Clustering Spam Emails into Campaigns

Mina SheikhAlishahi, Mohamed Mejri and Nadia Tawbi
Department of Computer science, Université Laval, Québec City, Canada

Keywords: Clustering Algorithm, Spam Emails, Machine Learning, Spam Campaign Detection.

Abstract: Spam emails constitute a fast growing and costly problems associated with the Internet today. To fight effectively against spammers, it is not enough to block spam messages. Instead, it is necessary to analyze the behavior of spammer. This analysis is extremely difficult if the huge amount of spam messages is considered as a whole. Clustering spam emails into smaller groups according to their inherent similarity, facilitates discovering spam campaigns sent by a spammer, in order to analyze the spammer behavior. This paper proposes a methodology to group large sets of spam emails into spam campaigns, on the base of categorical attributes of spam messages. A new informative clustering algorithm, named Categorical Clustering Tree (CCTree), is introduced to cluster and characterize spam campaigns. The complexity of the algorithm is also analyzed and its efficiency has been proven.

1 INTRODUCTION

Nowadays, the problem of receiving spam messages leaves no one untouched. According to McAfee (Labs, 2009) spam emails, also known as junk or unsolicited emails, constitute up to 90 percent of total amount of email messages. Based on Internet Threats Trend Report in the first quarter of 2012, an average of 94 billion spam emails were sent per day (Report, 2012). In the same year, Microsoft and Google (Rao and Reiley, 2012) estimate spam emails cost to American firms and consumers up to 20 billion dollars per year. Spam emails cause problems, from direct financial losses to misuses of traffic, storage space and computational power.

Given the relevance of the problem, several approaches have already been proposed to tackle this issue. Currently, the most used approach for fighting spam emails consists in identifying and blocking them (Carreras et al., 2001), (Drucker et al., 1999), (Seewald, 2007), on the recipient machine through filters, which generally are based on machine learning techniques or content features (Blanzieri and Bryl, 2008), (Tretyakov, 2004a), (Tretyakov, 2004b). Alternative approaches are based on the analysis of spam botnets, trying to find bot-masters and neutralizing the main vector of spam emails (John et al., 2009),(Leontiadis, 2011),(Xie et al., 2008), (Zhao et al., 2009).

Though some mechanisms to block spam emails already exist, spammers still impose non negligible cost to users and companies (Rao and Reiley, 2012). Thus, the analysis of spammers behavior and the identification of spam sending infrastructures is of capital importance in the effort of defining a definitive solution to the problem of spam emails.

Such an analysis, which is based on structural dissection of raw emails, constitutes an extremely challenging task, due to the following factors:

- The amount of data to be analyzed is huge and growing too fast every single hour.
- Always new attack strategies are designed and the immediate understanding of such strategies is paramount in fighting criminal attacks brought through spam emails (e.g. *phishing*).

To simplify this analysis, huge amount of spam emails should be divided into spam campaigns. A *spam campaign* is the set of messages spread by a spammer with a specific purpose (Calais et al., 2008), like advertising a specific product, spreading ideas, or for criminal intents e.g. phishing. Grouping spam messages into *spam-campaigns* reveals behaviours that may be difficult to be inferred when we look at a large collection of spam emails as a whole (Song et al., 2011). According to (Calais et al., 2008), in order to characterize the strategies and traffic generated by different spammers, it is necessary to identify groups of messages that are generated following the same procedure and that are part of the same campaign.

It is noteworthy to be mentioned that the problem of grouping a large amount of spam emails into smaller groups is an *unsupervised learning* task. The reason is that there is no labeled data for training a classifier in the beginning. More specifically, *supervised learning* requires classes to be defined in advance and the availability of a training set with elements for each class. In several classification problems, this knowledge is not available and unsupervised learning is used instead. The problem of unsupervised learning refers to trying to find hidden structure in unlabeled data (Ghahramani, 2004). The most known unsupervised learning methodology is *clustering*. Clustering is an unsupervised learning methodology that divides data into groups (clusters) of objects, such that object in the same group are more similar to each other than to those in other groups (Jain et al., 1999).

However, dividing spam messages into spam campaigns is not a trivial task due to the following reasons:

- Spam campaign types (classes) are not known beforehand.
- Feature extraction is difficult. Finding the elements that best characterize an email is an open problem addressed differently in various research works (Fette et al., 2007), (Bergholz et al., 2008), (Zhang et al., 2009), (Song et al., 2011).

For these reasons the most used approaches to classify spam emails is clustering them on the base of their similarities (Anderson et al., 2007), (Ramachandran and Feamster, 2006), (Song et al., 2011). However, the accuracy of current solutions is still somehow limited and further improvements are needed. While some categorical attributes, for example the language of spam message, are primary, discriminative and outstanding characteristics to specify a spam campaign, nevertheless in previous works (Kreibich et al., 2009), (Li and Hsieh, 2006), (Anderson et al., 2007), (Song et al., 2010a), (Song et al., 2010b),(Wei et al., 2008), (Calais Guerra et al., 2009), these categorical features are not considered, or the homogeneity of resulted campaigns are not on the base of these features.

In this short paper, after a thorough literature review on the clustering and classification of spam emails, we propose a preliminary work on the design of a *categorical* clustering algorithm for grouping spam emails, which is based on structural features of emails like language, number of links, email size etc. The rationale behind this approach is that two messages in the same format, i.e. similar language, size, same number of attachments, same amount of links, etc., are more probable to be originated from the same source, belonging thus to the same campaign. To this aim, we expect to extract categorical features (attributes) from spam emails, which are representative of their structure and that should clearly shape the differences between emails belonging to different campaigns.

The proposed clustering algorithm, named as *Categorical Clustering Tree* (CCTree), builds a tree starting from whole set of spam messages. At the beginning, the root node of the tree contains all data points, which constitutes a skewed dataset where non related data are mixed together. Then, the proposed clustering algorithm divides data points, step-by-step, clustering together data that are similar and obtaining homogeneous subsets of data points. The measure of similarity of clustered data points at each step of the algorithm is given by an index called *node purity*. If the level of purity is not sufficient, it means the data points belonging to this node are not sufficiently homogeneous and they should be divided into different subsets (nodes) based on the characteristic (attribute) that yields the highest value of *entropy*. The rationale under this choice is that dividing data on the base of the attribute which yields the greatest entropy helps in creating more homogeneous subset where the overall value of entropy is consistently reduced. This approach, aims at reducing the time needed to obtain homogeneous subsets. This division process of non homogeneous sets of data points is repeated iteratively till all sets are not sufficiently pure or the number of elements belonging to a node is less than a specific threshold set in advance. These pure sets are the leaves of the tree and will represent the different spam campaigns.

The usage of categorical attribute is crucial for the proposed approach, which exploits the Shannon Entropy (Shannon, 2001), which yields good results with categorical attributes.

After detailing the CCTree algorithm and briefly presenting categorical features for categorizing spam emails, we will discuss the algorithm efficiency proving its linear complexity.

The rest of the paper is structured as follows. Section 2 reports a literature review concerning the previous techniques used for clustering spam emails into campaigns. In Section 3, we describe the proposed methodology for clustering spam messages based on CCTree. In Section 4 the analysis of the proposed methodology is discussed. Finally, Section 5 is a brief conclusion and a sketch of some future directions.

2 RELATED WORK

To the best of our knowledge just a few works exist related to the problem of clustering spam emails into campaigns.

In (Kreibich et al., 2009), the basic idea for identifying campaigns is keywords that stand for specific types of campaign. In this study, at first campaigns are found manually based on keywords and then some interesting results are extracted from groups of campaigns. As the result of needing manual scanning of spam, it is not suitable to be used for large amount of data set.

In (Gao et al., 2010) although the authors focus on analysis of spam URLs in Facebook, their study of URLs and clustering spam messages is similar to our goal concerning spam emails. First, all wall posts that share the same URLs are clustered together, then the description of wall posts are analyzed and if two wall posts have the same description their clusters are merged.

In Li et al. (Li and Hsieh, 2006), the authors believe that spam emails with identical URLs are highly clusterable and mostly sent in burst. In their method, if the same URL exists in spam emails from source A and source B, and each has a unique IP address, they will be connected with an edge to each other and the connected components are the desired clusters.

Spamscatter (Anderson et al., 2007) is a method that automatically clusters destination web sites extracted from URLs in spam emails with the use of image shingling. In image shingling, images are divided into blocks and the blocks are hashed, then two images are considered similar if 70 percent of the hashed blocks are the same.

In (Zhang et al., 2009) the spam emails are clustered based on their images to trace the origins of spam emails. They are visually resembled if their illustrations, text, layouts, and/or background textures are similar.

In (Song et al., 2010a), J. Song et al. focus on clustering spam emails based on IP addresses resolved from URLs inside the body of these emails. Two emails belong to the same cluster, if their IP address sets resolved from URLs are exactly the same.

In previous work pairwise comparison of each two emails is required for finding the clusters. This kind of comparison has two problems: the time complexity is quadratic, which is not suitable for big data clustering, and furthermore finding clusters is based on just one or two features of messages, which causes the decreasing of precision. In what follows, spam emails are grouped with the use of clustering algorithms.

In (Song et al., 2011) the same authors of (Song et al., 2010a) mentioned that only considering IP addresses resolved from URLs is insufficient for clustering. Since web servers contain lots of Web sites with the same IP address, so each IP cluster in (Song et al., 2010a) consists of a large amount of spam emails sent by different controlling entities.

Thus, the authors clustered spam emails by IP addresses resolved from URLs in their new method, called O-means clustering, which is based on K-means clustering method. The distance is based on 12 features in the body of an email which are expressed by numbers and the euclidian distance is used to measure the distance between two emails.

In (Song et al., 2010b) after clustering spam emails according to O-means method, the authors found that 10 largest clusters had sent about 90 percent of all spam emails in their data set. Hence, the authors investigate these 10 clusters to implement heuristic analysis for selecting significant features among 12 featurs used in previous work. As a result they select four most important features which could effectively separate these 10 clusters from each other. Since the idea for clustering is based on k-means clustering, computationally NP-hard algorithms. Also it requires the number of clusters to be known from beginning.

In (Wei et al., 2008) Wei et al. focus on a set of eleven attributes extracted from messages to cluster spam emails. Two clustering methods have been used: the agglomerative hierarchical algorithm clusters the whole data set. Next, for some clusters containing too many emails, *the connected component with weighted edges algorithm* is used to solve the problem of false positive rate.

With the use of agglomerative clustering (Han et al., 2011) a global clustering is done based on common features of email attributes. In the beginning, each email is a cluster by itself and then clusters sharing common features are merged. In this model, edges connect two nodes (spam emails) based on the eleven attributes. The desired clusters are the connected components of this graph with the weight above a specified threshold. This method suffers from not being useful for large data set. For each pair of spam messages, it is needed to compare eleven attributes to find the connectivity. This means pairwise comparison is required eleven times for each pair of messages. The basic hypothesis in FP-Tree method (Calais et al., 2008) for clustering spam emails is that some parts of spam messages are static in the view of recognizing a spam campaign. In this work as an improvement of (Li and Hsieh, 2006), just URLs are not considered for clustering. For identifying spam campaigns, Frequent Pattern Tree (FP-Tree) as a signature based method, is constructed from some features extracted from spam emails. These features are: language of email, message layout, type of message, URL and subject. In this tree, each node after root depicts a feature extracted from the spam message that is shared by the sub-trees beneath. Thus, each path in this tree shows sets of features that co-occur in messages, with the property of non-increasing order of frequency of occurrences. FP-Tree is not an incremental method, i.e. when a new point is added to previous clustered data points, the process of clustering needs to be run from beginning. In their new method (Calais Guerra et al., 2009) named incremental FP-Tree, the frequencies are calculated when a new spam email is added. The point is inserted in the tree according to features it respects. The problem is that the properties of FP-Tree may not maintain any more; i.e. a parent node may become less frequent than its child. A set of operations "join", "division", "division with join" and "permutation" are suggested to reconstruct the tree periodically. In this model, when a point is added a traversal of the tree is required and then one of the suggested operations is executed. The problem of FP-Tree is that it is based on frequency of features rather than creating pure clusters in terms of homogeneity. The redundant features also are removed for specifying a campaign according the frequency property, while in our method redundant features are characterized based on purity or homogeneity of campaigns.

In summary, the previous works for clustering spam emails mainly could be divided in two categories: First group focus on pairwise comparison of each pair of emails, for example URL comparison, and second group that in which a clustering algorithm is used, for example O-means clustering. In general, the aforementioned previous works suffer from one of the following problems: 1) They consider one or two features for grouping spam messages, which decreases the accuracy, 2) The pairwise comparison is used, with quadratic time complexity, 3) The number of clusters is required as a former knowledge, 4) The features which create a pure cluster, could not be identified. In our proposed algorithm, we try to solve these problems.

3 THE PROPOSED METHODOLOGY

In this section we will present the Categorical Clustering Tree (CCTree) algorithm. The design goals that motivates the CCTree algorithm can be summarized as follows:

- The clustering algorithm should produce an informative structure which could be expressed easily and understood even by people who have no knowledge of the algorithm designs. This helps to shape the structure of each cluster, and then to infer the key factor of each cluster. For our problem, this helps shaping spam campaigns.
- There is no former knowledge about the final or desired number of clusters.
- Redundant features for specifying a campaign are removed
- The algorithm is robust and efficient even when analyzing very large set of data.

Hierarchical clustering algorithms are informative, self explicative structure, do not require to specify beforehand the number of clusters, and are mostly deterministic. Thus, it seems this category of clustering algorithms fulfills our goals.

The CCTree algorithm is hierarchical, thus, before detailing it, we will briefly report some preliminary notions on hierarchical clustering algorithms. Hierarchical clustering algorithms are divided into two categories:

- **Divisive Hierarchical Clustering:** top-down approach, which starts at the root with all data points and splits the tree. is more efficient compared
- **Agglomerative hierarchical clustering:** is a bottom up approach in which each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.

The divisive hierarchical clustering is more efficient than agglomerative (Cimiano et al., 2004) because of the lesser amount of needed comparisons. To be precise divisive clustering complexity is generally $O(n^2)$ while agglomerative clustering is $O(n^3)$. Thus, we propose a divisive hierarchical clustering algorithm with an easily expressed structure.

In what follows, we present our proposed methodology named Categorical Clustering Tree (CCTree).

3.1 Categorical Clustering Tree Construction (CCTree)

The general idea for constructing the proposed Categorical Clustering Tree (CCTree) comes from supervised learning algorithms known as *Decision Trees* (Quinlan, 1986), (Quinlan, 1993).

To create the CCTree, a set of objects is given in which each data point is described in terms of a set of categorical attributes, e.g. the language of a message. Each attribute represents the value of an important feature of data and is limited to assume a set of discrete, mutually exclusive values, e.g. the *Language* as an attribute can take its values or features as *English* or *French*. Then, a tree is constructed in which the leaves of the tree are the desired clusters, while other nodes contain non pure data needing an attribute-based test to separate them. The separation is shown with a branch for each possible outcome of the specific attribute values. Each branch or edge extracted from that parent node is labeled with the selected value which directs data to the child node. The attribute for which the Shannon entropy is maximum is selected to divide the data based on it. A *purity function* on a node, based on Shannon entropy, is defined. Purity function represents how much the data belonging to a node are homogeneous. A required threshold of node purity is specified. When a node purity is equal or greater than this threshold, the node is labeled as a leaf or terminal node.

The precise process of CCTree construction can be formalized as follows.

- **Input:** Let D be a set of data points, containing N tuples on a set A of d attributes and a set of stop conditions S. These parameters are explained as follows.

  Attributes. A set of d attributes $\{A_1, A_2, \ldots, A_d\}$ is given, where each attribute is a set of mutually exclusive values. Thus, the $i'th$ attribute could be written as $A_i = \{v_1^i, v_2^i, \ldots, v_{r_i}^i\}$, where r_i is the number of features of attribute A_i.

  For example A_i could be the *Language* of spam email, and the set of possible values is {*english, french, spanish, other*}.

  Data Points. A set D of N data points is given, where each data is a vector whose elements are the values of attributes, i.e. $D_i = (v_{i_1}^1, v_{i_2}^2, \ldots, v_{i_d}^d)$ where $v_{i_k}^k \in A_k$. For example we have: *spam 1 = (English, excel attachment, image based)*.

  Stop Conditions. A set of stop conditions S = $(\{M, \varepsilon\})$ is given where M is an upper bound for the number of points in a node to be considered as a leaf, and ε represents the minimum desired purity for each cluster. When a node purity is greater than ε, it will be labeled as a terminal node or leaf.

  To calculate node purity or measuring how much the data belonging to a node are homogeneous, the purity function is formally defined as bellow.

  In node i, the ratio of data respecting $k'th$ value of $j'th$ attribute, N_{kj}, divided by N_i, the total number of data belonging to node i, is extracted, shown by $p(v_{kj_i}) = \frac{N_{kj}}{N_i}$. Then the purity of node i, denoted by $\rho(i)$, is equal to:

  $$\rho(i) = -\sum_{j=1}^{d}\sum_{k=1}^{t_j} p(v_{kj_i})log(p(v_{kj_i})) \quad (1)$$

  where d is the number of attributes describing the elements and t_j is the number of features of $j'th$ attribute.

- **Output:** A set of clusters which are the leaves of the categorical clustering tree.

To formally explain the process of creating the tree, some definitions need to be introduced:

Distribution of Attribute Values. The distribution of the values of attribute $A_j = \{v_1^j, v_2^j, \ldots, v_{r_j}^j\}$ in node i is equal to:

$$D(A_{j_i}) = \{\ p(v_{1_i}^j), p(v_{2_i}^j), \ldots, p(v_{(r_j)_i)}^j)\}$$

Where $p(v_i^{kj}) = \frac{N_{kj}}{N_i}$, N_{kj} represents the number of elements in node i having feature v^k of attribute A_j, and N_i is the number of elements of node i.

Shannon Entropy. In information theory, entropy is a measure of the uncertainty of a random variable. More specifically the Shannon entropy (Shannon, 2001), as a measure of uncertainty, for a random variable X with k outcomes $\{x_1, x_2, \ldots, x_k\}$ is defined as follows:

$$\text{H(X)} = -\sum_{i=1}^{k} p(x_i)logp(x_i)$$

where $p(x_i) = \frac{N_i}{N}$, N_i is the number of outcomes of x_i, and N is total number of elements of X. The amount of Shannon entropy is maximal when all outcomes are equally likely.

Actually, Shannon entropy shows the purity or homogeneity of a set of data points.

We report in the following the process of creating the CCTree:

At the beginning all data points, as the set of N tuples, are assigned to the root of the tree. Root is the first new node. The clustering process is applied iteratively for each new created node.

For each new node of the tree, the algorithm checks if the stop conditions are verified and if the number of data points is less than a threshold M, or the purity, defined in (1), is less than or equal to ε. In this case, the node is labeled as a leaf, otherwise the node should be split.

In order to find the best attribute to be used to divide the data in a node, the Shannon entropy based on the distribution of each attribute values is calculated. The attribute for which the Shannon entropy is maximal is selected. The reason is that the attribute which has the most equiprobable distribution of values, generates the highest amount of chaos (non homogeneity) in a node. For each possible value of the selected attribute, a branch is extracted from the node, with the

label of that value, directing the data respecting that value to the corresponding child node. Then the process is iterated until each node is either a parent node or is labeled as a leaf. At the last step all final nodes or leaves of the tree are the set of desired clusters, named $\{C_1, C_2, \ldots, C_k\}$.
Figure 1 depicts an example of a small CCTree, whilst a formal description of algorithm is given in Algorithm 1.

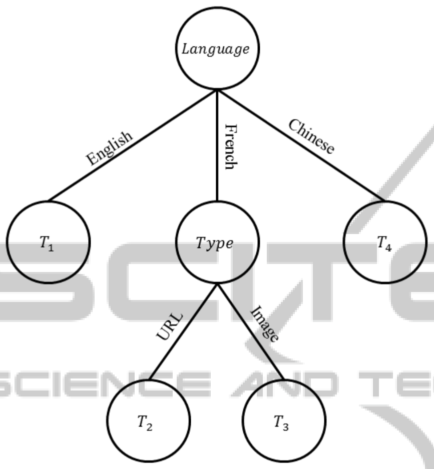

Figure 1: A small Categorical Clustering Tree (CCTree).

```
Input: Input: Data points $D_k$ , Attributes $A_l$,
       Attribute Values $V_m$,
       $node\_purity\_threshold$, $max\_num\_elem$)
Output: Clusters $C_k$
Root node $N_0$ takes all data points $D_k$
for each node $N_i$!=leaf node do
    if $node\_purity_i < node\_purity\_threshold ||$
    $num\_elem_i < max\_num\_elem$ then
        Label $N_i$ as leaf;
    else
        for each attribute $A_j$ do
            if $A_j$ yields max Shannon entropy
            then
                use $A_j$ to divide data of $N_i$;
                generate new nodes $N_{i_1}, \ldots, N_{i_t}$
                with $t$ = size of $V$ for attribute
                $A_j$ ;
            end
        end
    end
end
```

Algorithm 1: Categorical Clustering Tree (CCTree) algorithm.

3.2 Application to Spam Emails

To apply the proposed methodology to spam emails, a set of categorical features is extracted, which are representatives of the email structure. A preliminary list of extracted attributes is the following:

- **Language:** The language in which the email is written.
- **Links Amount:** Number of links found in the email. Five ranges (1 to 5, 5 to 10, 10 to 20, 20 to 50, more than 50) are used as values for this attribute.
- **Email Size:** The size of the raw email. Divided in 5 categories with the same rationale of the former attribute.
- **Number of Images:** Number of images in the body of spam messages, with same rational of the former attribute.
- **HTML:** boolean attribute. True if the email contains html tags.
- **Number of Attachment:** Number of attachments is selected from the set {1, 2, 3, 4, more}.
- **Attachment Type:** The attachment type is selected from the set {pdf, excell, word, no attachment, other}

The CCTree algorithm divides the emails in clusters on the base of formerly described features. Implementation of the proposed algorithm for testing and addition of other features will be object of future works.

4 COMPLEXITY OF THE PROPOSED METHODOLOGY

The proposed structure-based methodology for clustering spam emails into campaigns, while respects the aforementioned requirements of our problem, is linear in terms of complexity. This property becomes more impressive when it is compared with the complexity of previous works for grouping spam emails into campaigns, which are mostly based on pairwise comparison of spam messages, suffering from quadratic time complexity.

Here, we briefly discuss the precise time complexity of the proposed methodology. Let us consider n as the number of the whole data set, n_i the number of elements in node i, m the total number of features, v_l the number of features of attribute A_l, r the number of attributes, and $v_{max} = max\{v_l\}$.

For constructing a CCTree, it is needed to create an $n_i \times m$ matrix based on the data belonging to each non leaf node i, which takes $O(m \times n_i)$ time. The time for finding the appropriate attribute for dividing data based on it needs constant time. To divide the n_i points, based on the v_l features of selected attribute (A_l), $O(n_i \times v_l)$ time is required. This process is repeated in each non leaf node. Thus, if K is the maximum number of non leaf nodes, which arises in a complete tree, then the maximum time required for constructing a CCTree with n elements is equal to $O(K \times (n \times m + n \times v_{max}))$.

5 CONCLUSION AND FUTURE DIRECTIONS

Spam emails impose a cost which is non negligible, damaging users and companies for several millions of dollars each year. To fight spammers effectively, catch them or analyze their behavior, it is not sufficient to stop spam messages from being delivered to the final recipient.

Characterizing a spam campaign sent by a specific spammer, instead, is necessary to analyze the spammer behavior. Such an analysis can be used to tailor a more specific prevention strategy which could be more effective in tackling the issue of spam emails. Considering a large set of spam emails as a whole, makes the definition of spam campaigns an extremely challenging task. Thus, we argue that a clustering algorithm is required to group this huge amount of data, based on message similarities.

In this paper we have proposed a new categorical clustering algorithm named CCTree, that we argue to be useful in the problem of clustering spam emails. This algorithm, in fact, allows an easy analysis of data based on an informative structure. The CCTree algorithm introduces an easy-to-understand representation, where it is possible to infer at a first glance the criteria used to group spam emails in clusters. This information can be used, for example, by officers to track and persecute a specific subset of spam emails, which may be related to an important crime.

In this paper, we have mainly presented the theoretical results of our approach, leaving the implementation of the CCTree algorithm and its usage in clustering spam emails as a future work. Furthermore, we plan to extend the presented approach including labeling of the various clusters. In fact, we plan to use supervised learning approach to assign a label to the various clusters, on the base of spammer goals. Verifying both the efficiency and effectiveness of the proposed approach on a large dataset has also been planned as a future work. Though preliminary, in this work we have shown that the algorithm is efficient, due to the low complexity. Further extensions of this work plan to add a large set of features to best describe the structure of spam emails, in addition to the ones already presented in this work.

REFERENCES

Anderson, D., Fleizach, C., Savage, S., and Voelker, G. (2007). Spamscatter: Characterizing internet scam hosting infrastructure. In *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium.*

Bergholz, A., PaaB, G., Reichartz, F., Strobel, S., and Birlinghoven, S. (2008). Improved phishing detection using model-based features. In *In Fifth Conference on Email and Anti-Spam, CEAS.*

Blanzieri, E. and Bryl, A. (2008). A survey of learning-based techniques of email spam filtering. *Artif. Intell. Rev.*, 29(1):63–92.

Calais, P., Pires, D., Guedes, D., Meira, W., Hoepers, C., and Steding-Jessen, K. (2008). A campaign-based characterization of spamming strategies. In *CEAS.*

Calais Guerra, P., Pires, D., C. Ribeiro, M., Guedes, D., Meira, W., Hoepers, C., H.P.C Chaves, M., and Steding-Jessen, K. (2009). Spam miner: A platform for detecting and characterizing spam campaigns. *Information Systems Applications.*

Carreras, X., Marquez, L., and Salgado, J. (2001). Boosting trees for anti-spam email filtering. In *Proceedings of RANLP-01, 4th International Conference on Recent Advances in Natural Language Processing, Tzigov Chark, BG*, pages 58–64.

Cimiano, P., Hotho, A., and Staab, S. (2004). Comparing conceptual, divisive and agglomerative clustering for learning taxonomies from text.

Drucker, H., Wu, D., and Vapnik, V. (1999). Support vector machines for spam categorization. *Neural Networks, IEEE Transactions on*, 10(5):1048 –1054.

Fette, I., Sadeh, N., and Tomasic, A. (2007). Learning to detect phishing emails. In *Proceedings of the 16th International Conference on World Wide Web*, pages 649–656. ACM.

Gao, H., Hu, J., Wilson, C., Li, Z., Chen, Y., and Zhao, B. (2010). Detecting and characterizing social spam campaigns. In *Proceedings of the 10th annual conference on Internet measurement*, pages 35–47. ACM.

Ghahramani, Z. (2004). Unsupervised learning. In *Advanced Lectures on Machine Learning*, volume 3176 of *Lecture Notes in Computer Science*, pages 72–112. Springer Berlin Heidelberg.

Han, J., Kamber, M., and Pei, J. (2011). *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition.

Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323.

John, J., Moshchuk, A., Gribble, S., and Krishnamurthy, A. (2009). Studying spamming using botlab. In *Proceedings of the 6th USENIX symposium on Networked systems design and implementation*, NSDI09, pages 291–306, Berkeley, CA, USA. USENIX Association.

Kreibich, C., Kanich, C., Levchenko, K., Enright, B., Voelker, G., Paxson, V., and Savage, S. (2009). Spamcraft: an inside look at spam campaign orchestration. In *Proceedings of the 2nd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more*, LEET09, pages 4–4, Berkeley, CA, USA. USENIX Association.

Labs, M. A. (2009). Mcafee threats report: Second quarter 2009.

Leontiadis, N. (2011). Measuring and analyzing search-redirection attacks in the illicit online prescription drug trade. In *Proceedings of USENIX Security 2011*.

Li, F. and Hsieh, M. (2006). An empirical study of clustering behavior of spammers and groupbased anti-spam strategies. In *CEAS 2006 Third Conference on Email and AntiSpam*, pages 27–28.

Quinlan, J. R. (1986). Induction of decision trees. *Mach. Learn*, pages 81–106.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Ramachandran, A. and Feamster, N. (2006). Understanding the network-level behavior of spammers. *ACM SIGCOMM Computer Communication Review*, 36(4):291–302.

Rao, J. and Reiley, D. (2012). On the spam campaign trail. In *The Economics of Spam*, pages 87–110. Journal of Economic Perspectives, Volume 26, Number 3.

Report, T. (April, 2012). http://www.commtouch.com/threat-report-april-2012/.

Seewald, A. (2007). An evaluation of naive bayes variants in content-based learning for spam filtering. *Intell. Data Anal.*, 11(5):497–524.

Shannon, C. E. (2001). A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(1):3–55.

Song, J., Inque, D., Eto, M., Kim, H., and Nakao, K. (2010a). An empirical study of spam: Analyzing spam sending systems and malicious web servers. In *Proceedings of the 2010 10th IEEE/IPSJ International Symposium on Applications and the Internet*, SAINT '10, pages 257–260, Washington, DC, USA. IEEE Computer Society.

Song, J., Inque, D., Eto, M., Kim, H., and Nakao, K. (2010b). A heuristic-based feature selection method for clustering spam emails. In *Proceedings of the 17th international conference on Neural information processing: theory and algorithms - Volume Part I*, ICONIP'10, pages 290–297, Berlin, Heidelberg. Springer-Verlag.

Song, J., Inque, D., Eto, M., Kim, H., and Nakao, K. (2011). O-means: An optimized clustering method for analyzing spam based attacks. In *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, volume 94, pages 245–254.

Tretyakov, K. (2004a). Machine learning techniques in spam filtering. Technical report, Institute of Computer Science, University of Tartu.

Tretyakov, K. (2004b). Machine learning techniques in spam filtering. In *Data Mining Problem-oriented Seminar, MTAT*, volume 3, pages 60–79. Citeseer.

Wei, C., Sprague, A., Warner, G., and Skjellum, A. (2008). Mining spam email to identify common origins for forensic application. In *Proceedings of the 2008 ACM symposium on Applied computing*, SAC '08, pages 1433–1437, New York, NY, USA. ACM.

Xie, Y., Yu, F., Achan, K., Panigrahy, R., Hulten, G., and Osipkov, I. (2008). Spamming botnets: Signatures and characteristics. *SIGCOMM Comput. Commun. Rev.*, 38(4):171–182.

Zhang, C., Chen, W., Chen, X., and Warner, G. (2009). Revealing common sources of image spam by unsupervised clustering with visual features. In *Proceedings of the 2009 ACM symposium on Applied Computing*, SAC '09, pages 891–892, New York, NY, USA. ACM.

Zhao, Y., Xie, Y., Yu, F., Ke, Q., Yu, Y., Chen, Y., and Gillum, E. (2009). Botgraph: Large scale spamming botnet detection. *Proc. of 6th NSDI*.