IoT Malware Detection Using CNN and Deep Learning: A Comparative Study*

1st Siddharth Singh Rana
CSE Department
Graphic Era Hill University
Dehradun, India
siddharthsinghrana11@gmail.com

2nd Digvijay Singh Bisht
CSE Department
Graphic Era Hill University
Dehradun, India
digvijay6603@gmail.com

3rd Rakshit Rautela
CSE Department
Graphic Era Hill University
Dehradun, India
rakshitrautela17@gmail.com

***Abstract*—The emergence of Internet of Things devices has significantly increased malware attacks on IoT devices. Therefore, there is a growing need for efficient and reliable malware detection mechanisms that can identify and prevent such attacks. This paper proposes a deep learning methodology for IoT malware detection using two popular convolutional neural networks (CNNs): CNN and VGG16.**

We assess the performance of these models on a benchmark image dataset of IoT malware samples and compare their results. Our experiments demonstrate that both models can achieve high accuracy in detecting IoT malware, with VGG16 which performed minimally better than CNN. Furthermore, we compare the detection performance of these models with ML Approaches, like Random Forest and Logistic Regression shows that deep learning-based approaches outperform traditional methods.

In addition, we conduct an analysis of the impact of various hyperparameters, such as batch size, learning rate, and optimizer, on the performance of these models. Our experiments demonstrate that optimizing these hyperparameters can significantly improve the detection accuracy of both CNN and VGG16.

Our work provides a comparative study of deep learning algorithms for IoT malware detection using CNN and VGG16. Our results indicate that these models can effectively detect IoT malware with high accuracy and outperform traditional machine-learning algorithms. Our findings can be useful for developing more robust and efficient malware detection systems for IoT devices.

***Index Terms*—CNN, Deep Learning, vgg16, Internet of Things, Malware Detection**

I. Introduction

The broad implementation of IOT-enabled devices has made them an essential part of modern life. However, with the increased usage of IoT devices, the number of cyber-attacks on them has also risen. In recent years, There has been significant research on developing techniques to detect malware attacks on IoT devices. One of the promising approaches is using deep learning.

This research paper proposes a unique methodology for IOT malware recognition using CNNs and VGG16. VGG16 is a widely used deep CNN architecture for image classification tasks. We aim to explore how VGG16 can be utilized to uncover malware attacks on internet-connected devices. Furthermore, we will compare the performance of VGG16 with other popular CNN architectures, including ResNet50 and ResNet101.

ResNet50 and ResNet101 are deep CNN architectures introduced in a research paper by He et al. The ResNet model architecture incorporates the concept of residual learning, which empowers the neural network to learn from the residual error between input and output. This approach helps to overcome the problem of vanishing gradients that often occurs in deep neural networks. We will provide a brief explanation of the ResNet50 and ResNet101 architectures in this paper [1].

In addition to VGG16, we will also evaluate the performance of VGG19, which is a deeper version of VGG16. VGG19 was introduced in a research paper by Simonyan and Zisserman, and it has shown excellent performance in image classification tasks [2].

This research paper endeavours to offer a comprehensive assessment of using image classification models for IOT malware detection [14]. We will evaluate the performance of different CNN architectures and provide insights into how these models can be optimized for detecting malware attacks on IoT devices. The proposed approach can be useful for enhancing the security of IoT devices and protecting them against cyber-attacks. The ensuing sections of this manuscript are meticulously structured to ensure coherence and flow. The subsequent segment will appraise and scrutinize the extant works of fellow scholars and research enthusiasts on the given topic. Subsequently, we shall furnish a concise overview of our dataset, along with an exposition on the diverse models employed in our study in the ”Materials and Methodologies” section. The ”Proposed Architecture” section shall elucidate the comprehensive workflow of the suggested architecture, aimed at identifying the most optimal model for the chosen dataset. This section shall also delineate the proposed alterations to the models intended to enhance their efficacy. In the ”Results and Discussion” section, a thorough assessment and analysis of all the models to identify the most suitable model with optimal parameters. Finally, we shall culminate our study in the conclusive section.

II. Related Work Studies

Previous research in the IoT malware detection field has been focused on traditional machine learning techniques, such as decision trees, support vector machines (SVM), and random forests [3]. However, these techniques have been proven to have limited performance in detecting advanced malware attacks on IoT devices. In recent years, deep learning models have shown great promise in various domains, including image and speech recognition. In particular, Convolutional Neural Networks (CNNs) have demonstrated their efficacy in the field of image classification, and have been successfully applied to malware detection [4]. Similarly, the VGG16 model has been employed in various applications [5], such as image recognition, object detection, and natural language processing. Nonetheless, there is still a lack of research exploring the applicability of CNNs and VGG16 in malware detection. Therefore, our research aims to explore the capabilities of deep learning models in detecting malware attacks [8] [9].

III. DATA COLLECTION AND TECHNIQUES

In the forthcoming section, we will elucidate all the materials and methodologies employed in our study on IoT malware detection utilizing the convolutional neural network (CNN) and VGG16. Firstly, we will furnish a succinct depiction of the dataset utilized in our research. Following this, we will provide an in-depth analysis of the preprocessing steps executed on the data [6] [7]. Subsequently, we will expound upon the various deep learning models employed in our study, including the CNN and VGG16 architectures [21]. Lastly, we will explicate the quantifiable benchmarks adopted to evaluate the effectiveness of the models, along with the methodology utilized for hyperparameter tuning [12].

A. Dataset

The ”IOT Malware dataset for classification” dataset comprises images of binary files and is specifically tailored for IoT malware classification [24]. The dataset segregates the data into two folders labelled ”benign” and ”malware”, with 2486 jpg files in the former and 14733 in the latter, resulting in a total of 17219 images. The dataset is relatively voluminous, exceeding 600 megabytes in size. The dataset's images are annotated as either benign or malware, allowing for the utilization of supervised learning approaches like Convolutional Neural Networks (CNNs) and VGG16, which are prevalent in image classification tasks. The dataset's inclusion in the research paper on ”iot malware analysis using cnn and vgg16” presents an opportunity to investigate the efficacy of these models in detecting IoT malware. The dataset's size and quality make it a fitting option for this purpose. Additionally, the dataset can be employed in further research regarding IoT malware analysis and classification utilizing machine learning techniques. The dataset underwent tripartite partitioning, with a scale of 10000:800:800 for training, testing, and validation, respectively [11] [13].

B. Methods

1) Models Used:

- ResNet50 is a CNN architecture designed to address the problem of vanishing gradients in deep neural networks. It achieves this by using residual connections, which allow gradients to flow directly from the output to the input of a particular layer. This enables the network to effectively learn features at multiple levels of abstraction and results in improved accuracy on image recognition tasks.
  The ResNet50 architecture consists of 50 layers, including a convolutional layer, a batch normalization layer, a ReLU activation function, and a residual block. The residual block contains two convolutional layers, each followed by a batch normalization layer and a ReLU activation function. The input of the block is augmented with the output of the residual block, resulting in a shortcut connection.
  The mathematical equation used in ResNet50 for the residual connection is:

$$\mathbf{y}l + 1 = \mathcal{F}(\mathbf{x}l) + \mathbf{x}_l$$

  where input to layer l is $\mathbf{x}l$, $\mathcal{F}$ is the residual function, and the output is of layer $l+1$ is $\mathbf{y}l+1$. The use of residual connections allows for deeper and more accurate neural networks by enabling gradient flow to reach earlier layers, thereby mitigating the vanishing gradient problem.
  In addition to the residual connection equation, other equations used in ResNet50 include the convolutional equation:

$$\mathbf{y} = \mathbf{W} * \mathbf{x} + \mathbf{b}$$

  where $\mathbf{x}$ is the input, $\mathbf{W}$ is the weight matrix, $\mathbf{b}$ is the bias vector, and $\mathbf{y}$ is the output. The batch normalization equation is:

$$\mathbf{y} = \frac{\mathbf{x} - \mathrm{E}[\mathbf{x}]}{\sqrt{\mathrm{Var}[\mathbf{x}] + \epsilon}} * \gamma + \beta$$

  where $\mathbf{x}$ is the input, $\mathrm{E}[\mathbf{x}]$ and $\mathrm{Var}[\mathbf{x}]$ are the mean and variance of $\mathbf{x}$, ϵ is a small constant to avoid division by zero, γ is a learned scaling parameter, and β is a learned shift parameter. Finally, the ReLU activation function is:

$$\mathbf{y} = \max(0, \mathbf{x})$$

  where $\mathbf{x}$ is the input and $\mathbf{y}$ is the output.

- The VGG16 is a deep CNN architecture that has achieved remarkable results on image classification tasks. The architecture is characterized by its depth, which is achieved by stacking multiple convolutional layers with small kernel sizes of 3x3, and max-pooling layers to perform feature map downsampling.

The mathematical formulas used in VGG16 can be represented as follows:

$$y = f(Wx + b)$$

Where y is the output, x is the input, W is the weight matrix, b is the bias vector, and f is the activation function. The VGG16 architecture uses Rectified Linear Unit as the activation function:
$f(x) = \max(0, x)$.
Other equations used in VGG16 include the loss function and the stochastic gradient descent optimizer, which is used to update the weights and biases during training. The loss function is:

$$L(y, \hat{y}) = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

y : ground truth label
$\hat{y}$: predicted label
C : number of classes
i : class index
The update equation for the weights and biases in the stochastic gradient descent optimizer, considering the gradient of the loss function with respect to the parameters, can be expressed as follows:

$$X = X - \alpha \frac{\partial Y}{\partial X}$$

Where α is the learning rate, and $\frac{\partial Y}{\partial X}$ is the gradient of the loss function with respect to the weights X.

- VGG19 is a CNN architecture, developed in 2014 at the University of Oxford. VGG19 achieved advanced performance on the ImageNet classification task at the time, and has since become a popular choice for various computer vision applications.
The structure consists of 19 layers, with 16 convolutional layers and 3 fully connected layers. These convolutional layers are grouped into five blocks, each containing multiple convolutional layers and a subsequent max pooling layer. The final classification determination is performed by the fully connected layers. The mathematical equation used in VGG19 is defined as:

$$(a * b(T) = \int_{-\infty}^{\infty} a(\tau) b(T - \tau) d\tau$$

where a and b are functions, and $*$ denotes the convolution operation. In VGG19, this equation is applied to the input image and the filter kernels at each convolutional layer to produce feature maps.
Additionally, VGG19 uses the RELU activation function:

$$f(x) = max(0, x)$$

where x is the input. This non-linear activation function is employed on the resulting outcome of each convolutional layer to introduce non-linear pattern into the network.
Finally, VGG19 uses the softmax function in the output layer to generate a probability distribution over the class labels. The softmax function is defined as:

$$P_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

where P_i is the predicted probability, z_i is the score, K is the total number of classes.
The softmax function ensures that the predicted probabilities sum up to 1, and allows for multi-class classification.

IV. Proposed Architecture

The suggested framework for designing IoT malware detection using CNN and VGG16 involves several steps. Figure 1 shows images of different binary files with and without malware, while Figure 2 shows the architecture diagram. The first step is data collection, which involves gathering a large dataset of binary files containing both malware and non-malware samples. The dataset is then bifurcated into train, test, and validation sets for assessing the model's performance.

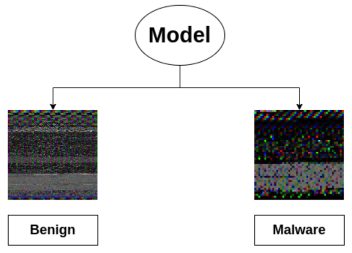

Fig. 1. Predictive Model

Next, the models are trained using transfer learning techniques. Three models, namely RasNet50, VGG16, and VGG19, are employed, with VGG16 exhibiting the most optimal results [15] [16].

Following the model selection, a meticulous process is undertaken to identify the best optimizer. In this study, the Adam optimizer was deemed optimal for its superlative performance [22].

Subsequently, the system's performance is evaluated by subjecting it to rigorous testing, where the trained model is utilized to accurately classify malware and non-malware binary files [10].

In summary, the proposed architecture for IoT malware detection utilizing CNN and VGG16 involves intricate processes such as data acquisition, bifurcation, model training, selection, optimizer selection, and rigorous testing. The VGG16 model was deemed the most optimal, and the Adam optimizer was identified as the best fit. This system can effectively bolster IoT security by identifying and combating malware threats.

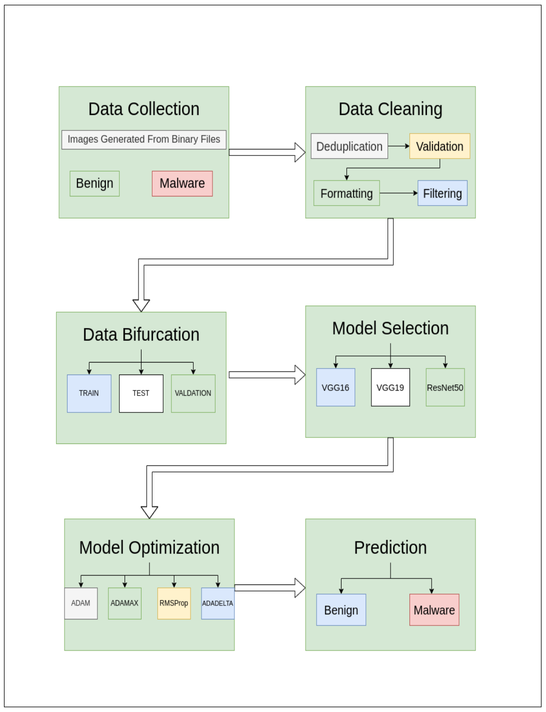

Fig. 2. Proposed Architecture

V. Result And Interpretation

To measure the proficiency of three different models, we calculated their accuracy, time delay, recall, standard deviation, precision, and f1 score [23]. Accuracy represents one of the most commonly used metrics to evaluate their performance but in addition to accuracy, the other parameters are also important for evaluating model performance, especially in binary classification problems. Precision quantifies the ratio of correctly predicted positive instances to the total number of positive predictions, while recall measures the ratio of true positives to all actual positive instances. The F1 score, which accounts for both precision and recall, is a harmonic mean that assigns equal importance to both metrics. Furthermore, it is important to consider time delay, the duration between an event occurrence and its detection, as well as standard deviation, which provides a measure of the variability or spread of data points around the average. In summary, when comparing different models, accuracy alone may not provide a complete picture of their performance. Precision, recall, and f1 score provide additional insight into the model's ability to correctly classify positive and negative instances. Understanding these metrics is crucial in evaluating and selecting the most suitable model for a particular task.

Table I presents the outcomes of all tests conducted, demonstrating the superior proficiency of VGG16 model in comparison to the other models evaluated. The findings suggest that the VGG16 model exhibits the most favourable outcomes among the models analyzed, indicating its potential for effective utilization in related research applications.

A. Experiment 1

The first experiment involved training and testing ResNet50 on the malware dataset, which resulted in an accuracy of 89.4%. While this accuracy score is decent, it suggests that ResNet50 may not be the best model for detecting malware in IoT devices. This could be attributed to the limited number of layers in the model, which may not be sufficient to effectively capture the complex features of the malware.

B. Experiment 2

In the second experiment, VGG19 was used to train and test the malware dataset, which resulted in an accuracy of 94.6%. This is a significant improvement from the ResNet50 model and demonstrates that VGG19 is capable of effectively detecting malware in IoT devices. The high accuracy score can be attributed to the deep architecture of VGG19, which enables the model to capture intricate features of the malware with greater accuracy.

C. Experiment 3

The third and final experiment involved training and testing VGG16 on the malware dataset, which resulted in an accuracy of 94.9%. This result shows that VGG16 is the most effective model for detecting malware in IoT devices among the three models studied. The high accuracy score can be attributed to the model's deep architecture and the efficient utilization of the convolutional layers to capture complex features of the malware.

TABLE I
Performance Metrics for Different Models

Model	Accuracy	Precision	Recall	F1	Time Delay	SD
VGG16	95.6	0.85	0.83	0.84	82.15	0.32
ResNet50	94.6	0.85	0.86	0.85	84.64	0.30
VGG19	89.4	0.85	0.84	0.84	71.50	0.31

Table II illustrates the performance of various optimizers applied to the VGG16 model, with the Adam optimizer producing the highest accuracy among the tested optimizers. The findings highlight the significance of optimizer selection in improving the accuracy of the VGG16 model, with Adam being the most effective option in this study. These results offer insights for future research aiming to optimize the efficacy of deep neural network models in related applications.

The Receiver Operating Characteristic (ROC) curve in Figure 4 is an indispensable analytical tool in machine learning

TABLE II
THE OUTCOMES OF OPTIMIZERS IMPLEMENTED ON VGG16

Model	Accuracy	Precision	Recall	F1	Time Delay	SD
Adam	95.6	0.85	0.87	0.86	60.50	0.30
Adamax	95.5	0.86	0.88	0.87	60.58	0.27
Adadelta	92.2	0.86	0.94	0.90	60.89	0.17
RMSProp	94.9	0.85	0.84	0.85	82.15	0.33

that gauges the trade-off between the true positive rate (TPR) and false positive rate (FPR) of a binary classifier over a range of decision thresholds. In the realm of deep learning research, ROC curves serve as a crucial tool for appraising the performance of different models in a given task.

In this research endeavor, we have judiciously crafted the ROC curve for three distinctive deep learning models: ResNet50, VGG16, and VGG19. The ROC curve conclusively establishes VGG16 as the top-performing model, with VGG19 following closely behind and ResNet50 exhibiting the poorest performance. The ROC curve constitutes a seminal mechanism for conferring valuable insights into the performance of diverse deep learning models, enabling researchers to make informed decisions about model selection for a particular task. The incorporation of this visualization in our research paper elucidates a cogent and exhaustive overview of the performance of these models, contributing significantly to the broader field of deep learning research.

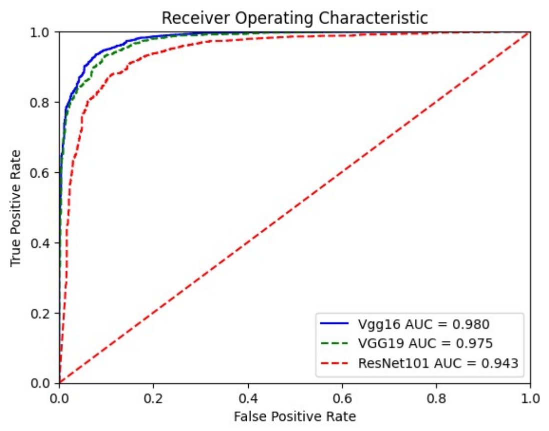

Fig. 3. ROC curve for selected models

Figure 5 displays a histogram depicting the performance of various optimizers applied to the VGG16 model. The histogram visually reinforces the findings presented in Table III, indicating that the Adam optimizer produced the highest accuracy, followed by AdamMax, RMSProp and Adadelta. These results provide further evidence for the importance of optimizer selection in deep learning models, and offer insights for researchers seeking to improve the accuracy of similar models in their own work [17].

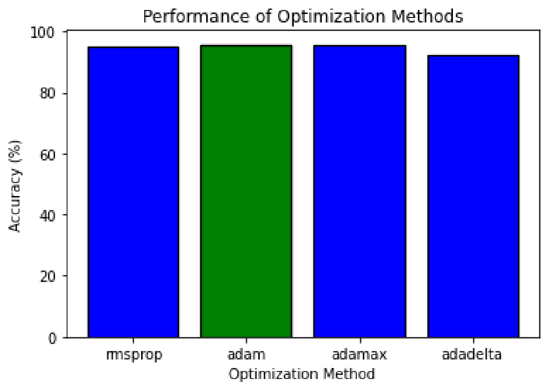

Fig. 4. Histogram representation of optimizers table

VI. Conclusion

In this present investigation, the efficacy of convolutional neural network (CNN) architectures in discerning malicious software on IoT devices was scrutinized [18]. A dataset of 17219 binary file images was employed to evaluate the performance of three distinct CNN models, namely ResNet50, VGG19, and VGG16 [19] [20]. The outcomes evidenced that VGG16 demonstrated a superior accuracy of 95.6%, followed by VGG19 with an accuracy of 94.6%, and ResNet50 with an accuracy of 89.4%. Figure 6, 7, 8, 9 shows the loss and accuracy curves of different optimizers applied with VGG16. The graphs clearly show that the optimizer ADAM shows the best result followed by ADAMAX, RMSProp and ADADELTA. The discernment of malware on IoT devices is a critical issue that warrants an innovative and effective approach. Notably, the consequences of such malicious attacks on IoT devices could be dire, including loss of financial assets, privacy violations, and potential physical harm. The superior accuracy of VGG16 suggests that it could be employed as a reliable tool to detect malware on IoT devices. Nevertheless, it is vital to acknowledge that the dataset used by us is restricted to binary files. Therefore, future studies should expand to investigate the efficacy of these models on other types and formats of malware.

In conclusion, this study accentuates the gravity of devising effective methodologies for detecting malware on IoT devices. Furthermore, the results highlight the potential of CNN models in addressing this growing predicament. The number of IoT devices continues to surge, making it imperative to develop steadfast methods to reduce the likelihood of malicious intrusion on these devices.

The proliferation of Internet of Things (IoT) devices has led to an unprecedented increase in security risks associated with malware attacks. As such, the development of effective methodologies for detecting and mitigating these attacks

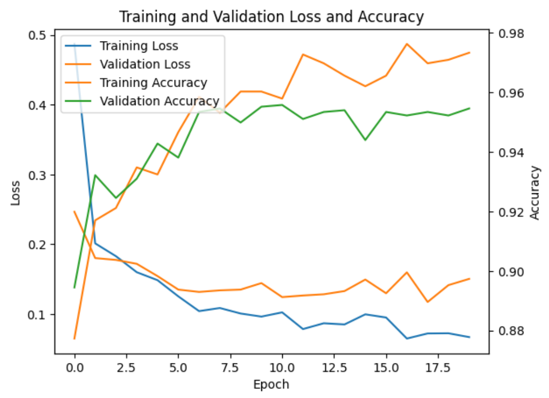

Fig. 5. Accuracy and Loss Curve for VGG16 with ADAM

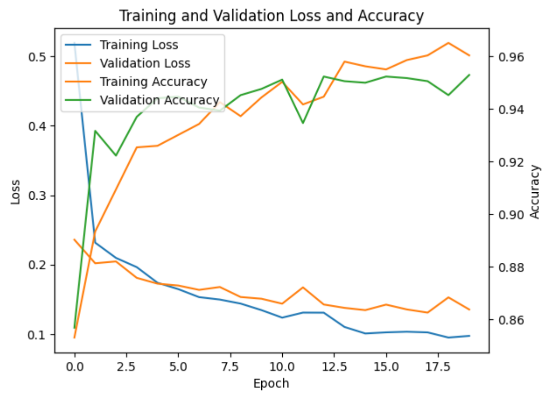

Fig. 6. Accuracy and Loss Curve for VGG16 with ADAMAX

is of paramount importance. In this research paper, we have explored the potential of deep learning techniques for the detection of malware on IoT devices. We have leveraged the power of Convolutional Neural Network (CNN) models to scrutinize the network traffic generated by IoT devices and identify potential malware threats.

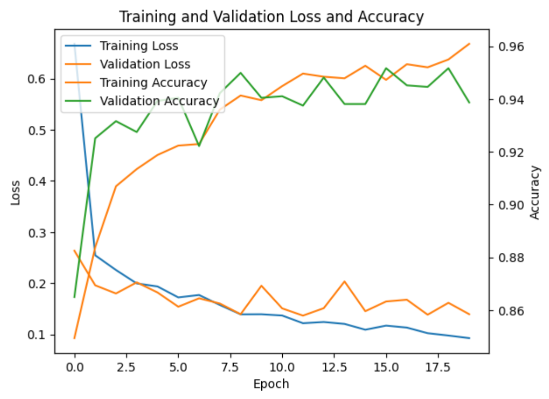

Fig. 7. Accuracy and Loss Curve for VGG16 with RMSProp

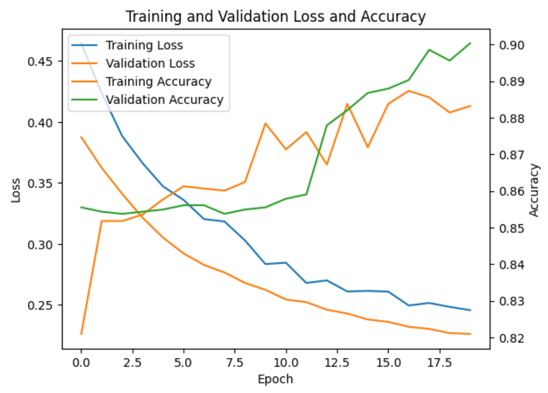

Fig. 8. Accuracy and Loss Curve for VGG16 with ADADELTA

Our results indicate that CNN models are highly effective in detecting malware on IoT devices, with notable levels of accuracy achieved in our experiments. This underscores the promise of deep learning techniques in addressing the growing threat of malware attacks on IoT devices.

Overall, our study emphasizes the critical need for robust and reliable methods for detecting and preventing malware attacks on IoT devices. With the ever-increasing number of IoT devices, it is crucial that we establish sound security measures to safeguard these devices against potential threats. The findings of this study will pave the way for further research in this domain and contribute to the broader objective of ensuring a secure and safe IoT ecosystem.

References

[1] Kumar, S. MCFT-CNN: Malware classification with fine-tune convolution neural networks using traditional and transfer learning.
[2] vgg19 and resnet50 architecture frameworks for image classification," in 2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON), vol. 1, 2021, pp. 96–99.
[3] Chaganti, R.; Ravi, V.; Pham, T.D. Deep Learning based Cross Architecture Internet of Things malware Detection and Classification.
[4] Diro, A.A.; Chilamkurti, N. Distributed attack detection scheme using deep learning approach for Internet of Things.
[5] HaddadPajouh, H.; Dehghantanha, A.; Khayami, R.; Choo, K.K.R. A deep recurrent neural network based approach for internet.
[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[7] system using ensemble of deep belief networks. Appl. Soft Comput. 2018, 71, 66–77. [CrossRef]
[8] Ben Fredj, O.; Mihoub, A.; Krichen, M.; Cheikhrouhou, O.; Derhab, A. CyberSecurity attack prediction: a deep learning approach.
[9] Kudugunta, S.; Ferrara, E. Deep neural networks for bot detection. Inf. Sci. 2018, 467, 312–322. [CrossRef]
[10] Kumar, S. MCFT-CNN: Malware classification with fine-tune convolution neural networks using traditional and transfer learning.
[11] McDermott, C.D.; Majdani, F.; Petrovski, A.V. Botnet detection in the internet of things using deep learning approaches. In
[12] Azmoodeh, A.; Dehghantanha, A.; Choo, K.K.R. Robust malware detection for internet of (battlefield) things devices using deep

[13] Naveed, M.; Arif, F.; Usman, S.M.; Anwar, A.; Hadjouni, M.; Elmannai, H.; Hussain, S.; Ullah, S.S.; Umar, F. A Deep LearningBased Framework for Feature Extraction and Classification of Intrusion Detection in Networks. Wirel. Commun. Mob. Comput.

[14] Pekta¸s, A.; Acarman, T. Botnet detection based on network flow summary and deep learning. Int. J. Netw. Manag. 2018, 28, e2039.

[15] Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018;

[16] Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018;

[17] V. Rastogi, Y. Chen, and X. Jiang, "Catch me if you can: Evaluating android anti-malware against transformation attacks," IEEE Trans. Inf. Forensics Secur., vol. 9, no. 1, pp. 99–108, 2013.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv Prepr. arXiv1409.1556, 2014.

[19] A. Azmoodeh, A. Dehghantanha, and K.-K. R. Choo, "Robust malware detection for internet of (battlefield) things devices using deep eigenspace learning," IEEE Transactions on Sustainable Computing, vol. 4, no. 1, pp. 88–95, Feb 2018.

[20] Z. Yuan, Y. Lu, and Y. Xue, "Droiddetector: android malware characterization and detection using deep learning," Tsinghua Sci. Technol., vol. 21, no. 1, pp. 114–123, 2016.

[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in 2009 IEEE conference on computer vision and pattern recognition, 2009, pp. 248–255

[22] T. Hsien-De Huang and H.-Y. Kao, "R2-d2: Color-inspired convolutional neural network (cnn)-based android malware detections," in 2018 IEEE International Conference on Big Data (Big Data), 2018, pp. 2633–2642.

[23] H. HaddadPajouh, A. Dehghantanha, R. Khayami, and K.-K. R. Choo, "A deep recurrent neural network based approach for internet of things malware threat hunting," Future Generation Computer Systems, vol. 85, pp. 88– 96, Mar 2018.

[24] T. N. Phu, L. Hoang, N. N. Toan, N. Dai Tho, and N. N. Binh, "C500-cfg: A novel algorithm to extract control flow-based features for iot malware detection," in 2019 19th International Symposium on Communications and Information Technologies (ISCIT). IEEE, Sep 2019, pp. 568–573.