Journal of Computers Vol. 30 No. 6, 2019, pp. 73-83
doi:10.3966/199115992019123006006

Visual Malware Classification Using Local and Global Malicious Pattern

Hamad Naeem[1], Bing Guo[1*], Muhammad Rashid Naeem[1], Danish Vasan[2]

[1] College of Computer Science, Sichuan University, Chengdu 610065, China
hamadnaeemh@yahoo.com, guobing@scu.edu.cn, rashidnaeem717@yahoo.com

[2] School of Software Engineering, Tsinghua University, Beijing, China
danish.wasan@gmail.com

Received 10 January 2018; Revised 7 August 2018; Accepted 5 September 2018

Abstract. Recently a huge trend in internet of things and an exponential increase in number of malware are helping malware producers to change malware variants through several automated techniques. Automated techniques may reuse some malware segments to produce variants, and these reuse segments can be helpful to distinguish malware families. Malware variants belonging to same class seem to be much analogous in structure and texture. For this reason, the similarity among malware variants can be used for malware variant family classification. This paper introduces a new malware feature extraction method for capturing local and global properties of images as preliminary features of malware families. The proposed method also reduces the feature dimensions through encoding based feature selection. The experiment is analyzed on three publically available datasets of windows system software. Preliminary experimental results indicate that proposed technique is effective to identify malware family.

Keywords: fisher encoding, GIST, hybrid feature extraction, image visualization, malware classification

1 Introduction

The Internet has become an important part of our daily life. Nearly 57% of the world's population is connected to the Internet [1]. We use it for banking, communicating, entertainment, shopping, and various other commercial and non-commercial activities. While another end, the proliferation of malware at an ever-increasing rate poses a serious threat in the post-internet world. Panda labs identified 227,000 Malware samples every day in 2016. Malware detection and classification has become one of the most critical problems in the field of cyber security [2]. Malware is an abbreviation for Malicious Software. Malware is defined as "Any software that does something that causes harm to a user, a computer, or a network". Malware is in the form of a script, an executable or any other software program. Generally, malware is classified as Worms, Viruses, Trojans, Ransom wares, Adware, Spywares, Bots, PUPs1, Rootkits, Shareware, and other malicious programs.

Malware analysis is a process of both malware detection and classification. Detection is a way of labeling malware as benign or malign. Whereas the classification is a process of figuring out the exact family of given malware. Malware analysis can be carried out in two ways static and dynamic. Static analysis detects binary file without real-time execution of code. It works by capturing format signatures of malware binary. There are various malware signature identification techniques such as string signatures, byte sequence n-grams, API calls and structure of the disassembled program to perform static analysis. According to latest survey of Shabtai et al. [3], static analysis achieved high rate of identification with machine learning classifiers. Also, it is complex due to unpacking of binary and mostly suffers from code obfuscation. Whereas dynamic analysis evaluates binary through real-time execution of code. It executes malware binary in a sandbox environment or Virtual machine. It identifies

* Corresponding Author

malware through system call monitoring, instruction trace and registry tracking activities. Compared to static analysis, dynamic analysis less suffers from code obfuscation. However, dynamic analysis is time intensive, and resource consuming as each malware binary execute for particular time duration [4]. Except for Static and dynamic analysis, there are several different ways to analyze malware binary such as Machine learning based methods, which operate on datasets of malware and benign files. They extract features from files in dataset to train different machine learning classifiers for malware detection.

Since malware authors are producing new malware quickly, and capability to detect unknown malware is still challenging for conventional techniques. Such as these methods are able to detect malware of specific operating system. They cannot work without having to know the nuances of each system. They do not operate without disassembling, unpack and execution. Also, they are less accurate, time and resource consuming. Currently different types of research have been made to detect malware using image processing methods such as graph entropy, image matrices, and texture analysis. However, binary texture features are more powerful to detect code obfuscation as they compute similarity using machine learning classifiers. Some authors tried to mention limitations of binary texture analysis in their studies. For example, Shaid and Maarof [5] highlighted that malware images normally contain information of code, data, text and program resources, while traditional malware image analysis fails to present entire information of malware image. Also, they have failed to detect code obfuscation. Similarly, Qixin et al. [6] mentioned that the binary texture analysis through GIST features is very time-consuming. Thus, methods which can identify modern malware upon being created are needed.

The paper presents a novel Local Global Malicious Pattern (LGMP) which uses hybrid visual features from binary files to detect malware. Firstly, it transforms binary files into 2D signals; Secondly, It extracts hybrid visual features of the image corresponding to each binary file using D-SIFT [7] and GIST [8]; Thirdly, it reduces feature dimensions using encode based feature selection; finally, these features use to train machine learning classifiers. Experimental results presented in Section 4 show that LGMP is simple and efficient; it separates malware families with high classification accuracy. Considering the obtained results LGMP operates without disassembling, and without having to know internal format of executable of any operating system. Also, LGMP is quite promising to extract proper features from binary level data.

Our contributions are:

- LGMP detects malware of any operating system without having to know internal structure of a file.
- LGMP enhances internal similarity between malware families through hybrid local and global features extraction.
- LGMP reduces time consumption and enhances overall detection accuracy through encode based features selection.

The rest of the paper is organized as follows. In Section 2, literature review is presented. In Section 3, the proposed malware analysis technique is described. The experimental results and discussion is explained in Section 4. Finally, the paper is concluded in Section 5.

2 Literature Review

More researches have been conducted on malware visualization to achieve high classification performance and reduce time. These studies classify as either static or dynamic visual analysis.

Trinius et al. [9] proposed dynamic malware detection model to analyze executable by using malware tree map and thread graph visualization. They collected information about API calls and operations of the performed actions in the sandbox. Syed Zainudeen et al. [5] proposed a malware behavior visualization technique. They captured the behavior of malware sample by executing it in a virtual environment. After that, they transformed the behavior to a color image using color map. Their model achieved detection rates ranging from 95.92%-98% while taking the 1102 samples.

Jae et al. [10] introduced static visualization using Image similarity matrix. Firstly, their method selected opcode sequences to generate image matrices and then computed the similarity using selective area matching. Their technique achieved 98% similarity rate. Eul et al. [11] proposed graph-based static visualization. Their method first converted executable files into grayscale images, and then generated entropy graphs from grayscale images. Their model achieved 97.9% similarity rate.

Ban et al. [12] developed a malware detection model that first extracts local features from each

malware image using SURF descriptor and then computes similarity through LSH (Local sensitive hashing) scheme. Their method achieved 85% classification accuracy. Lakshman et al. [8] proposed another static visual analysis method. It first extracts global features from each malware image using GIST descriptor and then computes similarity through the nearest neighbor classifier. Their experimental results showed 97.4% classification accuracy. Aziz and Anita [13] first extracts global features using GIST descriptor and then applies them to feed forward artificial neural network for classification. Their model achieved 96.3% classification rate. Barath et al. [14] introduced a new static visualization technique that first extracts global features using principal component analysis (PCA), and then uses nearest neighbor classification. Their method attained 96% classification accuracy. Kosmidis and Kalloniatis [15] detected malware in 25 families using GIST feature extraction technique. Their model achieved 91.6% detection accuracy using random forest classification.

Recently Agarap and Pepito [16] proposed a deep learning malware detection model. It first converts each malware binary to grayscale image and then trained the following DL models1 to classify each malware family: CNN-SVM, GRU-SVM, and MLP-SVM. The maximum classification accuracy 84.92% achieved with GRU-SVM deep learning model. Similar to above method, Singh et al. [17] presented a CNN based malware image classification approach. Their technique showed 96.08% classification rate for detecting 25 malware families.

Different from other binary texture analysis methods [8, 12-15], LGMP computes cohesion between samples of a family using hybrid local and global features. The main attention of these methods is either on local features' extraction or global features' extraction. Consequently, LGMP reduces misclassification risk and enhances classification rate. Also, our method pays more focus to feature selection and machine learning classification to reduce computational overheads. The methods [16] are faster to compute large datasets than LGMP, while LGMP is more accurate in large scale malware detection.

3 The Proposed Method

LGMP consists of three main stages, namely, malware binary preprocessing, features extraction and classification. In the first stage, it converted binary files into grayscale images for visualization. In the second stage, it used hybrid visual features of the image corresponding to each binary file using D-SIFT and GIST descriptors. Apart this, it reduced the dimensionality of features using encode based features selection. In the final stage, these features applied to train machine learning classifiers. The entire architecture of LGMP shown in Fig. 1. The detailed steps presented below.

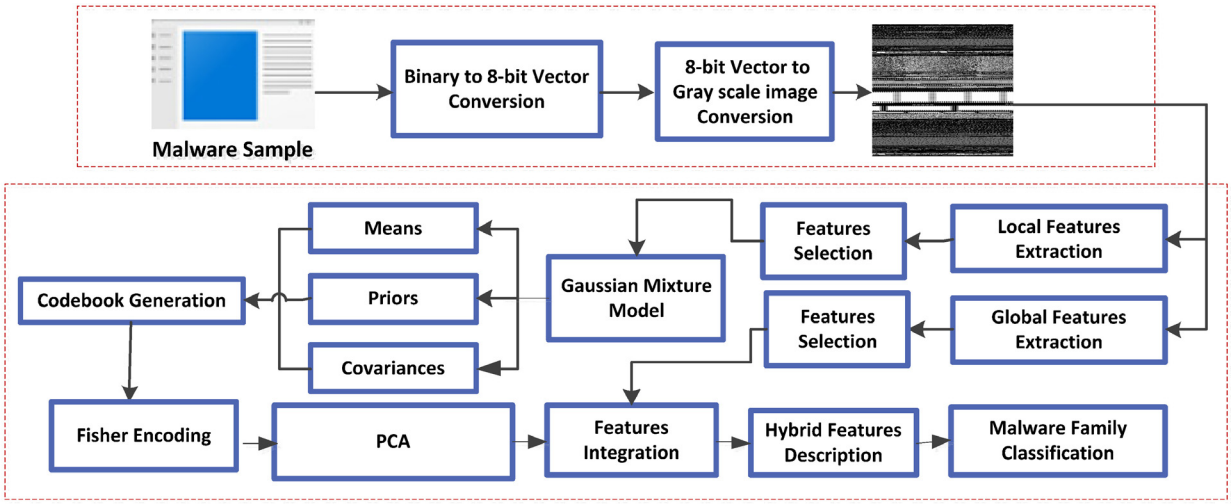


Fig. 1. Methodology of LGMP model

3.1 Malware Binary Preprocessing

The executable file normally contains DOS- header, NT-header, text, data and resource information of a program as shown in Fig. 2. In [8], the authors introduced a method to visualize the structure of an executable file as a grayscale image. LGMP selected their method for image visualization, as many of previous works referred to this standard.

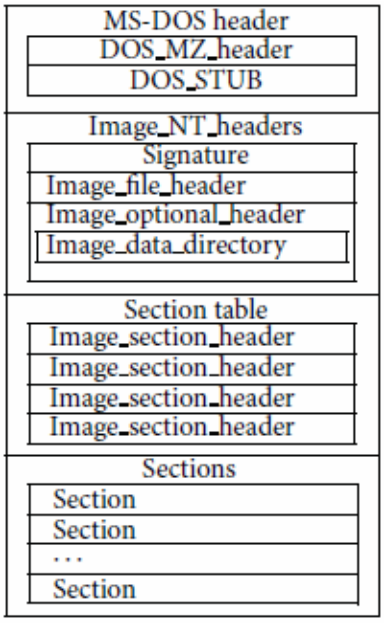
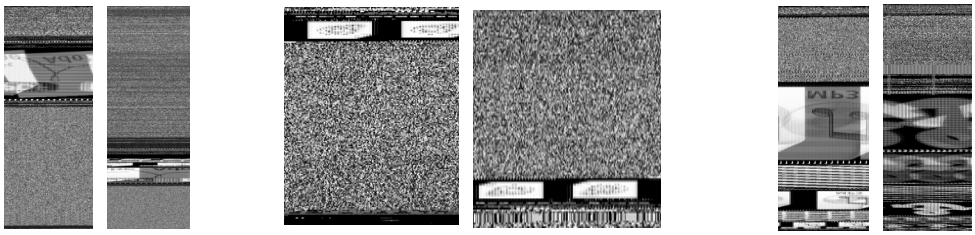


Fig. 2. Various Sections of PE file

In the first step, LGMP separated text, data and resource sections of the executable file and then read the bytes of each section. The process based on bytes to decimal conversion. The decimal displayed a grayscale image in the range [0-255]. The final image consisted of API calls, DLLs, and resources of the program. While rest of the parts such as DOS- header, and NT-header merged at the end of the image file. The width of each gray scale malware was 256 pixels, while height depended on file size. The results indicated that malware image in each family appeared in different texture styles and it was much difficult to find internal similarity between them as shown in Fig. 3. According to results, the images appear in the family always displayed different styles. The only texture features' extraction was not enough to classify malware as malware producers, usually modify program icons and other resources to fraud users.



Malware.Zbot Malware.Rbot Malware.Ircbot

Fig. 3. A chunk of converted malware gray scale images of three malware families

3.2 Feature Extraction

Local features description. In the first phase, LGMP extracted local features of malware image using D-SIFT feature description. D-SIFT firstly selected a dense grid of patches to detect interest points on malware image and then extracted 128-dimensional features from each patch. The total dimensionality of D-SIFT features computed by Eq. (1).

$$D = ceil(\frac{W-(4-1)*\eta_x}{\delta_x})*ceil(\frac{W-(4-1)*\eta_y}{\delta_y})*128 \quad \textbf{(1)}$$

Where W represented the width of the image δ_x and δ_y denoted horizontal and vertical steps respectively. η_x and η_y represented height and width of cell respectively. In the second phase, the Bag of Features (BOF) model used for key features' selection, and reduced total dimensionality of D-SIFT features set. BOF consisted of four important steps. (1) Firstly, the local salient points of malware image detected and then extracted using D-SIFT features descriptor as discussed in the previous phase. (2) Secondly, the dictionary of local features created using feature encoding schemes. To decrease dictionary creation time, fisher vector encoding scheme adopted. Fisher vector encoding executed by learning Gaussian mixture model (GMM). The parameters of GMM estimated by performing expectation-maximization on D-SIFT features' description as shown in Eq (2).

$$p(Y_t;\theta) = \sum_{i=1}^{k=100} \pi_i(Y_t;\mu_i,\sum i) \quad \textbf{(2)}$$

Where π_i, μ_i and $\sum i$ denoted mean, prior and covariance of GMM respectively

(3) Thirdly, the Fisher vector of dimension 2DK for D-SIFT features descriptor computed by concatenation of mean and variance vectors as shown in Eq (3). (4) After apply Fisher vector directly on D-SIFT, it gave suboptimal results. Due to that reason, PCA [17] applied to reduce the dimension of features vector upto 100.

$$\begin{gathered} u_{i=\frac{1}{Z\sqrt{\pi_i}}\sum_{t=1}^{Z} q_i(t)\sum_{i}^{-1/2}(Y_t-\mu_i)} \\ v_{i=\frac{1}{Z\sqrt{2\pi_i}}\sum_{t=1}^{Z} q_i(t)[(Y_t-\mu_i)\sum_{i}^{-1}(Y_t-\mu_i)-1]} \\ \text{Fisher FV}= [\overrightarrow{u_1},\overrightarrow{v_1},.....,\overrightarrow{u_i},\overrightarrow{v_i},] \end{gathered} \quad \textbf{(3)}$$

Global features description. In the first phase, global features of malware image extracted by using GIST features description [8]. GIST firstly used multistate and multidirectional Gabor filtration and then extracted global features of the image. Global features for malware images extracted by using the formula in Eq. (4).

$$\begin{aligned} &g_{mn}(x,y) = a^{-m}g(x^{'},y^{'}), a>1 \\ &x^{'} = a^{-m}(x\cos\theta + y\cos\theta) \\ &y^{'} = a^{-m}(-x\cos\theta + y\cos\theta) \\ &\theta = \frac{n\pi}{n+1} \end{aligned} \quad \textbf{(4)}$$

Where *g(x, y)* denoted Gabor filter, a^{-m} the represented scale factor of wavelet expansion, m denoted number of scales and n represented number of directions, θ represented filter direction respectively. Finally, the dimension of global features reduced up to 256. In our case*, we selected n*=5 and θ=5 for feature selection.

Features integration. The method of merging local and global features is known as features integration. LGMP combined local and global features using gradient weighting scheme as shown in Eq. (5).

$$hybrid\_features = wLocal + (1-w)Global \quad \textbf{(5)}$$

The value of *w* selected according to the contribution of both features. The gradient weight for each pixel of malware image computed using Eq. (6).

$$w(x,y) = 1 - e^{\frac{-((x-x_{feature})^2-(y-y_{feature})^2)}{2\sigma^2}} \quad \textbf{(6)}$$

The dimensionality of proposed hybrid features set was 356.

4 Datasets and Experiments

4.1 Datasets

Three different datasets are used to evaluate the LGMP. Each dataset has samples having different patterns. Dataset number 1 has 1245 files including nine windows network based Trojan malware families. The samples are obtained from vision research lab of University California (https://vision.ece.ucsb.edu/research/signal-processing-malware-analysis). Dataset number 2 has 5195 samples consisting of 25 windows malware families. The samples are obtained from vision research lab of University California (https://vision.ece.ucsb.edu/research/signal-processing-malware-analysis). The samples belong to Trojan, Virus, Worm, PWS, TDownloader, Backdoor and Rogue malware types. Finally, the dataset used in [18] is considered as dataset number 3. (https://drive.google.com/drive/folders/0B8tDVm9mNuusVDJodWkzU3BfUGs). It includes 4000 android malware samples and 2000 benign samples respectively.

4.2 Experimental Setup and Evaluation Metrics

The experiments performed on CPU version Intel i5-4258U @ 240 GHz, the RAM was 4.0 GB, the operating system was Windows 10, and Matlab version R2017a developed the proposed method. Matlab Vlfeat library used to apply machine learning classification on the collected dataset. The LGMP tested on two machine learning classifiers named Nearest Neighbor (KNN) and Support Vector Machine (SVM). However, the main classifier used in this paper is SVM. The reason for this choice is that LGMP achieved better results on that classifier. To evaluate the proposed method, two measures are used which are very popular in the machine learning context. These measures are as follows:

Three kinds of matrices True positive rate (TPR), True negative rate (TNR), False positive rate (FPR), F-measure and accuracy are used to evaluate performance. TP represents the number of malware samples of family A which classifies as A; FN represents the number of malware samples of family A which classifies as malware samples of another family as shown in Eq. (7).

$$TPR=\frac{TP}{TP+FN} \tag{7}$$

FP represents the number of malware samples of another family which classifies as malware samples of family A, and TN represents the number of malware samples of another family which classifies as not A as expressed in Eq (8).

$$FPR=\frac{FP}{FP+TN} \tag{8}$$

The overall classification performance assumed by accuracy, which is equal to the sum of all the families correctly classify divide by the whole number of dataset instances. The formulas of overall accuracy and F-measure shown in Eq (9) and Eq (10) respectively.

$$Accuracy=\frac{TP+TN}{TP+TN+FP+FN} \tag{9}$$

$$F-measure=\frac{2TP}{2TP+FP+FN} \tag{10}$$

4.3 Experimental Results

To evaluate the proposed LGMP, the two most common classification methods compared by randomly selecting the training set of 0.8 and a test set of 0.2 from malware dataset number3. The experiment performed on two traditional classifiers named KNN and SVM. The classification outcomes for both methods shown in Table. 1. The results showed that SVM was superior to KNN and SVM for LGMP. Due to the risk of the imbalanced dataset, the F-measure chose for comparison of all classifiers. The F-measure of SVM was 0.23% higher than that of KNN. Using SVM Classifier, most of the performance indicators achieved better results. Due to this reason, we considered SVM classifier for performing further experiments in this paper.

Table 1. Comparison of different classification algorithms

Classifier	Feature Size	TPR (%)	FPR (%)	F-measure (%)	Accuracy (%)
KNN=5	512	93.85	0.18	93.79	93.85
SVM	512	94.08	0.17	93.92	94.08

To evaluate performance of LGMP, the obtained results compared with those proposed in [19] and [8, 13-15] using three different datasets. Obtained results indicate accuracy and F-measure of the LGMP (Table 2, Table 3, Table 4; Fig. 4, Fig. 5, Fig. 6).

Table 2. Comparing accuracy of the LGMP using dataset-1

Sr. No.	No. of Training Samples (%)	Mean Accuracy (%)			
		LGMP		GIST [8, 13-15]	LBP [19]
		Encode based Feature Selection	Cluster based Feature Selection		
1	10	83.32	75.10	74.49	68.69
2	20	83.12	81.42	80.53	70.39
3	30	85.32	82.71	80.31	71.35
4	40	86.17	83.85	80.37	72.35
5	50	89.32	82.99	81.81	73.93
6	60	88.29	83.28	82.12	74.66
7	70	90.30	85.57	81.77	75.02
8	80	90.84	82.97	81.06	71.11
9	90	93.46	85.64	80.20	68.76

Table 3. Comparing accuracy of the LGMP using dataset-2

Sr. No.	No. of Training Samples (%)	Mean Accuracy (%)			
		LGMP		GIST [8, 13-15]	LBP [19]
		Encode based Feature Selection	Cluster based Feature Selection		
1	10	85.18	84.95	85.35	74.55
2	20	90.69	86.71	88.39	74.02
3	30	85.99	87.37	84.13	75.60
4	40	90.12	87.19	85.06	74.86
5	50	90.16	89.06	86.78	76.57
6	60	90.23	89.58	86.69	75.26
7	70	89.92	88.29	86.43	75.42
8	80	92.92	88.46	86.10	78.05
9	90	87.47	87.09	85.59	76.63

Table 4. Comparing accuracy of the LGMP using dataset-3

Sr. No.	No. of Training Samples (%)	Mean Accuracy (%)			
		LGMP		GIST [8, 13-15]	LBP [19]
		Encode based Feature Selection	Cluster based Feature Selection		
1	10	86.68	86.33	87.61	87.04
2	20	90.17	88.87	85.35	86.48
3	30	90.98	89.12	88.42	86.64
4	40	87.64	90.97	90.33	87.06
5	50	84.15	90.45	89.02	87.07
6	60	91.65	90.87	89.55	87.31
7	70	92.29	91.20	90.34	88.29
8	80	92.43	92.31	89.08	89.31
9	90	92.50	92.39	91.00	90.12

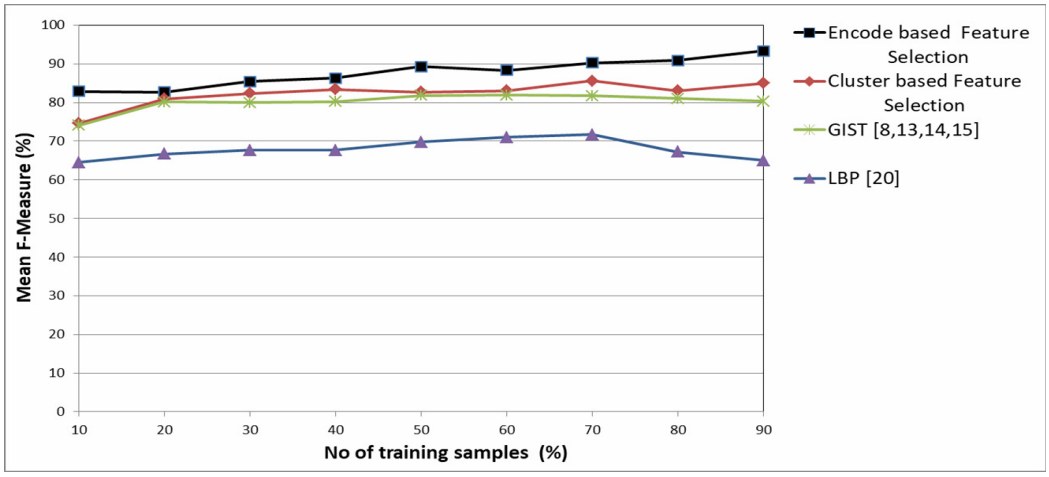


Fig. 4. Comparing F-measure of the LGMP using dataset-1

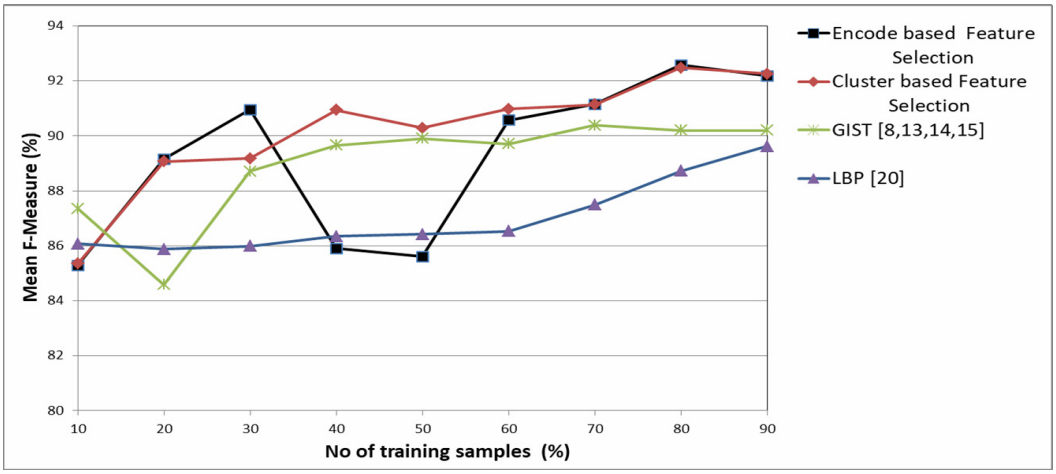


Fig. 5. Comparing F-measure of the LGMP using dataset-2

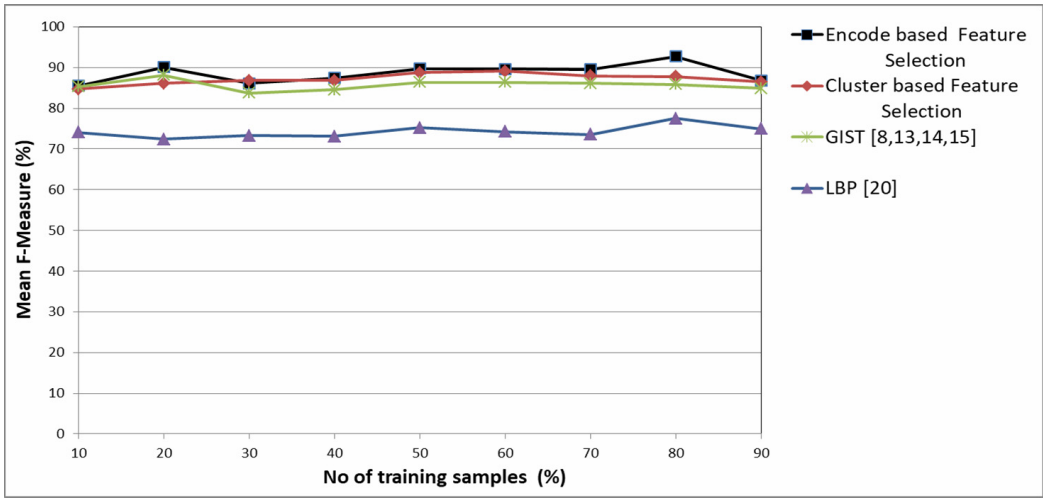


Fig. 6. Comparing F-measure of the LGMP using dataset-3

4.4 Comparison of LGMP with Other Malware Feature Extraction Algorithms

In this paper, two other algorithms choose for comparison with LGMP. One was LBP features of malware images [19], the second was the method proposed in [8, 13-15] which applied GIST features of malware images.

For experimentation with dataset number 1 and 3, we extracted the LBP feature of 59 dimensions and SVM used to classify these features. We also classified GIST feature of 512 dimensions with SVM. Besides, we extracted LGMP feature of 356 dimensions using encode or clustering based feature selection methods and then classified with SVM. For experimentation with dataset number 2, we extracted the LBP feature of 59 dimensions, GIST feature of 512 dimensions and LGMP feature of 512 dimensions using encode or clustering based feature selection methods and then classified with SVM. Each algorithm repeated 9 times for random sampling. For dataset number 1, the best average accuracy of LBP-SVM algorithm on the test set of 9 families was 75.02%, GIST-SVM was 82.12%, LGMP with cluster based feature selection-SVM was 85.64% and LGMP with encode based feature selection-SVM was 93.46%. For dataset number 2, the best average accuracy of LBP-SVM algorithm on the test set of 2 families was 75.02%, GIST-SVM was 88.39%, LGMP with cluster based feature selection-SVM was 89.58% and LGMP with encode based feature selection-SVM was 92.92%. For dataset number 3, the best average accuracy of LBP-SVM algorithm on the test set of 25 families was 90.12%, GIST-SVM was 91%, LGMP with cluster based feature selection-SVM was 92.39% and LGMP with encode based feature selection-SVM was 92.50%.

The weighted average of the precision and recall is the F1 score. It is the harmonic mean of precision and recall, which can be defined as Eq. (10). For experimental results of dataset number 1, LGMP with encode based feature selection-SVM obtained a highest F1 score of 93.31% on average, compared to LBP-SVM algorithm with 71.68%, GIST-SVM with 81.87%, and 85.59% of LGMP with a cluster based feature selection-SVM as shown in Fig. 4. For experimental results of dataset number 2, LGMP with encode based feature selection-SVM obtained a highest F1 score of 92.57% on average, compared to LBP-SVM algorithm with 89.62%, GIST-SVM with 90.38%, and 92.47% of LGMP with a cluster based feature selection-SVM as shown in Fig. 5. For experimental results of dataset number 3, LGMP with encode based feature selection-SVM obtained a highest F1 score of 92.76% on average, compared to LBP-SVM algorithm with 77.49%, GIST-SVM with 88.18%, and 89.12% of LGMP with a cluster based feature selection-SVM as shown in Fig. 6. Hence, it concluded that the classification accuracy of LGMP was over-reliant on neither size of training set nor feature dimensions.

4.5 Impact of LGMP on Run Time Cost

In LGMP the total model training time was much more than feature extraction. Therefore in this paper, the extraction time of different malware feature algorithms compared with LGMP by taking 100 samples. The extraction time of global feature was longer than a local feature, as shown in Fig. 7. The traditional GIST used in proposed methods [8, 13-15] took 13.93 seconds for processing 100 samples. LGMP with a cluster based feature selection was used to extract combine local and global feature, and it took 11.18 seconds for processing 100 samples. While LGMP with an encodes based feature selection took 11 seconds to extract combine local and global feature. Thus, it concluded that LGMP with an encode based feature selection was more suitable to process large-scale malware data as compared to the methods proposed in [8, 13-15]. While it is slightly lighter than LGMP with a cluster based feature selection regarding computation.

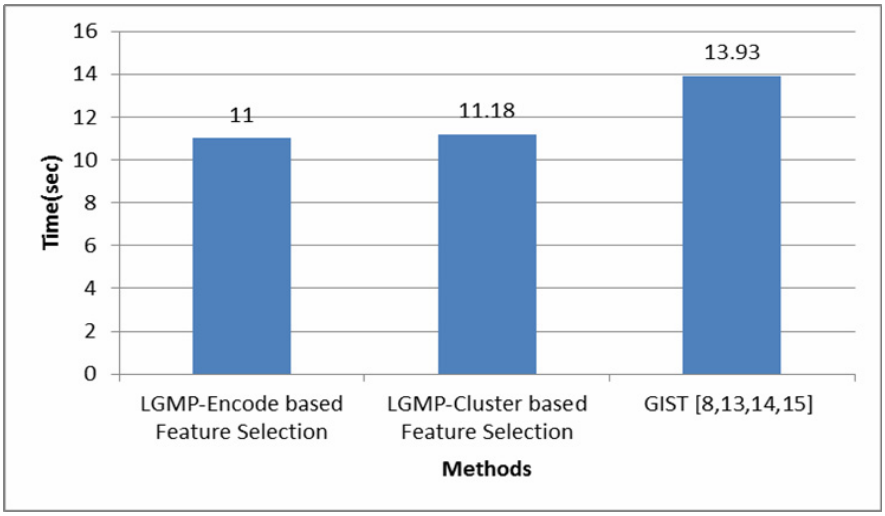


Fig. 7. Comparing F-measure of the LGMP using dataset-1

5 Conclusions and Future Directions

This paper proposed a malware family classification technique using visualized images and hybrid feature extraction model. The contributions of the paper are following:

- A malware visualization method is proposed which converts binary files into 8 bit vector and generates gray-scale images.
- A hybrid feature extraction technique is suggested to calculate similarities of malware using visualized images and to classify malware families.
- Experimental results show that proposed method can classify malware families with high F-measure rate.

In summary, the proposed method can be easily used to pre compute a huge number of known or unknown malware binaries and it does not require binary file disassembling for malware classification. Although, our method provides optimal malicious patterns for classification, but still there is a need to extract stronger malware features for classification. Therefore in future, we will use some more effective malware visualization methods such as binary to rgb image conversion. Another worthwhile research is to try some deep learning models such as CNN for malware image classification.

Acknowledgements

This work was supported in part by the State Key Program of National Natural Science Foundation of China under Grant No.61332001；The National Natural Science Foundation of China under Grant No. 61772352, 61472050; the Science and Technology Planning Project of Sichuan Province under Grant No. 2018ZDZX0010, 2017GZDZX0003, 2018JY0182.

References

[1] ICT Data and Statistics Division, Telecommunication Development Bureau, International Telecommunication Union, ICT Facts and Figures 2016. <https://www.itu.int/en/ITUD/Statistics/Documents/facts/ICTFactsFigures2016.pdf>, 2016.

[2] Pandalabs, Quarterly Report 2016. <http://www.pandasecurity.com/mediacenter/src/uploads/2016/05/Pandalabs-2016-Ti-EN-LR.pdf>, 2016.

[3] A. Shabtai, R. Moskovitch, Y. Elovici, C. Glezer, Detection of malicious code by applying machine learning classifiers on static features: A state-of-the-art survey, Information Security Tech. Report 14(1)(2009) 16-29.

[4] R. Moskovitch, C. Feher, N. Tzachar, E. Berger, M. Gitelman, S. Dolev, Y. Elovici, Unknown Malcode detection using OPCODE representation, in: D. Ortiz-Arroyo, H.L. Larsen, D.D. Zeng, D. Hicks, G. Wagner (Eds.), Intelligence and Security Informatics, Springer, 2008, pp. 204-215.

[5] S.Z.M. Shaid, M.A. Maarof, Malware behaviour visualization, Journal Teknologi 70(5)(2014) 25-33.

[6] W. Qixin, Q. Zheng, Z. Jinxin, Y. Hui, Y. Guangyi, H. Kuangsheng, Android Malware detection using local binary pattern and principal component analysis, in: Proc. 2017 Conference on Pioneering Computer Scientists, Engineers and Educators, 2017.

[7] D. Forsyth, P. Torr, A. Zisserman, Sift Flow: Dense Correspondence across Different Scenes, Springer, 2008.

[8] N.S. Lakshman, G.J. Karthikeyan, B.S. Manjunath, Malware images: visualization and automatic classification, in: Proc. 2011 ACM Conference on Visualization for Cyber Security, 2011.

[9] P. Trinius, T. Holz, J. Gobel, F.C. Freiling, Visual analysis of malware behavior using treemaps and thread graphs, in: Proc. 6th International Workshop on Visualization for Cyber Security, 2009.

[10] H.L. Jae, H. KyoungSoo, G.I. Eul, Malware analysis method using visualization of binary files, in: Proc. 2013 ACM Conference on Research in Adaptive and Convergent Systems, 2013.

[11] G.I. Eul, H. KyoungSoo, H.L. Jae, K. Boojoong, Malware analysis using visualized images and entropy graphs, International Journal of Information Security, 14(2014) 1-14.

[12] X. Ban, L. Chen, W. Hu, Q. Wu, Malware variant detection using similarity search over content fingerprint, in: Proc. 2014 IEEE Conference on Control and Decision, 2014.

[13] M. Aziz, P. Anita, Malware class recognition using image processing techniques, in: Proc. 2017 IEEE Conference on Data Management, Analytics and Innovation, 2017.

[14] N.N. Barath, D.B. Ouboti, M.K. Temesguen, Pattern recognition algorithms for Malware classification, in: Proc. 2016 IEEE Conference of Aerospace and Electronics, 2016.

[15] K. Kosmidis, C. Kalloniatis, Machine Learning and Images for Malware detection and classification, in: Proc. the 21st Pan-Hellenic Conference on Informatics, 2017. .

[16] A.F.M. Agarap, F.J. H. Pepito, Towards building an intelligent anti-malware system: a deep learning approach using Support Vector Machine (SVM) for Malware classification. <https://www.researchgate.net/publication/322221656_Towards_Building_an_Intelligent_Anti-Malware_System_A_Deep_Learning_Approach_using_Support_Vector_Machine_SVM_for_Malware_Classification>, 2017.

[17] A. Singh, A. Handa, N. Kumar, S.K. Shukla, Malware classification using image representation. <https://www.semanticscholar.org/paper/Malware-Classification-Using-Image-Representation-Singh-Handa/32ef4137c175e719d20d1aaec 1231078fe4fed67>, 2019.

[18] I.T. Joliffe, Principal Component Analysis. Springer Series in Statistics, Springer, 2002.

[19] H. TonTon, Y. Chia-Mu, K. Hung-Yu, R2-D2: Color-Inspired Convolutional Neural Network (CNN)-based Android Malware Detection, in: Proc. OWASP AppSec USA, 2017.