Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.083998

ARTICLE

A Multi-Modal Deep Learning Framework for Robust Polymorphic Malware Detection

Phil Steadman*, Paul Jenkins, Rajkumar Singh Rathore* and Chaminda Hewage

Cardiff School of Technologies, Cardiff Metropolitan University, Cardiff, UK

*Corresponding Authors: Phil Steadman. Email: st20280948@outlook.cardiffmet.ac.uk; Rajkumar Singh Rathore. Email: rsrathore@cardiffmet.ac.uk

Received: 14 April 2026; Accepted: 04 June 2026; Published: 23 July 2026

ABSTRACT: Modern malware is increasingly employing polymorphism, packing, and metamorphism to evade traditional signature-based detection. Because of this, there is an urgency to have more reliable classification systems. Visual malware analysis, where binaries are converted into grayscale images, has demonstrated potential in revealing structural patterns of malware family classification. However, recent methods mostly rely on single-stream, lightweight Convolutional Neural Networks (CNNs). These models have a major blind spot. The visual representation textures can be heavily obscured without changing the underlying malicious code, causing severe performance drops on newer or even rare malware classes. This paper presents a Hybrid Multi-Modal Deep Learning framework to fix this vulnerability. The proposed dual-stream architecture uses image recognition via EfficientNetB0 alongside metadata analysis using 1D-convolutional byte embeddings. This paper evaluated the framework on the modern MalwareVision-2025 dataset (approximately 125,000 samples) and the legacy Malimg dataset (9339 samples). On MalwareVision-2025, the model reached a weighted accuracy of 88.44% and achieved 100% benign recall on the evaluated split. The testing across both datasets shows that combining visual and structural features reduces modality collapse. This creates a much stronger system compared to using either input type on its own. In particular, the hybrid approach improves detection performance on deeply hidden contemporary threats, including the AveMariaRAT and CobaltStrike families.

KEYWORDS: Polymorphic malware; deep learning; multi-modal; convolutional neural networks; EfficientNetB0; cybersecurity; MalwareVision-2025; Malimg

1 Introduction

The cybersecurity landscape is constantly changing due to new methods being developed by attackers. Existing traditional malware detection systems primarily rely on methods such as static signature-based detection and heuristic detection. These traditional methods are now struggling to accurately respond to the rapid proliferation of modern malware, with estimates of around 560,000 new variants generated each day in addition to an existing repository of more than 1 billion types of malware [1]. More than 90% of today's malware operate using polymorphic/metamorphic evasion techniques to avoid detection and are created using packers, crypters and oligomorphic/polymorphic engines. These procedures continuously alter the binary data of executables while maintaining the core capabilities to deliver a malicious payload [2].

Many researchers have examined how best to address this issue, and many have turned to deep learning (DL) as the solution. One of the more commonly used approaches is the decomposition of malware images, which was first introduced by Nataraj et al. (2011) [3]. In the Malware-as-an-Image approach, raw binary

executable data is transformed into a two-dimensional greyscale image. The structure of the image (visual texture and replication) can be used with Convolutional Neural Networks (CNN) to extract the feature characteristics needed to successfully classify a particular sample of malware. These individualised visual features can almost always be identified as unique human fingerprints.

In the past, this approach worked very well, but additional examination of DL research concerning the classification of malware using only visual representations conducted between the years of 2020 and 2026 [4] will reveal two key limitations of the strictly visual classification method for malware: 1. By placing too great an emphasis on visual characteristics as an identification factor for malware, researchers created a major disadvantage because, during the packing process [5], the end product results in high uniformity within the context of visual data. As a result, the feature extraction properties of the CNN become compromised and ineffective. 2. In addition to further restricting the utility of the strictly visual classification method, to date the majority of artefacts used in studies of the visual portrayal of malware were derived from earlier datasets, such as Malimg (2011) or Microsoft BIG 2015. These two data sources exhibit significant concept drift i.e., the absence of modern malware families (i.e., agenttesla, async RAT, cobalt strike) and do not provide characterisation characteristics for contemporary malware detection techniques. Consequently, models trained with these datasets have demonstrated limited capabilities and perform poorly when compared to actual operational conditions and face the challenge of detecting new (novel) threat classes.

Apart from the technical weaknesses associated with the use of a single stream-based approach and the related limitations in classifying and identifying malware, there is an operational challenge to consider—alert fatigue for lines of security operations personnel working within live security operations centre environments (SOCs). Traditionally, and often through the use of simplified and non-optimised DL models of malware detection, the ability to differentiate between the highly obfuscated nature of malware and benign executables is limited. The high volume of false positive alerts created by a traditional intrusion detection system may preclude a sufficient number of qualified human analysts from being able to keep pace with providing the necessary level of attention needed to determine the nature of the malicious activity contained in an arbitrary executable file. Hence, any contemporary detection system must not only be able to maintain a high classification performance with novel threat classes but must also provide a near-perfect recall on benign (safe) files to show strong potential for reducing alert fatigue and ensuring continued viability within enterprise, personal, and small-scale environments. Although packing and crypting systems can effectively obscure the content of the executable body, they cannot totally eliminate all traces of the structural diagram associated with the executing binary and the supporting information contained within the file's metadata. Even if an executor was to apply extensive compression to a fully encrypted polymorphic payload, the subordinate properties of the executable header and metadata still provide a semi-valid entry point to allow the core functionality of the polymorphic file to function in a compatible manner when the host operating system executes the executable binary.

Fig. 1 demonstrates the reasons for the inability of traditional antivirus solutions to detect contemporary forms of malware (upper line). When malware enters a malware detection system, it is evaluated entirely based upon the visual representation of the executable by a standard single stream CNN. The use of packing techniques to obscure a significant portion of the binary code results in the AI effectively losing the feature identification to differentiate two pieces of malware, creating a false negative detection, allowing the malware to evade detection. Conversely, the proposed Hybrid approach of this paper addresses this weakness by always observing the visual characteristics and the structural attributes of the malware at the same time (lower line). As a result, the system is able to map the two data types together. Therefore, whether the visual data is obscured by a packing characteristic, the structural characteristics still detect the anomaly represented by the malware and trigger a successful detection process to generate a secure endpoint.

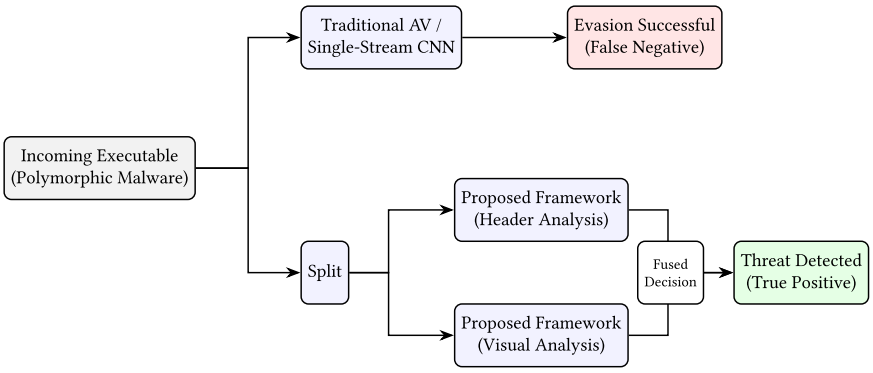

Figure 1: Operational scenario illustrating classic detection failure vs. the proposed hybrid detection process.

1.1 Paper Organisation

Section 2 provides a literature review of legacy single-stream methods vs. more contemporary multi-modal classification approaches. Section 3 describes the proposed methodology by introducing a dual-stream architecture and Categorical Focal Loss. Section 4 provides a description of how the setup, hardware specs, and computational scaling were completed. Section 5 presents the results, including a comparative analysis of classification performance, precision-recall characteristics, and visual interpretability. Finally, Section 6 concludes the paper with a summary of the findings and discussion on the potential for future research.

1.2 Research Contributions

This Research introduces a Hybrid Multi-Modal Deep Learning framework. The major contributions of this research are:

- The Dual-Stream architecture was benchmarked against both the MalwareVision-2025 and the previous Malimg datasets. This allows for more appropriate classification benchmarks in regards to the modern threat landscape and to also demonstrate the evolutionary progression of malware through advances in compilers and obfuscation technologies.
- This Research introduces a multi-head architecture that uses visual feature extraction via an EfficientNetB0 backbone in parallel with a unique 1-Dimensional Convolutional embedding stream. The combination of these two streams captures both the visual texture of the binary as well as the metadata (structure) surrounding it.
- The ablation studies [6] produce clear evidence that there are inherent problems associated with the use of older datasets and their propensity to bias deep learning models through excessive reliance on a single modality type [7]. Compared to single modality approaches, the proposed multi-modal method outperformed and produced higher accuracies for the hidden minority malware classes, demonstrating robust security potential by achieving 100% benign recall on the evaluated split.

2 Literature Review

The use of machine learning and, particularly, deep learning for malware classification has evolved greatly over the last decade. In particular, the work of Nataraj et al. (2011) laid the groundwork for "Malware-as-an-Image" through demonstrating that converting binary executable files into grayscale images (and using the images as input to the model) yields visual representations of distinct textures across malware families [3].

Single modality approaches have been successful in the past (e.g., Malimg) with older dataset types, but they have been less successful against the increasing use of polymorphism and packing techniques that effectively destroy the ability to identify visual signatures of these modern malware varieties.

To improve visual classification performance, researchers developed an ensemble of Convolutional Neural Networks (IMCEC) to extract higher-quality features from input images. Hemalatha et al. [8] also demonstrated that use of DenseNet-based models with an appropriate loss function helped improve classification accuracy on imbalanced datasets. However, both systems are solely based on image processing, which limits their success to scenarios where the image is still able to represent valuable information visually; thus, their ability to cope with obfuscation techniques that convert an image into pure random noise inhibits their performance.

Consequently, researchers have begun to focus their attention towards developing multi-modal architectures. An example of this is DSMalConv from Li et al. [7]. This model combines images of malware (in grayscale) with Attributed Control Flow Graph (ACFG) data to create representations of malware based on both visual patterns and process flow dependency. Due to the high computational requirements needed to create ACFGs, the model complexity necessitates resources that render it difficult to be rapidly deployed into real-time edge IoT environments. Related hybrid designs have also been explored beyond Windows PE malware. For example, Dong et al. [9] proposed an Android malware detection method combining CNN and DNN components, reinforcing the broader trend towards hybrid feature learning while targeting a different platform and threat representation.

The framework introduced herein builds upon the progress made thus far; with the introduction of an extremely efficient Dual-Stream architecture. In this model, costly Gated Channel Transformation (GCTs) are eliminated in favour of combining an advanced visual analysis system (EfficientNetB0) with a low-cost 1-Dimensional CNN that accepts raw header bytes directly. As evidenced in Table 1, the result is a continued ability to process large quantities of data while also not suffering from the typical modality collapse associated with the use of purely visual-based systems.

Table 1: Comparison of published works with the proposed framework.

Authors (Year)	Approach	Dataset(s)	Key Limitation/Challenge	Multi-Modal
Nataraj et al. (2011) [3]	Image-only (KNN)	Malimg	Fails on packed/encrypted malware	No
Hemalatha et al. (2021) [8]	DenseNet with balanced loss	Malimg, BIG 2015	Relies solely on visual representation	No
Li et al. (2026) [7]	Image + ACFG Fusion	Cross-architecture	ACFG extraction is computationally heavy	Yes
Dong et al. (2024) [9]	CNN + DNN hybrid	Android malware	Targets Android features rather than executable header/image fusion	Yes

(Continued)

Table 1 (continued)

Authors (Year)	Approach	Dataset(s)	Key Limitation/Challenge	Multi-Modal
Proposed Work	Hybrid (EfficientNetB0 + 1D CNN)	MalwareVision-2025 & Malimg	Strong benign recall; robust to polymorphism	Yes

3 Methodology

In order to conduct a comprehensive evaluation of the system's efficacy based on a variety of different threat environments, this research employed two separate data sets: Malimg and MalwareVision-2025.

Malimg is an industry-standard dataset used for visual malware classification that was proposed by Nataraj et al. in 2011 [3]. It consists of 9339 samples from 25 different families of malware. While this dataset successfully classified binary textures in the context of image-based computer vision, there are no current examples of modern obfuscation techniques that achieve the same level of performance for visual signatures as in earlier research. After multiple years of ongoing effort to increase classification accuracy, it has now virtually reached its maximum potential. For this research, this research will be using Malimg as a baseline for determining backward compatibility and also using it to illustrate a phenomenon labelled as "Mode Collapse" [10].

MalwareVision-2025 [11] is a recently released dataset comprising a repository of approximately 125,000 unique malware samples, including modern forms of polymorphic malware (e.g., Formbook, CobaltStrike, and AgentTesla), and benign executable samples (e.g., kernel32.dll). The experiments in this paper modelled 22 output classes, covering benign executables and the malware families retained after preprocessing and label harmonisation. The processed experimental subset contained 92,388 records in the training/validation CSV and 1869 records in the held-out test CSV. The public Kaggle metadata does not provide reliable per-sample packer or crypter labels; therefore, the dataset is treated as a family-labelled modern malware benchmark rather than as a packer-attributed corpus. This limitation is important because the observed obfuscation behaviour is inferred from family-level characteristics, byte/image structure, and classifier response rather than from ground-truth packer annotations. As the volume of research involving the MalwareVision-2025 data set has not yet reached any significant research papers, the potential to establish current, novel benchmarks for modern malware threat detection is significant. Table 2 summarises the held-out MalwareVision-2025 test-set class distribution used for final evaluation.

The system will utilise a two-stream neural network, which allows for simultaneous examination of both visual and non-visual features in parallel, as depicted in Fig. 2 (i.e., architecture). Each stream will be discussed in detail below.

3.1 Stream A—Extracting Visual Features

The Visual Features Extraction Stream is the first stream of data. The Visual Feature Extraction Stream converts the raw binary code to a 2D grayscale image; this allows the system to examine the structural arrangement of the code blocks and also analyse the byte structure of the code pattern [12]. The model selected for use in this data pipeline is EfficientNetB0, this model was selected because of its highly optimised Compound Scaling [13]. EfficientNet's Compound Scaling is different from just increasing the depth or the width of a network randomly; instead, it balances the size of depth, width, and resolution uniformly. Because of this Compound Scaling method, the EfficientNetB0 model is capable of accurately detecting

highly complex visual features with very little load on the computer system. The EfficientNetB0 model has been evaluated and found to be very accurate at detecting fine-grained visual and spatial texture anomalies.

Table 2: Held-out MalwareVision-2025 test-set class distribution used for final evaluation.

Class	Support	Class	Support	Class	Support
AgentTesla	311	Amadey	39	AsyncRAT	22
AveMariaRAT	39	Benign	74	CobaltStrike	39
Formbook	39	GCleaner	39	GandCrab	39
Gozi	39	GuLoader	39	Heodo	319
IcedID	39	Loki	39	LummaStealer	40
Mirai	479	NanoCore	39	Prometei	39
RedLineStealer	39	RemcosRAT	39	SilentBuilder	39
SmokeLoader	39	**Total**	1869		

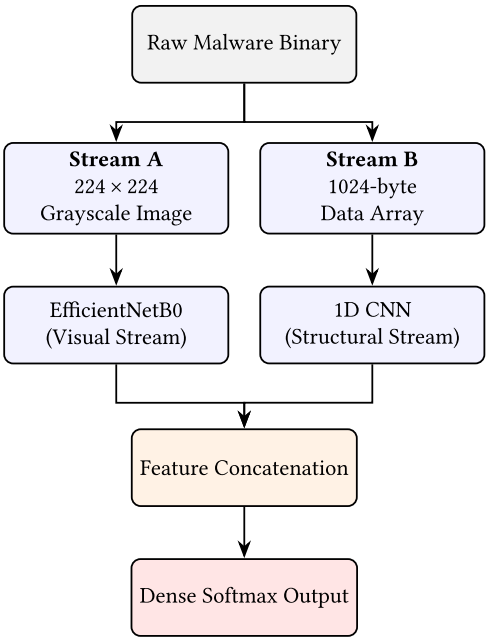

Figure 2: The proposed hybrid multi-modal architecture processes visual and structural streams in parallel before feature concatenation and softmax classification.

When a 224×224 pixel image is converted to a compact 128-length vector of visual descriptors (features) through the EfficientNet architecture, the result is a condensed vector regarding the various visual characteristics of the malware, the mathematical expression for this is as follows:

$$F_{img} = \phi_{dense}(\phi_{Eff}(I_{224\times224}; \theta_{Eff}); \theta_{dense}) \tag{1}$$

Here, $I_{224\times224}$ denotes the resized grayscale malware image, ϕ_{Eff} represents the EfficientNetB0 convolutional feature extractor parameterised by θ_{Eff}, and ϕ_{dense} denotes the fully connected projection layer parameterised by θ_{dense}. During training, the visual and structural streams are optimised within the same model, and their learned representations are later combined to produce the final classification output.

The EfficientNetB0 backbone is trained within the visual stream using parameters θ_{Eff}. To achieve the compact representation, the output of the base model is passed through a fully connected dense layer ϕ_{dense}. This combined trainable branch results in a condensed feature vector $F_{img} \in \mathbb{R}^{128}$ representing the visual characteristics of the malware based on the information provided in the 224 × 224 image.

3.2 Stream B—Extracting Structural Features

The Structural Feature Extraction Stream processes the early header and byte-level metadata of the executable. While visual textures are vulnerable to intense packing or metamorphism, early structural metadata required by the operating system to successfully load and execute the file often remains informative.

To process this data, the first 1024 bytes of the file's raw header are extracted and fed into a custom 1-Dimensional Convolutional Neural Network (1D CNN). The 1024-byte limit reflects a fixed-width engineering threshold for capturing early executable metadata. For Windows PE samples, this region typically includes the MZ DOS stub, PE signature, COFF file header, Optional Header, and the beginning of the section-table region. Header lengths are not fixed across compilers, linkers, and packers because the DOS stub, optional data directories, section table, and nonstandard fields can vary. However, the first 1024 bytes provide a reasonably stable representation while avoiding the computational overhead and padding noise associated with substantially larger byte windows. Because the dataset does not provide authoritative packer-specific header-length annotations, the 1024-byte value is reported as an engineering threshold rather than a statistical percentage of all possible headers. The integer byte values are first passed through an embedding layer to map the discrete byte values into dense 128-dimensional continuous vectors. This is followed by two 1D convolutional blocks designed to capture local structural dependencies, interleaved with max-pooling to downsample the spatial dimension and extract the most prominent structural anomalies.

The exact architectural configuration of the 1D CNN is detailed in Table 3.

Table 3: Architecture of the 1D CNN (structural stream).

Layer Type	Output Shape	Kernel/Pool Size	Filters/Units
InputLayer	(None, 1024)	–	–
Embedding	(None, 1024, 128)	–	128
Conv1D (ReLU)	(None, 1022, 64)	3	64
MaxPooling1D	(None, 511, 64)	2	–
Conv1D (ReLU)	(None, 509, 128)	3	128
GlobalMaxPooling1D	(None, 128)	–	–
Dense (ReLU)	(None, 128)	–	128

The output of the Global Max Pooling operation is passed through a final fully connected Dense layer to produce the structural feature vector $F_{hdr} \in \mathbb{R}^{128}$.

3.3 Feature Fusion and Categorical Focal Loss

To achieve the multi-modal classification, the visual feature vector $F_{img} \in \mathbb{R}^{128}$ and the structural feature vector $F_{hdr} \in \mathbb{R}^{128}$ are combined using straightforward vector concatenation. This results in a fused feature representation:

$$F_{fused} = F_{img} \oplus F_{hdr} \quad \text{where} \quad F_{fused} \in \mathbb{R}^{256} \tag{2}$$

This fused vector is passed through a fully connected Dense layer consisting of 256 units, followed by a Dropout layer to mitigate overfitting. Finally, the network utilises a Dense output layer with 22 units (corresponding to the 22 malware and benign classes) and a softmax activation function to output the final probability distribution.

To train the model effectively across highly imbalanced datasets with both legacy and contemporary polymorphic threats, Categorical Focal Loss was employed instead of standard Cross-Entropy Loss. Focal loss dynamically scales the loss based on prediction confidence, heavily penalising misclassifications on hard examples (e.g., deeply obfuscated novel threats) while down-weighting the loss contribution from easy, well-represented classes (e.g., standard legacy malware). The standard hyperparameters for Focal Loss were selected: the focusing parameter was set to $\gamma = 2.0$ to smoothly adjust the rate at which easy examples are down-weighted, and the balancing parameter was set to $\alpha = 0.25$ to handle class frequency disparities.

3.4 Addressing Class Imbalance in Cybersecurity

In the world of cybersecurity, there is a long-standing class imbalance between the number of threats detected in the field vs. the number of threat types found. Many malware families are highly common, while there are very few rare and innovative types of malware. When a neural network trains on a highly imbalanced dataset, there is a high possibility that the model will develop strong biases based on training on the common malware families. This means that the neural network will learn to achieve a high overall accuracy rate based on its ability to continue to predict the common classes of threat highly aggressively and, as a result, will perform extremely poorly when attempting to identify the rare and or less frequently found malware classes.

To address this potential issue, the proposed solution for the MHS would utilise a Categorical Focal Loss function to optimise the training of the neural network and provide a more equitable distribution of training resource allocation. The proposed solution to the Categorical Focal Loss function would modify the manner in which the neural network improves based on its certainty of correctness (or high level of certainty). Therefore, if the γ parameter is set to $\gamma = 2.0$, then the model will cease to be rewarded for predicting common and easy-to-detect examples correctly, while conversely, it will be very heavily penalised for predicting an instance incorrectly, especially on exceptionally hard-to-predict instances. Thus, forcing the neural network to stop taking the easiest path of success and instead start to focus its full analytical and predictive capabilities on identifying rare, unique malware instances that are hidden from standard and less effective forms of malware identification.

The exact Python implementation of the Categorical Focal Loss function utilised in this framework is provided in Appendix A.

Categorical Focal Loss was created to reduce the impact of penalties on simple examples. By doing this purposefully, it forces the model to start to learn really well how to deal with the hardest cases.

4 Experimental Design and Computational Scale

This section provides a brief description of the hardware and software environment used in the data pipeline. It leverages the desktop setup to protect the GPU from memory saturation by using a typical desktop workstation.

The framework was constructed and evaluated using a standard Ubuntu 24.04 LTS Desktop Computer with an NVIDIA GeForce RTX 2060 SUPER GPU and 64 GB of DDR4 RAM. The code was built in JupyterLab to execute the processing pipeline. The machine used Keras/TensorFlow 2.18.0 in conjunction with Scikit-learn for evaluating the model.

The MalwareVision-2025 dataset has over 125,000 files, and loading the entire dataset at once would result in an unusable system. For that reason, a custom Python data generator was built that loads the required training data into (RAM/GPU) only at the time it is needed. The DataLoader implemented the Keras Sequence pattern rather than relying on a fully materialised array in memory. Specifically, the DataLoader used the image filename and class folder for the current sample, loaded and resized the RGB image, read the precomputed header features h_0–h_1023 in integer format from the CSV, and returned the image tensor, header tensor, and encoded one-hot class label. Thus, this architecture preserved deterministic order per epoch if required while allowing for thread-safe generation of batches. Multi-threading was also employed to minimise I/O bottlenecks and improve GPU utilisation during training.

The above can be summarized for the DataLoader as follows: for each shuffled batch index, collect the corresponding filenames and labels; load the image tensor from the correct class folder; resize and normalize the image tensor; collect the CSV header columns h_0–h_1023; encode the class label; and return ([image_batch, header_batch], label_batch) to Keras.

4.1 Training Hyperparameters

The model was optimised with Adam Optimiser and began with a learning rate of 0.001. Training occurred in batches of 32 samples for 50 epochs. To prevent overfitting and promote optimal convergence, an early stopping mechanism was used to halt training if the validation loss had not improved for five continuous epochs. For MalwareVision-2025, the processed training CSV contained 92,388 records and was split into stratified training and validation subsets using 80% for training, 20% for validation, and a fixed seed (s = 42). A separate held-out test CSV contained 1869 records. The held-out test set contained 479 Mirai samples, 319 Heodo samples, 311 AgentTesla samples, 74 benign samples, 40 LummaStealer samples, 22 AsyncRAT samples, and 39 examples for each of the remaining 16 minority families. Thus, this design maintains class proportions during training and validation while keeping the held-out test set isolated from early stopping and model selection.

Ablation experiments were conducted using Image-Only, Header-Only, and Hybrid baselines with the same training/validation/testing partitions, random seed, optimiser, batch size, early-stopping mechanism, and Categorical Focal Loss settings. The only significant differences were the input streams and architecture branches. Therefore, the ablation results can be attributed to modality effects rather than different training methods.

5 Experimental Results and Discussion

This section will discuss the experimental results. This includes the classification performance results, precision-recall results, and how well the model balanced the diversity of the feature streams, along with heat maps showing what areas of the malware the AI uses to evaluate.

5.1 Classification Performance and Operational Advantage

The Hybrid architecture performed well; achieving a weighted accuracy of 88.44% on the very obfuscated dataset MalwareVision-2025. In live scenarios such as Security Operations Centres (SOCs), the ability to differentiate between benign files and malicious files with a very high degree of accuracy is critical. A high false-positive rate will often result in alert fatigue, and with thousands of erroneous alerts, it causes the analyst's ability to perform well to decrease and also increases the likelihood that they miss an actual threat. As detailed in Table 4, the proposed model achieved a Recall score of 1.0 (100%) on benign files in the evaluated split. Fig. 3 shows the benign-vs.-malware confusion matrix for the combined-input model.

Table 4: Classification report (expanded high-priority families).

Family	Precision	Recall	F1-Score
Benign	0.987	1.000	0.993
AgentTesla	0.780	0.887	0.830
GandCrab	0.975	1.000	0.987
GCleaner	0.907	1.000	0.951
CobaltStrike	0.850	0.872	0.861
AsyncRAT	0.636	0.636	0.636
Formbook	0.686	0.615	0.649
AveMariaRAT	0.679	0.487	0.567
Weighted Avg.	0.882	0.884	0.881

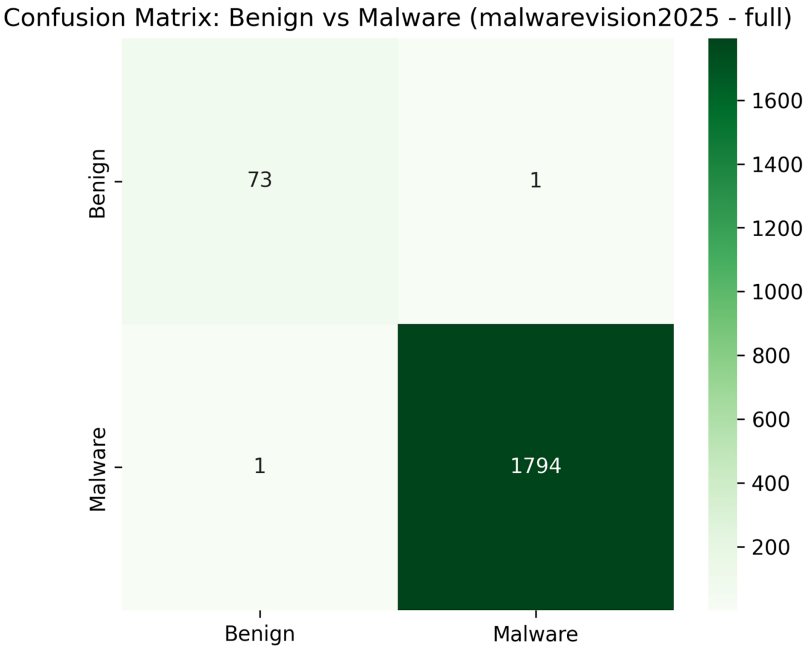

Figure 3: Confusion matrix (Benign vs. Malware)—using both inputs.

5.2 Expanded Performance Metrics by Dataset

To conduct a more thorough evaluation of the proposed Hybrid architecture, other performance measures were obtained, including Overall Accuracy, Macro-Averaged F1-Scores, and Weighted-Averaged F1-Scores over both datasets.

The Macro-Averages are presented in Table 5, these values illustrate the limitations of standalone or one-modal type architectures. The Images-Only model under the legacy Malimg dataset exhibited a catastrophic failure; it achieved an overall accuracy of only 15.61% with a Macro F1-Score of 0.097. The Headed-Only stream created a strong baseline for comparison with an accuracy of 94.80%. However, incorporating the use of both the Images and Header streams into the Hybrid architecture yielded a peak Weighted F1-Score of 0.967.

Table 5: Performance metrics expanded by modality and dataset.

Metric	Image-Only Stream	Header-Only Stream	Hybrid Framework (Combined)
Dataset: Malimg (Legacy)			
Overall Accuracy	15.61%	94.80%	97.69%
Macro Avg Precision	0.102	0.868	0.889
Macro Avg Recall	0.140	0.885	0.920
Macro Avg F1-Score	0.097	0.873	0.902
Weighted Avg Precision	0.112	0.931	0.959
Weighted Avg Recall	0.156	0.948	0.977
Weighted Avg F1-Score	0.113	0.937	0.967
Dataset: MalwareVision-2025 (Modern)			
Overall Accuracy	82.50%	86.68%	88.44%
Macro Avg Precision	0.750	0.785	0.809
Macro Avg Recall	0.679	0.738	0.782
Macro Avg F1-Score	0.698	0.747	0.791
Weighted Avg Precision	0.825	0.861	0.882
Weighted Avg Recall	0.825	0.867	0.884
Weighted Avg F1-Score	0.817	0.856	0.881

All types of modern obfuscation utilise an entirely different method of new obfuscation tech; therefore, they all operate poorly under the MalwareVision-2025 dataset. As a result, those types of shallow CNNs cannot go any deeper (Image-Only stream bottlenecked at an Overall Accuracy of only 82.50%). However, the Dual-Stream Architecture improved the overall accuracy to 88.44%, with a corresponding Weighted F1-Score of 0.881. The Hybrid's Macro F1-Score of 0.791 demonstrates how incorporating an additional data type improves detection quality on rare/heavily obfuscated malware classes.

5.3 Precision-Recall Correlation and Intra-Class Variance

To evaluate how the Hybrid architecture performed on imbalanced minority classes, it was important to evaluate the performance using Precision-Recall (PR) and Receiver Operating Characteristic (ROC) curves. Fig. 4 is the PR curve, which compares the model's ability to detect true positives successfully (Precision), as well as its ability to capture all actual targets (Recall). Fig. 5 graphically depicts the ROC curve, plotting True Positive Rate (TPR) against False Positive Rate (FPR) (Note: the closer to the top left corner of the graph, the better the prediction performance).

When looking at the AveMariaRAT family/enhanced metrics, it shows clearly that static visual analysis architectures do not work. The orange PR curve in Fig. 4 starts with a perfect Precision of 1.0 but then it falls quickly as the Recall increases (from 0.15 to 1.0) due to the multiple methods these RATs use to hide the actual content in files. As a result, the files look different from one another, while simultaneously they also look identical to entirely different malware families such as Formbook. Both RAT families have been used in phishing-type campaigns and are often classified as such because of Steganography [14,15].

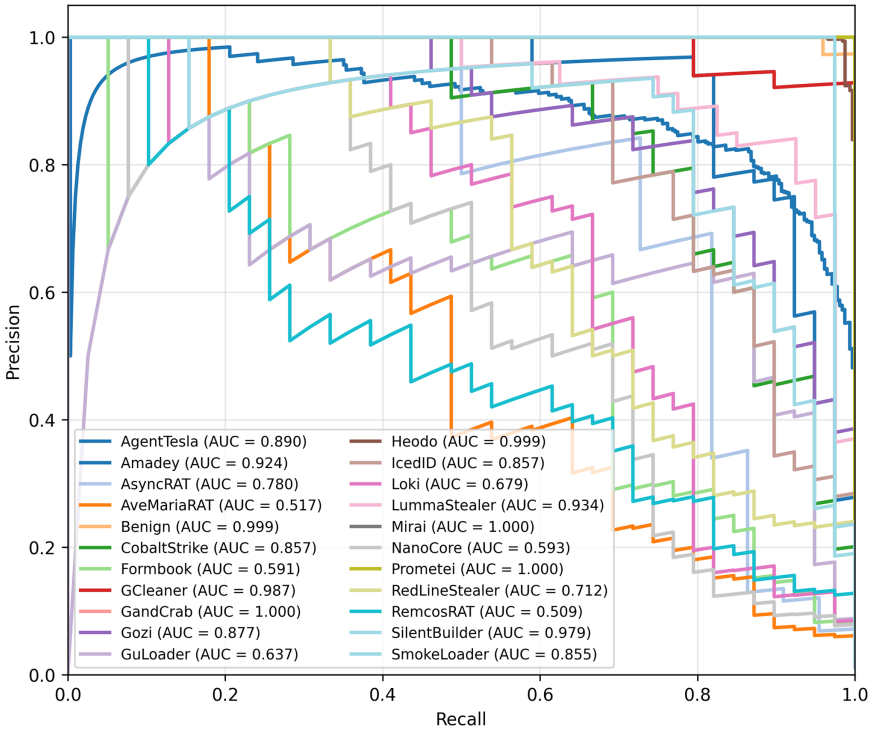

Figure 4: Precision-Recall curve illustrating intra-class variance.

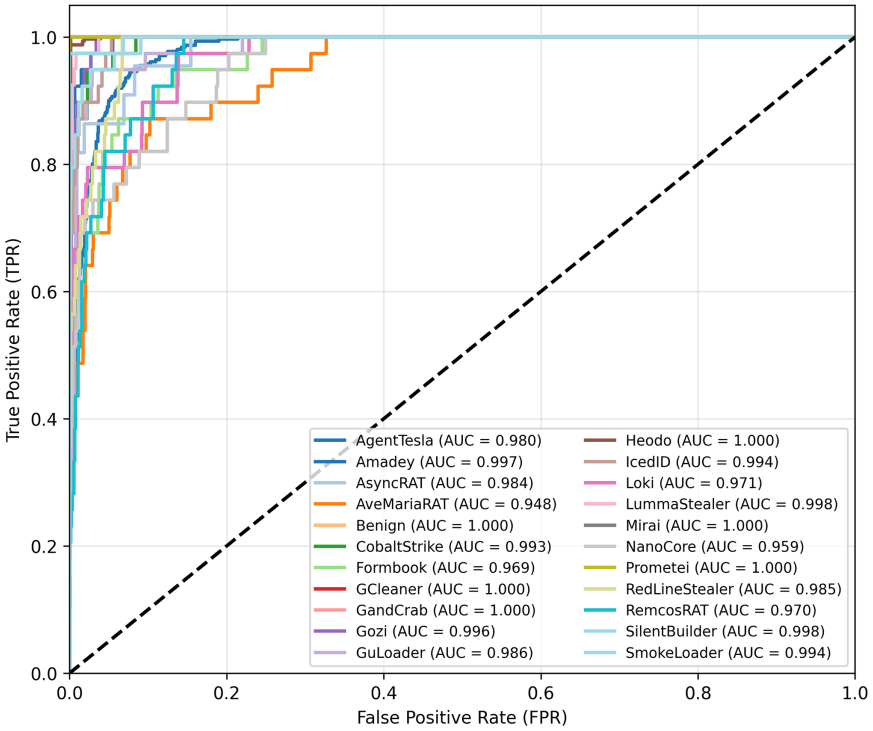

Figure 5: ROC curve illustrating intra-class variance.

5.4 Modal Variance across Datasets

To validate the necessity of the Hybrid architecture in detection technology, an ablation study was executed over both datasets: Malimg and MalwareVision-2025.

Previous research [4,16,17] shows that lightweight CNNs are suitable for classifying older datasets such as Malimg using a visual encoding technique on uncompressed textures; however, this algorithm is not effective for classifying newer datasets that contain very advanced or heavily obfuscated formats. Time to transition to a deeper architecture means that the pixelated historical images will be transformed into a very low; hence, the major 224 × 224 image size becomes the major problem, leading to the prediction bias in the majority-class (illustrated in Table 6). Given the drastic change in the dataset's variance, it is clear that contemporary endpoint security should not rely solely on visual streams.

Table 6: Ablation study: F1-scores across modalities and datasets.

Dataset	Malware Family	Image-Only	Header-Only	Hybrid
Malimg (Legacy)	Applee.A	0.000	0.965	1.000
	Allaple.L	0.000	0.954	1.000
	Yuner.A	0.992	0.985	0.998
	Lolyda.AA1	0.000	0.941	0.988
	Lolyda.AA2	0.000	0.930	0.980
	C2LOP.P	0.000	0.972	0.995
	Swizzor.gen!I	0.000	0.920	0.975
	Dialplatform.B	0.850	0.990	0.995
	Fakerean	0.000	0.988	1.000
	Alueron.gen!J	0.000	0.970	0.991
MalwareVision	AveMariaRAT	0.431	0.259	0.567
	Formbook	0.529	0.517	0.649
	GCleaner	0.938	0.902	0.951
	CobaltStrike	0.685	0.805	0.861
	AgentTesla	0.729	0.812	0.830
	AsyncRAT	0.345	0.727	0.636
	RedLineStealer	0.406	0.554	0.571
	RemcosRAT	0.419	0.522	0.588
	Loki	0.486	0.302	0.559
	NanoCore	0.392	0.556	0.594

A significant finding from this study was that the Image-only modality had a complete absence of performance (F1-score = 0) for the majority of minority malware families in the legacy Malimg dataset (e.g., Applee.A, Allaple.L, Lolyda.AA1). The low performance of this particular modality can be primarily attributed to the overfitting of the model to the legacy Malimg dataset due to the incompatibility of the deep learning framework and the characteristics of the dataset. Although earlier studies have been successful in classifying uncompressed legacy samples using shallow convolutional networks based on global visual texture, the framework proposed in this paper utilises EfficientNetB0, a deep convolutional network designed to extract high-resolution, localised detail. As such, EfficientNetB0 cannot find meaningful local features in low-resolution images, and therefore, it overfits to irrelevant microtextures. Therefore, the visual-only model was unable to learn how to differentiate the rare classes of malware.

The results of the ablation study indicated, however, that using the Structural (Header) modality resolved the problem with the Image-only modality. By combining the structural metadata stream with the visual stream, the Hybrid model was able to restore detection and achieve near-perfect F1-scores for the previously undetected families of malware.

As the transition to a modern MalwareVision dataset, neither stream holds total dominance (Table 6) due to the greater variability. The CNN changes its reliance between the two streams of features based upon how the malware is hidden. For AveMariaRAT, the image stream outperformed the header stream. For CobaltStrike and Lokibot, the header stream provided significantly greater accuracy. For Benign data the image stream outperformed the other methods. Thus, the results demonstrate that modern packers alter a file's internal structure and external texture in very different ways. Consequently, the Hybrid architecture acts as a stabilisation tool for modern detection.

5.5 Interpretation of Imaging Data and Visual Attention

Using the EfficientNetB0 backbone from the last layer and the Gradient-weighted Class Activation Mapping (Grad-CAM), the research demonstrated how the visual stream arrives at its decisions and identify the False Positive (FP) feature regions. Grad-CAM highlights the highly specific areas of a binary visual representation that directly influence the AI; thus, it provides a model for solving the deep learning black-box problem. Therefore, security analysts can easily verify whether their conclusions are correct about the presence or absence of structural anomalies in the model.

The attention maps confirm the assessment that the framework alters its focus depending upon how the malware variant was constructed. For example, in the Grad-CAM images in Fig. 6, the intense red area occupies the upper edge of the binary image. This is significant because the 1D-to-2D projection of these first few bytes corresponds to the file header and early structural region. Therefore, the visual stream can identify header-related patterns while reducing reliance on encrypted or obfuscated payload features. Conversely, in Fig. 7, again observing a significantly distinct activation from the early header region to the executable code and/or data segments corresponding to the middle and bottom of the binary image, respectively, as shown in the attention map in Fig. 7. This activation occurs because the network has successfully isolated the visual features that comprise the executable payload. Thus, these visual representations provide further evidence that the model successfully resolves and separates both the early header and executable body features depending upon the technique executed by the malware.

5.6 Comparison to State-of-the-Art Algorithms

To understand the utility of this work, this research compared the performance metrics against previous research as seen in Table 7.

Although many models claimed to have near-perfect accuracy rates from 98%–99%, it is important to consider the relative context of these studies. In this regard, prior studies in which near-perfect accuracy levels are reported (as in Agarap [18], Kalash et al. [19], Hemalatha et al. [8]) predominantly tested solely on legacy datasets (Malimg, MaleVis, or BIG 2015) that have no attributes reflective of contemporary packing and encryption methods. Additionally, the previous studies demonstrated the performance deficiencies of visually-based models when provided with highly disordered, obfuscated payloads, therefore lowering their real-world effectiveness against zero-day threats when compared to older benchmark studies.

Recent advancements in multimodal models report high F1-scores. However, each of the methods has a significant drawback; in particular, the DSMalConv architecture created by Li et al. [7] extracts Attributed Control Flow Graphs (ACFGs); this feature has a tremendous overhead in terms of computational

time. Likewise, Alsubai et al. [20] which achieved highly rated metrics regarding IoT malware examined extremely large ensemble models (YoloV7 and DenseNet161) and would be extremely difficult to implement on resource-constrained endpoints. Dong et al. [9] further demonstrate that CNN/DNN hybrid mechanisms can be effective in Android malware detection, although that setting uses a different executable ecosystem and feature space. In contrast, the EfficientNetB0 network was purposely designed for efficiency.

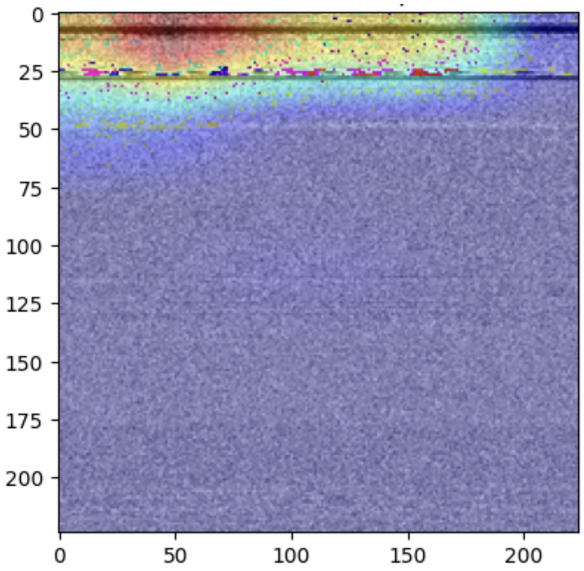

Figure 6: A Grad-CAM heatmap illustrating target header attention. The high activation (red) score highlights how well the model has captured the top of the PE header and associated DOS Stub, while completely bypassing the heavily obscured payload data.

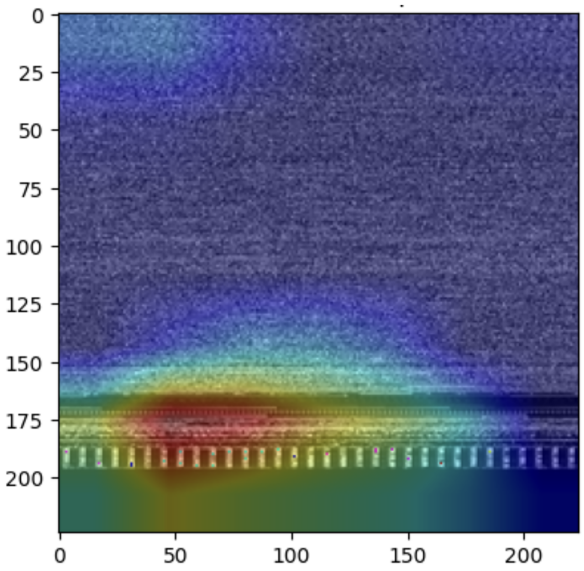

Figure 7: A Grad-CAM Heatmap illustrating target payload attention. Rather than having a focus on the header area as above, the model appears to have moved towards the lower region of the file where high activation corresponds closely to those areas represented on the visual display of properties associated with both executable binary body and data segments.

Table 7: Contextual comparison of the Hybrid Framework against related malware detection methods, separated by dataset setting to avoid implying a direct like-for-like comparison.

Study Year (Author)	Tested Dataset	Arch/Approach	Reported Accuracy	Reported F1-Score
Legacy image-based or earlier benchmark datasets				
Agarap (2019) [18]	Malimg	GRU-SVM (Unimodal)	84.92%	0.850
Kalash et al. (2018) [19]	Malimg	Deep CNN (Unimodal)	98.52%	N/A
Hemalatha et al. (2021) [8]	MaleVis	DenseNet (Unimodal)	98.21%	N/A
IoT, Android, and cross-architecture settings				
Alsubai et al. (2023) [20]	IoT Malware	YoloV7 + DenseNet161	98.65%	0.985
Dong et al. (2024) [9]	Android malware	CNN + DNN hybrid mechanism	96.80%	N/A
Li et al. (2026) [7]	Cross-architecture	DSMalConv (Multimodal)	99.76%	0.997
Modern executable benchmark used in this paper				
This Study	MalwareVision-2025	EfficientNetB0 + 1D CNN (Hybrid)	88.44%*	0.881*

Note: *Evaluated on a modern executable dataset with obfuscated families; achieved 100% benign recall on the evaluated split. Results across table groups are contextual rather than directly comparable because datasets differ in age, platform, class composition, and obfuscation difficulty.

The Hybrid Framework achieved a weighted accuracy of 88.44% and a weighted F1-score of 0.881 on the more challenging MalwareVision-2025 dataset, with 100% benign recall under the evaluated split. This result should not be seen as proof that alert fatigue is eliminated in all operational deployments, rather, it indicates that the proposed model has strong potential to reduce false-positive-driven analyst workload under the tested conditions.

6 Discussion and Conclusions

Testing results indicate that modern malware is now too complex for single AI model approaches. Specifically, the actions of concealing malware make its resulting feature set indistinguishable from noise, making only 1D models ineffective against modern malware. As a solution, the research developed a Hybrid model consisting of a dual-stream architecture integrating the analysis of visual features from one stream while simultaneously using the other stream to analyse structural metadata to identify the actual methods and tools used to hide the malicious behaviour of the payload.

In the testing against the MalwareVision-2025 dataset, the tests obtained weighted accuracy rates of 88.44% with a weighted F1-Score of 0.881. Additionally, the evaluated split obtained 100% benign recall under the tested conditions, which is an essential factor for reducing alert fatigue within live emission SOC environments. The ablation studies confirmed that the exclusive reliance on either only the grey scale image or only the headers alone results in weaker performance against deeply concealed malware families such as AveMariaRAT and Formbook, while also reducing modality collapse through the dual-stream hybridising of these two streams.

AI-assisted code vulnerability detection and executable provenance analysis are not experimentally evaluated in this study. They are therefore treated as forward-looking applications rather than as validated claims of the current framework.

6.1 Adversarial Robustness and Human-in-the-Loop Adaptation

As the cybersecurity arms race escalates, attackers are increasingly leveraging AI to generate polymorphic malware variants at scale. An attacker utilising an LLM could theoretically generate thousands of evasive

binaries, each with distinct visual textures but identical malicious payloads. The hybrid model's reliance on intrinsic byte-level structural features inherently increases its robustness against such AI-powered visual obfuscation compared to single-stream visual models.

To maintain resilience against these rapidly evolving AI-generated threats, the framework would benefit from continuous retraining through a human-in-the-loop system. Implementing a Reinforcement Learning from Human Feedback (RLHF) loop within the SOC—where security analysts provide direct feedback on misclassified or anomalous samples—would allow the dual-stream model to dynamically adapt. Actively aligning the model using crowd-sourced expert feedback, similarly to techniques used to refine LLM code generation, ensures the system remains robust against novel vulnerabilities and emerging threat patterns.

6.2 Limitations

Although the Hybrid Multi-Modal approach shows significant potential for detecting new polymorphic malware, acceptable operational capabilities are inherent to the multi-modal data fusion process, but there are some recognizable limitations.

First, while the two parallel streams support the visual component by providing additional context to avoid vulnerabilities from packing and encryption as seen in traditional approaches to Dual-Stream architecture, a unique risk to the visual (EfficientNetB0) stream is the potential for targeted, adversarial machine-learning (ML) attacks. This is achieved when a highly "intelligent" attacker would be able to mathematically derive "imperceptible" binary patterns that would sectionally modify arbitrary pixels in the malware.exefile such that the visual classifier would predict the false-positive visually inspectable information with little or no change to the executability of the malware itself [21]. Patil et al. [21] specifically evaluated adversarial malware-image attacks against multiple model architectures, including EfficientNetB0. Their findings show that adversarial perturbations can reduce classifier confidence or induce an alternate classification. The current study does not perform an adversarial robustness evaluation; therefore, this remains an open validation requirement for future work.

Second, the data structure stream in the proposed model has been optimised around a fixed early-byte representation rather than a full executable-format parser. This gives the model a simple and efficient structural input, but it also means that format-specific interpretation is limited. To adapt the architecture more explicitly to ELF or Mach-O formats, the current byte window could be replaced or supplemented by format-specific structural streams. In practical terms, the relevant header region would need to be identified, the byte-window length adjusted to match ELF program/section headers or Mach-O load commands, metadata fields mapped into comparable tensors, and the structural branch retrained on labelled binaries from the target operating-system ecosystem.

Lastly, although this model was developed with the intention of operating efficiently within a Security Operations Center (SOC) using dedicated GPU processing hardware, the application of this model on constrained edge devices (i.e., IoT gateway devices and/or on single-board computers such as the Raspberry Pi) presents a challenge [4]. The memory use and latency of the EfficientNetB0 backbone of the model is currently greater than the computing thresholds that any SOC edge computing environment would have. Therefore, to support on-device inference for the IoT ecosystem the use of aggressive techniques such as post-training quantization and/or distillation techniques would be required to create an ultra-lightweight derivative of the current architecture.

6.3 Future Research Directions

In addition to exploring adversarial LLM-generated malware and RLHF, future technical research directions will focus on three key areas: First, the introduction of Dynamic Attention Modelling using Transformer-routed weighting to enable the architecture to establish which of the two streams maintains the highest importance and relevancy per analysed file. Second, evaluating the framework's susceptibility to adversarial machine learning techniques specifically focused on targeted perturbations applied at the pixel level with the intent of evading detection by EfficientNetB0. Finally, the architecture will be enhanced by introducing quantisation and model pruning techniques for minimising the footprint of dynamic device requirements while maintaining performance speed. A further future-work direction is to test whether the same dual-stream idea can assist with AI-generated executable provenance or CI/CD malware screening; these applications require separate datasets and experiments before any operational claim can be made.

Acknowledgement: Not applicable.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm that they contributed to the following: Conceptualisation, Phil Steadman; methodology, Phil Steadman; software development, Phil Steadman; validation, Phil Steadman, Paul Jenkins, Rajkumar Singh Rathore and Chaminda Hewage; formal analysis, Phil Steadman; investigation, Phil Steadman; resources, Phil Steadman; data management, Phil Steadman; writing the original draft preparation, Phil Steadman; editing/reviewing the writing of the manuscript, Phil Steadman and Paul Jenkins; visualization, Phil Steadman; investigation/supervision, Phil Steadman, Paul Jenkins, Rajkumar Singh Rathore and Chaminda Hewage. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The Malimg data set is publicly available from all standard scholarly research repositories and is a legacy data set that serves as an established benchmark. The MalwareVision-2025 data set used in this research paper is also publicly available on the Kaggle data repository [11]. To ensure experimental reproducibility the data partitions used stratified sampling and a fixed seed. The visual stream of data went through pre-processing where raw executable binary files were converted to 2-dimensional image files. The structural stream pre-processed data isolated the first 1024 bytes of each executable into the CSV feature columns h_0–h_1023. Contact the corresponding author for detailed information about additional pre-processing scripts used for this research.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A: Categorical Focal Loss Implementation

```
def categorical_focal_loss(gamma=2.0, alpha=0.25):
   def focal_loss_fixed(y_true, y_pred):
       # Normalise predictions
       y_pred /= tf.reduce_sum(y_pred, axis=-1, keepdims=True)
       # Clip to prevent extreme values and log(0) errors
       epsilon = tf.keras.backend.epsilon()
       y_pred = tf.clip_by_value(y_pred, epsilon, 1. - epsilon)
       # Calculate Cross Entropy: -y * log(y_hat)
       cross_entropy = -y_true * tf.math.log(y_pred)
       # Apply the Focal Loss formula: alpha * (1 - y_hat)^gamma *
```

```
        CE
        loss = alpha * tf.pow(1 - y_pred, gamma) * cross_entropy
        # Sum over the classes
        return tf.reduce_sum(loss, axis=1)
    return focal_loss_fixed
```

References

1. Avhankar MS, Pawar J, Kumbhar V. A comprehensive survey on polymorphic malware analysis: challenges, techniques, and future directions. Commun Appl Nonlinear Anal. 2025;32(9):2765–76.
2. Singh JR, Dhir S, Singh A. To study and analyze obfuscated, polymorphic, and zero-day malware behaviors: a systematic literature review. In: 2025 Modern Electronics Devices and Intelligent Communication Systems (MEDCOM). Piscataway, NJ, USA: IEEE; 2025. p. 670–4.
3. Nataraj L, Karthikeyan S, Jacob G, Manjunath BS. Malware images: visualization and automatic classification. In: Proceedings of the 8th International Symposium on Visualization for Cyber Security. New York, NY, USA: ACM; 2011. p. 1–7.
4. Steadman P, Jenkins P, Rathore RS, Hewage C. Low-cost malware detection with artificial intelligence on single board computers. Future Internet. 2026;18(1):46. doi:10.3390/fi18010046.
5. Ramamoorthy J, Shashidhar KN, Varol C. Packers and features: efficacy of static analysis for packed linux malware. In: 2025 13th International Symposium on Digital Forensics and Security (ISDFS). Piscataway, NJ, USA: IEEE; 2025. p. 1–6.
6. Gibert D, Zizzo G, Le Q. Towards a practical defense against adversarial attacks on deep learning-based malware detectors via randomized smoothing. In: Computer Security. ESORICS 2023 International Workshops: CPS4CIP, ADIoT, SecAssure, WASP, TAURIN, PriST-AI, and SECAI. Berlin/Heidelberg, Germany: Springer; 2024. p. 683–99.
7. Li Z, Tang J, Huang H, Yu L, Malekian R, Xiao F. DSMalConv: multi-modal malware detection based on dempster-shafer evidence uncertainty. IEEE Trans Depend Secur Comput. 2026;23:5589–603.
8. Hemalatha J, Roseline SA, Geetha S, Kadry S, Damaševičius R. An efficient DenseNet-based deep learning model for malware detection. Entropy. 2021;23(3):344. doi:10.3390/e23030344.
9. Dong S, Shu L, Nie S. Android malware detection method based on CNN and DNN bybrid mechanism. IEEE Trans Industrial Inform. 2024;20(5):7744–53. doi:10.1109/tii.2024.3363016.
10. S D, Shankar MG, Daniel E, R BGV. Enhancing security in IoMT using federated TinyGAN for lightweight and accurate malware detection. Sci Rep. 2026;16(1):7116. doi:10.1038/s41598-026-37830-2.
11. Chauhan M. MalwareVision-2025: image-based malware dataset. Kaggle; 2025 [cited 2026 May 21]. Available from: https://www.kaggle.com/datasets/mohitchauhan04/malwarevision.
12. He K, Kim DS. Malware detection with malware images using deep learning techniques. In: 2019 18th IEEE International Conference on Trust, Security and Privacy In Computing And Communications/13th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE). Piscataway, NJ, USA: IEEE; 2019. p. 95–102. doi:10.1109/TrustCom/BigDataSE.2019.00022.
13. Tan M, Le QV. EfficientNet: rethinking model scaling for convolutional neural networks. arXiv:1905.11946. 2020.
14. Zhang X. Phishing campaign delivering three fileless malware: AveMariaRAT/BitRAT/PandoraHVNC—Part II. 2022 [cited 2026 Jun 2]. Available from: https://www.fortinet.com/blog/threat-research/phishing-campaign-delivering-fileless-malware-part-two.
15. Zhang X. Infostealer malware FormBook spread via phishing campaign—Part II. 2025 [cited 2026 Jun 2]. Available from: https://www.fortinet.com/blog/threat-research/infostealer-malware-formbook-spread-via-phishing-campaign.
16. Steadman P, Jenkins P, Rathore RS, Hewage C. Challenges in implementing artificial intelligence on the Raspberry Pi 4, 5 and 5 with AI HAT. In: Naik N, Jenkins P, Prajapat S, Grace P, editors. The International Conference

on Computing, Communication, Cybersecurity and AI. Cham, Switzerland: Springer Nature; 2024. Vol. 884, p. 147–57.
17. Steadman P, Rathore RS, Hewage C, Jenkins P. AI optimisers and malware detection using imagary on the Rasberry Pi. In: 2025 7th International Conference on Information Systems and Computer Networks (ISCON). Piscataway, NJ, USA: IEEE; 2025. p. 1–8.
18. Agarap AF. Towards building an intelligent anti-malware system: a deep learning approach using support vector machine (SVM) for malware classification. arXiv:1801.00318. 2019.
19. Kalash M, Rochan M, Mohammed N, Bruce NDB, Wang Y, Iqbal F. Malware classification with deep convolutional neural networks. In: 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS). Piscataway, NJ, USA: IEEE; 2018. p. 1–5. doi:10.1109/NTMS.2018.8328749.
20. Alsubai S, Dutta AK, Alnajim AM, rahaman Wahab Sait A, Ayub R, AlShehri AM, et al. Artificial intelligence-driven malware detection framework for internet of things environment. PeerJ Comput Sci. 2023;9:e1366. doi:10.7717/peerj-cs.1366.
21. Patil S, Varadarajan V, Walimbe D, Gulechha S, Shenoy S, Raina A, et al. Improving the robustness of AI-Based malware detection using adversarial machine learning. Algorithms. 2021;14(10):297. doi:10.3390/a14100297.