International Journal of Computer and Communication Technology

Volume 7 | Issue 4 Article 9

October 2016

IMPROVED NEAR DUPLICATE MATCHING SCHEME FOR E-MAIL SPAM DETECTION

M. SIVA KUMAR REDDY
Department of CSE, Madanapalli Institute of Technology and Science, Madanapalli, Andhra, Pradesh, India., msivakumarreddy@gmail.com

B. KRISHNA SAGAR
Department of CSE, Madanapalli Institute of Technology and Science, Madanapalli, Andhra, Pradesh, India., bkrishnasagar@gmail.com

IMPROVED NEAR DUPLICATE MATCHING SCHEME FOR E-MAIL SPAM DETECTION

M. SIVA KUMAR REDDY[1] & B. KRISHNA SAGAR[2]

[1,2]Department of CSE, Madanapalli Institute of Technology and Science, Madanapalli, Andhra, Pradesh, India.

Abstract - Today the major problem that the people are facing is spam mails or e-mail spam. In recent years there are so many schemes are developed to detect the spam emails. Here the primary idea of the similarity matching scheme for spam detection is to maintain a known spam database, formed by user's feedback, to block the subsequent near-duplicate spam's. 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. To detect fastly near duplicates and duplicate spam mails in Cosdes, we propose a new approach SimHash.

Keywords *- Spam mails, Emails, Near Duplicate SimHash, Spam Trees.*

I. INTRODUCTION

Internet is the most widely used area. In internet most widely used are E-mails. E-mails play a major role for the communication between the people .The people who are using emails cannot verify the duplicate and near duplicate web documents creating the more problems on the web search engines. These documents will increase the space required to store the index, slow down the searching results and the annoy users. According to the data availability on the internet, the huge data are shorts texts such that mobile phone short messages, instant messages, chat log, BBS titles etc.

The statistical information is given by the Information Industry Ministry of china that more than 1.56 billion mobile phone short messages are sent each day in Mainland China. You already know how much of email is spam, but here are a bunch of other factoids as per [9] you may not be aware of:

- **90%** of spam is in English. A year ago it was 96%, so spam is getting more "international."
- **88%** of all spam is sent from botnets (networks of compromised PCs).
- **91%** of spam contains some form of link.
- Unsolicited newsletters are increasing and are now the second most common type of spam.
- Spam from webmail services like Gmail and Hotmail isn't as common as you might think. Only **0.7%** of spam is sent from webmail accounts.
- **1 in 284** emails contain malware.
- **1 in 445** emails are phishing emails.
- As many as **95 billion** phishing emails were in circulation in 2010.
- Unfortunately, the status of duplicate and near duplicate messages is very complex. Among these especially near duplicates and spam mails.

These differences may result from several causes: 1) same contents appearing on different sites are all crawled, processed and indexed; 2) mistake introduced while parsing these loosely structured and noisy text (HTML page may contain ads., and it is known as shorting of semantics useful for parsing); 3) manual typos (all information on Internet are created by people originally) and manual revising while being referred and reused; 4) explicit modification to make the short message suitable for difference usage.

Checking may be applicable manually when the scale of repository is small. E.g. hundreds or hundreds or thousands of instances. When the amount of instances increases to millions and more, obviously, it becomes impossible for human beings to check them one by one, which is tedious, costly and prone to error. Resorting to computers for such kind of repeatable job is desired, of which the core is an algorithm to measure the difference between any pair of short messages, including duplicated and near duplicated ones.

In Section 2, we define near duplicate and the construction of SP Tree and in section 3 we describe how SimHash works, in section 4 SimHash advantages and disadvantages A brief review of conventional work is presented in Section 4, followed by conclusion in Section 5.

2. Preliminaries

2.1 Near Duplicate

Near-duplicate spam detection is to exploit reported spams and to subsequently block one which have similar content. The definition of similarity

between two e-mails are diverse for different forms of email. representing e-mails based mainly on content text, we represent 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.

Near-Duplicate:

Let I ={t1, t2, ...tn} be the set of valid HTML Tags with two types of newly created HTML tags <mytext/> and <anchor/>. An e-mail abstraction derived as <e1, e2, . . . , ei; . . . , em>, which is an ordered list of tags, where ei Є I.
The definition of near duplicate is: "Two e-mail abstractions A=<a1, a2, . . . ,ai, . . . , an> and B=<b1, b2, . . . , bi, . . . , bm> are viewed as
near-duplicate if for all ai= bi and n = m.

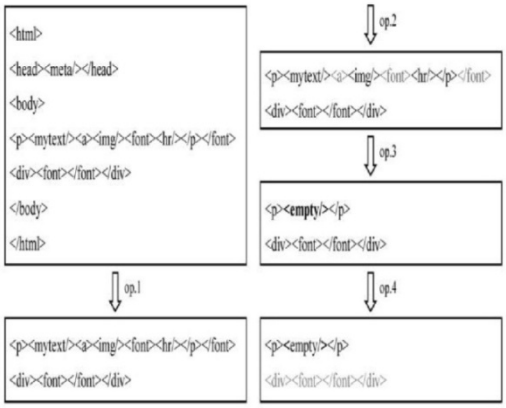
2.2 Related Works

Since the e-mail spam problem is increasingly serious various techniques have been explored to solve the problem. They can be categorized into the categories: 1) content-based methods,2) non content-based methods, and 3) others. Researchers analyze e-mail content text and model this problem as a binary text classification task. The solutions of this category are Naive Bayes, and Support Vector Machines (SVMs) methods. 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. Markov random 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 noncontent 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 blocked list and allowable mail list.

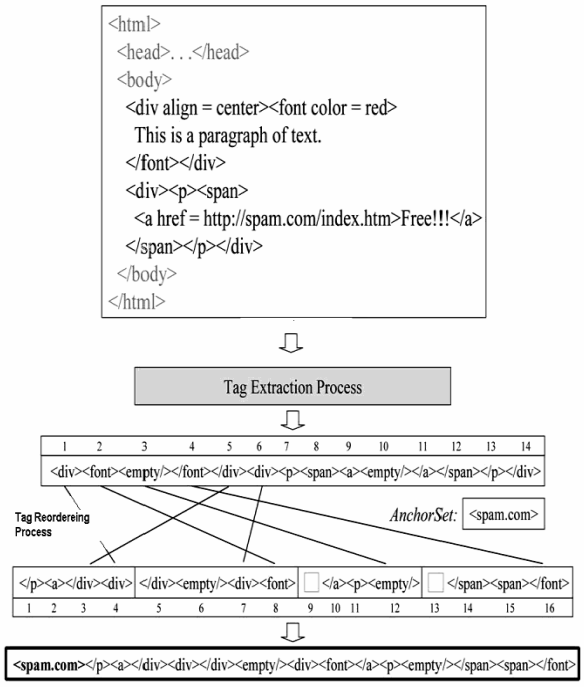
2.3 Structure Abstraction Generation

We propose the COSDES as a specific procedure SAG to generate the e-mail abstraction using HTML content in e-mail. Procedure SAG 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/>.

An example of the preprocessing step in Tag Extraction Phase of SAG.



Procedure flow of Structure Abstraction Generation

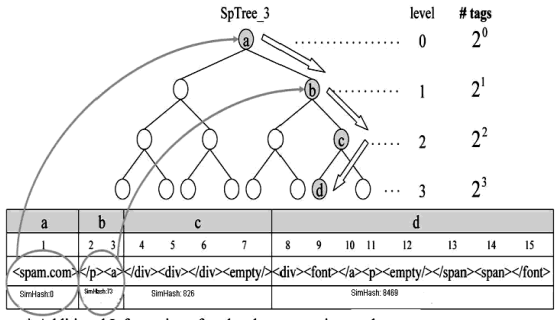


2.4 Design of Spam Tree

SP tree is a data structure to facilitate the process of near-duplicate matching. SpTable and SpTrees (Sp stands for spam) are proposed to store large amounts of the e-mail abstractions of reported spams. Several SpTrees are the kernel of the database, and the e-mail abstractions of collected spams are maintained in the corresponding SpTrees. According to near duplicate definition, two e-mail abstractions are possible to be near-duplicate only when the numbers of their tags are identical.

For efficient matching Sp Trees are designed to be binary trees. The branch direction of each SpTree is determined by a binary hash function. If the first tag of a subsequence is a start tag (e.g.,), this

subsequence will be placed into the left child node. A subsequence whose first tag is an end tag (e.g.,) will be placed into the right child node. Since most HTML tags are in pairs and the proposed e-mail abstraction is reordered in SAG, subsequences are expected to be uniformly distributed. Moreover, on level i of each SpTree (with the root on level 0), each node stores subsequences whose tag lengths are equal to 2i. For instance, as shown in Fig, the subsequence <spam:com> is placed into level 0, the subsequence </p><a> (whose tag length is 21) is placed into level 1, and so forth.



* Additional Information of each subsequence in a node:

- Internal nodes (e.g., a, b, c): *spam_id* , *timestamp*
- External nodes (e.g., d): *spam_id*, *user_id*, *timestamp*, *length_EA*, S_R

Figure: Illustration of SP Tree with an example

3. Near-Duplicate Detection by SimHash

Charikar's SimHash [4], actually, is a fingerprinting technique that produces a compact representation of the objects may be documents or images. So, it allows for various processing, once applied to original data sets, to be done on the compact sketches, a much smaller and well formatted (fixed length) space. With documents, SimHash works as follows: a Web document is converted into a set of features, each feature tagged with its weight. Then, we transform such a high dimensional vector into an *f bit* - fingerprint where *f* is quite small compared with the original dimensionality.

The calculation of the hash is performed in the following way:

1. Document is splitted into tokens (words for example) or super-tokens (word tuples)
2. Each token is represented by its hash value; a traditional hash function is used
3. Weights are associated with tokens
4. A vector V of integers is initialized to 0, length of the vector corresponds to the desired hash size in bits
5. In a cycle for all token's hash values (h), vector V is updated: ith element is decreased by token's weight if the ith bit of the hash h is 0, otherwise ith element is increased by token's weight if the ith bit of the hash h is 1
6. Finally, signs of elements of V correspond to the bits of the final fingerprint

Sample program to show how SimHash works:

```
public class HtmlSimhash {private static final
Logger LOG =
Logger.getLogger(HtmlSimhash.class );
public static void main(String[] args) {
Tap inputTap = new Hfs( new TextDelimited(new
Fields("docid","body"), " " ),args[0] );
Tap outputTap = new StdoutTap();
// create the flow
Flow simhashFlow = Simhash.simhash(inputTap,
outputTap, 1, HtmlText.tokenizer(3));
simhashFlow.complete(); // or add to your
Cascade, etc
}
}
```

In this paper, we show that SimHash is indeed effective and efficient in detecting both duplicate (with $k = 0$) and near-duplicate (with $k > 0$) (see the two typical examples in TABLE II.) among large short message repository. However, we also notice that due to the born feature of short messages, $k = 3$ may not be an Ideal parameter for. For example, as shown in TABLE III. , $k = 2$ is enough to detect the one-character difference, but k has to be 5 to detect the same pair of messages with two-character difference. Besides, with the same one-character difference, short messages require larger k for effective detection. This may be explained by an observation, that the same difference, e.g. having one different character on the same position of two spam messages, would be more influential to short text than to long text.

This is a paper focusing on discussing Solution for real application. Firstly, we demonstrate a series of practical values of SimHash-based approach by experiments and our experience.

Secondly, we point out that $k = 3$ may be suitable for near-duplicated spam mail detection, but obviously not suitable for short messages.
Thirdly, we propose one empirical choice, $k = 5$, as applied on our Online short message search.

Table I. Typical Near-Duplicates of Spam Mails with Differences Highlighted in grey

1.International Monetary Fund congratulate you as our Ten(10) Star Prize Winner in our 2011 End of Year IAP held in London.This makes you a cash prize of £750,000.00 GBP
2. IMF congratulate you as our Ten(10) Star Prize Winner in our 2011 End of Year IAP held in London.This makes you a cash prize of £750,000.00 GBP

Table II. Example: detect duplicate with 0 k = and near-duplicate with 0 k > (with differences highlighted in gray)

K=0	1.Great Opportunity -- IT Professionals only IIPM LOOKING FOR INDIAN PROFILES 2.Great Opportunity -- IT Professionals only IIPM LOOKING FOR INDIAN PROFILES
K>0	1. Your e-mail has won you, (£750,000.00.Pounds) from COCA COLA NATIONAL LOTTERY On our 2011 charity bonanza 2. Your e-mail has won you, ($750,000.00.Dollors) from COCA COLA NATIONAL LOTTERY On our 2011 charity bonanza

Table III. Example: detect same long text but more diffrence requires larger *k* (with differences highlighted in gray)

K=2	1.We are Pleased to inform you that you have won a prize money of GBP750,000.00 2.We are Pleased to inform you that you have won a prize money of INR750,000.00
K=5	1) Your e-mail address attached to Winning number 20-12jan-2010-02MSW, serial number S/N-00168, drew the lucky numbers 887-13-866-37-10-83 (2) Your e-mail address attached to Winningnumber20-12DEC-2010-02MSW, serial number S/N-00168, drew the lucky numbers 887-13-865-37-10-83

Table IV. Example: detect same diffrence but shorter text requires larger k (with differences highlighted in gray)

K=2	1. Your e-mail has won you, (£750,000.00.Pounds) from COCA COLA NATIONAL LOTTERY On our 2011 charity bonanza 2. Your e-mail has won you, ($750,000.00.Dollors) from COCA COLA NATIONAL LOTTERY On our 2011 charity bonanza
K=5	1.Great Opportunity --- IT Professionals only IIPM seeing FOR INDIAN PROFILES ! 2.Great Opportunity -- IT Professionals only IIPM LOOKING FOR INDIAN PROFILES *

4. Advantages and Disadvantages of SimHash

SimHash has several advantages for application based on our experience:

1. Transforming into a standard fingerprint makes it applicable for different media content, no matter text, video or audio;
2. Fingerprinting provides compact representation, Which not only reduces the storage space greatly? but allows for quicker comparison and search.
3. Similar content has similar SimHash code, which permits easier distance function to be? determined for application.
4. It is applicable for both duplicate and near duplicate Detection, with $k = 0$ and $k > 0$ respectively.
5. Similar processing time for different setting of k if via the proposed divide-and-search mentioned above, and this is valuable for practice since we are able to detect more near duplicates with no extra cost.
6. The search procedure of similar encoded objects is easily to be implemented in distributed environment based on our implementation experience.
7. From the point of software engineering view, this procedure may be implemented into standard module and be re-used on similar applications, except that the applicants may determine the related parameters themselves.

5. Challenges to Detect Spam E-Mails

In this day and age, spammers are becoming more and more sophisticated. They are finding ways to trick people into thinking their unsolicited junk messages are worth the time you spend reading them. While many users are savvy enough to figure out what's real and what's bogus among their electronic correspondence, there are many out there who take what they receive at face value and open it.

This is alright though because sometimes the electronic junk mail swindlers are clever enough to pull the wool over our eyes. It's in the best interests of your computer's health and your sanity to research how to tell if an email is spam or genuine. We researched this topic extensively and generated a list of the top five ways to tell if an email is spam. These rules can help you when spam slips through the protection of your Spam filter.

Here are the some of the following list:

If it ends up in Spam Folder:

You might be reading this entry and thinking "Duh!" But you would be surprised how many people go rummaging through their spam folder like there's something they need in there. Unless you accidentally categorized legitimate emails as spam, you can be pretty sure that all the emails you need will appear in your inbox. Sometimes emails from certain websites end up in the spam folder. You must deal with those on a case-by-case basis to determine whether or not they're legitimate of pushing garbage into your inbox.

Look at the Email Address:

Legitimate companies send emails through a server based out of their company website (for example, support@microsoft.com). If you see a long string of numbers in front of the @ sign or the name of a free email service before the .com (or any other domain),

you need to question the legitimacy of the email in question.

Look at the Content:
Keep an eye out for emails that say you need to do something right at that second or within a certain number of hours. Also, be wary of any emails that include links. Most companies tell you what to do, but they never direct you to where to do it with a link. Finally, rampant grammatical and spelling errors within the body of an email are good signs that it's spam. Spammers don't care enough about the actual messages they're sending to take the time to make them make sense.

If it asks for personnel Information:
Most institutions you deal with come right out and say they're never going to ask for personal information in an email. They don't need to ask you for your personal information anyway because they usually have it on hand. So, if you get an email that asks you for any personal information, no matter how legitimate it might seem, delete it right away. Personal information is only meant to be entered in secure, encrypted forms, not emails where anyone and everyone can get their hands on your information.

Look at the Greeting:
When you receive a genuine email, the sender addresses you directly, using either your first or last name. If you receive an email where they refer to you as a "Valued Customer" or as a member of some company, its spam. Senders of your genuine emails want to get your attention, so they always address you directly. We don't know about you, but when we read "Dear Valued Customer," our eyes begin to glaze over and our mouse cursor can't drag it to the trash fast enough.

VI. CONCLUSION AND FUTURE WORK

Uses an innovative tree structure, SpTrees, to store large amounts of the e-mail abstractions of reported spams. To achieve efficient matching with balanced tree structure, SpTrees are designed to be binary trees. The branch direction of each SpTree is determined by a binary hash function.

The improvement is limited since we map each subsequence in a node of an SpTree to a hash value. Therefore, the subsequences that have some prefix tags in common still can be differentiated with one comparison. In this paper, Instead of mapping each subsequence in a node of an SpTree to a hash value using a binary hash function we propose to replace it with a special hash function, namely Simhash.
The advantage of this over other hash functions is that it sets a minimum on the number of members that the two sets must share in order to match. This mitigates the effect of extremely common set members on data clusters.

SimHash based approach is Fast, Flexible, Customizable (HtmlSimhash), Scalable and is patented.

ACKNOWLEDGEMENT

We are greatly delighted to place my most profound appreciation to all my friends and faculty for encouragement and kindness in carrying out the paper. Their pleasure nature, directions, concerns towards us and their readiness to share ideas rejuvenated our efforts towards our goal. We also thank the anonymous references of this paper for their valuable comments.

REFERENCES

[1] Chi-Yao Tseng, Pin-Chieh Sung, and Ming-Syan Chen "Cosdes: A Collaborative Spam Detection System with a Novel E-Mail Abstraction Scheme" IEEE transactions on knowledge and data engineering, vol. 23, no. 5, may 2011

[2] 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.

[3] 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.

[4] 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.

[5] 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.

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

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

[8] 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.

[9] http://royal.pingdom.com/2011/01/19/ email-spam-statistics/

◈◈◈