HYBRID FIREFLY-ANFIS ALGORITHM TO DETECT MALICIOUS SOFTWARE

Zahid Manzoor, M.Tech Scholar, Department of Computer Science and Engineering, Desh Bhagat University, Mandi Gobindgarh
Navneet Kaur Sandhu, Assistant Professor, Department of Computer Science and Engineering, Desh Bhagat University, Mandi Gobindgarh

Abstract: Malware is a broad term that refers to a wide range of hostile, intrusive, or annoying software. Android malware is growing and increasing in number and sophistication, which requires the development of more effective malware detection systems. The detection of malware includes mechanisms for identifying and safeguarding against damage by viruses, worms, trojan horses, spyware and other malicious code. The main purpose of this paper is to develop an ANFIS-FA hybrid system to anticipate mobile malware. Firefly algorithm is discussed in this paper to optimize the adaptive neuro-fuzzy inference system. In order to examine the numerical model abilities NS2 simulator is used. For best results, the training and testing values of proposed ANFIS-FA is compared with the existing ANFIS-PSO, ANFIS-DE and ANFIS-ACO.
Keywords: *Malware, Malicious, Software, Firefly Algorithm, Neuro Fuzzy, Mobile Malware.*

1. INTRODUCTION

Malware means malicious software designed to be accessed or installed on the computer without the user's consent. Each software that harms a user, computer or network may be regarded as malware, including viruses, Trojan horses, worms, rootkit, scareware and spyware. It can appear as code, scripts, active content, or other software. Malware is currently mainly used in order to steal sensitive information from others, personal, financial or business information. Sometimes malware is widely used against government or corporate websites to collect or disrupt guarded information. Malware, on the other hand, is frequently used against individuals in order to obtain personal information such as social security numbers, bank or credit card numbers, and so on. Malwares range from simple ones designed to distract or annoy the user to complex ones that capture sensitive data from the host machine and send it to remote servers. Malwares of various types can be found on the Internet. Table 1 discusses some of the more popular ones.

1.1 Damages

- Data Loss

When activated, many viruses and Trojans will try to delete files or wipe hard drives, but even if the infection is detected early, the infected files may have to be deleted.

- Botnets

Many types of malware also attempt to take control of the user's computer, transforming it into a "bot" or "zombie." Hackers create networks of these seized computers, combining their processing power for tasks such as cracking password files or sending out mass emails.

- Account Theft

Many types of malware also attempt to take control of the user's computer, transforming it into a "bot" or "zombie." Hackers create networks of these seized computers, combining their processing power for tasks such as cracking password files or sending out mass emails.

Table 1. Types of Malicious Programs

Type	Example	Description
Virus	Macro Virus, Boot Virus, Logic Bomb Virus.	Attaches itself to a program and propagates copies of itself to other programs.
Trojan Horse	Keylogging Trojans, Backdoor Trojans, Remote Access Trojans.	A program that contains unexpected additional functionality.

Worm	Dabber, Code Red, Doomjuice.	A program that propagates copies of itself to other computers.
Zombie	Flashback, Windigo, Conficker.	Program activated on an infected machine that is activated to launch attacks on other machines.
Spam	Comment Spam, E-mail Spam, Trackback Spam.	Used to send large volumes of unwanted e-mail.
Adware	Pop-up Ads	A malware that automatically delivers advertisement.

- Financial Losses

A keylogger allows a hacker to gain access to a credit card or bank account, which he can then use to run up charges or drain the account.

The remainder of the paper is structured as follows: Section 2 depicts a review of the literature on malware detection techniques. The proposed firefly and ANFIS method is described in Section 3. Section 4 delves into the experimental setup, simulation environment, performance metrics, results, and observations. Finally, Section 5 contains the conclusion.

2. LITERATURE SURVEY

This section provided a detailed review of malicious software detection techniques used by various researchers in the past. Malware detection [6] is the process of detecting the presence of malware on a host system or determining whether a particular program is malicious or benign. Malware detection methods are classified into three types: signature-based, behavioral-based, and heuristic-based. Signature-based malware detection is the most commonly used method by commercial antiviruses, but it can only be used in cases where the malware is completely known and documented [7]. However, techniques such as machine learning have been used to distinguish between normal and abnormal patterns in suspicious applications. [8] presented a systematic review of machine learning-based malware detection and classification approaches. A total of 67 research papers are reviewed in order to address the problem of malware detection and classification on the Windows platform.

Shhadat *et al.* [9] investigated the performance of machine learning algorithms in the detection and classification of malware. Experiments demonstrate greater precision in all binary and multi-classifiers: Decision-trees are 98.2 percent and Random-Forests for multi-classification are 95.8 percent. However, the accuracy and reminder are slightly lower, perhaps due to the skewed data and low frequency in the original datasets of the benign files. However, the accuracy of the Nave Bayes classifier increased significantly from 55% to 91% for binary classification and from 72.34 percent to 81.8 percent for multi-classification.

Sharma *et al.* [10] presented a malware approach that improved accuracy by using the kaggle Microsoft malware classification dataset. The WEKA GUI-based machine learning tool is used to investigate five classifiers (Random Forest, LMT, NBT, J48 Graft, and REPTree) that detect malware with nearly 100% accuracy. A comparison is made between the top 20 characteristics obtained with fishing points, information gain, gain ratio, chi square, and symmetrical uncertainty features.

SIGPID, a malware detection system that uses permission analysis to cope with the rapid increase in the number of Android malware has been introduced by Li *et al.* [11]. The use of the permission data to determine the most significant permissions which are effective in distinguishing between benign apps and malicious apps is developed instead of extracting and analyzing all Android permissions. SIGPID then uses a classification method based on machine learning to classify various malware families and benign applications.

The assessment was carried out by **Narudin *et al.* [12]** with machine learning classifiers to detect mobile malware effectively, selecting relevant network functions for classification inspection and identifying an ideal classification based on true positive rates. Different learning classifiers for a large collection of file spectrums are evaluated to improve the malware detection result for the optimum classifier for mobile malware detection. Bayes network, multi-layer perceptron, decision tree (J48), K-nearest neighbor, and random forest were chosen as classifiers. The MalGenome Project samples were divided into 49 families, each containing 1,260 Android malware samples, but only 1,000 were used. The machine learning process was divided into three stages: (1) data collection, which included network traffic; (2) feature selection and extraction; and (3) the machine learning classifier. The top 20 free applications from Google Play are chosen for the standard dataset. For the Genome Malware dataset, the experimental results show that the random forest classifier has a detection rate accuracy of 99.99 percent.

Arslan *et al.* [13] developed a static-based methodological approach for distinguishing between malicious and benign applications. With an accuracy rate of 91.95 percent, classification is made for 6,500 malicious and 900 benign entities.

Jerlin and Marimuthi [14] proposed a Rete-based MDNBS technique for detecting and classifying malware in API call sequences. The primary goal of this article is to improve malware detection accuracy for the given dataset. MDNBS, a new classification technique, is used to classify malware as worms, viruses, Trojans, or normal. By implementing the Rete algorithm, a set of rules is generated based on the pattern matching process. The primary benefits of this technique are increased detection rate, reduced time consumption, and reduced computational complexity. TPR, FPR, precision, recall, f-measure, and processing time are used to evaluate the results.

Mahindru and Sangal [15] emphasised the importance of designing a malware detection framework that uses a select set of features to help us determine whether an Android app belongs to the malware or benign class. The experiment was carried out on over 500,000 Android apps. Empirical results demonstrate that the model developed in parallel with each of the four separate machine learning algorithms (that means the highest detection rate of 98.8% for real world application malware is a deeper study algorithm, the highest level in detection of malware, the Y-MLP and nonlinear ensemble decision trees approaches) and rough set analysis as an ingredient sub-sets selection algorithm.

Two different methodologies were discussing **Martinelli et al. [16]** for detecting malicious android-focused samples. The first approach is based on the technique of machine learning and the second approach is based on model control. The model-based checking method results in an accuracy equal to 1 by evaluating 1000 malicious Android applications in the real world. For the sensing and sanitization of other large families the proposed method can be easily applied.

D'Angelo et al. [17] addressed malware detection in the mobile Android environment by introducing a novel approach that makes use of a suitable combination of neural networks. These networks are fed sparse matrices that resemble two-dimensional images and represent signatures of a mobile app's behaviour over time. Autoencoders can also automatically extract the most significant and distinctive features from such matrices that have been shown effective in detecting malware when the network is trained on a reduced number of samples once they have been submitted to an artificial neural network classification system. The resulting framework is shown to be able to outperform more complex and sophisticated methods of machine learning in the classification of malware.

The Malware Classification model was proposed by **Unver and Bakour et al. [18]** to detect malware samples in the Android environment. The proposed model is based on converting certain files to grayscale images from the Android app source. Certain image-based local functions and global characteristics have been extracted from constructed gray-scale image data sets for training the proposed model, including four different types of local features and three different types of global features.

Chen et al. [19] have introduced imbalanced classification methods, including SMOTE, SVM, SVM and C 4.5 cost-sensitive methods (C4.5 CS) to determine malicious network behavior, and ultimately detect malicious applications. To prevent performance degradation, it is proposed to classify

imbalanced data gravitation based (IDGC) algorithms. In addition, authors are developing a simple S-IDGC model to further reduce IDGC time costs without compromising classification performance. In addition, the S-IDGC model is developed. In order to provide users with considerable autonomy such as multiple choices of the desired categorizers or traffic features, a machine learning comparative benchmark prototype is also proposed.

Ma et al. [20] presented an Android malware detection combination method based on the algorithm of the machine learning. Three types of system API datasets: API use data sets (indicates which API the CFG has), API frequency data sets (indicates how many times the API used by CFG) and CFG based API sequence data sets. A 2-class classification model is built for each data set, and the model is used to determine whether the incoming application is malicious. Precision, recall and F-score is evaluated and compared using standard classification measures. The combination method constructs an ensemble model and achieves detection accuracy of 98, 98%.

3. PROPOSED WORK

This section shows how the suggested firefly algorithm was developed utilizing an adaptive neuro fuzzy inference system.

Mobile device ubiquity and appeal are likely to grow in the near future. According to the Global Web Index, in 2015, 80 percent of internet users owned at least one smartphone, and online mobile purchasing increased by 150 percent over 2014. Mobile devices have become targets for cybercriminals such as virus authors and hackers due to their widespread use and the amount of personal information saved on them. Android devices are one of the most targeted platforms due to the size of the market and the open nature of the operating system.

To combat the high number of harmful mobile applications, a number of malware detection strategies have been developed in the literature. Traditional research has provided a hybrid strategy for determining ideal parameters to aid in the detection of mobile malware. A hybrid technique, termed adaptive neuro fuzzy inference system (ANFIS) and particle swarm optimization, was proposed in previous work. However, PSO has several drawbacks. The particle swarm optimization (PSO) algorithm has a poor iterative convergence rate and is prone to falling into a local optimum in high-dimensional space. As a result, a firefly method with an adaptive neuro-fuzzy inference system is developed (ANFIS). The Firefly algorithm (FA) has the benefit of being efficient for certain tasks and requiring a limited number of repetitions.

3.1 Firefly Algorithm

Yang introduced the firefly method [21, 22] as a member of the nature-inspired metaheuristic optimization algorithm family in 2008. It was inspired by the flashing patterns and activity of fireflies. In essence, FA employs the three idealized rules outlined below [23]:

i. Regardless of gender, unisex fireflies are attracted to one other.
ii. ii. The attractiveness is proportional to the brightness, and both diminish as the distance between them rises. As a result, if there are two flashing fireflies, the one with the lower brightness will travel toward the one with the higher brightness. If there is no one brighter than a specific firefly, it will migrate at random.
iii. The landscape of the objective function determines the brightness of a firefly.

Firefly algorithm has two parameters [24]:

- *Intensity of light (I):* The intensity of light is calculated by using equation 1, where I is the intensity of the light and d is the distance.

$$I(d) = \frac{i}{d^2} \qquad (1)$$

- *Attractiveness (λ):* The attractiveness of light is proportionate to the intensity of light of other flies adjacent to it. It is calculated as;

$$\lambda = \lambda_0 \, e^{-\gamma d^2} \qquad (2)$$

where λ_0 is the attractiveness at d=0.

The distance (d) between two fireflies F and f is calculated by;

$$d = \| F\text{-}f \| = \sqrt{\sum (F - f)^2} \qquad (3)$$

During this process, the distance and attractiveness of each firefly from the brighter ones is calculated, and this impacts the moving process of each firefly differently. The fireflies are ranked based on their performance after successfully completing this moving procedure.
In general, the flowchart for the firefly algorithm is as follows [25].

3.2 Adaptive Neuro Fuzzy Inference System (ANFIS)

Techniques involving artificial intelligence and machine learning provide a variety of classification approaches for dealing with day-to-day difficulties. ANFIS is one of the most often used classifiers among these approaches. ANFIS [26, 27] is a smart Neuro-Fuzzy technique for modelling and controlling ill-defined and uncertain systems. It combines the characteristics of neural networks and fuzzy logic. Data may be quickly learned by neural networks.

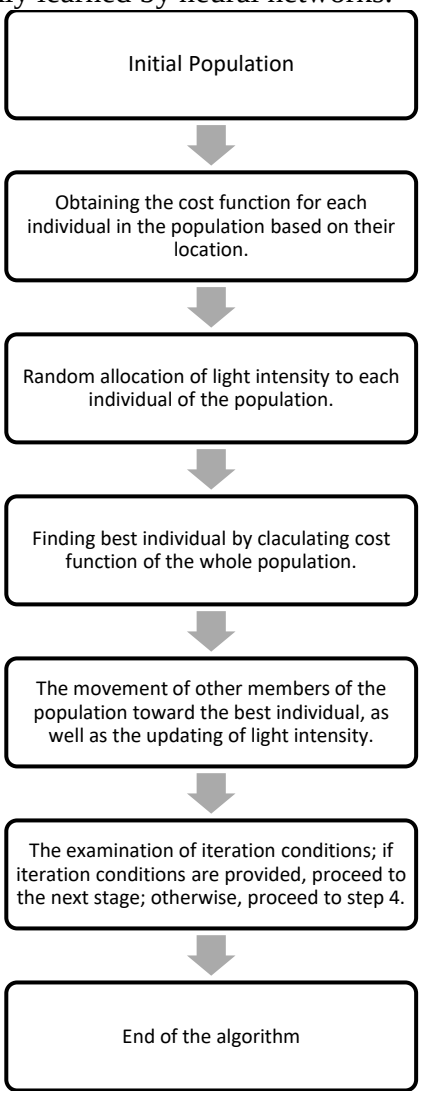

Figure 1. Flowchart of Firefly Algorithm

However, interpreting the knowledge gathered by it is tough because the meaning linked with each neuron and each weight is rather complex to comprehend. Fuzzy logic, on the other hand, cannot learn from data. However, fuzzy-based models are simple to understand since they use linguistic concepts rather than numbers and the structure of IF-THEN rules. Figure 1 depicts a typical ANFIS controller's block diagram.

4. EXPERIMENTAL SETUP

This section contains information on the simulation software, performance metrics, and simulation results. The simulation is carried out on the Windows 10 operating system within the Network Simulator (NS) environment.

4.1 Performance Metrics

The r^2 and RMSE were used to make comparison between the real and predicted values for the soft computing method.

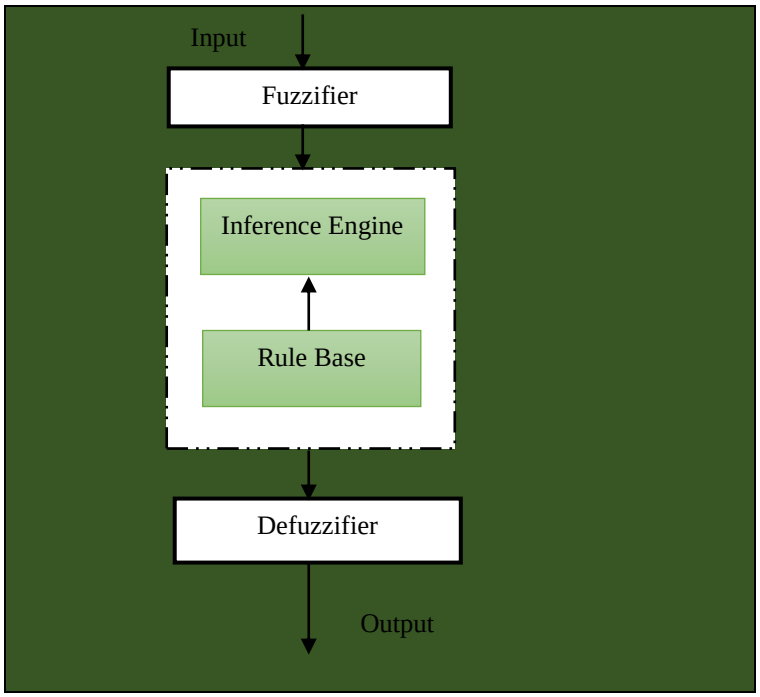

Figure 1. ANFIS Block Diagram

- *Root Mean Sqaure Error (RMSE):* RMSE is defined as the square root of the mean of the square of all of the error.

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(Ai-Pi)^2}{n}}$$

- *Coefficient of Determination (r^2):* Coefficient of determination is used to analyze how differences in one variable can be a explained by a difference in a second variable.

$$r^2 = \frac{\left[\sum_{i=1}^{n}(Ai-Ai')(Pi-Pi')\right]^2}{\sum_{i=1}^{n}(Ai-Ai')\sum_{i=1}^{n}(Pi-Pi')}$$

where n is the total number of test data

A_i is the ANFIS value

P_i is the measurement value

4.2 Performance Results

Table 4.1: RMSE and r^2 Values for Training ANFIS

Parameter	*Training ANFIS-PSO*	*Training ANFIS-ACO*	*Training ANFIS-DE*	*Training ANFIS-FA*
RMSE	0.43133	0.45769	0.44144	0.27435
r^2	0.7692	0.7311	0.7413	0.69892

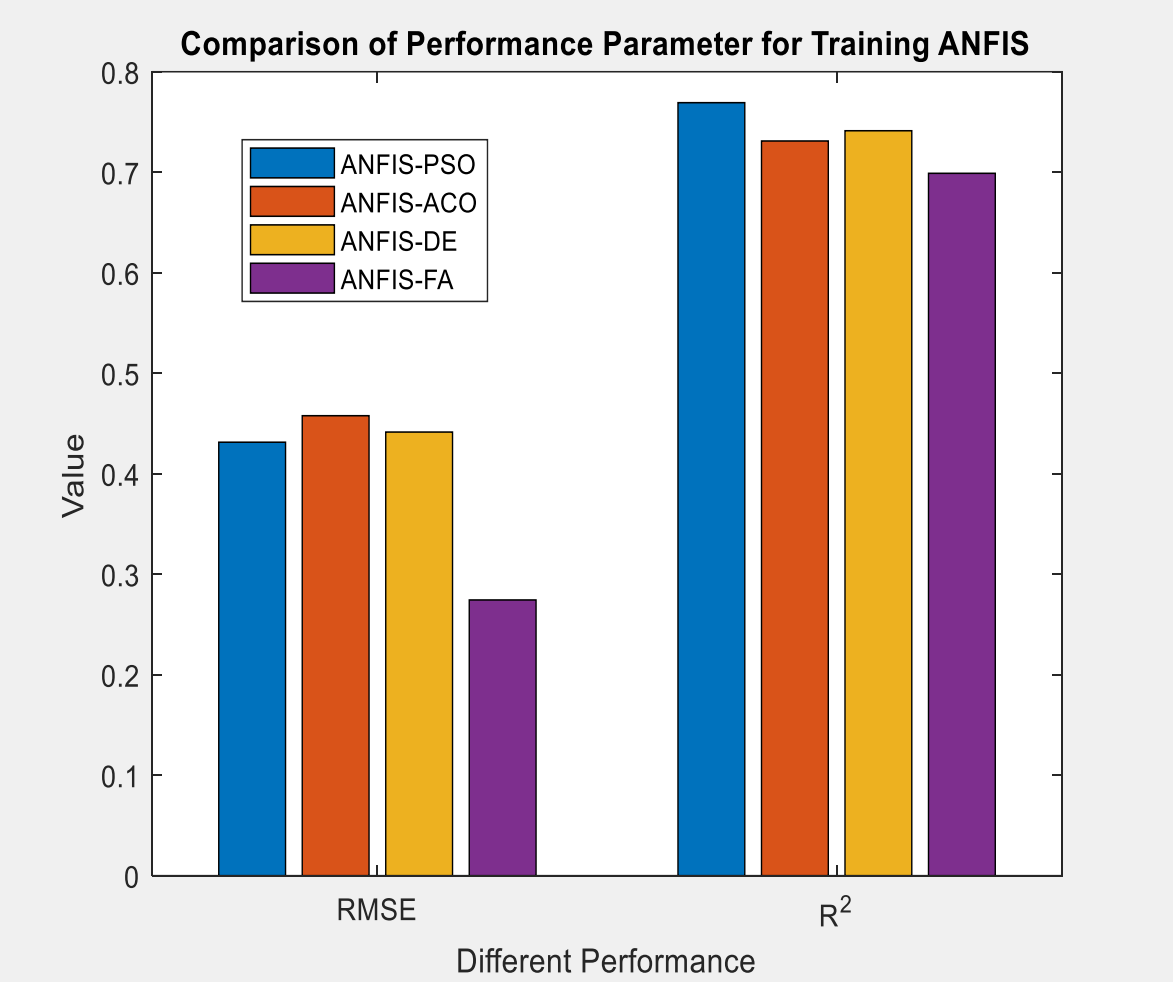

Figure 4.1: Training ANFIS

Table 4.2: RMSE and r^2 Values for Testing ANFIS

Parameter	*Testing ANFIS-PSO*	*Testing ANFIS-ACO*	*Testing ANFIS-DE*	*Testing ANFIS-FA*
RMSE	0.43106	0.45932	0.44244	0.31755
r^2	0.7721	0.7392	0.7562	0.59524

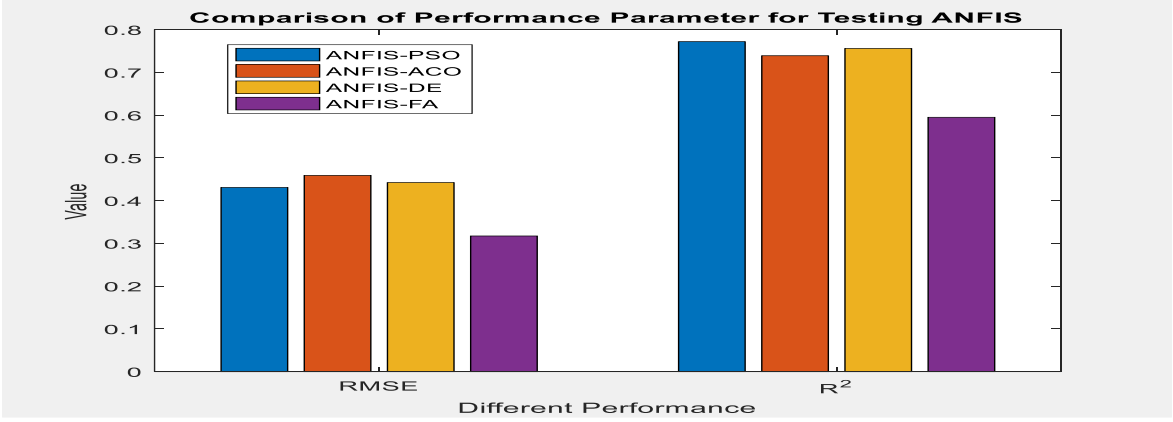

Figure 4.2: Testing ANFIS

In this study, a novel hybrid (ANFIS and FA) method was proposed to forecast the best parameters of a mobile malware analysis. The ANFIS-FA is compared to three hybrid optimization methods: ANFIS-PSO, ANFIS-ACO, and ANFIS-DE. The proposed work's findings demonstrated the utility

of the proposed method. ANFIS-FA, for example, outperforms other approaches with RMSE of 0.27437 in training and 0.3175 in testing. Its coefficient of determination (r2) improves as well, reaching 0.69892 in training and 0.59524 in testing.

5. SUMMARY

A novel hybrid FA+ANFIS method was proposed in this study to forecast the best parameters of a mobile malware detection. The ANFIS-FA is compared with existing ANFIS-PSO, ANFIS-DE and ANFIS-ACO hybrid methods, demonstrating the utility of the proposed method.

In future, it is planned to extend the experiments to other widespread malware threats with the aim to enforce the methodology proposed in this work. Evaluation the proposed method for the detection and the sanitization of malicious payloads in iOS samples will also take into consideration.

REFERENCES

[6] Tchakounté, F., & Hayata, F., "Supervised Learning Based Detection of Malware on Android," *Mobile Security and Privacy*, pp. 101–154, 2017.

[7] Bazrafshan, Z., Hashemi, H., Fard, S. M. H., & Hamzeh, A., "A survey on heuristic malware detection techniques," *The 5th Conference on Information and Knowledge Technology*, pp. 113-120, 2013.

[8] Gibert, D., Mateu, C., & Planes, J., "The rise of machine learning for detection and classification of malware: Research developments, trends and challenges," *Journal of Network and Computer Applications*, 102526, 2020

[9] Shhadat, I., Bataineh, B., Hayajneh, A., & Al-Sharif, Z. A., "The Use of Machine Learning Techniques to Advance the Detection and Classification of Unknown Malware," *Procedia Computer Science*, 170, 917–922, 2020.

[10] Sharma, S., Rama Krishna, C., & Sahay, S. K., "Detection of Advanced Malware by Machine Learning Techniques," 2019.

[11] Li, J., Sun, L., Yan, Q., Li, Z., Srisa-an, W., & Ye, H., "Significant Permission Identification for Machine-Learning-Based Android Malware Detection," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3216–3225, 2018.

[12] Narudin, F. A., Feizollah, A., Anuar, N. B., & Gani, A., "Evaluation of machine learning classifiers for mobile malware detection," *Soft Computing*, vol. 20, no. 1, pp. 343–357, 2014.

[13] Arslan, R. S., Doğru, İ. A., & Barişçi, N., "Permission-Based Malware Detection System for Android Using Machine Learning Techniques," *International Journal of Software Engineering and Knowledge Engineering*, vol. 29, no. 01, pp. 43–61, 2019.

[14] Jerlin, M. A., & Marimuthu, K., "A New Malware Detection System Using Machine Learning Techniques for API Call Sequences," *Journal of Applied Security Research*, vol. 13, no. 1, pp. 45–62, 2017.

[15] Mahindru, A., & Sangal, A. L., "MLDroid—framework for Android malware detection using machine learning techniques," *Neural Computing and Applications*, 2020.

[16] Martinelli, F., Mercaldo, F., Nardone, V., Santone, A., & Vaglini, G., "Model checking and machine learning techniques for HummingBad mobile malware detection and mitigation," *Simulation Modelling Practice and Theory*, 105, 102169, 2020.

[17] D'Angelo, G., Ficco, M., & Palmieri, F., "Malware detection in mobile environments based on Autoencoders and API-images," *Journal of Parallel and Distributed Computing*, 2019.

[18] Ünver, H. M., & Bakour, K., "Android malware detection based on image-based features and machine learning techniques," *SN Applied Sciences*, vol. 2, no. 7, 2020.

[19] Chen, Z., Yan, Q., Han, H., Wang, S., Peng, L., Wang, L., & Yang, B. (2018). Machine learning based mobile malware detection using highly imbalanced network traffic. *Information Sciences*, vol. 433-434, pp. 346–364, 2018.

[20] Ma, Z., Ge, H., Liu, Y., Zhao, M., & Ma, J., "A Combination Method for Android Malware Detection Based on Control Flow Graphs and Machine Learning Algorithms," *IEEE Access*, 2019.

[21] Fister, I., Perc, M., Kamal, S. M., & Fister, I., "A review of chaos-based firefly algorithms: Perspectives and research challenges," *Applied Mathematics and Computation*, vol. 252, pp. 155–165, 2015.

[22] Shahid, A. H., & Singh, M. P., "Computational intelligence techniques for medical diagnosis and prognosis: Problems and current developments," *Biocybernetics and Biomedical Engineering*, vol. 39, 3, pp. 638-672, 2019.

[23] Su, H., & Yue Cai., "Firefly algorithm optimized extreme learning machine for hyperspectral image classification," *2015 23rd International Conference on Geoinformatics*, 2015.

[24] Nandy, S., Sarkar, P. P., & Das, A., "Analysis of a Nature Inspired Firefly Algorithm based Back-propagation Neural Network," *International Journal of Computer Applications*, vol. 43, no. 22, Apr. 2012.

[25] Moghadam, R. G., Izadbakhsh, M. A., Yosefvand, F., & Shabanlou, S., "Optimization of ANFIS network using firefly algorithm for simulating discharge coefficient of side orifices," *Applied Water Science*, vol. 9, no. 4, 2019.

[26] Yawar Rasool Mir, Navneet Kaur Sandhu, "Intrusion Detection System with Deep Learning and Machine Learning Techniques," 2019 JETIR May 2019, Volume 6, Issue 5

[27] Mathur, N., Glesk, I., & Buis, A., "Comparison of adaptive neuro-fuzzy inference system (ANFIS) and Gaussian processes for machine learning (GPML) algorithms for the prediction of skin temperature in lower limb prostheses," *Medical Engineering & Physics*, vol. 38, no. 10, *1083–1089*, 2016.

[28] Al-Hmouz, A., Jun Shen, Al-Hmouz, R., & Jun Yan., "Modeling and Simulation of an Adaptive Neuro-Fuzzy Inference System (ANFIS) for Mobile Learning," *IEEE Transactions on Learning Technologies*, vol. 5, no. 3, pp. 226–237, 2012.