International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:12, 2007

Layout Based Spam Filtering

Claudiu N.Musat

Abstract**—**Due to the constant increase in the volume of information available to applications in fields varying from medical diagnosis to web search engines, accurate support of similarity becomes an important task. This is also the case of spam filtering techniques where the similarities between the known and incoming messages are the fundaments of making the spam/not spam decision. We present a novel approach to filtering based solely on layout, whose goal is not only to correctly identify spam, but also warn about major emerging threats.

We propose a mathematical formulation of the email message layout and based on it we elaborate an algorithm to separate different types of emails and find the new, numerically relevant spam types.

Keywords**—** Clustering, layout, k-means, spam.

I. Introduction

AMONG many other applications, spam filtering is a domain that requires the computation of a rather large number of inputs. Various methods are in use, ranging from Bayesian word filters to neural networks, taking even more numerous inputs: words, message headers, known spam web domains etc. Unfortunately, so far the layout of the document (in this case an email message, although results can be easily extrapolated because of the generality of the email) has been the aim of little research, if any. This lack of interest could be blamed on the difficulty to separate all junk email from the legitimate ones using just a layout filter, some being so similar that they are practically inseparable. However, formatting, either plain text or html, is a valuable source of information, and different types of emails, such as most newsletters, phishing, or 419 scams, can be singled out considering this criterion only.

II. Problem Formulation

A commonly used phrase – "what you see is what you get" can be more than a slogan or a metaphor to express truthfulness – it can also be the way an experienced eye relates to an email. Sometimes a simple glance is enough to make the decision – should it be read or is it just plain spam? There are numerous changes that spammers can use to trick the classic filters – from poisoning with random words to inserting stories about Robin Hood, from changing the links and addresses to forging headers, but usually they do it while respecting a structure they have noticed to be effective. An important part of this structure is its layout.

The massive text and standard signatures of the 419 scams, the multitude of links and addresses in newsletters, the logos in phishing messages are only a few examples of obvious visual identifying items widely known. But, though widely encountered and full with valuable information, these patterns are, to the best of our knowledge, scarcely used in identifying the email type.

But methods using only these structures are not enough to guarantee a 100% accurate prediction, so for better results, a combination with standard email elements such as keywords or header heuristics is in order, giving birth to a more complex spam signature system.

Such a system allows a better performance – compared to just considering now classical criteria – on exactly those types of emails most dangerous (phishing messages), or or most misclassified (newsletters), leading to an improvement in proactive detection methods, usually considered weak.

Also, identifying spam breakouts and updating filters is easier done regarding the format of the spam waves, in order to skip the minor variations inserted on purpose by the sender.

III. Proposed Method

The proposed classification method could be intuitively separated into three distinct steps. The first is a training phase, based on either a pre-classified email corpus or an unsorted incoming flow of messages. This requires determining the layout structures of the train items, then grouping them into clusters using the k-means algorithm[1]. A selection phase follows, whose aim is to determine the layout types that best separate the training messages. Last in line is the analysis of the new emails, based on determining the previously extracted centroids that are most similar to them. Similarity in this case is defined in terms of a more or less complex distance function[2]. The smaller the distance value, the more similar are the two objects and the query type suitable to determine it – the nearest neighbor[3].

IV. Layout Structure

We have focused on two ways to characterize an email's layout. . The first uses directly quantifiable properties such as its size, the total number of new lines, blank lines, links, addresses or parts. The set of values can be regarded as a feature vector that represents the email's position in the n-dimensional feature space H_n. Determining how similar two

Manuscript submitted January 9, 2006. This work was supported entirely by SOFTWIN S.R.L. 5th, Fabrica de Glucoza Str. Bucharest, Romania.

C. N. Musat is with the Bitdefender AntiSpam Laboratories, 5th, Fabrica de Glucoza Str. Bucharest, Romania, phone +40-21-233-0780, fax +40-21-233-0763 e-mail: cmusat@bitdefender.com.

such vectors are is simply finding the Euclidian or Manhattan distance between the two points in the feature space H_n.

These simple features give us a vague idea as to how the email might look, but they're certainly not enough, so a more complex structure must be used, one that tells us where the blank lines , links or addresses are positioned, how far apart are the new lines – the size of the paragraphs and what the sequence of text, html and even blank lines is and what their respective sizes are. Since we are not interested in what these items represent, just whether they are present and if so, where exactly they are, each type can be given an index – and without loss of generality that index can be an integer. So the text structure of the body of a message like: "Hi, read this: \n http://www.a.com \n a@b.com \n\n\n" could be represented by the string 1134000 – considering 1 – short line, 3 – link, 4-email address, 0 – blank line. The MIME [17] part structure is treated in the same manner, only with different items and indexes. The two are then combined, giving birth to a new space with *m* dimensions H_m.– in our case *m* being 2.

The only remaining problem after generating these index strings is to find a way to establish how similar two points in H_m are, and combine the result with the distance between the representative points of the emails structures under consideration in H_n to find the overall distance between the messages – the distance between the points in the space containing H_m and H_n, referred to as H_{m+n}.

V. Similarity Distance Functions

For the simple case of objects represented by low or medium dimensional feature vectors, the similarity between two objects is typically defined by an appropriate distance function of their representative points in the feature space H_n (e.g. Euclidean distance or Manhattan distance).

These distances however are no longer feasible when the vectors have a variable length, the case of our text/part structure strings. For the latter type of vectors, the distances above no longer help, and using the edit distance becomes appropriate.

The notion of edit distance originated in a paper by Wagner and Fischer [4] on comparing two character strings. In their vision, there are three types of edit operations: deleting a character, inserting a character, and changing one character into another. For a given assignment of costs to all such edit operations, one can use dynamic programming to compute the minimum-cost sequence of operations required to convert one given string to another.

Using the notions above, the total distance between the layout structures can now be computed as the sum of the Manhattan distance between the points in H_n and the edit distances between the corresponding text/part structure strings – the distance in H_m.

VI. K Means

The *k-means* algorithm [1] is by far the most popular clustering tool used in scientific and industrial applications. The name comes from representing each of *k* clusters C_j in the H_{m+n} space by the mean (or weighted average) c_j of its points, the so-called *centroid*

These centroids should be placed carefully because different starting locations generate different results. Therefore, a good choice is to place them as far away from each other as possible. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no more points are pending, the first step is completed and an early grouping is done. At this point we need to re-compute k new centroids as mass centers of the clusters resulting from the previous step. After we are in possession of the k new centroids, a new binding has to be done between the same training points and the new set of centroids. Thus, a loop has been generated, as a result of which we may notice that the k centroids change their location step by step. The algorithm stops when no more changes are made – two consecutive centroid sets are identical.

VII. Layout Structure Clustering

According to the above, the initial steps are to choose *k* centroids from the training data represented by the previously extracted layout structures, and then label each training sample according to the nearest centroid. The difficulty comes in re-computing the coordinates of the centroid. Though easily done for its coordinates in H_n by simply assigning them the cluster averages, the H_m components (the structure strings) cannot be reconstructed from those of the surrounding elements. Thus we propose an approach similar to the k-Medoids: from the examples within the cluster, for each type of structure string choose the one "closest" to the other train elements of its type and record it as the the new centroid's component. The process continues while the resulting centroids differ from the former, or we run out of time – this being a real time operation, depending on recent spam and resulting in an immediate update.

Choosing the correct value for *k* has a critical impact on the functionality, but since there is an unpredictable human factor involved – the spammer, one cannot find the exact value for *k* that best classifies the most recent spam and legitimate messages. So the best that one can do, therefore, is to empirically estimate a value *K* that depends on the total number of emails N, choose an interval centered in that value and run the algorithm for all the integer values inside.

$$k \in [K - \varepsilon, K + \varepsilon]$$

Though it may seem this wastes a lot of time, because of the fact that the different run instances are independent, running them simultaneously, on different machines becomes an easy and time effective solution. The outcome of this phase is thus a collection of j centroid sets *G*, where:

$$j = \sum_{k \in [K-\varepsilon, K+\varepsilon]} M_k \quad (1)$$

and M_k is the number of iterations needed to reach a valid solution for the current instance of the training phase, referred as the number of generations per run. G becomes the set of generations obtained in the training phase as a whole.

The choice to consider all the intermediary solutions as candidates for the final solution is based on the fact that we do not search for the most compact cluster sets possible, but for the ones that best separate one class of emails from the others.

VIII. Solution Filtering

The remaining issue is to find the solution that best fits each of our problems. As previously said, the first issue, to separate the legitimate messages from spam, requires finding those layout clusters that contain just messages of the same type, or in which the main type represents a crushing majority. Thus an indicator P linking the number of the majority of messages $N_{majority}$ and the total number of cluster elements N_{total} is desirable.

$$P = \frac{N_{majority}}{N_{total}} \quad (2)$$

By using this indicator, we can eliminate certain clusters from the start, if P does not exceed an imposed threshold, for instance the majority must exceed two thirds of the total number of messages.

The resemblance of the cluster's elements with respect to the average distance $\overline{D}$ between the forming elements and the centroid – its "density" ρ,

$$\rho = \frac{1}{\overline{D}} \quad (3)$$

and the impact that it will have on future predictions – the number of elements in the cluster N_{total} are also of great importance. Summing up, and using the notations above we come to the importance function, the one used in ordering the centroid list.

$$\eta = \rho \cdot P \cdot N_{total} \quad (4)$$

But since the clusters come from different training processes, it is likely that some of them will overlap, completely or partially. Thus, in order to reduce the number of relevant centroids and thus the number of comparisons needed to classify a new item, these redundancies must be eliminated. If a less relevant cluster is totally contained by a more relevant one, it can be eliminated. Also, the fact that the list is ordered has a high importance, because, though clusters may share regions in H_{m+n} we know how to classify a new item in those regions – to the centroid of the more relevant cluster.

The end of the process leaves us with an ordered and redundancy free list of centroids and their coverage in the H_{m+n} space that represent the representative spam and legitimate email layout structures encountered.

Secondly comes the problem of separating the incoming spam flow, some of the clusters being representative for major spam bursts. Since we already know that all the messages are spam, the aim is nothing more than finding the most compact and numerically representative clusters. Thus, the chosen selection criterion is the sum of the distances between each email layout structure and its corresponding centroid – the smaller the value, the more compact the clusters.

Afterwards, by applying a relevance coefficient

$$\gamma = N_{total} \cdot \overline{D} \quad (5)$$

to each centroid, and a minimum threshold $\gamma_{\min}$, we obtain the centroids corresponding to the layout structure of a probable spam outbreak.

IX. Classifying a New E-Mail Message

The method used to classify a new e-mail layout structure is derived from the nearest neighbor method [3]. In brief, when asked to make a prediction about an unknown point, the nearest-neighbor classifier finds the closest (according to some distance metric) training-point to the unknown point and predicts the category of that training-point. In our case, the training points are replaced by the previously discussed ordered centroid list, and the "closest" centroid is defined as the centroid with the highest relevance whose average distance to the training points in H_{m+n} within its cluster is smaller than the distance to the new item.

But, since we have no guarantee that the whole H_{m+n} space is covered by the centroid list, on the contrary – an email with 10^{10} lines is quite unlikely to be found within the known space, the classification outcome can also be "unknown" – no centroids were found "close enough".

X. Experimental Results

Because of the volatility of the spam, the train data is composed of the latest emails available only. Consequently, the first type of tests, involving the issue of detecting the most important spam, phishing and normal email layouts were run using the current day flow: spam (N_S = 1514 messages) and phishing (N_P = 84 messages), and a selection of legitimate emails, as different as possible (N_L =927 messages), resulting a total of $N = N_S + N_P + N_L$ = 2525 messages.

Also, the numbers of centroids used were $k \in [K-\varepsilon, K+\varepsilon]$, where $K = 2\sqrt{N}$ = 100 centroids and $\varepsilon = 10$.

Having generated the best centroids' list, the results of classifying the N_L legitimate emails had 97.1% accuracy with 27 messages classified as spam (false positives), the classification of the phishing messages had a 92.9 accuracy, with 6 messages being tagged wrongly as spam and 78 correctly identified as phishing, and the spam messages were correctly tagged in 95.5% of the cases, 46 being misclassified (false negatives), and 23 marked as "unknown". The 85

legitimate messages labeled "unknown" have been considered correctly classified, because they do not fit the definition of a false positive.

The second type of tests were run over 5 consecutive days on the incoming flow, a total of 7314 emails, and 8 major breakouts, with more than 50 similar messages each, were detected.

XI. Conclusion and Future Work

TABLE I
ACCURACY RESULTS

Type of Messages	Correctly Classified	Misclassified	Unknown	Rate [%]
Legitimate	815	27	85	97.1
Phishing	78	6	0	92.9
Spam	1445	69	23	95.5

In this paper, we have proposed a new algorithm for spam detection based solely on the email message structural similarities, this being, to the best of our knowledge, the first kind of work in its area. Despite the various filtering techniques like Naïve Bayes Classification [5], Markov chains [6], Neural Networks[7] or support vector machines (SVMs) [8], not only does the spam phenomenon persist, , but also constitutes a growing threat. The emergence of new methods to bypass the classical word based filters generated the need to consider all the relevant email message features. We have demonstrated that the technique is a viable solution for spam filtering, by producing results at the level of the classical spam classifiers. Though encouraging, our quantitative results cannot present a guarantee that they could be reproduced on any email corpus, but emphasize the fact that most emails can be more accurately classified if layout factors are included. The main advantage of this method is that it is complementary to the existing ones, and their predictions can easily be merged, for example by adding a confidence level to each one, followed by an addition, or by integrating the key layout structures as inputs to a neural network.

The major drawback of the filter, the time required for its training, requires complementing it with a dimensionality reduction algorithm, along with a better choice of the layout feature components. Among these, a combination between layout and word filtering – the positions within the message of certain keywords – could play a major part in enhancing the detection rates.

References

[1] J. B. MacQueen "Some Methods for classification and Analysis of Multivariate Observations, Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability", 1967 Berkeley, University of California Press, 1:281-297

[2] P. L. Hammer "Distance-based classification methods", 1999, INFOR, Canadian OR Society Vol.37, s. 337-352

[3] T. M. Cover. Estimation by the Nearest Neighbor Rule. IEEE Transactions on Information Theory, IT-14(1):50--55, 1968

[4] E.S.Ristad, P.N.Yianilos "Learning String Edit Distance" [Online]. Available: http://www.pnylab.com/pny/papers/sed/sed.pdf

[5] P. Graham. A plan for spam., 2002 [Online]. Available: http://www.paulgraham.com/spam.html.

[6] H. Lee, A. Y. Hg "Spam Deobfuscation using a Hidden Markov Model, 2005 [Online]. Available: http://ai.stanford.edu/~ang/papers/ceas05-spamdeobfuscation.pdf

[7] C. Miller "Neural Network-based Antispam Heuristics", 2005 [Online]. Available: http://www.mn-issa.org/whitepapers/Symantec/AntiSpam%20Heuristics%20White%20Papers.pdf .

[8] J. C. Burges "A Tutorial on Support Vector Machines for Pattern Recognition" 1998 "Data Mining and Knowledge Discovery", 2, 121–167, Kluwer Academic Publishers, Boston, USA.

[9] H. J. Mucha, H. Sofyan: "Nonhierarchical Clustering" ch.9.3. [Online]. Available: http://www.quantlet.com/mdstat/scripts/xag/html/xaghtmlframe149.html

[10] P. Berkhin, "Survey of Clustering Data Mining Techniques", 2002, Accrue Software, Available: www.ee.ucr.edu/~barth/EE242/clustering_survey.pdf

[11] R. Ng, J. Han. "Efficient and effective clustering method for spatial data mining", 1994. Proceedings of the 20th VLDB conference Santiago, Chile, 144-155.

[12] J. Zhang, M. Zhu, D. Papadias, Y. Tao, D. L. Lee "Location-based Spatial Queries" 2003, ACM SIGMOD San Diego, USA

[13] T.Seidl, H. P. Kriegel "Optimal Multi-Step k-Nearest Neighbor Search", 1996, ACM SIGMOD Seattle, USA

[14] U. Luxburg, O. Bousquet "Distance-Based Classification with Lipschitz Functions", 2004, Journal of Machine Learning Research 5, 669–695

[15] S. Dixit, S. Gupta, C. V. Ravishankar "An Online Detection and Control System for SMS Spam", 2005, Proceedings of the IASTED International Conference Communication, Network and Information Security, Phoenix, AZ, USA.

[16] R. M. Hayes, "Mathematical models in information retrieval", 1963 Natural Language and the Computer, McGraw-Hill, New York, USA.

[17] RFC 2045 [Online] Available: http://rfc.net/rfc2045.html