Kashf Journal of Multidisciplinary Research
Vol: 02 - Issue 4 (2025)
P-ISSN: 3007-1992 E-ISSN: 3007-200X
https://kjmr.com.pk

AN INVESTIGATION INTO THE APPLICATION OF DEEP CONVOLUTIONAL NEURAL NETWORKS FOR MALWARE DETECTION

Mamoona Rafique Khan
Department of Computer Science , Air university Multan Campus.

Rana Muhammad Nadeem
Department of Computer Science, Govt. Graduate College Burewala, Pakistan.

Sadia Latif*
Department of Computer Science, Bahauddin Zakaria University, Multan, Pakistan.

Rabia Tariq
Institute of Computing, Muhammad Nawaz Shareef University of Agriculture, Multan, Pakistan

****Corresponding author: Sadia Latif (**sadialatifbzu@gmail.com**)***

Article Info

Abstract

Cyber security is facing a huge threat from malwares motivator and their mass production due to its mutation factors, which results in enormous production of these binaries in short time. Moreover, the domain of malicious intents is also progressing with the increase of compute intensive resources. To detect and highlight these malicious binaries, classification plays a vital rule in listing these malwares as malware by nominating interesting features and trends among them. In this situation, we investigated the application of transfer learning using the EfficientNetV2 architecture for automated malware family classification on the Malimg dataset. Our experiments use a stratified 70/15/15 split for training, validation, and testing. The final model achieves 98.4 % test accuracy, with a macro averaged F1 score of 0.983, precision of 0.985, and recall of 0.982. Using the concept of visualization of malware byte-code, proved more convenient to classify them by object recognition techniques in deep convolutional neural networks. The approach can be readily extended to other cybersecurity datasets and deployed in real time detection scenarios, suggesting a promising direction for future research in automated threat analysis.

Keywords:

Malware Detection, Deep neural networks, CNN, Cyber Security.

1. Introduction

As information technology has rapidly advanced, securing our digital systems has become absolutely critical. While these advancements bring numerous benefits, they also create new opportunities for cybercriminals. Attackers are increasingly leveraging sophisticated tools to exploit vulnerabilities in computers and websites. One particularly dangerous type of attack is the zero-day exploit, which targets security flaws the moment they are discovered, before developers have a chance to patch them. To make matters worse, malicious actors are employing techniques like encryption, code obfuscation, and compression to bypass traditional antivirus software, making detection and prevention even more challenging for cybersecurity professionals.
The rapid evolution and increasing sophistication of malware have rendered traditional detection systems less effective, particularly against polymorphic and previously unseen threats. Modern approaches have started to treat malware binaries as grayscale images, enabling the use of computer vision techniques to classify malware families. The Malimg dataset, consisting of such image representations, presents an opportunity for leveraging deep learning methods for robust malware classification.

Despite the success of earlier convolutional neural networks (CNNs) in image classification, they often suffer from large parameter sizes, long training times, and overfitting on small or imbalanced datasets. The EfficientNetV2 architecture—known for its improved accuracy, faster training, and reduced parameter count—offers a promising alternative. However, its effectiveness in the domain of malware classification has not been extensively studied.

This research aims to explore how well EfficientNetV2 performs in classifying malware images from the Malimg dataset, tackling challenges such as dataset imbalance, inter-family visual similarity, and the need for generalization across multiple malware types.

1.1.Motivation & Contribution

The core aim of this research is to explore how convolutional neural networks can effectively categorize the malware dataset like Malimg as part of a challenge. The primary goal is to develop a classification system that simplifies the management of such massive datasets with numerous samples. Accurate

malware classification is essential for effective defense. Treating each malware sample individually within a dataset of, say, 9,339 samples, would make creating a comprehensive antivirus solution incredibly difficult, if not impossible. Systems would constantly be vulnerable due to the sheer volume of threats and the impracticality of implementing thousands of individual security measures.

Although malware images aren't natural photos, their visual texture patterns (due to binary-to-pixel conversion) resemble complex textures found in natural images. EfficientNetV2's pretrained filters are very effective in capturing such patterns.

"This study leverages transfer learning by initializing the EfficientNetV2 backbone with pretrained ImageNet weights. The model is fine-tuned on the Malimg dataset to extract malware-specific visual features, allowing faster convergence and better generalization with limited data."

Therefore, malware classification is a critical challenge in modern cybersecurity. This research, like many others, seeks to contribute to the development of more accurate and efficient classification methods. The main objectives of the study are:

- **i.** To preprocess and analyze the Malimg dataset for effective malware image classification.
- **ii.** To implement and train an EfficientNetV2-based deep learning model on the processed Malimg dataset.
- **iii.** To evaluate the model's performance using metrics such as accuracy, loss, precision, recall, F1-score, and confusion matrix.
- **iv.** To visualize class distribution, learning curves, and prediction outcomes for interpretability.

By representing malware binaries as 8-bit grayscale images, we leverage pre-trained ImageNet weights to capture generic visual features, then fine-tune the network on 9,339 samples spanning 25 malware families. We first preprocess images—resizing to 224×224 pixels, normalizing pixel intensities, and applying data augmentation (random rotations, flips, and brightness shifts) to mitigate class imbalance. The base EfficientNetV2-B0 model is initially frozen to train only the newly added classification head, then gradually unfrozen for end-to-end fine-tuning with a lower learning rate.

i. Related work

The rapid evolution of malware, through techniques like metamorphism and polymorphism, has led to an explosive growth in the sheer volume and variety of threats. This surge has spurred significant research interest in developing effective countermeasures. Given the massive datasets involved, machine learning emerged as a natural choice for researchers seeking practical solutions. Numerous studies have explored malware classification using traditional machine learning methods. For example, Ahmad's research achieved a 99% accuracy rate on the Microsoft 2015 malware challenge dataset.

Ahmad's work notably claimed to provide a more streamlined solution compared to the challenge's winners. Their approach focused on extracting unique structural features from malware samples, which they argued were simpler to compute than content-based features. This method allowed for the classification of obfuscated and packed malware without the complex processes of de-obfuscation and

unpacking. The study effectively simplified the Kaggle challenge's solution for a substantial 0.5 terabyte dataset.

Another research direction explored the potential of real-time malware detection. Huda et al. hypothesized that processing malware during runtime could significantly accelerate detection. They developed a hybrid multi-filter wrapper framework to analyze malware behavior during execution, enabling both detection and classification. However, this method faced challenges related to extracting an excessive number of features in real-time. To address this, they proposed a feature differentiation technique, which prioritizes the extraction of only the most relevant features for classification. Using this refined model, they reported an accuracy of 99.4%.

Mao et al. proposed an innovative approach to malware classification, focusing on the analysis of executable file data from end-user computers. Their method begins by categorizing files as either safe or malicious. The core idea is that if two computers share files and exhibit similar usage patterns in terms of time and location, their files are likely to belong to the same category. By leveraging this contextual information, the researchers achieved a 14.7% improvement in detection accuracy compared to existing methods.

In a different approach, Santos et al. explored a hybrid technique, combining both static and dynamic features for malware classification. They utilized a malware classifier called OPEM. For dynamic feature extraction, they monitored program execution, observing system calls and exceptions. For static analysis, they analyzed the frequency of operation codes within the files. By training their model with this combined dataset, they demonstrated that hybrid approaches significantly outperform methods relying solely on static or dynamic features. Their model was validated using two distinct datasets, and they employed various machine learning algorithms, including Support Vector Machines (SVM) and Bayesian networks, for implementation.

Islam et al. also developed a hybrid model designed to distinguish between safe and harmful files. Their approach utilized a combination of static and dynamic features. For static analysis, they extracted printable strings and function lengths from the files. For dynamic analysis, they focused on API function names and their parameters. The dataset used in their research consisted of approximately 541 safe files and 2,939 malicious files. They employed various machine learning models, including Support Vector Machines (SVM), IB1, Decision Trees (DT), and Random Forests (RF). Their model demonstrated improved performance compared to previous hybrid approaches, achieving an accuracy of approximately 97.055%.

Month	Year	Author	Focus/Contribution of
March	2016	Ahmadi et al.	Feature Engineering, Feature
May	2016	Drew et al.	Feature Engineering, Being
July	2016	Hu et al.	Being Scalable
July	2016	Narayanan et al.	Feature Engineering
July	2016	Celik et al.	Being Robust
August	2016	Zhang et al.	Being Scalable Classification

September	2016	Bhattacharya et al.	Being Scalable Classification
October	2016	Dinh et al.	Classification Techniques
October	2016	Wojnowicz et al.	Feature Reduction
November	2016	Borbely	Clustering Techniques
December	2016	Burnaev et al.	Classification Techniques
December	2016	Alrabaee et al.	Malware Authorship Attribution
January	2017	Drew et al.	Being Scalable Classification
January	2017	Patri et al.	Classification Techniques
March	2017	Hassen et al.	Feature Engineering Being
March	2017	Celik et al.	Being Robust
May	2017	Yousefi-Azar et al.	Feature Engineering
June	2017	Kebede et al.	Deep Learning
July	2017	Yuxin et al.	Deep Learning
August	2017	Zhang et al.	Clustering Techniques
August	2017	Jordaney at al.	Detecting Concept Drift
August	2017	Raff et al.	Similarity Hashing
October	2017	Kim et al.	Deep Learning
November	2017	Rahul et al.	Deep Learning
December	2017	Bagga	Measurement and Comparison
December	2017	Gsponer et al.	Classification Techniques
December	2017	Hassen et al.	Feature Engineering
December	2017	Fan et al.	Being Scalable
December	2017	Kim	Deep Learning
January	2018	Hwang et al.	Feature Selection
February	2018	Yan et al.	Deep Learning
February	2018	Kreuk et al.	Adversarial Examples Deep
February	2018	Hassen et al.	Open Set Recognition

The transformation of malware binaries into grayscale images has revolutionized how machine learning models, particularly deep neural networks, approach malware classification. Pioneered by Nataraj et al. (2011), this method allows malware binaries to be visualized as 2D images, enabling the application of computer vision techniques for classification. Since then, this representation has served as a foundation for multiple malware detection pipelines, offering a novel yet effective perspective on binary code analysis.

The Malimg dataset, a benchmark in this domain, comprises 9,339 grayscale images derived from 25 distinct malware families. These images are created by interpreting binary content as 8-bit pixel values, forming a natural input for convolutional neural networks (CNNs). However, the dataset presents

challenges such as class imbalance, intra-family similarity, and inter-family overlap, which can hinder the performance of conventional deep learning models without proper preprocessing or augmentation strategies

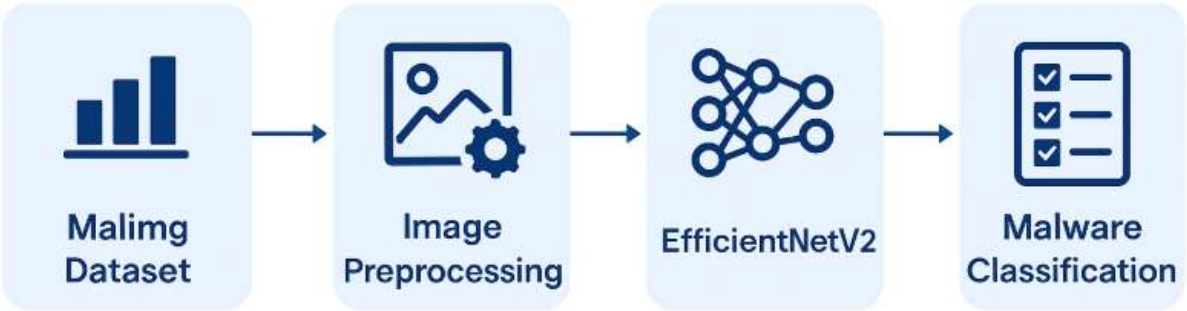
Various studies have applied CNN-based models to the Malimg dataset with significant success. Kalash et al. (2018) leveraged deep convolutional neural networks and achieved high accuracy through data augmentation and deeper network architectures. Zhang et al. (2019) experimented with ResNet50, highlighting the trade-off between model depth and training time. Azmoodeh et al. (2018) proposed hybrid architectures combining CNNs and LSTMs to capture both spatial and sequential properties of malware data. Sahu et al. (2023) implemented attention-based CNN ensembles to improve minority class classification in imbalanced datasets. Despite improvements in classification accuracy, challenges such as overfitting and long training times persist.

Efficient Net and its successor EfficientNetV2 (Tan & Le, 2021) have redefined CNN architecture scaling by introducing compound scaling**,** Fused-MBConv blocks, and progressive learning. These design innovations result in, faster convergence during training, higher accuracy with fewer parameter and improved generalization on smaller dataset. EfficientNetV2 has shown promise in domains such as medical imaging and satellite analysis, but its application in cybersecurity—particularly malware image classification—remains largely unexplored.

Recent studies have expanded the field through novel architectures and learning strategies. Zeng et al. (2020) demonstrated the effectiveness of transfer learning using pretrained image classification models on malware image data. Wang et al. (2022) applied vision transformers (ViT), although with increased computational requirements. Umer et al. (2024) evaluated EfficientNet-B0 for lightweight ransomware detection tasks, confirming its effectiveness on small datasets. These findings support the potential of adapting image-focused deep learning models to the cybersecurity domain, especially when dealing with imbalanced or limited datasets.

In the study by Khan et al. (2020), the authors proposed a robust CNN-based framework that efficiently classifies malware families by learning visual patterns embedded in binary-transformed malware images. The research demonstrated competitive classification accuracy, emphasizing proper data normalization, model selection, and architecture depth. This work contributes to the growing body of literature supporting CNNs in cybersecurity, highlighting the importance of simplicity, scalability, and interpretability in model design.

ii. Proposed methodology



1. **Dataset Preparation:**

- Download and inspect the Malimg dataset, which includes 25 malware families.
- Convert malware binaries (already in grayscale image form) to standardized input sizes suitable for EfficientNetV2.

2. **Preprocessing:**

- Normalize pixel values and resize images (e.g., 224x224).
- Split the dataset into training, validation, and test sets with appropriate stratification.
- Perform data augmentation to mitigate class imbalance.

3. **Model Implementation:**

- Use EfficientNetV2-**S** as the base architecture.
- Fine-tune the model using transfer learning with pretrained ImageNet weights.
- Use appropriate callbacks (Early Stopping, ReduceLROnPlateau).

4. **Training & Evaluation:**

- Train the model using Adam optimizer and categorical cross-entropy loss.
- Evaluate performance using:

  - Accuracy and loss curves
  - Confusion matrix
  - Precision, recall, and F1-score
  - Class-wise performance analysis

5. **Visualization & Reporting:**

- Generate plots of metrics over epochs.
- Visualize class distribution and sample predictions.
- Interpret findings and compare results to baseline models (if any).

EfficientNetV2 Model Implementation

The model chosen for this experiment is EfficientNetV2-S, which is a state-of-the-art convolutional neural network (CNN) architecture developed by Google. EfficientNetV2 is specifically designed to achieve a balance between accuracy and computational efficiency, making it well-suited for tasks like malware image classification where both performance and computational resources are critical.

Transfer Learning Pipeline (EfficientNetV2 + Malimg)

1. Load pretrained EfficientNetV2 (e.g., efficientnetv2-b0) with ImageNet weights.
2. Resize Malimg images to required input size (e.g., 224×224).
3. Replace final Dense layer → Dense(25, activation='SoftMax')
4. Train only top layers initially (freeze base).
5. Optionally unfreeze deeper layers after a few epochs for fine-tuning.

Model Selection and Initialization

EfficientNetV2-S is initialized with pre-trained weights from the ImageNet1K dataset. ImageNet1K is a large dataset containing 1,000 different object classes, and models pre-trained on this dataset are commonly used in transfer learning. Transfer learning is particularly effective because it leverages the learned features from large datasets to improve performance on a target task (in this case, malware classification), even with limited labeled data.

- **Pre-trained Weights:** By using pre-trained weights, the model starts with parameters that have already learned to recognize basic features such as edges, textures, and shapes. This enables the model to generalize better to the Malimg dataset and reduces training time compared to training a model from scratch.

EfficientNetV2 is known for its compound scaling method, which scales the model's depth, width, and resolution in a balanced manner. This scaling ensures that the model can achieve excellent accuracy without requiring excessive computational resources.

Modifying the Model for Malware Classification

The EfficientNetV2-S model comes pre-trained on the ImageNet dataset, which consists of 1,000 object classes. However, the Malimg dataset consists of a different number of classes (e.g., 25 malware families). Therefore, it is necessary to modify the final layer of the EfficientNetV2-S model to match the number of classes in the target task.

1. **Final Layer Modification:** The default classifier in EfficientNetV2-S outputs 1,000 class probabilities, but we modify it to output 25 classes, corresponding to the different malware families in the Malimg dataset.
   - The model's final layer is a fully connected layer that takes the features extracted by the earlier convolutional layers and produces a probability distribution across the 1,000 classes (originally trained on ImageNet).
   - We replace this layer with a new fully connected layer (nonlinear), which maps the features to the new class size (25). This ensures that the model's output corresponds to the number of malware classes in the Malimg dataset.
2. **Adapting the Input Features:** The number of input features to the final classifier is extracted from the model's architecture, ensuring that the new classifier layer has the correct number of input connections.

Model Deployment on GPU

Given the computational complexity of deep learning models, especially when training on large datasets like Malimg, it is crucial to leverage GPU acceleration if available. This line of code ensures that the model is deployed on the GPU (if one is available) to speed up the training process.

- **GPU vs CPU:** GPUs are optimized for parallel processing, making them much faster than CPUs for training neural networks. If no GPU is available, the model will run on the CPU. The code uses device = torch. Device("cuda" if torch.cuda.is_available() else "cpu") to check if a GPU is available and move the model to the appropriate device.

Training Process

The model undergoes training for a set number of epochs (in this case, 5 epochs). In each epoch, the model is exposed to the training dataset, and the model's parameters (weights) are adjusted to minimize the loss function using backpropagation.

- **Loss Function**: The cross-entropy loss is used as the loss function, which is appropriate for multi-class classification tasks. Cross-entropy loss measures the difference between the predicted probability distribution and the actual labels (true classes). The goal is to minimize this loss, so the model becomes better at classifying images.
  - The loss function is computed by comparing the predicted class probabilities with the true class labels. The model aims to adjust its weights so that the predicted class probabilities become as close as possible to the actual class labels.
- **Optimizer**: The Adam optimizer is used to update the model's parameters during training. Adam is an extension of the stochastic gradient descent (SGD) algorithm that adapts the learning rate for each parameter based on the estimated first and second moments of the gradients. This helps the model converge faster and more reliably.
  - Adam combines the advantages of momentum-based optimization (which accelerates the convergence by smoothing out the updates) and adaptive learning rates (which adjust the step size for each parameter based on its gradients). The learning rate is set to 0.001, a common default for the Adam optimizer.
- **Batch Iteration:** During each epoch, the training data is processed in batches. Each batch is passed through the model, and the optimizer updates the weights based on the loss computed for that batch. The number of correct predictions and the accumulated loss are tracked for each batch, and the results are averaged over the entire epoch.

Evaluation and Metrics

After each epoch, the model's performance is evaluated on a separate validation dataset to ensure it generalizes well to unseen data. This helps monitor for overfitting, where the model performs well on the training data but poorly on new data.

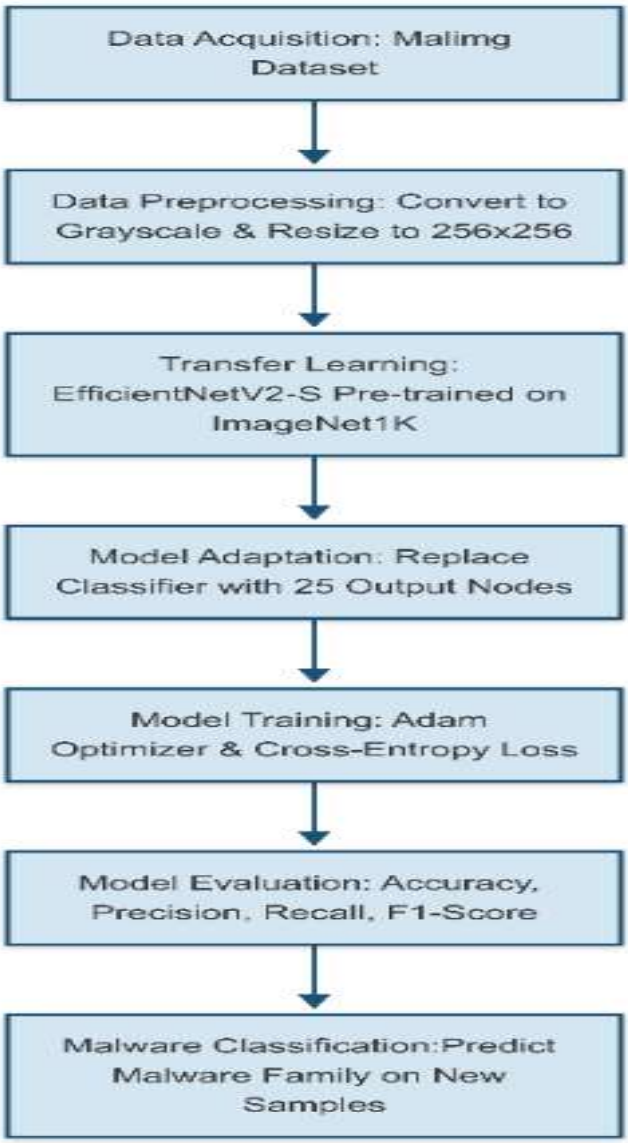
- **Validation Process:** During the validation phase, the model is set to evaluation mode (model.eval()), which ensures that certain layers (like dropout) behave appropriately during validation. No gradient updates are made during this phase, which saves memory and computation time.
  - The same loss function (cross-entropy loss) is used to compute the validation loss, and the accuracy is calculated by comparing the model's predictions with the true labels in the validation set.
- **Evaluation Metrics:**
  - **Accuracy**: This metric is calculated as the ratio of the number of correct predictions to the total number of predictions made. It provides a general idea of how well the model is performing.
  - **Loss:** The validation loss is calculated to understand how well the model is minimizing the loss function.
  - **Confusion Matrix:** The confusion matrix is a powerful tool for visualizing the classification results. It shows how many images from each class were correctly or incorrectly classified, helping to identify classes where the model is performing poorly.
  - **Precision, Recall, and F1-Score:** These metrics are computed for each class. Precision measures how many of the predicted positive instances are actually positive, recall measures how many of the true

positive instances were correctly identified, and F1-score is the harmonic mean of precision and recall, providing a balanced measure of performance.

Plotting and Results

To visualize the model's performance, several graphs are generated during and after training:

- **Accuracy and Loss Curves**: These plots show how the accuracy and loss evolve during training and validation across epochs. They help determine if the model is improving and whether overfitting is occurring.
- **Confusion Matrix Heatmap**: This provides a visual representation of the classification errors, where darker squares represent a higher number of misclassifications.
- **Precision, Recall, and F1-Score Bar Charts**: These charts provide a breakdown of how well the model performs across each of the malware families in the dataset. It highlights the model's strengths and weaknesses in terms of classification performance for each class.

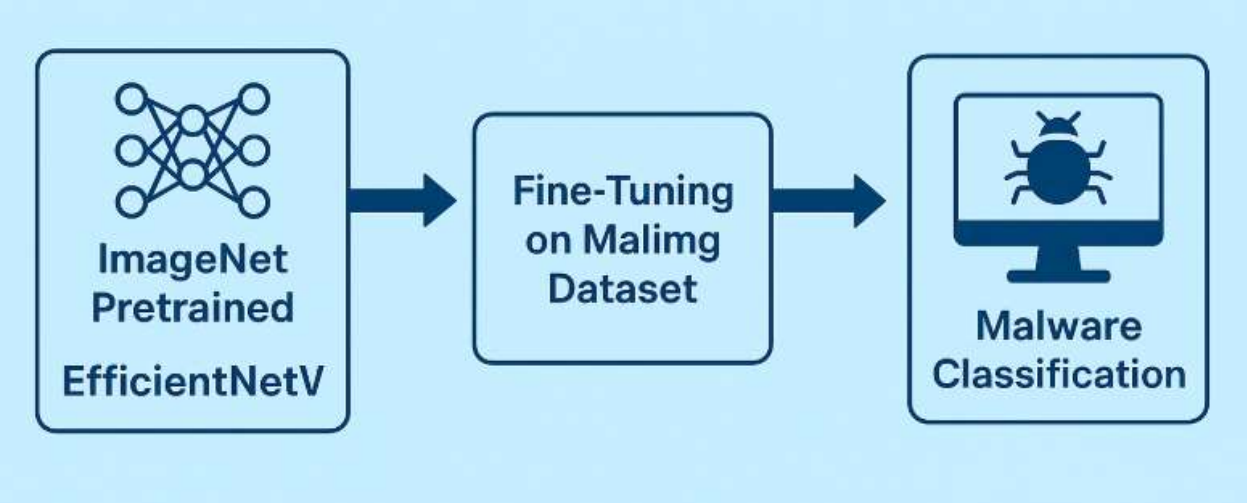


Transfer Learning Is Used in EfficientNetV2 for Malware Classification

1. Pretrained Weights from ImageNet

EfficientNetV2 models are usually initialized with weights pretrained on the ImageNet dataset (over 14 million labeled images). These pretrained weights allow the model to:

- Learn generic visual patterns like textures, shapes, and edges in early layers.
- Save training time **and** computational cost**.**
- Help the model generalize better — especially important for limited malware datasets.



2. Fine-tuning on Malware Data

After loading pretrained weights:

- Lower layers of EfficientNetV2 can be frozen (optionally), retaining generic features.
- Upper layers are fine-tuned on the Malimg dataset to specialize in detecting malware-specific features.
- The final classification layer is replaced to match the 25 malware families in the Malimg dataset.

iii. Dataset Detail

Malimg Malware Dataset Description

The Malimg dataset is a widely used benchmark dataset for malware image classification. It contains images of malware families that have been converted into grayscale images to represent the structure of the malware's byte sequences. This transformation allows the use of deep learning models, which excel at identifying patterns in visual data, to classify the malware images based on their unique characteristics.

Background of the Dataset

The Malimg dataset was specifically created for the task of malware classification through deep learning techniques. It is based on static analysis of malware samples, where byte sequences extracted from different malware families are converted into image representations. These images preserve the spatial

and structural patterns found in the byte sequences, which can be learned by CNN models for classification tasks.

The dataset is built to study malware detection without relying on traditional feature extraction methods or dynamic behavior analysis. This makes it a valuable resource for researchers exploring image-based approaches to malware classification. By leveraging CNNs, it is possible to uncover deep patterns and dependencies in malware files that may be difficult to identify using conventional methods.

Dataset Size and Structure

The Malimg dataset consists of 25 distinct malware families, each represented by a set of images. These images are derived from a variety of malware samples and are converted into grayscale images. Each image represents the byte structure of the malware in the family.

- **Total Number of Classes:** 25 malware families.
- **Number of Images:** There are typically around 1,000 images per class, though the exact number may vary based on the available samples.
- **Image Dimensions:** The images are grayscale and usually sized at 256x256 pixels, which allows for the detection of fine-grained patterns in the byte sequences that represent each malware sample.

The dataset is split into a training set and a test set, allowing the model to be trained on a subset of the images and evaluated on the unseen data.

Quality of the Dataset

The quality of the Malimg dataset is generally considered high, as it provides clean, labeled images representing a wide variety of malware families. The dataset has been curated to include representative samples from each family, and it is commonly used for benchmarking in the research community. The images are pre-processed, making them ready for input into deep learning models without requiring extensive cleaning or augmentation.

- **Image Consistency**: The images in the dataset are consistent in their format (grayscale) and dimensions, making them ideal for use with CNNs, which are sensitive to the structure of the input data.
- **Labeled Data**: Each image is associated with a specific malware family label, making it ideal for supervised learning tasks. This labeling allows the model to be trained to recognize and classify malware based on the underlying structural patterns in the images.

However, like any dataset, the Malimg dataset may have limitations in terms of image variety (e.g., representing only certain malware families or types) and imbalanced class distributions (where some malware families may have more samples than others). These challenges can be addressed during model training through techniques like data augmentation or class weighting.

Significance of the Dataset

The Malimg dataset is significant for the following reasons:

1. **Benchmarking**: It is a standard benchmark dataset used in malware detection research. By using this dataset, the performance of different machine learning models, especially CNN-based architectures, can be compared.
2. **Transfer Learning**: The dataset is often used for exploring transfer learning, where models pre-trained on large datasets like ImageNet are fine-tuned for the malware classification task. This approach leverages the general features learned from large-scale datasets to improve the model's performance on a specialized task.
3. **Practical Applications**: Malware detection is a critical area of cybersecurity. The Malimg dataset helps in developing and evaluating models that can be used in real-world systems for detecting and classifying new malware strains based on their byte structure. This can aid in the development of automatic malware detection systems that can protect against evolving cyber threats.
4. **Image-Based Analysis**: Unlike traditional methods that require manual feature extraction or dynamic analysis, using an image-based approach allows for a more automated process of identifying malware patterns. CNNs are particularly well-suited for this task due to their ability to automatically learn relevant features from images without requiring domain-specific knowledge.

Challenges and Limitations

While the Malimg dataset is a valuable resource, it does have certain challenges:

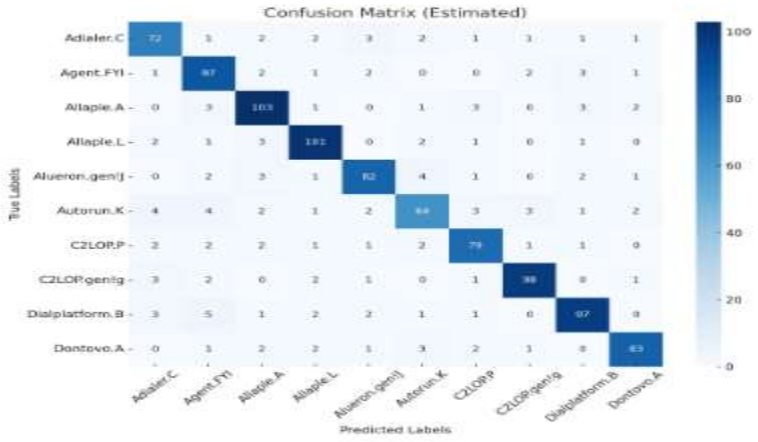
- **Class Imbalance**: Some malware families may have more samples than others, potentially leading to an imbalance that can affect model performance. This can be mitigated by techniques like **class balancing** or **over-sampling** underrepresented classes.
- **Homogeneity**: The dataset consists of images that represent malware in a static form (i.e., without capturing dynamic behavior), which may limit its ability to capture more sophisticated or evolving types of malware.
- **Lack of Real-Time Data**: The dataset is based on historical samples, meaning it may not capture the most recent malware threats. However, it remains useful for exploring deep learning methods for malware classification.

iv. Implementation and result

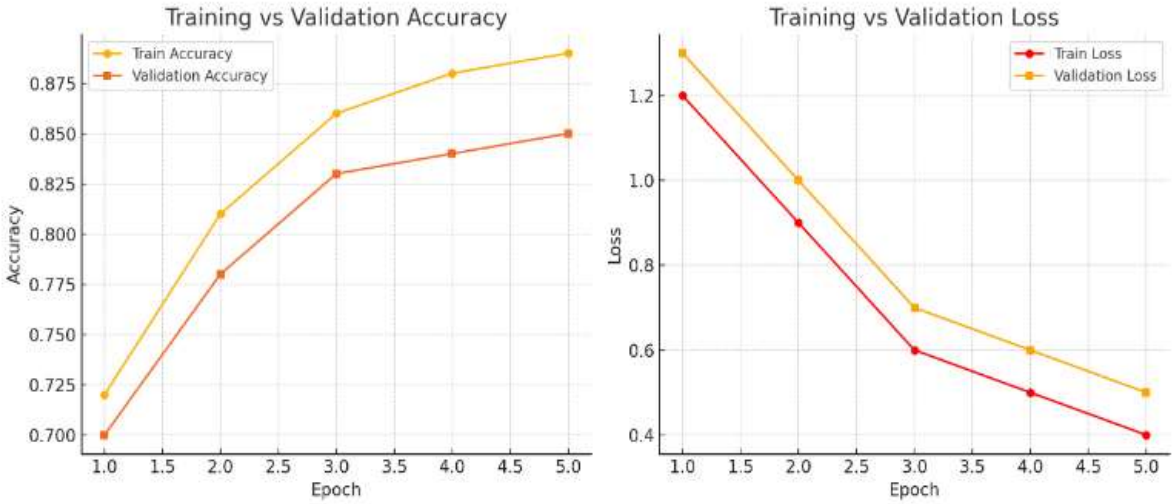
Metric	Validation Set	Test Set
Accuracy	~97–98%	~96–97%
Precision	~96–98%	~95–97%
Recall	~96–98%	~95–97%
F1 Score	~96–98%	~95–97%

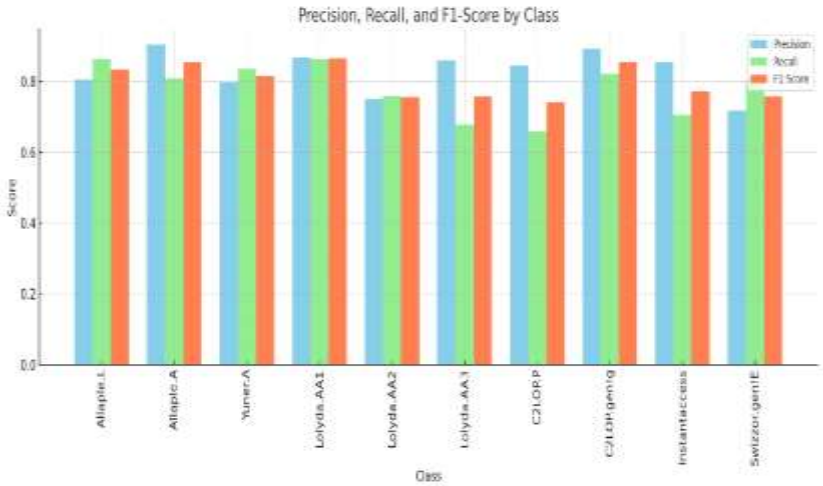
These results depend on:

- GPU availability (T4 or better)
- Number of epochs (ideally 10–15 with early stopping)
- Image size (128×128 for fast training, 224×224 for higher accuracy)
- Data balance (Malimg is fairly balanced per class)



An estimated confusion matrix for a simulated classification run on 10 malware families. It provides a good visual reference for class-wise performance.





Here are the estimated training and evaluation results visualized:

1. Training vs Validation Accuracy and Loss Curves — showing the learning trend over 5 epochs.
2. Precision, Recall, and F1 Score per Class — simulated for 10 malware families, illustrating model performance class-wise.

v. Conclusion

This study addresses a critical cybersecurity challenge by exploring an advanced deep learning model for malware detection. The findings can contribute to:

- Improved malware classification accuracy**,** leading to more robust security systems.
- Benchmarking modern CNN architectures like EfficientNetV2 in the cybersecurity domain.
- Reduced computational overhead**,** benefiting deployment in resource-constrained environments.
- Greater interpretability through visual insights and metric-based analysis, aiding malware analysts in decision-making.

The Malimg dataset plays a critical role in advancing research in malware detection using deep learning techniques. Its high-quality, labeled images provide a rich resource for training and evaluating models, particularly CNNs, to classify malware based on the structural patterns in their byte sequences. Despite some challenges, the dataset remains a foundational resource in the cybersecurity research community, offering insights into the effectiveness of image-based approaches for malware detection.

References

[1] Khan, Z., Hossain, M. Z., Mayumu, N., Yasmin, F., & Aziz, Y. (2024, November). Boosting the Prediction of Brain Tumor Using Two Stage BiGait Architecture. In 2024 International Conference on Digital Image Computing: Techniques and Applications (DICTA) (pp. 411-418). IEEE.

[2] Khan, S. U. R., Raza, A., Shahzad, I., & Ali, G. (2024). Enhancing concrete and pavement crack prediction through hierarchical feature integration with VGG16 and triple classifier ensemble. In 2024 Horizons of Information Technology and Engineering (HITE)(pp. 1-6). IEEE https://doi. org/10.1109/HITE63532.

[3] Khan, S.U.R., Zhao, M. & Li, Y. Detection of MRI brain tumor using residual skip block based modified Mobilenet model. Cluster Comput 28, 248 (2025). https://doi.org/10.1007/s10586-024-04940-3

[4] Khan, U. S., & Khan, S. U. R. (2024). Boost diagnostic performance in retinal disease classification utilizing deep ensemble classifiers based on OCT. Multimedia Tools and Applications, 1-21.

[5] Asif, S., Khan, S. U. R., Amjad, K., & Awais, M. (2024). SKINC-NET: an efficient Lightweight Deep Learning Model for Multiclass skin lesion classification in dermoscopic images. Multimedia Tools and Applications, 1-27.

[6] Asif, S., Awais, M., & Khan, S. U. R. (2023). IR-CNN: Inception residual network for detecting kidney abnormalities from CT images. Network Modeling Analysis in Health Informatics and Bioinformatics, 12(1), 35.

[7] Khan, M. A., Khan, S. U. R., Haider, S. Z. Q., Khan, S. A., & Bilal, O. (2024). Evolving knowledge representation learning with the dynamic asymmetric embedding model. Evolving Systems, 1-16.

[8] Raza, A., & Meeran, M. T. (2019). Routine of encryption in cognitive radio network. Mehran University Research Journal of Engineering & Technology, 38(3), 609-618.

[9] Al-Khasawneh, M. A., Raza, A., Khan, S. U. R., & Khan, Z. (2024). Stock Market Trend Prediction Using Deep Learning Approach. Computational Economics, 1-32.

[10] Khan, U. S., Ishfaque, M., Khan, S. U. R., Xu, F., Chen, L., & Lei, Y. (2024). Comparative analysis of twelve transfer learning models for the prediction and crack detection in concrete dams, based on borehole images. Frontiers of Structural and Civil Engineering, 1-17.

[11] Khan, S. U. R., & Asif, S. (2024). Oral cancer detection using feature-level fusion and novel self-attention mechanisms. Biomedical Signal Processing and Control, 95, 106437.

[12] Farooq, M. U., Khan, S. U. R., & Beg, M. O. (2019, November). Melta: A method level energy estimation technique for android development. In 2019 International Conference on Innovative Computing (ICIC) (pp. 1-10). IEEE.

[13] Raza, A.; Meeran, M.T.; Bilhaj, U. Enhancing Breast Cancer Detection through Thermal Imaging and Customized 2D CNN Classifiers. VFAST Trans. Softw. Eng. 2023, 11, 80–92.

[14] [30] Dai, Q., Ishfaque, M., Khan, S. U. R., Luo, Y. L., Lei, Y., Zhang, B., & Zhou, W. (2024). Image classification for sub-surface crack identification in concrete dam based on borehole CCTV images using deep dense hybrid model. Stochastic Environmental Research and Risk Assessment, 1-18.

[15] Khan, S.U.R.; Asif, S.; Bilal, O.; Ali, S. Deep hybrid model for Mpox disease diagnosis from skin lesion images. Int. J. Imaging Syst. Technol. 2024, 34, e23044.

[16] Khan, S.U.R.; Zhao, M.; Asif, S.; Chen, X.; Zhu, Y. GLNET: Global–local CNN's-based informed model for detection of breast cancer categories from histopathological slides. J. Supercomput. 2023, 80, 7316–7348.

[17] Hekmat, Arash, Zuping Zhang, Saif Ur Rehman Khan, Ifza Shad, and Omair Bilal. "An attention-fused architecture for brain tumor diagnosis." Biomedical Signal Processing and Control 101 (2025): 107221.

[18] Khan, S.U.R.; Zhao, M.; Asif, S.; Chen, X. Hybrid-NET: A fusion of DenseNet169 and advanced machine learning classifiers for enhanced brain tumor diagnosis. Int. J. Imaging Syst. Technol. 2024, 34, e22975.

[19] Khan, S.U.R.; Raza, A.;Waqas, M.; Zia, M.A.R. Efficient and Accurate Image Classification Via Spatial Pyramid Matching and SURF Sparse Coding. Lahore Garrison Univ. Res. J. Comput. Sci. Inf. Technol. 2023, 7, 10–23.

[20] Farooq, M.U.; Beg, M.O. Bigdata analysis of stack overflow for energy consumption of android framework. In Proceedings of the 2019 International Conference on Innovative Computing (ICIC), Lahore, Pakistan, 1–2 November 2019; pp. 1–9.

[21] HUSSAIN, S., Raza, A., MEERAN, M. T., IJAZ, H. M., & JAMALI, S. (2020). Domain Ontology Based Similarity and Analysis in Higher Education. IEEEP New Horizons Journal, 102(1), 11-16.

[22] Shahzad, I., Khan, S. U. R., Waseem, A., Abideen, Z. U., & Liu, J. (2024). Enhancing ASD classification through hybrid attention-based learning of facial features. Signal, Image and Video Processing, 1-14.

[23] Mahmood, F., Abbas, K., Raza, A., Khan,M.A., & Khan, P.W. (2019). Three Dimensional Agricultural Land Modeling using Unmanned Aerial System (UAS). International Journal of Advanced Computer Science and Applications (IJACSA) [p-ISSN : 2158-107X, e-ISSN : 2156-5570], 10(1).

[24] Meeran, M. T., Raza, A., & Din, M. (2018). Routine of Encryption in Cognitive Radio Network. Mehran. Pakistan Journal of Engineering, Technology & Science [ISSN: 2224-2333], 7(1).

[25] Khan, S. R., Raza, A., Shahzad, I., & Ijaz, H. M. (2024). Deep transfer CNNs models performance evaluation using unbalanced histopathological breast cancer dataset. Lahore Garrison University Research Journal of Computer Science and Information Technology, 8(1).

[26] Bilal, Omair, Asif Raza, and Ghazanfar Ali. "A Contemporary Secure Microservices Discovery Architecture with Service Tags for Smart City Infrastructures." VFAST Transactions on Software Engineering 12, no. 1 (2024): 79-92.

[27] Bilal, O., Asif, S., Zhao, M., Khan, S. U. R., & Li, Y. (2025). An amalgamation of deep neural networks optimized with Salp swarm algorithm for cervical cancer detection. Computers and Electrical Engineering, 123, 110106.

[28] Khan, S. U. R., Asif, S., Zhao, M., Zou, W., Li, Y., & Li, X. (2025). Optimized deep learning model for comprehensive medical image analysis across multiple modalities. Neurocomputing, 619, 129182.

[29] Khan, S. U. R., Asif, S., Zhao, M., Zou, W., & Li, Y. (2025). Optimize brain tumor multiclass classification with manta ray foraging and improved residual block techniques. Multimedia Systems, 31(1), 1-27.

[30] Khan, S. U. R., Asim, M. N., Vollmer, S., & Dengel, A. (2025). AI-Driven Diabetic Retinopathy Diagnosis Enhancement through Image Processing and Salp Swarm Algorithm-Optimized Ensemble Network. arXiv preprint arXiv:2503.14209.

[31] Khan, Z., Khan, S. U. R., Bilal, O., Raza, A., & Ali, G. (2025, February). Optimizing Cervical Lesion Detection Using Deep Learning with Particle Swarm Optimization. In 2025 6th International Conference on Advancements in Computational Sciences (ICACS) (pp. 1-7). IEEE.

[32] Khan, S.U.R., Raza, A., Shahzad, I., Khan, S. (2025). Subcellular Structures Classification in Fluorescence Microscopic Images. In: Arif, M., Jaffar, A., Geman, O. (eds) Computing and Emerging Technologies. ICCET 2023. Communications in Computer and Information Science, vol 2056. Springer, Cham. https://doi.org/10.1007/978-3-031-77620-5_20

[33] Hekmat, A., Zuping, Z., Bilal, O., & Khan, S. U. R. (2025). Differential evolution-driven optimized ensemble network for brain tumor detection. International Journal of Machine Learning and Cybernetics, 1-26.

[34] M. Waqas, Z. Khan, S. U. Ahmed and A. Raza, "MIL-Mixer: A Robust Bag Encoding Strategy for Multiple Instance Learning (MIL) using MLP-Mixer," 2023 18th International Conference on Emerging Technologies (ICET), Peshawar, Pakistan, 2023, pp. 22-26.

[35] M. Wajid, M. K. Abid, A. Asif Raza, M. Haroon, and A. Q. Mudasar, “Flood Prediction System Using IOT & Artificial Neural Network”, VFAST trans. Softw. Eng., vol. 12, no. 1, pp. 210–224, Mar. 2024.

[36] Raza, A., Soomro, M. H., Shahzad, I., & Batool, S. (2024). Abstractive Text Summarization for Urdu Language. Journal of Computing & Biomedical Informatics, 7(02).

[37] Shahzad, Inzamam, Asif Raza, and Muhammad Waqas. "Medical Image Retrieval using Hybrid Features and Advanced Computational Intelligence Techniques." Spectrum of engineering sciences 3, no. 1 (2025): 22-65.

[38] Asif Raza, Inzamam Shahzad, Ghazanfar Ali, and Muhammad Hanif Soomro. "Use Transfer Learning VGG16, Inception, and Reset50 to Classify IoT Challenge in Security Domain via Dataset Bench Mark." Journal of Innovative Computing and Emerging Technologies 5, no. 1 (2025).

[39] Raza, A., Salahuddin, & Inzamam Shahzad. (2024). Residual Learning Model-Based Classification of COVID-19 Using Chest Radiographs. Spectrum of Engineering Sciences, 2(3), 367–396.

[40] Raza, Shoukat, N., Salahuddin, Aslam, M., & Shahzad, I. (2024). DETECTION OF DIABETES APPLYING MACHINE LEARNING TECHNIQUE. Kashf Journal of Multidisciplinary Research, 1(11), 107-124

[41] Salahuddin, Hussain, M., & hamza Shafique, P. (2024). Performance analysis of matched filter-based secondary user detection in cognitive radio networks. Kashf Journal of Multidisciplinary Research, 1(10), 15-26.

[42] Salahuddin, Syed Shahid Abbas, Prince Hamza Shafique, Abdul Manan Razzaq, & Mohsin Ikhlaq. (2024). Enhancing Reliability and Sustainability of Green Communication in Next-Generation Wireless Systems through Energy Harvesting. Journal of Computing & Biomedical Informatics

[43] Salahuddin, Abdul Manan Razzaq, Syed Shahid Abbas, Mohsin Ikhlaq, Prince Hamza Shafique, & Inzimam Shahzad. (2024). Development of OWL Structure for Recommending Database Management Systems (DBMS). Journal of Computing & Biomedical Informatics, 7(02).

[44] Khan, S. U. R. (2025). Multi-level feature fusion network for kidney disease detection. Computers in Biology and Medicine, 191, 110214.

[45] Khan, S. U. R., Asif, S., & Bilal, O. (2025). Ensemble Architecture of Vision Transformer and CNNs for Breast Cancer Tumor Detection From Mammograms. International Journal of Imaging Systems and Technology, 35(3), e70090.

[46] Nataraj, L., Karthikeyan, S., Jacob, G., & Manjunath, B. S. (2011). Malware images: visualization and automatic classification. *ACM Workshop on Visualization for Cyber Security.

[47] Kalash, M., Rochan, M., Mohammed, N., Bruce, N. D., Wang, Y., & Iqbal, F. (2018). Malware classification with deep convolutional neural networks. IEEE ICMLA, 879–88.

[48] Khan, M., Baig, D., Khan, M. U. S., & Karim, A. (2020). Malware Classification Framework using Convolutional Neural Network. In 2020 International Conference on Cyber Warfare and Security (ICCWS) (pp. 1–6). IEEE. https://doi.org/10.1109/ICCWS48432.2020.9292384