The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 18, No. 1, pp.211-216, Feb. 28, 2018. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2018.18.1.211
JIIBC 2018-1-28

멀웨어 검출을 위한 기계학습 알고리즘과 특징 추출에 대한 성능연구

A Study on Performance of ML Algorithms and Feature Extraction to detect Malware

안태현*, 박재균*, 권영만**

Tae-Hyun Ahn*, Jae-Gyun Park*, Young-Man Kwon**

요 약 이 논문에서는 알려지지 않은 PE 파일이 멀웨어의 여부를 분류하는 방법을 연구하였다. 멀웨어 탐지 영역의 분류 문제에서는 특징 추출과 분류가 중요하다. 위와 같은 목적으로 멀웨어 탐지를 위해 우리는 어떠한 특징들이 분류기에 적합한지, 어떠한 분류기가 선택된 특징들에 대해 연구하였다. 그래서 우리는 멀웨어 탐지를 위한 기능과 분류기의 좋은 조합을 찾기 위해 실험하였다. 이를 위해 두 단계로 실험을 실시하였다. 1 단계에서는 Opcode, Windows API, Opcode + Windows API의 특징들을 이용하여 정확도를 비교하였다. 여기에서 Opcode + Windows API 특징이 다른 특징보다 더 좋은 결과를 나타내었다. 2 단계에서는 나이브 베이즈, K-NN, SVM, DT의 분류기들의 AUC 값을 비교하였다. 그 결과 DT의 분류기가 더 좋은 결과 값을 나타내었다.

Abstract In this paper, we studied the way that classify whether unknown PE file is malware or not. In the classification problem of malware detection domain, feature extraction and classifier are important. For that purpose, we studied what the feature is good for classifier and the which classifier is good for the selected feature. So, we try to find the good combination of feature and classifier for detecting malware. For it, we did experiments at two step. In step one, we compared the accuracy of features using Opcode only, Win. API only, the one with both. We founded that the feature, Opcode and Win. API, is better than others. In step two, we compared AUC value of classifiers, Bernoulli Naïve Bayes, K-nearest neighbor, Support Vector Machine and Decision Tree. We founded that Decision Tree is better than others.

Key Words : Malware, PE File Format, Opcode, Windows API Calls, Machine Learning, Bernoulli Naïve Bayes, Decision Tree, Support Vector Machine, K-nearest neighbor

Ⅰ. Introduction

The term "Malware" stands for malicious software, and it usually means the hostile software application. The malware can be discriminated by the capability of replication, propagation, self-execution and corruption of the operating system[1]. According to McAfee's latest report, the large number of new samples for the malware are being distributed every day[2].

Because of the large number of samples for these

*준회원, 을지대학교 의료IT학과
*종신회원, 을지대학교 의료IT학과(교신저자)
접수일자 : 2017년 12월 29일, 수정완료 : 2018년 1월 29일
게재확정일자 : 2018년 2월 9일

Received: 29 December, 2017 / Revised: 29 January, 2018 / Accepted: 9 February, 2018
**Corresponding Author: ymkwon@eulji.ac.kr
Dept. of Medical IT, Eulji University, Korea

new malware, there are many difficulties in detecting and analyzing malware. There is a classical and popular way to detect malware, which is signature based method[3]. Signature was extracted manually from large-scale malware sample data by heuristic-based analysis. However, this signature-based detection systems have several disadvantages and require high maintenance costs to continue signature updates.

Also, malware detection methods can be divided by Static Analysis and Dynamic Analysis[4]. Static Analysis is the testing and evaluation of an application by examining the code without executing the application. So in this method, the performance of detection depends on feature vectors of files. On the other hand, dynamic analysis is the testing and evaluation of an application while executing the program in the virtual environment. It reveals subtle defects or vulnerabilities whose cause is too complex to be discovered by static analysis[5].

Recently, many research efforts has been reported on data mining techniques[6]. These methods use the many kind of feature extraction methods and data mining algorithms. In this paper, our purpose is to compare performance of classifiers and to analyze the correlation between classifiers and features. Therefore, we extracted Opcode and Windows API Calls as the feature vector. And we apply Naïve Bayes (NB), Decision Tree (DT), Support Vector Machine (SVM) and K-nearest neighbor (K-NN) algorithms for malware detection. Also, to evaluate performance and to analyze the correlation, we used Receiver Operating Characteristic (ROC) metric and Analysis of Variance (ANOVA).

II. RELATED WORK

In this paper, we used Static Analysis, that is automated-behavior based malware detection using machine learning algorithm. The algorithms used Bernoulli Naïve Bayes, Decision Tree, Support Vector Machine and K-nearest neighbor

1. PE File Format and Feature vectors

The PE (Portable Executable) file format is an executable file format such as EXE, DLL, Object code used in the Windows operating system. We extracted Opcode and Windows API Calls, and them are used feature vectors. Opcode is a core part of a machine instruction that embodies an operator executed by a machine and provides functions of logical operation, program flow control, memory processing, and arithmetic operation[7]. Windows API Calls stands for Application Programming Interface, which means an interface created so that it can control functions provided by the operating system and programming language so that it can be used in applications.

2. Naive Bayes

The Naive Bayes algorithm is based on Bayes' theorem[8]. It is as follows.

$$p(c_i|w) = p(w|c_i)p(c_i)/p(w) \qquad (1)$$

Where conditional probability $p(c_i|w)$is posterior probability about belong in which classes c_i when feature vector w of data was given. Also, $p(w|c_i)$ is prior probability, $p(c_i)$ is probability of class, and $p(w)$ is probability of feature vector. It selects the class with highest posterior probability, when datasets were given. Also, the naive Bayes is that each of feature is assumed to be independent of each other to obtain posterior probability. Therefore, prior probability is simply computed.

In this paper, we used Bernoulli Naive Bayes because there are two classes(malware or benign). If w_j be the Boolean expressing the occurrence or absence of the $j'th$ term from the feature, then the likelihood of a feature vector w given a class c_i is given by

$$\prod_{i=1}^{n} p_{wj}^{kj}(1-p_{kj})^{(1-w_j)} \quad (2)$$

Where, p_{kj} is the probability of class c_i generating the term w_j[9].

3. Support Vector Machine

Idea of Support Vector Machine is to obtain hyperplane maximizing margin between the classes, where hyperplane is expressed as form of $g(x) = W^t x + b$. Given feature vector w and a class vector c, Support Vector Machine require the solution of the following optimization problem.

$$\min_{w,b,\xi} \frac{1}{2} W^T W + C\sum_{i=1}^{n}\xi_i \quad (3)$$

subject to $c_i(w^T \varnothing(w_i) + b) \geq 1 - \xi_i$

$\xi_i \geq 0, i = 1,...,n$

Where, ξ_i is the variable to allow some degree of misclassification, C is a hyperparameter to control it. And, $\varnothing(w_i)$ is function mapping the data into a higher dimensional space to solve non-linear problem.

From above the problem, using dual problem, we can obtain following classification function[10].

4. K-Nearest Neighbor

K-NN is a type of instance-based learning, and it doesn't construct model. but, it stores instances of the training data. K-NN Classifier implements learning based on nearest K instances from each instance, where we used Euclidean metric to compute distance between the instances. Learning classify class from a simple majority vote of the nearest neighbors of each point[11]. Where it is important to select K properly.

5. Decision Tree

Given feature vector w and a class vector c, a decision tree recursively partitions the space such that the samples with the same labels are grouped together.

Let the data at node m, be represented by Q. For each candidate split $\theta = (w_i, t_m)$ consisting of a feature w_i and threshold t_m, partition the data into $Q_l(\theta)$ and $Q_r(\theta)$ subsets.

The impurity at m is computed using an impurity function $H()$, the choice of which depends on the task being solved[12].

$$G(Q,\theta) = \frac{n_l}{N_m} H(Q_l(\theta)) + \frac{n_r}{N_m} H(Q_r(\theta)) \quad (4)$$

Select the parameters that minimizes the impurity

$$\theta^* = \mathrm{arg}min_\theta G(Q,\theta)$$

For measures of impurity, Gini and Cross-Entropy are used. In this paper, we used following cross-entropy.

$$H(X_m) = -\sum_k p_{mk}\log(p_{mk}) \quad (5)$$

$$p_{mk} = 1/N_m \sum_x I(y_i = k)$$

Where X_m is the training data in node m, and p_{mk} is the proportion of class k observations in node m.

Ⅲ. IMPLEMENTATION OF THE PROPOSED SYSTEM

In this paper, we extracted Opcode and Windows API Calls from PE file using pefile[13] and capstone[14]. And they were used as features. Afterward, machine learning algorithms classified from features to whether PE files are malware or not. Entire flowchart showed in figure 1.

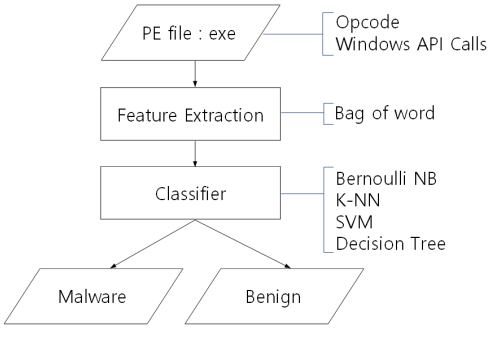


그림 1. 실험순서
Fig. 1. Experimental System

In this paper, we prepared 1224 files for experiment. Among that, the 445 files are benign and extracted from the Windows file directory(windows\system32). In that directory, we exclude the files with non-PE file format. We crawled 779 malware files from the Web sites of "Virusshare[15]" and "malwareurls. joxeankoret[16]", "malc0de[17]", "malwareblacklist[18]" and used it for learning. We extracted malware consists of Trojan 418, PUP 176, Virus 58, Backdoor 34, Adware 29, Downloader 21, Spyware 13, and so on.

To test classifiers, we used scikit-learn library of python. In the case of Bernoulli NB, we set value of alpha to 0.5, where the alpha is hyperparameter used in smoothing of maximum likelihood. In the case of K-NN, because performance is highest when k is 8 of 0-10, we used the number. In the case of SVM, kernel used Radial Basis Function(RBF).

To evaluation performance of classifier, we obtained accuracy, true positive rate and false positive rate. Also, ROC curve and ANOVA are used.

Ⅳ. EXPERIMENT AND RESULT

In the paper, we split dataset into training data and test data, where their rate is 70% and 30%. Also, as a result of feature extraction, the number of unique terms extracted from Opcode is 346, the one extracted from windows API Calls is 693, and the one extracted from both Opcode and Windows API Calls is 1038. In the process of feature extraction, we extracted features (bag of words) as Boolean vector, where Boolean means whether there is each term or not. It was used instead of term frequency. For example, 'move' was extracted from Opcode of PE file 3 times. In the Boolean vector, position of 'move' is 1 or TRUE.

To find the good combination of feature and classifier for detecting malware, we did the experiment as two steps. At the first step, we try to find what the feature is good for classifier. In the next step, we try to find the which classifier is good for the selected feature. We did the first step through 30 times trial to increase confidence of experimental result, and calculated the average and variance of accuracy. It is showed in Figure 2 (The exact values were shown in Table 1). From it, you can see that the feature, Opcode and Win. API, is better than others because of high accuracy and small variance between classifiers. It means that the features is robust for classifiers. Then we proceeded the step two with Opcode and Win. API feature.

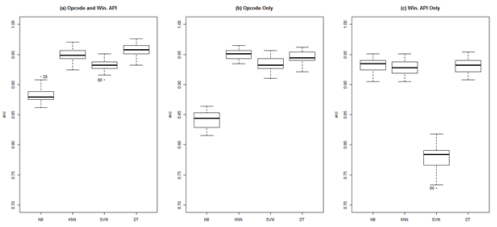


그림 2. 각 특징들의 분류기 정확도 비교
Fig. 2. Accuracy comparison of classifier for each features

In the Step two, we have to find the adequate classifier for that feature. One way to compare classifier is to measure the area under the curve(AUC) that is ROC. The Receiver Operating Characteristic (ROC) curve plot true positive rate(TPR) on the Y axis, and false positive rate(FPR) on the X axis, where TPR is probability of correctly identified malware. Also, FPR is probability of wrongly identified benign, when a detector identifies benign as a malware. The motivation of ROC curve means that the top left corner of the plot is the "ideal" point. However, it is not very realistic, but it does mean that a larger area under the curve (AUC) is usually better[19].

The result of second step is shown in Figure 3 (a), where Decision Tree is better than others. In addition, the ROC curve for other features also were showed in (b) and (c). As summary for experiments, Table 1 showed AUC, mean accuracy and standard deviation within classifier.

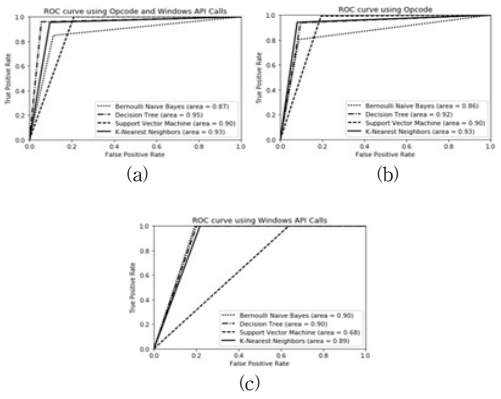


그림 3. 분류기에 따른 ROC 곡선
Fig. 3. ROC curve by Classifier

표 1. 분류 결과
Table 1. Result of Classification

Feature Extraction	Opcode and Win. API			Opcode			Win. API		
	AUC	ACC	SD	AUC	ACC	SD	AUC	ACC	SD
Bernoulli NB	0.87	0.88	0.0130	0.86	0.84	0.0142	0.90	0.93	0.0105
K-NN (K=8)	0.93	0.95	0.0116	0.93	0.95	0.0079	0.89	0.90	0.0923
SVM	0.90	0.93	0.0103	0.90	0.93	0.0101	0.68	0.78	0.0213
Decision Tree	0.95	0.96	0.0106	0.92	0.95	0.0106	0.90	0.93	0.0114

As a result of classification, In the Bernoulli Naïve Bayes, when only using windows API Calls, the accuracy and AUC of the malware detection rate was the highest. However, in the Decision Tree and SVM, the method using Opcode and Win. API showed the highest value of 0.96 and 0.95. Also, When using the decision tree and Opcode and Win. API, the accuracy was the highest at 0.96, AUC was also the highest at 0.95.

V. CONCLUSION

In this paper, we addressed way that classify whether unknown PE file is malware or not. In the classification problem of malware detection domain, feature extraction and classifier are important. For that purpose, we studied what the feature is good for classifier and the which classifier is good for the selected feature. In other word, we found the good combination of feature and classifier for detecting malware. Thus, we did experiments at two step.

In step one, we showed that the feature, Opcode and Win. API, is better than others[20]. Then, in step two, we also showed that way using the feature and Decision Tree is better than others. This results were shown through accuracy and variance of classifier and AUC of ROC curve.

References

[1] G Bala Krishna, V Radha, K Venugopala Rao, “Review of Contemporary Literature on Machine Learning based Malware Analysis and Detection Strategies,” Global Journal of Computer Science and Technology, vol. 16, Issue. 5, version 1.0, pp 11-16, 2016.

[2] B Kolosnjaji, A Zarras, G Webster, C Eckert, “Deep Learning for Classification of Malware System Call sequences,” in Australasian Joint Conference on Artificial Intelligence, pp 137-149, 2016.

[3] Z. Bu et al., McAfee Threats Report: Second Quarter 2012, McAfee Labs, 2012.

[4] Ga-Young Bae et al., “Applying Machine Learning Algorithm to Method for Detecting Malware Using Opcode”, Journal of Korea Information and Communications Society Summer Conference 2016, Vol.60, pp1327-1328, 2016.

[5] Seung-Won Lee, Reversing Important Principles: Malware analyst's reversing talk, Insight, pp 141-143, 2012.

[6] Ye, Yanfang, et al. "A Survey on Malware Detection Using Data Mining Techniques," ACM Computing Surveys (CSUR) vol.50, no.3, 41p, 2017.

[7] Jeong-been Park, Kyoung-Soo Han, Eul-Gyu Im,

"Malware Classification Using Worth Opcodes," Proceedings of the Korea Information Science 2014 Korea Computer Conference, pp943-945, Jun, 2014.

[8] R. Swinburne, "Bayes' Theorem," Philosophical Review of France and the Foreigner, vol. 194, no. 2, pp250-251, 2004.

[9] Python Library, scikit-learn, Bernoulli naïve bayes, http://scikit-learn.org/stable/modules/naive_bayes.html.

[10] Tong, Simon, and Daphne Koller. "Support vector machine active learning with applications to text classification." Journal of machine learning research, pp 45-66, Nov 2001.

[11] Han, Eui-Hong Sam, George Karypis, and Vipin Kumar. "Text categorization using weight adjusted k-nearest neighbor classification." Pacific-asia conference on knowledge discovery and data mining. Springer, Berlin, Heidelberg, 2001.

[12] Safavian, S. Rasoul, and David Landgrebe. "A survey of decision tree classifier methodology." IEEE transactions on systems, man, and cybernetics Vol. 21. No. 3 pp. 660-674, 1991

[13] E. Carrera, Pefile, https://github.com/erocarrera/pefile.

[14] Capstone, capstone, http://www.capstone-engine.org.

[15] virusshare, https://virusshare.com.

[16] joxeankoret, http://malwareurls.joxeankoret.com.

[17] malc0de, http://malc0de.com.

[18] malwareblacklist, http://www.malwareblacklist.com.

[19] Hanley, James A., and Barbara J. McNeil. "The meaning and use of the area under a receiver operating characteristic (ROC) curve." Radiology Vol. 143, No.1 pp 29-36. 1982.

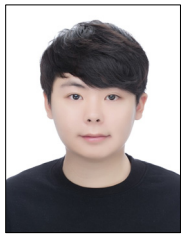
[20] Tae-Hyun Ahn, Sang-Jin Oh, Young-Man Kwon, "Malware Detection Method using Opcode and windows API Calls", The Journal of The Institute of Internet, Broadcasting and Communication, Vol. 17, No. 6, pp. 11-17, Dec 2017.
DOI: https://doi.org/10.7236/JIIBC.2017.17.6.11

저자 소개

안 태 현(준회원)

- 2016.2 을지대학교 의료IT마케팅학과 학사
- 2016.3 ~ 을지대학교 의료IT학과 석사 과정 재학 중

박 재 균(준회원)



- 2012.3 ~ 을지대학교 의료IT학과 학사 과정 재학 중

권 영 만(종신회원)

- 1985.2 KAIST 전기및전자공학과 석사
- 1998.2 KAIST 정보통신공학과 박사수료
- 2007.2 광운대학교 전자공학과 박사
- 1993.3 ~ 을지대학교 의료IT학과 교수

This work was supported by the R.O.K. National Research Foundation under grant NRF-2017R1D1A1B03036372.