Computer Science and Engineering 2022, 12(1): 30-38
DOI: 10.5923/j.computer.20221201.03

Detection of Android Adware Using Transfer Learning with Computer Vision

Kabombo Katutwa*, Dani E. Banda, Julien Shabani

Department of Electrical and Electronics Engineering, University of Zambia, Lusaka, Zambia

Abstract Adware, or advertising-supported malware, is a term used to describe unwanted software that displays advertisements. Adware can infect and root-infect a device, forcing it to download specific adware types and allowing attackers to steal personal information therefore making it a major threat category for consumers. Little research has been conducted on android adware analysis. In this research, the goal is to leverage the use of visualization and transfer learning to detect android adware, to show that a pre-trained neural network can be used to detect android adware from benign application package (APK) files and to compare the performance of selected pre-trained neural networks in accomplishing this task. Using the CICMalDroid2020 dataset, the Dalvik Executable (DEX) files were extracted from APK files and then converted to grayscale images. These images were then used as input to pre-trained deep neural networks that have been trained on the ImageNet dataset. The pre-trained models used are VGG16, ResNet 50, Inception v3, EfficientNet v2 and MobileNet v2. The results show a maximum performance measurement 92% on the F1 – score metric which was achieved by the MobileNet v2 model.

Keywords Android Adware, Computer Vision, Deep learning, Pre-trained model, Convolutional Neural Network (CNN), Transfer learning

1. Introduction

People use their smartphones in both their personal and professional lives today thereby making them an essential aspect of life. There are an estimated 2.6 billion active smartphone users [1]. The rise in smartphone users has also led to an increase in malicious programs targeting mobile devices, that is, mobile malware. Malicious software (malware) is a program that has an intention of causing harm to the operating system kernel or some security sensitive application or data without the user's consent [1]. Criminals attempt to take advantage of flaws on other people's smartphones for their own gain. Additionally, over the past years' malware authors have become less recreational-driven and more profit-driven as they are actively searching for sensitive, personal, and enterprise information [2].

There has been an increase in the usage of handheld devices with the android platform having the larger market share, in 2018 the worldwide mobile application (app) downloads were approximated to be 194 billion [3]. With the growth in the adoption of technology, malware has become an increasing problem.

* Corresponding author:
kabombo@gmail.com (Kabombo Katutwa)
Received: Jul. 28, 2022; Accepted: Aug. 15, 2022; Published: Aug. 23, 2022
Published online at http://journal.sapub.org/computer

Android is the most popular smartphone operating system in the world as of 2018 [4]. Since its release, sales of smartphones running on the android platform has grown strongly over the years [4]. Because of this popularity, android presents itself as an attractive attack platform for adversaries looking to maximize their impact on victims [5].

Applications like WeChat, TikTok, and mobile banking applications are used in our daily lives and continue to play an increasingly important role. Most of these applications have access to users' private information such as their location, debit/credit card, and contact information. Almost all applications access the users' private data, and although this provides users with better personalized services it may also result in information leakage of private data and economic loss. As cited by [6], Dogru et al. stated that android malicious applications keep on emerging continuously, and this security concern has gained increasing attention in both industry and academic fields.

The android operating system has 72% mobile operating system market share for the period May 2019 to May 2020 [7].

By mobile operating system market share, android is the dominant player and with this, in combination with its liberal and open application marketplaces (compared to Apple's locked-down iOS application ecosystem), has meant that it has quickly become the mobile platform of choice for malware authors [8].

1.1. Background

To address the plethora of malware and its associated security concerns, researchers are constantly developing and upgrading malware detection systems. Mobile malware detection is the process of classifying unknown mobile applications into benign and malicious [9].

There are different types of malicious software and one of these is called adware. Adware is a form of malware that downloads and displays unwanted advertisements, which are often offensive and always unsolicited [10].

Advertisements are used to promote or sell a product, an idea, or a service. The advent of the internet, and later the smartphone, has pushed marketing strategies towards digitizing advertisements as it's the new norm and the digital channel is rapidly growing with no signs of slowing down [11]. Shahzad et al. [12] states that the adware problem is growing continuously due to the profound monetary gains for adware developers, and it is largely agreed upon that user awareness of adware and its possible negative effects is still minimal.

Avast, a global leader in digital security and privacy, reported that adware continues to be the most significant threat on android phones and tablets, with 45% of mobile threats being adware in the first five months of 2021 [13].

1.2. Problem Statement

Because of the popular use, android devices present themselves as an attractive attack platform for numerous types of malwares with the following problems:

a) Proliferation of malware targeted at android mobile devices.
b) Malware authors using obfuscation techniques to make their malware difficult to detect using traditional static and dynamic methods of analysis.
c) Numerous zero-day malware that easily evades signature-based malware detection.
d) Highly specialized skills are required to analyse malware, and this involves huge volume of data for malware analysts to review which is time consuming and may lead to analysis fatigue.
e) The approach of using computer vision and transfer learning has not been utilized in analysing mobile android adware classification and detection.

1.3. Objectives

The objectives of this research are:

a) Examine android adware.
b) Investigate transfer learning and computer vision techniques in android adware detection.
c) Evaluate the performance of pre-trained neural networks used in this research in detecting android adware.

1.4. Research Questions

To achieve the research objectives, an attempt to answer the following questions was made:

a) Can computer vision be applied in the detection of android adware?
b) Are the selected pre-trained deep neural network models effective in the detection of Android adware?
c) Which pre-trained deep neural network models used in (b) will produce the best performance metric outcomes in the detection of android adware from benign apps?

1.5. Research Contributions

This research aims to contribute to the body of knowledge by firstly carrying out research work that leverages computer vision and transfer learning in to detecting android adware apps from benign android apps. Through knowledge obtained by reviewing various literature, previous work in this area has not utilized computer vision and transfer learning to detect android adware from benign apps. Secondly, performance evaluation was conducted on selected pre-trained convolutional neural networks to determine which of the pre-trained models performed the most effective and efficient in detecting android adware in this research.

1.6. Significance of Study

This research is important in that it will aid malware analysts in their efforts of analysing android adware from benign apps. In addition, it will reduce the time of carrying out malware analysis as it does not require expert knowledge in feature extraction due to the ability of deep learning models utilizing the automatic extraction of features on samples being tested. Though a lot of work has been done on android malware detection in general, very little focus has been put on the adware family [14], this research therefore expands on the work done in this area.

1.7. Scope of Study

The scope of the research will be as follows:

a) The research will be focused on adware on the android platform.
b) The dataset used will focus solely on android adware and benign files that are free and publicly available.
c) The research will utilize selected pre-trained deep neural networks.

2. Related Works

Malware authors are using numerous obfuscation tactics to evade traditional static and dynamic malware analysis techniques used in anti-malware solutions. The use of artificial intelligence and visualization techniques has shown great effectiveness and efficiency in classifying and detecting different types of malwares.

Image classification is a fundamental problem in computer vision [15]. In the early stage, Haralick et al. [16] describes some easily computable textural features based on gray tone

spatial dependencies and illustrates their application in category-identification tasks.

In 2011 Nataraj et al. [17] proposed a simple yet effective method for visualizing and classifying malware using image processing techniques. Malware binaries were visualized as grayscale images, with the observation that for many malware families, the images belonging to the same family appear very similar in layout and texture.

The image classification method does not require any disassembly or execution of the actual malware code. Moreover, the image textures used for classification provide more resilient features in terms of obfuscation techniques, and for encryption [17].

Image based detection and classification has proved to be effective, because it leverages the structural similarity between the known and new malware samples [18]. Moreover, visual analytics helps analysts to recognise patterns in malwares' code and behaviour, thus helping them to come up with better results [18].

Gennissen [19] proposed a system for detecting Android malware where ten (10) types of images were created by adding domain knowledge to image conversion. With the Dalvik opcode and API information, the classes.dex was converted into a fractal shaped Hilbert curve image and used a CNN model with two layers.

Thakuri et al. [20] carried out work detecting android malware using its image sections. In their work, android malware images are created using different sections of malicious applications. Four types of malware images generated were the android manifest, classes.dex, resources, and certificate files. To extract features from the malware images the GIST algorithm was used. To perform the classification various machine learning classifiers such as Support Vector Machines (SVM), K-Nearest Neighbors, Random Forest, and Naïve Bayes are used in [20]. Their work compared the results obtained by these different classifiers resulting in the model GIST +SVM outperforming all other classifiers.

The author [21] performed experiments to gauge the difficulty in identifying each family of adware from the benign apps without ads and benign apps with ads. He observed that the deep neural network multilayer perceptron (MLP) identifies each family with higher accuracy compared to traditional machine learning approaches.

Alani et al. [22] presented a machine learning system called AdStop that detects android adware. They used features from the network flow of outgoing and incoming traffic to an android device.

Dobhal et al [23], carried out research on android adware. They used several machine learning algorithms namely Logistic Regression (LR), Linear Discriminant Analysis (LDA), K-Nearest Neighbors (K-NN), Classification and Regression Trees (CART), and Naive Bayes (NB). They used two approaches, one for binary classification and multi-class classification.

Bagui et al [24] highlighted the need to develop better methods of detecting adware due to the rise of more and complex evasive malware, specifically adware as, very little focus has been put on the adware family compared to other types of malwares.

Mohammed et al. [24] demonstrated the use of deep learning that includes transfer learning-based approach in malware detection tasks and showed that the features learned by the deep CNN models (ResNet) performed significantly better than the hand-crafted GIST features proposed in the literature, and slightly better than the features learned by the proposed shallow CNN model.

Yadav et al. [25] presented an EfficientNet-B4 CNN-based method for android malware detection. It used image-based malware representations of the bytecode obtained from android DEX files as input to the EfficientNet-B4 network to extract relevant features. The extracted features are then passed through a GAP layer and fed into a softmax classifier.

Zhang et al. [26] proposed an android malware detection model based on the time convolution neural network (TCN). Their approach used an image-based analysis method to avoid the malicious applications, which may have symmetric encryption, and confusion of the traditional static analysis process. Firstly, they directly read and fused the XML files and two DEX files to create grayscale image data sets with different feature combinations. Secondly, they then carried out experiments by using two-dimensional CNN and MobileNet v2 to analyse the differences of the feature combinations. From their results they found that adding visual features of XML files can reduce the dependence of the malware detection model on single DEX file features.

Bhodia et al. [31] used transfer learning based on pre-trained deep learning models that have been trained on massive image datasets. They carried out numerous experiments with the pre-trained models and compared them with simple machine learning algorithm, namely the K-Nearest Neighbors (K-NN) that used features directly extracted from executable files and not image analysis. These experiments showed that the pre-trained models outperformed the K-NN in zero-day simulated experiments.

The authors in [32] stated that CNN based on the VGG16 and Inception v3 pre-trained models are being used in the malware classification and intrusion detection research.

Our study is centred on applying visualization techniques and transfer learning as this allows for automatic feature engineering of the input image files so that we can classify adware files from benign files. Previous studies to classify and detect android adware apps from benign apps have not utilized computer vision and transfer learning.

3. Methodology

The method used in this research follows the steps outlined below:

a) Data Collection and Pre-processing.
b) Conversion of adware and benign APK files to grayscale images.

c) Training the pre-trained neural network models on the images obtained in (b).
d) Testing and evaluating the trained models.

Figure 1 shows a visual representation of our proposed approach. The figure shows the steps that we follow in implementing our solution. The pre-processing stage involves preparing the collected dataset to the required input format for each pre-trained neural network. The transfer learning step involves the training of the pre-trained models using the grayscale images obtained in the previous step. The evaluation and interpretation steps involve testing the trained models on data not used during the training phase and evaluating the performance metrics values respectively. This is explained in detail the following sections:

3.1. Data Collection and Pre-processing

The collected android APK files were obtained from the Canadian Institute of Cybersecurity at the University of New Brunswick from the dataset named CICMalDroid2020 [27] – [28].

The collected sample files were a total of three thousand and forty-eight (3,048) Adware and benign APK files. During the pre-processing some APK files were corrupted resulting in a lower number of files used in the experiments. The successfully processed files were broken down into, one thousand five hundred and fifteen (1515) android adware and one hundred and eight (108) benign APK files respectively.

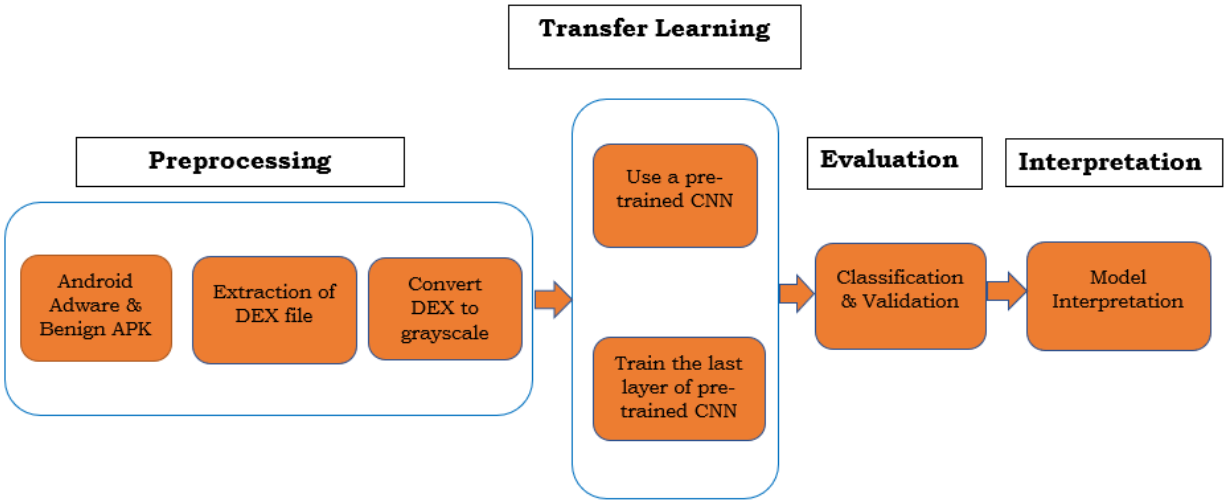

Figure 1. Proposed Approach

3.1.1. Conversion of DEX File to Image

The APK files were processed to extract the classes.dex files from the APK and once the DEX files were extracted, we converted the DEX files into portable network graphics (PNG) format grayscale images. Example of the image files is shown in figure 2:

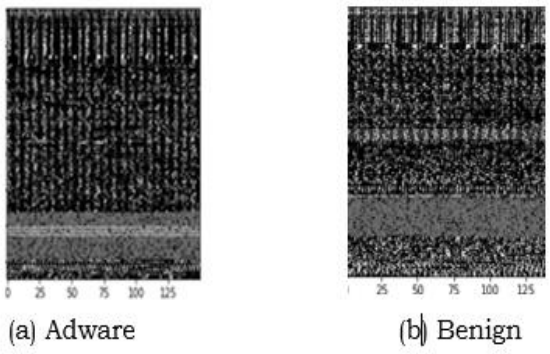

Figure 2. Sample Adware and Benign images

The images were split into training, validation and testing sets in the ratio of 70%, 20% and 10% respectively.

Each input image file was resized to align with the default input image size of each pre-trained model that we used as indicated in the table 1.

Table 1. Image Input Size

SNO	PRE-TRAINED MODEL	IMAGE INPUT SIZE
1.	VGG16	224 * 224
2.	ResNet 50	224 * 224
3.	Inception V3	299 * 299
4.	EfficientNet v2	300 * 300
5.	MobileNet v2	224 * 224

3.2. Transfer Learning

The models used in the study were built using pre-trained models that are trained on a large-scale dataset called ImageNet [29]. It is a large database or dataset of over 14 million images. It was designed by academics for use in computer vision research. The models that we used were namely VGG16, ResNet50, Inception v3, EfficientNet v2 and MobileNet v2.

3.3. Evaluation

To evaluate the performance of the models four (40 performance measures of a binary machine learning (ML) -based classifiers were used. The following terms were used to quantify the results of each model using the confusion matrix:

True Positive (**TP):** an image is an ADWARE image and classifier marks it is as ADWARE

True Negative (**TN):** an image is a BENIGN image and classifier marks it is as BENIGN

False Positive (**FP):** an image is a BENIGN image and classifier marks it is as ADWARE

False Negative (**FN):** an image is an ADWARE image and classifier marks it is as BENIGN

Each of the performance measures used and their respective formula is defined below:

Accuracy: is the measure of all the correctly identified cases. It is most used when all the classes are equally important.

$$\textbf{Accuracy} = \frac{(\mathrm{TP}+\mathrm{TN})}{((\mathrm{TP}+\mathrm{FP}\ +\mathrm{TN}+\mathrm{FN}))} \quad (1)$$

Precision: is the ratio of the probability that an app is classified as an adware app correctly.

$$\textbf{Precision} = \frac{\mathrm{TP}}{(\mathrm{TP}+\mathrm{FP})} \quad (2)$$

Recall: is the ratio of the total adware apps that are classified as adware.

$$\textbf{Recall} = \frac{\mathrm{TP}}{(\mathrm{TP}+\mathrm{FN})} \quad (3)$$

F-Measure or F1 score: It is a measure of a test's accuracy by considering both the precision and recall scores into a single measure of performance, where an F1-score is usually between 0.0 and 1.0 which closer to 1 is good while closer to 0.0 is poor performance.

$$\textbf{F}-\textbf{Measure}\ (\textbf{F1 score}) = \frac{2*(\mathrm{Precision}*\mathrm{Recall})}{(\mathrm{Precision}+\mathrm{Recall})} \quad (4)$$

3.4. Interpretation

With the establishment of the performance metrices the effectiveness of the models was derived from these values, with the highest percentage values indicating the best performing model in this research.

3.5. Development Environment

To perform the research work we used Jupyter Notebook development environment. For the dataset pre-processing, and Google Colaboratory or "Colab" for short was used to develop and train the models. Additionally, Anaconda Integrated Development Environment (IDE) was also used. This is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment.

- For the pre-processing, a system with the following specifications Intel (R) Core (TM) i3-6100U CPU @ 2.30GHz, 2304 MHZ, 2 Core(s), 4 Logical Processor(s)
- 8 GB RAM and 256 Solid State Disk Drive

For the development and testing of the model, Colab was used due to providing free access to computing resources and the following advantages:

- It has pre-installed machine learning libraries such as Keras and TensorFlow.
- It allows saving of work to the cloud.
- Google Research provides their dedicated graphical processing units (GPUs) and tensor processing units (TPUs) for personal machine learning projects.

4. Results and Discussion

For each of the selected pre-trained CNN model stated in section 3.2 training was conducted and their performances compared with each trained model's performance output in this research.

The training and validation accuracy values began at high values due to the pre-trained model utilized the existing knowledge learnt previously on the ImageNet [29] database. The difference in the training and validation accuracies are an indication of the minimal model overfitting as the difference in values was not too wide. The training accuracy was derived from the training dataset and the validation accuracy shows how well the model performed on this data (the validation dataset) that the model had not previously seen.

The graphs below show the training and validation accuracy of each pre-trained model.

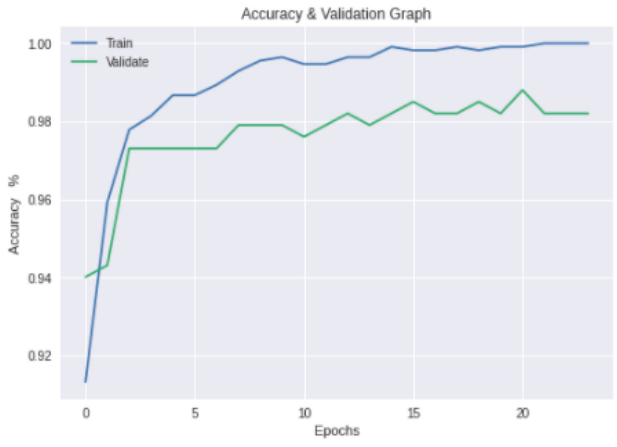

Figure 3. VGG16 Training & Validation Accuracy

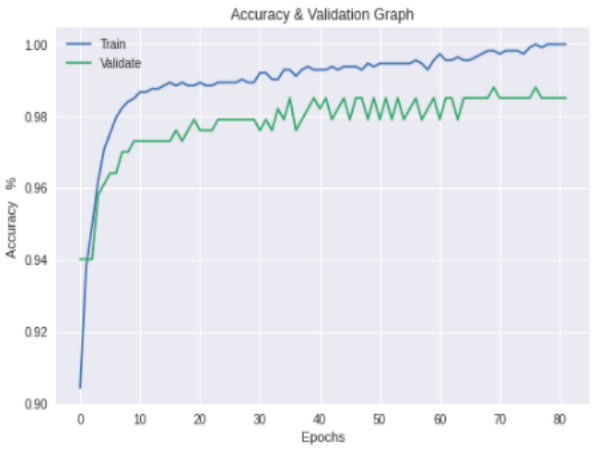

Figure 4. ResNet 50 Training & Validation Accuracy

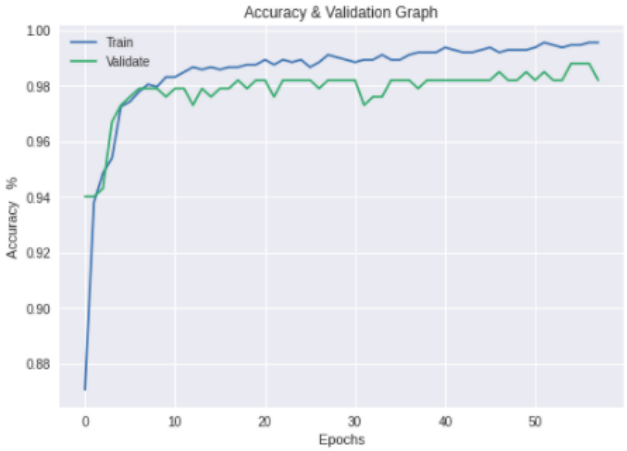

Figure 6. EfficientNet v2 Training & Validation Accuracy

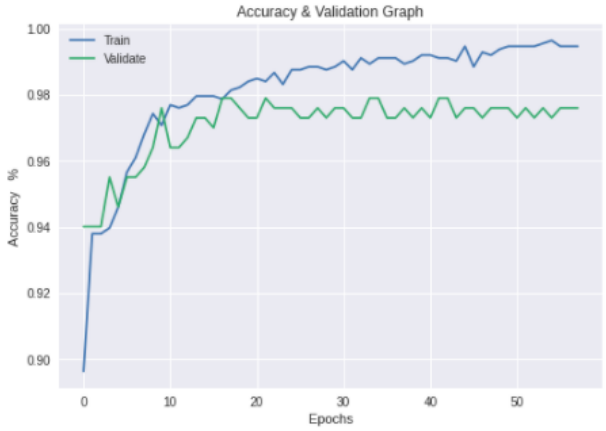

Figure 5. Inception v3 Training & Validation Accuracy

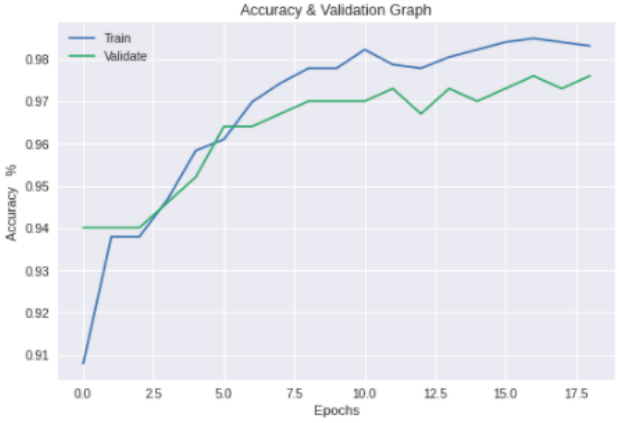

Figure 7. MobileNet v2 Training & Validation Accuracy

Table 2. Pre-Trained Model Performance

PRE-TRAINED MODEL	ACCURACY	PRECISION	RECALL	F1-SCORE	TRAINING DURATION (MINUTES)	EPOCHS
VGG 16	84%	91%	92%	91%	19	21
RESNET 50	81%	89%	91%	90%	45	82
INCEPTION v3	82%	90%	93%	91%	29	58
EFFICIENTNET v2	83%	90%	91%	90%	35	58
MOBILENET v2	83%	91%	94%	92%	26	19

From the experimental results obtained as shown table 2, we can see that the models produced model accuracy averaging 83% and an F1 -score average of 91%.

On the F1 score metric, the MobileNet v2 had the highest value of 92% while the lowest F1 score was recorded by the ResNet 50 and EfficientNet v2 having a percentage of 90%.

The training duration of the models varied amongst the models with the VGG16 model having recorded the least amount of training time of nineteen (19) minutes and the ResNet 50 pre-trained model took the longest by running for a duration of forty-five (45) minutes. The number of epochs also varied with the MobileNet v2 pre-trained model having the least number of epochs and ResNet 50 having the highest number of epochs.

4.1. Comparison with Prior Work

Bagui and Benson [14], used network traffic data available in the CICAndMal2017 dataset by Lashkari et al. [30] to analyse and classify adware families. They performed feature selection using information gain and classification was performed using traditional machine learning techniques, specifically J48 Decision Tree, Naïve Bayes and OneR. The results of the three classifiers, used as binary classifiers presented a highest average classification rate of 68% and highest average Attack Detection Rates (ADR) of 62.64% for the Adware families using the J48 Decision Tree classifier. This compared to the accuracy and F1-Score obtained in our research, which had a higher rate of 83% and

91% respectively. This shows that the use of computer vision in the research results obtained shows a significant improvement over previous work for classification of adware malware done in [14].

When a comparison is made with the proposed solution in the work done in [30], it was observed that the detection accuracy they had a higher accuracy and F1 score of 97.08% and 97.09% respectively compared to the accuracy and F1-score that was obtained in this research.

Additionally, we compare the work done by Dobhal et al [23] in binary classification of Android Adware using various machine learning algorithms by illustrating results in the table 3.

Table 3. Comparison of the Proposed System with State-of-the-Art Approaches

Machine Learning Algorithm	F1 - Score	Accuracy
Logistic Regression (LR)	88%	71.12%
Linear Discriminant Analysis (LDA)	88%	79.28%
K-Nearest Neighbors (KNN)	83%	89%
Classification And Regression Trees (CART)	94%	91.72%
Naive Bayes (NB)	87%	77%
Average in proposed approach	***91%***	***83%***

As can been seen in table 3, apart from the Classification and Regression Trees (CART) algorithm, the F1 score the proposed approach outperforms the other algorithms used in [23].

In Suresh's dissertation [21] experiments were done with machine learning algorithms like Random Forests (RF), Support Vector Classifier (SVC), Ad boost and MLP respectively showing that using dynamic features alone for detecting android adware had an accuracy of about 76% however this was improved when experimented with combined features which resulted in an accuracy of about 84%. This does show that our obtained experimental results performed better when compared to the dynamic features approach used by [21]. However, the use of combined features approach in [21] shows a better performance than the proposed approach by 1% as the highest accuracy results obtained in our proposed approach was 83%.

5. Conclusions

Based on the outcome of the experimental work done in our research we confirmed that the use of visualization, image processing techniques and transfer learning when applied to binary classification and detection of android adware from android benign apps performs well as can be seen from the accuracy and F1 -Score metric values that we obtained.

In comparison, each pre-trained deep neural network's performance results showed a small variance on all the performance metrics of accuracy, precision, recall and F1-score with the MobileNet v2 obtaining the highest F1 score. This leads us to conclude that pre-trained models that are created and adequately trained to solve a similar problem or task can be used to successfully detect android adware from benign apps. However, in contrast there was notable variation in the time taken to train the models with the VGG 16 and ResNet 50 taking the shortest time and taking the longest time respectively.

The study has shown that indeed transfer learning and computer vision sufficiently achieve the intended objectives and that the overall performance is adequate.

We can therefore conclude that our experimental study was able to meet the set objectives with limitations noted and recommended as future works and areas of improvement.

5.1. Recommendations

The work in this research aimed to establish the use of computer vision and transfer learning in android adware verses Android benign app binary detection. The study was able to show that this approach performs well and is comparable in performance and in some instances outperforms other methods. We therefore recommend for adoption of this approach in classifying and detecting android adware from benign apps.

5.2. Future Work

For future work, research can be done that uses the proposed approach to conduct multi-class classification of android adware into its various families. Other possible future work would be to combine the DEX file and other files in the APK file, then subsequently convert the resulting files into images that are applied to pre-trained CNN. Additionally, developing a mobile app, based on the work in this research, that can detect android adware in real-time while downloading APK files from the google playstore can be pursued.

ACKNOWLEDGEMENTS

I would like to thank my supervisors Dr. Dani.E. Banda, and Mr. Julien Shabani of School of Engineering at the University of Zambia for the invaluable guidance and contributions through out the period of conducting this research.

REFERENCES

[1] J. DE WIT, "Dynamic detection of mobile malware using real-life data and machine learning", Masters Thesis, University of Twente, 2018.

[2] K. Tam, A. Feizollah, N. B. Anuar, R. Salleh, and L. Cavallaro, "The evolution of Android malware and Android analysis techniques", ACM Computing Surveys (CSUR), vol. 49, no. 4, p. 76, 2017.

[3] https://www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/ [accessed 11th February 2022].

[4] https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/ [accessed 11th February 2022].

[5] Sophos Ltd, SophosLabs 2018 Malware Forecast. Oxford, UK: Abingdon Science Park, 2017.

[6] X. Jiang, B. Mao, J. “Guan and X. Huang, "Android Malware Detection Using F”ne-Grained Features", Scientific Programming, vol. 2020, pp. 1-2, 2020. Available: https://www.hindawi.com/journals/sp/2020/5190138/. [Accessed” 24 June 2020].

[7] "Mobile Operating System Market Share Worldwide | StatCounter Global Stats", StatCounter Global Stats, 2020. [Online]. Available:https://gs.statcounter.com/os-market-share/mobile/worldwide. [Accessed: 23- May- 2020].

[8] Sophos Ltd, SophosLabs 2018 Malware Forecast. Oxford, UK: Abingdon Science Park, 2017. A. Karnik, “Performance of TCP congestion control with rate feedback: TCP/ABR and rate adaptive TCP/IP,” M. Eng. thesis, Indian Institute of Science, Bangalore, India, Jan. 1999.

[9] N. Lu, D. Li, W. Shi, P. Vijayakumar, F. Picc“alli and V. Chang, "An efficient combined deep neural network based malware detection framewo”k in 5G environment", Computer Networks, vol. 189, p. 107932, 2021. Available: 10.1016/j.comnet.2021.107932 [Accessed 25 June 2022].

[10] J. Gao, L. Li, P. Kong, T. F. Bissyandé and J. Klein, "Should You Consider Adware as Malware in Your Study?" 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER), 2019, pp. 604-608, doi: 10.1109/SANER.2019.8668010.

[11] Waizel. A, "Improving the Detection of New Malware Classes” y Transfer Learning", Master of Science, BEN-GURION UNIVERSITY OF THE NEGEV, 2015.

[12] Mohanta, A. and Saldanha, A., 2020. Malware analysis and detection engineering. 1st ed. New–York: Apress, pp.5 - 7.

[13] Avast Reports Continued Dominance of Adware Among Android Threats. 2021. Avast Reports Continued Dominance of Adware Among Android Threats. [online] Available at: <https://press.avast.com/avast-reports-continued-dominance-of-adware-among-android-threats> [Accessed 16 June 2022.

[14] S. Bagui and D. Benson, "Android Adware Detection Using Machine Learning", International Journal of Cyber Research and Education, vol. 3, no. 2, pp. 1-19, 2021. Available: 10.4018/ijcre.2021070101.

[15] Koli, J. D. “RanDroid: Android malware detection using random machine learning classifiers.” 2018 Technologies for Smart-City Energy Security and Power (ICSESP). IEEE, 2018.

[16] Haralick RM, Shanmugam K, Dinstein I. Textural features for image classification. IEEE Trans Syst Man Cybern. 1973; 3(6): 610-621.

[17] L. Nataraj, S. Karthikeyan, G. Jaco and B. Manjunath, "Malware images”, Proceedings of the 8th International Symposium on Visualization–for Cyber Security - VizSec '11, 2011. Available: 10.1145/2016904.2016908.

[18] Jung, J., Choi, J., Cho, S., Han, S., Park, M. and Hwang, Y., 2018. Android malware detection using convolutional neural networks and data section images. Proceedings of the 2018 Conference on Research in Adaptive and Convergent Systems.

[19] J. Gennissen, L. Cavallaro, V. Moonsamy, and L. Batina, Gamut: sifting through images to detect android malware, Bachelor thesis, Royal Holloway University, London, UK, 2017.

[20] Deepak Thakur, Jaiteg Singh, Parvez Faruki, Tanya Gera, Classification of Android Malware using its Image Sections. International Journal of Advanced Trends in Computer Science and Engineering, volume 9 No. 4, July - August, 2020, doi.org/10.30534/ijatcse/2020/288942020.

[21] Suresh, S., 2018. Analyzing Android Adware. Master of Science (MS). San Jose State University.

[22] M. Alani and A. Awad, "AdStop: Efficient flow-based mobile adware detection using machine learning", Computers & amp; Security, vol. 117, p. 102718, 2022. Available: 10.1016/j.cose.2022.102718.

[23] Dobhal, D., Das, P., Aswal, K., 2019. Detection of Android Adwares by using Machine Learning Algorithms. International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-8 Issue-4S, April 2019.

[24] Mohammed, T. M., Nataraj, L., Chikkagoudar, S., Chandrasekaran, S., & Manjunath, B. S. (2021). Malware Detection Using Frequency Domain-Based Image Visualization and Deep Learning. arXiv preprint arXiv: 2101.10578.

[25] Yadav, P., Menon, N., Ravi, V., Vishvanathan, S., Pham, D. T., EfficientNet Convolutional Neural Networks-based Android Malware Detection”, Computers & Security (2022), doi: https://doi.org/10.1016/j.cose.2022.102622.

[26] Zhang,W.; Luktarhan, N.; Ding, C.; Lu, B. Android Malware Detection Using TCN with Bytecode Image. Symmetry, 2021, 13, 1107. https://doi.org/10.3390/sym13071107.

[27] Samaneh Mahdavifar, Andi Fitriah Abdul Kadir, Rasool Fatemi, Dima Alhadidi, Ali A. Ghorbani; Dynamic Android Malware Category Classification using Semi-Supervised Deep Learning, The 18th IEEE International Conference on Dependable, Autonomic, and Secure Computing (DASC), Aug. 17-24, 2020.

[28] Samaneh Mahdavifar, Dima Alhadidi, and Ali A. Ghorbani (2022). Effective and Efficient Hybrid Android Malware Classification Using Pseudo-Label Stacked Auto-Encoder, Journal of Network and Systems Management 30 (1), 1-34.

[29] “ImageNet,” www.image-net.org. https://www.image-net.org/index.php. Accessed 1st January 2022.

[30] Arash Habibi Lashkari, Andi Fitriah A. Kadir, Laya Taheri, and Ali A. Ghorbani, “Toward Developing a Systematic Approach to Generate Benchmark Android Malware Datasets and Classification”, In the proceedings of the 52nd IEEE International Carnahan Conference on Security Technology (ICCST), Montreal, Quebec, Canada, 2018.

[31] Bhodia N, Prajapati P, Troia FD, Stamp M. Transfer learning for image-based malware classification. ArXiv 2019. arXiv: 1903. 11551.

[32] D. Vasan, M. Alazab, S. Wassan, H. Naeem, B. Safaei and Q. Zheng, "IMCFN: Image-based malware classification

using fine-tuned convolutional neural network architecture", Computer Networks, vol. 171, p. 107138, 2020. Available: 10.1016/j.comnet.2020.107138.