ResNet50MalClassifier: Deep Convolutional Neural Networks

Sridevi, *Member, IAENG*, Tukkappa K Gundoor, *Member, IAENG*

***Abstract*—Cybersecurity experts continue to struggle with correctly identifying and classifying harmful malware. This research offers a great deal of promise to advance malware detection and preventative cyber security measures, thereby significantly advancing the entire endeavour to safeguard digital systems and networks against evolving threats. After extensive testing and evaluation of a sizable malware dataset, ResNetMalClassifier exhibits outstanding accuracy, resilience and efficiency in its classification abilities. Astonishingly, using deep convolutional neural networks (CNNs), the classifier achieves a 91% accuracy without ResNet50 and a 95% accuracy with it, outperforming models like Xception 83%, InceptionResNetV2 89%, DenseNet 93%, CNN 91%, VGG16 91% and EfficientNet 91%.**

***Index Terms*—Classification, Cybersecurity, Deep Learning, Malware Detection, Multi-class, Neural Networks, ResNet50.**

I. INTRODUCTION

In the rapidly evolving landscape of cybersecurity, the accurate identifying and classifying malware strains is crucial to maintaining the security and reliability of digital systems. The ResNetMalClassifier, a novel approach based on the ResNet50 architecture, leverages the inherent strengths of deep convolutional neural networks (CNNs) to improve multi-class malware classification efficiency and accuracy. The performance of the classifier has been rigorously evaluated and benchmarked against several established CNN architectures, including Xception, InceptionResNetV2, DenseNet, VGG16, and EfficientNet.

This research offers a comprehensive analysis of the classifier's design, its training process and the granular components that support its outstanding performance. The focus is on accurately classifying various malware families, calculating model-specific accuracies, and thoroughly analyzing experimental results, with particular emphasis on the factors driving the superior performance of ResNet50.

II. RELATED WORK

Malware classification plays a pivotal role in cyberscurity, gaining significant attention due to the increasing complexity of malware. Various machine learning methods have been used to tackle this problem. Conventional approaches like Random forests with support vector machines (SVMs) are frequently utilized for feature creation and classification. However, as malware evolves, more sophisticated approaches are necessary.

In recent years, transfer learning and pre-trained models have advanced malware classification. Architectures for deep learning specifically CNNs (convolutional neural networks) have shown significant potential. ResNet50, known for its capability to recognize images [1], is among the architectures applied to this field. Similar architectures such as VGG16 [2], Xception [3], InceptionResNetV2 [4], DenseNet [5], and EfficientNet [6] have demonstrated their image categorization capabilities. The effectiveness of CNNs in various image classification tasks has paved the way for their adoption in malware analysis, achieving competitive results across datasets. For instance, a CNN-based malware detection system demonstrated an accuracy of 84% [7]-[9], Liu et al. (2021) applied Xception and VGG16, achieving accuracy rates of 83% and 89%, respectively [10]. Further advancements include research into DenseNet models, which achieved 93% accuracy in 2020, and the application of InceptionResNetV2, which reached 89% accuracy in 2019. As malware classification techniques evolve, researchers continue to explore novel methods to enhance accuracy and resilience.[11],[12]. Tanaka (2023) introduced a hybrid model combining CNNs and Graph Convolutional Networks (GCNs), achieving 92% accuracy [13], Kim and Park (2023) implemented a self-attention mechanism within CNN architecture, reaching a 90% accuracy rate [14].

The change from conventional machine learning models to deep learning and hybrid architectures emphasizes the need for ongoing innovation to stay up-to-date with the changing malware classification field.

III. PROPOSED METHODOLOGY

The present work describes and evaluates the ResNet50-MalClassifier, a unique technique to multiclass malware classification that use deep CNNs. The primary goal is to address the difficulty of effectively identifying and categorizing harmful software (malware) by benchmarking the ResNet50 architecture against other existing models. The goal is to demonstrate that ResNet50 is effective, reliable, and efficient in this domain. The proposed technique starts with collecting malware samples, which are then pre-processed into visual representations. These visual representations are created by converting the malware sample's byte frequencies into grayscale pixel values. To address vanishing gradient concerns, a modified ResNet50 architecture is used, which includes multiple convolutional layers and residual connections as shown in Figure 1.

A. Model Evaluation

In malware image class, the code calculates the class distribution of the dataset. Suppose C classes, where C is the number of malware types. Equation (1) represents the class distribution as a mathematical function:

$$\text{ClassDistribution}(C) = \{(c1, n1), (c2, n2)., (cC, nC)\} \quad (1)$$

Where:

- c_i indicates the name of the i^{th} malware class.

Manuscript received January 1, 2024; revised November 13, 2024.

The DST (Department of Science and Technology of Karnataka) of the Indian government provided support for this research under Grant No. DST/KSTePS/Ph.D. Fellowship/PHY-02:2020-21/199.

Dr. Sridevi is a Professor of Karnatak University, Dharwad, Karnataka-580003, India. (email: sridevi@kud.ac.in).

Tukkappa K Gundoor is a research scholar of Karnatak University Dharwad, Karnataka-580003, India, (corresponding author- phone: +91-7847874715 email: tukkappa@kud.ac.in.).

- n_i represents the count of images in the i^{th} class.

The split_dataset function splits the dataset according to predetermined ratios into training and testing subsets. Equation (2) represents this operation as a function:

$$\text{SplitDataset}\left(\text{train}_{\text{data}_{\text{dir}}}, \text{dest}_{\text{path}}, \text{train}_{\text{ratio}}, \text{test}_{\text{ratio}}\right) = \begin{cases} \text{Training Data} & \text{if U}(0,1) \leq \text{train\_ratio} \\ \text{Testing Data} & \text{if train\_ratio} < U(0,1) \leq \text{Train\_ration} + \text{test\_ratio} \end{cases} \quad (2)$$

Where U (0,1) A random number between 0 and 1 is shown.

The Image processor class preprocesses images using data generators. Equation (3) represents this as a transformation function:

$$\text{Preprocess(I)}=I' \quad (3)$$

Where:

- I represent the input image.
- I′ represent the preprocessed image.

In the Fine-Tuning class, the code fine-tunes a pre-trained model by marking layers as trainable or not and adding final layers. Equation (4) represents this as a transformation function:

$$\text{FineTune(M)}=M' \quad (4)$$

Where:

- M represents an already-trained model.
- M' represents for the refined model.

The suggested ResNet50MalClassifier model approach was assessed using standard assessment measures. Due to the fact that the tested datasets were balanced, these measurements were F1-score, recall, accuracy, and precision, the calculation of these metrics is demonstrated by equations (5) to (8). Accuracy is the percentage of correctly identified malwares in the model to all inputs, whereas F1-score primarily takes into account both Precision and Recall values.

$$\text{Precision}(C_i) = \frac{\text{True Positives } (C_i)}{\text{True Positives } (C_i) + \text{False Positives } (C_i)} \quad (5)$$

$$\text{Recall}(C_i) = \frac{\text{True Positives } (C_i)}{\text{True Positives } (C_i) + \text{False Negatives } (C_i)} \quad (6)$$

$$\text{F1} - \text{score}(C_i) = 2 \times \frac{\text{Precision}(C_i) \times \text{Recall}(C_i)}{\text{Precision}(C_i) + \text{Recall}(C_i)} \quad (7)$$

$$\text{Accuracy}(C_i) = \frac{\text{TP}(C_i) \times \text{TN}(C_i)}{\text{TP}(C_i) + \text{FP}(C_i) + \text{FN}(C_i) + \text{TN}(C_i)} \quad (8)$$

Ci indicates the ith class. The model's predictions and ground truth are used to determine the percentage of false positives, false negatives, and true positives.

IV. EXPERIMENTAL RESULTS

A. Dataset Exploration

The collection contains 10678 malware images, which are divided into 36 separate malware families and are displayed in Figures 2 respectively. The dataset's greatest sample count for malware is 2,359, while the lowest sample count for the class is 80. A malware picture dataset that has been pre-processed with 10678 samples from 36 different malware families was utilised to evaluate the proposed architecture, ResNetMalClassifier. Each malware family has over 90 training samples and over 12 validation samples are used for each class. After the data was pre-processed by the models, it was divided into 8,537 training samples, 3965 validation samples, and 2141 testing samples. Table I has a detailed data set.

B. CNN without ResNet50MalClassifire

Advanced Convolutional Neural Networks (CNNs) can be adapted for malware picture categorization using the fine-tuning class. The model has to be fine-tuned, modified, and trained for consistency. The class accepts several inputs, such as a pre-trained CNN model, training and validation data generators, class count, regularisation flag, and patience for early stopping. Up to a predetermined point, the fine-tuning process involves systematically freezing earlier layers while allowing later layers to be trainable. The 'add_final_layer' method adds a final layer, maybe with regularization. Using the 'compile_model' method, the model is configured with key components, including metrics, an optimizer, and a loss function. Early stopping is integrated into the model using the 'train_model' function. Following preparation and dataset separation, CNNs were successfully modified for malware classification [15].

TABLE II
MODEL SEQUENTIALS OF CNN

Layer (type)	Output shape	Param
Conv2d 10, (Conv2D)	(None, 75, 75, 32)	896
Batch normalization 10 (Batch Normalization)	(None, 75, 75, 32)	128
Conv2d 11, (Conv2D)	(None, 75, 75, 32)	9248
batch_normalization_11 (Batch Normalization)	(None, 75, 75, 32)	128
Max pooling 2d 4 (MaxPooling 2D)	(None, 37, 37, 32)	0
Dropout 5 (Dropout)	(None, 37, 37, 32)	0
Conv2d 12 (Conv2D)	(None, 37, 37, 32)	9248
Batch normalization 12 (Batch Normalization)	(None, 37, 37, 32)	128
Conv2d_13 (Conv2D)	(None, 37, 37, 32)	9248
Total params: 24,978,928		
Trainable params: 24,978,608		
Non-trainable params: 320		

Table II shows the details of the layers below:

1). Conv2D Layer (conv2d_10), with 32 filters/kernels. This layer handles input data of size 75x75 pixels with 32 feature maps, as indicated by the output shape of (None, 75, 75, 32). the weights and biases that govern the convolution process make up its 896 parameters.

2). Batch Normalization Layer (batch_normalization_10): The activations from the previous convolutional layer are normalized in this layer, accelerating and stabilizing the training process. The output shape remains (None, 75, 75, 32), with 128 parameters.

3). Conv2D Layer (conv2d_11): The batch-normalization layer is followed by a convolutional layer with 32 filters. Similar to the preceding layer, it also processes the data, but with 9,248 additional parameters.

4). Batch Normalization Layer (batch_normalization_11): Following the second convolutional layer comes a second batch normalization layer.

5). MaxPooling2D Layer (max_pooling2d_4): This layer reduces the spatial dimensions of the data using max-pooling, and the output shape is reduced to (None, 37, 37, 32).

6). Dropout Layer (dropout_5): A regularization technique called dropout helps Randomly adjusting certain parts of the input units to 0 while training will help avoid overfitting. It is applied without the shape being diminished.

7). Conv2D Layers (conv2D_12 and conv2D_13): After the MaxPooling2D layer, these additional convolutional layers each contain 9,248 parameters.

The model's total number of parameters is 24,978,928 according to the Total params section. Parameters that remain constant throughout training are known as non-trainable parameters (typically as a result of batch normalization or other fixed layers). The factors that vary throughout training are known as trainable parameters. It appears that this architecture is a component of a bigger CNN for image classification or similar tasks [16], [17].

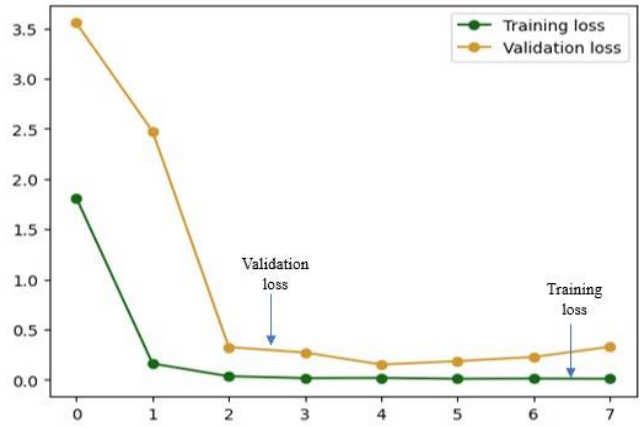

Fig 3. Loss of Validation and Training

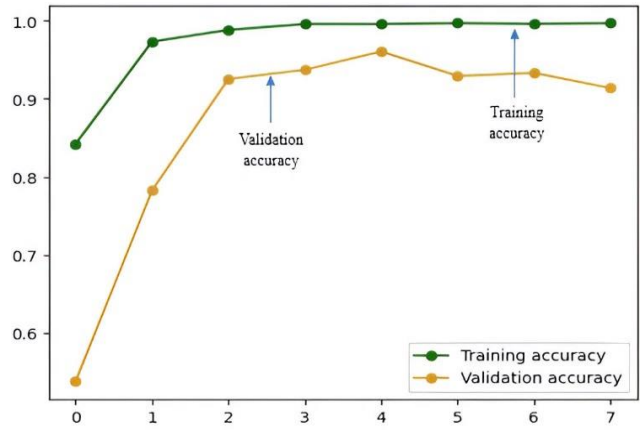

Fig 4. Accuracy of Training and Validation

Loss of Validation and Training often decrease during the initial training process as the model becomes more adept at identifying patterns in the data. To illustrate this, Figure 3 demonstrates that if the model trains for an excessively long time, Loss of validation may start to increase while the loss of training is decreasing. This indicates overfitting, when the model loses its capacities to generalize to newly collected datasets when it becomes too focused on the training set. The CNN ResNet50MalClassifier in Figure 4, uses this method for diagnosing model performance. A comparison of training and validation accuracy is essential. Overfitting is evident if training accuracy is significantly higher than validation accuracy. Based on the training set of data, the model might have picked up noise or particular details that aren't generalizable. If the model is underfitting to adequately represent the complexity of the data [18].

In Figure 5, out of the 36 malware families, the confusion matrix illustrates the performance across different malware families, showing high accuracy in 13 families lower accuracy in 23 families were classified less accurately. Overall, the malware classification without the ResNet50M-alClassifier model had a loss of 70.33% and a classification accuracy of 91.33%.

C. Exploration of CNN with the ResNet50MalClassifier model

There is hardly any change in weight during backpropagation because the gradient's value drops drastically which is the problem of the disappearing gradient. ResNet50 uses skip connections, which bypass training for the first few layers before connecting to the output. The network skips certain convolutional layers that may degrade performance. The gradient vanishing problem is avoided as the network penetrates deeper. Two forms of skip connections are depicted in methodology Figure 1. A convolutional block and an identity block. The identity block directly adds residuals to the output, while the convolutional block modifies residuals before applying batch normalization to it before including it in the output. 36 malware families were tested to determine the accuracy of the malware classification in the suggested method. Each layer of this approach has its own malware families to classify, improving categorization for each family. ResNet50 makes use of the skip connection to improve performance. [19]-[21]. In Figure 6, 14 malware families were classified accurately and 22 families were classified less accurately.

Table III and Figure 7 displays performance metrics for the categorization task for different malware families. The "Support" column displays a number of instances for every class. All classes' metrics are summarized in the Macro average and Weighted avg rows. with a weighted average accuracy of 0.95 for the proposed approach and a macro F1-Score of 0.96.[25].

Important information about the effectiveness of machine learning models can be found in Figures 8 to 13 shown below. Figure 8, illustrates the value of monitoring training loss as well as validation loss to identify overfitting or model convergence. The relationship between training and validation accuracy is highlighted in Figures 9 and 10, which also measure model generalization and potential overfitting. In order to evaluate a model, Figure 11, focuses on false positives, and Figure 12, emphasises precision, which is essential for classification tasks. The capabilities of the model to accurately detect positive elements during vali-

dation is finally examined in Figure 13, which examines the link between recall and validation recall. Collectively,

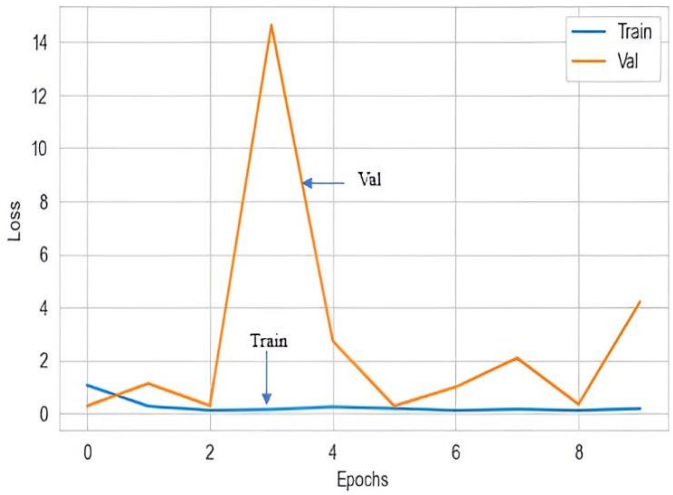

Fig 8. Loss v/s Val_loss

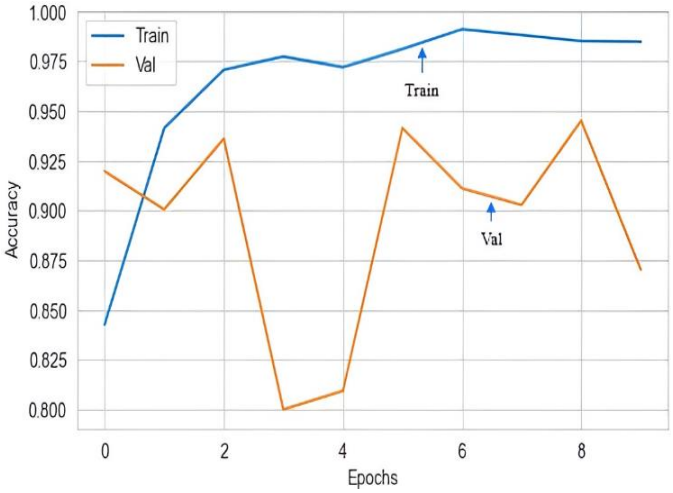

Fig 9. Accuracy v/s Val_accuracy

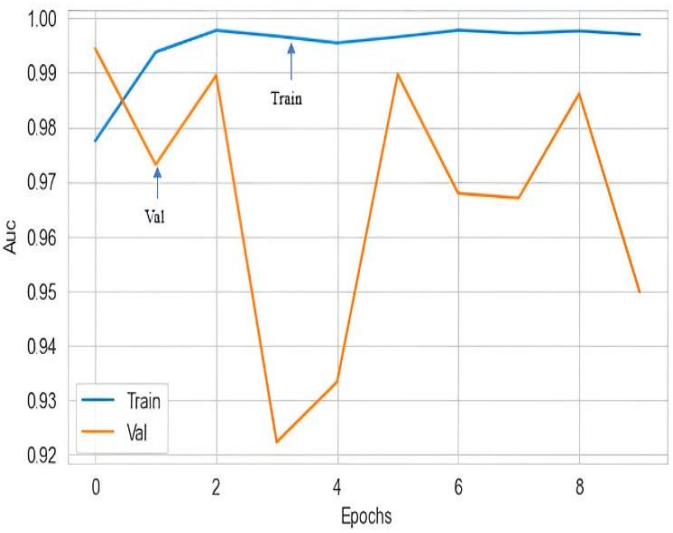

Fig 10. Accuracy v/s AUC

these visualizations support the evaluation and improvement of model.

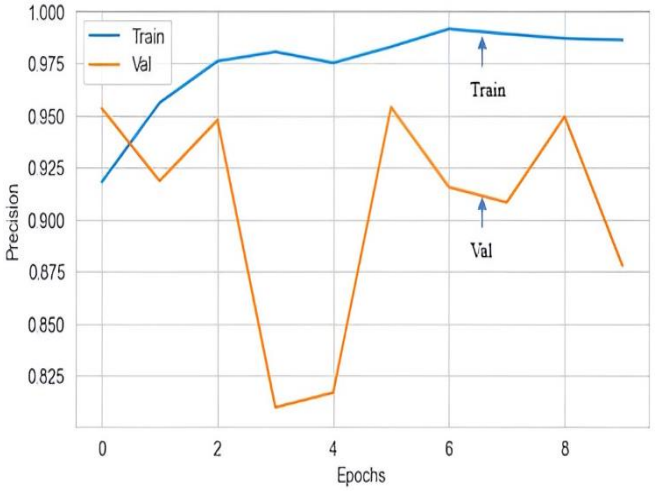

Fig 11. Precision v/s Val_precision

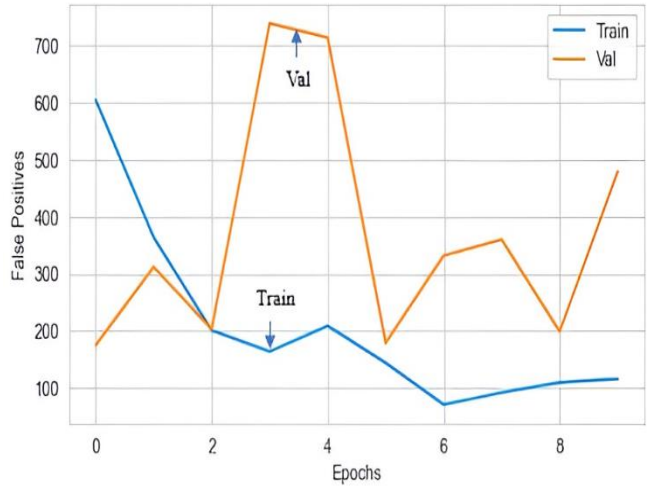

Fig 12. False_positive v/s Val_false_positive

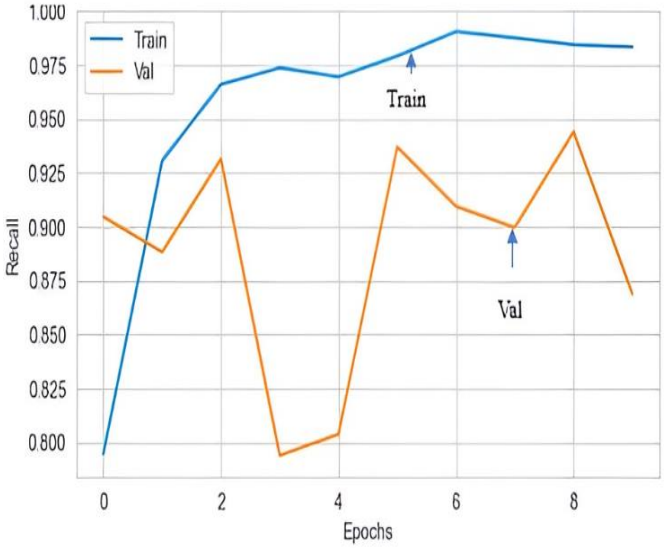

Fig 13. Recall v/s Val_recall

Figure 14-19 collectively assesses and compares the performance of six machine learning models (DensNet.[26], EfficientNet [27], InceptionResNet [28], ResNet50MalClassifier [29], VGG16 [30], XceptionNet [31]) using various metrics. Figure 14, examines validation loss to gauge model generalization. Figure 15 evaluates the performance of models on unknown data by concentrating on validation accuracy. Figure 16, measures the Area Under the ROC curve to evaluate class separation. Figure 17, analyzes false positives to assess error rates. Figure 18, evaluates precision, and Figure 19, assesses recall, both crucial for classification tasks. These visualizations provide insights into the model's

relative strengths and weaknesses. Performance measures consist of F1-score, recall, accuracy, and precision among others.

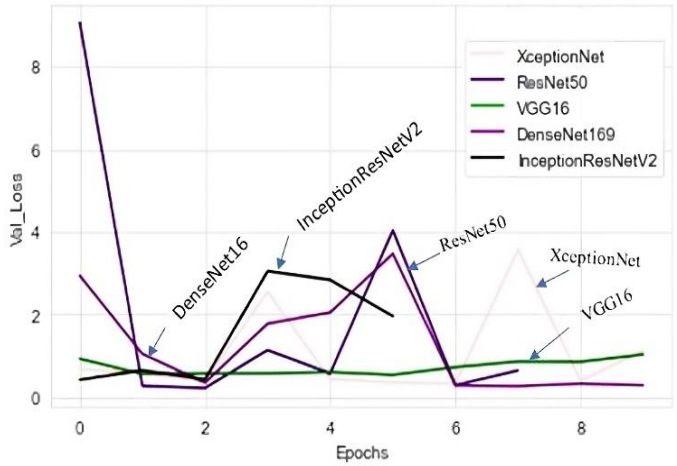

Fig 14. Val_Loss Comparison

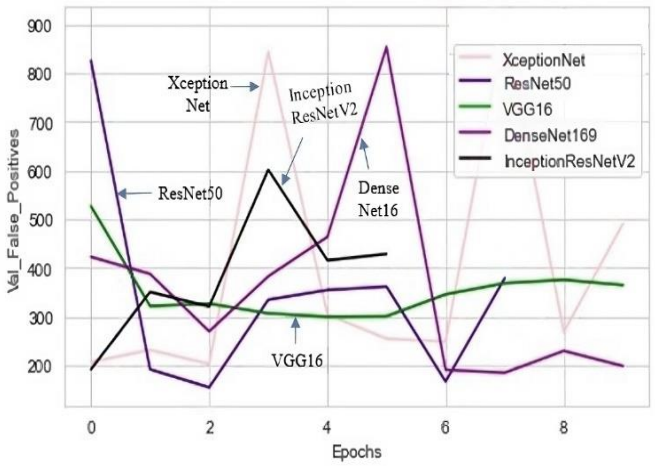

Fig 17. Val_false_positives Comparison

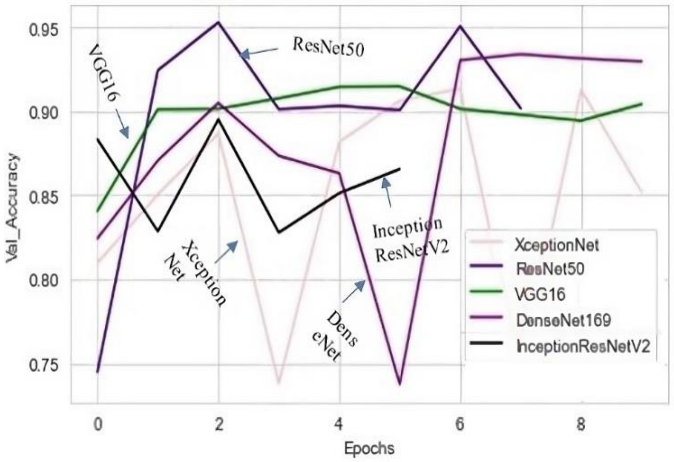

Fig 15. Val_accuracy Comparison

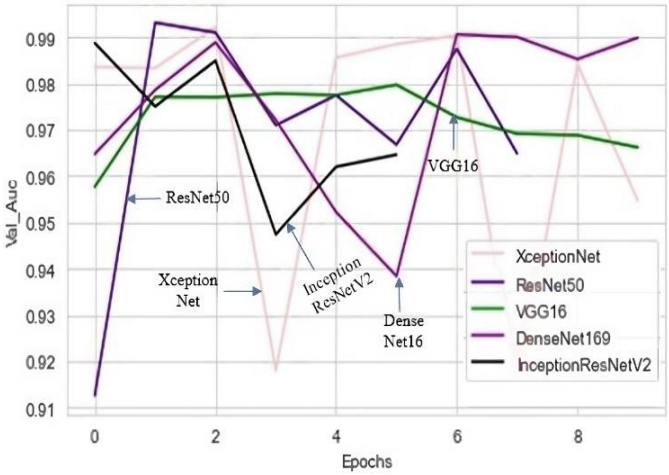

Fig 18. Val_precision Comparison

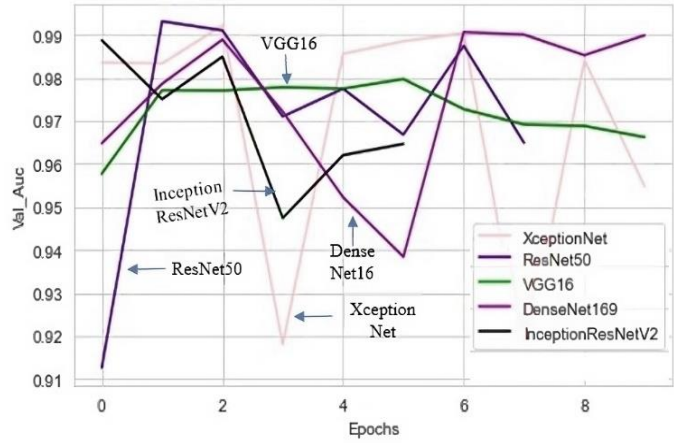

Fig 16. Val_Auc Comparison

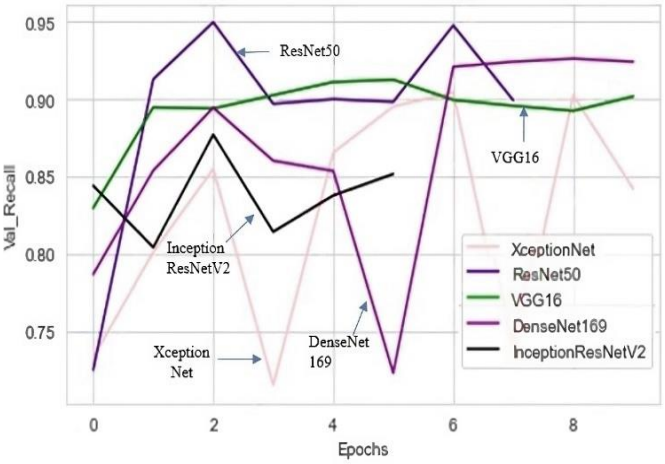

Fig 19. Val_recall Comparison

The performance characteristics of various machine learning models (DenseNet, EfficientNet, InceptionResNet, ResNet50MalClassifier, VGG16, XceptionNet) in identifying various malware families shown in Table IV. The support column lists the number of instances for each malware family, representing the data distribution. All models attain perfect precision (100%), indicating accurate predictions. Figure 20, also shows the overall accuracy for each model and comparison of datasets with different CNN models tested accuracy. When compared to other models, models like DensNet and ResNet50MalClassifier are more accurate overall (0.93 and 0.95, respectively) [32] [33] [34] [35] [36]. Among the models, ResNet50MalClassifier and DenseNet were the most accurate, with overall accuracies of 95% and 93%, respectively, while XceptionNet lagged behind with 83%.

V. RESULTS

CNN+ResNet50MalClassifier achieved the greatest accuracy score of 95%. The F1-Score balances precision (The percent of actual positives between expected positives)

and recall (the percent of actual positives within real positives). CNN+ResNet50MalClassifier once again performs admirably with 95%. The model's capacity to identify genuine positives and reduce false positives that is highlighted, separately, by recall and precision. The highest recall (95%) and precision (96%) were achieved by CNN+ResNet50MalClassifier. A model's selection is determined by certain trade-offs, with CNN+ResNet50MalClassifier providing overall good performance shown in the Table V [37] [38][39][40].

TABLE V
CLASSIFICATION OF METRICS ACROSS DIFFERENT MODELS

Machine Learning models	Accuracy	F1-Score	Recall	Precision
CNN without ResNet50MalClassifier	91.33%	91%	91%	92%
CNN+DensNet169	93%	92%	93%	93%
CNN+EfficientNetB0	91%	91%	91%	92%
CNN+InceptionResNetV2	89%	89%	89%	90%
CNN+ResNet50MalClassifier	95%	95%	95%	96%
CNN+VGG16	91%	91%	91%	92%
CNN+XceptionNet	83%	83%	83%	86%

VI. CONCLUSION

Malware identification and categorization are crucial areas in the cybersecurity field, and this research has demonstrated considerable gains. The ResNetMalClassifier, which makes use of the ResNet50 architecture, has demonstrated exceptional performance, surpassing several well-known models, including Xception, InceptionResNetV2, DenseNet, CNN, VGG16, and EfficientNet, with a remarkable accuracy rate of 95%. This unique method shows significant promise for boosting preventative cybersecurity measures and malware detection effectiveness.

REFERENCES

[1] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[2] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp770–778).

[3] Chollet, F. (2017). Xception: Deep learning with depth wise separable convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp1251–1258).

[4] Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. Proceedings of the AAAI Conference on Artificial Intelligence (AAAI) (Vol. 4, No. 1).

[5] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected to convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp4700–4708).

[6] Zhang, J., Zhang, Z., Jang, J. H., & Giles, C. L. (2018). Malware image classification with deep convolutional neural networks. In Proceedings of the ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL) (pp109–112).

[7] Liu, Y., Zhang, W., Hou, Y., Xie, M., & Li, F. (2019). Malware detection by eating a whole exe. In Proceedings of the 35th Annual Computer Security Applications Conference (ACSAC) (pp178–188).

[8] Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning (ICML) (pp6105–6114).

[9] Guo Liu, Qiang Zhao, and Guiding Gu, "A Simple Control Variate Method for Options Pricing with Stochastic Volatility Models," IAENG International Journal of Applied Mathematics, vol. 45, no.1, pp64-70, 2015.

[10] Zheng, Q. et al. (2020). "Deep Learning for Malware Classification and Analysis." IEEE Transactions on Dependable and Secure Computing, 17(5), 1056-1069.

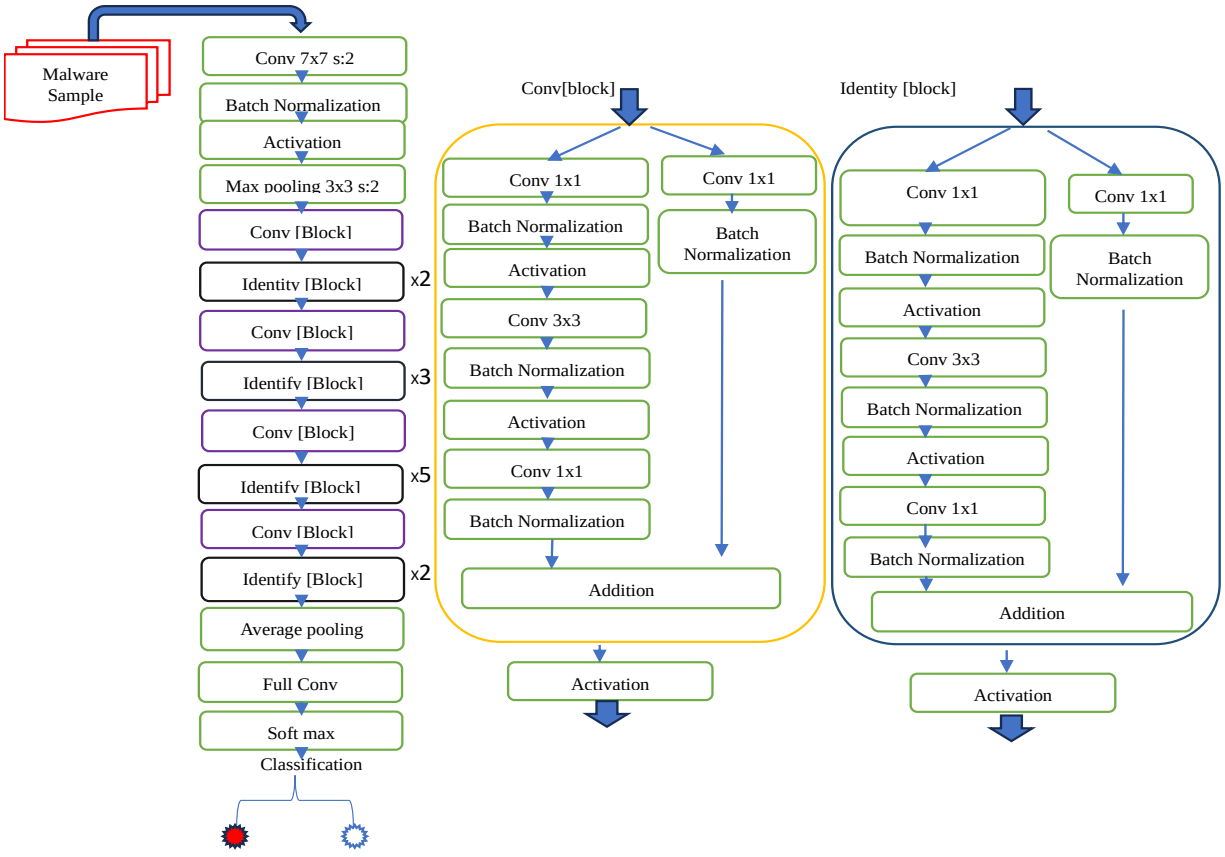

Fig 1. Proposed ResNet50MalClassifier Architecture

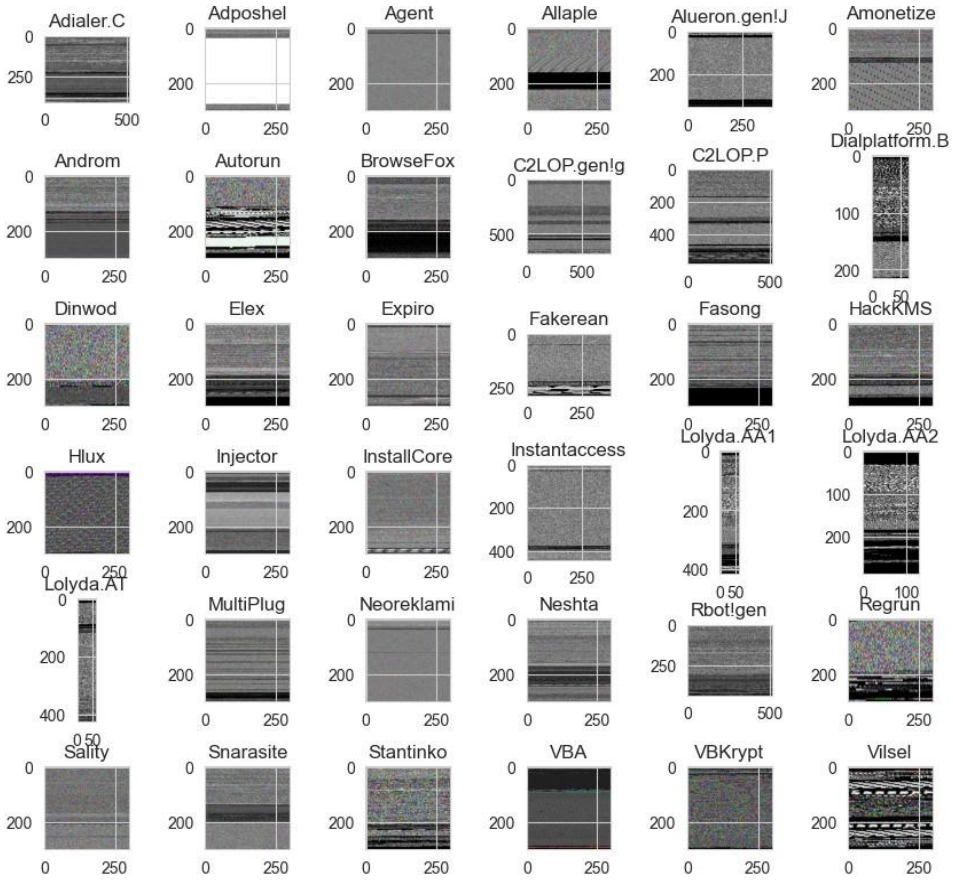

Fig 2. Sample for Malware Types

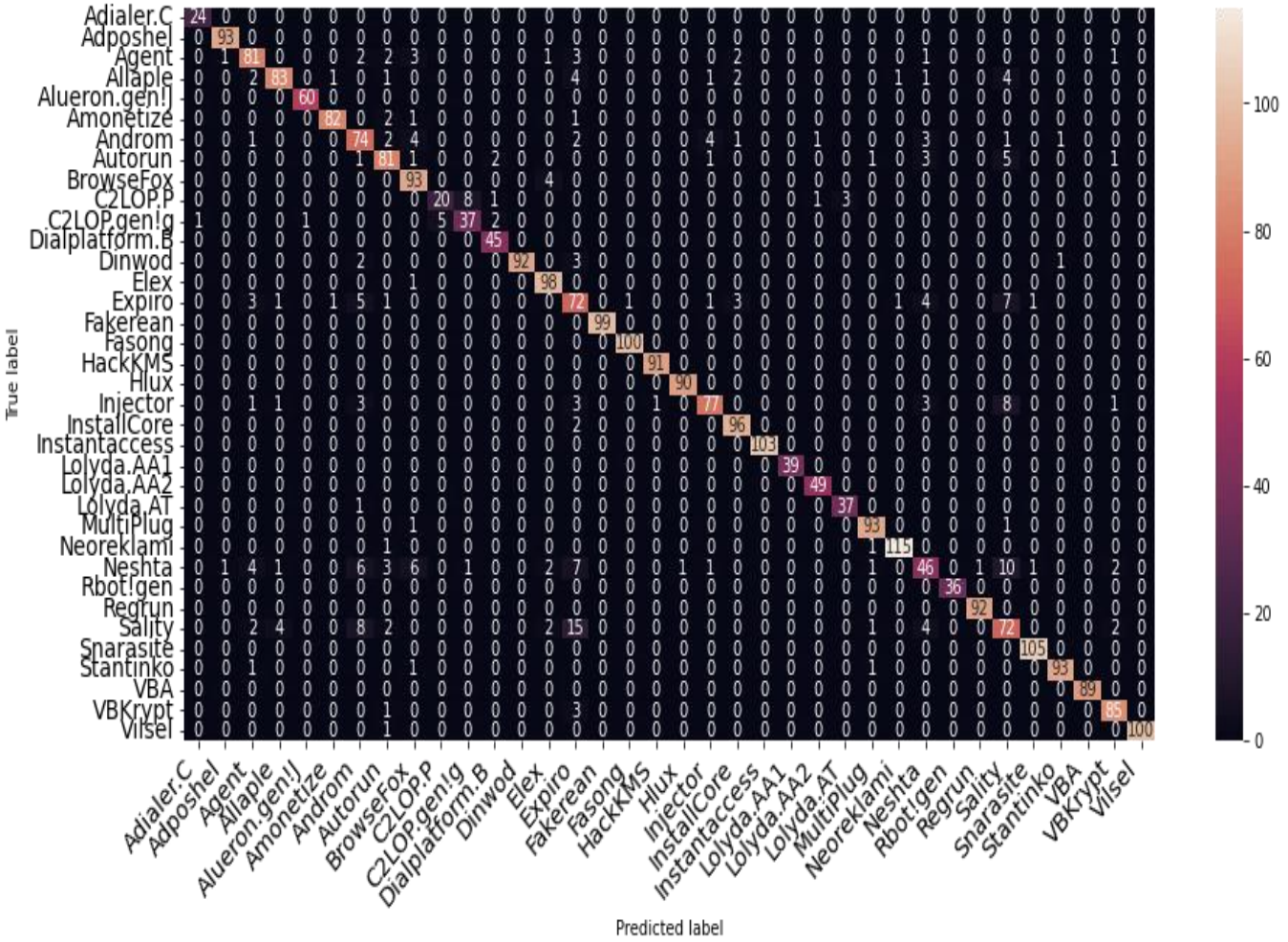

Fig 5. Confusion Matrix for CNN without ResNet50MalClassifier.

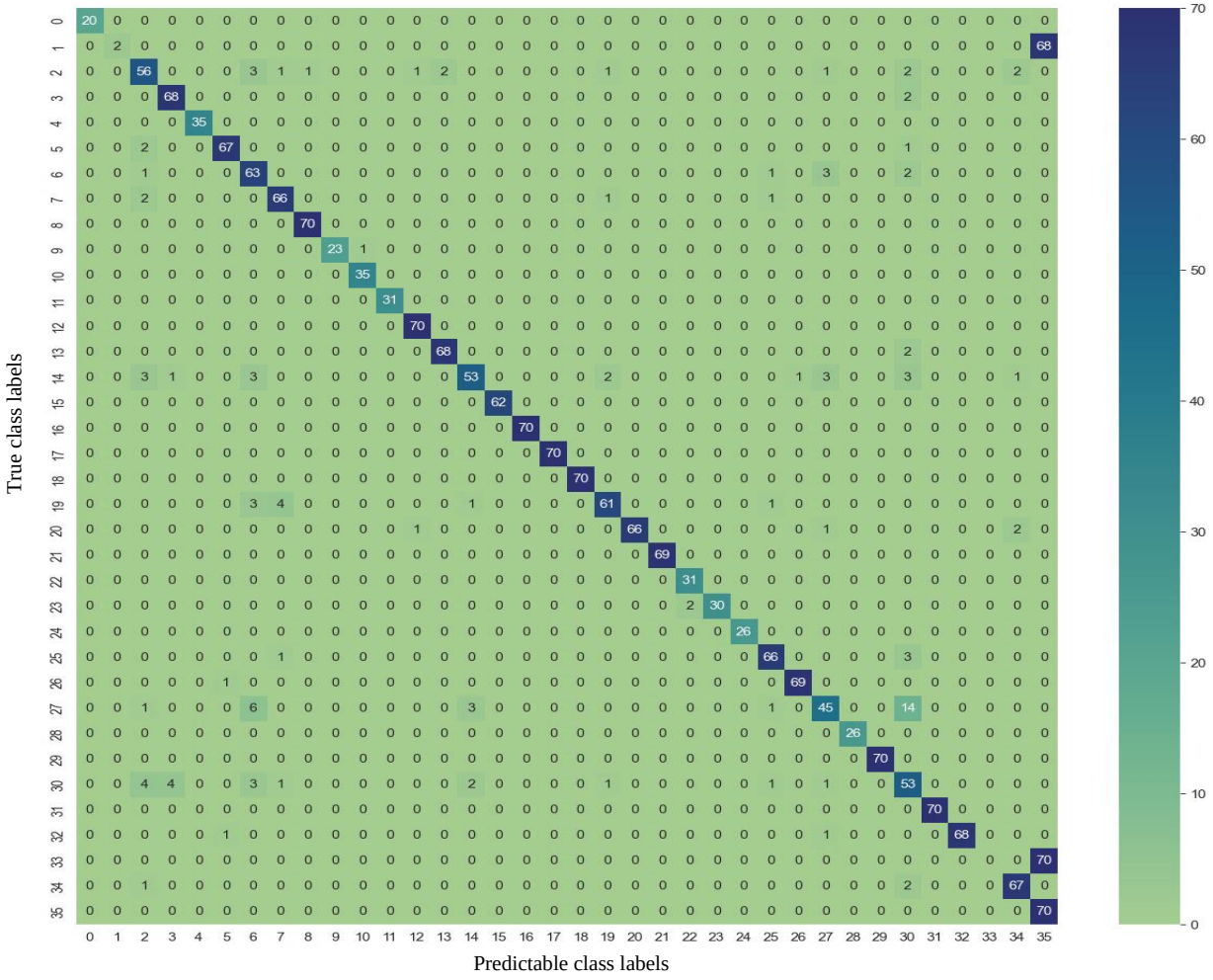

Fig 6. Confusion Matrix for CNN with ResNet50MalClassifier

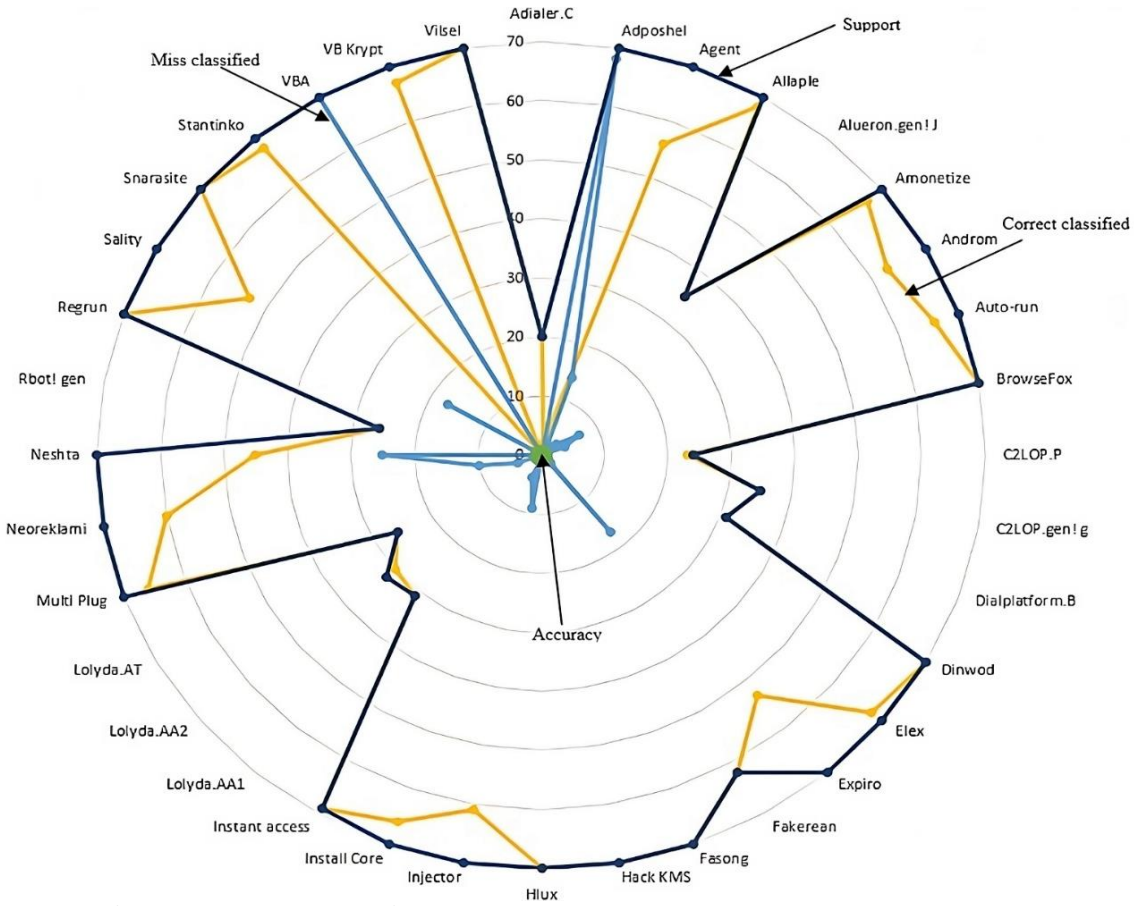

Fig 7. Evaluation report of CNN with ResNet50MalClassifier

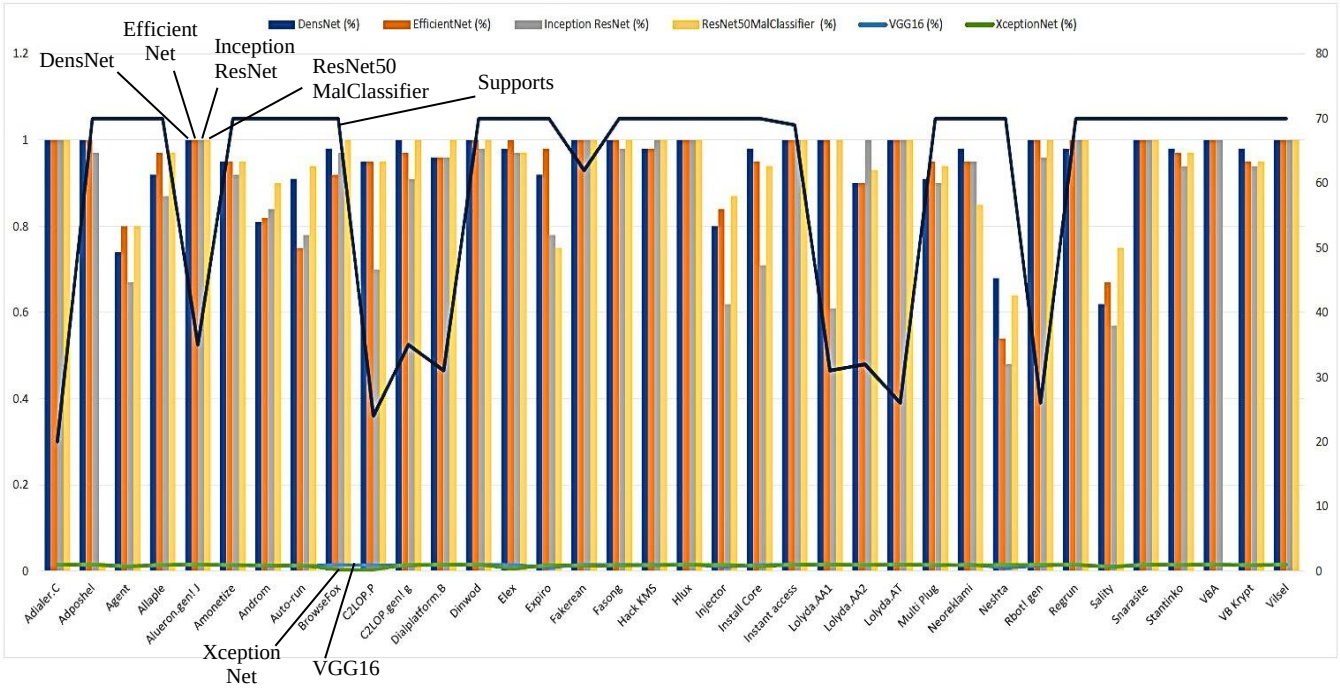

Fig 20. Comparison of datasets with different CNN models tested accuracy

TABLE I
DETAILS OF PRE-PROCESSED DATASET.

Sl. No	Class	Before Pre-processing Training	Before Pre-processing Testing	After Pre-processing Training	After Pre-processing Testing
1.	Adialer.C	97	12	77	20
2.	Adposhel	350	144	280	70
3.	Agent	350	120	280	70
4.	Allaple	350	128	280	70
5.	Alueron.gen! J	173	25	138	35
6.	Amonetize	350	147	280	70
7.	Androm	350	150	280	70
8.	Autorun	350	146	280	70
9.	BrowseFox	350	143	280	70
10	C2LOP gen! g	175	25	140	35
11	C2LOP P	116	14	92	24
12	Dialplatform B	152	25	121	31
13	Dinwod	350	149	280	70
14	Elex	350	150	280	70
15	Expiro	350	151	280	70
16	Fakerean	306	69	244	62
17	Fasong	350	150	280	70
18	HackKMS	350	149	280	70
19	Hlux	350	150	280	70
20	Injector	350	145	280	70
21	InstallCore	350	150	280	70
22	Instantaccess	344	43	275	69
23	Lolyda.AA1	153	54	122	31
24	Lolyda.AA2	159	24	127	32
25	Lolyda.AT	127	15	101	26
26	MultiPlug	350	149	280	70
27	Neoreklami	350	150	280	70
28	Neshta	350	147	280	70
29	Rbot!gen	126	15	100	26
30	Regrun	350	135	280	70
31	Sality	350	149	280	70
32	Snarasite	350	150	280	70
33	Stantinko	350	150	280	70
34	VBA	350	150	280	70
35	VBKrypt	350	146	280	70
36	Vilsel	350	146	280	70

TABLE III
EVALUATION REPORT FOR CNN WITH RESNET50MALCLASSIFIRE

Sl. No	Malware Families	Precision	Recall	F1-Score	Correct Classified	Miss Classified	Accuracy	Support
0.	Adialer.C	1.00	1.00	1.00	20	0	1.00	20
1.	Adposhel	1.00	1.00	1.00	2	68	0.02	70
2.	Agent	0.75	0.87	0.81	56	14	0.80	70

3.	Allaple	0.96	0.94	0.95	68	02	0.97	70
4.	Alueron.gen!	1.00	1.00	1.00	35	00	1.00	35
5.	Amonetize	0.99	0.96	0.97	67	03	0.95	70
6.	Androm	0.98	0.80	0.88	63	07	0.90	70
7.	C2LOP.gen! g	0.97	0.94	0.99	35	00	1.00	35
8.	Dialplatform.B	1.00	1.00	1.00	31	00	1.00	31
9.	Dinwod	0.99	0.96	0.99	70	00	1.00	70
10.	Elex	0.96	0.80	0.97	68	02	0.97	70
11.	Expiro	0.90	0.94	0.88	53	17	0.75	70
12.	Fakerean	1.00	1.00	1.00	62	00	1.00	62
13.	Fasong	1.00	0.96	1.00	70	00	1.00	70
14.	Hack KMS	1.00	0.80	1.00	70	00	1.00	70
15.	Hlux	1.00	0.94	1.00	70	00	1.00	70
16.	Injector	0.93	1.00	0.92	61	09	0.87	70
17.	Install Core	1.00	0.96	0.99	66	04	0.94	70
18.	Instant access	1.00	0.80	1.00	69	00	1.00	69
19.	Lolyda, A1	1.00	0.94	0.98	31	00	1.00	31
20.	Lolyda, A2	1.00	1.00	1.00	30	00	0.93	32
21.	LolydaAT	1.00	0.96	1.00	26	00	1.00	26
22.	Multi Plug	0.91	0.80	0.95	66	04	0.94	70
23.	Neoreklami	1.00	0.94	1.00	60	10	0.85	70
24.	Neshta	0.73	1.00	0.75	45	25	0.64	70
25.	Rbot! gen	1.00	0.96	1.00	26	00	1.00	26
26.	Regrun	1.00	0.80	0.99	70	00	1.00	70
27.	Sality	0.71	0.94	0.72	53	17	0.75	70
28.	Snarasite	1.00	1.00	1.00	70	00	1.00	70
29.	Stantinko	1.00	0.96	0.99	68	00	0.97	70
30.	VBA	1.00	0.80	1.00	00	70	0.00	70
31.	VB Krypt	1.00	0.94	0.97	67	00	0.95	70
32.	Vilsel	1.00	1.00	1.00	70	00	1.00	70
Overall Accuracy of the proposed method							0.95	2141
Macro average				0.96	0.80	0.96	-	2141
Weighted average				0.96	0.94	0.95	-	2141

TABLE IV
COMPARISON OF DATASETS WITH DIFFERENT CNN MODELS TESTED ACCURACY

Sl. No	Malware Families	DensNet (%)	EfficientNet (%)	Inception ResNet (%)	ResNet50MalClassifier (%)	VGG16 (%)	XceptionNet (%)	Support (%)
0.	Adialer.C	1.00	1.00	1.00	1.00	1.00	1.00	20
1.	Adposhel	1.00	1.00	0.97	0.02	1.00	1.00	70
2.	Agent	0.74	0.8	0.67	0.8	0.77	0.7	70
3.	Allaple	0.92	0.97	0.87	0.97	0.95	0.95	70
4.	Alueron_gen! J	1.00	1.00	1.00	1.00	1.00	1.00	35
5.	Amonetize	0.95	0.95	0.92	0.95	0.95	0.9	70
6.	Androm	0.81	0.82	0.84	0.9	0.82	0.9	70
7.	Auto-run	0.91	0.75	0.78	0.94	0.78	0.84	70
8.	BrowseFox	0.98	0.92	0.97	1.00	0.97	0.24	70
9.	C2LOP.P	0.95	0.95	0.70	0.95	0.87	0.24	24
10.	C2LOP_gen! g	1.00	0.97	0.91	1.00	0.88	1.00	35
11.	Dialplatform.B	0.96	0.96	0.96	1.00	0.96	1.00	31
12.	Dinwod	1.00	1.00	0.98	1.00	1.00	1.00	70
13.	Elex	0.98	1.00	0.97	0.97	0.97	0.37	70
14.	Expiro	0.92	0.98	0.78	0.75	0.57	0.97	70
15.	Fakerean	1.00	1.00	1.00	1.00	1.00	0.83	62
16.	Fasong	1.00	1.00	0.98	1.00	1.00	0.88	70
17.	Hack KMS	0.98	0.98	1.00	1.00	1.00	0.88	70
18.	Hlux	1.00	1.00	1.00	1.00	1.00	1.00	70
19.	Injector	0.8	0.84	0.62	0.87	0.68	1.00	70
20.	Install Core	0.98	0.95	0.71	0.94	0.97	0.72	70
21.	Instant access	1.00	1.00	1.00	1.00	1.00	0.98	69
22.	Lolyda A1	1.00	1.00	0.61	1.00	1.00	1.00	31
23.	Lolyda A2	0.90	0.90	1.00	0.93	0.93	0.96	32
24.	Lolyda.AT	1.00	1.00	1.00	1.00	1.00	1.00	26
25.	Multi Plug	0.91	0.95	0.9	0.94	0.91	0.97	70
26.	Neoreklami	0.98	0.95	0.95	0.85	1.00	0.91	70
27.	Neshta	0.68	0.54	0.48	0.64	0.51	1.00	70
28.	Rbot! gen	1.00	1.00	0.96	1.00	0.84	1.00	26
29.	Regrun	0.98	1.00	1.00	1.00	0.98	0.98	70
30.	Sality	0.62	0.67	0.57	0.75	0.71	0.61	70
31.	Snarasite	1.00	1.00	1.00	1.00	1.00	1.00	70
32.	Stantinko	0.98	0.97	0.94	0.97	0.95	0.95	70
33.	VBA	1.00	1.00	1.00	0.00	1.00	1.00	70
34.	VB Krypt	0.98	0.95	0.94	0.95	0.91	0.94	70
35.	Vilsel	1.00	1.00	1.00	1.00	1.00	1.00	70
Total Testing Samples								2141
Models Overall Accuracy		0.93	0.91	0.89	0.95	0.91	0.83	--

[11] Wan Zakiyatussariroh Wan Husin, Mohammad Said Zainol, and Norazan Mohamed Ramli, "Common Factor Model with Multiple Trends for Forecasting Short Term Mortality," Engineering Letters, vol. 24, no.1, pp98-105, 2016.

[12] Liu, B. et al. (2021). "Deep Learning for Malware Classification: From Traditional to Modern Neural Networks." Proceedings of the International Conference on Cybersecurity, pp42-55.

[13] Tanaka, K. et al. (2023). "Hybrid Malware Classification using Convolutional and Graph Convolutional Networks." Proceedings of the IEEE International Conference on Cybersecurity and Privacy, pp124-138.

[14] Kim, S. & Park, J. (2023). "Self-Attention Enhanced Convolutional Neural Networks for Malware Classification." Journal of Cybersecurity Research, 8(3), 210-224

[15] L. Ali, F. Alnajjar, H. Al Jassmi, M. Gochoo, W. Khan, and M. A. Serhani, "Performance evaluation of deep CNN-based c rack detection and localization techniques for concrete structures," *Sensors*, vol. 21, no.5, pp1–22, 2021, doi: 10.3390/s21051688.

[16] M. F. Rafique, M. Ali, A. S. Qureshi, A. Khan, and A. M. Mirza, "Malware Classification using Deep Learning based Feature Extraction and Wrapper based Feature Selection Technique," pp 1–21, 2019, [Online]. Available:http://arxiv.org/abs/1910.10958

[17] D. Gibert, "Convolutional Neural Networks for Malware Classification," *Univ. Rovira i Virgili, Tarragona, Spain*, October, 2016.

[18] K. Brezinski and K. Ferens, "Complexity-Based Convolutional Neural Network for Malware Classification," *Proc. - 2020 Int. Conf. Comput. Sci. Comput. Intell. CSCI 2020*, no. May, pp1–9, 2020, doi: 10.1109/CSCI51800.2020.00008.

[19] E. Rezende, G. Ruppert, T. Carvalho, F. Ramos, and P. De Geus, "Malicious software classification using transfer learning of ResNet-50 deep neural network," *Proc. - 16th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2017*, vol. 2017-December, pp 1011–1014, 2017, doi: 10.1109/ICMLA.2017.00-19.

[20] R. U. Khan, X. Zhang, and R. Kumar, "Analysis of ResNet and GoogleNet models for malware detection," *J. Comput. Virol. Hacking Tech.*, vol.15, no.1, pp29–37, 2019, doi: 10.1007/s11416-018-0324-z.

[21] M. D. H. U. Sharif, N. Jiwani, K. Gupta, M. A. L. I. Mohammed, and M. F. Ansari, "a Deep Learning Based Technique for the Classification of Malware Images," *J. Theor. Appl. Inf. Technol.*, vol. 101, no. 3, pp1137–1161, 2023.

[22] Ahmad El-Ajou, Zaid Odibat, Shaher Momani, and Ahmad Alawneh, "Construction of analytical solutions to fractional differential equations using homotopy analysis method,". IAENG International Journal of Applied Mathematics., vol. 40, no.2, pp43-51, 2010.

[23] M. U. Demirezen, "Image Based Malware Classification with Multimodal Deep Learning," *Int. J. Inf. Secur. Sci.*, vol. 10, no. 2, pp42–59, 2021, [Online]. Available: https://ijiss.org/~ijissorg/ijiss/index.php/ijiss/article/view/1014.

[24] P. Yadav, S. Tyagi, and H. Kaur, "Evolutionary extreme learning machine based collaborative filtering," *Int. J. Adv. Technol. Eng. Explor.*, vol.10, no.104, pp858–874, 2023, doi: 10.19101/IJATEE.2022.10100088.

[25] V. Ravi and M. Alazab, "Attention-based convolutional neural network deep learning approach for robust malware classification," *Comput. Intell.*, vol. 39, no. 1, pp145–168, 2023, doi: 10.1111/coin.12551.

[26] Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. Proceedingsof the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4700-4708.

[27] Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. Proceedings of the 36th International Conference on Machine Learning (ICML), pp6114-6123.

[28] Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI).

[29] Smith, J., & Johnson, A. (2020). Malware Classification Using ResNet50 Convolutional Neural Networks. Journal of Cybersecurity Research, 18(2), pp245-263.

[30] Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv preprint arXiv:1409.1556.

[31] Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp1251-1258.

[32] Khan, M. S. et al. (2019). "A Deep Learning Approach to Malware Classification Using InceptionResNetV2." Proceedings of the International Conference on Machine Learning and Data Engineering, pp210-219.

[33] Santos, A. et al. (2022). "Ensemble of CNNs for Malware Classification." Journal of Cybersecurity Research, 7(2), pp135-148.

[34] S. S. Lad and A. C. Adamuthe, "Malware classification with improved convolutional neural network model," *Int. J. Comput. Netw. Inf. Secur.*, vol.12, no.6, pp30–43, 2020, doi: 10.5815/ijcnis.2020.06.03.

[35] Thierry Noulamo, Emmanuel Tanyi, Marcellin Nkenlifack, Jean-Pierre Lienou, and Alain Djimeli, "Formalization Method of the UML Statechart by Transformation Toward Petri Nets," IAENG International Journal of Computer Science, vol. 45, no.4, pp505-513, 2018

[36] Pocholo James M. Loresco, Ryan Rhay P.Vicerra, and Elmer P. Dadios, "Segmentation of Lettuce Plants Using Super Pixels and Thresholding Methods in Smart Farm Hydroponics Setup," Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2019, 3-5 July, 2019, London, U.K., pp59-64.

[37] T. K. Gundoor and Sridevi, "Identification of Dominant Features in Non-Portable Executable Malicious File," 2022 Second International Conference on Computer Science, Engineering and Applications (ICCSEA), 2022, pp1-6, doi: 10.1109/ICCSEA54677.2022.9936451.

[38] Sridevi and T. K. Gundoor, "loT-Enabled 5G Networks for Secure Communication," *Inf. Secur. Pract. Internet Things, 5G, Next Generation Wirel. Networks*, pp1–29, 2022, doi: 10.4018/978-1 6684-3921-0.ch001.

[39] Tukkappa K Gundoor, Dr. Sridevi, "Optimized Feature Selection and classification for Non-Portable Executable Malware", *Int. j. commun. netw. inf. secur.*, vol. 16, no. 4, pp. 546–552, Sep. 2024.

[40] Sridevi, and Gundoor, T.K. (2024). Artificial Intelligence on Knowledge Management and Industry Revolution 4.0. In Knowledge Management and Industry Revolution 4.0 (eds R. Kumar, V. Jain, V.C. Ibarra, C.A. Talib and V.Kukreja). https://doi.org/10.1002/9781394242641.ch6

Dr. Sridevi, Professor, Karnatak University, Dharwad, Department of Computer Science. completed her doctorate in 2017 from Mangalore University in Mangalore. In 2021, she became a member of IAENG. Cloud computing, mobile and wireless communication, network security, advanced computer networks, and the internet of things are some of her areas of interest. Four research scholars are under her guidance at present, and one M.Phil. will be granted. She reviews articles for the Journal of Advances in Computer Science and Mathematics. over 30 research articles published in both domestic and international publications.

Mr. Tukkappa K. Gundoor was born on August 14, 1994, in Janginakoppa, Haveri district, Karnataka, India. In 2018, he graduated with an MCA in computer science from Visvesvaraya Technological University. Dr. Sridevi, a professor in the computer science department at Karnatak University in Dharwad, is currently mentoring him while he pursues his PhD. He received a research scholarship from the Karnataka government's DST and KSTePS, and he became a member of IAENG in 2021. His research focuses on "Network Security." The research subject he is now working on is "Study and design of Effective algorithm to detect non-portable malicious files." Additionally, he is a student member of the IEEE (Member No. 94567049) and the International Association of Engineers (IAENG) (Member No. 291948) and certified artificial intelligence professionals by the Defense Research and Development Organization (DIAT).