A Cumulative Timeline Feature Selection Approach for Data Classification

Rafiqul Islam
School of Computing, Charles Sturt University

mislam@csu.edu.au

Abstract

This paper presents a novel feature vector generation technique of malware data which retains high classification accuracy over an extended time period. The proposed approach is to combine the features and accumulating these features with intervals over time. Experimental results show that the proposed method maintains constant classification accuracy and with a standard deviation of 0.92 over the extended time period. These results strongly support the hypothesis that it is possible to develop any classification strategy that will work well into the future.

Key Words: Timeline, Cumulative, Malware, Integrated.

1. INTRODUCTION

Malware authors use various obfuscation techniques to transform a malicious program into undetectable variants with the same core functionalities of the parent malware program. The study in [1] investigates malicious attacks on several websites by creating web honey pots and collecting website-based malware executables over a period of five months. In their study, they collect and analyse malware samples using 6 different antivirus programs, and conduct the same experiment four months later using the updated versions of the 6 programs to determine their efficacy. This work demonstrates that, with training on older malware, some anti-virus software can significantly improve detection rates.

Many researchers have argued that any classification strategy which has been successful in a given time period will not work at a much later date due to changes the data characteristics of nature of the data in particular malicious data. This philosophy is supported by the work in [2 - 11] which indicates that current techniques failed to find a distinctive pattern of malicious data which can be used to identify future malicious data. The argument is that malware evolves with time and eventually becomes unrecognizable from the original form; in addition completely new malware is designed which is unlike any known malware and so would not be detected by anti-virus software constructed to detect known types of malware. In fact, the assumption that malware completely unlike earlier malware is being designed on a major scale is known to be false as indicated by the statistics in [9] showing that barely 25 % of data in 2006 is not a variant of known malware data.

Despite the strong support in the literature for the idea that current detection methods will not easily detect future malware, in this paper, we demonstrate that it is possible to develop a malware detection strategy which retains high accuracy over an extended time period. Therefore, this research provides a significant outcome to classify the future malware data.

2. EXPERIMENTAL SET-UP

2.1 Timeline data preparation

The date of a malware data was that associated with the file when the file was collected. We exported all files, along with their dates, into our database and based on the dates broke the data into groups

To generate groups of malware for use in the testing, we begin with the earliest malware and add month by month across the timeline until all data are grouped. As the first data group, MG_1, we take the earliest-dated 10% of the files. The second data group, MG_2, comprises the data collected immediate after for MG_1, and so on. In all, this results in N malware data groups which are labeled $MG_1 ... MG_N$. Figure 1 indicates the spread of malware across the N groups with each bar corresponding to a group.

Throughout the test, the set of cleanware files is treated as a single group CG. However, when it is tested against a particular malware group, depending on the comparative size of the two groups, the cleanware group may be divided into subgroups.

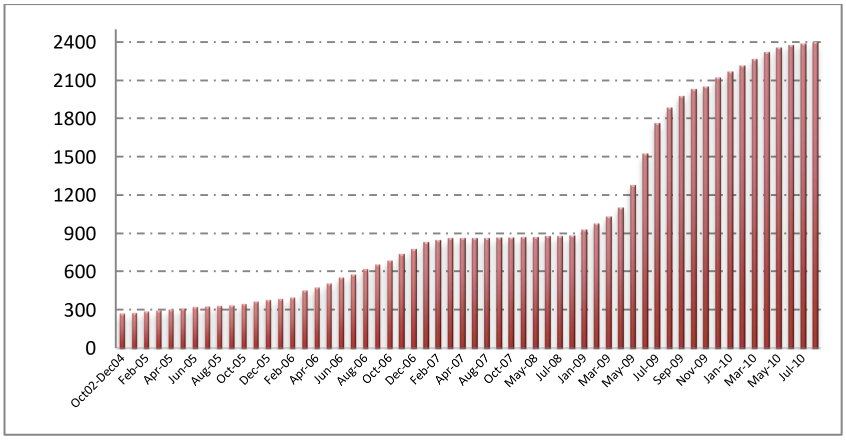

Figure 1. Number of malware executables accumulated by date.

2.2 Test data preparation:

In test data preparation we follow the guidelines in [10-13], use an equal portion of malware and of cleanware data. Figure 2 shows the data preparation process. The selected malware group MG_i is compared with CG. If $|MG_i|$ is smaller than $|CG|$ then we compute the integer part of $|CG|/|MG_i|$ and the integer reminder $0 \leq R < |MG_i|$ as in

$$|CG| = k|MG_i| + R, \text{ for some positive integer k.}$$

We then divide CG into k disjoint groups of equal size. If R > 0, then the remaining elements must be padded out to a (k+1)'st group CG_{k+1}. However, if R = 0, this set is empty and is not used.

If $|MG_i|$ is bigger than $|CG|$ then we compute the integer part of $|MG_i|/|CG|$ and proceed in the same way. This procedure is repeated for every malware group.

2.3 The WEKA interface

In our classification process, we input the feature vectors into the WEKA classification system [6] for which we have written an interface. In all experiments, 10-fold cross validation is applied to ensure a thorough mixing of the features. In this procedure, we first select one group of malware data from a particular data set and divide it into ten portions of equal size; then we select cleanware data of the same size as the group of malware data and also divide it into ten portions. The portions are then tested against each other.

To establish the training set, our detection engine takes nine portions from each of the malware and cleanware to set up the training set and the remaining portions from both malware and cleanware are used for the testing set. As is customary, the training set is used to establish the model and the testing set is used to validate it. The whole process is repeated so that every portion of both malware and cleanware is chosen as testing data; the results are then averaged. In order to ensure that the input vectors are trained and tested over a broad spectrum of classifiers, we chose the following four classifiers from WEKA as they represent differing approaches to statistical analysis of data: Support Vector Machines (SVM), Random Forest (RF), Decision Table (DT) and IB1.

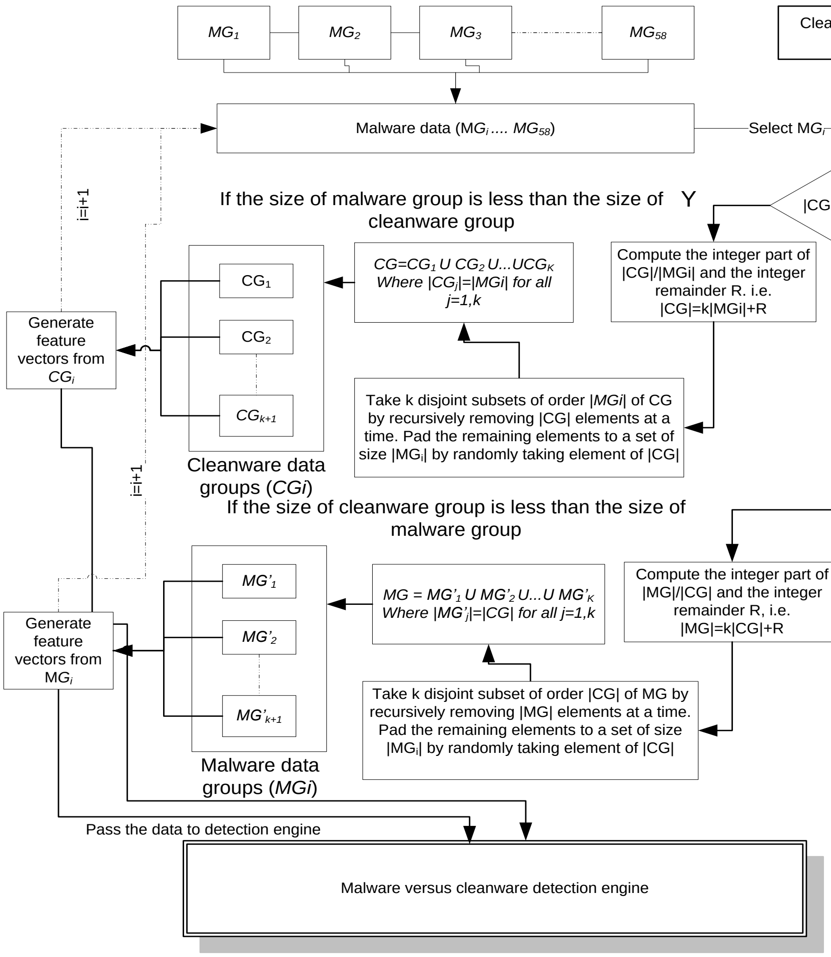
MG_1
MG_2
MG_3
MG_{58}
Cleanware Data (CG)
Select CG
Malware data (MG_i.... MG_{58})
Select MG_i
i=i+1
If the size of malware group is less than the size of cleanware group
Y
|CG|>|MGi|
N
CG_1
CG_2
CG_{k+1}
$CG=CG_1 \cup CG_2 \cup ... \cup CG_K$ Where $|CG_j|=|MGi|$ for all j=1,k
Compute the integer part of |CG|/|MGi| and the integer remainder R. i.e. |CG|=k|MGi|+R
Generate feature vectors from CG_i
Take k disjoint subsets of order |MGi| of CG by recursively removing |CG| elements at a time. Pad the remaining elements to a set of size $|MG_i|$ by randomly taking element of |CG|
Cleanware data groups (CGi)
i=i+1
If the size of cleanware group is less than the size of malware group
MG'_1
MG'_2
MG'_{k+1}
$MG = MG'_1 \cup MG'_2 \cup ... \cup MG'_K$ Where $|MG'_j|=|CG|$ for all j=1,k
Compute the integer part of |MG|/|CG| and the integer remainder R, i.e. |MG|=k|CG|+R
Generate feature vectors from MG_i
Take k disjoint subset of order |CG| of MG by recursively removing |MG| elements at a time. Pad the remaining elements to a set of size $|MG_i|$ by randomly taking element of |CG|
Malware data groups (MGi)
Pass the data to detection engine
Malware versus cleanware detection engine

Figure 2. Feature generation

3. EXPERIMENTS AND RESULTS

We ran the entire experiment using each of the four base classifiers SVM, IB1, DT and RF. In addition, each test was run five times and the results averaged in order to ensure that any anomalies in the experimental set-up were discounted.

Figure 3 shows the average results over the timeline data (this is the average of 5 separate tests for each data-group). This average manages to stay above the 80% mark over most of the time period, but drifts under in the immediate previous year of data for some classifiers. As expected, the classifier RF is best on average.

In order to claim that we have a reasonable malware detector over the eight year period, we now focus on the change in accuracy over this time. We therefore compute the standard deviation of the accuracy data from the N values for each of the feature sets and for each of the classifiers, to determine the variation from the mean in each case. Table 1 presents the standard deviation data for the twelve cases. We can see that the Dynamic test shows consistent, and good, performance for all classifiers except IB1; excluding IB1, the difference in spread of the remaining three results is 0.75 points. This is by far the best of our results; however, note that the IB1 result for the dynamic test is worse than all results for the PSI test and worse than two results for FLF. FLF, as expected, has the worst standard deviations, but not all values are worse than those for the PSI test: RF performs slightly better for FLF than SVM does for PSI. The difference in spread of standard deviations for FLF is 1.86 points. There is insufficient support here for removing FLF as a test of the presence of malware. Interestingly, the PSI test has consistent results but with fairly large standard deviations; the difference in spread is 1.49 points.

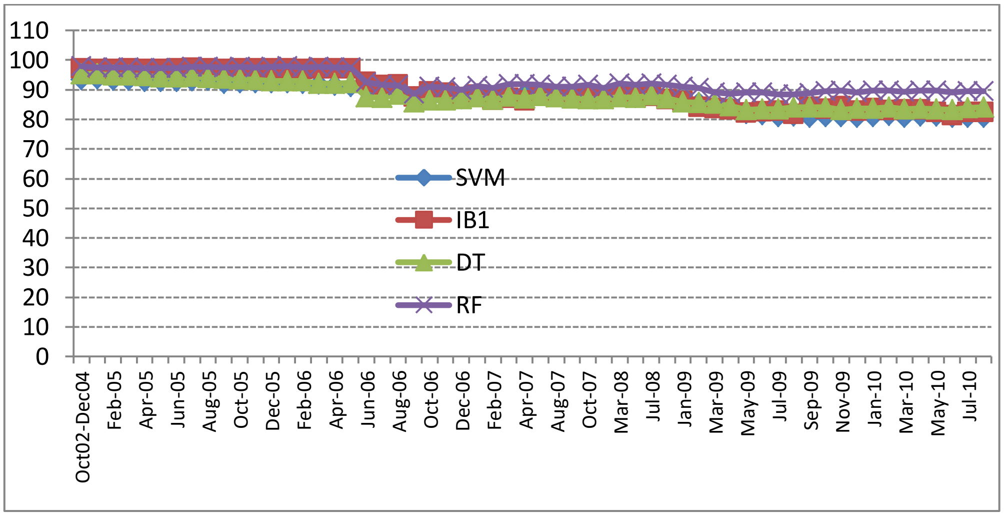

Figure 3. Average timeline results of three tests.

Table 1. Comparison of standard deviations

Feature set	SVM	IB1	DT	RF
FLF	7.422030	5.854992	7.228582	5.755763
PSI	5.849069	5.341312	5.684248	4.354974
Dynamic	0.921473	6.902749	1.668326	1.289221

Turning to an analysis of which classifier is the best, IB1 clearly gives the most consistent results (the difference in spread being 1.56 points), but all three standard deviations are much larger than we would want in a malware detector, and so one of our conclusions would be to exclude this particular classifier from future analysis work on malware. In this case, RF is the best remaining classifier for the FLF experiment, and while RF is not as good as SVM for the dynamic test, it gives better results than the DT classifier for this test, and so we would highly recommend retaining RF in future malware detection analysis work.

4. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a cumulative timeline feature vector generation approach and demonstrated that it retains high accuracy over an extended period of time. The results presented in

Section III indicate that our method retains consistent accuracy over the eight year period. Our approach to feature collection is novel in that we accumulate the features over time segments of an eight year span. In progressively adding additional malware over the time period, we thereby strengthen the accuracy of the test. The implication for anti-virus engines is that they are then able to use previously detected malware to provide features based on which to test new executables.

The results presented in Section III, indicate that no one feature type is the most significant over the eight year span. In our experiment, the integrated features are shown to act independently, and each contributes value to the analysis. However, one conclusion of the discussion in Section III would be to exclude the IB1 classifier from future analysis work on malware, but to retain RF. Therefore, it is expected that combining static and dynamic features in an integrated manner could give a better detection rate; we will explore this in our future work.

REFERENCES

[1.] I. You and K. Yim. Malware obfuscation techniques: A brief survey. In *Broadband, Wireless Computing, Communication and Applications (BWCCA), 2010 International Conference*, pp. 297–300. IEEE, 2010.

[2.] M. Fossi, E. Johnson, T. Mack, D. Turner, J. Blackbird, M. K. Low, T. Adams, D. Mckinney, S. Entwisle, M. P. Laucht, C. Wueest, P. Wood, D. Bleaken, G. Ahmad, D. Kemp, and Samnani. Symantec global internet security threat report trends for 2009, technical report, 2009.

[3.] H. Tang, B. Zhu, and K. Ren. A new approach to malware detection. *Advances in Information Security and Assurance*, pp. 229–238, 2009.

[4.] P. Barford and V. Yegneswaran. An inside look at botnets. *Malware Detection*, pp. 171–191, 2007.

[5.] M. Bailey, J. Oberheide, J. Andersen, Z. Mao, F. Jahanian, and J. Nazario. *Automated Classification and Analysis of Internet Malware*, chapter Recent Advances in Intrusion Detection, pp. 178–197. 2007.

[6.] Mark, H. , Eibe F., Geoffrey H., Bernhard, P., Peter R., Ian H. W. (2009); The WEKA Data Mining Software: An Update; *SIGKDD Explorations*, Volume 11, Issue 1.9

[7.] R. Tian, L.M. Batten, and S.C. Versteeg. Function length as a tool for malware classification. In *Proceedings of the 3rd International Conference on Malicious and Unwanted Software: MALWARE 2008*, pp. 69–76, 2008.

[8.] J. Lee, C. Im, and H. Jeong. A study of malware detection and classification by comparing extracted strings. In *Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication*, p. 75. ACM, 2011.

[9.] M. Braverman, J. Williams, and Z. Mador. Microsoft security intelligence report January-June 2006. Accessed on 19th May 2011 CYVEILLANCE.

[10.] V.P. Nair, H. Jain, Y.K. Golecha, M.S. Gaur, and V. Laxmi. Medusa: Metamorphic malware dynamic analysis using signature from api. In *Proceedings of the 3rd international conference on Security of information and networks*, pp. 263–269. ACM, 2010.

[11.] Islam, R., Tian, R., Batten, L.M., Versteeg, S., (2013). Classification of malware based on integrated static and dynamic features. J. Network and Computer Applications 36, 646–656

[12.] R. Tian, L. Batten, R.l Islam, and S. Versteeg. An automated classification system based on the strings of trojan and virus families. In *Proceedings of the 4rd International Conference on Malicious and Unwanted Software: MALWARE 2009*, pp. 23–30, 2009.

[13.] R. Tian, R. Islam, L. Batten, and S. Versteeg. Differentiating malware from cleanware using behavioural analysis. In *Malicious and Unwanted Software (MALWARE), 2010 5th International Conference*, pp. 23 –30, 2010.