International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 04 | April - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Green Cloud Computing vs Grid Computing: A Study on Energy Efficiency and Environmental Sustainability

Dr. Navneet Kaur Sandhu
Assistant Professor
Desh Bhagat University

Er. Harwinder Singh
Assistant Professor
Desh Bhagat University

The rising demand for computational resources in the digital age has brought significant attention to the environmental impact of data centers and distributed computing infrastructures. This study presents a comparative analysis of energy efficiency in Green Cloud Computing and Grid Computing, focusing on two critical aspects: carbon footprint reduction and load balancing. While cloud computing has evolved with virtualization and dynamic resource allocation to optimize energy usage, grid computing offers decentralized resource sharing with its own efficiency strategies. The concept of "green" in cloud computing introduces energy-aware architectures and renewable energy integration, aiming to minimize ecological impacts. This paper evaluates the operational models, energy consumption patterns, and carbon emission profiles of both paradigms through theoretical modeling and empirical data. Load balancing mechanisms, which directly influence energy use and system performance, are also analyzed to determine their roles in optimizing resource utilization. The findings suggest that green cloud computing, with its integrated energy-aware technologies, generally offers superior energy efficiency and lower carbon emissions compared to traditional grid computing, though each paradigm presents unique strengths in specific use cases. This research contributes to the ongoing efforts in sustainable computing by highlighting key strategies for reducing environmental impact while maintaining high performance and reliability.

1.1 Introduction

Due to an increasing reliance on technology, it is becoming increasingly necessary for online information and data to be protected in every way possible. Data integrity has become one of the most important aspects for organisations to consider as the internet and computer networks grow in size. Security [1, 2] is one of the most important factors to consider when working over the internet, LAN, or other method, regardless of how small or large your company is. While there is no network that is immune to attacks, a stable and efficient network security system is essential to protecting client information. Business can reduce the risk of data theft and sabotage with a good network security system. Using network security, one can protect your workstations from spyware that can harm your computer. Also, it ensures that the data that is shared is kept secure. When information is broken down into many parts and encrypted, it can be transmitted over multiple paths without being intercepted by a third party.

There are a variety of methods available to assist small businesses in preventing unauthorised access to their computer systems. Password authentication is one of the most common methods of user authentication. Because passwords can be guessed or stolen, some businesses employ more sophisticated authentication methods, such as coded ID cards, voice recognition software, retinal scanning systems, or handprint recognition systems. All of these systems ensure that the person attempting to gain access to the computer network is a legitimate user. They also allow you to track computer activity and hold users accountable for how they use the system. E-mails and other external documents can be authenticated using digital signatures. This technology verifies the origin of documents and aids in the prevention of e-mail spoofing.

1.1.1 Advantages and Disadvantages of a Security System

With numerous online threats to both data and identity, computer system security is critical. Regardless of the importance of strong digital security, no security system is perfect. There are numerous types of security systems, each with advantages and disadvantages.

1.1.1.1 Firewall Systems

Firewalls restrict traffic to and from the computer on which they are installed, or to which they are connected if you use a hardware firewall. Suspicious or unauthorised network activity can be detected and blocked, which is an important step in identifying Trojan viruses and hijacking attempts. Firewalls, on the other hand, can flag legitimate programs as having unauthorized access, necessitating the creation of security exceptions and the modification of some settings. They can also slow down your computer's performance because they must be constantly connected to the internet in order to effectively secure your computer and network.

1.1.1.2 Webcam Surveillance

Using a webcam, you can monitor your home or business from the comfort of your own computer. In addition, many of the newer systems offer remote viewing via mobile apps or the Internet. Untrained intruders can disable these cameras, leaving the surveillance target completely unobserved.

1.1.1.3 Password Authentication Systems

A password-authenticated login is required by many websites and software programs. That's especially important if you're storing personal information! A strong password can be difficult to crack and take a long time. It is possible to crack passwords if they are not very strong. This poses a considerable risk, as if someone manages to gain access to your computer, they can be used to lock you out of all of your services and steal your identity. This can be done by trying every possible combination of characters until one finally succeeds.

1.1.1.4 Anti-virus Software

Anti-virus software is the foundation of any security tool set. It scans for malicious software and files and removes them from the computer. Real time protection is included in many anti-virus and anti-malware suites. This feature prevents viruses and other software from being installed in the first place. An unprotected computer is susceptible to infection from a wide variety of viruses. Some of these programs require you to add security exceptions from the program's settings menu during scans and updates. Scans can also take a lot of time, and false positives can cause legitimate files to be blocked or deleted as a result of a false positive. Cybercriminals are constantly developing new ways to defeat anti-virus software.

1.2 Security Attacks

Passive attacks and active attacks are two useful ways to categorise security attacks [4]. A passive attack attempts to learn or use information from the system while having no effect on system resources. An active attack attempts to change or disrupt system resources.

1.3 Security Services

A security service is defined as a service provided by a protocol layer of communicating open systems, which ensures adequate security of the systems or of data transfers. Confidentiality, integrity, authentication and availability (CIAA) are the four security services for a mobile network [6-7].

1.4 Malicious Software

Malicious software (Malware) is a software that is intentionally included or inserted in a system for harmful purpose. Malware is typically composed of code written by cyber attackers with the intent of causing extensive damage to data and systems or gaining unauthorized access to a network. Malware is usually delivered as a link or an e-mail file and the user has to click the link or open the malware file. The majority of malware is introduced when users unknowingly install applications from third-party application stores, and many of the applications in these unofficial stores are repackaged versions of original applications found in Google Play Store [8]. Malicious software can be divided into two categories:

- *Those that need a host program:* These are essentially the fragments of programs that can't exist independently of some actual application program, utility or system program. Typical example of these are viruses, backdoors and logic bombs.
- *Those that are independent:* These are self-contained programs that can be scheduled and run by the operating system. The examples are worms and zombies.

1.5 Meta-heuristic Techniques

Metaheuristic algorithms are the general framework for optimization problems. They are not problem dependent and are heavily deployed in different domains. Due to rise in number of malware, malware detection techniques are updated very often. Different types of metaheuristics algorithms can be classified into following categories (figure 1.7):

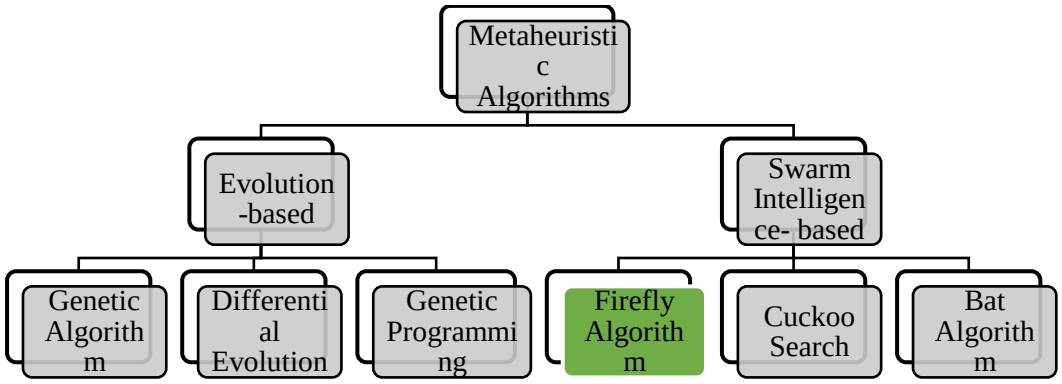

Figure 1.5 Classification of Metaheuristic Algorithm

Literature Survey

Assegie proposed a malware detection and classification model based on the optimal K-nearest Neighbor. The malware detection model proposed is based on the call sequence analysis and classification Application Programming Interface (API). This dataset is collected from the online Kaggle data repository, consisting of 42,797 malicious call sequences (MIPs) and 1,079 non-malicious call sequences (APIs). The KNN algorithm is used to create a model to detect malware for the dataset. Finally, the precision of the malware detection model based on the proposed KNN is evaluated and the results show that 98.17 percent of malware is accurate using this model.

Nguyen *et al.* proposed the lemmatization module for the detection, in accordance with the user permission logs, of the malware intrusion run on mobile devices. Built on data complexity, the physical and logical taxonomies are suggested that the collected raw data be mapped according to the specific domain knowledge into a light-weight semantic formalization. Moreover, in combination with the taxonomy mapping, several selected NLP techniques are used to remove noise from the collected log data and inaccuracies.

Wan *et al.* have developed a cloud-based malware detection system in which mobile devices compete on the resource for limited broadcasting and cooperate to enhance the security server malware detection. In comparison with the Q-learning based scheme, a moving offloading strategy based on hotbooting-Q has been proposed that improves the malware detection performance. The proposed discharge schemes accelerate training speed, improve the accuracy of malware detection, reduce detection time and therefore improve utility.

Dali Zhu *et al.* extracted a total of 323 features using FlowDroid to extract data streams from Android apps (3,000 benign Google Play Store applications and 8,000 malware from The Android Malware Genome Project, VirusShare, etc. Then DeepFlow, a malware detection architecture based on a deep DBN learning mode is implemented. The findings show that DeepFlow significantly outperforms traditional machine learning methods and achieves high F1 results in suitable parameters.

Wei-Ling et *al.* suggested a Robotium program in an Android sandbox that could automatically trigger and monitor Android applications. A robotium program is an automatic trigger program for IT identification that can in meaningful order click mobile applications. The model is 97% accurate and the FPR is less than 4%.

In order to increase traffic classifier efficiency, **Lashkari *et al.* [31]** proposed a mobile malware detection model based on nine traffic features. The model also uses classification methods to distinguish malware families including flow, package-based and time based features. The analysis reveals that the proposed feature set has more than 93% detection precision and a probability of 92% success on characterization with an average false positive rate of less than 0.08%, good enough and necessary to detect malware in real world systems.

Xiao *et al.* has designed a malware detection game based on the cloud and produced a NE of the game that shows how a mobile device selects its download rate to balance transmission costs with the detection rate. For the dynamical game in time varieties of radio networks, a Q-learning malware detection strategy was proposed, which is further improved by the use of the Dyna architecture and the known model of the radio channel. The simulation results reveal that the proposed Malware Detection based on Q-learning improves accuracy by 40%, reducing detection time by 15%, and increasing mobile device utilities by 47% compared to the 100 mobile device benchmark strategy. The PDS malware detection is extremely accurate and useful while the Dyna-Q system offers the quickest response.

Fasano *et al.* proposes an Android platform malware detector that takes 38 features into account in 3 categories, i.e. CPU, Memory and Network, which is then considered to be an input for 4 controlled machine algorithms. With the Hoeffding Tree classification algorithm, an accuracy of 0.928 and a reminder of 0.938 are achieved.

Lu *et al.* presented a SVM-based mechanism to detect the malware and normal apps. The proposed idea scanning and recording features for both required and used permissions of the list. LibSVM is adopted to classify the unknown apps. There are 200 malware application and 200 normal applications in this experiment with an achieved accuracy of 99%.

Malik and Kaushal proposed a method named CREDROID which, by offline analyses of network traffic logs, identifies malicious applications based on their DNS query and the data transmitted by it to remote servers. A pattern-based detection is proposed rather than a signature-based detection which does not identify polymorphic malware. CREDROID is a semi-automated approach that uses various factors like the remote server that connects to the app, sends data and uses the protocol to communicate to identify applications' credibility. The focus of work is network traffic generation in 63 per cent of applications in a normal malware data set.

3.1 Software Used: NS2

NS2 [68] is an event-driven simulator that is free and open source. Research in computer communication networks is the primary focus of the project. NS is a discrete event simulator in which the progression of time is determined by the timing of events as maintained by a scheduler. Since its inception in 1989, NS2 has attracted tremendous interest from industry, academia, and government. NS2 now includes modules for a variety of network components such as routing, transport layer protocol, application, and so on. NS2 is compatible with Linux, Mac, and Windows. Your own protocol can also be added to the code. A simple scripting language makes it easy for researchers to set up NS2 and observe the results. Open-Source Network Simulator NS2 is without a doubt the most widely used network simulator available today.

It covers a wide range of applications (Web, FTP, CBR), protocols (transport and routing protocols), network types (Satellite links, wired and wireless LAN), network elements (mobile nodes, wireless channel models, link and queue models), and traffic models (exponential, uniform). NS-2 is built on a C++ object-oriented simulator and an OTcl interpreter (an object-oriented extension of Tool Command Language TCL).

OTcl scripts written by users are translated by Network Simulator as shown in figure 3.1. In addition, Tcl has a simple syntax and is platform independent. In parallel with the OTcl script's interpretation, NS creates two main analysis reports at once. An example of this is the Network Animator (NAM) object, which displays animations of the simulation.

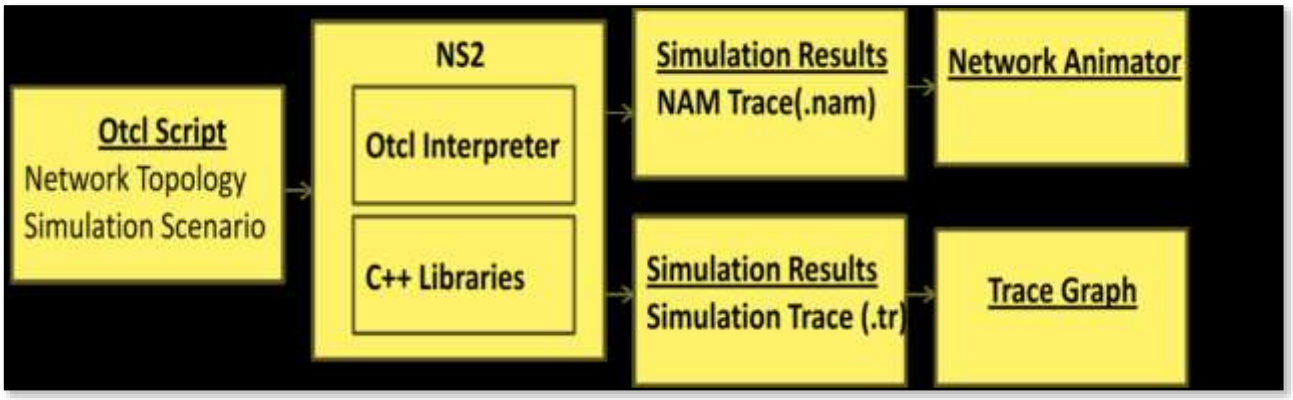

Figure 3.1: Structure of NS2

3.2 Problem Formulation

Mobile device ubiquity is 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.

3.3 Methodology Used

The two techniques used in this work to handle malicious software are discussed as follows:

3.3.1 Firefly Algorithm

Yang introduced the firefly method [69-71] 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 [72]:

a) Regardless of gender, unisex fireflies are attracted to one other.
b) 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.
c) The landscape of the objective function determines the brightness of a firefly.

Firefly algorithm has two parameters [73]:

- *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.

$$I(d) = \frac{i}{d^2} \quad (3.1)$$

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

$$\lambda = \lambda_0 \, e^{-\gamma d^2} \quad (3.2)$$

where λ_0 is the attractiveness at d=0.

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

$$d = \| F\text{-}f \| = \sqrt{\sum (F - f)^2} \quad (3.3)$$

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 shown in figure 3.2 [74]:

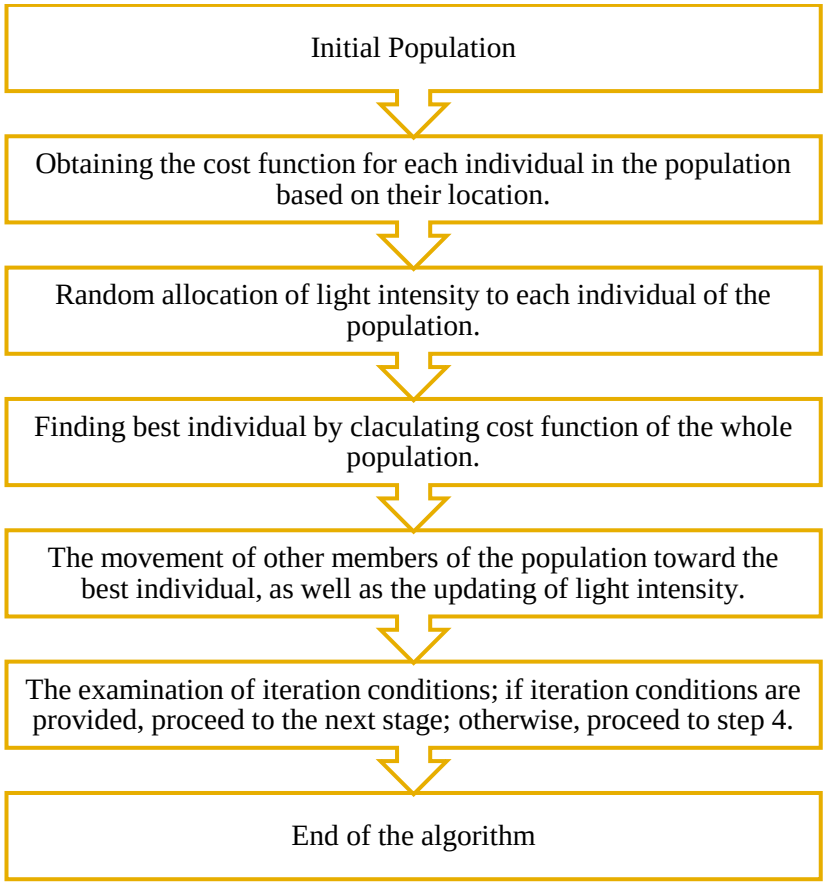

Figure 3.2: Flowchart of Firefly Algorithm

3.3.2 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 [75, 76] 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. 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 3.3 depicts a typical ANFIS controller's block diagram.

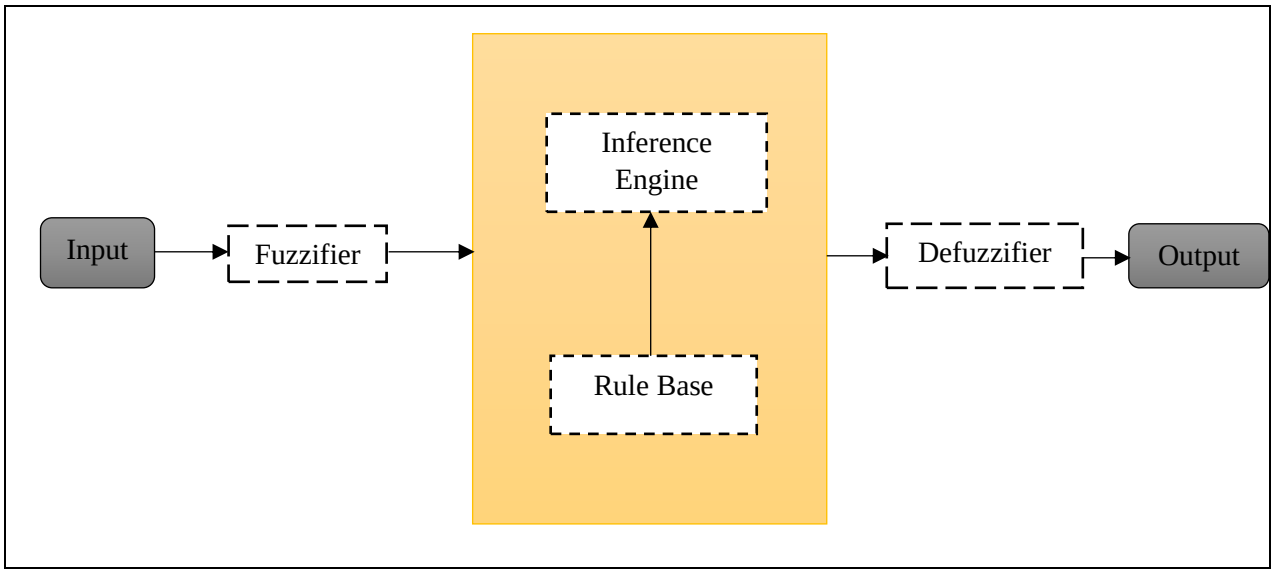

Figure 3.3: ANFIS Block Diagram

3.4 Performance Metrics

In order to evaluate the performance of methods and determine the importance of parameters, available experimental data were analyzed and compared with theoretical predictions. A comparison of the real values with the predicted values was made using r^2 and RMSE, which were used to compare the real and predicted values.

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

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(Ai-Pi)^2}{n}} \qquad \text{(Eq. 3.1)}$$

- *Coefficient of Determination (r^2):* The coefficient of determination is used to examine how differences in one variable can be explained by differences in another.

$$r^2 = \frac{\left[\sum_{i=1}^{n}(Ai-Ai')(Pi-Pi')\right]^2}{\sum_{i=1}^{n}(Ai-Ai')\sum_{i=1}^{n}(Pi-Pi')} \qquad \text{(Eq. 3.2)}$$

where n is the total number of test data

A_i is the ANFIS value

P_i is the measurement value

Results and Discussion

The results are discussed and the findings of the experiments carried out in this section are reported in visual form. Tables 4.1 and 4.2 present the summary of comparison between training ANFIS and testing ANFIS. The performance analysis of mobile malware using ANFIS-FA is graphically represented in figure 4.1 and figure 4.2.

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

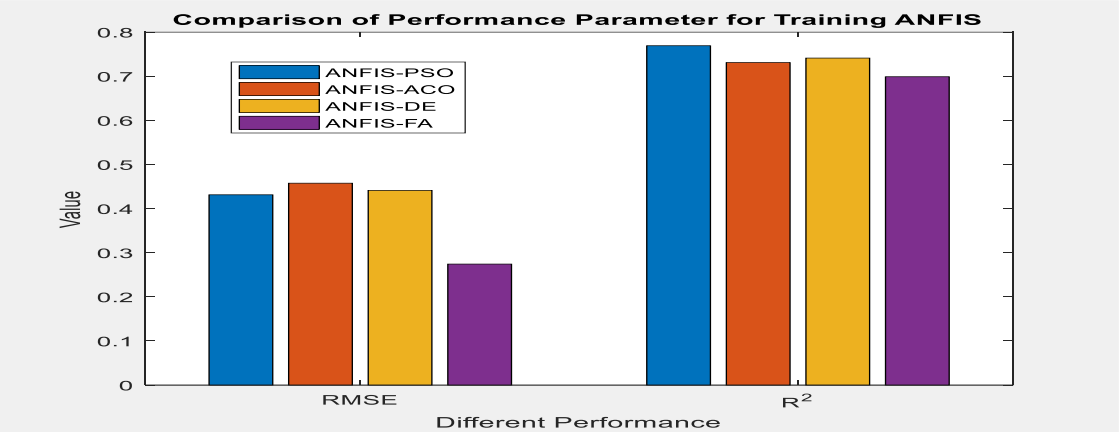

Figure 4.1: Training ANFIS

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

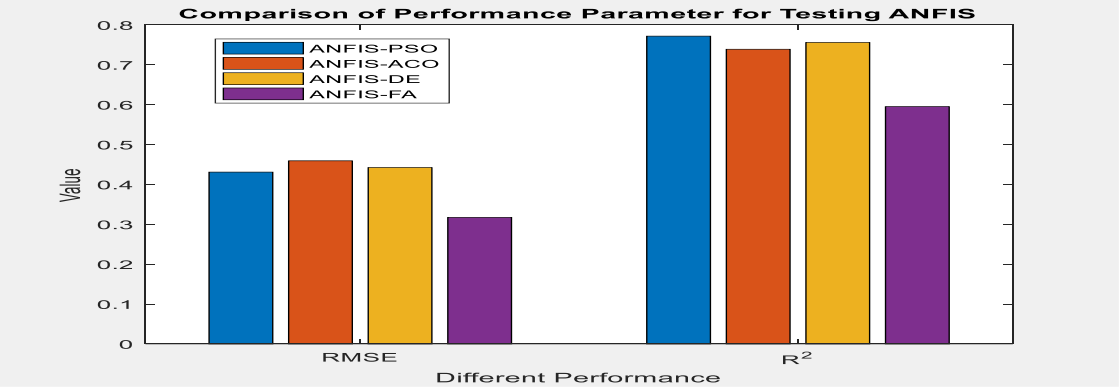

Figure 4.2: Testing ANFIS

Conclusion

A malicious software is any type of software or code which hooks private information, data from the computer, computer operations, or (and) simply just to do malicious goals of the author on the computer system, without permission from the computer users. Security attacks and services are described, along with meta-heuristic techniques used by security researchers to combat security attacks, in this work.

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 (r^2) improves as well, reaching 0.69892 in training and 0.59524 in testing.

References

[1] Aron, L., & Hanacek, P., "Overview of security on mobile devices. *2015 2nd World Symposium on Web Applications and Networking (WSWAN)*, pp. 1-11, 2015.

[2] La Polla, M., Martinelli, F., & Sgandurra, D., "A Survey on Security for Mobile Devices," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 446–471, 2013.

[3] Almuairfi, S., Veeraraghavan, P., & Chilamkurti, N., "A novel image-based implicit password authentication system (IPAS) for mobile and non-mobile devices," *Mathematical and Computer Modelling*, vol. 58, no. 1-2, pp. 108–116, 2013.

[4] Simmonds, A., Sandilands, P., & van Ekert, L., "An Ontology for Network Security Attacks," *Applied Computing*, pp. 317–323, 2004.

[5] Farina, P., Cambiaso, E., Papaleo, G., & Aiello, M., "Understanding DDoS Attacks from Mobile Devices," *2015 3rd International Conference on Future Internet of Things and Cloud*, 2015.

[6] Patel, N. A., "A Survey on Security Techniques used for Confidentiality in Cloud Computing," *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)*, 2018.

[7] Majumdar, S., Chakraborty, S., & Das, S., "A New Advanced User Authentication and Confidentiality Security Service," *International Journal of Computer Applications*, vol. 93, no. 11, May 2014.

[8] Kaushik, P., & Jain, A., "Malware Detection Techniques in Android," *International Journal of Computer Applications*, vol. 122, no. 17, pp. 22-26, July 2015.

[9] Qamar, A., Karim, A., & Chang, V., "Mobile malware attacks: Review, taxonomy & future directions," Future Generation Computer Systems, vol. 97, pp. 887-909, 2019.

[10] Felt, A.P., Finifter, M., Chin, E., Hanna, S., & Wagner, D., "A Survey of Mobile Malware in the Wild," SPSM, pp. 1-12, Oct. 2011.

[11] Tahir, R., "A Study on Malware and Malware Detection Techniques," I.J. Education and Management Engineering, vo. 2, pp. 20-30, Mar. 2018.

[12] Landage, J., & Wankhade, M. P., "Malware and Malware Detection Techniques: A Survey," International Journal of Engineering Research & Technology (IJERT), vol. 2, no. 12, Dec. 2013.

[13] Leach, A. R., "Ligand-Based Approaches: Core Molecular Modeling," *Comprehensive Medicinal Chemistry II*, pp. 87–118, 2007.

[14] Slowik, A., & Kwasnicka, H., "Evolutionary algorithms and their applications to engineering problems," *Neural Computing and Applications*, vol. 32, pp. 12363-12379, Mar. 2020.

[15] Sahay, S. K., Sharma, A., & Rathore, H., "Evolution of Malware and Its Detection Techniques," *Advances in Intelligent Systems and Computing*, pp. 139–150, 2019.

[16] Hsieh, W.-C., Wu, C.-C., & Kao, Y.-W., "A study of android malware detection technology evolution," *2015 International Carnahan Conference on Security Technology (ICCST)*, pp. 135-140, 2015.

[17] Katoch, S., Chauhan, S. S., & Kumar, V., "A review on genetic algorithm: past, present, and future," *Multimedia Tools and Applications*, 2020.

[18] McCall, J., "Genetic algorithms for modelling and optimization," *Journal of Computational and Applied Mathematics*, vol. 184, no. 1, pp. 205–222, 2005.

[19] Lambora, A., Gupta, K., & Chopra, K., "Genetic Algorithm- A Literature Review," *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, 2019.

[20] Guo, Z., Huang, H., Deng, C., Yue, X., & Wu, Z., "An Enhanced Differential Evolution with Elite Chaotic Local Search," *Computational Intelligence and Neuroscience*, 2015.

[21] Bilal, Pant, M., Zaheer, H., Hernandez, L. G., Abraham, A., "Differential Evolution: A review of more than two decades of research," *Engineering Applications of Artificial Intelligence*, vol. 90, 103479, 2020.

[22] Altenberg, L., "Evolutionary Computation. Encyclopedia of Evolutionary Biology," pp. 40–47, 2016.