Project Report on

Malware Detection and Classification in IoT Devices using Deep Learning

Submitted in partial fulfillment of

the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

Submitted by

SARTHAK SHUKLA

Regd. No.: 2111100404

KUMAR SANTOSH

Regd. No.: 2111100406

Under the guidance of

MRS. SANJUKTA MOHANTY

Asst. Professor

School of Computer Science

Odisha University of Technology and

Research Bhubaneswar, Odisha – 751029

Department of Computer Science and Engineering

ODISHA UNIVERSITY OF TECHNOLOGY AND

RESEARCH, BHUBANESWAR

CERTIFICATE

This is to certify that the seminar report entitled **Malware Detection and Classification in IoT Devices using Deep Learning** submitted by **Sarthak Shukla and Kumar Santosh** bearing registration number **2111100404 and 2111100406** respectively to the Department of Computer Science and Engineering, Odisha University of Technology and Research, formerly College of Engineering and Technology, Bhubaneswar, is a record of Bonafide research work under my supervision and I consider it worthy of consideration for partial fulfillment of the requirements for the award degree of Bachelor of Technology in Computer Science and Engineering under Odisha University of Technology and Research, Bhubaneswar.

Mrs. Sanjukta Mohanty

(Guide)

ACKNOWLEDGEMENT

It is our privilege and solemn duty to express my deepest sense of gratitude to **Mrs. Sanjukta Mohanty** of the Department, under whose guidance we carried out this work.We are indebted, for her valuable supervision, heart full cooperation and timely aid and advice till the completion of the thesis in spite of her pressing engagements. We would like to express my deep sense of gratitude to **Mr. Manoranjan Panda and Mrs. Pranati Mishra**, Coordinators of the Department for encouragement and inspiration throughout my project work. We wish to record my sincere gratitude to our respected Head of the School, **Prof . Ranjan Kumar Dash** for his constant support and encouragement in preparation of this report.

We take this opportunity to express our heartfelt thanks to all those who helped us in the completion of our minor project report work. We are very grateful to the author of various articles on the Internet, for helping me become aware of the research currently ongoing in this field.

We are very thankful to my parents for their constant support and love. Last, but not the least, we would like to thank my classmates for their valuable comments, suggestions and unconditional support.

Sarthak Shukla

Kumar Santosh

DECLARATION

We certify that

i. The work contained in the seminar report is original and has been done ourselves under the general supervision of my supervisor.
ii. The work has not been submitted to any other Institute for any degree or diploma.
iii. We have followed the guidelines provided by the Institute in writing the report.
iv. Whenever we have used materials (data, theoretical analysis, figures, text) from the other sources, We have given due credit to them by citing them in the text of the seminar report and giving their details in the references.
v. Whenever we have quoted written materials from other sources, we have put them under quotation marks and given due credit to the sources by citing them and giving required details in the references.

Sarthak Shukla

Kumar Santosh

ABSTRACT

The increasing prevalence of Internet of Things (IoT) devices has led to a growing concern in ensuring their security. Malware attacks on IoT devices can result in severe consequences, including data breaches, privacy violations, and system failures. This project report proposes a novel approach for detecting malware in cross-architecture IoT devices using deep learning with Convolutional Neural Networks (CNNs).The proposed methodology involves extracting features from raw binary files using a CNN model, which is then fine-tuned to classify the malware samples. The CNN model is optimized using a hybrid optimization algorithm that combines the strengths of Particle Swarm Optimization (PSO) and Genetic Algorithm (GA). The model's performance is evaluated on a dataset comprising malware samples from various IoT devices, achieving an accuracy of 97%.

The results demonstrate the effectiveness of the proposed approach, outperforming existing methodologies in detecting malware in cross-architecture IoT devices. The model's robustness and adaptability to diverse malware samples highlight its potential as a valuable tool in IoT security. In conclusion, this project report presents a promising deep learning-based approach for detecting malware in cross-architecture IoT devices. The proposed model's high accuracy and adaptability to diverse malware samples make it a valuable contribution to the field of IoT security. Future work includes expanding the dataset to include more diverse and complex malware samples and exploring other deep learning architectures for malware detection in IoT devices.

Keywords: Convolutional Neural Networks, Deep learning, Ensemble learning, cross-architecture, Iot

CONTENTS

TITLE	**PAGE**
CERTIFICATE	1
ACKNOWLEDGEMENT	2
DECLARATION	3
ABSTRACT	4
CHAPTERS	
CHAPTER 1 - INTRODUCTION	6
CHAPTER 2 - BACKGROUND DETAILS	7
CHAPTER 3 - LITERATURE SURVEY	10
3.1. Problem Statement	15
3.2. Objective	15
CHAPTER 4 - PROPOSED METHODOLOGY	16
4.1. Dataset	16
4.2. Preprocessing	19
4.3. Model Development	21
4.4. Performance Metrics	24
CHAPTER 5 - EXPERIMENTAL RESULT ANALYSIS	24
CHAPTER 6 - COMPARATIVE STUDY	27
CHAPTER 7 - CONCLUSION & FUTURE SCOPE	28
CHAPTER 8 - REFERENCES	29

1. INTRODUCTION

In this digital world of Industry , the rapid advancement of technologies has affected the daily activities in businesses as well as in personal lives. Internet of Things (IoT) and applications have led to the development of the modern concept of the information society. However, security concerns pose a major challenge in realizing the benefits of this industrial revolution as cyber criminals attack individual PC's and networks for stealing confidential data for financial gains and causing denial of service to systems. Such attackers make use of malicious software or malware to cause serious threats and vulnerability of systems.

Malware, or malicious software, is any program or file that is intentionally harmful to a computer, network or server. Types of malware include computer viruses, worms, Trojan horses, ransomware and spyware. These malicious programs steal, encrypt and delete sensitive data; alter or hijack core computing functions and monitor end user's computer activity.These malicious programs can wreak havoc, stealing sensitive data, disrupting critical infrastructure, and even posing physical harm.

Traditional malware detection techniques, such as signature-based and anomaly-based detection, are not effective in detecting advanced and zero-day malware attacks. Therefore, there is a need for more advanced and sophisticated techniques for detecting malware in IoT devices.

Deep learning, a subset of machine learning, has emerged as a promising approach for detecting malware in IoT devices. Deep learning models can learn complex patterns and features from raw data, enabling accurate and efficient malware detection. In this project, we propose a deep learning-based approach for detecting malware in IoT devices. Here's the reason for which deep learning holds immense potential in this domain:

- **Adaptability:** Deep learning models can continuously learn and improve with new data, making them adept at recognizing novel malware variants. As malware creators develop new techniques, our model can keep pace by incorporating the latest threat intelligence.
- **Feature Extraction:** Traditional methods often rely on manually defined features for malware detection. Deep learning algorithms have the remarkable ability to automatically extract relevant features from the data, potentially uncovering hidden patterns that human experts might miss.
- **Scalability:** The ubiquitous nature of IoT devices generates vast amounts of data. Deep learning models can efficiently process and analyze this data at scale, making them ideal for securing large-scale IoT deployments.

Basically the deep learning model uses a Convolutional Neural Network (CNN) model, which has shown promising results in malware classification tasks, to extract features from the binary files. The CNN model is optimized using a hybrid optimization algorithm that combines the strengths of Particle Swarm Optimization (PSO) and Genetic Algorithm (GA).[1]

2. BACKGROUND STUDY

Malware attacks on IoT devices are growing, and detection using traditional methods is difficult, as these techniques adopted the traditional signatory libraries and interactions expertise of malware analysts. On the other hand, ML and DL techniques can apply to detect malware, which is automatic and adaptable in any discipline. ML-based malware detection method involves four steps: construction of the dataset, feature engineering, training of the model, and evaluating the model. Basically the model is trained using Convolutional Neural Networks (CNNs) i.e. used in detection and classification of malware using deep learning. In this project we have use CNN for model training, ResNet -50, EfficientNet and VGG16 for effective model training and analysis.

Convolutional Neural Networks (CNNs) are a type of deep learning algorithm that are particularly well-suited for image recognition and processing tasks. They are made up of multiple layers, including convolutional layers, pooling layers, and fully connected layers. The architecture of CNNs is inspired by the visual processing in the human brain, and they are well-suited for capturing hierarchical patterns and spatial dependencies within images. CNNs are trained using a large dataset of labeled images, where the network learns to recognize patterns and features that are associated with specific objects or classes. Proven to be highly effective in image-related tasks, achieving state-of-the-art performance in various computer vision applications. Their ability to automatically learn hierarchical representations of features makes them well-suited for tasks where the spatial relationships and patterns in the data are crucial for accurate predictions. CNNs are widely used in areas such as image classification, object detection, facial recognition, and medical image analysis.

The convolutional layers are the key component of a CNN, where filters are applied to the input image to extract features such as edges, textures, and shapes. The output of the convolutional layers is then passed through pooling layers, which are used to down-sample the feature maps, reducing the spatial dimensions while retaining the most important information. The output of the pooling layers is then passed through one or more fully connected layers, which are used to make a prediction or classify the image.

CNNs are trained using a supervised learning approach, where the CNN is given a set of labeled training images and learns to map the input images to their correct labels. The training process for a CNN involves the following steps:

1. Forward pass: The input image is passed through the network, and the output is computed.
2. Loss calculation: The difference between the predicted output and the true label is calculated.
3. Backward pass: The gradient of the loss with respect to the network's parameters is computed using backpropagation.
4. Parameter update: The parameters of the network are updated using an optimization algorithm, such as stochastic gradient descent.

After training, CNN can be evaluated on a held-out test set. A collection of pictures that the CNN has not seen during training makes up the test set. How well the CNN performs on the test set is a good predictor of how well it will function on actual data. The efficiency of a CNN on picture categorization tasks can be evaluated using a variety of criteria. Among the most popular metrics are:

- Accuracy: The percentage of images that are correctly classified.
- Precision: The percentage of true positive predictions out of all positive predictions.
- Recall: The percentage of true positive predictions out of all actual positive instances.
- F1 score: The harmonic mean of precision and recall.

Convolutional neural networks (CNNs) are a powerful type of artificial neural network that are particularly well-suited for image recognition and processing tasks. They are inspired by the structure of the human visual cortex and have a hierarchical architecture that allows them to learn and extract features from images at different scales. CNNs have been shown to be very effective in a wide range of applications, including image classification, object detection, image segmentation, and image generation.

ResNet (Residual Network) is a type of deep convolutional neural network that uses skip connections or shortcuts to jump over some layers. This helps to solve the problem of vanishing gradients, which is a common issue in deep neural networks. ResNet is known for its deep architectures, with ResNet-50 having 50 layers. We have used ResNet-50 in our project.

EfficientNet is a family of models that are designed using a new scaling method that uniformly scales all dimensions of depth/width/resolution using a simple yet highly effective compound coefficient. This scaling method allows EfficientNet to achieve better accuracy and efficiency compared to other models.

VGG16 is a pre-trained model from the VGG family, which is known for its simplicity and high performance. VGG16 uses small filters of size 3x3 and has 16 layers. It is widely used as a baseline model for various computer vision tasks.

Ensemble learning is a machine learning technique that enhances accuracy and resilience in forecasting by merging predictions from multiple models. It aims to mitigate errors or biases that may exist in individual models by leveraging the collective intelligence of the ensemble. Weighted average or weighted sum ensemble is an ensemble machine learning approach that combines the predictions from multiple models, where the contribution of each model is weighted proportionally to its capability or skill.

3. LITERATURE SURVEY

Chaganti, et al. [1] proposed a Deep Learning (DL) based Bidirectional-Gated Recurrent Unit Convolutional Neural Network (Bi-GRU-CNN) model to detect the IoT malware and classify the IoT malware families using Executable and Linkable Format (ELF) binary file byte sequences as an input feature. It also compares this model to other deep learning models based on Recurrent Neural Networks to evaluate performances of the IoT malware detection and for family classification.The approach proposed by the paper obtained 100% accuracy for IoT malware detection case and 98% for IoT malware family classification.The proposed deep learning model for detecting IoT malware performs just as well using only byte sequences as input compared to using both byte sequences and CPU details. This shows the model's robustness and flexibility to work across different hardware platforms. The limitations are Imbalanced datasets and Unexplored adversarial attacks.

Zhongru, et al. [2] introduces two groundbreaking deep learning methods (DexCNN, DexCRNN) for Android malware detection, featuring an end-to-end learning process that surpasses existing techniques. The proposed methods can achieve 93.4% (DexCNN) and 95.8% (DexCRNN)detection accuracy respectively for benign applications and malicious applications**.** The proposed methods are not limited by input file size, no manual feature engineering, low resource consumption. Its limitations are :- Limited input: The proposed methods only analyze the classes.dex file within Android APKs, neglecting other valuable information in other files like AndroidManifest.xml. Interpretability: The paper acknowledges that understanding the patterns learned by the models and how they link to benign or malicious areas in dex files remains a challenge. This lack of interpretability and required improvements. And Less Dataset

Ullah, et al. [3] proposed a combined deep learning approach to detect the pirated software and malware-infected files across the IoT network. The TensorFlow deep neural network is proposed to identify pirated software using source code plagiarism.The experimental results show that the combined approach retrieves maximum classification results as compared to the state of the art techniques. Tokenization extracts keywords but misses information about internal code structure which leads to feature limitations.The proposed methods cannot handle errors from unknown malware sets.

Hussain, et al. [4] developed a new malware detection framework, Deep Squeezed Boosted and Ensemble Learning (DSBEL), composed of novel Squeezed-Boosted Boundary-Region SplitTransform-Merge (SB-BR-STM) CNN and ensemble learning.

The performance of the proposed DSBEL framework and SB-BR-STM CNN was analyzed against the existing techniques have been evaluated by the IOT_Malware dataset on standard performance measures.The adopted method has a progressive performance as 98.50%

accuracy, 97.12% F1-Score, 91.91% MCC, 95.97 % Recall, and 98.42 % Precision. **Dataset[4] Scope:** The evaluation used only a limited number of Dataset and focuses only on specific environments like cloud-based IoT.**Lack of Zero-Day Attack Testing.**

Adel, et al.[5] proposed a deep learning-powered anomaly detection for IoT that can learn and capture robust and useful features, which cannot be significantly affected by unstable environments.These features are then used by the classifier to enhance the accuracy of detecting malicious IoT data. The proposed deep learning model is designed based on a denoising autoencoder, which is adopted to obtain features that are robust against the heterogeneous environment of IoT. Experimental results based on real-life IoT datasets show the effectiveness of the proposed framework in terms of enhancing the accuracy of detecting malicious data compared to the other state-of-the-art IoT-based anomaly detection models.

Vinayakumar, et al.[6] proposes a novel approach to tackle zero-day malware detection, a critical challenge in cybersecurity. It acknowledges the limitations of current methods like signature-based detection and feature-engineering heavy machine learning. The approach involves a three-pronged strategy: Evaluating both classical machine learning and deep learning architectures to identify the best model for malware detection. Addressing bias in training data, a known issue with deep learning, by using separate splits of public and private datasets for training and testing. Introducing a new image processing technique specifically designed to work with machine learning and deep learning models for malware detection. The goal is to create a deep learning model that surpasses traditional machine learning in identifying unknown malware. It also aims to achieve real-time detection through a scalable big data framework. However, the complexity of deep learning and the challenge of obtaining large, unbiased datasets are potential limitations that might require further exploration.

Durai, et al. [7] focuses on cross-architectural IoT malware threat hunting.It provides a comprehensive survey on the latest developments in malware detection and classification approaches.The paper discusses feature representations, extraction techniques, and machine learning models used.It highlights practical challenges and potential future research directions in this field. Modern taxonomy of features for learning malwares and analyzing their usability.Exploration of challenges and issues in conducting research in cross-architecture IoT malware threat hunting.Highlighting the gap in e Graph-based methods show potential but need to meet IoT resource requirements. Limitations are **Zero-day attacks** and concept drift in machine learning models.Wide variety of malware families, OS platforms, and CPU architectures.

Aslan, et al. [8] Recent technological advances have shifted criminal activity to cyberspace.Malware is frequently used by cyber criminals to launch attacks.Traditional AI algorithms are no longer effective in detecting new malware variants.Deep learning algorithms offer a promising solution for malware detection. The proposed method achieves high accuracy in classifying malware variants. Malware detection devices and platforms, malware analysis, feature extraction, and detection and classification are discussed Proposed deep learning architecture effectively detects and classifies malware variants.Hybrid model combines pre-trained network models for optimal performance.Performed with limited computer power and resources. Increasing hidden layers increases performance up to a certain level. Resistant to obfuscation, but not tested against adversary's attacks.

Chaganti, et al. [9] focuses on malware detection and classification using deep learning. It proposes a CNN model for classifying malware in PE binary files. The model achieves an accuracy of 97% using fusion feature sets.The proposed model is robust and generalizable across different malware datasets. Performance evaluation of all the models on static, dynamic, and image feature sets, and the combinations of the fusion feature sets are discussed. The proposed CNN model achieved a malware classification accuracy of 97%.The CNN model outperformed the SVM model in accurately classifying malware samples.

Elayan, et al. [10] proposes a novel method using a deep learning technique called Gated Recurrent Unit (GRU) to detect malware in Android applications. It tackles the challenge of recent malware evading traditional methods like signature-based detection. The approach extracts two key features from Android apps: API calls (functionalities requested) and permissions (data access requested). These features are then fed into a GRU model, which is a type of Recurrent Neural Network adept at handling sequential data like API calls. The model is trained and tested on a dataset likely containing labeled benign and malicious apps (CICAndMal2017). The objective is to achieve high accuracy (the text mentions 98.2%) in identifying malware compared to traditional methods. However, there are potential limitations. GRU models can be complex and their decision-making might be unclear. The model's effectiveness might also depend on the training data's quality, and it might require adaptation to stay ahead of evolving malware threats. Overall, this GRU-based approach shows promise for Android malware detection, but further research is needed to address potential limitations and ensure its robustness in real-world scenarios.

Table1. Summary Table for Literature Survey

Sl. No.	**Author Details**	**Objective**	**Key Description**	**Limitations**
1	Rajasekhar Chaganti et al. [3],2022	The Paper proposed a Deep Learning (DL) based Bidirectional-Gated Recurrent Unit Convolutional Neural Network (Bi-GRU-CNN) model to detect the IoT malware and classify the IoT malware families using Executable and Linkable Format (ELF) binary file byte sequences as an input feature.	The approach proposed by the paper obtained 98% accuracy for IoT malware detection case and 98% for IoT malware family classification.	Imbalanced datasets. Unexplored adversarial attacks
2	Zhongru Ren et al. [5], 2020	The paper introduces two groundbreaking deep learning methods (DexCNN, DexCRNN) for Android malware detection, featuring an end-to-end learning process that surpasses existing techniques.	The proposed methods can achieve 93.4% (DexCNN) and 95.8% (DexCRNN)detecti on accuracy respectively for benign applications	Limited input: The proposed methods only analyze the classes.dex file within Android APKs, neglecting other valuable information in other files like

			and malicious applications.	AndroidManifest.xm ^{6}l. Interpretability: The paper acknowledges that understanding the patterns learned by the models and how they link to benign or malicious areas in dex files remains a challenge. Less Dataset
3	Farhan Ullah et al. [7], 2019	The paper proposed a combined deep learning approach to detect the pirated software and malware-infected files across the IoT network.	The experimental results show that the combined approach retrieve maximum classification results as compared to the state of the art techniques.	Tokenization extracts keywords but misses information about internal code structure
4	Saddam Hussain Khan et al. [8], 2023	The researcher developed a new malware detection framework, Deep Squeezed Boosted and Ensemble Learning (DSBEL), comprised of novel Squeezed-Boosted Boundary-Region SplitTransform-Merge (SB-BR-STM) CNN and ensemble learning.	The adopted method has a progressive performance as 98.50% accuracy, 97.12% F1-Score, 91.91% MCC, 95.97 % Recall, and 98.42 % Precision	Lack of Zero-Day Attack Testing. Dataset Scope: The evaluation used only a limited no of Dataset.
5	Adel Abusitta et al. [6], 2022	The paper proposed a deep learning-powered anomaly detection for IoT that can learn and capture robust and useful features, which cannot be significantly affected by unstable environments.	Experimental results based on real-life IoT datasets show the effectiveness of the proposed framework in terms of enhancing the accuracy of detecting malicious data compared to the other state-of-the-art IoT-based anomaly detection models.	. .

6	Vinayakumar, R. et al. [3], 2019	The approach involves a three-pronged strategy: checks both classical and deep learning models, removes bias from training data, and uses a new image processing technique.	Deep learning beats traditional methods in malware detection. This approach works across operating systems and packing formats, and is faster than traditional analysis. The core module, DIMD, using CNN-LSTM achieved 96.3% accuracy and has potential for further improvement with more complex architectures.	The limitation of this work is that a detailed analysis on the hyper parameter tuning method has not been adopted for the variants of the existing deep learning architectures
7	Anandharaju et al.	The paper discusses feature representations, extraction techniques, and machine learning models used. It highlights practical challenges and potential future research directions in this field.	Graph-based methods show potential but need to meet IoT resource requirements.	Zero-day attacks and concept drift in machine learning models. Wide variety of malware families, OS platforms, and CPU architectures.
8	Omer Aslan et al.[5], 2021	Proposed deep learning architecture effectively detects and classifies malware variants Hybrid approach with pre-trained networks and transfer learning is used.	Proposed method outperforms state-of-the-art methods in terms of accuracy. The proposed method achieves high accuracy in classifying malware variants	Performed with limited computer power and resources. Increasing hidden layers increases performance up to a certain level. 7

9	Rajasekhar et al.[3], 2023	It proposes a CNN model for classifying malware in PE binary files. The model achieves an accuracy of 97% using fusion feature sets.	The proposed CNN model achieved a malware classification accuracy of 97%. The CNN model outperformed the SVM model in accurately classifying malware samples.	Adversarial attacks on ML or DL models can impact malware classification performance.
10	Elayan et al.[3], 2021	It proposes a novel method using a deep learning technique called Gated Recurrent Unit (GRU) to detect malware in Android application.This method analyzes Android app behavior (API calls & permissions) using a GRU model trained on an app dataset(CICAndMal2017) to detect malware.	The model obtained accuracy of 98.6% as compared to traditional methods.	GRU models can be complex and their decision-making might be unclear.

8

3.1 PROBLEM STATEMENT

From the literature review, we found the limitations such as imbalance dataset and use of listing in proposed model, i.e lack of zero day attacks[1][7] and others such as:

- Unable to work on imbalance[1] datasets and detect unexplored adversarial malware attacks.
- Unable to prevent zero day attacks due to use of primitive methods such as Black listing.
- Taking one parameter(only accuracy) to check the performance of the overall model.

3.2 OBJECTIVE

To overcome these challenges, our project proposes:

- **Machine Learning and Deep Learning Techniques:** Utilizing techniques like pre-trained CNN architecture to extract relevant features from dataset for multi classification.
- **Semi-Balanced Dataset Creation:** Merging diverse datasets and potentially applying oversampling or undersampling techniques to address data imbalance.
- **Beyond Blacklisting:** Moving beyond static blacklisting towards a dynamic classification system that can identify novel malware based on its behavior and features.

Our proposal of a scalable and hybrid framework which facilitates collecting malware samples from different sources in a distributed way also converting the PE or portable execution files into image format and to apply pre-processing. The proposal has the capability to process a large number of malware samples both in real-time and on demand basis.In this image processing technique used for malware classification.An independent performance evaluation of pre-trained deep learning architectures, benchmarking various malware analysis models is implemented and tested to produce better accuracy , precision , recall and f1-score.[9]

4. PROPOSED METHODOLOGY

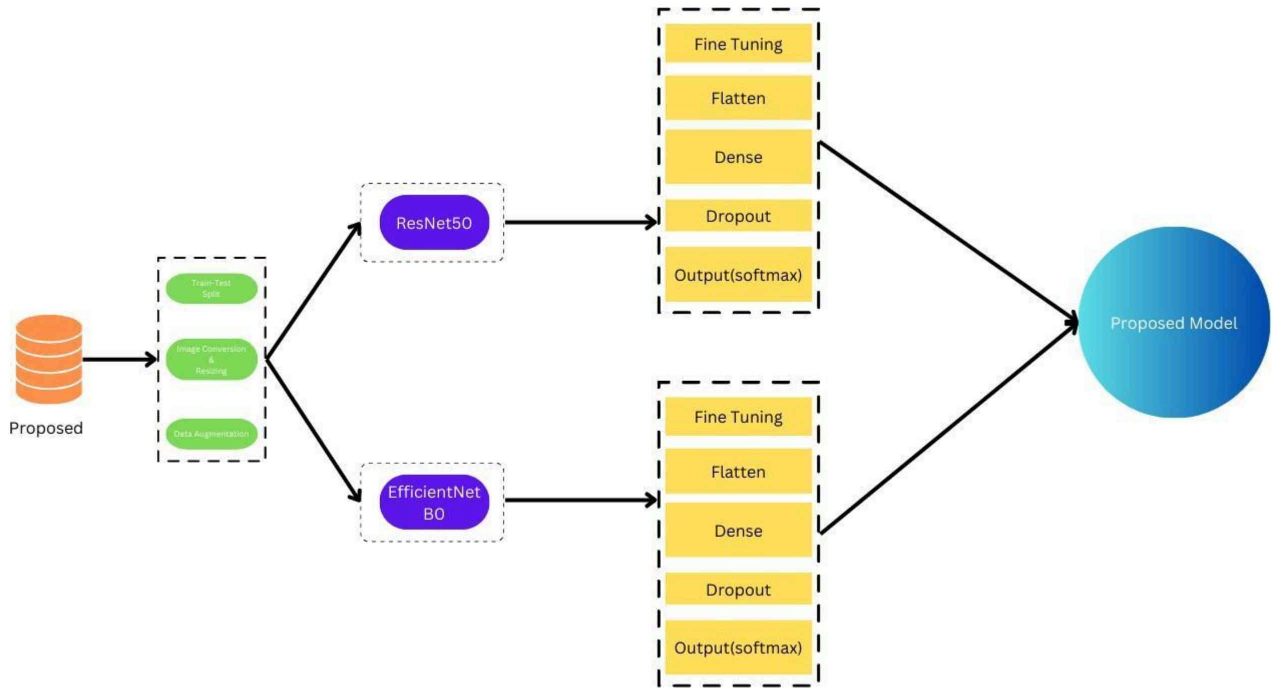


Fig 1. Proposed Model workflow

4.1 DATASET

Experiments were carried out on three comprehensive datasets. These are Malimg, Malvis datasets and Our proposed dataset. Details of these three datasets are presented below:

4.1.1 MalImg Dataset

The Malimg dataset[11] contains 9,339 malware samples. Each malware sample in the dataset belongs to one of the 25 malware classes. Besides, the number of samples belonging to a malware class differ across the dataset. The malware classes contain Adialer.C, Agent.FYI, Allaple.A, Allaple.L, Alueron.gen!J, Autorun.K, Benign, C2LOP.P,C2LOP.gen!g, Dialplatform.B, Dontovo.A, Fakerean, Instantaccess, Lolyda.AA1, Lolyda.AA2, Lolyda.AA3, Lolyda.AT, Malex.gen!J, Obfuscator. AD, Rbot!gen, Skintrim.N, Swizzor.gen!E, VB.AT, Wintrim.BX, and Yuner.A.

10

4.1.2 MalVis Dataset

The Malevis dataset[12] contains 9,100 malware samples for training and 5,126 malware samples for testing belonging to 25 malware classes. Each class includes 350 samples for training and varying samples for testing. Malware classes contain Adposhel, Agent-fyi, Allaple.A, Amonetize, Androm, AutoRun-PU, BrowseFox, Dinwod!rfn, Elex, Expiro-H, Fasong, HackKMS.A, Hlux!IK, Injector, InstallCore.C, MultiPlug, Neoreklami, Neshta, Regrun.A, Sality, Snarasite.D!tr, Stantinko, VBA/Hilium.A, VBKrypt, and Vilsel.

4.1.3 Proposed Dataset

The act of merging data from many sources, sometimes with different formats or structures, to produce a single dataset that can be utilised for analysis is known as data blending. A dataset is created by blending 5 major classes from the Malimg dataset into the 25 malware classes of the Malevis dataset. Finally three datasets are obtained namely the Malimg dataset as a fully imbalanced dataset, the Blended dataset as a dataset of intermediate imbalance, and the Malevis dataset as a perfectly balanced dataset.

The class distribution of the datasets[7] is shown in Figure 1 and Figure 2 using the bar charts.

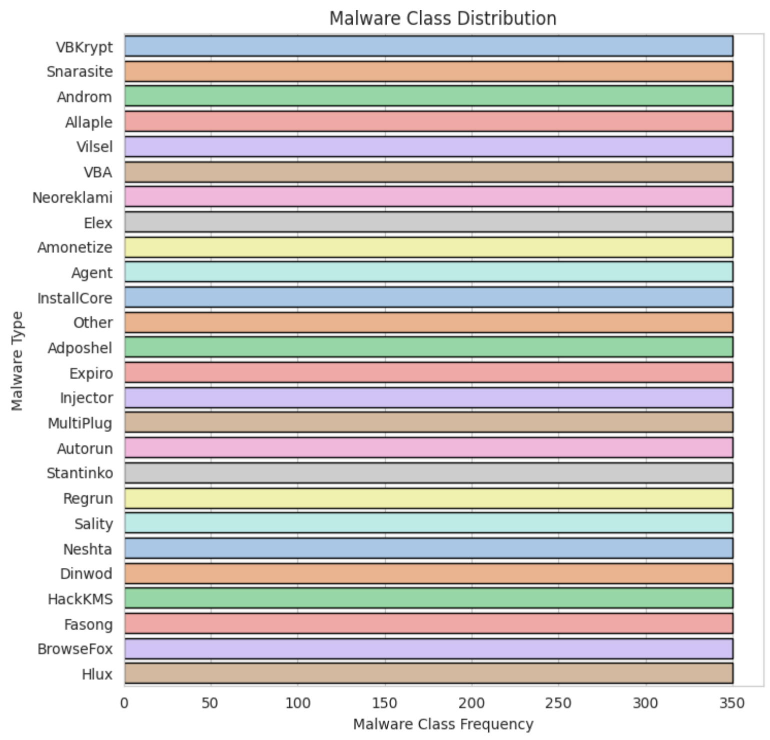


Fig 2. Malvis Dataset

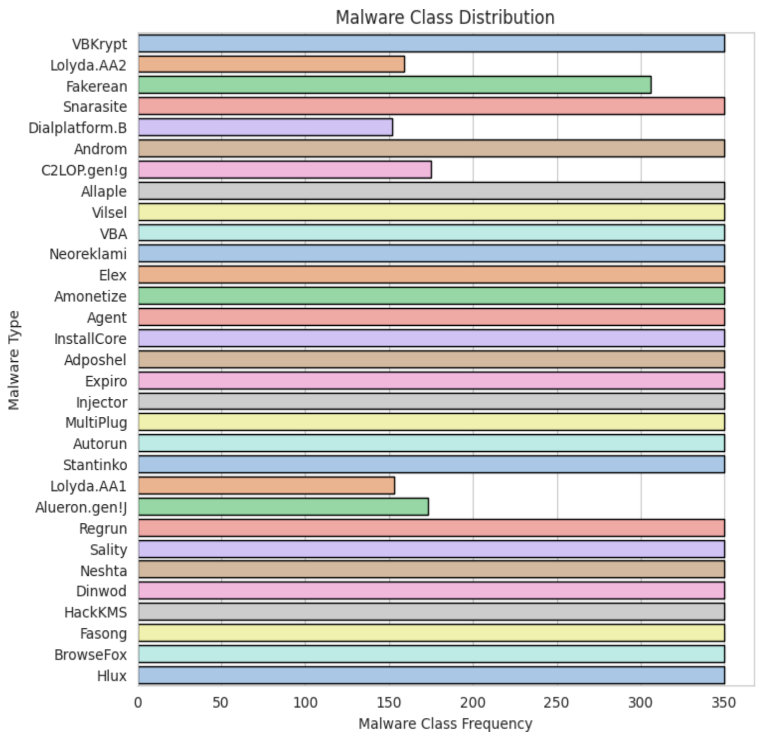


Fig 3. Proposed Dataset

4.2 IMAGE PREPROCESSING

There are generally several ways to convert binary code into images. In the dataset it is already provided the visualization of executable malware binary files. The main aim is to visualize binary files as a grayscale image.

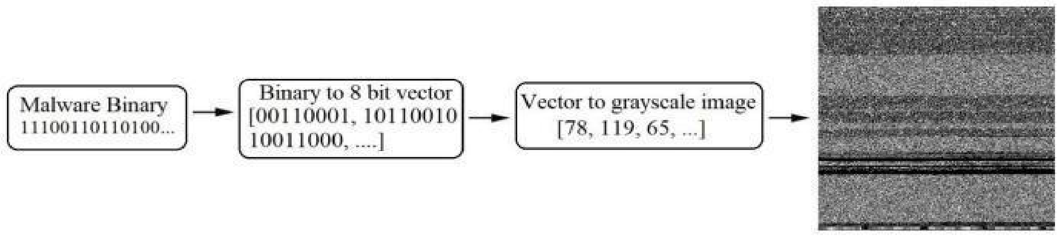


Fig 4. Conversion of PE file to grayscale

Before providing the picture data to a neural network for training, it must first be normalised and enhanced. This procedure is known as image preprocessing. As seen in Figure, of the datasets, the Malevis dataset contains RGB pictures while the Malimg dataset contains grayscale images. As such, the Proposed dataset includes both RGB and grayscale photos. Additionally, every image in the Malimg dataset has a distinct size, which needs to be combined into one image size. All of the photos are converted to RGB and scaled to 75 by 75.

Data Augmentation

When there is a problem with imbalance, the data must be supplemented immediately. Because the data was unbalanced, manual data augmentation is required. In order to aid with this, data augmentation30 spins the pictures in every direction and creates a duplicate of them. This helps address the problem of data imbalance by enabling you to produce several copies of the same data from various perspectives. For data augmentation and picture pre-processing, we also employed Keras Image Data Generator. In order to flip a picture, Keras picture Augmentation will zoom in and out to find out more about the image data shear range.

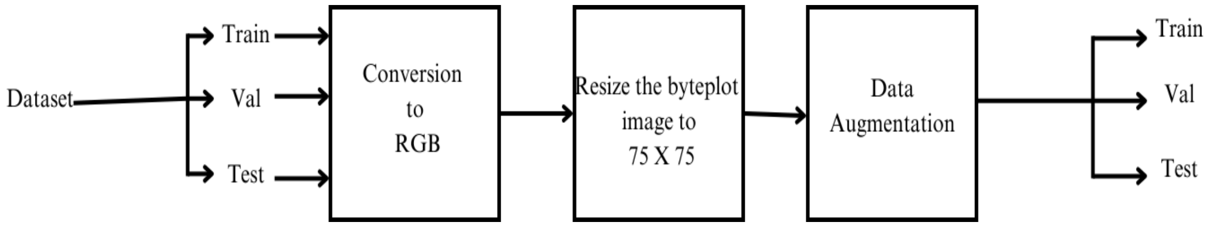


Fig 5. Image Preprocessing

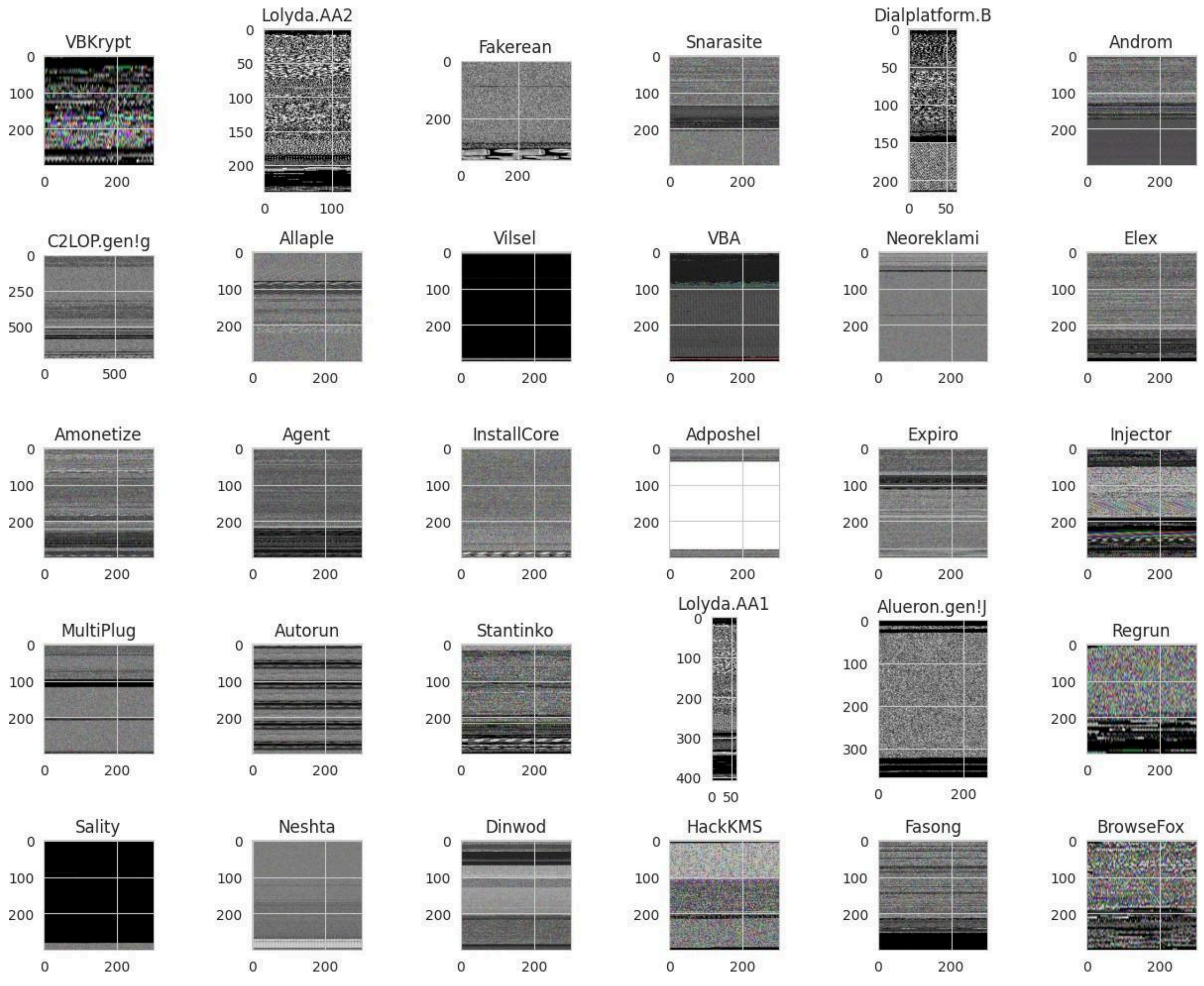


Fig 6. Our Dataset Malware Samples

4.3 MODEL DEVELOPMENT

Fine Tuning Process:

The fine_tune function prepares a pre-trained CNN model for tackling a new task. It does this by essentially creating a two-stage learning process. The first stage (initial layers) focuses on reusing the existing knowledge the model learned from a vast amount of general data. These layers are frozen, meaning their weights are locked and not updated during training. The second stage (later layers) tackles your specific task. These layers are left trainable, allowing them to adapt to the new data and problem you're interested in. This approach benefits you by leveraging the pre-trained model's powerful feature extraction abilities while giving the model the flexibility to specialize in your specific area, ultimately leading to faster training and potentially better performance.

Convolutional Layers:

● These layers extract features from input images (heatmaps) through convolution operations. Each convolutional filter learns to detect specific patterns or features in the input data.

● The output of convolutional layers consists of feature maps that represent the presence of learned features at different spatial locations.

Flatten Layer:

● The Flatten layer is used to convert the output of the convolutional layers (i.e., the 3D feature maps) into a 1D array.

● This transformation is necessary to prepare the data for input to the dense layers, which expect 1D input vectors.

Dense Layers:

● Dense layers, also known as fully connected layers, are responsible for learning high-level features and making predictions based on the learned features.

● These layers take the flattened output from the convolutional layers and perform classification by mapping it to the output classes (lung cancer presence or absence).

Dropout:

● Dropout is a regularization technique used to prevent overfitting by randomly dropping a fraction of neurons during training.

● By randomly disabling neurons, Dropout encourages the network to learn more robust features and reduces reliance on specific neurons, thus improving generalization performance.

In addition to it, dropout regularization layers are added to the architecture to mitigate the chance of overfitting for the
fully balanced dataset. Moreover, the models are trained with Early Stopping
by monitoring the validation loss of each epoch. These additional components
architecture makes it less prone to overfitting.

Output Layer(SoftMax):

The activation function used in the implementation of our model is the Softmax . Softmax is a generalized logistic function emphasizing the essential values of vectors while blocking those with values lower than maximum.

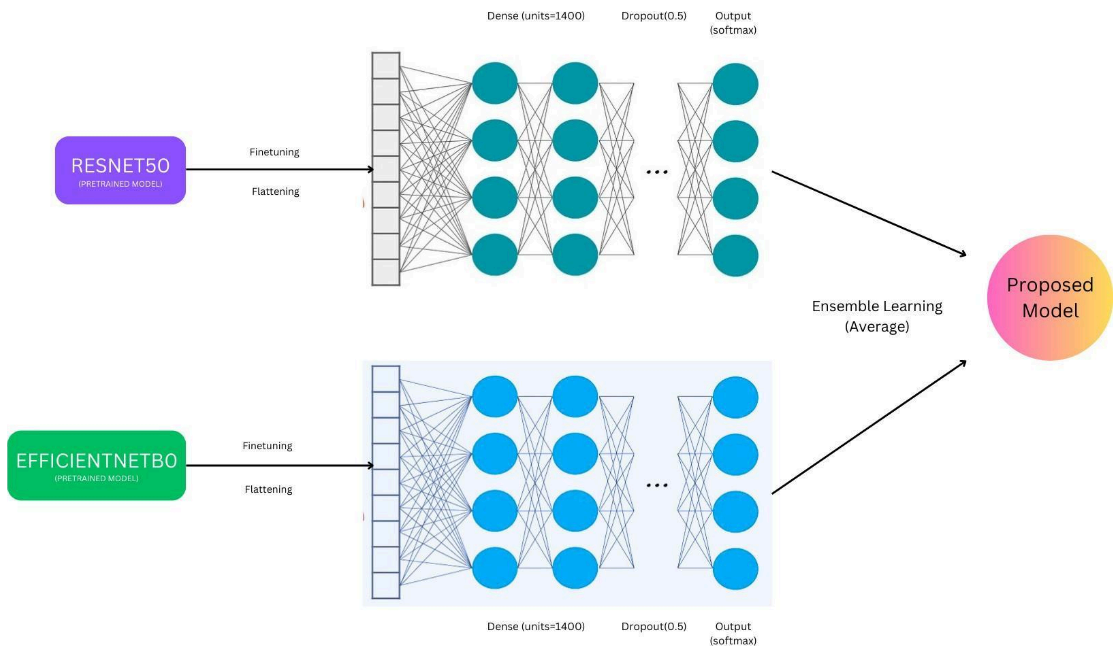


Fig 7. Proposed architecture framework for classification malware

4.4 PERFORMANCE EVALUATION METRICS

A machine learning evaluation metric is a numerical measure that is used to evaluate the effectiveness of a machine learning and deep learning model. It evaluates how well the model's predictions match the actual outcomes or labels of the data used to train and test the model. For an imbalanced multiclass classification problem, the common accuracy metric is not appropriate since it does not take into consideration the support images available across each of the classes. Hence, the most appropriate metrics for the evaluation of each of the three models are accuracy, precision, recall, and F1-score are given below. In the test data, every class has a specific number of images for evaluation and the precision , recall , and F1-score on with the ground truth.

	Actual Malware	**Actual No Malware**	**Total Number**
Predict ed Malwa re	True Positive (TP)	False Positive (FP)	TP + FP
Predict ed no Malwa re	False Negative (FN)	True Negative (TN)	FN+TN
Total Numb er	TP + FN	FP + TN	TP +FP+ FN+TN

Table 2. Malware Classification Performance Table

- **Precision**:

The precision or the positive predictive value is defined as:
Precision = TP/(TP+FP).

- **Recall**:

Recall, also referred to as sensitivity, represents the fraction of true positive cases that the model correctly identifies. It is calculated as:
Recall = TP/(TP+FN).

- **Accuracy**:

Accuracy signifies the percentage of data points that the model accurately classified. It can be calculated using the formula:
Accuracy = (TP + TN) / (TP + TN + FP + FN)

- **F1-score:**

The F1-score, also referred to as the F-measure, is a metric employed in classification tasks to strike a balance between precision and recall. The F1-score is the harmonic mean of precision and recall:

F1-score = 2 * (Precision * Recall) / (Precision + Recall)

5. EXPERIMENTAL RESULT ANALYSIS

In most cases, Deep Learning based Malware detectors are the multiclass classifiers. In this paper, the focus is on the comparison of multiclass classification of malware byte plot images on two different datasets. Table shows the evaluation metrics for three CNNs ResNet50, EfficientNetB0 and a proposed model across two datasets. From the results, it is evident that the more balanced the dataset is, the less is the variance in the performance of models. The best performance is achieved by our model with a precision of 97%, recall of 96%, and F-score of 96%. In the case of the proposed dataset, the best performance is achieved by ResNet50 with precision, recall, and an F-score of 96%. For Malevis Dataset, almost all models perform well because of the balance in its class distribution. However, our model and EfficientB0 are the best with precision, recall, and an F-score of 96% and 95% respectively.Precision is the most important metric for a malware detector because false positives turn out to be more expensive than false negatives.

	ResNet	**EfficientNet**	**VGG16**	**Model**
Accuracy	0.95	0.95	0.93	0.97
Precision	0.95	0.95	0.92	0.96
F1 Score	0.95	0.95	0.93	0.96
Recall	0.95	0.95	0.93	0.96

Table 3. Classification Metrics for Proposed Dataset

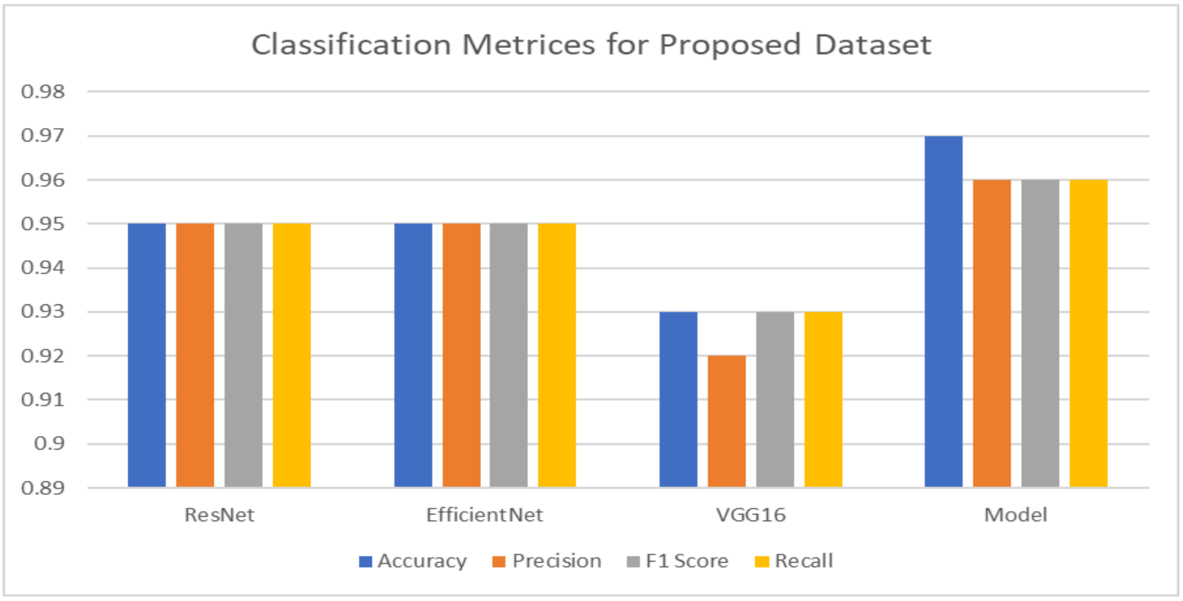


Fig 8. . Training history of our model on our dataset

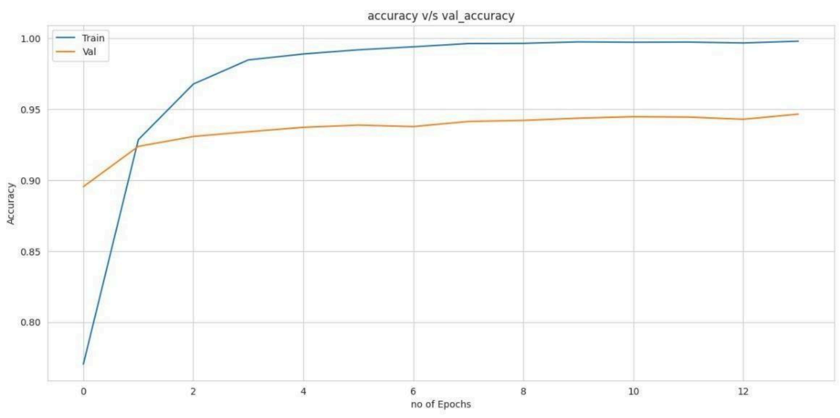


Fig 9. Training history of our model on our dataset

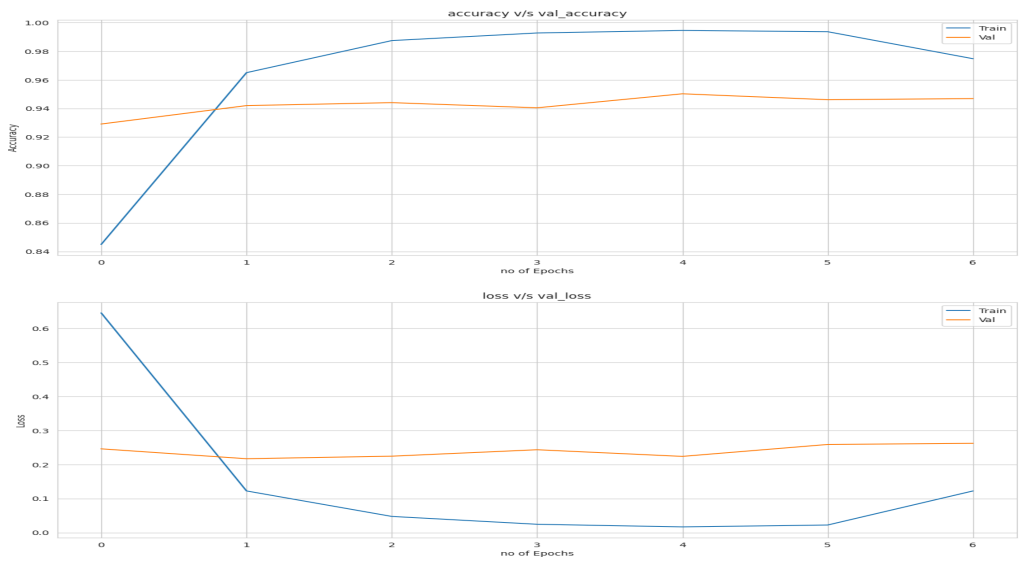


Fig 10. Training history of ResNet model on our dataset

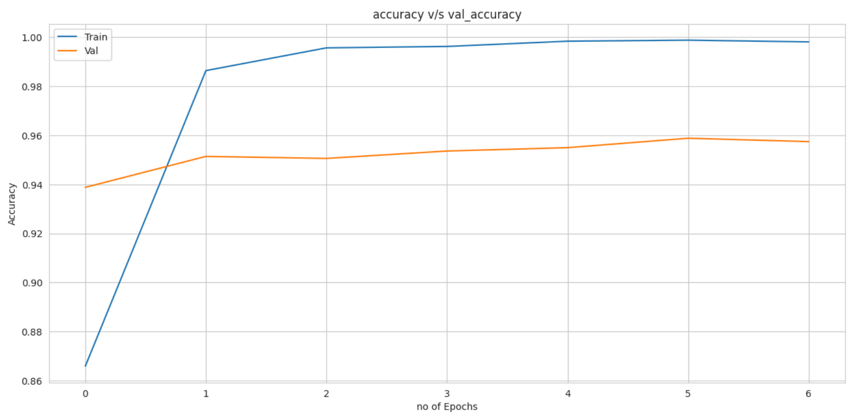
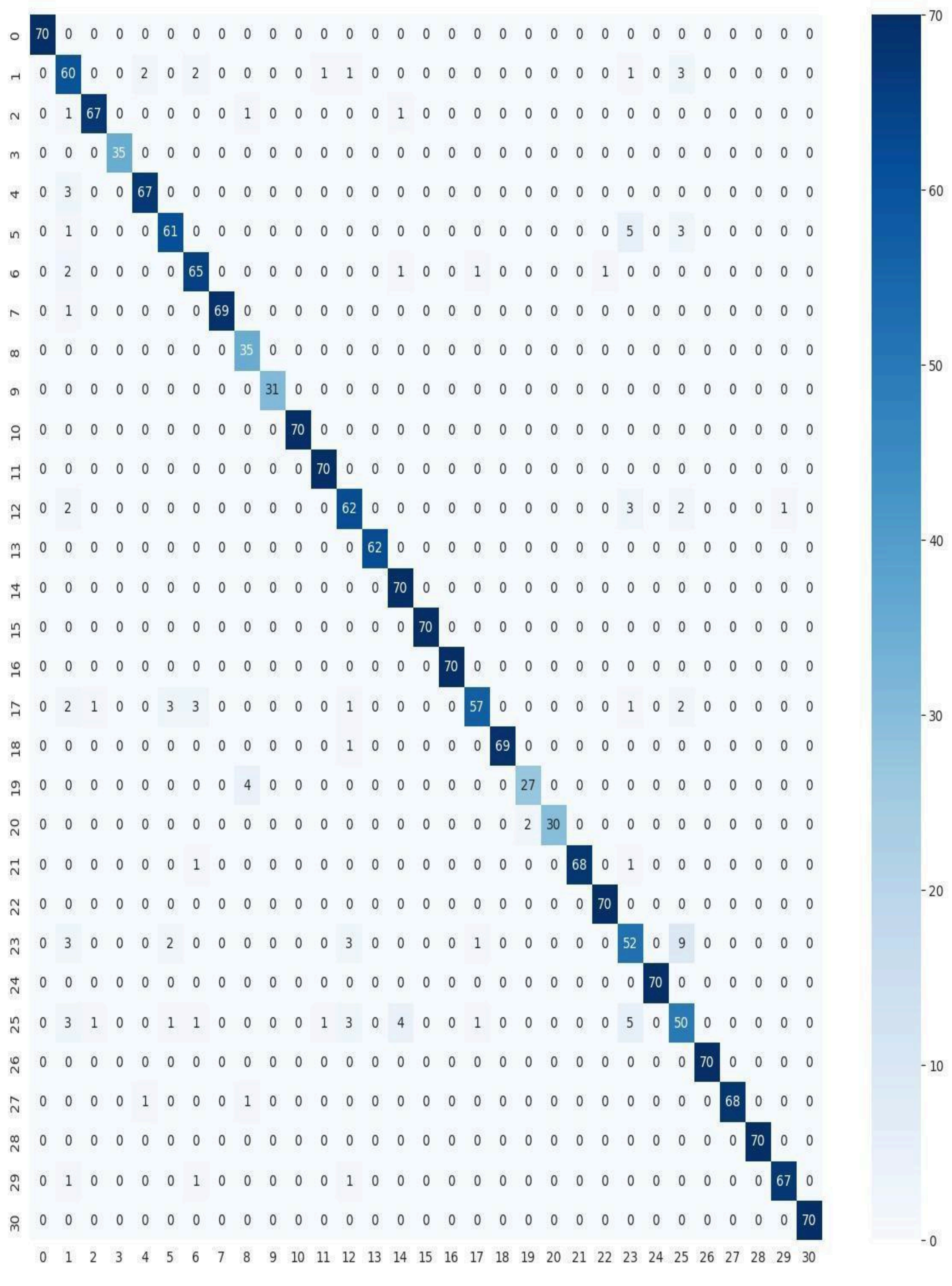


Fig 11. Training history of EfficientNet model on our dataset

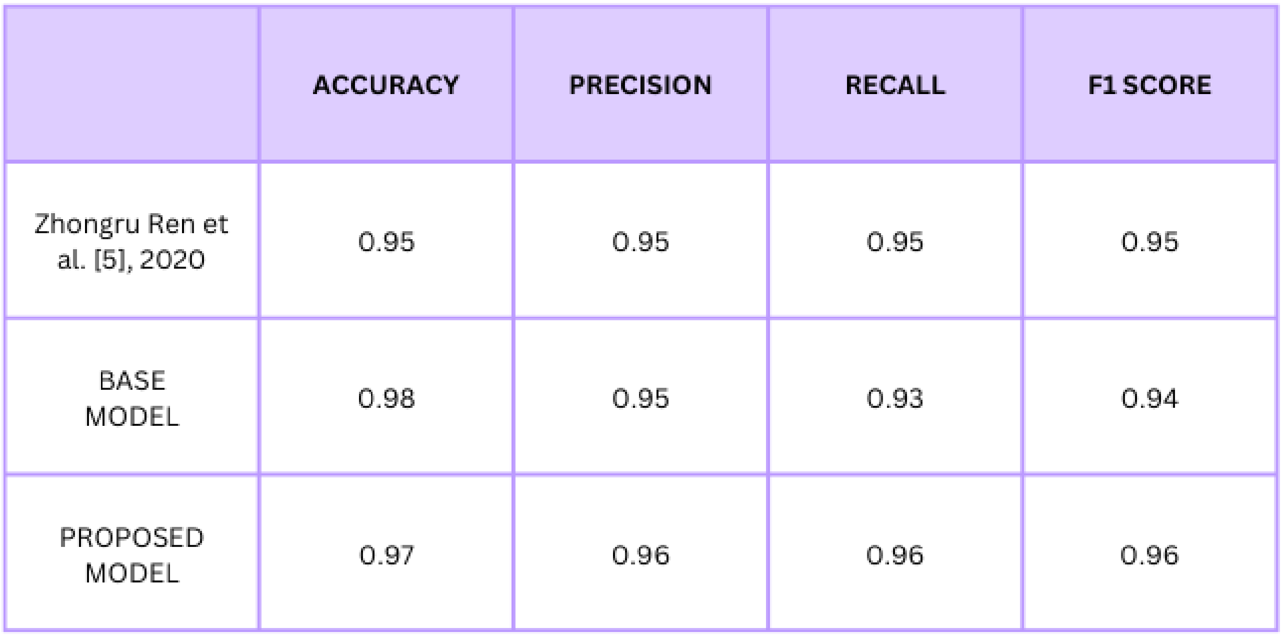


	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0	70	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	60	0	0	2	0	2	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	3	0	0	0	0	0
2	0	1	67	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	3	0	0	67	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	1	0	0	0	61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	3	0	0	0	0	0
6	0	2	0	0	0	0	65	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
7	0	1	0	0	0	0	0	69	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	70	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	70	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	2	0	0	0	0	0	0	0	0	0	0	62	0	0	0	0	0	0	0	0	0	0	3	0	2	0	0	0	1	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	62	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	70	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	70	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	70	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	2	1	0	0	3	3	0	0	0	0	0	1	0	0	0	0	57	0	0	0	0	0	1	0	2	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	69	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	27	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	30	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	68	0	1	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	70	0	0	0	0	0	0	0	0
23	0	3	0	0	0	2	0	0	0	0	0	0	3	0	0	0	0	1	0	0	0	0	0	52	0	9	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	70	0	0	0	0	0	0
25	0	3	1	0	0	1	1	0	0	0	0	1	3	0	4	0	0	1	0	0	0	0	0	5	0	50	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	70	0	0	0	0
27	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	68	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	70	0	0
29	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	67	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	70

Fig 12. Confusion Metrics of Proposed Model

6. COMPARATIVE STUDY

From the literature survey, it is seen that most of the papers[1][2] consider accuracy as the primary metric for an imbalanced malware dataset like the Malimg dataset. Accuracy is not the overly appropriate metric with reference to imbalanced classification resulting in model evaluation biased to the majority classes. So we have produced a model and run on the dataset which produces better overall results in precision, f1 score and recall. Precision is the most important metric for a malware detector because false positives turn out to be more expensive than false negatives. This paper also contributes a blended dataset.



	ACCURACY	PRECISION	RECALL	F1 SCORE
Zhongru Ren et al. [5], 2020	0.95	0.95	0.95	0.95
BASE MODEL	0.98	0.95	0.93	0.94
PROPOSED MODEL	0.97	0.96	0.96	0.96

Table 4. Comparative Study analysis

7. CONCLUSION & FUTURE SCOPES

Our dataset is created by the data blending of the Malimg dataset and the Malevis dataset. The newly prepared dataset has an intermediate class imbalance compared to the two parent datasets. The new dataset proposed hence considered to be the novelty of the paper. Also we are able to train a better model than the existing pre trained model. Finally, a maximum precision of 95% is obtained for the imbalanced dataset, a maximum precision of 97% is obtained for the intermediate imbalance dataset, and a maximum precision of 96% is obtained for the perfectly balanced dataset by Ensemble Learning Model. On the other hand, VGG16 were sensitive to class imbalance dataset and perform immensely well with balanced dataset. Our model is able to detect malware in case of zero day attack and will also get better once trained with BIG 2015.

In Future:

- **Use of Hyperparameter:** Our future scope is to use hyperparameters in ensemble learning to improve the accuracy. By optimizing these hyperparameters, the overall accuracy and robustness of the ensemble model for malware classification can be significantly improved.
- **Use of BIG2015 Microsoft Dataset:** The previous models are trained on BIG and classified dataset i.e BIG 2015 Microsoft dataset and Dumpware10 dataset which can be done using our ensembling model to enhance its accuracy and train it well.
- **Use averaging ensembling technique:** Enhance the model with weighted variance instead of averaging ensembling technique.

.

REFERENCES

[1] Chaganti, Rajasekhar, Vinayakumar Ravi, and Tuan D. Pham. "Deep learning based cross architecture internet of things malware detection and classification." Computers & Security 120 (2022): 102779.

[2] Ren, Zhongru, et al. "End-to-end malware detection for android IoT devices using deep learning." Ad Hoc Networks 101 (2020): 102098.

[3] Ullah, Farhan, et al. "Cyber security threats detection in internet of things using deep learning approach." IEEE access 7 (2019): 124379-124389.

[4] Khan, Saddam Hussain, et al. "A new deep boosted CNN and ensemble learning based IoT malware detection." Computers & Security 133 (2023): 103385.

[5] Abusitta, Adel, et al. "Deep learning-enabled anomaly detection for IoT systems." Internet of Things 21 (2023): 100656.

[6] Vinayakumar, R., et al. "Robust intelligent malware detection using deep learning." IEEE access 7 (2019): 46717-46738.

[7] Raju, Anandharaju Durai, et al. "A survey on cross-architectural IoT malware threat hunting." IEEE Access 9 (2021): 91686-91709.

[8] Aslan, Ömer, and Abdullah Asim Yilmaz. "A new malware classification framework based on deep learning algorithms." Ieee Access 9 (2021): 87936-87951.

[9] Chaganti, Rajasekhar, Vinayakumar Ravi, and Tuan D. Pham. "A multi-view feature fusion approach for effective malware classification using Deep Learning." Journal of Information Security and Applications 72 (2023): 103402.

[10] Elayan, Omar N., and Ahmad M. Mustafa. "Android malware detection using deep learning." Procedia Computer Science 184 (2021): 847-852.

[11] https://www.kaggle.com/datasets/sohamkumar1703/malevis-dataset

[12] https://www.researchgate.net/figure/Data-description-of-Malimg-Dataset_tbl4_3586 6 4952