# Protecting Data from Malware Threats using Machine Learning Technique

Mozammel Chowdhury
School of Computing & Mathematics
Charles Sturt University, Australia
Email: mochowdhury@csu.edu.au

Azizur Rahman
School of Computing & Mathematics
Charles Sturt University, Australia
Email: azrahman@csu.edu.au

Rafiqul Islam
School of Computing & Mathematics
Charles Sturt University, Australia
Email: mislam@csu.edu.au

***Abstract*—Cyber attacks against sensitive data have become as serious threats all over the world due to the rising applications of computer and information technology. New malware or malicious programs are released everyday by cyber criminals through the Internet in an attempt to steal or destroy important data. Hence, research on protecting data receives tremendous interest in the cyber community. In order to cope with new variants of malicious software, machine learning techniques can be used for accurate classification and detection. This paper proposes an efficient scheme for malware detection for protecting sensitive data from malicious threats using data mining and machine learning techniques. Experimental results shows that the proposed approach gives better performance compared to other similar methods.**

***Keywords—Cyber security, Malware detection, Data protection, Machine learning.***

## I. Introduction

Malicious software or malware breaches the secrecy and integrity of data and causes unauthorized leakage of information. In recent years, cyber attacks are increasing alarmingly because of the emerging applications of computer and Internet. Hundreds of thousands of new malicious programs are released by cyber criminals through the Internet in an attempt to steal or destroy important data. Hence, efficient detection and prevention of malicious insiders for protecting valuable data is of critical importance in the computer user community.

Typical approaches for malware detection can be classified into two categories: (i) signature-based approach and (ii) anomaly-based approach [1]. Signature-based methods are very efficient and fast to detect known malware [2]. Most of today's commercial anti-malware techniques use a set of signatures to detect malicious programs. A signature could represent a series of bytes in the file or a cryptographic hash of the file or its sections. In a signature-based malware detection system, the signature or fingerprint of a file is analysed and then compared with a database of signatures of known malicious files. If the signature is not available in the dataset, it means that the file is begin (clean ware) other than malicious (maleware). However, most of the malwares can generate new variants each time it is executed and a new signature is generated. Therefore, signature based approaches fail to detect unknown malwares which are not in the database. In addition, another limitation of signature-based detection methods is that they require human knowledge to update the signature database by new signatures [3].

On the other hand, anomaly-based detection approaches use API call sequences instead of byte sequence matching [3, 4]. Although anomaly-based detection approaches use the knowledge of normal behavior patterns and performs better than the signature based approach, they have however, high false alarm rate. Furthermore, the behavior based methods are slow as the real-time detectors on the final host and they often need virtual machine technologies.

In recent years machine learning has been proposed to circumvent the challenges of the conventional malware detection methods. Machine learning has been proven to be capable of detecting new malware variants [5]. Machine learning techniques used for detection and classification of malicious entity include support vector machines, decision tree, random forest, and Naive Bayes [6].

However, machine learning approaches have shortcomings of increasing false alarm rate due to weak feature selection and inefficient classifier generation. Therefore, accurate detection of malware is still a key challenge for the cyber community. This paper proposes a comprehensive framework to detect and classify malicious software in order to protect data from their attacks using a machine learning based classification algorithm.

The rest of the paper is organized as follows. Section II demonstrates a review of related works. The proposed system for malware detection is described in Section III. Experimental results are reported and discussed in Section IV. Finally, Section V concludes the paper.

## II. Related Works

Many researchers have proposed machine learning approaches for malware detection. Machine learning algorithms used in malware classification include association classifiers, support vector machines, decision tree, random forest and Naive Bayes. In this section, we present few references to exemplify such methods.

Wang et al. [7] proposed a method based on Support Vector Machine associated with static and dynamic analysis of file samples. Static analysis of the samples were performed by dumping the program's import table from the Portable Executable (PE) structure, and recording all DLLs used and API function calls. In dynamic analysis, each sample was executed in a VMWare sandbox environment and monitored changes in registry, system folders/files, and network states. The results of each type of analysis were used as features in the Support Vector Machine, and information gain was calculated to observe the influence of feature selection.

An approach was proposed by Chavan and Zende [8] for spyware detection using data mining and supervised machine learning techniques. They extracted n-gram features from file samples and used supervised learning algorithms to classify files as spyware or benign. N-grams of various sizes (centered on n = 5) and several supervised learning algorithms were compared to create a better performing solution than existing anti-virus software.

Kolter and Maloof [9] proposed a scheme using boosted decision trees based on n-grams which provided better results than both the Naive Bayes classifier and Support Vector Machines. The authors [10] used association rules on Windows API execution sequences to distinguish between malware and clean program files.

Chouchane et al. [11] used Hidden Markov Models in their scheme to detect whether a given program file is (or is not) a variant of a previous program file. To reach a similar goal, Stamp et al. [12] applied Profile Hidden Markov Models, which have been previously used with great success for sequence analysis in bioinformatics.

The capacity of neural networks to detect polymorphic malware is explored in [13]. Self-Organizing Maps were used by Yoo [14] to identify patterns of behaviour for viruses in Windows executable files. Some malware categorization schemes employ clustering techniques [15]. However, these classification techniques require a large number of training samples to build the classification models. Moreover, machine learning approaches can have shortcomings of increasing false alarm rate due to weak feature selection and inefficient classifier generation. Hence, accurate detection of malicious software is still a key challenge for the cyber community. This paper proposes a comprehensive framework to detect and classify malicious software in order to protect data from their threat using machine learning classification technique.

## III. PROPOSED APPROCH

The proposed malware detection system is consisted of the following major components: (i) Data collection, (ii) Pre-processing, (iii) Features extraction, (iv) Feature reduction, (v) Classification, and (vi) Detection. The architecture of the proposed approach is depicted in Fig. 1.

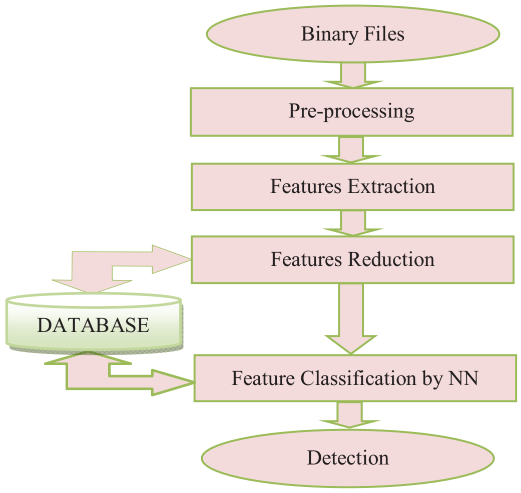


Fig. 1. Architecture of the proposed malware detection system.

### *A. Data Collection*

In this work, we have collected 52,185 executable files in total consisting of 41,265 recent malicious files and the remaining being benign files. Table 1 represents the malware and cleanware datasets used in this experiment. The malicious files are collected from VX Heaven [6] and the benign files are collected from online sources: Download.com and Softpedia.com. The malware collection consists of Trojans, backdoors, hack tools, root kits, worms and other types of malware.

**Table 1.** Malware and Cleanware Dataset

| File Type | | Number of Files |
|---|---|---|
| Malware | Backdoor | 5,340 |
| | Virus | 13,645 |
| | Rootkit | 423 |
| | Trojan | 11,435 |
| | Worm | 9,556 |
| | Exploit | 329 |
| | Other | 537 |
| Cleanware | | 10,920 |
| **Total** | | **52,185** |

### *B. Pre-processing*

The collected files are raw executable files; they are stored in the file system as binary code. To make them suitable for our work, we have preprocessed them. First, we unpack the executables in a restricted environment called virtual machine (VM). In order to unpack the packed executables automatically, we use the tool PEid [17].

### *C. Feature Extraction*

After preprocessing of the collected malware and cleanware files, we extract two types of general features from each of these files: n-gram features and Windows Portable Executables (PE) based on their API calls.

*1) N-gram Features*: An n-gram is a sequence of substrings with a length of n-grams. The benefit of using n-gram technique is that it can capture the frequency of words having a length of n-grams [18]. We extract the n-gram (5-gram) features from each of these malware and cleanware executable files. Empirically we have found that n-grams of size 5 (each sequence is exactly 5 bytes) provide the most overall accurate results. Hence, we choose to evaluate our experiment with n-gram size of 5. The Term Frequency (TF) is used to estimate the frequency of the n-gram features that appeared in a file. A matrix is creted containing malware and cleanware file with TF n-gram vectors.

*2) N-gram Features:* The Portable Executable (PE) header contains information about how the operating system manages a resources allocated to a program. Features from the PE Header are extracted. The PE feature mining is done with a package called "pedump" [9].

### *D. Feature Reduction*

After feature extraction, the important step is the feature reduction. In this step, most informative features are selected and the best one is examined based on the calculation of the classifier accuracy that corresponds with the number of features that are selected using different feature selection methods. In this work, we use Principal Components Analysis (PCA) [19] for feature selection. The PCA is employed for increasing computation speed due to its capacity for dimensionality reduction. It is based on converting a large number of variables into a smaller number of uncorrelated variables by finding a few orthogonal linear combinations of the original variables with the largest variance.

### *E. Classification and Detection*

After processing the binary files, we generate the training and test data sets with the malware and clean ware files. Classification process is divided into two stages: training and testing. In the training phase, a training set of malicious and benign files is provided to the system. The learning algorithm trains a classifier. The classifier learns from the labelled data samples. In the testing phase, a set of new malicious and benign files, which are not used in the training phase are fed into the classifier. 80% of the malware and cleanware files are used for training while the remaining files (20%) are used for testing. In this work, we employ an artificial neural network (ANN) with a feed-forward perceptron to build the classifier. Fig. 2 shows the architecture of the proposed ANN classifier. The training set is passed into the classifier to train it and then justify the effectiveness with the test dataset. The number of epochs for this experiment is 35,000 and the minimum error margin is 0.002.

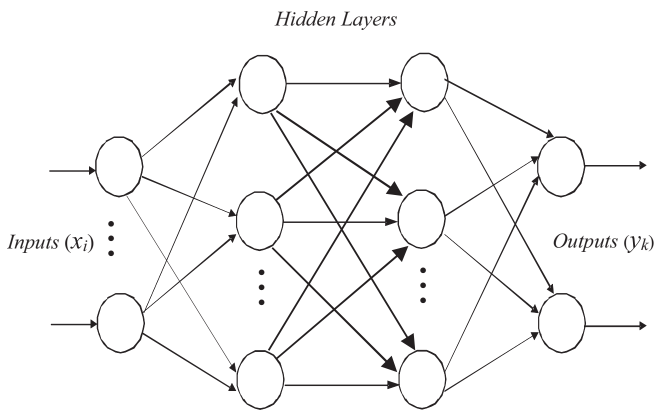


Fig. 2. Neural network architecture for malware classification.

## IV. Experimental Results And Discussions

In this section we present the experimental results of our proposed approach for malware classification and detection. Experiments are carried out with the collected datasets of both malware and cleanware. To evaluate the performance of the proposed method, we estimate the following evaluation metrics:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \qquad (1)$$

$$TPR = \frac{TP}{TP + FN} \qquad (2)$$

$$FPR = \frac{FP}{FP + FN} \qquad (3)$$

Where, *TPR* = True Positive Rate, also known as sensitivity; and *FPR* = False Positive Rate.

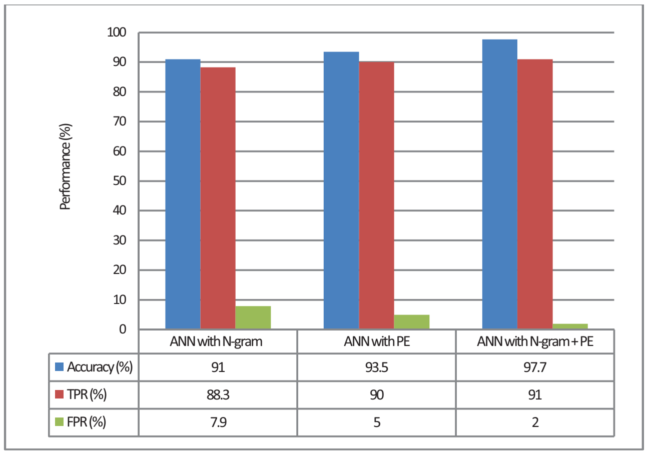
*TP* (true positive) = No of malware files correctly identified as malware,

*FP* (false positive) = No of malware files incorrectly identified as cleanware,

*TN* (true negative) = No of cleanware files correctly identified as cleanware,

*FN* (false negative) = No of cleanware files incorrectly identified as malware,

We perform experiments using combined features and individual ones to test our approach. Experimental result reported in Fig. 3 clearly shows that the proposed classifier gives higher accuracy in case of using integrated features. We compare our proposed method with other similar methods including, Support Vector Machine (SVM), Decision Tree (J48), Naïve Bayes, and Random Forest using similar features. Experimental results are reported in Fig. 4 and Fig. 5. The results clearly indicate that our proposed method provides better performance compared to other similar methods.



| | ANN with N-gram | ANN with PE | ANN with N-gram + PE |
|---|---|---|---|
| 🟦 Accuracy (%) | 91 | 93.5 | 97.7 |
| 🟥 TPR (%) | 88.3 | 90 | 91 |
| 🟩 FPR (%) | 7.9 | 5 | 2 |

Fig. 3. Performance of the proposed method with individual features integrated features.

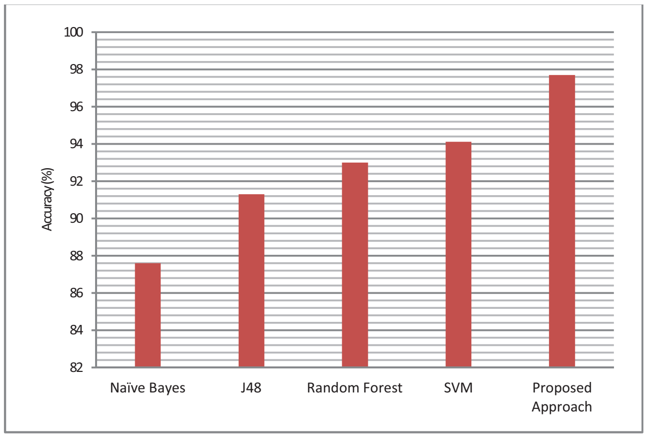


Fig. 4. Comparison of malware detection accuracy for different methods.

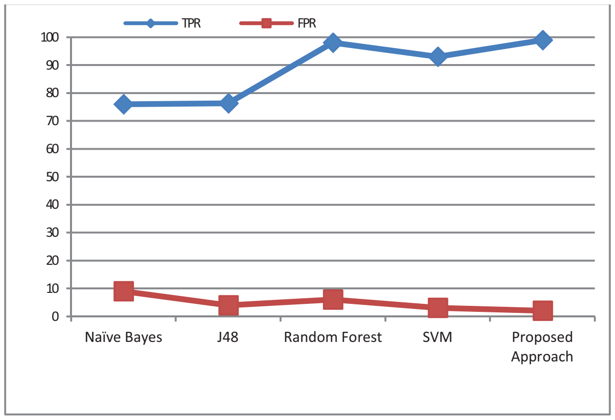


Fig. 5. Comparison of TPR and FPR for different methods.

## V. Conclusion

This paper proposes an efficient and robust malware detection system using machine learning classification algorithm. We have explored the variations of parameters and their effect on the accuracy of detection. Our approach combines the use of n-gram and PE features to achieve more accuracy. Experimental evaluation confirms that our proposed method provides better performance compared to other similar methods. In the future work, we aim to use more variant features in conjunction with each other to achieve more detection accuracy and reduce false positive.

## References

[1] Islam R, Tian R, Batten LM, and Versteeg S (2013). Classification of malware based on integrated static and dynamic features. Journal of Network and Computer Applications 36:646–656.

[2] K. Tang, M.T. Zhou, Z. Z -H (2010). An enhanced automated signature generation algorithm for polymorphic malware detection, J. of Electronic Science and Technology of China, 8:114–121.

[3] I. Gurrutxaga , Evaluation of Malware clustering based on its dynamic behaviour. Seventh Australasian Data Mining conference, Australia, pp. 163–170, 2008.

[4 Tian R, Islam R, Batten L, Versteeg S. Differentiating malware from cleanware using behavioural analysis. Int. conference on malicious and unwanted software: MALWARE 2010; 2010. p. 23–30.

[5] Hadžiosmanović, D., Simionato, L., Bolzoni, D., Zambon, E., and Etalle, S. 2012. N-Gram Against the Machine: On the Feasibility of the N-Gram Network Analysis for Binary Protocols. Research in Attacks, Intrusions, and Defenses. Springer. 354-373.

[6] Chan P. K. and Lippmann R.. Machine learning for computer security. Journal of Machine Learning Research, vol. 6, pp. 2669–2672, 2006.

[7] Wang, T., Horng, S., Su, M., Wu, C., Wang, P., & Su, W. (2006). A surveillance spyware detection system based on data mining methods. IEEE Congress on Evolutionary Computation, Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada. 3236-3241.

[8] Chavan, M. k., & Zende, D. A. (2013). Spyware solution: Detection of spyware by data mining and machine learning technique. International Conference on Advanced Research in Engineering and Technology, Vijayawada, India

[9] J. Z. Kolter and M. A. Maloof, “Learning to detect and classify malicious executables in the wild,” Journal of Machine Learning Research, vol. 7, pp. 2721–2744, December 2006, special Issue on Machine Learning in Computer Security.

[10] Y. Ye, D. Wang, T. Li, and D. Ye, “Imds: intelligent malware detection system,” in KDD, P. Berkhin, R. Caruana, and X. Wu, Eds. ACM, 2007, pp. 1043–1047.

[11] M. R. Chouchane, A. Walenstein, and A. Lakhotia, “Using Markov Chains to filter machine-morphed variants of malicious programs,” in Malicious and Unwanted Software, 2008. MALWARE 2008. 3rd International Conference on, 2008, pp. 77–84.

[12] M. Stamp, S. Attaluri, and S. McGhee, “Profile hidden markov models and metamorphic virus detection,” Journal in Computer Virology, 2008.

[13] R. Santamarta, “Generic detection and classification of polymorphic malware using neural pattern recognition,” 2006.

[14] I. Yoo, “Visualizing Windows executable viruses using self-organizing maps,” in VizSEC/DMSEC ’04: Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security. New York, NY, USA: ACM, 2004, pp. 82–89.

[15] Kolter JZ, Maloof MA. Learning to detect malicious executables in the wild. In: Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining. New York, NY, USA: ACM; 2004. p. 470–8.

[16] VX Heaven collection, VX Heaven website, available at: http://vx.netlux.org

[17] PEid Unpacker. http://www.peid.info/

[18] Jain, S. and Meena, Y. K. 2011. Byte Level n–Gram Analysis for Malware Detection. Computer Networks and Intelligent Computing. Springer. 51-59.

[19] Xu, X. and Wang, X. 2005. An Adaptive Network Intrusion Detection Method Based on PCA and Support Vector Machines. Advanced Data Mining and Applications. Springer. 696-703.