RESEARCH ARTICLE OPEN ACCESS

A Two-Way Spam Detection System With A Novel E-Mail Abstraction Scheme

B.Venkata Ramana, U.Mahender, R.V.Gandhi, Nalugotla Suman
Assistant Professor, Holy Mary Institute of Technology & Science, Hyderabad, RR Dist.
Assistant Professor, TKR College of Engineering & Technology, Hyderabad, RR Dist.
Assistant Professor, Mother Theresa Institute of Engineering & Technology, Palamaner, Chitoor Dist.
Assistant Professor, Hi-Point College of Engineering & Technology, Hyderabad, RR Dist.

Abstract:
E-mail communication is indispensable nowadays, but the e-mail spam problem continues growing drastically. In recent years, the notion of collaborative spam filtering with near-duplicate similarity matching scheme has been widely discussed. The primary idea of the similarity matching scheme for spam detection is to maintain a known spam database, formed by user feedback, to block subsequent near-duplicate spams. On purpose of achieving efficient similarity matching and reducing storage utilization, prior works mainly represent each e-mail by a succinct abstraction derived from e-mail content text. However, these abstractions of e-mails cannot fully catch the evolving nature of spams, and are thus not effective enough in near-duplicate detection. In this paper, we propose a novel e-mail abstraction scheme, which considers e-mail layout structure to represent e-mails. We present a procedure 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. Moreover, we design a complete spam detection system Cosdes (standing for COllaborative Spam Detection System), which possesses an efficient near-duplicate matching scheme and a progressive update scheme. The progressive update scheme enables system Cosdes to keep the most up-to-date information for near-duplicate detection. We evaluate Cosdes on a live data set collected from a real e-mail server and show that our system outperforms the prior approaches in detection results and is applicable to the real world.
Key Terms: Spam detection, e-mail abstraction, near-duplicate matching.

I. INTRODUCTION

E-Mail communication is prevalent and indispensable nowadays. However, the threat of unsolicited junk emails, also known as spams, becomes more and more serious. According to a survey by the website Top Ten REVIEWS 40 percent of e-mails were considered as spams in 2006. The statistics collected by MessageLabs1 show that recently the spam rate is over 70 percent and persistently remains high. The primary challenge of spam detection problem lies in the fact that spammers will always find new ways to attack spam filters owing to the economic benefits of sending spams. Note that 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 Although the techniques used by spammers vary constantly, there is still one enduring feature: spams with identical or similar content are sent in large quantities and successively. Since only a small amount of e-mail users will order products or visit websites advertised in spams, spammers have no choice but to send a great quantity of spams to make profits. It means that even with developing and employing unexpected new tricks, spammers still have to send out large quantities of identical or similar spams simultaneously and in succession. This specific feature of spams can be designated as the near-duplicate phenomenon, which is a significant key in the spam detection problem In view of above facts, the notion of collaborative spam filtering with near-duplicate similarity matching scheme has recently received much attention. The primary idea of the near-duplicate matching scheme for spam detection is to maintain a known spam database, formed by user feedback, to block subsequent spams with similar content. Collaborative filtering indicates that user knowledge of what spam may subsequently appear is collected to detect following spams. Overall, there are three key points of this type of spam detection approach we have to be concerned about. First, an effective representation of e-mail (i.e., e-mail abstraction) is essential. Since a large set of reported spams has to be stored in the known spam database, the storage size of e-mail abstraction should be small. Moreover, the email abstraction should capture the near-duplicate phenomenon of spams, and should avoid accidental deletion of non spam e-mails (also known as hams). Second, every incoming e-mail has to be matched with the large database, meaning that the near-duplicate matching process should be substantially efficient. Finally, the latest

spams have to be included instantly and successively into the database so as to effectively block subsequent near-duplicate spams Although previous researchers have developed various methods on near-duplicate spam detection , these works are still subject to some drawbacks. To achieve 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. One major problem of these abstractions is that they may be too brief and thus may not be robust enough to withstand intentional attacks. A common attack to this type of representation is to insert a random normal paragraph without any suspicious keywords into unobvious position of an e-mail. In such a context, if the whole e-mail content is utilized for hash based representation, the near-duplicate part of spams cannot be captured. In addition, the false positive rate (i.e., the rate of classifying hams as spams) may increase because the random part of e-mail content is also involved in e-mail abstraction. On the other hand, hash-based text representation also suffers from the problem of not being suitable for all languages. Finally, images and hyperlinks are important clues to spam detection, but both of them are unable to be included in hash-based text representation. We explore to devise a more sophisticated email abstraction, which can more effectively capture the near duplicate phenomenon of spams. Motivated by the fact that email users are capable of easily recognizing similar spams by observing the layouts of e-mails, we attempt to represent each e-mail based on the e-mail layout structure. Fortunately, almost all e-mails nowadays are in Multipurpose Internet Mail Extensions (MIME) format with the text/html content type. That is, HTML content is available in an e-mail and provides sufficient information about e-mail layout structure. In view of this observation

1.1 Purpose

We propose the specific procedure Structure Abstraction Generation (SAG), which generates an HTML tag sequence to represent each e-mail. Different from previous works, SAG focuses on the e-mail layout structure instead of detailed content text. In this regard, each paragraph of text without any HTML tag embedded will be transformed to a newly defined tag Since we ignore the semantics of the text, the proposed abstraction scheme is inherently applicable to e-mails in all languages. This significant feature is superior to most existing methods. Once 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. Note that 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. Moreover, 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.

1.2 Scope

To the best of our knowledge, there is no prior research in considering e-mail layout structure to represent e-mails in the field of near-duplicate spam detection. In summary, the contributions of this paper are as follows:

1. We propose the specific procedure SAG 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.
2. We devise an innovative tree structure, Sp Trees, to store large amounts of the e-mail abstractions of reported spams. Sp Trees contribute to the accomplishment of the efficient near-duplicate matching with a more sophisticated e-mail abstraction.
3. We design a complete spam detection system Cosdes with an efficient near-duplicate matching scheme and a progressive update scheme. The progressive update scheme enables system Cosdes to keep the most up-to-date information for near duplicate detection.

1.3 Motivation

We devise an innovative tree structure, Sp Trees, to store large amounts of the e-mail abstractions of reported spams, and Sp Trees contribute to substantially promoting the efficiency of matching. In the design of the near-duplicate matching scheme based on Sp Trees, we aim at reducing the number of spams and tags which are required to be compared. By integrating above techniques, in this paper, we design a complete spam detection system COllaborative Spam Detection System (Cosdes). Cosdes possesses an efficient near-duplicate matching scheme and a progressive update scheme. The progressive update scheme not only adds in new reported spams, but 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. In addition, to withstand intentional attacks, a reputation mechanism is also provided in Cosdes to ensure the truthfulness of user feedback.

1.3.1 Definitions

The central 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 diverse. Unlike most prior works representing e-mails based mainly on content text, we 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.

The purpose of creating the <anchor> tag is to minimize the false positive rate when the number of tags in an e-mail abstraction is short. The less the number of tags in an e-mail abstraction, the more possible that a ham maybe matched with known spams and be misclassified as a spam Therefore, when the number of tags in an e-mail abstraction is smaller than a predefined threshold, for each anchor tag <a>, we specifically record the targeted domain name or e-mail address, which is a significant clue for identifying spams.

1.3.2 Abbreviations

1. Structure Abstraction Generation:

An automatic abstract generation system including a document structure analyzer is described. From a document, the system extracts a text structure representing rhetorical relations among sentences and sentence chunks. The system evaluates sentence importance based on the analyzed structure and decides which sentence should be discarded from an abstract. It also attempts to generate an abstract consistent with the original text by replacing connective expressions.

1.3.3 Model Diagram

Modules:

1. Abstraction Generation
2. Database Maintenance
3. Spam Detection

1.3.3.1 Abstraction Generation:

In this module we generate an email abstraction. Here we use SAG (Structure Abstraction Generation) procedure to generate the email abstraction. First read html/text content type based input mail. This module composed of three major phases.

1. Tag Extraction phase

In this phase we read input mail and get the each and tags. Transform each text into <mytext/> tag, add all the anchor tag and add the remaining tags. Preprocess the tag sequence.

2. Tag Reordering Phase

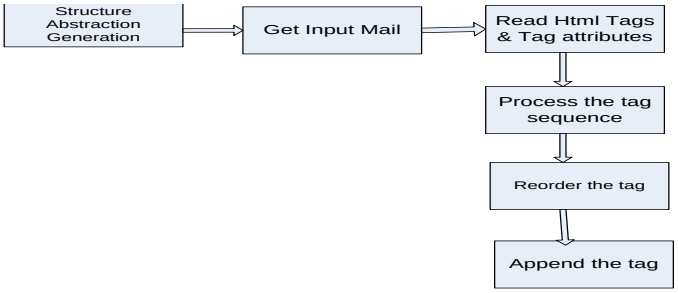
In this phase we reorder each and every tag. Assign the position number. Add all the tags with the position number (EA).

3. Appending Phase

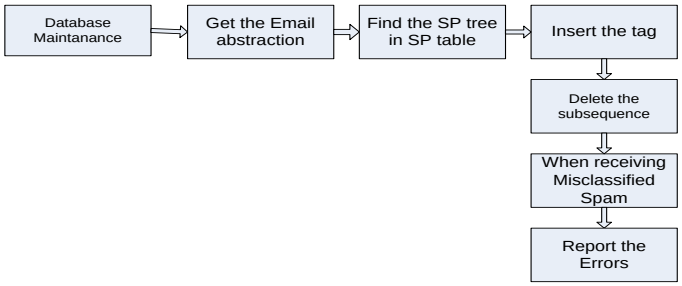
Append the anchor set in front of EA.H355344

Module Diagrams for each Module:

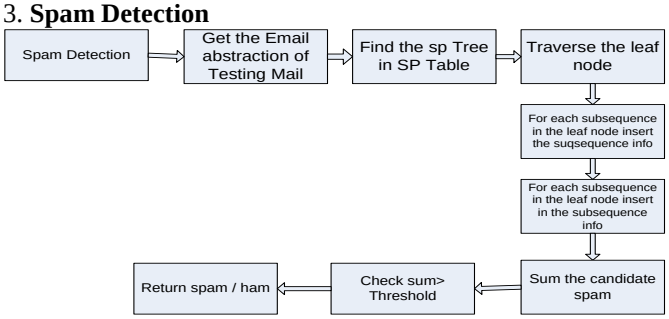
1. Tag extraction Phase:



2. **Database Maintenance:**



3. **Spam Detection**



II. IMPLEMENTATION

Based on what features of e-mails are being used, previous works on spam detection can be generally classified into three categories:

1) content-based methods,

2) Non content-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 and Support Vector Machines (SVMs) 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 and improved SVMs 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 mark field model , neural network and logic regression and certain specific features, such as URLs and images have also been

taken into account for spam detection. The other group attempts to exploit non content information such as e-mail header, e-mail social network , and e-mail traffic to filter spams. Collecting notorious and innocent sender addresses (or IP addresses) from e-mail header to create black list and white list is a commonly applied method initially. Mail Rank examines the feasibility of rating sender addresses with algorithm Page Rank in the e-mail social network, and in , modified version with update scheme is introduced. Since e-mail header can be altered by spammers to conceal the identity, the main drawback of these methods is the hardness of correctly identifying each user. In the authors intend to analyze e-mail traffic flows to detect suspicious machines and abnormal e-mail communication.

Existing System:

Various methods on near-duplicate spam detection have been developed. These works are still subject to some drawbacks. To achieve 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. One major problem of these abstractions is that they may be too brief and thus may not be robust enough to withstand intentional attacks. A common attack to this type of representation is to insert a random normal paragraph without any suspicious key-words into unobvious position of an e-mail. In such a context, if the whole e-mail content is utilized for hash-based representation, the near-duplicate part of spams cannot be captured. In addition, the false positive rate (i.e., the rate of classifying hams as spams) may increase because the random part of e-mail content is also involved in e-mail abstraction. On the other hand, hash-based text representation also suffers from the problem of not being suitable for all languages. Finally, images and hyperlinks are important clues to spam detection, but both of them are unable to be included in hash-based text representation.

2.2.1 Disadvantages of Existing System

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 language as been less studied yet.

The insertion of a randomized and normal paragraph can easily defeat this type of spam filters. Moreover, since the structures and features of different languages are diverse, word and substring extraction may not be applicable to e-mails in all languages

2.3 Proposed System

In this paper, we design a complete spam detection system COllaborative Spam DEtection System (Cosdes). Cosdes possesses an efficient near-duplicate matching scheme and a progressive update scheme. The progressive update scheme not only adds in new reported spams, but 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. In addition, to withstand intentional attacks, a reputation mechanism is also provided in Cosdes to ensure the truthfulness of user feedback.

2.3.1 Advantages of Proposed System

This advantageous property is verified with our data set that consists of 15 percent English e-mails and 80 percent Chinese ones. In addition, to further investigate the components of Cosdes, we evaluate the detection performance when either the sequence preprocessing step or the anchor-appending step of procedure SAG is removed. The FP rate increases to a certain unacceptable value, our system can simply response by slightly decreasing the value of Sth. The property of simple threshold setting is also an advantageous feature of Cosdes.

Algorithm:

The following Algorithms are used,

SAG Structured Abstraction Generation:

This algorithm is used to generate the e-mail abstraction using HTML content in e-mail. It is composed of three major phases, Tag Extraction Phase, Tag Reordering Phase, and <anchor> Appending Phase. In Tag Extraction Phase, the name of each HTML tag is extracted, and tag attributes and attribute values are eliminated. In addition, each paragraph of text without any tag embedded is transformed to <mytext/>. <anchor> tags are then inserted into AnchorSet, and the first 1,023 valid tags are concatenated to form the tentative e-mail abstraction. The following sequence of operations is performed in the preprocessing step.

1. Front and rear tags are excluded.
2. Nonempty tags that have no corresponding start tags or end tags are deleted. Besides, mismatched nonempty tags are also deleted.
3. All empty tags are regarded as the same and are replaced by the newly created <empty=> tag. Moreover, successive <empty=> tags are pruned and only one <empty=> tag is retained.
4. The pairs of nonempty tags enclosing nothing are removed.

Fig. 1. Algorithmic form of procedure SAG.

```
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
```

On purpose of accelerating the near-duplicate matching process, we reorder the tag sequence of an e-mail abstraction in Tag Reordering Phase. The main objective of appending <anchor> tags is to reduce the probability that a ham is successfully matched with reported spams when the tag length of an e-mail abstraction is short.

2. Cosdes System:

Cosdes deals with four circumstances by handlers. When receiving a reported spam it handles insertion. When receiving a testing mail it detect whether the mail is spam or ham. When receiving misclassified ham it handles the error report handler.

Fig. 2. Algorithmic form of system Cosdes

```
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
1  switch (circumstance)
2  case: when receiving a reported spam
3     if (EA.reporter.S_R > S_initial);
4        Trigger Insertion Handler(EA);
5        Increase S_R of the reporter in RepTable; // Rep: Reputation
6     break;
7  case: when receiving a testing email
8     Trigger Matching Handler(EA, S_th);
9     if (the testing email is classified as a spam);
10       Trigger Insertion Handler(EA);
11    break;
12 case: when receiving a misclassified ham
13    Trigger Error Report Handler(EA);
14    break;
15 case: for every T_d
16    Trigger Deletion Handler(T_m);
17    break;
End
```

3. Insertion Handler:

In the insertion handler the corresponding SpTree is found in SpTable according to the tag length of the inserted spam, and nowNode is assigned as the root of this SpTree. Insert the subsequences of the e-mail abstraction along the path from root to leaf. If nowNode is an internal node, the subsequence with 2^i tags is inserted into level

i. Meanwhile, the hash value of this subsequence is computed. Then, nowNode is assigned as the corresponding child node based on the type of the next tag. If the next tag is a start (end) tag, nowNode is assigned as the left (right) child node. Finally, when nowNode is processed to a leaf node, the subsequence with remaining tags is stored.

Fig. 3. Algorithmic form of Insertion Handler

Procedure of Insertion Handler
Input: *EA*: the email abstraction required to be inserted
1 Find the corresponding *SpTree* in *SpTable* according to *EA.tag_length*;
2 *nowNode* = *SpTree.root*;
3 **for** (*i* = 0 to *SpTree.height*)
4 **if** (*nowNode* is not a leaf node)
5 Insert the subsequence with 2^i tags;
6 Compute the hash value of this subsequence;
7 *nowNode* = the corresponding child node;
8 **else** // *nowNode* is a leaf node
9 Insert the subsequence with remaining tags;
10 Compute the hash value of this subsequence;
End

SCREEN SHOTS

Fig: Home page for the Project

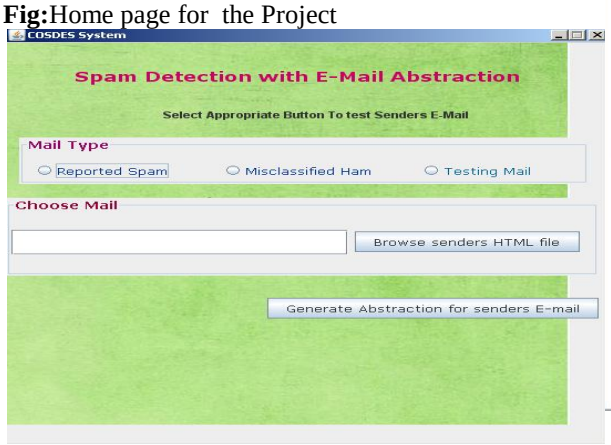


Fig: Select the Option for Testing the Html file as Input mail

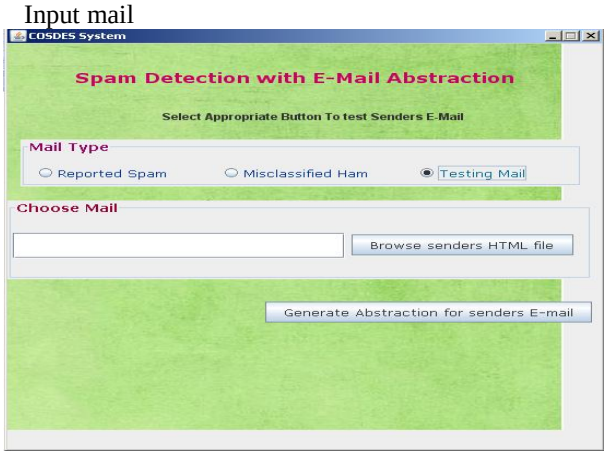


Fig: Testing the mail

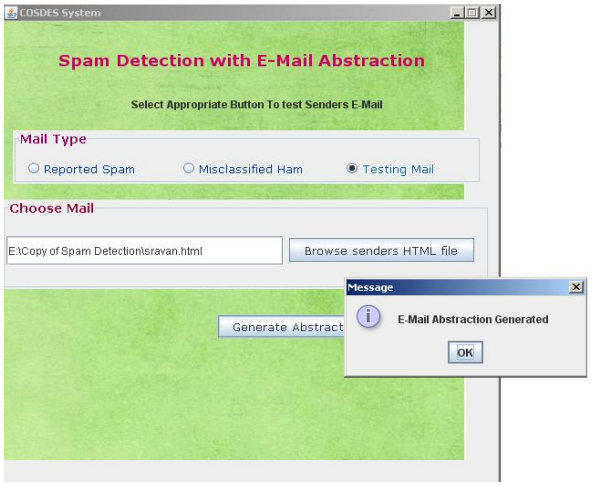


Fig: Select the Appropriate mail as Spam/Ham

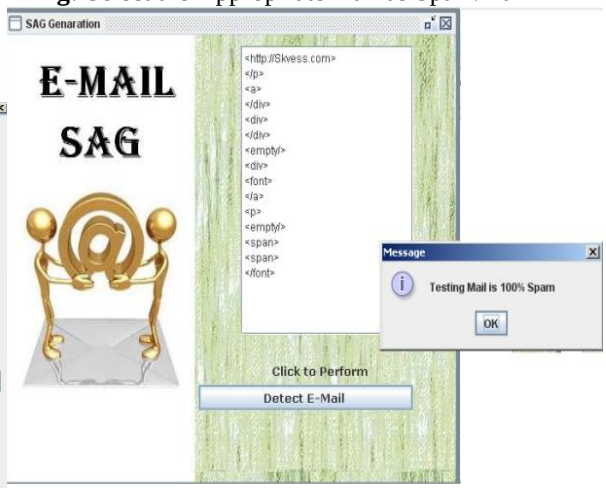


Fig: After Detection again detect the current status of spam mail

Fig: Abstraction for Misclassified Ham

Fig: Handling the Receiver's Ham Mail

Fig: Insertion of Subsequence tags to Receiver's Mail

III. Conclusion And Future Enhancements

In the field of collaborative spam filtering by near-duplicate detection, a superior e-mail abstraction scheme is required to more certainly catch the evolving nature of spams. Compared to the existing methods in prior research, in this paper, we explore a more sophisticated and robust e-mail abstraction scheme, which considers e-mail layout structure to represent e-mails. 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. Moreover, 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. In the experimental results, we show that Cosdes significantly outperforms competitive approaches, which indicates the feasibility of Cosdes in real-world applications.

REFERENCES:

[1] 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.

[2] 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.

[3] 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.

[4] 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.

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

[6] A.C. Cosoi, "A False Positive Safe Neural Network; The Followers of the Anatrim Waves," Proc. MIT Spam Conf., 2008.

[7] 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.

[8] 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.

[9] 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.

[10] H. Drucker, D. Wu, and V.N. Vapnik, "Support Vector Machines for Spam Categorization," Proc. IEEE Trans. Neural Networks, pp. 1048-1054, 1999. A. Kolcz and J. Alspector, "SVM-Based Filtering of Email Spam with Content-Specific Misclassification Costs," Proc. ICDM Workshop Text Mining.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

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

[16] K.M. Schneider, "Brightmail URL Filtering," Proc. MIT pam Conf.,2004.

[17] 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.

Author's Profile

Mrs. B.Venkata Ramana, Post Graduated in Computer Science (M.Tech), JNTUH, 2010, and Graduated in Computer Science & Engineering (B.Tech) From JNTU Hyderabad, 2005.She is working presently as an Assistant Professor in Department of Computer Science & Engineering in **Holy Mary Institute of Technology & Science**, RR Dist, A.P, INDIA. She has 6+ years Experience. Her Research Interests Include Software Engineering, Cloud Computing, Operating Systems & Information Security.

Mr. U.MAHENDER, Post Graduated in Computer Science & Engineering (M.Tech) ,HOLY MARY INSTITUTE OF TECHNOLOGY AND SCIENCE, Jawaharlal Nehru Technological University Hyderabad , In 2010 . And Bachelor Of Techonology (B.TECH), in Vidya Bharathi Institute Of Technology ,Jawaharlal Nehru Technological University, Hyderabad, in 2007. He is working presently as an Assistant Professor in Department of Computer Science & Engineering in **TKR COLLEGE OF ENGINEERING AND TECHNOLOGY** medbowli, meerpet, RR Dist, and A.P, INDIA. He has 3+ years Experience.

Mr R.V.GANDHI, Post Graduated in Computer Science & Engineering (M.Tech) , Jawaharlal Nehru Technological University Hyderabad , 2009 and Bachelor of Technology (B.Tech) in Computer Science & Engineering, Jawaharlal Nehru Technological University, Hyderabad , 2007. He is working presently as an Assistant Professor in Department of Computer Science & Engineering in **Mother Theresa Institute of Engineering and Technology**, Melumoi, Palamaner, Chitoor Dist, A.P, INDIA. He has 4+ years Experience.

Mr. Nalugotla Suman working as an Assistant Professor in the Department of Computer Science and Engineering at **Hi-Point college of Engineering and Technology**, Hyderabad, RR dist Andhra Pradesh, India. He has received M.Tech in Software Engineering from JNTUH, Hyderabad, Andhra Pradesh, India. He has 6 years of teaching experience. His research interests are Data Mining, Software Engineering, Computer Networks, High Performance Computing and Cloud Computing.