Malware Classification Using Recurrence Plots and Deep Neural Network

Sara Sartoli *, Yong Wei†, Shane Hampton‡
Department of Computer Science and Information Systems, University of North Georgia
Dahlonega, Georgia
Email: *sara.sartoli@ung.edu, †yong.wei@ung.edu, ‡schamp5484@ung.edu

***Abstract*—In this paper, we introduce a method for visualizing and classifying malware binaries. A malware binary consists of a series of data points of compiled machine codes that represent programming components. The occurrence and recurrence behavior of these components is determined by the common tasks malware samples in a particular family carry out. Thus, we view a malware binary as a series of emissions generated by an underlying stochastic process and use recurrence plots to transform malware binaries into two-dimensional texture images. We observe that recurrence plot-based malware images have significant visual similarities within the same family and are different from samples in other families. We apply deep CNN classifiers to classify malware samples. The proposed approach does not require creating malware signature or manual feature engineering. Our preliminary experimental results show that the proposed malware representation leads to a higher and more stable accuracy in comparison to directly transforming malware binaries to gray-scale images.**

***Index Terms*—Malware Classification, Deep Learning, Recurrence Plots, Visualization, Static Analysis**

I. Introduction

Malware is a malicious code that performs harmful or unauthorized actions on target computer systems [1]. Traditional malware analysis methods extract and create a model of known malicious behavior such as a specific sequence of instructions, called signature [2]. Creating malware signatures and updating signature databases often needs manual work and cannot cope with the rapid growth of malware-based attack groups as reported by Symantec and AvTest [3], [4]. This results in an increasing number of previously unknown malware that is not detected by traditional virus scanners.

Other approaches to analyze malware employ machine learning and deep learning to identify and classify malware based on their similarities. The classification process consists of two steps: 1) taking a feature vector of an instance to be classified, and 2) mapping the feature vector to a class label [5]. Classification problems are solved by finding a mapping function $f : x \rightarrow y$, that is defined as follows:

$$y = f(x) \tag{1}$$

where x is a feature vector and y is a class label. In many applications, mapping functions between features and classes are so complicated and obtaining analytic forms of these functions is often impossible. Thus, a numerical representation of the mapping function is more desirable. Motivated by neural science research, convolutional neural networks (CNNs) are proposed to address the aforementioned classification challenge. CNNs are well suited for approximating complicated and highly nonlinear mapping functions. CNN-based classifiers have had great success in learning features and image classification tasks [7] [6], as well as in one-dimensional (1D) data classification. Some examples of 1D data classification include using on-body sensor-collected time series to identify human activities [8], speech recognition [9] and biological sequences classification [10]. Researchers have also used CNN in the cybersecurity domain. In the context of malware analysis, Nataraj et. al transform malware binaries to grayscale images in order to take advantage of image processing techniques [11]. Inspired by the similarities among malware images and recent success of CNN, Kalash et al. [12] develop a system that successfully classifies malware using CNN with a higher performance than other alternatives such as Support Vector Machine (SVM). Application of CNN and image processing techniques to malware analysis has been promising. However, this area is not fully explored yet. For instance, recurrence plots (RP) are also known to expose patterns that are caused by the dynamic behavior of processes [13] [14]. This motivated us to explore the research question how effective are recurrence plots for visualizing and classifying malware binaries?

In this paper, we investigate a novel method for visually analyzing malware binaries. The proposed method uses RPs to transform malware binaries into two-dimensional texture images and applies deep CNN classifiers to categorize malware images into families. We perform experiments to compare the performance of deep CNN using recurrence plots and using images directly transformed from malware binaries. Our experimental results show that the RP images not only have significant visual similarities but also lead to higher accuracy than directly transformed malware images.

II. Related Work

In this section, we briefly review the recent works on malware classification using images and deep learning. Interested readers can refer to [2] for a comprehensive survey on malware analysis techniques.

First and second authors contributed equally to this work.

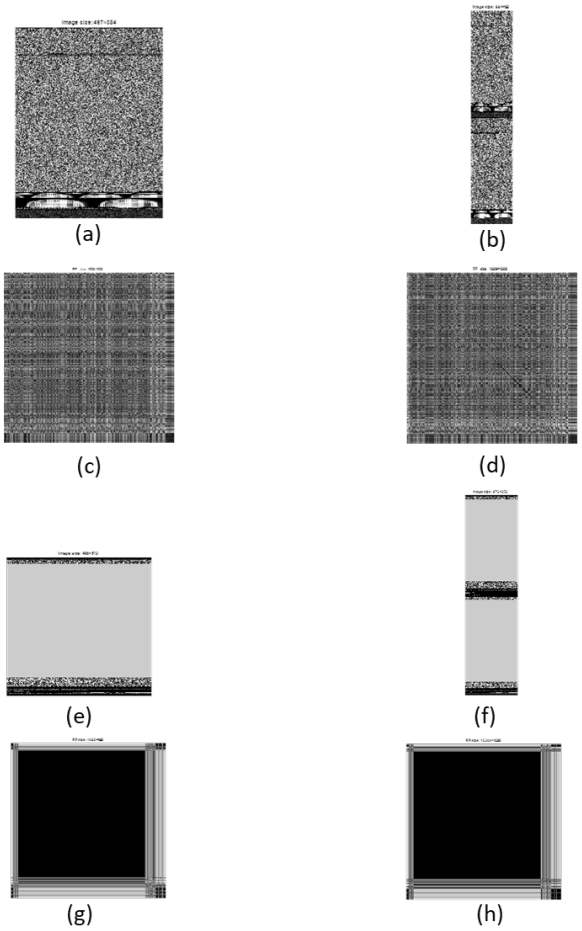

Fig. 1. Comparison of RPs vs. Directly Converted Images with Different Sizes. (a)-(b): Directly converted images from TDownloader family, of sizes 497×384 and 994×192 respectively (Retrieved from Malimg dataset). (c)-(d): RPs of the malware file, of size 4092×4092 and 16308×16308. (e)-(f): Directly converted images of an example in the 11th category, of sizes 488×512 and 976×256. (g)-(h): RPs of the malware file, of size 4092×4092 and 16308×16308.

Nataraj et al. [11] represent malware binaries as grayscale images. More specifically, they first read each binary as a vector of 8-bit unsigned integers. Then they transform the vectors into a two-dimensional array directly in a row-major order. Each element in the two-dimensional array is an integer number between 0 (i.e. black) and 255 (i.e. white). The authors show that malware instances that belong to the same class have similar visual appearances. Using the direct conversion method introduced by [11], Kalash et al. [12] convert malware binary files into grayscale images and develop a CNN-based architectures to classify malware images. They perform experiments on two benchmark datasets, Malimg and Microsoft Malware, and report the accuracy of 98.59% and 99.57%. The proposed system takes advantage of the handcrafted GIST features extracted by Gabor filters. Adjustments of the filter parameters are empirical, and the feature extraction and classification steps are separated. We use CNNs to automatically extract features of the input data, and furthermore, parameters of feature extraction filters are automatically optimized jointly with those of the CNN classifiers during the training process. Besides that we observed that using direct conversion, the malware images change significantly as the image size changes in the same category. On the other hand, the patterns of the RPs of the same malware category are very similar and do not vary when the sizes of data are different (In the above examples shown in Figure 1, 4096 and 16384 data points are used respectively to generate the RPs). One point that is worth emphasizing is that patterns in RPs of different malware categories differentiate from each other clearly. This is a salient feature of RPs when they are used for classification.

Other researchers have used operational code [15], n-gram [17] and system calls as features [18], [20]. For instance, Gibert [15] uses one-dimensional CNN to classify malware instances using x86 instructions. The paper treats opcodes in an executable as words in a sentence and utilize a CNN architecture that is originally introduced by Yoon Kim [16] for sentence classification. In another work, Xiaofeng et. al [18] treats malware instance as dyanmic sequence of behavior. The authors develop a Long Short Term Memory (LSTM) model to classify malware using an API call sequence $X = x_0, x_t, \ldots x_n$, where x_i is a system call function obtained by simulating malware execution in a sandbox. Tang et. al. [19] obtain malware UDP and TCP flows and use recurrence quantification analysis to classify malware traffic data. Unlike the aforementioned work, we do not use dynamic analysis to simulate malware behavior. We use static analysis of malware binaries, yet view instructions in binaries as a series of emissions generated by the underlying dynamic process and model them similar to time series.

In addition to the aforementioned works, this research is also inspired by [22], in which Hatami et. al. converted time series to two-dimensional images using recurrence plots for classification by deep CNN. We view both time series in [22] and malware binary files as one-dimensional sequences generated by an underlying stochastic process. They share similarities of occurrence and recurrence behaviors and can be visualized by RPs.

III. Method

This section describes our method for visualizing (Subsection A) and classifying (Subsection B) malware executable files.

A. Using recurrence plots to visualize malware binaries

Like any software, malware programs contain programming components such as logical branches, iterations, function/system calls, and certain special-purpose code blocks. A malware executable file consists of a series of data points of compiled machine codes that represent programming components. The occurrence and recurrence behavior of these programming components is determined by the common tasks malware samples in a particular family perform. Thus, we view a malware binary as a series of emissions generated by an underlying stochastic process, which is determined

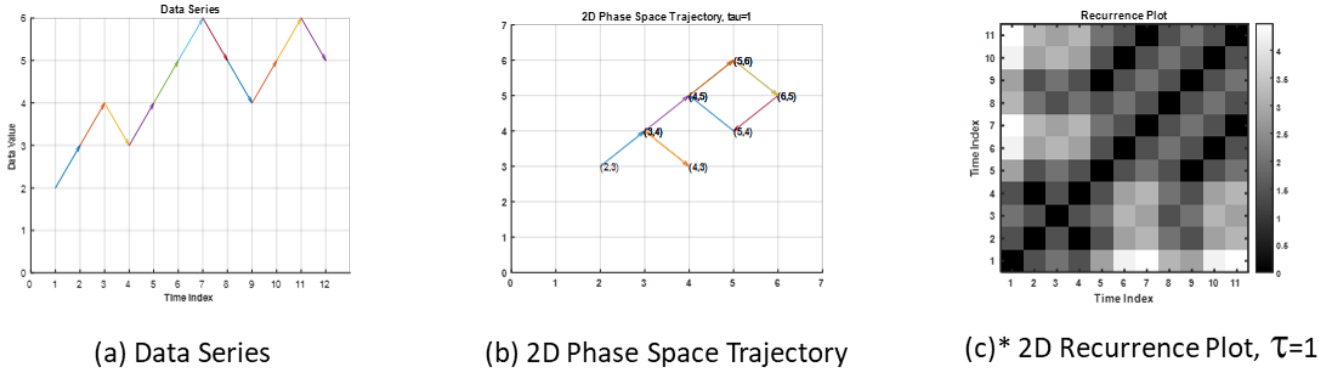

Fig. 2. Steps to calculate recurrence plot. (a) Data Series. The values of data series are [2, 3, 4, 3, 4, 5, 6, 5, 4, 5, 6]. (b) 2D Phase Space Trajectory with embedded time delay of 1 (m=2, τ=1). The 2D states are (2,3), (3,4), (4,3), (3,4), (4,5), (5,6), (6,5), (5,4), (4,5), (5,6), (6,5). Please note that the states (3,4), (4,5), (5,6) and (6,5) are visited twice. (c) Recurrence plot (m=2, τ=1). In the R-matrix, $R_{i,j} = Euclidean\_distance(S_i, S_j)$. *Generated partially using the method in [23].

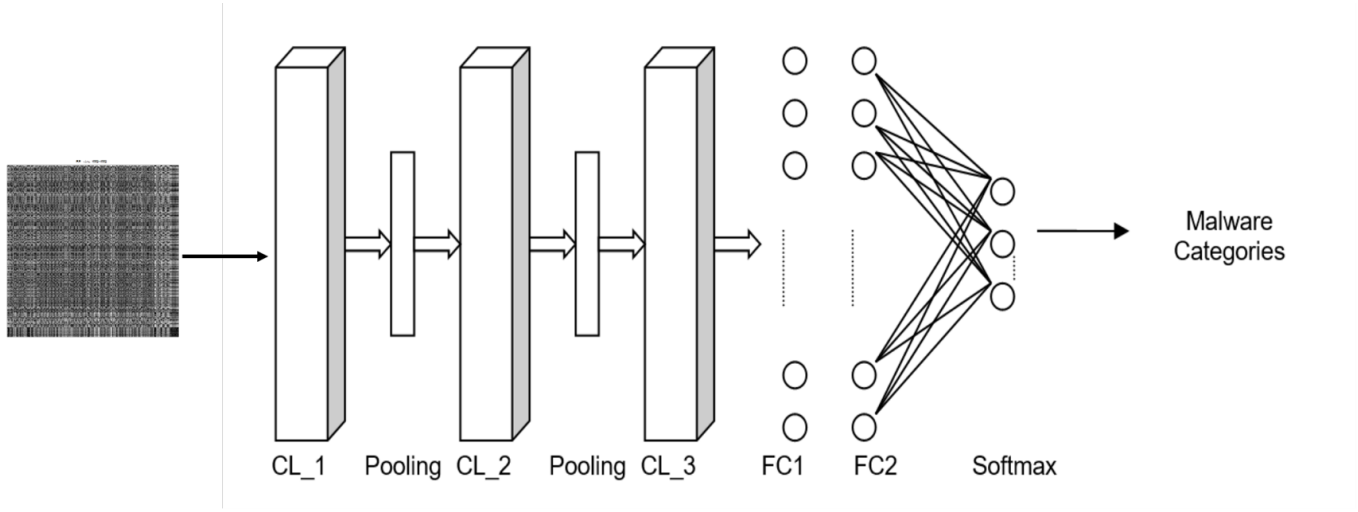

Fig. 3. Architecture of the CNN used in the experiments. CL_1, CL_2, and CL_3 are convolutional layers, FC1 and FC2 are the fully connected layers, and Softmax is the classification layer.

by the tasks that are carried out by the malware. Each malware has a different underlying stochastic process while malware samples belonging to the same category share similar underlying stochastic processes. We use recurrence plots to visualize and characterize the recurrence of states [13] [24]. A state in the phase space is defined as follows:

$$\overrightarrow{S_i} = (u_i, u_{i+1}, ..., u_{i+(m-1)\tau}), \overrightarrow{S_i} \in R^m \quad (2)$$

in which u_i is the i-th observation (or emission) in the data series, m and τ are dimension and embedded time delay of the RP respectively.The definition of recurrence plot is given as follows:

$$R_{i,j} = \Theta(\varepsilon - \| \overrightarrow{S_i} - \overrightarrow{S_j} \|), \overrightarrow{S_i} \in R^m, i, j = \{1, 2, ..., k\} \quad (3)$$

in which ε is a pre-defined threshold, $\| \cdot \|$ is the norm, θ is the Heaviside function, k is the considered number of states. Values of $R_{i,j}$ is one or zero because of the Heaviside function. In our application, to avoid losing detail information due to binarization of thresholding, $\| \cdot \|$ is used in the R-matrix. Thus, the recurrence plots in this paper are two-dimensional gray-level images. Figure 2 shows the steps to calculate a two-dimensional recurrence plot with an embedded time delay of 1 (m=2, τ=1). RP consists of single dots, diagonal, vertical and horizontal lines and blocks. A diagonal line represents local variations, vertical and horizontal lines represent the stability of states. Of course, there are typology patterns in RPs that can not be easily identified visually. Hence, convolutional neural networks are used to learn the deep features of the RPs of malware binary executable files in order to classify them into categories.

B. Convolutional Neural Networks for Malware Classification

We train a CNN to approximate the mapping function between the feature vectors of malware samples and their

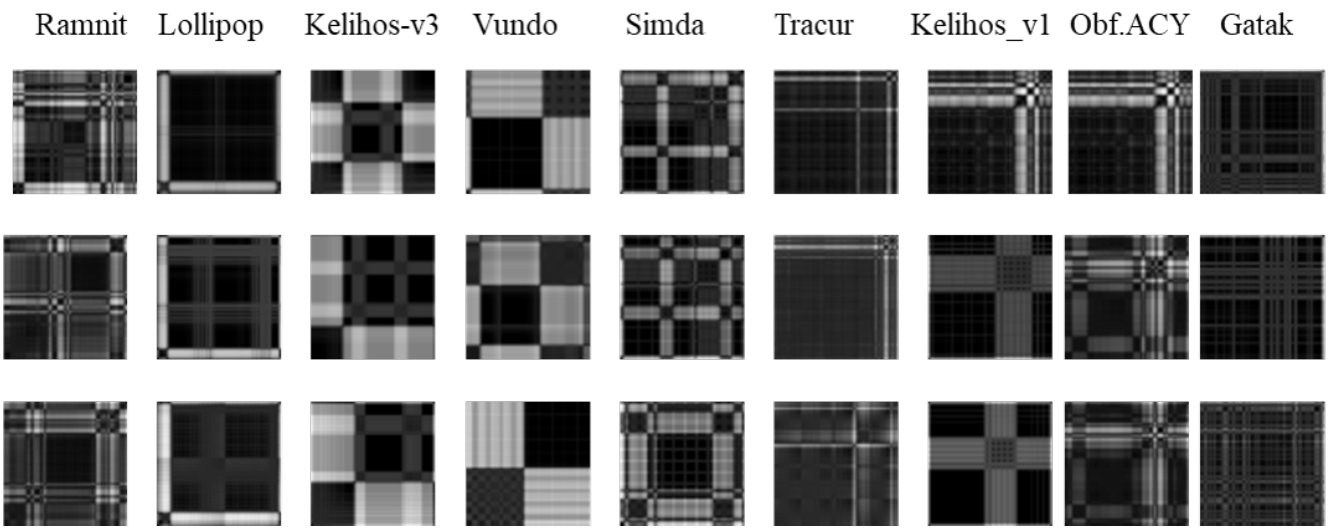

Fig. 4. The images in each column are samples of one malware family

families through supervised learning. In the supervised learning mode, both artificial neural networks (ANNs) and CNNs take pairs of training instances, defined as (V_i, C_i) , in which V_i and C_i are the i-th feature vector and class respectively, to adjust their parameters algorithmically. The performance of an ANN depends on the selection of input features from raw data [25]. The choice of input features is empirical, i.e. trial and error. An advantage of CNNs over ANNs is that CNNs can automatically extract features from raw data. This salient characteristic is achieved by the convolution layers in a CNN [26]. In the case of gray-level images, the behavior of CNN's convolution layers is formulated as follows:

$$actmap(x,y) = \sum_{n_1=-\infty}^{+\infty} \sum_{n_2=-\infty}^{+\infty} g(n_1,n_2) \cdot f(x-n_1, y-n_2) \tag{4}$$

in which g is the gray-level image, f a two-dimensional filter, and $actmap(x,y)$ the output (called activation map) at location (x,y). A convolutional layer consists of a number of filters. The output of each filter (i.e. activation map) can be considered as an abstraction of data. In the proposed malware classification network, there are multiple convolutional layers. A convolution layer sends its data abstractions to the layer following it. This process is viewed as a process of generating abstraction at multiple levels.

The CNN used for experiments consists of three convolutional layers of two-dimensional filters. There is a max-pooling layer following the first two convolutional layers to smooth out their activation maps. The first fully connected layer takes the "flattened" activation maps of the third convolutional layer as its inputs. The softmax layer facilitates multiple-class classifications. The architecture of the CNN used for experiments is depicted in Figure 3.

IV. Experiments

In this experiments we apply CNNs to both recurrence plots and images obtained by direct conversion as proposed by [11]. We compare the performance of classifiers using the two different types of malware images.

TABLE I
Dataset Description

No	Family Name	Description	Samples
1	Ramnit	Worm	1541
2	Lollipop	Adware	2478
3	Kelihos ver3	Backdoor	2942
4	Vundo	Trojan	478
5	Simda	Backdoor	42
6	Tracur	Trojan	751
7	Kelihos ver1	Backdoor	398
8	Obfuscator.ACY	Obfuscated malware	1228
9	Gatak	Backdoor,Trojan	1013

A. Dataset

We perform experiments on 2015 Microsoft malware dataset from Kaggle [27], [28] website. The dataset includes 10,868 malware samples. For each malware, there are two files: a hexadecimal .byte file and a .asm file. The two files are generated using the IDA disassembler tool. The files are labeled with nine malware families: Ramnit, Lollipop, Kelihos ver3, Vundo, Simda, Tracur, Kelihos ver1, Obfuscator.ACY and Gatak.Table I shows each malware family and the number of samples for each family.

B. Evaluation Metrics

We use accuracy, precision and recall to evaluate our experiments. We calculate the performance measures as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

$$Recall = \frac{TP}{TP + FN} \quad (7)$$

In these formulas TP is the true positives, TN is the true negative, FP is the false positive and FN is the false negative.

TABLE II
ACCURACY OF MALWARE CLASSIFICATION, USING RP AND DIRECTLY CONVERTED IMAGES (%)

	Best Accuracy	Mean Accuracy	Standard Deviation
Using RP	97.3	96.8	0.2
Using Direct Conversion	97.4	95.7	7.0

C. Experiments and Evaluation

The system is implemented in Matlab 2019b. All experiments are conducted in CentOS Linux 7 (Core) on a node of 8 NVIDIA Volta V100 GPUs, 16GB/GPU memory and 192GB RAM. The malware sizes in the original dataset are not the same. Thus, for the purpose of CNN training, they are resized to 4096 points by downsampling the binary files. Application of RP with m = 3, τ = 2 is utilized to obtain RP images of 4092×4092. The RP images are resized to 64×64 for CNN training with bilinear interpolation. Figure 4 shows examples of malware RPs of all of 9 families. We observe that malware samples that belong to one family are visually similar (i.e. show similar occurrence and recurrences) and are different from samples in other families. The reason might be that malware have similar behaviors in the same category and thus include similar components.

The size of the 2D filters in the convolutional layers is $w_k \times w_k$, $k \in \{1, 2, 3\}$. In our experiments $w_{1,2,3} \in \{3, 5, 7, 9\}$. The stride of filters in the three convolutional layers are $s_1 \in \{2, 4\}$, $s_2 = s_1/2$, $s_3 = 1$. The number of filters are $n_1 \in \{8, 16\}$, $n_2 = n_1 * 2$, $n_3 = n_1$. The number of neurons in the two fully connected layers are $N_1 \in \{250, 350, 450\}$ and $N_2 = N_1/2$ respectively. The number of outputs of the softmax classification layer is the same as the number of malware classes in experiments. To determine optimal combinations of the aforementioned CNN structural parameters, a parameter sweep is performed. Then simulation of CNNs with the optimal structures are performed using shuffled data for both training and testing. 80% randomly selected data are used to train the CNNs, the rest 20% are used to test their performance. Zero padding is done in a way that the output of a layer has the same size as its input. CNN parameters are updated using the stochastic gradient descent with momentum algorithm with 0.6 momentum. The initial learning rate is 0.01 with a drop rate of 0.1 per every 20 epochs. The maximum number of epochs used in the training is 45.

Table II lists the best and mean accuracies of the CNN with the optimal combination of CNN structural parameters. For the purpose of comparison, accuracies of using images converted directly from executable binary malware files are also listed in the table. Results in Table II show that even though the best classification accuracies of the CNN with the optimal structural parameters are close to the results of using directly converted images (97.3% vs. 97.4%), the CNNs using recurrence plots have a higher mean accuracy than the CNNs using directly converted images (96.8% vs. 95.7%) with a much small standard deviation (0.2% vs. 7.0%). This demonstrates that the RPs represent the characteristics of executable binary malware files better than the directly converted images. Our approach yields more stable classification accuracy than the method using directly converted images. The better performance of classification using RPs is also confirmed by the classification precision and recall, listed in Table III and IV respectively.

V. CONCLUSION

In this work, we propose a novel method to use recurrence plots for the convolutional neural networks in order to classify malware executable binary files. The occurrence and recurrence behaviors of programming components in binary malware files are determined by the tasks a malware program performs. Hence, recurrence plots can represent the characteristics of malware programs. Executable binary files of malware are converted to two-dimensional recurrence plots. Convolutional neural networks with three convolutional and two fully connected layers are trained to classify the recurrence plots, thus malware executable binary files. Experimental results show the proposed method yields higher and more stable classification accuracy, precision, and recall than using directly converted images.

In the future, we plan to apply and evaluate the proposed method to other malware data sets. Furthermore, we plan to perform more grid search to find a better and possibly deeper architecture of convolutional neural networks as researchers have shown that deeper CNN systems reduce the error [29] and improve performance [30]. Another possible future direction is to explore the performance of existing state-of-the-art deep learning systems based on transfer learning. Finally, another interesting future direction is to examine how the application of recurrence plots can be extended to dynamic analysis techniques where malware is executed in a sandbox and its behavior is captured and analyzed.

ACKNOWLEDGMENT

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562. More specifically, the work uses PSC GPU-AI (Bridges GPU Artificial Intelligence) and PSC Storage (Bridges Pylon) through allocation IRI200010P.

TABLE III
MEAN PRECISION OF MALWARE CLASSIFICATION, USING RP AND DIRECTLY CONVERTED IMAGES (%)

Malware Family	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Mean Precision Using RP	94.4	97.3	99.9	96.5	98.8	91.9	97.0	96.3	94.1
Mean Precision Using Directly Converted Images	92.4	96.4	99	95.6	96.7	88.6	97.8	94.2	93.5

TABLE IV
MEAN RECALL OF MALWARE CLASSIFICATION, RP AND DIRECTLY CONVERTED IMAGES (%)

Malware Family	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Mean Recall Using RP	95.9	98.0	100.0	95.6	100.0	90.7	97.2	89.9	98.1
Mean Recall Using Directly Converted Images	93.8	96.1	100.0	94.1	98.7	89.6	95.5	89.3	96.6

DATA AND CODE AVAILABILITY

Code and data used in this work are available upon request. Please email your request to the corresponding author.

REFERENCES

[1] NIST Glossary, available at: https://csrc.nist.gov/glossary/term/malware, accessed on July 2020

[2] Idika, N. and Mathur, A.P., 2007. A survey of malware detection techniques. Purdue University, 48, pp.2007-2.

[3] Symantec Global Internet Security Threat Report, April 2019, available at: https://symantec-enterprise-blogs.security.com/blogs/, accessed on July 2020

[4] AV-TEST Malware Statistics Trends Report, available at: https://www.av-test.org/en/statistics/malware/, accessed on July 2020

[5] Litjens, G., Kooi, T., Bejnordi, B.E., Setio, A.A.A., Ciompi, F., Ghafoorian, M., Van Der Laak, J.A., Van Ginneken, B. and Sánchez, C.I., 2017. A survey on deep learning in medical image analysis. Medical image analysis, 42, pp.60-88.

[6] Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[7] He K, Zhang X, Ren S, Sun J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE international conference on computer vision 2015 (pp. 1026-1034).

[8] Yang, J., Nguyen, M.N., San, P.P., Li, X.L. and Krishnaswamy, S., 2015, June. Deep convolutional neural networks on multichannel time series for human activity recognition. In Twenty-Fourth International Joint Conference on Artificial Intelligence.

[9] Abdel-Hamid, O., Mohamed, A.R., Jiang, H. and Penn, G., 2012, March. Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition. In 2012 IEEE international conference on Acoustics, speech and signal processing (ICASSP) (pp. 4277-4280). IEEE.

[10] Alipanahi, B., Delong, A., Weirauch, M.T. and Frey, B.J., 2015. Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning. Nature biotechnology, 33(8), pp.831-838.

[11] Nataraj, L., Karthikeyan, S., Jacob, G. and Manjunath, B.S., 2011, July. Malware images: visualization and automatic classification. In Proceedings of the 8th international symposium on visualization for cyber security (pp. 1-7).

[12] Kalash, M., Rochan, M., Mohammed, N., Bruce, N.D., Wang, Y. and Iqbal, F., 2018, February. Malware classification with deep convolutional neural networks. In 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS) (pp. 1-5). IEEE.

[13] Eckmann, J.P., Kamphorst, S.O. and Ruelle, D., 1995. Recurrence plots of dynamical systems. World Scientific Series on Nonlinear Science Series A, 16, pp.441-446.

[14] Silva, D.F., De Souza, V.M. and Batista, G.E., 2013, December. Time series classification using compression distance of recurrence plots. In 2013 IEEE 13th International Conference on Data Mining (pp. 687-696). IEEE.

[15] Gibert, D., 2016. Convolutional neural networks for malware classification. University Rovira i Virgili, Tarragona, Spain.

[16] Kim, Y., 2014. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[17] Pang, Y., Xue, X. and Namin, A.S., 2015, December. Predicting vulnerable software components through n-gram analysis and statistical feature selection. In 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA) (pp. 543-548). IEEE.

[18] Xiaofeng, L., Xiao, Z., Fangshuo, J., Shengwei, Y. and Jing, S., 2018. ASSCA: API based sequence and statistics features combined malware detection architecture. Procedia Computer Science, 129, pp.248-256.

[19] Tang, Zheng-Zhi, Xuewen Zeng, Zhichuan Guo, and Mangu Song. "Malware Traffic Classification Based on Recurrence Quantification Analysis." IJ Network Security 22, no. 3 (2020): 449-459.

[20] Kolosnjaji, B., Zarras, A., Webster, G. and Eckert, C., 2016, December. Deep learning for classification of malware system call sequences. In Australasian Joint Conference on Artificial Intelligence (pp. 137-149). Springer, Cham.

[21] Ahmadi, M., Ulyanov, D., Semenov, S., Trofimov, M. and Giacinto, G., 2016, March. Novel feature extraction, selection and fusion for effective malware family classification. In Proceedings of the sixth ACM conference on data and application security and privacy (pp. 183-194).

[22] Hatami, N., Gavet, Y. and Debayle, J., 2018, April. Classification of time-series images using deep convolutional neural networks. In Tenth international conference on machine vision (ICMV 2017) (Vol. 10696, p. 106960Y). International Society for Optics and Photonics.

[23] Chen, Y. and Yang, H., 2012. Multiscale recurrence analysis of long-term nonlinear and nonstationary time series. Chaos, Solitons & Fractals, 45(7), pp.978-987.

[24] Marwan, N. and Kurths, J., 2002. Nonlinear analysis of bivariate data with cross recurrence plots. Physics Letters A, 302(5-6), pp.299-307.

[25] May, R., Dandy, G. and Maier, H., 2011. Review of input variable selection methods for artificial neural networks. Artificial neural networks-methodological advances and biomedical applications, 10, p.16004.

[26] Zeiler, M.D. and Fergus, R., 2014, September. Visualizing and understanding convolutional networks. In European conference on computer vision (pp. 818-833). Springer, Cham.

[27] Kebede, T.M., Djaneye-Boundjou, O., Narayanan, B.N., Ralescu, A. and Kapp, D., 2017, June. Classification of malware programs using autoencoders based deep learning architecture and its application to the microsoft malware classification challenge (big 2015) dataset. In 2017 IEEE National Aerospace and Electronics Conference (NAECON) (pp. 70-75). IEEE.

[28] Microsoft Malware Classification Challenge (BIG 2015), available at https://www.kaggle.com/c/malware-classification, accessed on Sep 23, 2020

[29] Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[30] Abri, F., Siami-Namini, S., Khanghah, M.A., Soltani, F.M. and Namin, A.S., 2019, December. Can machine/deep learning classifiers detect zero-day malware with high accuracy?. In 2019 IEEE International Conference on Big Data (Big Data) (pp. 3252-3259). IEEE.

[31] Towns, J., Cockerill, T., Dahan, M., Foster, I., Gaither, K., Grimshaw, A., Hazlewood, V., Lathrop, S., Lifka, D., Peterson, G.D. and Roskies, R., 2014. XSEDE: accelerating scientific discovery. Computing in science & engineering, 16(5), pp.62-74.