Malware Family Classification Based on Novel Features from Frequency Analysis

Changhee Choi, Kyeongsik Lee, Hwaseong Lee, Ilhoon Jeong, and Hosang Yun

***Abstract*—In the past, the number of malware was small, and signature-based anti-virus program could be used to effectively protect the system. Cyber attackers create a large number of variants of malwares with automated tools to avoid signature-based anti-virus programs. Creating signature for all the variants is quite expensive task. To solve this problem, defensive side has been tried to automatically detect the malware variants. Classifying malware families can be one way to solve them. In this paper, we extract novel features from frequency analysis of malware to classify malware family. We separate the malware into section level and apply DCT/DFT to each section. Experimental results show that the proposed method can achieves high accuracy and low operation cost.**

***Index Terms*—Bootstrap aggregating, discrete cosine transform, frequency analysis, malware family, malware image, machine learning, Microsoft malware classification challenge.**

I. Introduction

Since the first virus, "Creeper" was created in 1971, cyber-attacks with malware have been constantly ongoing [1]. Cyber security specialists make the signature such as MD5, SHA256 and Yara rule to defeat the malwares. However, it is easy to manipulate the malware to avoid the signature set made from anti-virus program. For example, it can change the signature without affecting the program even if only one byte is modified. Malware variants can be generated automated tools such as UPX packer, themida, VM Protect and so on [2]-[4]. However, seed malwares are not that many. If we can classify malware family based on big data, we do not need all the signature for malware variants. Since the attacker side can easily create malware variants automatically, the defensive side can also handle them automatically.

Related researched has been studied. Bayer *et al.* studied about dynamic analysis of malware for collecting execution traces [5]. They generalized the execution traces into pre-defined profile and feed them to efficient scalable clustering algorithm. Their method fundamentally inherits strong points and weak points of dynamic analysis. Best benefit of dynamic analysis is that we do not care about the packing of malware. On memory, packed binary can only be resolved for behavior. The critical problem is trace dependency, which is based only on the one or more specific execution paths. Nataraj *et al*. proposed the novel method to visualize malware and classification method [6]. In previous approaches, researcher consider the malware as only program, code, or bundle of instruction. However, they represent the malware as binary string of zeros and ones. They reshape that into a matrix and viewed as an image. From pre-processed malware images, they used Gabor filter to extract image texture pattern. In experiments, 25 malware families with totally, 9,458 malwares were used. K-nearest neighbors with Euclidean distance with 320 GIST features. The total accuracy is about 97.2%. However, there is a greater contribution to the new approach than the experimental results. They also plus additional features related with dynamic analysis of malware and construct huger database with 63,002 malwares from 531 families [7]. Experimental results with not a sharp machine learning algorithm and large number of groups proves that their features are valid for malware family classification. Ahmadi *et al.* proposed the new features set from static analysis of malware [8]. They tested their method with Microsoft Malware Challenge dataset in Kaggle [9]. Their method achieved a very high accuracy with XGBoost [10]. Kirat *et al.* proposed the new method to extract feature set by signal processing method [11]. They handled malware as 2-D images and they resize them. Sub-band filtering was applied to image and whole image is sliced into sub-block. Finally, they extract GIST features from each block. Unfortunately, there is no experiment on group classification, but there is a great contribution for presenting new effective feature set. Choi *et al.* analyzed the malware with frequency point from signal processing area [12]. Unlike past researches, they represent malware as 1-D signal not 2-D image. They pointed out the code of malware is sequence of instruction and does not have correlation in reshaped 2-D malware image. Their research shows that the signal processing techniques can classify malware families.

In this paper, we analysis structure of the malware code and borrow signal processing techniques. We extract the new concept of feature set from transformed malware and classify the malware family. Experimental results show that the proposed method achieved the good classification performance similar to other feature set.

The rest of this paper is organized as follows. Section II describe the our previous about frequency analysis of malware briefly. In Section III, the proposed feature set are explained. Classification algorithm which we used in our experiments was described in Section IV. In Section V, we provide the experimental results and their interpretation. Finally, we conclude our research in Section VI.

II. Frequency Analysis

We analyzed the malware from the viewpoint of frequency domain of malware signal [12]. Microsoft Malware

Manuscript received May 19, 2018; revised August 17, 2018. This work was supported by Agency for Defense Development (ADD).

Changhee Choi, Kyeongsik Lee, Hwaseong Lee, Ilhoon Jeong, and Hosang Yun are with Agency for Defense Development (ADD), Daejeon, South Korea (e-mail: changhee84@add.re.kr, n0fate@add.re.kr, hslee@add.re.kr, ihjeong@add.re.kr, yun_hosang@add.re.kr).

DOI: 10.7763/IJCTE.2018.V10.1214

Challenge dataset [9] which they used in study provide two representation type of malware payload: byte, asm. We found out asm type file reversed by IDA [13]. It is presumed that byte type is derived from asm type file by IDA script. To separate malware by section, address in asm type was matched with byte type. Note that malware dataset does not have header to prevent possible abuse of active malware. We calculated discrete cosine transform in equation (1) with each section of malware. To raise degree of purity, direct current (DC) and low alternating current (AC) were neglected. We analyzed each transformed malware by family as shown in Fig. 1. Red circle means the malware group signature. Unlike previous precise signature, our signature is flexible so that they can accommodate various malware variants. To capture that flexible signature, we extracted various statistical feature set in this paper as describe in Section III.

$$X(\xi) = \int_{-\infty}^{\infty} x(t)\cos(2\pi\xi t)dt \quad (1)$$

In addition, we test Fourier transform as above procedure in equation (2) to find out more features. Unlike cosine transform, there are real part, imaginary part, magnitude and angle in Fourier transform.

$$X(\xi) = \int_{-\infty}^{\infty} x(t)e^{-2\pi i\xi t}dt \quad (2)$$

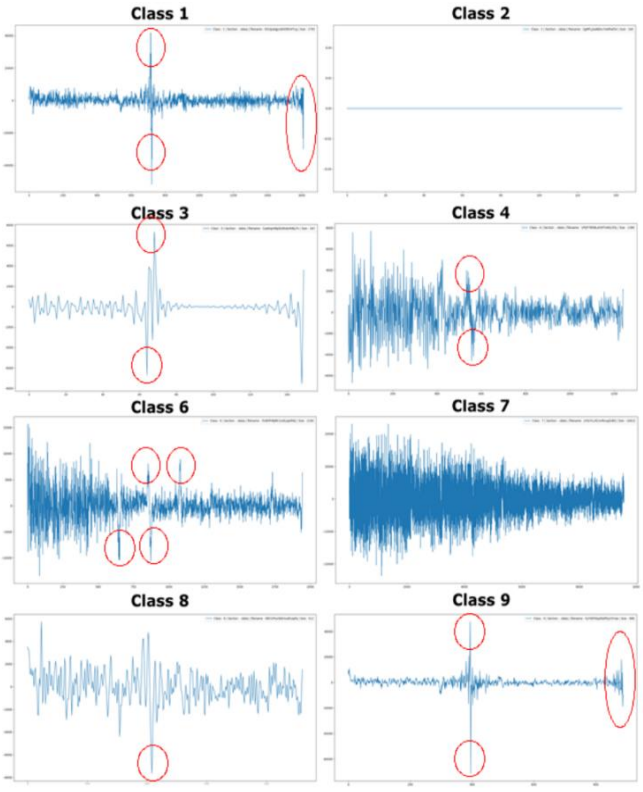

Fig. 1. Idata section of malware transformed by discrete cosine transform.

III. Feature Bank

A. Statistical Features

We collect statistic features from python "scipy.stats" module and Wolfram MathWorld [14], [15]. Our feature set is described in Table I.

TABLE I: Feature Set

Feature name	Description
Geomean	Geometric mean. $(\prod_{i=1}^{n} x_i)$
Kurtosis	Fourth standardized moment.(μ^4/σ^4)
Skewness	Third standardized moment.(μ^3/σ^3)
K-stat(k=1~4)	Unique symmetric unbiased estimator of n-th cumulant k_n. $(k_1 = \mu,\ k_2 = \frac{n}{n-1}m_2,\ \ldots)$
K-stat-var(k=1~4)	Variance of k-stat $(var(k_1) = \frac{k_2}{n},\ var(k_2) = \frac{k_4}{n} + \frac{2k_2^2}{n-1}, \ldots)$
Trim mean	Mean of trimming distribution from both tails.
SNR	Signal to noise ratio.(P_s/P_n)
Bayes_mvs	Bayesian confidence intervals for the mean, variance, and standard deviation
Sem	Standard error of the mean
Iqr	Interquartile range of the data
Chisqure	Test value of Chisqure test
Power_divergence	Test value of Cressie-Read power divergence statistic and goodness of fit test
Willconxon	Test value of Wilcoxon signed-rank test
Jarque_bera	Test value of the Jarque-Bera goodness of fit test
Shapiro	Test value of the Shapiro-Wilk test
Anderson	Test value of the Anderson-Darling test
Circmean	Circular mean.
Circvar	Circular variance.
Circstd	Circular standard deviation.
Moment(n=1~5)	n-th central moment. $(m_k = \frac{1}{2}\sum_{i=1}^{n}(x_i - \bar{x})^k)$

B. N-Gram

A n-gram is a continuous sequence of N items from a give sample of material. In the past, n-gram was used to analyze the text [16]. But recently, n-gram is regarded as the general feature that can be used in various fields. For example, it can be used in author profiling and author classification [17].

There are two ways to apply n-gram to the proposed method. The first is that compute n-gram with each section of malware payload. Since same family of malware has similar instruction sequence and frequency, it can be a valid method. For the most popular Intel instruction set, the length of op-code is from 1 to 3 [18]. To perfectly cover the Intel instruction set, 3-gram is good choice for machine learning. In case of 3-gram, there are 256 * 256 * 256 = 16,777,216 cases. The number of cases will be the number of columns and it cannot be handled by current computer performance. Considering computing power, 1 or 2-gram methods are generally used. This approach was proved in previous research paper [8]. Including this features certainly can import our performance, however, we tested frequency related features only to analyze the performance of the newly proposed feature set only. Second way is that make bins of numerical range. Since maximum and minimum values are different for each feature, we normalize the feature vector. Finally, we calculated 1- gram with 255 bins.

IV. Gradient Boosting Tree

Boosting is a family of machine learning algorithm to combine weak learners into strong learners for reducing variance and bias. There are many boosting algorithms such as adaptive boosting (AdaBoost), linear programming boosting (LPBoost), boosting, and so on [19]. In order to

accommodate large amounts of data, gradient boosting tree algorithms are often used in the machine learning field in recent years. This method has been widely used because of the well-developed XGBoost library [12]. To summarize this algorithm, the whole dataset is divided randomly without replacement into smaller datasets. For each sub-dataset, classification and regression tree (CART) was constructed with objective function and it was trained by optimization. Finally, sub-trees are merged by majority voting algorithm. Fig. 2 described first boosting algorithm proposed in [20] simply [21]. In Fig. 2, the CART is used as learner.

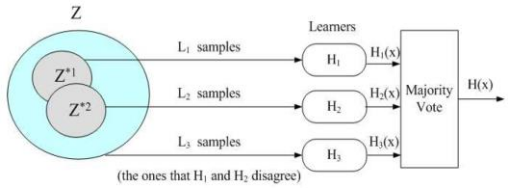

Fig. 2. Simple graphical description of boosting algorithm.

V. Experimental Results

A. Experimental Setup

Our test system is equipped with Intel Xeon E5-2670 (12 cores, @2.3GHz), NVIDIA GeForce GTX 1060 (1152 cores, 3GB), 128GB memory and 2TB SSD storage. In feature extraction phase, CPU intensive operation by multi core system. We use python 3.6 with Anaconda 5.1 which contains scipy, numpy pandas and sklearn. In machine learning phase, we used XGBoost [9] library for python. In our experiments, Microsoft Malware Dataset 2015 was used (9 families, 10,868 samples with label). Number of each family is biased as depicted in Fig. 3.

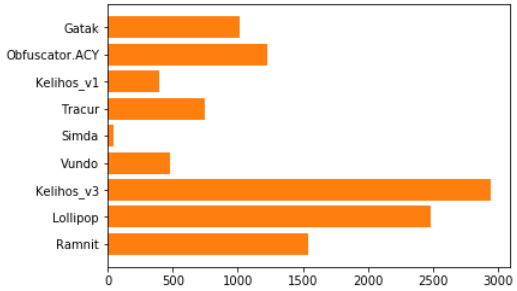

Fig. 3. Malware family distribution in Microsoft malware dataset 2015.

B. Parameter Tuning

To find out best parameter set, we conducted brute-force technic to classification algorithm. In this phase, subsampling for boosting tree is not applied due to computing power. We set the invariant parameter value in tuning process as shown in Table II and Table III shows the parameters in turning process.

TABLE II: Feature Set

Parameter Name	Value
Geomean	Geometric mean. $(\prod_{i=1}^{n} x_i)$
booster	gbtree
objective	multi:softprob
scale_pos weight	1
seed	0
predictor	gpu-predictor

TABLE III: Specification of Parameters in Tuning Process

Parameter Name	Description	Start	End	Step
max_depth	Maximum depth of a tree. It is related with complexity, accuracy and overfitting	10	100	20
eta	Step size shrinkage used in update to prevents overfitting	0.01	1	6
min_child _weight	Minimum sum of instance weight in child	0	10	6
colsample _bytree	Subsample ratio of columns when constructing each tree	0.2	1	5

Finally, we can get the best parameter set: max_depth=10, eta=0.4, min_child_weight=2.0, colsample_bytree=0.6 when logloss = 0.043.

C. Results

We conducted the 5-fold cross validation with optimized parameter set in Section V-B. Total accuracy is 0.987 and f1 score is 0.987. Multi-class log loss is 0.0434.

VI. Editorial Policy

The submitting author is responsible for obtaining agreement of all coauthors and any consent required from.

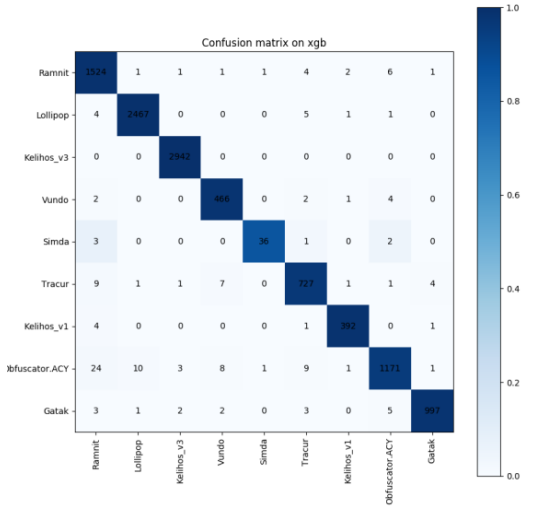

Fig. 4. Confusion matrix without bagging.

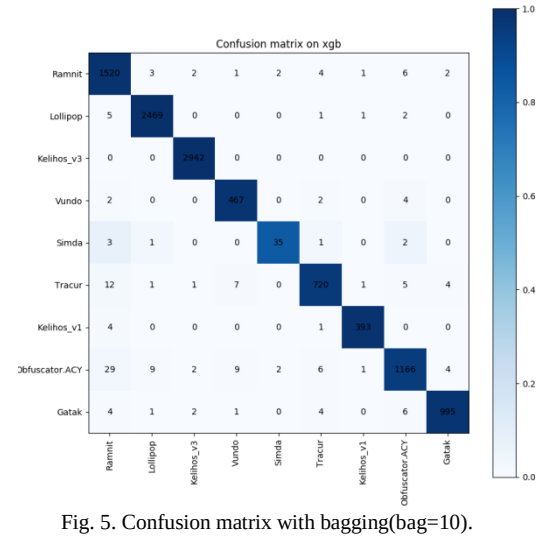

Fig. 5. Confusion matrix with bagging(bag=10).

The confusion matrix is depicted in Fig. 4. This result is quite reasonable, however, it is less than the result of winner of the Kaggle competition [22]. If total accuracy overwhelms all previous feature set, it could be tremendous contribution. Since the purpose of this paper is not increase the accuracy extremely, we focused on invention of novel feature. To reduce overfitting, we perform the bootstrap aggregating method with same condition of previous test. This result is more reliable when unknown test inputs come in. Total accuracy is 0.985 and f1 score is 0.985. Multi-class log loss is 0.0475 as seen in Fig. 5.

VII. Conclusion

In this paper, we investigate the malware code and invent the novel feature set by using signal processing techniques. To prove the proposed method, we extract the statistical features in frequency domain of malware signal. 1-gram (histogram with 255 bins) is also considered in this research. In experiments, we can get the quite reasonable total accuracy 0.985 for classification. Newly invented feature set are expected to be used as a subset for malware family classification.

The grand goal of research of frequency analysis in malware signal is to transform the domain of malware code into proper domain for deep learning. In other words, we want to make homogeneous cells without losing its intrinsic properties. To achieve this goal, we will find out more appropriate domain for malware. Also we will modify previous deep learning network or develop the new one.

References

[1] R. A. Clarke and R. Knake, *Cyber War*, Ecco Press, 2010.
[2] UPX. [Online]. Available: https://upx.github.io
[3] Themida. [Online]. Available: https://www.oreans.com/
[4] VMProtect. [Online]. Available: https://vmpsoft.com
[5] U. Bayer, P. M. Comparetti, C. Hlauschek, C. Kruegel, and E. Krida, "Behavior-based malware clustering," in *Proc. the 16th Annual Network and Distributed System Security Symposium*, 2009, pp. 8-11.
[6] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, "Malware images: Visualization and automatic classification," in *Proc. the 8th International Symposium on Visualization for Cyber Security*, 2011, p. 4.
[7] L. Nataraj, V. Yegneswaran, P. Porras, and J. Zhang, "A comparative assessment of malware classification using binary texture analysis and dynamic analysis," in *Proc. the 4th ACM Workshop on Security and Artificial Intelligence*, 2011, pp. 21-30.
[8] M. Ahmadi, D. Ulyanov, S. Semenov, M. Trofimov, and G. Giacinto, "Novel feature extraction, selection and fusion for effective malware family classification," in *Proc. the 6th ACM Conference on Data and Application Security and Privacy*, 2016, pp. 183-194.
[9] XGBoost. (2018). [Online]. Available: https://github.com/dmlc/xgboost
[10] Microsoft malware classification challenge (Big 2015). *Kaggle*. [Online]. Available: http://www.kaggle.com/c/malware-classification
[11] D. Kirat, L. Nataraj, G. Vigna, and B. S. Manjunath, "SigMal: A static signal processing based malware triage," in *Proc. the 2013 Annual Computer Security Applications Conference*, 2013, pp.89-98.
[12] C. Choi, K. Lee, H. Lee, I. Jeong, C. Yoo, and H. Yun, "Frequency analysis of malware for family classification," in *Proc. the Autumn Conference of Korea Institute Military Science and Technology*, 2017, pp.607-608.
[13] IDA. [Online]. Available: http://www.hex-rays.com
[14] SciPy. [Online]. Available: https://scipy.org
[15] Wolfram MathWorld. [Online]. Available: http://mathworld.wolfram.com
[16] W. B. Cavnar and J. M. Trenkle, "N-gram-based text categorization," *Journal of Ann Arbormi*, vol. 48113, no. 2, pp. 161-175, 1994.
[17] V. Kešelj, F. Peng, N. Cercone, and C. Thomas, "N-gram-based author profiles for authorship attribution," In *Proc. the Pacific Association for Computational Linguistics*, 2013, pp. 255-264.
[18] Intel, Intel 64 and IA-32 architectures, software developer's manual. (2016). [Online]. Available: https://www.intel.com
[19] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pa, *Data Mining, Fourth Edition: Practical Machine Learning Tools and Techniques*, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2016.
[20] R. E. Schapire, "The strength of weak learnability," *Journal of Machine Learning*, vol. 5, no. 2, pp. 197-227, 1990.
[21] R. Polikar, "Ensemble learning," in *Ensemble Machine Learning*, Springer, 2012, pp. 1-34.
[22] Microsoft malware winners' interview: 1st place, "NO to overfitting!" (2015). *Kaggle*. [Online]. Available: http://blog.kaggle.com/2015/05/26/microsoft-malware-winners-interview-1st-place-no-to-overfitting/

Changhee Choi was born in South Korea in 1984. He received the B.S in computer science from Yonsei University of Seoul, South Korea in 2008. He received the M.S and the Ph.D. in computer science from KAIST, Daejeon, South Korea in 2010 and 2013, respectively. In 2013, he joined the Agency for Defense Development (ADD), Daejeon, South Korea. His current research interests are cyber security, digital image forensics, machine learning, and image processing.

Kyeongsik Lee was born in Korea on 1984. He received the BS degree in computer science from Sejong University in 2009. He received the MS degree in information management security, Korea University in 2011. His research interests are digital forensics and incident response.

Hwaseong Lee received the M.S and the Ph.D. in information security from Korea University, Seoul, Korea. In 2013, she joined the Agency for Defense Development (ADD).

IlHoon Jung received the M.S in information security from Korea University, Seoul, Korea, in 2013. He is a senior researcher in Agency for Defense Development (ADD) since 2014. His current research interest focuses on machine learning based cyber security and digital forensic based incident response analysis technique.

Hosang Yun received his M.S in computer science from Korea University, Seoul, Korea, in 1990. He received the Ph.D. in computer science from KAIST, Daejeon, South Korea in 2002. He is a principal researcher in Agency for Defense Development (ADD) since 2000. His current research interest focuses on cyber security and anomaly detection based intrusion detection technique.