JCBI
Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606

ICASET
2024

Research Article
Collection: Intelligent Computing of Applied Sciences and Emerging Trends

Malware Image Analysis: A Deep Learning Perspective on Security Analysis

Muhammad Arshman Ali[1*], Mohsin Javed[1], Muhammad Ismail Kashif[1], Khadija Tuz Zahra[2], Ayesha Qureshi[3], and Abdul Waheed[4]

[1]Department of Computer Science, National College of Business Administration and Economics, Lahore, 54660, Pakistan.
[2]Department of Computer Science, Azteca University, Mexico City, 56600, Mexico.
[3]Department of Information Technology, Islamia University of Bahawalpur, Bahawalpur Punjab, Pakistan.
[4]Department of Computer Science, Institute of Southern Punjab, Multan, Pakistan.
[*]Corresponding Author: Muhammad Arshman Ali. Email: rajputarshman@gmail.com

Academic Editor: Salman Qadri **Published:** April 01, 2024

Abstract: Malware, also referred to as malicious software, encompasses software deliberately designed to disrupt or harm the normal operations of a computer system. There has been a surge in malware attacks in recent times, resulting in substantial financial losses for various entities such as enterprises, governments, financial institutions, healthcare providers, and others. This surge is attributed to the ease with which the reuse of scripts can generate novel forms of malware. Effective antivirus software relies on the classification of malware to safeguard against such attacks. Previous studies have employed both static and dynamic assessments; however, these approaches exhibit notable limitations in the context of reverse engineering. In this research, we introduce DenseMal, a visually-assisted malware classification system. It stands out for its rapid and accurate classification capabilities. Through a comprehensive evaluation on the publicly accessible MalIMG dataset, we scrutinized various approaches and their classifiers. DenseMal utilizes a contrast-limited adaptive histogram equalization method on images of malware samples to enhance the similarity between components belonging to the same malware family. This enhancement significantly boosts DenseMal's precision in identifying malware families. To ensure the efficacy of our framework, we initially developed a proof-of-concept implementation, subjecting it to meticulous testing. The results of extensive testing affirm that DenseMal adeptly classifies malware samples, achieving an average accuracy, precision, and recall of 96.79%, 89.91%, and 89.92%, respectively. Moreover, security engineers benefit from a user-friendly visualization tool that leverages DenseMal, facilitating further validation of its effectiveness.

Keywords: Malware Analysis; Deep Learning; CNN; Malware Classification.

1. Introduction

Malware, in the current digital age, presents a significant danger by adjusting itself to bypass conventional security procedures. Conventional techniques, such as signature-based detection, face difficulties in keeping up with the swift development of novel strains. This constraint necessitates the use of a more dynamic methodology for the investigation of malware. The process of "malware image analysis" entails closely examining the visual characteristics of malware in order to identify and reveal subtle trends [1]. Our objective is to improve the process of identification by utilizing deep learning models to extract valuable information from these visual inputs. The paper investigates contemporary malware threats, identifies the constraints of current approaches, and introduces the incorporation of deep learning concepts into cybersecurity. We want to narrow the divide between the development of malware and the need for flexible security solutions.

This work employs deep learning to categorize malware image detection into five distinct groups. The picture recognition technique involves several phases. Every individual step is crucial in the process of picture classification for the purpose of identifying malware images from various datasets. This method includes some stages:

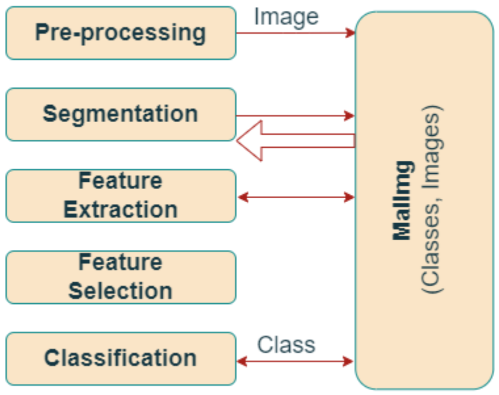

Figure 1. Classification of malware images Using DL Techniques

1. Pre-processing: During this stage, the image undergoes resampling, noise reduction, and color correction to ensure a visually clear and consistent image [2] [3].
2. Segmentation: In the second stage, the virus is divided into segments using different methods and models, including MobileNet, DenseNet, Inception V3, VGG16, and VGG19 [4].
3. Feature Extraction: At this stage, features are obtained by many methods, including the use of ABCD rules with a CCN-based model and a machine learning-based model [5].
4. Feature Selection: Following feature extraction, we choose features for categorization [6]. This process involves feature normalization, feature reduction, feature scaling and so on [6, 7].
5. Classification: During the last stage of image recognition pattern, we classified datasets into different categories of malware using a range of advanced deep learning models such as VGG16, Inception V3, and DenseNet [8].

Figure 1 demonstrates how malware image classification is done in different phases. The DenseMal Model employs malware images to accurately differentiate between 25 unique malware families, such as Yuner.A, Allaaple.A, VB.AT, and Instantaccess. The DenseMal model is capable of extracting prominent features of malware to aid in the identification of cyber pictures [9]. To create a reliable classifier, we reduce the total number of trainable parameters in the DenseMal model [10].

The CNN model's accuracy is being negatively affected by the issue of class imbalance within the Malimg dataset. The issue at hand is closely linked to the challenges being encountered by the CNN model. For improving the accuracy of the DenseMal model, we utilize a technique called SMOTE Tomek [11] to increase the number of samples by combining them from the Malware photos in each class. The confusion matrix allows us to determine all the discernible properties linked with the categorization of Malware pictures. Proposed DenseMal demonstrated superior performance compared to six baseline classifiers, namely MobileNet, EfficientNet-B0, DenseNet, VGG16, VGG19, and InceptionV3, across various assessment measures such as accuracy (ACC), area under the curve (AUC), precision (PRE), recall (REC), loss, and F1-score. In comparison to the latest state-of-the-art (SOTA) classifications, the DenseMal model yielded significant and remarkable results.

2. Literature Review

Deep learning educates itself on the features of files by studying data sets that include malicious and harmless files. The fundamental process of IMCFN consists of two parts: the production of malicious images and the fine-tuning of CNN. As a potent instrument of artificial intelligence, seven has found use in a variety of domains, including voice recognition and picture recognition [12]. In addition, [13] developed a method for classifying malware that they dubbed MCSC. This method integrated malware classification and visualization with the methods of deep learning. They first retrieved the Opcode orders from the malicious program, and then they encrypted those commands using SimHash. The SimHash values were used as pixels, and the resulting pictures were transformed into a grayscale format [14]. Last but not least, they discovered families of malware by using CNN to train photos. Their technique was able to obtain an excellent classification rate for small-scale application settings; however, it was not applicable for usage in large-scale application environments where quicker virus detection was required. A paradigm for deep learning was developed by [15] that did not make use of reverse engineering. Despite using 10,860 samples from 9 different malware families, their model obtained a classification accuracy of 96.79%.

[16] Proposed a convolutional neural network (CNN) based deep learning framework for the identification of malicious software. Tests were conducted on 25 distinct malware families included in the Malimg dataset. Their model attained a precision rate of 98% when implemented on a dataset consisting of 9,339 samples. They conducted tests on only 10% of the samples from each family, which were selected randomly.

[17] Explored the utilization of image-based techniques to identify potentially suspicious actions performed by systems. The authors suggested utilizing a fusion of hybrid image-based methodologies and CNN-based deep learning architectures to effectively classify malware. These strategies were provided as a method for detecting malicious software. Two CNN models were introduced: the Unidirectional GRU (UniGRU) model and the Bidirectional GRU (BiGRU) model [18]. In addition, they evaluated and contrasted the efficacy of these models with that of other contemporary CNN architectures, such as the Unidirectional LSTM (UniLSTM) and the Bidirectional LSTM (BiLSTM). The experiments were conducted on two publicly available datasets: the Microsoft Malware Classification Challenge (BIG, 2015) dataset and the Malimg dataset. Their model attained an average accuracy of approximately 96%; yet, it failed to consider the time spent on overhead tasks [19]. This review seeks to elucidate the recent breakthroughs and significant results in the realm of cybersecurity by analyzing and contrasting various methodologies.

3. Materials and Methods

The DenseMal model, along with six other popular deep learning classifiers (including MobileNet, DenseNet, VGG16, VGG19, inceptionV3, EfficientNet-B0), underwent comprehensive testing utilizing the experimental technique outlined in this section.

3.1. Dataset Description

The goal of our study involved utilizing a Malimg primary dataset obtained from Kaggle [20], which is a publicly accessible dataset specifically employed for this research. The dataset contains a comprehensive collection of malware images from different types. The images are divided into training, testing, and validation folders using dataset class subfolders, with a split ratio of 70/20/10. The collection consisted of three categories of malware pictures, including Yuner.A, Allaaple.A, and VB.AT, along with others such as Instantaccess. The dataset comprises 25 distinct classes and a total of 9339 samples.

3.2. Workflow of proposed DenseMal

The proposed project entails creating a model using a (CNN) Convolutional Neural Network structure to classify lung nodules as either malignant or benign. The initial stage entails gathering malware photos and preparing them using data rescaling and implementing data augmentation techniques to improve the quality of the data. Data normalization is conducted to shrink and standardize the data. SMOTE Tomek is employed to achieve dataset balancing in the context of image data.

The model is subsequently trained on the latent patterns in the data, utilizing 70% of the data for this purpose. Testing is conducted on 20% of the data, while 10% of the data is reserved for model validation. The CNN architecture comprises five layers and 64 nodes, with a kernel size of 3x3 and a max pooling2D layer. The softmax activation function is employed at the output layer to perform binary classification. The objective of the proposed study is to enhance the precision of classifying malware photos by employing convolutional neural network (CNN) techniques and classification algorithms that are compatible with constrained computational resources.

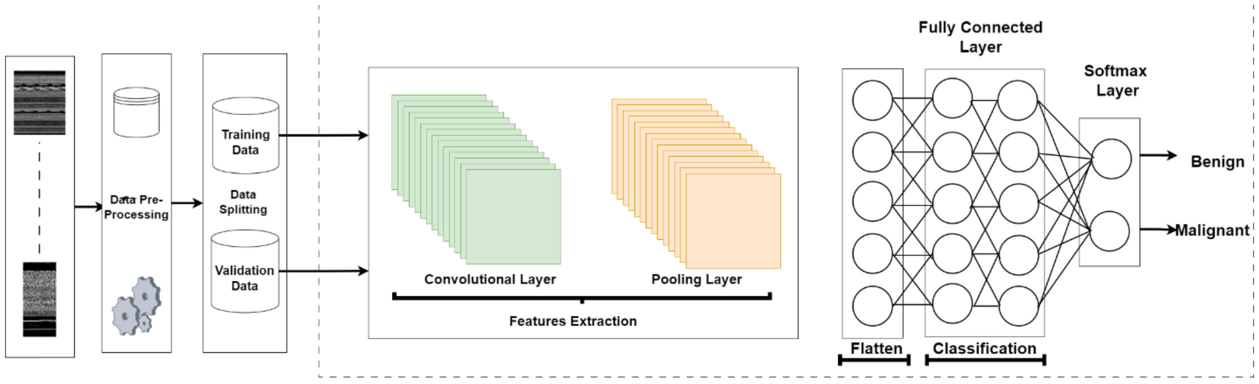

Figure 2. Proposed Model

The suggested technique utilizes Convolutional Neural Networks (CNNs) to accurately detect and classify images of malware. A Convolutional Neural Network (CNN), which is a type of DenseMal model, excels in the task of image processing and classification. Convolutional neural networks operate by sequentially applying convolutional and pooling layers to an input picture, thereby extracting progressively intricate information from the image. Subsequently, the CNN's output is commonly directed into one or many fully connected layers, which carry out the ultimate classification. Ultimately, the outputs are consolidated into an output layer, which yields the ultimate outcome indicating whether the image sample includes malware or not. The AdaBoost algorithm, a widely used ensemble learning technique, combines multiple weak classifiers (in this case, separate CNN models) to create a more powerful classifier. AdaBoost assigns a larger weight to samples that are wrongly categorized by the current set of weak classifiers during training, and a lower weight to data that are successfully classified. This allows the algorithm to focus on the most challenging samples, hence enhancing the overall accuracy of the algorithm. The pipeline we outlined appears to be a potent approach for picture classification since it amalgamates the benefits of ensemble learning with the capabilities of VGG-16. The performance of the pipeline will be influenced by the quality of the training data, the architecture of the CNN models, and the hyperparameters used for both the CNNs and the AdaBoost models.

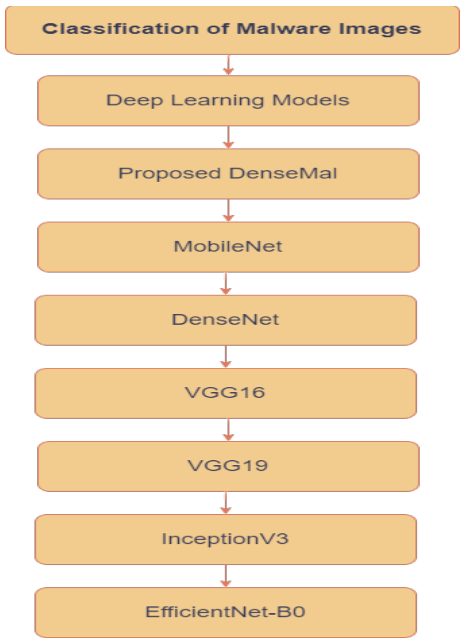

Figure 3. Pattern recognition

The AdaBoost algorithm is commonly used in image classification applications to augment the performance of a CNN-based pipeline, rather than replacing the (CNNs) Convolutional Neural Networks. (CNNs) Convolutional Neural Networks have demonstrated exceptional proficiency in image classification, especially when trained on large and diverse datasets. Pooling layers reduce the spatial dimensions of feature maps and create resistance to small translations in the input. On the other hand, convolutional layers acquire a set of filters designed to identify specific features in the input images. Typically, fully linked layers are used at the end of the CNN to do the final classification.

Several efficient VGG-16 methods, like as VGG, ResNet, and Inception, employ convolutional neural networks (CNNs) for the purpose of image categorization. These models have shown exceptional performance on various widely used image classification benchmarks, including as ImageNet, reaching the highest level of performance currently available. Ultimately, although the AdaBoost algorithm is a powerful technique for ensemble learning, it is not commonly used as the primary picture classification algorithm for VGG-16. Instead, CNN-based pipelines are commonly used and have achieved significant success in computer vision tasks.

4. Results

This section showcases the outcomes achieved by the utilization of the DenseMal model, as well as five other baseline models, namely MobileNet, DenseNet, VGG16, VGG19, inceptionV3, and EfficientNet-B0. The models have produced comprehensive findings, which are reported in Table 5. In order to accurately assess the effectiveness of deep neural networks, we employed identical settings for each network.

4.1. Experimental Setup

A total of eight models were successfully implemented using Keras. The models consisted of the six initial models, along with the DenseMal model, both with and without the utilization of the SMOTE Tomek approach as recommended. Python, a computer language, is also utilized in the creation of techniques that are not directly linked to Convolutional Neural Networks (CNN). In order to conduct the experiment, a personal computer (PC) equipped with Windows 10 operating system, 32 gigabytes of RAM, and an NVIDIA graphics processing unit (GPU) with 8 gigabytes of memory was employed.

4.2. Accuracy Comparison of Recent Deep Models with Proposed DenseMal

By applying the SMOTE Tomek technique to the identical dataset, we conducted a comparison between our proposed DenseMal model and six distinct baseline networks. Prior to proceeding with the implementation of SMOTE Tomek, we conducted a comparison of the suggested DenseMal. The DenseMal model integrates up sampling, yielding remarkable implications for the model being presented. Table 5 shows that the suggested DenseMal model with up sampling, together with MobileNet, DenseNet, VGG16, VGG19, inceptionV3, and EfficientNet-B0 achieved accuracies of 96.79%, 92.92%, 87.73%, 91.40%, 93.12%, 89.78%, and 95.80% respectively. Figure 4 demonstrates the notable enhancement achieved by employing the recommended MobileNet, DenseNet, VGG16, VGG19, InceptionV3, EfficientNet-B0 models, together with the implementation of up sampling.

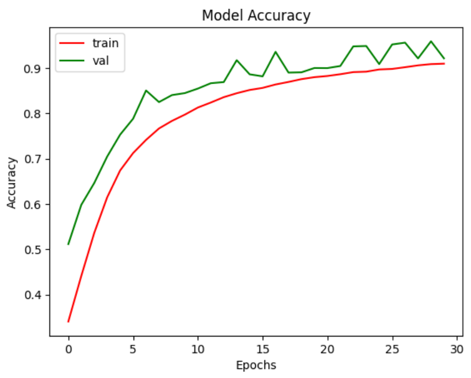

Figure 4. Accuracy of DenseMal after applying SMOTE Tomek

4.3. Comparison of DenseMal with Baseline Models in Terms of AUC

As previously said in this study, our suggested model DenseMal utilizes a Convolutional Neural Network (CNN) with several units that are highly proficient in identifying different classes of COVID-19 and Lung Cancer. In order to validate our proposed DenseMal, we conducted a comparison with six baseline networks. The baseline models, including MobileNet, DenseNet, VGG16, VGG19, inceptionV3, and EfficientNet-B0, achieved AUC values of 93.55%, 93.70%, 94.98%, 92.72%, 91.56%, and 93.23% respectively. Fig. 5 illustrates that the DenseMal algorithm, when combined with the SMOTE Tomek technique, achieved an AUC value of 97.93% using the provided datasets. After conducting a thorough analysis, we have concluded that the AUC results of the DenseMal model with SMOTE Tomek are better than the other six baseline CNN-based models, namely MobileNet, DenseNet, VGG16, VGG19, inceptionV3, and EfficientNet-B0.

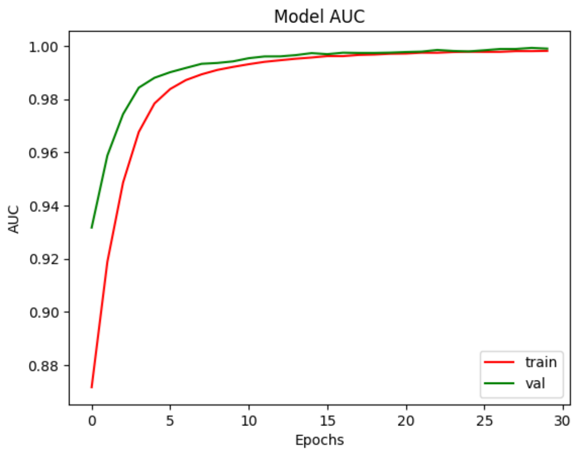

Figure 5. AUC of proposed DenseMal

4.4. Comparison of DenseMal with Baselines Models in Terms of Loss

The discrepancy between the real values and the anticipated values is calculated using loss functions. The loss was computed using a categorical cross-entropy technique in this study. The results, conversely, are more impressive when the network is built using up-sampled images. The proposed DenseMal model with up sampling achieved a loss value of 0.0427. In comparison, MobileNet, DenseNet, VGG16, VGG19, InceptionV3, and EfficientNet-B0 obtained loss values of 0.2616, 0.2033, 0.1647, 0.1886, 0.3117, and 0.1344, respectively. Figure 6 illustrates the considerable improvement in DenseMal loss when using SMOTE Tomek.

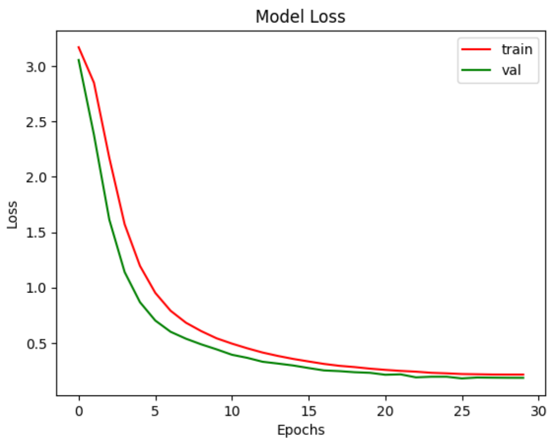

Figure 6. Loss curve of proposed DenseMal

4.5. Comparison of DenseMal with Baselines Models in Terms of Recall

Recall is calculated by dividing the number of true positive predictions by the total number of actual positive instances. High recall levels reflect the identification of a larger number of positive samples. The recall curve is used to compare the recommended DenseMal with the baseline networks, as depicted on Figure 7. The recall values achieved by the suggested DenseMal model with up sampling were 89.92%. The recall values for MobileNet, DenseNet, VGG16, VGG19, inceptionV3, and EfficientNet-B0 were 87.09%, 89.04%, 88.00%, 93.12%, 86.45%, and 94.44%, respectively. The recall performance of the proposed model has been noted to be exceptional.

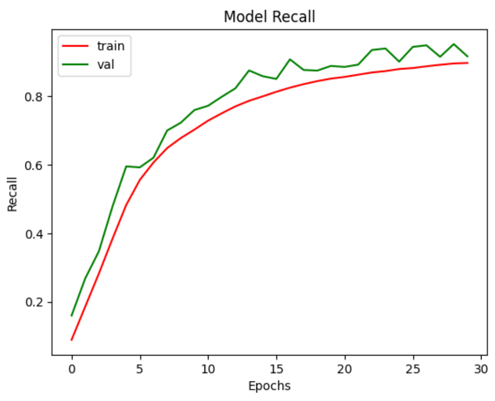

Figure 7. Recall for proposed DenseMal

4.6. Comparison of DenseMal with Baselines Models in Terms of Precision

To determine the precision value, we conducted a comparison between the DenseMal model and six other models: MobileNet, DenseNet, VGG16, VGG19, InceptionV3, and EfficientNet-B0. The DenseMal model, which incorporates the SMOTE Tomek technique, achieved a precision rate of 89.91%. The MobileNet model attains a precision score of 88.83%. The Dense-Net yields a lower result of 89.63% in comparison to other baseline models. The VGG16, VGG19, inceptionV3, and EfficientNet-B0 achieved precision values of 82.71%, 84.80%, 84.38%, and 85.79%, respectively. Fig. 8 displays the comprehensive findings.

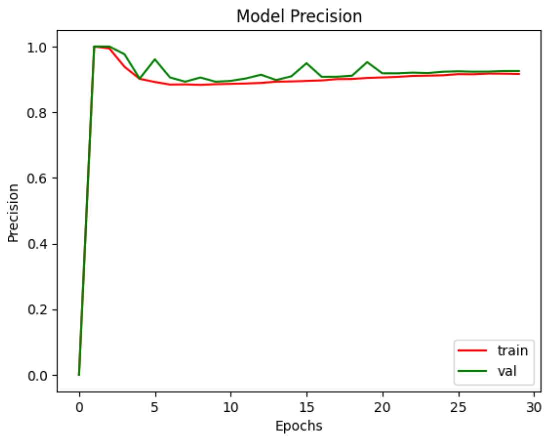

Figure 8. Precision of Proposed DenseMal

4.7. Comparison of DenseMal with Baselines Models in Terms of F1-score

The SMOTE Tomek model achieves an F1-score of 97.89%. The DenseMal model, when not utilizing the class imbalance method, achieved an F1-score of 79.99%. The F1-score values for six baseline models, namely MobileNet, DenseNet, VGG16, VGG19, inceptionV3, and EfficientNet-B0, were 88.29%, 91.41%, 93.04%, 92.88%, 89.62%, and 93.85% respectively. Fig. 9 shows that the DenseMal with SMOTE Tomek approach earned the greatest F1 score.

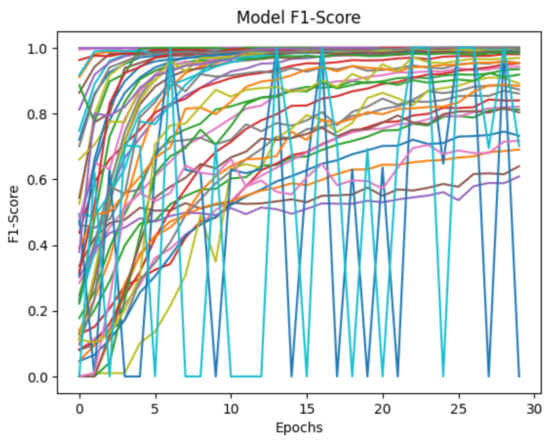

Figure 9. F-1 score for proposed DenseMal

4.8. Comparison of DenseMal with Baselines Models in Terms of ROC

It is specifically implemented to examine the effectiveness of diagnostic testing, particularly in predicting outcomes using binary or multi-classification. The AU(ROC) is employed to assess the effectiveness of a classifier; a higher AUC signifies a more efficient classifier. We assessed the precision of our proposed DenseMal algorithm on the curve, both with and without up sampling, by generating a dataset. Utilizing the identical dataset, this curve assesses the performance of the proposed DenseMal model with and without up sampling among six other models: MobileNet, DenseNet, VGG16, VGG19, inceptionV3, and EfficientNet-B0. The ROC values of the proposed DenseMal with up sampling were 0.9682, 0.9239, 0.9241, 0.9171, 0.9163, 0.9201, and 0.9315, respectively. Figure 10 illustrates the notable enhancement of the proposed DenseMal algorithm when combined with the SMOTE Tomek technique, as demonstrated by the ROC curve.

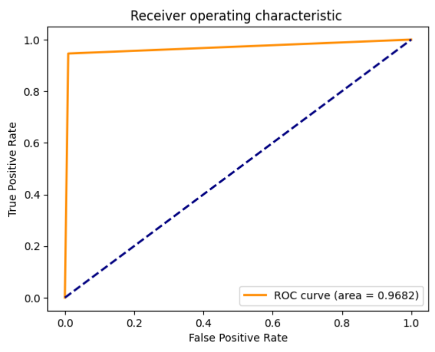

Figure 10. ROC Curve for proposed DenseMal

4.9. Comparison of DenseMal with Baselines Models in Terms of Other Networks

In order to assess the performance of the DenseMal model, we analyzed it with other networks using the confusion matrix. Figure 11 demonstrates that the technique applied to up sampled data results in significant improvement for DenseMal.

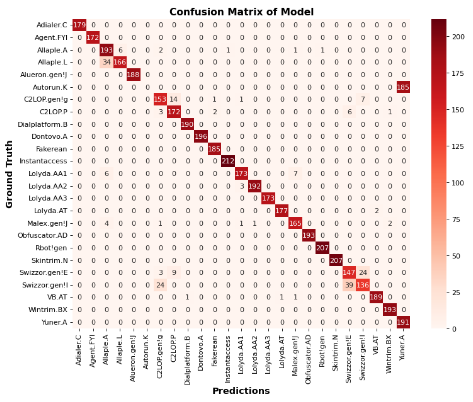

Figure 11. Confusion Matrix for DenseMal Model

5. Conclusions

This work formulated and evaluated the DenseMal model for categorizing the 25 distinct varieties of Mal-ware samples, including Yuner.A, Allaaple.A, VB.AT, and Instantaccess. These malicious software programs are currently experiencing a rise in frequency and causing harm to devices on a global scale. The use of flawed, protracted, and insufficient testing methods, coupled with a failure to promptly identify malware, has directly resulted in the compromise of vital infrastructure and data. Given the significant frequency of occurrences, it is imperative to establish a testing procedure that is both expeditious and effective. The DenseMal model has been provided to recognize the 25 different forms of malware pictures. Subsequently, these blocks are employed for the categorization of malwares at their initial stages. The upgraded structure is composed of multiple layers in each convolutional block, which are then utilized. In this inquiry, we utilize the SMOTE Tomek approach to generate samples. This approach enables us to address the issues related to the disparity in the dataset and the requirement to maintain an equitable distribution of samples across all classes. The DenseMal model achieved an AUC of 97.93%, precision of 88.83%, recall of 89.92%, F1-score of 97.89%, and accuracy of 96.79%. Therefore, it can be inferred that DenseMal can serve as a significant and influential element for the security practitioner. A limitation of our research is that the DenseMal model with SMOTE Tomek, which we have suggested, is not appropriate for analyzing malware data. For enhanced categorization outcomes of malware samples, we will incorporate Federated Learning alongside a CNN model in the future.

Funding: This research received no external funding.
Data Availability Statement: The datasets utilized for this study are discussed in the dataset's description section and are publicly available.
Conflicts of Interest: The authors declare no conflict of interest.

References

1. V. Vasani, A. K. Bairwa, S. Joshi, A. Pljonkin, M. Kaur, and M. Amoon, "Comprehensive Analysis of Advanced Techniques and Vital Tools for Detecting Malware Intrusion," Electronics, vol. 12, no. 20, p. 4299, 2023.
2. Q. Jiang, Y. Mao, R. Cong, W. Ren, C. Huang, and F. Shao, "Unsupervised decomposition and correction network for low-light image enhancement," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 10, pp. 19440-19455, 2022.
3. P. Matysiak, M. Grogan, M. Le Pendu, M. Alain, E. Zerman, and A. Smolic, "High quality light field extraction and post-processing for raw plenoptic data," IEEE Transactions on Image Processing, vol. 29, pp. 4188-4203, 2020.
4. M. I. Palma Salas, P. De Geus, and M. Botacin, "Enhancing Malware Family Classification in the Microsoft Challenge Dataset via Transfer Learning," in Proceedings of the 12th Latin-American Symposium on Dependable and Secure Computing, 2023, pp. 156-163.
5. F. Sultan, M. Kaleem, N. Ahmad, S. Rashid, M. A. Mushtaq, and N. Ahmad, "A Systematic Analysis of Skin Cancer Detection Using Machine Learning and Deep Learning Techniques," Journal of Computing & Biomedical Informatics, vol. 6, no. 01, pp. 144-159, 2023.
6. I. Ali, M. Muzammil, I. U. Haq, A. A. Khaliq, and S. Abdullah, "Deep feature selection and decision level fusion for lungs nodule classification," IEEE Access, vol. 9, pp. 18962-18973, 2021.
7. M. Naveed et al., "A Deep Learning-Based Framework for Feature Extraction and Classification of Intrusion Detection in Networks," Wireless Communications and Mobile Computing, vol. 2022, 2022.
8. J. Hemalatha, S. A. Roseline, S. Geetha, S. Kadry, and R. Damaševičius, "An efficient densenet-based deep learning model for malware detection," Entropy, vol. 23, no. 3, p. 344, 2021.
9. T. Poleto, M. M. Silva, T. R. N. Clemente, A. P. H. de Gusmão, A. P. d. B. Araújo, and A. P. C. S. Costa, "A risk assessment framework proposal based on bow-tie analysis for medical image diagnosis sharing within telemedicine," Sensors, vol. 21, no. 7, p. 2426, 2021.
10. Y. Bai, X. Lu, and B. Xu, "A Probabilistic Fuzzy Classifier for Motion Intent Recognition," IEEE Transactions on Fuzzy Systems, 2023.
11. A. d. P. Rodrigues, A. S. Luna, and L. Pinto, "An evaluation strategy to select and discard sampling preprocessing methods for imbalanced datasets: A focus on classification models," Chemometrics and Intelligent Laboratory Systems, vol. 240, p. 104933, 2023.
12. J. Zhang and D. Tao, "Empowering things with intelligence: a survey of the progress, challenges, and opportunities in artificial intelligence of things," IEEE Internet of Things Journal, vol. 8, no. 10, pp. 7789-7817, 2020.
13. H. Elgazzar, K. Spurlock, and T. Bogart, "Evolutionary clustering and community detection algorithms for social media health surveillance," Machine Learning with Applications, vol. 6, p. 100084, 2021.
14. H. Hashemi, M. E. Samie, and A. Hamzeh, "IFMD: image fusion for malware detection," Journal of Computer Virology and Hacking Techniques, vol. 19, no. 2, pp. 271-286, 2023.
15. M. H. U. SHARIF, N. JIWANI, K. GUPTA, M. A. MOHAMMED, and D. M. F. ANSARI, "A DEEP LEARNING BASED TECHNIQUE FOR THE CLASSIFICATION OF MALWARE IMAGES," Journal of Theoretical and Applied Information Technology, vol. 101, no. 1, 2023.
16. H. Naeem et al., "Malware detection in industrial internet of things based on hybrid image visualization and deep learning model," Ad Hoc Networks, vol. 105, p. 102154, 2020.
17. K. Shaukat, S. Luo, and V. Varadharajan, "A novel deep learning-based approach for malware detection," Engineering Applications of Artificial Intelligence, vol. 122, p. 106030, 2023.
18. 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.
19. A. S. Luccioni, S. Viguier, and A.-L. Ligozat, "Estimating the carbon footprint of bloom, a 176b parameter language model," Journal of Machine Learning Research, vol. 24, no. 253, pp. 1-15, 2023.
20. "Malimg Dataset." https://www.kaggle.com/datasets/manmandes/malimg