to the *Thunderbird* interface and allow the operations of downloading (2) or uploading (3) the filter(s) from/to the repository. Button 4 integrates both operations, allowing the users to simultaneously upload their local filters and download the available filters. The name and port of the repository manager might also be configured by the user.

IV. Experiments and Results

This section shows the improvements results obtained by the system. The experiments use real spam/ham data to simulate all the dynamics of the proposed anti-spam system.

A. Ham and Spam Data

For evaluation purposes, there should be real mailboxes from different users, possibly from a social network, and collected during the same time period. However, due to privacy issues, it is very difficult to obtain data with such characteristics and to make it publicly available. As a reasonable alternative, we adopt a realistic mixture of real spam and ham emails, in a strategy similar to what has been proposed in [15][6]. The ham emails are from the Enron public repository, which was originally proposed for a global evaluation of filters, i.e., all Enron user messages were merged into a single dataset. Our personalized approach is based on five distinct datasets, whose ham is related with the cleaned-up form of the five largest Enron mailboxes from the same time period: kaminski- v (kam), farmer-d (far), beck-s (bec), lokay-m (lok) and kitchen-l (kit). As these are all Enron employees, it is reasonable to assume that they would know each other, i.e., belong to the same social network.

Regarding spam, we adopted the Bruce Guenter spam collection (http://untroubled.org/spam/) that is based in fake e-mails published in the Web (spam traps), collected during the years of 2006 and 2007. We selected messages with Latin characters, since the ham messages use this character type and non-Latin spam would be too easy to discriminate. Moreover, we removed duplicate messages by comparing MD5 signatures of the body messages. The mixture procedure that we propose is based on the date field (using GMT time zone) that each message was received. The intention is to preserve the time order of the messages. This type of mixture is more realistic when comparing with the random sampling procedures presented in [15][6]. Given that the Enron messages are from a different time period, we first added 6 years to the date field of all Enron emails, before performing the mixture. Let S_t denote a spam message received at time t, $S_{i,f} = (S_{t_i}, S_{t_{i+1}}, \ldots, S_{t_f})$ the time ordered sequence of the Bruce Guenter spam, $H_{u,i,f}$ and $S_{u,i,f}$ the sequences of ham/spam messages for user u from time t_i to t_f. Given the time period $t \in (t_i, \ldots, t_f)$, the algorithm randomly samples $|S'_{i,j}|$ spam messages from $S_{i,j}$. Next, the spam set $S_{u,i,j}$ is defined by randomly sampling emails from $S'_{i,j}$, with a probability of P for each message selection. The cardinality of $S'_{i,j}$ is given by:

$$|S'_{i,j}| = \frac{R \cdot \sum_{i=1}^{L} |H_{u,i,j}|}{P \cdot L} \quad (2)$$

where L denotes the total number of users available at the time period and R is the overall spam/ham ratio (including all users and time data). Given that there are different time periods for each user (Table I), we set four time sequences (i.e., t_i and t_j values) to be used by the mixture algorithm (Figure 2 a)). The mixture algorithm is affected by two parameters (R and P) that are fixed in this work to reasonable values. The global spam/ham ratio was set to $R = 1$, although the individual ratios can range from 0.6 to 1.5 (see Table I). Furthermore, the spam/ham ratios evolve through time (see Figure 2 b)). Turning to P, the probability of spam selection defines the percentage of common spam messages between users. If two users have similar Web exposure profiles, then they should receive similar spam. It is assumed that this scenario occurs with the five Enron Employees, hence we set $P = 0.5$. Under this value, any two users receive around 50% of the same spam, 3 users have around 25% of spam, and so on. The obtained corpus is named S-Enron and it is maed publicly available at http://www3.dsi.uminho.pt/pcortez/S-Enron.

B. Evaluation Procedure

Spam suffers from concept drift, i.e., the amount of spam received, the ham/spam ratio and even the content itself

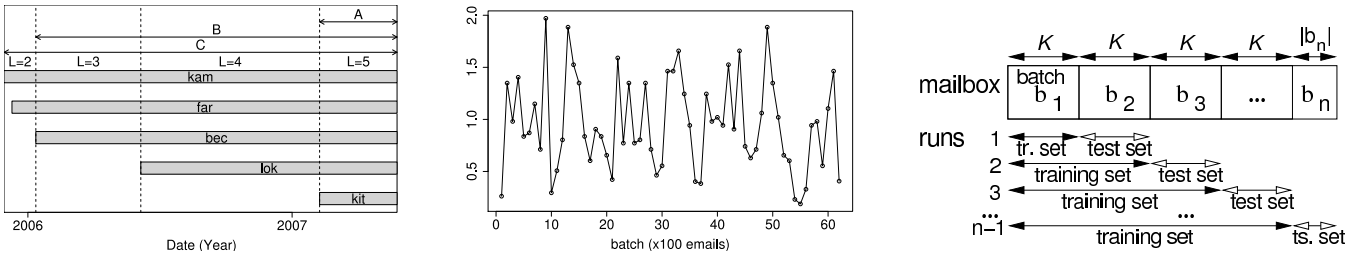

Figure 2. a) Chronological view of the S-Enron mailboxes b) Fluctuation of the spam/ham ratio for the far user c) Incremental retraining procedure

Table I
THE S-ENRON DATASETS CHARACTERISTICS

user	ham size	spam size	time period	spam /ham
kam	4363	2827	[12/05,05/07]	0.6
far	3294	2844	[12/05,05/07]	0.9
bec	1965	2763	[01/06,05/07]	1.4
lok	1455	2202	[06/06,05/07]	1.5
kit	789	623	[02/07,05/07]	0.8

evolve through time [16]. Thus, to evaluate the spam filters, we adopt the realistic incremental retraining evaluation procedure, which periodically trains and tests filters. This procedure is more realistic than the simple 50% train/test split adopted in [6]. Under the retraining evaluation, a mailbox is divided into batches of K adjacent messages ($b_1, \ldots, b_n$, $|b_n|$ may be less than K) [15]. For $i \in \{1, \ldots, n-1\}$, the filter is trained with $\mathcal{D}_u = b_1 \cup \ldots \cup b_i$ and tested with the emails from b_{i+1} (Figure 2 c)). Figure 2 b) plots an example the evolution of the far mailbox spam/ham ratio over distinct batches, with $K = 100$ and for C the time period of Figure 2 a). Typically, a spam filter outputs a probability $\in [0, 1]$. The corresponding predicted class is given by $spam$ if $Pr_f(M) > D$, where $D \in [0.0, 1.0]$ is a decision threshold defined by the user. For a given D and test set, it is possible to compute the true (TPR) and false (FPR) positive rates (with $TPR = TP/(TP + FN)$ and $FPR = FP/(TN + FP)$, where TP, FP, TN and FN denote the number of true positives, false positives, true negatives and false negatives). The receiver operating characteristic (ROC) curve measures the performance of a two class classifier across the range of all threshold (D) values, showing FPR (x-axis) versus TPR (y-axis) [17]. The overall accuracy is measured by the area under the curve (AUC= $\int_0^1 ROCdD$). The ideal classifier will have an AUC of 1.0, while a random one would present an AUC of 0.5. As the cost of losing ham (FP) is much higher than receiving spam (FN), D is often set to favor ROC points in the low false-positive region. Therefore, we use also the metric TPR at a specific FPR = r (denoted here as $TPR@FPR = r$), where r is a value close to 0.0 [7]. Under the incremental retraining procedure, one ROC will be computed for each b_{i+1} batch and the overall averaged results are be presented.

C. Results

This section presents the experiment results obtained when considering five users of the Enron collection. The results of the two collaborative methods ($M1$ and $M2$ available in the prototype) are compared with the default behavior of the classifier using a single local filter (Ind).

Figure 3, presenting the averaged results of the ROC curves for the users, clearly shows an improvement in the accuracy of the spam classifiers. In most of the cases, there is a significant gain in the spam classification accuracy, since higher true positive rates (TPR) are obtained for distinct values of false positive rates (FPR). A more detailed analysis of the data plotted in Figure 3 is given in Table II presenting, for each user and method, the AUC and $TPR@FPR$=0.01 values. As explained, for spam detection purposes this second metric considers a specific low value for the false positive rate achieved by the classifier. As observed in Table II, and for the majority of the users, there is a significant improvement in this metric, clearly showing the good performance of the devised solution. The user *lok* was the unique exception, with no improvements regarding the $TPR@FPR$=0.01 analysis, even though the gains observed in AUC values. The averaged AUC and $TPR@FPR$=0.01 values presented in Table II (the last line) clearly show the advantages of using the proposed solution, as substantial improvements are observed in both metrics. Even more noticeable is the difference of the values for metric $TPR@FPR$=0.01 as, on average, a value of 34.5 is obtained for the local filter vs 69.0 and 69.1 for the collaborative approach using $M1$ and $M2$ methods, respectively. As observed, both $M1$ and $M2$ methods achieved similar spam detection levels. As mentioned before, this is explained by the dataset that was used, where the users belong to the same organization and, probably, share common interests. Moreover, they were assumed as having similar levels of exposure to spam reception. However, for scenarios with a higher heterogeneity level of user profiles or with the existence of malicious filters, the use of the method $M2$ will be crucial to assure a correct selection of the filters.

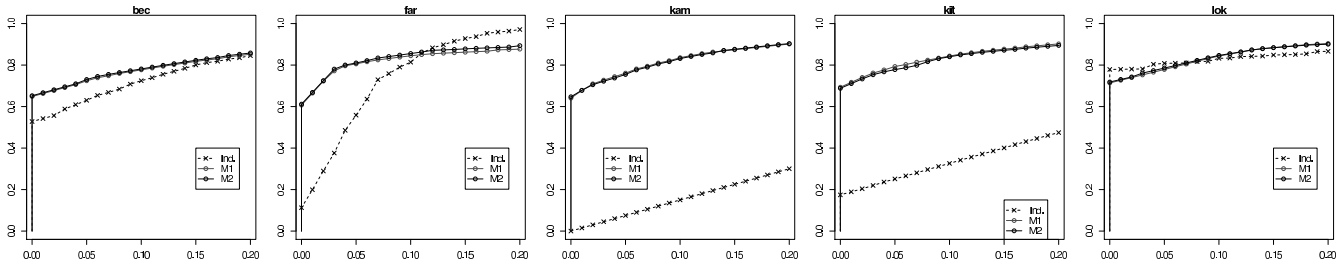

Figure 3. ROC curves of the e-mail users considered in the experiments

Table II
SUMMARY OF THE RESULTS- ROC CURVES AND $TPR@FPR$

	AUC			TPR@FPR=0.01		
user	**Ind.**	**M1**	**M2**	**Ind.**	**M1**	**M2**
kam	62.1	**94.5**	**94.5**	1.5	67.8	**67.9**
far	**93.5**	92.2	92.9	19.9	66.5	**66.8**
bec	91.5	92.1	**92.2**	54.2	66.3	**66.7**
lok	91.4	**93.8**	93.7	**78.0**	72.7	73.1
kit	74.6	**94.5**	94.3	18.9	**71.7**	71.1
Average	82.6	93.4	**93.5**	34.5	69.0	**69.1**

V. CONCLUSION

This paper proposes a novel anti-spam approach, where users with related interests have the opportunity to collaborate in order to improve spam detection performances achieved by current techniques adopted in the Internet. To better assess the viability of the proposed system, a dataset consisting of a realistic mixture of *spam* and *ham* messages was used to simulate the dynamics of the proposed solution. As described, the proposed system, even resorting to a small number of users, clearly outperformed the local filtering approach and is expected to improve the robustness and resilience levels of anti-spam mechanisms.

ACKNOWLEDGMENT

Research supported by FCT grant PTDC/EIA/64541/2006

REFERENCES

[1] Messaging Anti-Abuse Working Group. *Email Metrics Program: The Network Operators' Perspective. Report #10 - third and fourth quarter 2008*, S. Francisco CA, USA, March 2009.

[2] A. Ramachandran and N. Feamster. *Understanding the Network-Level Behavior of Spammers*. In Proc. of SIGCOMM'06, pp. 291- 302, 2006.

[3] V. Cheng and C. Li, *Personalized Spam Filtering with Semisupervised Classifier Ensemble,* in IEEE/WIC/ACM International Conference on Web Intelligence, 2006.

[4] C. Kanich et al., *Spamalytics: An Empirical Analysis of Spam Marketing Conversion,* in Computer and Communications Security Conference (CCS08). ACM, pp. 27-31, 2008.

[5] J. Mendez, I. Cid, D. Glez-Pena, M. Rocha, and F. Fdez-Riverola, *A Comparative Impact Study of Attribute Selection Techniques on Naive Bayes Spam Filters,* in 8th Industrial Conference on Data Mining, LNAI 5077, pp. 213-227, 2008.

[6] Z. Zhong et al., *ALPACAS:A Largescale Privacy-Aware Collaborative Antispam System,* Proc. INFOCOM, pp. 556-564, 2008.

[7] M. Chang, W. Yih, and C. Meek, *Partitioned Logistic Regression for Spam Filtering,* in 14th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining, pp. 97-105, 2008.

[8] A. Gray and M. Haahr, *Personalised, Collaborative Spam Filtering,* in 1st Conf. on E-Mail and Anti-Spam CEAS, 2004.

[9] B. Nelson et al. , *Exploiting Machine Learning to Subvert Your Spam Filter,* in 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats. ACM Press, pp. 1-9, 2008.

[10] E. Blanzieri and A. Bryl. *A survey of learning-based techniques of email spam filtering*. Artificial Intelligence Review, 29(1):63-92, 2008.

[11] R. Grossman, M. Hornick, and G. Meyer, *Data Mining Standards Initiatives,* Communications of ACM, vol. 45, no. 8, pp. 5961, 2002.

[12] A. Garg, R. Battiti, and R. Cascella, *May I borrow your filter? Exchanging filters to combat spam in a community,* in 20th Int. Conf. on Advanced Information Networking and Applications, Vol. 2, pp.489-493, 2006.

[13] H. Kim et al., *Preventing session table explosion in packet inspection computers,* IEEE Trans. on Computers, pp. 238-240, 2005.

[14] H. Yu, M. Kaminsky, P. Gibbons, and A. Flaxman, *Sybilguard: Defending against sybil attacks via social networks,* IEEE/ACM Trans. on Networking (TON), vol. 16, no. 3, pp. 576-589, 2008.

[15] V. Metsis, I. Androutsopoulos, and G. Paliouras, *Spam Filtering with Naive Bayes Which Naive Bayes?,* in Third Conf. on Email and Anti-Spam (CEAS), pp. 125134, 2006.

[16] T. Fawcett, *"In vivo spam filtering: A challenge problem for KDD,* SIGKDD Explorations, vol. 5, no. 2, pp. 140148, 2003.

[17] T. Fawcett, *An introduction to ROC analysis, Pattern recognition letters,* vol. 27, no. 8, pp. 861874, 2006.