International Journal of Scientific Engineering and Technology (ISSN : 2277-1581)
Volume No.2, Issue No.7, pp : 613-618 1 July 2013

COSDES of Junk E-Mail with Junk Free System Scheme

M. Chitra, D. Eswari

Computer Science and Engineering, P.S.R.Rengasamy College of Engineering for women, Sivakasi
Email : chitu.pandian@gmail.com, eswaridoss87@gmail.com

Abstract— ***E-mail communication is indispensable now, but the e-mail spam problem is continuously growing more. In recent years, the notion of collaborative spam filtering with near-duplicate similarity matching scheme has been discussed widely. The idea of the similarity matching scheme for spam detection is, to maintain a database formed by user feedback and to block near-duplicate spams. The previous works mainly represent each e-mail by an abstraction derived from e-mail content text. These abstractions of e-mails cannot catch the evolving spams, and are thus not effective enough in near-duplicate detection. A procedure to generate the e-mail abstraction using HTML content in e-mail, and newly devised abstraction which can be more efficient in capturing the duplicate phenomenon of spam is presented here. COSDES (COllaborative Spam DEtection System), a complete spam detection system, possesses an efficient near -duplicate matching scheme and a progressive update scheme. The forward-looking update scheme enables system COSDES to keep the most up-to-date information for near-duplicate detection. This system evaluates on a live data set collected from an e-mail server and shows that this system performs better than the previous approaches in detection results and is applicable to the real world.***

Keywords—Spam detection, e-mail abstraction, duplicate matching.

1. INTRODUCTION

Nowadays E-mail communication is prevalent and indispensable. The threat of junk e- mails also known as spams becomes more and more serious. According to a survey by the website Top Ten REVIEWS [11], 40 percent of e-mails were considered as spams in the year 2006. Statistics collected by MessageLabs1 shows that the recently the spam rate is over 70 percent and persistently remains high. Existing filters generally perform well when dealing with clumsy spams, which have duplicate content with suspicious keywords or are sent from an identical notorious server. Therefore, the next stage of spam detection research should focus on coping with cunning spams which evolve naturally and continuously.

Based on the features of e-mails being used, previous works on spam detection can be generally classified into three categories: 1) content-based methods, 2) noncontent-based methods, and 3) others. Initially, researchers analyze e-mail content text and model this problem as a binary text classification task. Representatives of this category are Naive Bayes [14], [20] and Support Vector Machines (SVMs) [1], [10], [15], [27] methods. In general, Naive Bayes methods train a probability model using classified e-mails, and each word in e-mails will be given a probability of being a suspicious spam keyword. As for SVMs, it is a supervised learning method, which possesses outstanding performance on text classification tasks. Traditional SVMs [10] and improved SVMs [1], [15], [27] have been investigated. While above conventional machine learning techniques have reported excellent results with static data sets, one major disadvantage is that it is cost-prohibitive for large-scale applications to constantly retrain these methods with the latest information to adapt to the rapid evolving nature of spams. The spam detection of these methods on the e-mail corpus with various languages has been less studied yet. In addition, other classification techniques, including Markov random field model [3], neural network [6] and logic regression [2], and certain specific features, such as URLs [26] and images [19], [29] have also been taken into account for spam detection. The authors make use of spam-vocabulary patterns produced by Teiresias pattern discovery algorithm. In [16], the I-Match signature determined by a set of unique terms shared by spams and the I-Match lexicon is put to use. In [21], the content similarity of e-mails computed using extracted words is measured.

Though previous researchers have developed various methods on near-duplicate spam detection [7], [8], [12], [16], [17], [22], [23], [24], [25], [30], [31], these works are still subject to some drawbacks. For achieving the objectives of small storage size and efficient matching, prior works mainly represent each e-mail by a succinct abstraction derived from e-mail content text. Moreover, hash-based text representation is applied extensively. The major problem of these abstractions is that they may be too brief and thus may not be robust enough to withstand intentional attacks.

In this paper, section 2 describes about the preliminaries including the definition of near-duplicate and email abstraction scheme is given. In Section 3, the complete system model of Cosdes is depicted. In section 4, proposed system, Junk free mail system is described. The implementations are shown in Section 5, and finally, this paper is concluded with Section 6.

2. PRELIMINARIES

In this section, the definition of near-duplicate is presented in Section 2.1. The email abstraction scheme is described in Section 2.2.

This paper is about exploring a devise for sophisticated email abstraction, which can more effectively capture the near-duplicate phenomenon of spams. Almost all e-mails nowadays are in Multipurpose Internet Mail Extensions (MIME) format with the text/html content-type. That is, the HTML content is available in an e-mail and provides sufficient information about e-mail layout structure. In view of that observation, the specific procedure Structure Abstraction Generation (SAG), which generates an HTML tag sequence to represent each e-mail is proposed here. Different from previous works, the procedure of SAG focuses on the e-mail layout structure instead of detailed content text. In this regard, each paragraph of text without any HTMLtag embedded will be transformed to a newly defined tag <*mytext/*>.

Definition 1 ***(<mytext/>).*** *<mytext/> is a newly defined tag that represents a paragraph of text without any HTML tag embedded.*

The semantics of the text is ignored, so the proposed abstraction scheme is inherently applicable to e-mails in all languages. This feature is superior to most existing methods. Once the e-mails are represented by our newly devised e-mail abstractions, two e-mails are viewed as near-duplicate if their HTML tag sequences are exactly identical to each other. Even when spammers insert random tags into e-mails, the proposed e-mail abstraction scheme will still retain efficacy since arbitrary tag insertion is prone to syntax errors or tag mismatching, meaning that the appearance of the e-mail content will be greatly altered. The proposed procedure SAG also adopts some heuristics to better guarantee the robustness of our approach. While a more sophisticated e-mail abstraction is introduced, one challenging issue arises: how to efficiently match each incoming e-mail with an existing huge spam database. To resolve this issue, an innovative tree structure, SpTrees, to store large amounts of the e-mail abstractions of reported spams, and SpTrees contribute to substantially promoting the efficiency of matching is introduced.

Cosdes possesses an efficient near-duplicate matching scheme and a progressive update scheme. The progressive update scheme adds in new reported spams, and also removes obsolete ones in the database. With **Cosdes** maintaining an up-to-date spam database, the detection result of each incoming e-mail can be determined by the near-duplicate similarity matching process.

2.1 Definition for Near- Duplicate

The main idea of near-duplicate spam detection is to exploit reported known spams to block subsequent ones which have similar content. For different forms of e-mail representation, the definitions of similarity between two e-mails are different. Unlike most prior works representing e-mails based mainly on content text, this system investigate representing each e-mail using an HTML tag sequence, which depicts the layout structure of e-mail, and look forward to more effectively capturing the near-duplicate phenomenon of spams. Initially, the definition of <*anchor*> tag is given as follows:

Definition 2 ***(<anchor>).*** *The tag <anchor> is one type of newly defined tag that records the domain name or the e-mail address in an anchor tag.*

For example, an anchor tag <a href="http://arbor.ee.ntu.edu.tw/index.htm"> is transformed to <arbor.ee.ntu.edu.tw>. The anchor tag <a href="mailto:cytseng@arbor.ee.ntu.edu.tw"> is transformed to <cytseng@arbor.ee.ntu.edu.tw>. The need for creating the <anchor> tag is to minimize the false positive rate when the number of tags in an e-mail abstraction is short. One of the common attacks to this type of representation is to insert a random normal paragraph without any suspicious keywords into unobvious position of an e-mail.

```
Procedure SAG
Input: the email with text/html content-type,
       the tag length threshold (Lth_short) of the short email
Output: the email abstraction (EA) of the input email
1   // Tag Extraction Phase
2   Transform each tag to <tag.name>;
3   Transform each paragraph of text to <mytext/>;
4   AnchorSet = the union of all <anchor>;
5   EA = the concatenation of <tag.name>;
6   Preprocess the tag sequence of EA;
7   // Tag Reordering Phase
8   for (each tag of EA) // pn: position number
9     tag.new_pn = ASSIGN_PN (EA.tag_length, tag.pn);
10    Put the tag to the position tag.new_pn;
11  EA = the concatenation of <tag.name> with new_pn;
12  // <anchor> Appending Phase
13  if (EA.tag_length < Lth_short)
14    Append AnchorSet in front of EA;
15  return EA;
End
```

Fig .1. Algorithm of Procedure SAG

Definition 3 ***(Tag Length).*** *The tag length of an e-mail abstraction is defined as the number of tags in an e-mail abstraction.*

The two e-mail abstractions are near-duplicate only if they are exactly identical to each other. The major reason is that there are numerous HTML tag patterns appearing commonly and frequently. Partial matching of HTML tag sequences will cause much higher rate of false positive error, and the complexity will

be too high to achieve efficient matching. In addition, for further speed-up, while the tag length of an e-mail abstraction is longer, we even apply a looser matching criterion, which does not degrade detection results.

2.2. E-MAIL ABSRACTION SCHEME

In this section, a new e-mail abstraction scheme is introduced. In the section 2.2.1, procedure SAG is presented to represent the generation process of an e-mail abstraction and the devised data structures SpTable and SpTrees are illustrated in Section 2.2.2.

2.2.1 Structure Abstraction Generation

The specific procedure SAG to generate the e-mail abstraction using HTML content in e-mail is proposed here. SAG is explained with the example in Fig. 2, and the algorithmic form of SAG is given Fig. 1. Procedure SAG is composed of three phases, Tag Extraction Phase, Tag Reordering Phase, and Appending Phase. The following sequences of operations are carried out in the preprocessing step.

1. Front and rear tags (gray area in Fig. 2) are removed.
2. Nonempty tags that have no corresponding start tags or end tags and mismatched nonempty tags are removed.
3. All empty tags are regarded as the same and are replaced by the newly created <empty/> tag. Moreover, successive <empty/> tags are reduced and only one <empty/> tag is retained.
4. The pairs of nonempty tags enclosing nothing are removed.

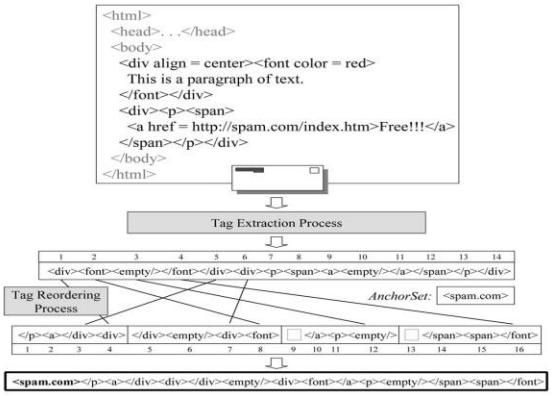

Fig .2. Example procedure flow of SAG

2.2.2 Design of SpTables And SpTrees

SpTables and SpTrees (sp stands for spam) are presented to store large amounts of the e-mail abstractions of reported spams. That HTML format or content is extracted in tag extraction phase.Then they are arranged in tag reordering phase. HTML content are checked in reverse order and they checked according to the algorithm of SAG. The procedure SAG is presented to show the generation process of an e-mail abstraction. As in Fig. 3 SpTrees are the kernel of the database, and the e-mail abstractions of the spams that are collected are maintained in the SpTrees.

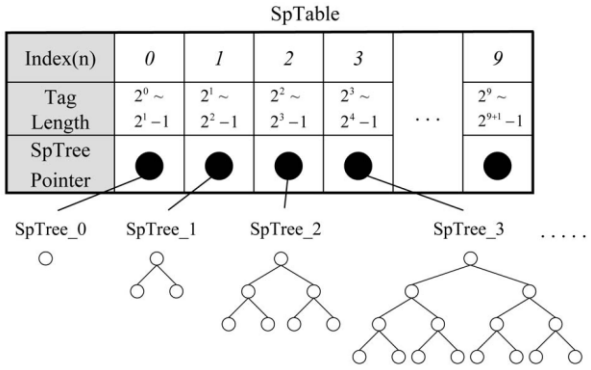

Fig. 3.Structure of SpTables and SpTrees

3. COLLABORATIVE SPAM DETECTION SYSTEM

A complete collaborative spam detection system Cosdes is introduced in this section. The system model of Cosdes is explained in Section 3.1.

System model of COSDES

In Fig.4 the system model of Cosdes is illustrated, and the algorithm of Cosdes is outlined in Fig. 5. Three parameters, T_m (the maximum time span for reported spam's being retained in the system), T_d (the time span for triggering Deletion Handler), and S_{th} (the score threshold for determining spams) should be given for Cosdes.

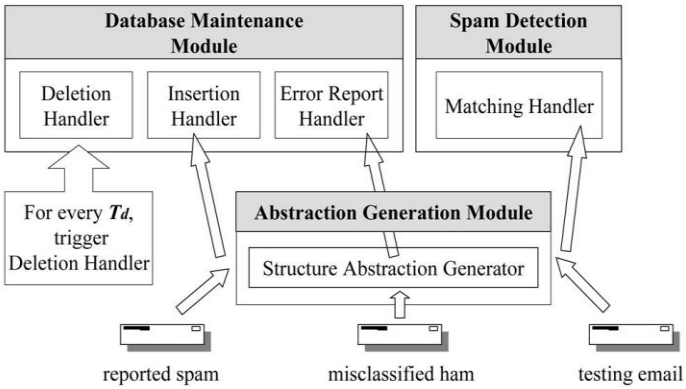

Fig. 4. System Model of Cosdes

International Journal of Scientific Engineering and Technology (ISSN : 2277-1581)
Volume No.2, Issue No.7, pp : 613-618 1 July 2013

Cosdes includes three major modules Abstraction Generation, Database Maintenance and Spam Detection. Overall, Cosdes is self-adjusting and retains the most up-todate spams for near-duplicate detection.

System Cosdes
Input: T_m: the maximum time span for reported spams being retained in the system,
T_d: the time span for triggering Deletion Handler,
S_{th}: the score threshold for determining spams

```
switch (circumstance)
case: when receiving a reported spam
    if (EA.reporter.S_R > S_initial);
        Trigger Insertion Handler(EA);
        Increase S_R of the reporter in RepTable; // Rep: Reputation
    break;
case: when receiving a testing email
    Trigger Matching Handler(EA, S_th);
    if (the testing email is classified as a spam);
        Trigger Insertion Handler(EA);
    break;
case: when receiving a misclassified ham
    Trigger Error Report Handler(EA);
    break;
case: for every T_d
    Trigger Deletion Handler(T_m);
    break;
End
```

Fig. 5.Algorithm of Cosdes

4. PROPOSED SYSTEM

4. 1 JUNK FREE MAIL SYSTEM

This system allows the user to do all the mail operations and to send and receive mail in a proper manner without spam mails. A Spam detection application is, first of all a mail server application. To conceptualize a Spam detection application, additional information like Spam Keywords is added to identify the Spam mail from the database. The specific procedure SAG is proposed to generate the e-mail abstraction using HTML content in e-mail, and this newly-devised abstraction can more effectively capture the near-duplicate phenomenon of spams. Here, the Spam detection in E-mail is performed using ASP.NET.

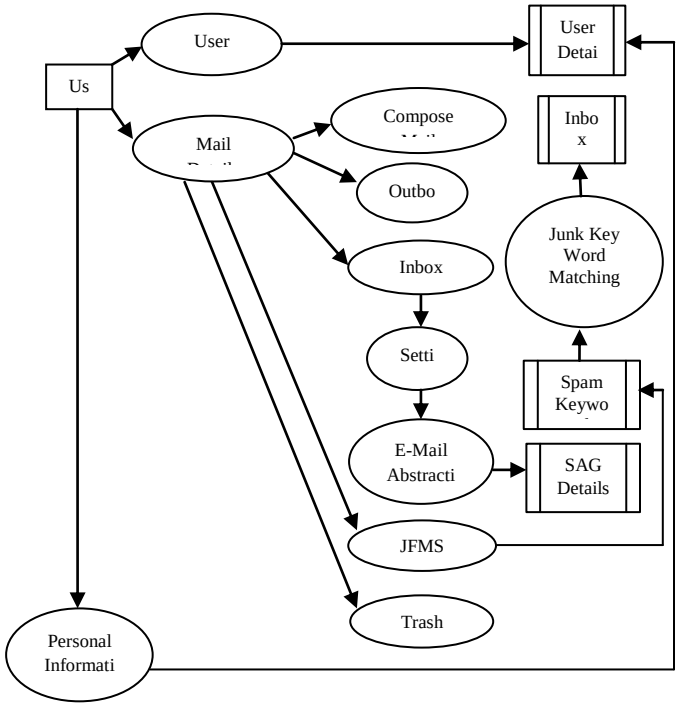

Fig. 6. DFD for Junk Free Mail System

4.1.1 Login Form

This module involves logging into the user's account page. If the user does not type the username and password correctly , it will show a message to prompt the user to retype the correct username and password . This will continues till the user types the correct username and password.

4.1.2 New Account Form

This module is mandatory for new user. This module is depicted in Fig .7. The user should type their details in the appropriate textboxes to create an account. This will be stored in the database for reference.

4.1.3 Inbox Layout Form

This module is updated regularly when the other user send the
messages to the current user.

4.1.4 Compose Layout Form

This module is used to compose a mail to other user from the current user. It consists of subject, content area, send and forward button. Once the entire text box has been filled and send button is clicked it starts its validation of the content

which is known as tag extraction phase which is for detecting the spam.

4.1.5 Trash Layout Form

This module stores all the deleted mails. This will continue till the user deletes the unwanted mails. This module is used by the user for retrieving the deleted mails from the inbox, outbox, junk mail etc.

4.1.6 Outbox Layout Form

This module is updated regularly when the user sends the messages to the other user. This outbox form consists of user name, subject, date, day with time etc..

4.1.7 Junk Mail Layout Form

This module stores all the spam mails. Fig .9 depicts this module. Here the spam mails are automatically updated through the settings by running the e-mail abstraction.

4.1.8 Settings

This module is for detecting the spam and filtering it out. When a spam mail is received into the inbox, spam detection and filtering process starts. Fig .8 depicts the settings of this system. Once the text box is filled with the spam keyword and by clicking the button show junk it display the messages which matches with the spam keyword by clicking the check box and click the button run email abstraction it display the grid which consists of subject length, mail content length and E-mail abstraction and enter the text box with junk name and click the button mark as spam it will be moved to the junk mail.

4.2 Advantages of Junk Free Mail System

- Matching is done with the Html tags and its length.
- Keywords are used for easy matching.
- Maintaining a Spam Database for matching.
- Time is reduced due to the automatic processing.

5. IMPLEMENTATIONS

In this section, the implementation of the junk free system with the snap shots is depicted. Fig .7 depicts the creation of user account, Fig .8 depicts the settings for junk free mail system, and Fig 9. depicts the junk mail system.

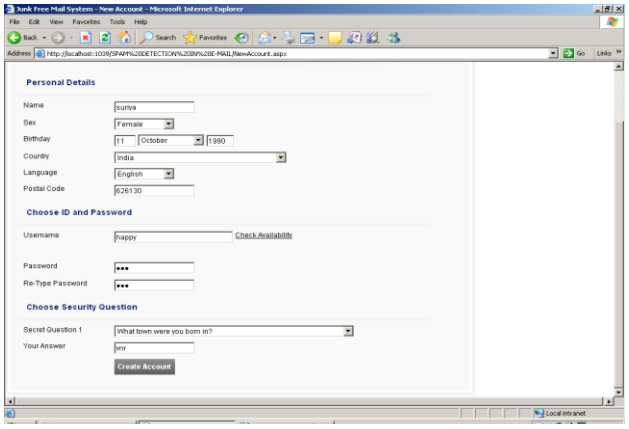

Fig .7. User account creation

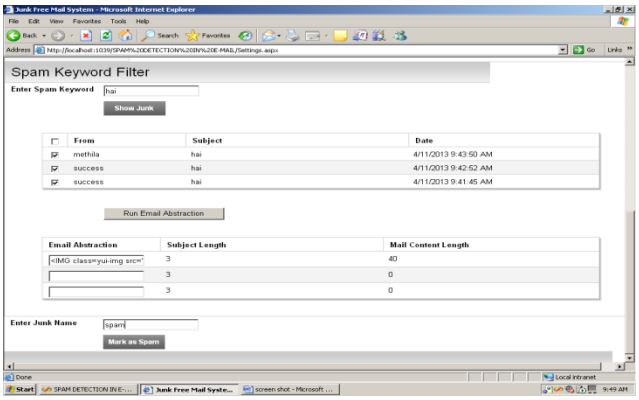

Fig .8. Settings for spam keyword filter

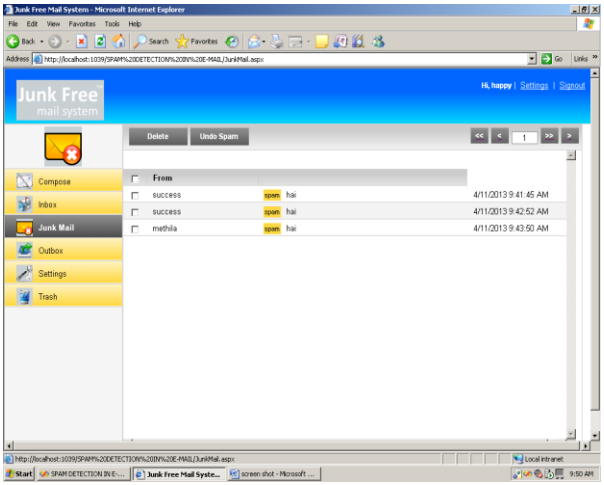

Fig .9. Junk Mail

6. CONCLUSION AND FUTURE WORK

In the field of collaborative spam filtering by near-duplicate detection, the superior e-mail abstraction scheme is required to more certainly catch the spams. The specific procedure SAG is given to generate the e-mail abstraction using HTML content in e-mail, and newly-devised e-mail abstraction can more effectively capture the near-duplicate phenomenon of spam's. A complete spam detection system Cosdes has been designed to efficiently process the near-duplicate matching and to progressively update the known spam database. Consequently, the most up-to-date information can be invariably kept to block subsequent near-duplicate spams.

Furthermore, since image spam is constantly evolving, we believe it is a constant battle to find new features that can effectively defeat new image spam techniques. Such as maintaining a E-mail abstraction database and marking the unwanted mail as spam. DNS-like system can be used to detect the images or hyperlink by using pixels in different techniques.

REFERENCES

i. *E. Blanzieri and A. Bryl, "Evaluation of the Highest Probability SVM Nearest Neighbor Classifier with Variable Relative Error Cost," Proc. Fourth Conf. Email and Anti-Spam (CEAS), 2007.*

ii. *M.-T. Chang, W.-T. Yih, and C. Meek, "Partitioned Logistic Regression for Spam Filtering," Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data mining (KDD), pp. 97-105, 2008.*

iii. *S. Chhabra, W.S. Yerazunis, and C. Siefkes, "Spam Filtering Using a Markov Random Field Model with Variable Weighting Schemas," Proc. Fourth IEEE Int'l Conf. Data Mining (ICDM),pp. 347-350, 2004.*

iv. *P.-A. Chirita, J. Diederich, and W. Nejdl, "Mailrank: Using Ranking for Spam Detection," Proc. 14th ACM Int'l Conf. Information and Knowledge Management (CIKM), pp. 373-380, 2005.*

v. *R. Clayton, "Email Traffic: A Quantitative Snapshot," Proc. of the Fourth Conf. Email and Anti-Spam (CEAS), 2007.*

vi. *A.C. Cosoi, "A False Positive Safe Neural Network; T he Followers of the Anatrim Waves," Proc. MIT Spam Conf., 2008.*

vii. *E. Damiani, S.D.C. di Vimercati, S. Paraboschi, and P. Samarati, "An Open Digest-Based Technique for Spam Detection," Proc. Int'l Workshop Security in Parallel and Distributed Systems, pp. 559-564, 2004.*

viii. *E. Damiani, S.D.C. di Vimercati, S. Paraboschi, and P. Samarati, "P2P-Based Collaborative Spam Detection and Filtering," Proc.Fourth IEEE Int'l Conf. Peer-to- Peer Computing, pp. 176-183, 2004.*

ix. *P. Desikan and J. Srivastava, "Analyzing Network Traffic to Detect E-Mail Spamming Machines," Proc. ICDM Workshop Privacy and Security Aspects of Data Mining, pp. 67-76, 2004.*

x. *H. Drucker, D. Wu, and V.N. Vapnik, "Support Vector*

a. *Machines for Spam Categorization," Proc. IEEE Trans. Neural Networks, pp. 1048-1054, 1999.*

xi. *D. Evett, "Spam Statistics," http://spam-filter-review.topten reviews.com/spam-statistics.html, 2006.*

xii. *[A. Gray and M. Haahr, "Personalised, Collaborative Spam Filtering," Proc. First Conf. Email and Anti- Spam (CEAS), 2004.*

xiii. *S. Hershkop and S.J. Stolfo, "Combining Email Models for False Positive Reduction," Proc. 11th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD), pp. 98-107, 2005.*

xiv. *J. Hovold, "Naive Bayes Spam Filtering Using Word- Position-Based Attributes," Proc. Second Conf. Email and Anti-Spam (CEAS),2005.*

xv. *A. Kolcz and J. Alspector, "SVM-Based Filtering of Email Spam with Content-Specific Misclassification Costs," Proc. ICDM Workshop Text Mining, 2001.*

xvi. *A. Kolcz, A. Chowdhury, and J. Alspector, "The Impact of Feature Selection on Signature-Driven Spam Detection," Proc. First Conf. Email and Anti-Spam (CEAS), 2004.*

xvii. *J.S. Kong, P.O. Boykin, B.A. Rezaei, N. Sarshar, and V.P. Roychowdhury, "Scalable and Reliable Collaborative Spam Filters: Harnessing the Global Social Email Networks," Proc. Second Conf. Email and Anti-Spam (CEAS), 2005.*

xviii. *T.R. Lynam and G.V. Cormack, "On-Line Spam Filter Fusion," Proc. 29th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR), pp. 123-130, 2006.*

xix. *B. Mehta, S. Nangia, M. Gupta, and W. Nejdl, "Detecting Image Spam Using Visual Features and Near Duplicate Detection," Proc. 17th Int'l Conf. World Wide Web (WWW), pp. 497-506, 2008.*

xx. *V. Metsis, I. Androutsopoulos, and G. Paliouras, "Spam Filtering with Naive Bayes—Which Naive Bayes?" Proc. Third Conf. Email and Anti-Spam (CEAS), 2006.*

xxi. *M.S. Pera and Y.-K. Ng, "Using Word Similarity to Eradicate Junk Emails," Proc. 16th ACM Int'l Conf. Information and Knowledge Management (CIKM), pp. 943-946, 2007.*

xxii. *I. Rigoutsos and T. Huynh, "Chung-Kwei: A Pattern- Discovery-Based System for the Automatic Identification of Unsolicited Email Messages (SPAM)," Proc. First Conf. Email and Anti-Spam (CEAS), 2004.*

xxiii. *S. Sarafijanovic and J.-Y.L. Boudec, "Artificial Immune System for Collaborative Spam Filtering," Proc. Second Workshop Nature Inspired Cooperative Strategies for Optimization (NICSO), 2007.*

xxiv. *S. Sarafijanovic, S. Perez, and J.-Y.L. Boudec, "Improving Digest- Based Collaborative Spam Detection," Proc. MIT Spam Conf., 2008.*

xxv. *S. Sarafijanovic, S. Perez, and J.-Y.L. Boudec, "Resolving FP-TP Conflict in Digest-Based Collaborative Spam Detection by Use of Negative Selection Algorithm," Proc. Fifth Conf. Email and Anti- Spam (CEAS), 2008.*

xxvi. *K.M. Schneider, "Brightmail URL Filtering," Proc. MIT Spam Conf., 2004.*

xxvii. *D. Sculley and G.M. Wachman, "Relaxed Online SVMs for Spam Filtering," Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR), pp. 415-422, 2007.*

xxviii. *C.-Y. Tseng, J.-W. Huang, and M.-S. Chen, "Promail: Using Progressive Email Social Network for Spam Detection," Proc. 10th Pacific-Asia Conf. Knowledge Discovery and Data Mining (PAKDD), pp. 833-840, 2007.*

xxix. *Z. Wang, W. Josephson, Q. Lv, and K.L.M. Charikar, "Filtering Image Spam with Near-Duplicate Detection," Proc. Fourth Conf. Email and Anti-Spam (CEAS), 2007.*

xxx. *K. Yoshida, F. Adachi, T. Washio, H. Motoda, T. Homma, A. Nakashima, H. Fujikawa, and K. Yamazaki, "Density-Based Spam Detector," Proc. 10th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD), pp. 486-493, 2004.*

xxxi. *F. Zhou, L. Zhuang, B.Y. Zhao, L. Huang, A.D. Joseph, and J.D. Kubiatowicz, "Approximate Object Location and Spam Filtering on Peer-to-Peer Systems," Proc. ACM/IFIP/USENIX Int'l Middleware Conf., pp. 1-20, 2003*