International Journal of Computer Sciences and Engineering
Vol.**12**, Issue.**8**, pp.**18-24**, August **2024**
ISSN: 2347-2693 (Online)
Available online at: www.ijcseonline.org

ISSN: 2347-2693 (E)

Research Article

Development of a Model for the Detection of Malicious Activities on Edge Computing

Irhirhi M.[1*], V.T. Emmah[2]

[1,2]Computer Science/Science, River State University, Port Harcourt, Nigeria

**Corresponding Author: irhirhimoses@gmail.com*

Received: 18/Jun/2024; **Accepted**: 20/Jul/2024; **Published**: 31/Aug/2024. **DOI**: https://doi.org/10.26438/ijcse/v12i8.1824

Abstract: The unique characteristics of edge computing, such as limited resources and decentralized architecture, pose distinct challenges to traditional security measures. As the adoption of edge computing continues to proliferate across diverse domains, the security of edge devices becomes a paramount concern. This paper outlines a comprehensive approach for the detection of malicious activities (DDoS, Okiru and PartofHorizontalPortScan) on edge computing devices. The proposed solution leverages a combination of anomaly detection, Recurrent Neural Network (RNN) algorithm, and behaviour analysis tailored to the constraints of edge devices. By considering the specific context of edge environments, the model aims to distinguish between normal and malicious behaviour in edge computing, offering a proactive defence against emerging threats. Furthermore, the integration of threat intelligence feeds enhances the system's ability to adapt to evolving attack vectors. The efficiency of the proposed solution ensures minimal impact on the performance of resource-constrained edge devices. This paperwork contributes to the ongoing efforts to fortify the security of edge computing ecosystems. By addressing the unique challenges associated with these devices, the proposed RNN algorithm provides a robust and adaptive framework for the detection and mitigation of malicious activities in edge computing, safeguarding the integrity and reliability of edge computing applications with an accuracy of 99.9%.

Keywords: Edge Computing, Malicious packets, recurrent neural network, Random Forest Classifier

1. Introduction

A malware attacks poses a significant risk to both edge The use of technology and the protection of digital systems and data. In modern times, there is a type of malicious software that may avoid basic detection methods that rely on matching specific patterns. As a result, anti-malware solutions have adopted machine learning algorithms to detect and categorise previously unrecognised harmful files. Deploying an edge node offers a beneficial chance to provide malware detection services that rely on machine learning algorithms. When Internet of Things (IoT) devices lack sufficient computational power, edge nodes can act as secure gateways to protect them. Additionally, [1] has proposed that an edge node have the ability to select and provide appropriate malware detection services for Internet of Things (IoT) devices. Malware can be identified using many techniques, such as signature matching, static analysis, dynamic analysis, and the assistance of highly skilled human experts, although this can be rather expensive. An edge node can assist the provision of signature matching or static analysis services. Nevertheless, when confronted with exceedingly complex and evasive malware, it is prudent to depend on cloud-based tools for thorough analysis. Moreover, an edge node possesses the potential to provide uninterrupted malware analysis services, even in circumstances when the reliability of the cloud or internet connection is compromised. As stated in [2], the presence of privacy and data leakage problems can be minimised because most questionable files are dealt with by internal edge nodes instead of public clouds.

Edge computing is a paradigm that involves transferring some or all of the responsibilities of cloud computing to localised edge devices as needed. This strategy can improve performance in instances where the network architecture presents a limitation on the prompt delivery of services. Edge nodes, similar to danger monitors or sensors, are widely dispersed across the Internet. Although they may lack the same hardware and processing capabilities as data centres, they are nonetheless capable of effectively running complex parallel applications and providing fast service for intricate computations. The primary issue at hand is the ability to thoroughly scan and identify mobile malware, considering the increasing use of mobile smartphones, tablets, and Internet-of-Things (IoT) devices. The identification of malicious software on mobile devices has unique challenges mostly due to the limited capacity of mobile hardware. Moreover, the ability to efficiently handle large volumes of applications simultaneously to accommodate a wide user base poses a specific challenge for cloud infrastructures [3].

The significant proliferation of smartphone technology and its widespread utilisation have resulted in significant economic advantages on a global level. For example, in 2016, the worldwide mobile market earned an approximate annual revenue of $1.05 trillion. The Android operating system developed by Google has consistently had the most market share among all device operating systems. Android OS achieved a substantial market share of 88% in the third quarter of 2016. Moreover, starting from late 2013, Android OS has continuously represented at least 75% of all mobile sales. Conversely, Apple's iOS has faced difficulties in exceeding a market share of 25% over the same timeframe. Nevertheless, the prevalence of malicious software and weaknesses in mobile devices has risen in tandem with the expanding commercial dominance of Android. This can be partially due to the inherent openness of the Android platform, the continuous connectivity of mobile devices, and the overwhelming preference for Android over other platforms [4].

2. Related Work

[5] introduces a novel detection method for edge computing that utilises pre-existing machine learning models to categorise a suspicious file as either benign, malicious, or unpredictable. Unlike current models that simply make a binary decision of benign or harmful, this technique expands the classification options. By appending a basic sigmoid function to existing deep learning models, the new technique can utilise any currently available models for malware detection. During the testing phase, the new strategy assesses the confidence of the model's forecast by interpreting the sigmoid value. As a result, the new method may choose only highly accurate predictions, reducing false predictions related to confusing static-analysis characteristics. By doing research on actual malware datasets, the authors have verified that the new technique significantly enhances the accuracy, precision, and recall of current deep learning models. The accuracy has been increased from 0.96 to 0.99. However, there are some data points that are considered unpredictable and may require further analysis by either dynamic algorithms or human experts. These data points can be stored in the cloud for inspection.

[6] introduces an edge computing-powered method for detecting malware. This system effectively identifies different cyberattacks (malware) by transmitting large volumes of intelligent industrial IIoT traffic data to edge servers for in-depth analysis using deep learning techniques. The proposed malware detection system comprises three tiers (edge device, edge, and cloud layers) and utilises four significant functions (model training and testing, model deployment, model inference, and training data transmission) for deep learning at the edge. The Malign dataset was used to conduct tests on a malware detection system. This system combined a convolutional neural network with image visualisation technology. The results showed an impressive overall classification accuracy of 98.93%, precision of 98.93%, recall of 98.93%, and F1-score of 98.92%. [7] Develop a privacy-preserving federated learning system using support vector machine (SVM) and secure multi-party computation approaches. Furthermore, it demonstrates the practicality of the Android malware dataset provided by the National Institute of Information and Communication Technology (NICT) in Japan. The provided experiments assess the efficacy of the trained classifier using the recommended PPFL approach. The evaluation also assesses the performance of the PPFL classifier in comparison to the centralised training system. This assessment is conducted for numerous use scenarios, including multiple data sets and different features on different mobile devices. The results demonstrate that the performance of the PPFL classifier surpasses that of the centralised training system. Furthermore, the confidentiality of app information, such as API and authorisation information, as well as trained local models, is guaranteed. This study represents the pioneering effort in developing an Android malware detection system that utilises a privacy-preserving federated learning approach. [8] Present a deep learning model that supports multi-input of multi-modal data, enabling the simultaneous integration of digital features and visual information across several dimensions. The model contains the concurrent training of three sub-models, as well as the ensemble training of a separate sub-model. The four sub-models possess the potential to engage in parallel processing on separate devices and can be efficiently utilised in edge computing scenarios. The model possesses the ability to actively obtain multi-modal characteristics and produce predictive results. The model demonstrates a detection rate of 97.01% and a false alarm rate of only 0.63%. The experimental results offer proof that the suggested technique is superior and effective. The study undertaken by [9] focusses on data centres (DCs) and supercomputers (SCs). These facilities have recently installed state-of-the-art monitoring systems that provide improved resolution. This advancement has generated new opportunities for doing studies like as anomaly detection and security measures. Nevertheless, it has also brought out new difficulties in terms of handling the significant amount of data produced by these systems. This study offers an extensive examination of a new approach designed to improve the security of data centres and smart cities. The proposed strategy entails the application of artificial intelligence-driven edge computing technologies, with a specific emphasis on high-resolution power utilisation statistics. The pAElla technique is specifically developed to tackle the challenge of detecting malware in real-time within a monitoring system for data centres and smart cities. This system relies on an out-of-band Internet of Things framework. This approach involves analysing power measurements using power spectral density, while also employing autoencoders. The results exhibit great potential, as indicated by an F1-score approaching 1, and minimal occurrences of false alarms and missed malware. The authors do a comparison analysis of their suggested method, pAElla, and the State-of-the-Art (SoA) Molecular Dynamics (MD) techniques. They show that pAElla has a wider ability to detect different types of malware within the

domain of DCs/SCs. Moreover, pAElla significantly outperforms existing methods in terms of accuracy. In their investigation, [10] utilised a traditional edge node to do Android virus detection. The authors' work entails gathering a significant number of mobile malware and benign samples. The main goal of the study is to demonstrate the effectiveness of edge computing nodes in providing conventional security services for edge devices. Moreover, they have the ability to scale up the operation in order to accommodate increasing volumes of data. The study undertaken by [11] addresses the lack of scientific literature on mobile malware. The authors particularly highlight a wider array of permissions that may be employed for different purposes, such as acquiring user credentials via sensors or monitoring a user's activity. The objective of this study is to use behavioural analysis to determine the utilisation of permissions and employ static and dynamic analysis to comprehend the behaviour of application logic that has not been implemented yet, in order to discover specific scenarios. In addition, the authors proposed a detection engine with two layers that utilises hybrid feature analysis. The empirical data derived from doing experiments with genuine mobile malware. The IoT data clearly shows that our proposed methodology, which includes permission-related characteristics, outperforms other detection engines. [12] utilised MATLAB R2021a to perform experimental simulations in their work. The purpose of these simulations was to verify the precision of estimating the ideal chance of malware transmission and to evaluate the efficiency of IoT systems helped by edge computing in different topologies. The researchers in this study specifically analysed the curvature patterns to create forecasts on the dissemination of malware in the Internet of Things (IoT). They accomplished this by altering various aspects of the IoT system, such as the success rate of disseminating IoT malware, the effectiveness of intrusion detection systems as a service (IDSaaS) in identifying the malware, and the speed at which packets are received.

The framework "DNNdroid" was introduced by the authors [13]. It functions using the ideas of federated learning. The data regarding newly installed applications is retained alone on the user's device and is not revealed to the developer. During this phase, data from all users is collected simultaneously to enable the training of the model using a federated learning technique, leading to the creation of an enhanced categorisation model. The main challenge in this situation is to the user's difficulty in detecting the existence of malware within an application. The testing results show that the cloud server earned an F1 score of 97.8%, indicating a recall rate for clients that is higher than 0.95 and a false positive rate. This evaluation was performed using a dataset comprising 100,000 unique Android applications, each having a user base of at least 500 users. The process of federation was iterated 50 times in all. [14] conducted a study where they developed a new method called CloudSEC. This method uses an evidential reasoning network to detect the transfer of data within the edge-cloud context in real-time. Firstly, the concept of vulnerability correlation is introduced. An evidence reasoning network is constructed by utilising knowledge about the network system's vulnerability and environmental information. This network is used to enhance the ability to reason and move laterally. The testing results demonstrate that CloudSEC provides a strong guarantee for quick and efficient analysis of data, together with immediate detection of attacks.

3. Methodology

A. **Edge Computing Server:** The edge server is a computer, either real or virtual, that is located at the outermost part of a network, nearer the data source or the end users. Its goal is to do computations locally, eliminating the requirement for communication with a remote cloud server. The result is decreased delay and bandwidth use.

B. **Malware Data:** Information about malware is commonly referred to as "malware data." Malware characteristics such as signatures, behaviors, and file formats may be recorded. It's what the edge computing system reads and uses to determine whether or not malware is there.

C. **Data Preprocessing:** In order to prepare data for analysis, preprocessing steps include cleansing, transforming, and organizing. Tasks including deleting duplicates, dealing with missing values, normalizing data, and putting data into a suitable format for the next steps may fall under this category in the context of malware detection.

D. **Feature Selection with Random Forest:** An essential part of machine learning, feature selection involves picking out useful characteristics (features) from the dataset. When it comes to feature selection, Random Forest is one of the most widely utilized ensemble learning methods. To do this, numerous decision trees are built and their results are combined. Those features that hold true across all of these trees are the most relevant ones.

E. **Model Output (Benign and Malware):** The model then generates predictions based on the processed data and selected features. In this scenario, it divides the information into "Benign" (safe) and "Malware" categories (malicious). For a given input sample, the output shows the categorization outcome.

Algorithm 3.1: Detection of Malware on Edge Devices

```
1. Input: MalwareFeatures
2: Output: Malware, Benign
3: model = outlier(MalwareFeatures)
4: initialized threshold = 0.5
4: for i in MalwareFeatures do
5:    pred =  model.detect(i)
6:    loss = MeanSquareError(MalwareFeatures -pred)
7:    threshold= Mean(pred)
8:    if (loss \geq threshold):
9             return Malware found
```

```
10.     elif:
11.         return normal MalwareFeatures found
12.     endif
13. endFor
```

4. Results

The experiment was conducted on google colab and flask framework. The experimental phase has to do with the analysis phase, the implementation of the Deep Learning (DL) model, and the deployment of the model to edge computing environment for the detection and prevention of malware attacks on edge computing.

A. Data Analysis

For performing analysis, pandas, seaborn, and matplotlib library was used in conducting analysis on the dataset. The analysis was conducted so that a proper insight on the dataset before training the RNN model can be seen. The analysis phases are checking if they exist some null and duplicate values in the dataset. Pandas data was used in achieving this. Secondly, a bar chart was plotted to check if the number of classes (different types of the malware attacks on edge devices) have the same number of instances. The bar chat in Figure 2 shows the number of instances of each of the different types of malware attacks. From the bar chart, it is seen clearly that the number of instances of the different classes of the malware attacks are not the same. That simply makes the dataset imbalance, which means that if the data imbalance is not resolved, the RNN classifier will produce high rate of false positives and false negatives. To solve the data imbalance problem, random over sampling needs to be performed. This was achieved using an over-sampling technique called RandomOverSampler. This was used to perform undersampling on the dataset, making all the classes have equal number of instances. The down sampled data can be seen in the bar chart in Figure 3.

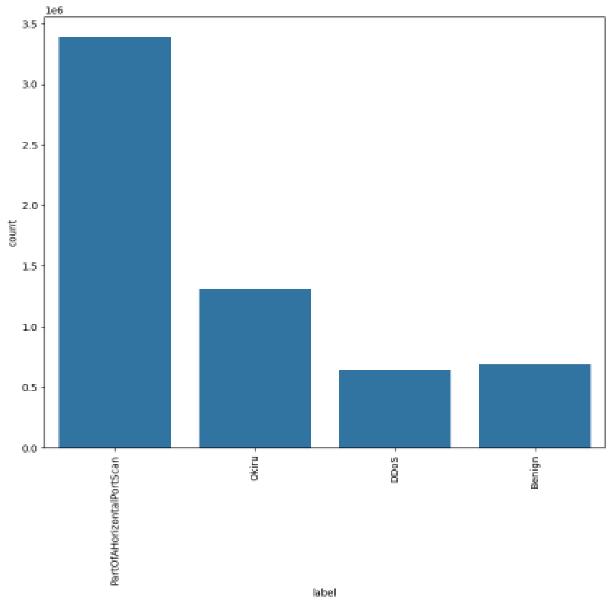


Figure 2: Countplot of the Imbalance Class

The countplot shows the distribution of the different types of malicious attacks on edge devices. With portofhorizontalscan having highest number of occurrences.

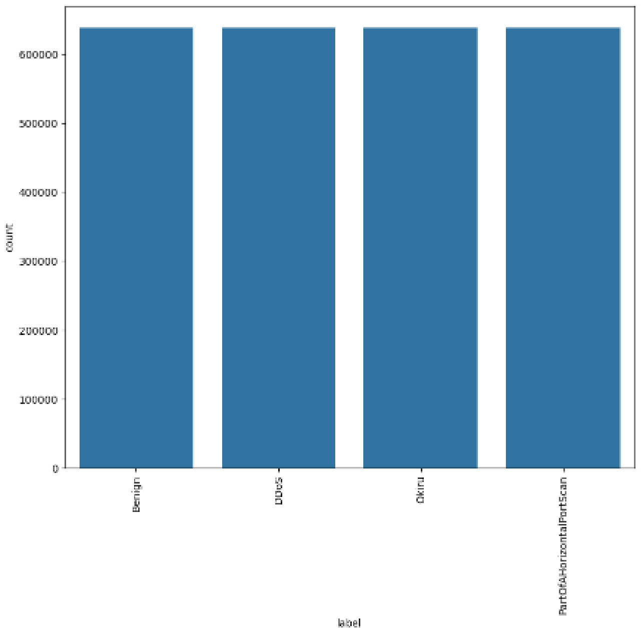
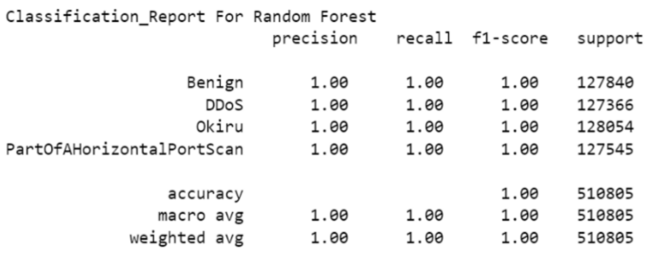


Figure 3: Countplot of the balance Class

The countplot shows that the classes of the malware attacks on edge devices have same occurrences. This shows that the dataset imbalance has been solved using RandomOverSampling technique.

B. Model Parameter Tuning and Training

This section describes the parameters and the processes used in training the Recurrent Neural Network (RNN) model for the detection and prevention of malicious activities on edge devices. The RNN model was trained by fine tuning it's hyper parameters. The fine-tuned parameters of the RNN model has three layers, one input layer with input neuron of 256, a hidden layer with an input neural of 256, and finally the output layer with dense layer 5. The hyper parameters used here are relu and softmax for activation functions, optimizer = 'adam', and loss ='categorical_consentropy', batch_size=64, and epoch =7. The result of the RNN model for both training and evaluation can be seen in Table 1. The evaluation of the RNN model was validated on a test data. The evaluation matrix used are classification report and confusion matrix can be seen in Figure 5 and 6. The graphical analysis of Table 1 can be seen in Figure 7, and Figure 8.



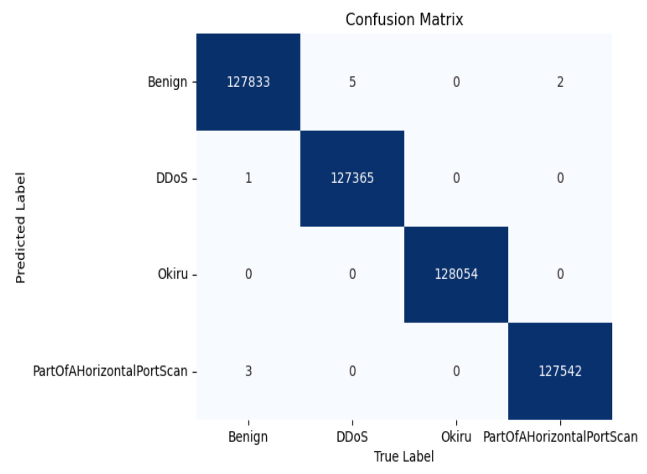
```
Classification_Report For Random Forest
                           precision    recall  f1-score   support

                   Benign       1.00      1.00      1.00    127840
                     DDoS       1.00      1.00      1.00    127366
                    Okiru       1.00      1.00      1.00    128054
PartOfAHorizontalPortScan       1.00      1.00      1.00    127545

                 accuracy                           1.00    510805
                macro avg       1.00      1.00      1.00    510805
             weighted avg       1.00      1.00      1.00    510805
```

Figure 5: Classification Report of the Model on test data

The classification report shows the performance of the model on test data, with the model achieving accuracy of 99.99%.



```
True Positive Rate: 0.9999921486110893
False Positive Rate: 3.9112001126425634e-05
```

Figure 6: Confusion Matrix

The confusion matrix shows the total number of correct and incorrect detection made by the model in detecting malicious activities on edge devices.

Table 1: Simulation of the Model on 10 Steps

```
Epoch 1/10
63851/63851 [==============================] –
329s 5ms/step – loss: 0.0124 – accuracy: 0.9963 – val_loss:
0.0011 – val_accuracy: 0.9999
Epoch 2/10
63851/63851 [==============================] –
367s 6ms/step – loss: 8.1246e-04 – accuracy: 0.9999 –
val_loss: 6.7327e-04 – val_accuracy: 0.9999
Epoch 3/10
63851/63851 [==============================] –
409s 6ms/step – loss: 0.0012 – accuracy: 0.9999 – val_loss:
9.5968e-04 – val_accuracy: 0.9999
Epoch 4/10
63851/63851 [==============================] –
383s 6ms/step – loss: 8.6852e-04 – accuracy: 0.9999 –
val_loss: 8.3973e-04 – val_accuracy: 0.9999
Epoch 5/10
63851/63851 [==============================] –
518s 8ms/step – loss: 0.0015 – accuracy: 0.9999 – val_loss:
5.7470e-04 – val_accuracy: 1.0000
Epoch 6/10
63851/63851 [==============================] –
513s 8ms/step – loss: 0.0012 – accuracy: 0.9999 – val_loss:
7.7087e-04 – val_accuracy: 0.9999
Epoch 7/10
63851/63851 [==============================] –
504s 8ms/step – loss: 9.4816e-04 – accuracy: 1.0000 –
val_loss: 7.3796e-04 – val_accuracy: 0.9999
Epoch 8/10
63851/63851 [==============================] –
489s 8ms/step – loss: 0.0011 – accuracy: 1.0000 – val_loss:
5.6745e-04 – val_accuracy: 1.0000
Epoch 9/10
63851/63851 [==============================] –
636s 10ms/step – loss: 8.5152e-04 – accuracy: 1.0000 –
val_loss: 0.0036 – val_accuracy: 1.0000
Epoch 10/10
 63851/63851 [==============================] –
 395s 6ms/step – loss: 0.0020 – accuracy: 1.0000 – val_loss:
           4.4836e-04 – val_accuracy: 1.0000
```

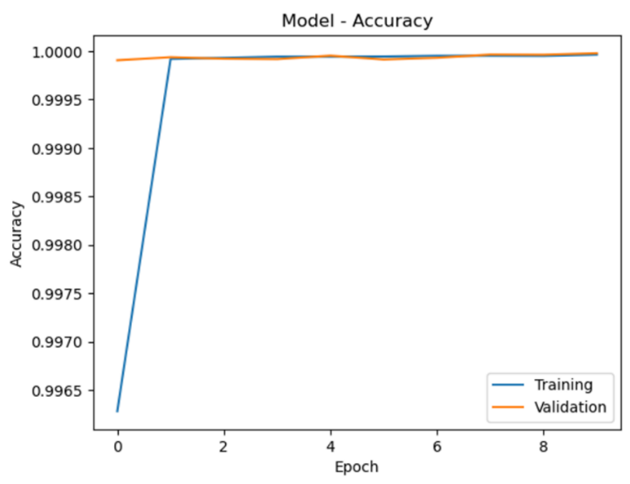


Figure 7: Training Accuracy For Both Training and Validation.

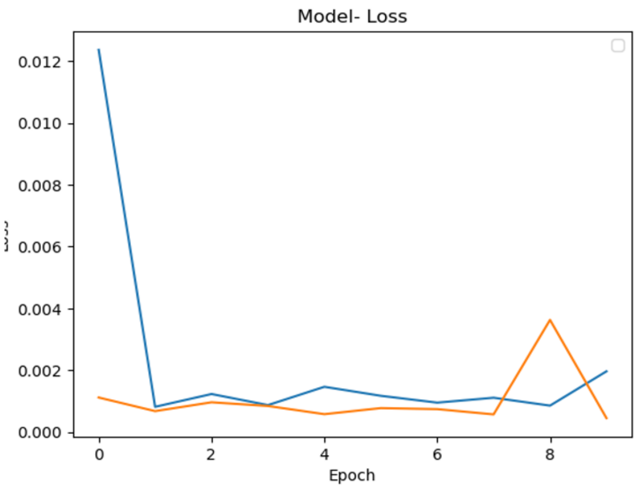


Figure 8: Loss values for training and Validation.

C. Deployment to Web

The model was deployed to web for further detection of malicious activities on edge computing. The deployed web application can be seen in Figure 9,10, and 11.

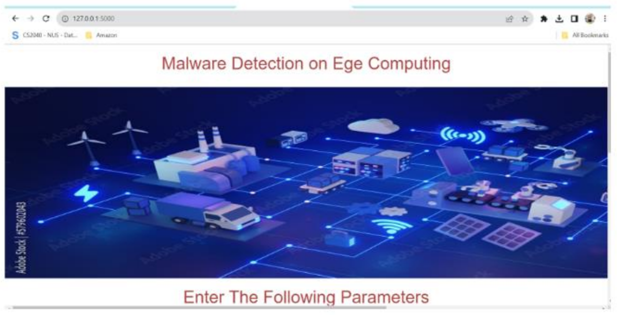
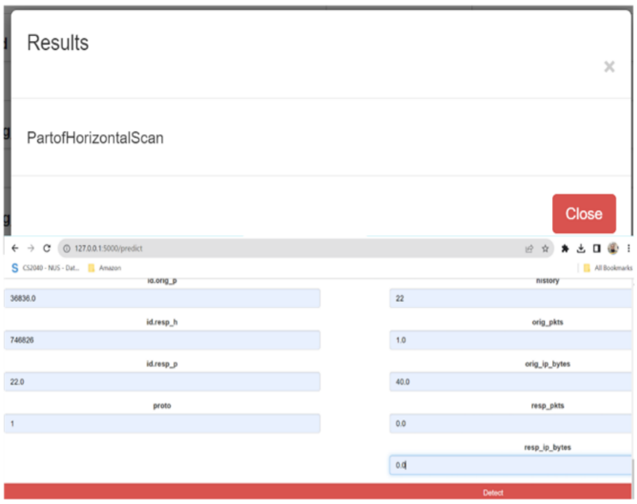

Malware Detection on Ege Computing
Enter The Following Parameters

Figure 9: Homepage


Results

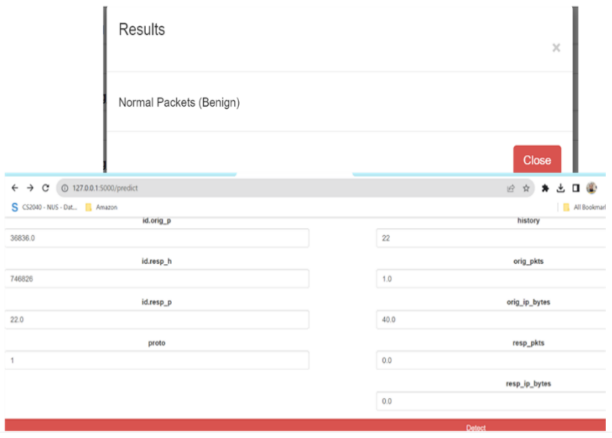
PartofHorizontalScan

127.0.0.1:5000/predict

id.orig_p: 36836.0
history: 22
id.resp_h: 746826
orig_pkts: 1.0
id.resp_p: 22.0
orig_ip_bytes: 40.0
proto: 1
resp_pkts: 0.0
resp_ip_bytes: 0.0

Figure 10: Malware detected.

The result from Figure 10 shows that that network packets that comes into the system are of malicious packets. The type of malware that was detected is partofhorizontalportscan


Results

Normal Packets (Benign)

127.0.0.1:5000/predict

id.orig_p: 36836.0
history: 22
id.resp_h: 746826
orig_pkts: 1.0
id.resp_p: 22.0
orig_ip_bytes: 40.0
proto: 1
resp_pkts: 0.0
resp_ip_bytes: 0.0

Figure 11: Normal Packets Detected.

The result from Figure 11 shows that that network packets that comes into the system are of good packets that cannot harm the system.

5. Discussion of Results

Figure 5 shows the classification report. The classification report provides a comprehensive summary of the model's performance by presenting metrics such as precision, recall, and F1-score for each class. Precision represents the accuracy of positive predictions, recall measures the model's ability to capture all positive instances, and the F1-score is the harmonic mean of precision and recall. A classification report is particularly useful when dealing with imbalanced datasets or when different classes have varying degrees of importance. An accuracy of 99.98% suggests that the model is highly accurate in its predictions, but the classification report can offer insights into potential issues, such as class-specific misclassifications. Figure 6 shows the confusion matrix. The confusion matrix is a tabular representation of a classification model's performance, breaking down the predicted and actual class labels. It consists of four main components: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). A confusion matrix allows for a detailed examination of where the model is making errors. In a scenario with an accuracy of 99.98%, a confusion matrix would show a high count of true positives and true negatives and minimal false positives and false negatives. However, it is important to carefully scrutinize these values, especially in the context of the specific problem at hand, as a high accuracy can sometimes mask issues related to class imbalances or misclassifications in critical classes. Figure 7 shows the accuracy for both training and validation. The results indicates that the model's performance has been evaluated on both training and validation datasets, revealing a high level of accuracy. Specifically, the model achieved an accuracy rate of approximately 99.98% on both sets. This suggests that the model has demonstrated robust learning and generalization capabilities, performing exceptionally well not only on the data it was trained on but also on new, unseen data in the validation set. The high accuracy implies a strong ability to correctly classify and predict outcomes, making the model reliable for the given task. Figure 8 shows the loss of the model. The result indicates the model's loss values during both the training and validation phases. Loss is a metric that quantifies the dissimilarity between the predicted outputs of the model and the actual target values. In this context, the training loss of 0.002 suggests that, on average, the model's predictions deviate by a small amount from the true values in the training dataset. Similarly, the validation loss of 0.001 indicates a comparable level of accuracy on a separate dataset not used for training, reflecting the model's generalization performance. Lower loss values typically signify better model performance, with minimized discrepancies between predicted and actual outcomes.

The result from Figure 10 shows that that network packets that comes into the system are of malicious packets. The type of malware that was detected is partofhorizontalportscan. The result from Figure 11 shows that that network packets that comes into the system are of good packets that cannot harm the system.

6. Conclusion and Future Scope

The combination of edge computing and a malware detection system utilising a Recurrent Neural Network (RNN) trained with features taken from a Random Forest model has shown impressive efficacy. This approach represents a substantial advancement in safeguarding edge computing settings, with an accuracy of 99.9% and a false positive rate of 0.00%. The model's strong ability to detect and reduce malware risks at the edge highlights the possibility of merging machine learning approaches to improve cybersecurity. The exceptional level of accuracy enhances the security framework of edge computing systems and also lays the foundation for a more secure and robust future in the constantly changing field of cybersecurity.

Subsequent research should prioritise the regular update of the malware detection model using fresh data in order to stay abreast of the most recent threats. This entails deploying scalable solutions to manage the rising data volumes that accompany the expansion of edge computing networks. In addition, incorporating this approach into current cybersecurity frameworks will offer a more thorough security solution. Further research should investigate methods to enhance the performance of the RNN and Random Forest algorithms with the goal of enhancing detection speed and efficiency. In addition, integrating threat intelligence feeds and promoting collaborative security initiatives can improve the system's capabilities. Consistent security audits, strict compliance with data protection requirements, and continuous research and development are essential for maintaining a strong defence against new cyber threats.

Conflict of Interest
The authors declare that they do not have any conflict of interest.

Funding Source
None

Authors' Contributions
Author-1 conceived the study. Author-2 developed the protocol and analyzed the data. Author-3 wrote the first draft. All authors reviewed and approved the final manuscript.

Acknowledgements
We acknowledge the support of our colleagues and reviewers.

References

[1] J. Abawajy, S. Huda, S. Sharmeen, M. M. Hassan, and A. Almogren, "Identifying cyber threats to mobile-IoT applications in edge computing paradigm," *Future Generation Computer Systems*, Vol.**89**, pp.**525-538, 2018.**

[2] A. Anand, S. Patil, and P. Kulkarni, "A survey on edge computing security: Threats, attacks, and defenses," *Journal of Ambient Intelligence and Humanized Computing*, Vol.**12**, No.**5**, pp.**4871-4891, 2021.** https://doi.org/10.1007/s12652-021-03246-6

[3] W. G. Hatcher, J. Booz, J. McGiff, C. Lu, and W. Yu, "Edge computing based machine learning mobile malware detection," **2017.**

[4] R. H. Hsu et al., "A privacy-preserving federated learning system for android malware detection based on edge computing," in *2020 15th Asia Joint Conference on Information Security (AsiaJCIS)*, pp.**128-136, 2020.**

[5] H. M. Kim and K. H. Lee, "IIoT malware detection using edge computing and deep learning for cybersecurity in smart factories," *Applied Sciences*, Vol.**12**, No.**15**, pp.**7679, 2022.**

[6] K. J. Kim and J. H. Kim, "A survey of security threats and countermeasures in edge computing," *Journal of Supercomputing*, Vol.**76**, No.**8**, pp.**5834-5864, 2020.** https://doi.org/10.1007/s11227-020-03154-5

[7] Y. J. Kim, C. H. Park, and M. Yoon, "FILM: filtering and machine learning for malware detection in edge computing," *Sensors*, Vol.**22**, No.**6**, pp.**2150, 2022.**

[8] W. Lian, G. Nie, Y. Kang, B. Jia, and Y. Zhang, "Cryptomining malware detection based on edge computing-oriented multi-modal features deep learning," *China Communications*, Vol.**19**, No.**2**, pp.**174-185, 2022.**

[9] A. Libri, A. Bartolini, and L. Benini, "pAElla: Edge AI-based real-time malware detection in data centers," *IEEE Internet of Things Journal*, Vol.**7**, No.**10**, pp.**9589-9599, 2020.**

[10] A. Mahindru and H. Arora, "Dnndroid: Android malware detection framework based on federated learning and edge computing," in *International Conference on Advancements in Smart Computing and Information Security*, Cham: Springer Nature Switzerland, pp.**96-107, 2022.**

[11] Y. Shen, S. Shen, Z. Wu, H. Zhou, and S. Yu, "Signaling game-based availability assessment for edge computing-assisted IoT systems with malware dissemination," *Journal of Information Security and Applications*, Vol.**66**, pp.**103140, 2022.**

[12] Z. Tian et al., "Real-time lateral movement detection based on evidence reasoning network for edge computing environment," *IEEE Transactions on Industrial Informatics*, Vol.**15**, No.**7**, pp.**4285-4294, 2019.**

[13] Y. Wang, L. Zhang, C. Xie, and Q. Wu, "A survey of security and privacy issues in edge computing," *IEEE Network*, Vol.**34**, No.**6**, pp.**164-172, 2020.** https://doi.org/10.1109/MNET.011.2000184

[14] Y. Yang, Y. Liu, C. Wang, and X. Tang, "A dynamic malicious activity detection scheme for edge computing based on deep learning," *IEEE Access*, Vol.**8**, pp.**220217-220231, 2020.** https://doi.org/10.1109/ACCESS.2020.3048144

AUTHORS PROFILE

Moses Irhirhi earned his B.Sc. from Delta State University, Abraka, M.Sc from Rivers State University, Port Harcourt, in Computer Science in 2014 and 2024 respectively. He has a certificate in HSE Supervision and has been in the educational field for about 8 years.

V.T Emmah is a lecturer in the department of computer science, Rivers State University. He has a P.HD in computer Scince.. He has 15 years of teaching experience and 10 years of research experience.