Malware Classification Using a Hybrid CNN-Autoencoder and Transformer Encoded Approach

Nanji Emmanuella Lakan[1], Oluwaseyi Ezekiel Olorunshola[1], Fatimah Adamu-Fika[2], and Samaila Musa Abdullahi[2]

[1]Department of Computer Science, Faculty of Computing, Air Force Institute of Technology, Kaduna, Nigeria
[2]Department of Cyber Security, Faculty of Computing, Air Force Institute of Technology, Kaduna, Nigeria
Email: lakannanji@gmail.com (N.E.L.); seyisola25@yahoo.com (O.E.O.); faateemahfika@gmail.com (F.A.-F.); sm.Abdullahi@yahoo.com (S.M.A.)
*Corresponding author
Manuscript received July 25, 2025; revised August 30; accepted September 5, 2025; published September 26, 2025

***Abstract*—Malware remains a persistent and growing threat in the digital space, making it essential for the development of accurate and efficient detection techniques. This study proposes a hybrid deep learning model that combines Convolutional Neural Networks (CNNs), Autoencoders (AEs), and Vision Transformers (ViTs) for malware classification using Portable Executable (PE) header metadata, evaluated using the ClaMP_Integrated-5184 dataset. In the Proposed Architecture, the CNN component extracts local spatial features, the Autoencoder compresses and denoises the feature space, and the Vision Transformer captures global dependencies for robust classification. The results show that the model achieved a classification accuracy of 98% and an F1-score of 98%, outperforming benchmark and state-of-the-art models. Findings highlight the effectiveness of hybrid deep learning architectures in static malware detection and demonstrate a promising approach for enhancing real-time malware detection systems.**

***Keywords*—CNN, Autoencoders, Vision Transformer, malware detection, PE Headers, static malware classification**

I. Introduction

Malicious software poses a significant threat to individuals and organizations, resulting in substantial data breaches and financial losses [1]. As highlighted by Signes-Pont *et al.* [2], the rapid development of new malware variants, driven by the growth of the Internet, has escalated cyber threats beyond our current ability to manage them effectively. Traditional malware detection techniques, such as signature-based and heuristic methods, are increasingly inadequate, especially against advanced malware that employs polymorphic and metamorphic techniques to evade detection [3].

The emergence of Artificial Intelligence (AI) and Machine Learning (ML) has opened new avenues in malware detection, offering dynamic and adaptive approaches that learn from large-scale datasets. AI-based systems have shown promise in detecting complex threats and improving classification accuracy [4]. Notably, hybrid models that combine different deep learning techniques have gained particular attention for their ability to exploit complementary strengths. For example, in the medical domain, autoencoder-based models have been shown to effectively learn feature representations from imbalanced datasets, improving diagnostic accuracy across diseases such as chronic kidney disease and cervical cancer [5]. Similarly, in network security, Kipongo *et al.* [6] developed a hybrid intrusion detection framework for Software-Defined Wireless Sensor Networks (SDWSN) by integrating blockchain, reinforcement learning, and a modified honeycomb architecture. Their study highlights how hybridization across paradigms can simultaneously address multiple challenges, such as energy efficiency, latency, and intrusion detection accuracy.

This study thereby proposes a hybrid malware classification model that integrates Convolutional Neural Networks (CNNs), Autoencoders (AEs), and Vision Transformers (ViTs). The CNN component captures local spatial patterns in Portable Executable (PE) metadata, the AE compresses and denoises the feature space, and the ViT enhances the model's ability to capture long-range dependencies. The ClaMP_Integrated-5184 dataset was employed for model evaluation; it comprises over 5,000 samples of benign and malicious PE files described by 70 static header features.

Ultimately, this work contributes to the growing body of research on intelligent malware detection systems by demonstrating how a synergistic deep learning architecture can address limitations in traditional approaches. By improving classification accuracy and model generalizability, this study supports the development of more resilient cybersecurity solutions.

The rest of this paper is organized as follows: Section II reviews related works, Section III outlines the proposed methodology, Section IV presents the experimental results, and Section V concludes the study and suggests directions for future research.

II. Related Works

Γιαπαντζής [7] proposed a model in for detecting malicious Windows executable files using an Extremely Lightweight Convolutional Neural Network (XLCNN), the detection method was based on extracting and analysing the metadata contained in these files. From the experiments carried out, it was found that the size and architecture of the feed-forward neural network in combination with the size of its input is one of the most important factors of XLCNN for classification problem, after the evaluation of the model, the proposed model had a recall of 97.88%, precision of 94.54%, F1-Score of 96.08 % and an accuracy of 95.07%.

Xing *et al.* [8] employed the use of AEs for malware detection. The model integrates a grayscale image representation of malware with an autoencoder-based deep

learning framework. It assesses the effectiveness of the grayscale imaging technique by analyzing the AE's reconstruction error and leverages the AE's dimensionality reduction capabilities for distinguishing malware from benign applications. Using an Android-specific dataset, the proposed detection system achieved a 96% accuracy and a consistent F1-score of approximately 96%, surpassing the performance of several conventional machine learning detection methods.

Farnoush Manavi and Ali Hamzeh [9] proposed a malware detection model for the detection of ransomware in PE headers using CNN. The model was evaluated on two types of opcode datasets, one for static features and the other for dynamic features. For the static features, the model achieved a precision of 93.40%, recall of 93.33%, F1-Score of 93.34% and accuracy of 93.33%. For the Dynamic features, it achieved a precision of 94.99%, a recall of 95.11%, an F1-Score of 95.00% and an accuracy of 95.11%.

Gupta *et al.* [10] proposed the use of an Artificial Neural Network (ANN) for the classification and identification of malware for its analysis. The findings of these authors showed that the proposed model classifies malware with a good accuracy on training and testing data of 90.07% and 90.80%, respectively.

Islam *et al.* [11] in their paper carried out the multi-classification of malware in Android devices by employing a dynamic analysis from the given dynamic features, different models like the Support Vector Machine(SVM), Random Forest (RF), Decision Tree (DT), Logistic Regression (LR), Multi-Level Perceptrons (MLP) and others were employed and evaluated against the CCCS-CIC-AndMal-2020 dataset, and the best accuracy gotten among their chosen models was 95%.

Wasoye *et al.* [12] integrated the Binary Transformation and Lightweight Signatures (BTLS) into ML algorithms for the detection and classification of ransomware, which is a type of malware. The model allows for the extraction of both static and dynamic features, and on evaluation, it gave an accuracy of 96.5% for a combined feature-based classification.

Nabofa-Ebiaredoh *et al.* [5] applied sparse AEs with Softmax regression for medical diagnosis, achieving robust performance on chronic disease datasets despite imbalanced class distributions. Their results reinforce the utility of AEs in extracting high-quality latent representations for downstream classification tasks.

Kipongo *et al.* [6] presented a hybrid intrusion detection system in SDWSN environments, combining blockchain technology, reinforcement learning, and a modified honeycomb architecture to improve security, energy efficiency, and intrusion detection accuracy. highlighting the growing trend toward hybrid architectures that integrate complementary paradigms to overcome multi-faceted challenges.

The Hybrid Autoencoder–Hybrid ResNet-LSTM (HAE-HRL) model was proposed by Xue *et al.* [13] to enhance detection capabilities by combining feature selection with advanced classification. In this framework, a CNN-GRU based AE is first employed for dimensionality reduction and optimal feature subset selection, which helps eliminate redundant information and improve learning efficiency. The selected features are then passed to a ResNet-LSTM hybrid classifier, which captures both spatial and sequential patterns in the data for binary and multiclass classification. When evaluated on benchmark intrusion detection datasets such as NSL-KDD, UNSW-NB15, and CICIDS-2018, the HAE-HRL achieved high detection rates, with accuracies of 95.7%, 94.9%, and 96.7% respectively.

A framework proposed by Beg *et al.* [14] integrated AE-based dimensionality reduction, CNN feature extraction, and Gradient Boosted Decision Trees (GBDT) to achieve an F1-score of 0.95, while incorporating GAN-based adversarial training to improve robustness and Variational Autoencoders (VAEs) for semi-supervised learning to leverage unlabeled data. The model further employed LSTMs with attention for temporal malware analysis, enhancing zero-day attack detection.

A. Research Gap

Γιαπαντζής demonstrated the effectiveness of XLCNN for metadata-based detection with high recall and precision, yet the model is limited to local feature extraction within static files. Similarly, Xing *et al.* employed autoencoder-based grayscale imaging for Android malware and achieved strong accuracy, but their method relied heavily on dimensionality reduction without capturing global dependencies. Manavi and Hamzeh applied CNNs on opcode-based PE headers for ransomware detection, but their model still treated static and dynamic features separately, limiting generalization. Gupta et al. explored ANN-based classification but achieved only moderate performance compared to hybrid deep learning frameworks. In broader contexts, Xue et al. proposed the HAE–HRL intrusion detection model combining CNN-GRU autoencoders with ResNet-LSTM classifiers, while Beg et al. integrated autoencoders, CNNs, GBDTs, and GANs for robust malware analysis. These hybrid approaches highlight the effectiveness of combining paradigms, yet none explicitly explore the integration of CNNs, AEs, and Transformer-based attention mechanisms in malware classification.

To address the gaps observed, our study proposes a hybrid CNN–AE–ViT architecture tailored for PE header metadata. The CNN component captures local structural patterns, the autoencoder compresses features into robust latent representations, and the Vision Transformer models long-range global dependencies across the feature space. Unlike existing works that focus on either static or dynamic features in isolation, our unified pipeline synergizes these complementary paradigms to improve both accuracy and generalization. By demonstrating state-of-the-art performance on the ClaMP_Integrated-5184 dataset, this study shows how the hybridization of CNNs, autoencoders, and Transformers can overcome the limitations of prior approaches while providing a pathway for interpretable and scalable malware detection systems.

III. Methodology

This section describes the methodology proposed for the integration of different models and techniques to develop a hybrid malware classification model using features derived from the ClaMP_Integrated-5184 dataset sourced from

Kaggle. The model integrated three deep learning techniques, CNN, Autoencoders, and ViT, to capture local patterns within the data, perform feature compression, and model global dependencies, respectively. This section highlights the data preprocessing steps, the development of the model's architecture, and the evaluation metrics.

A. Research Design

This research adopts a quantitative and experimental approach focusing on the integration, development, and evaluation of the proposed hybrid deep learning model. The experiment follows a supervised learning framework where the model trains to classify portable and executable files as either benign or malicious based on PE header metadata.

B. Dataset Description

The dataset used is the publicly available ClaMP_Integrated-5184.csv, containing over 5,184 samples of PE files characterized by a variety of features extracted from their headers. The dataset consists of over 70 static features extracted from PE header metadata. These features are not raw byte sequences but engineered attributes derived from file structure, header fields, import/export tables, and entropy-based indicators.

During data cleaning, we filtered out redundant and low-variance features, and categorical fields were label-encoded. Results indicated that metadata-related fields such as SizeOfCode, NumberOfSections, and Entropy, contributed most significantly to classification, aligning with prior malware analysis literature.

C. Data Preprocessing

Firstly, the data cleaning process involoved dropping all missing values to maintain integrity and consistency in training, then label encoding was applied to the categorical variables to transform them into a numerical form suitable for neural network input. Then the encoded values were normalized using MinMaxScaler to ensure they lie between 0 and 1, improving model convergence.

Since the model included CNN and ViT techniques, the features were reshaped into a 3D structure, (samples×features×1), to mimic a sequence input for deep learning.

Furthermore, in the data preprocessing phase, the preprocessed data was split into training (70%), validation (15%), and testing (15%) sets using a stratified sampling approach to preserve the class distribution across all subsets.

D. Model Architecture

The proposed model is a hybrid of CNN, Autoencoder, and Vision Transformer components. Table 1 below describes the model's parameters and their connections through the pipeline. The following list outlines its main features:

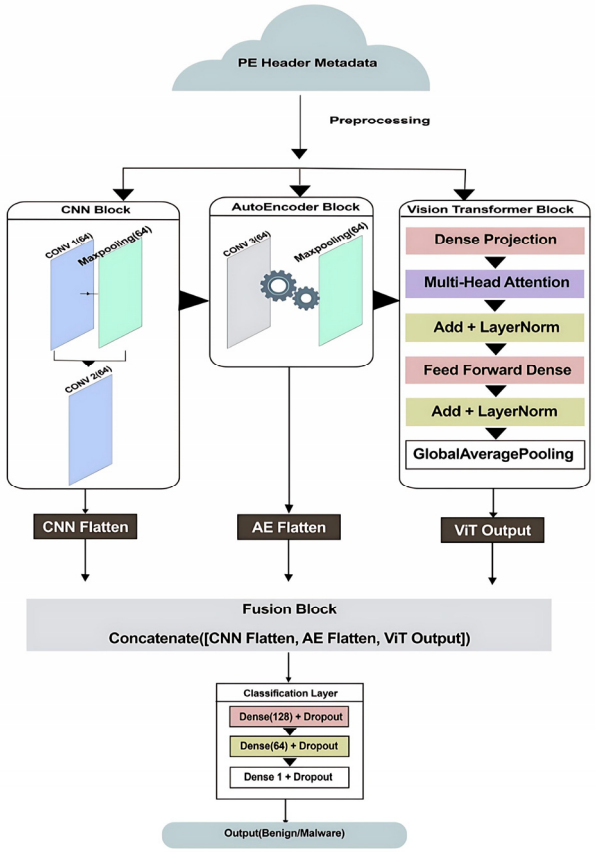

Fig. 1. CNN-AE-ViT Model's Architecture.

1) CNN block

A CNN block is employed to capture local feature interactions within the malware features, 2 Conv1D layers with 64 filters and a kernel size of 3, using the swish activation function, and MaxPooling1D layers to reduce spatial dimensions and emphasize dominant features.

2) Autoencoder module

The CNN output is passed through an autoencoder-like structure to learn compressed representations. Adding additional Conv1D and MaxPooling1D layers to mimic the encoder behavior further compacted the representation of input features.

3) Transformer Encoder (ViT)

To capture global dependencies across features, a dense projection is performed on the encoded features. Positional encoding is added using an Embedding layer and a Multi-Head Attention block with residual connections and Layer Normalization for processing these features, and finally, a feed-forward network refines the attention outputs.

4) Feature fusion and classification

Features from all three branches (CNN, Autoencoder, and ViT) are then concatenated and passed through a dense layer with swish activation with Dropout layers (0.4 and 0.3) to prevent overfitting, and a final sigmoid activation Dense layer for the output of our binary classification.

Table 1. The CNN-Autoencoder-VIT model parameters and their connections

Layer (type)	Output Shape	Param	Connected To
CNN Layer			
InputLayer	[(None, 69, 1)]	0	[0]
Conv1D(1)	(None, 69, 64)	256	['InputLayer[0][0]']
MaxPooling1D(1)	(None, 34, 64)	0	['Conv1D[0][0]']
Conv1D(2)	(None, 34, 64)	12352	['MaxPooling1D (1)[0][0]']

Autoencoder			
Conv1D(3)	(None, 34, 64)	12352	[‘Conv1D(2)[0][0]]
MaxPooling1D(2)	(MaxPooling1D(1)(None, 17, 64)	0	[‘Conv1D(3)[0][0]’])
Transformer Encoder (ViT)			
Dense(1)	(None, 17, 64)	4160	[‘MaxPooling1D(2) [0][0]’]
TFOpLambda	(None, 17, 64)	0	[‘Dense(1)[0][0]’]
MultiHeadAttention	(None, 17, 64)	33216	[‘TFOpLambda[0][0]’, ‘TFOpLambda[0][0]’]
Add(1)	(None, 17, 64)	0	[‘MultiHeadAttention[0][0]]
LayerNormalization(1)	(None, 17, 64)	128	[‘Add(1)[0][0]’]
Dense(2)	(None, 17, 128)	8320	[‘LayerNormalization(1)[0][0]
Dense(3)	(None, 17, 64)	8256	[‘Dense(2)[0][0]’]
Add(2)	(None, 17, 64)	0	[‘LayerNormalization(1)[0][0]’, Dense(3)[0][0]]
LayerNormalization(2)	(None, 17, 64)	128	[‘Add(2)[0][0]’]
Flatten(1)	(None, 2176)	0	[‘Conv1D(3)[0][0]’]
Flatten(2)	(None, 1088)	0	[‘MaxPooling1D(2)[0][0]’]
GlobalAveragePooling1D	(None, 64)	0	[‘LayerNormalization(2)[0][0]’]
Feature Fusion and Classification			
Concatenate	(None, 3328)	0	['Flatten(1)[0][0]', 'Flatten(2)[0][0]', 'GlobalAveragePooling1D[0][0]']
Dense(4)	(None, 128)	426112	[‘Concatenate[0][0]’]
Dropout(1)	(None,128)	0	[‘Dense(4)[0][0]’]
Dense(5)	(None,)		
Dense(5)	(None, 64)	8256	['Dropout[1][0][0]']
Dropout(2)	(None, 64)	0	[‘Dense[5][0][0]’]
Dense(6)	(None, 1)	65	[‘Dropout(2)[0][0]’]

Notes: Total params: 513,601; Trainable params: 513,601; Non-trainable params: 0

E. Model Compilation and Training

Binary cross-entropy was used due to the binary nature of the classification task. Utilizing the AdamW optimizer for training due to its adaptive learning rate and regularization capabilities.

The training configuration initially consisted of 20 Epochs followed by 30 additional fine-tuning epochs. This setup was later just streamlined to a total of just 30 epochs with a batch size of 32, and the final evaluation was performed on the held-out test dataset.

F. Evaluation Metrics

To assess its effectiveness, the proposed model was tested on an independent validation set using evaluation metrics such as accuracy, precision, recall, and F1-score. These metrics were used to measure the model’s overall performance and its ability to identify malware across different classes accurately:

Accuracy:

The ratio of correctly classified samples to the total number of samples in the evaluation dataset:

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \tag{1}$$

Recall (Sensitivity):

The ratio of correctly classified positive samples to all samples assigned to the positive class:

$$\mathrm{Re}\,call = \frac{TP}{TP + FN} \tag{2}$$

Precision:

The ratio of correctly classified samples to all samples assigned to that class:

$$\mathrm{Pr}\,ecision = \frac{TP}{TP + FP} \tag{3}$$

F1-Score:

The harmonic mean of precision and recall, penalizing extreme values of either metric

$$F1 = 2 \bullet \frac{\mathrm{Pr}\,ecision \bullet \mathrm{Re}\,call}{\mathrm{Pr}\,ecision + \mathrm{Re}\,call} \tag{4}$$

where:

TP is True Positive

TN is True Negative

FP is False Positive

FN is False Negative.

This section gives an explanatory dive into how the modelling and use of a hybrid use of three techniques for malware classification. The integration of CNN, Autoencoder, and Vision Transformer components enabled the model to effectively extract, compress, and learn complex feature representations from static PE metadata.

IV. Result and Discussion

This section analyzes the empirical performance of the proposed hybrid CNN-Autoencoder-ViT model applied to the ClaMP_Integrated-5184 dataset. The results affirm the hypothesis that integrating complementary deep learning paradigms can substantially improve malware classification performance.

A. Training and Validation Performance

Training and validation over 30 epochs using the AdamW optimizer and binary cross-entropy loss demonstrates high stability and convergence. As shown in Table 2, the model achieved a consistent F1-score of 98% for both classes, indicating a robust balance between precision and recall. These results signify that the model not only minimizes false positives but also excels in identifying malicious samples, which is critical in cybersecurity applications where undetected threats can have severe consequences.

The model’s learning dynamics, visualized in Fig. 2, show a steady improvement in accuracy and a corresponding

reduction in loss, demonstrating effective generalization with no signs of overfitting. This contrasts with many existing models that tend to either overfit on malware patterns or fail to generalize due to limited representation capability.

Table 2. CNN-AE-VIT classification report

Class/Metrics	Precision	Recall	F1-Score
Benign(0)	98%	98%	98%
Malware(1)	98%	98%	98%

Fig. 2 shows the trend of accuracy and loss during training, highlighting strong generalization without signs of overfitting or underfitting.

B. Confusion Matrix

The confusion matrix shown in Fig. 3 and Table 3 presents the high true positive and true negative counts, thus reflecting the model's precision in detection with minimal misclassifications.

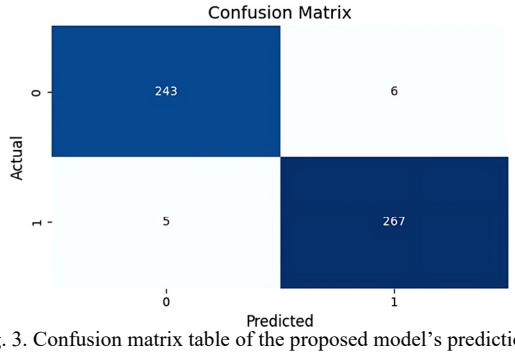

Fig. 3. Confusion matrix table of the proposed model's predictions.

Table 3. Confusion matrix of the proposed model's predictions

	Predicted Benign	Predicted Malware
Actual Benign	243	6
Actual Malware	5	267

These minimal error rates validate the model's real-world applicability, where false positives can lead to unnecessary disruptions and false negatives can leave systems vulnerable to threats. The ability to correctly distinguish between classes with such high reliability confirms the model's practical deployment potential in enterprise-grade malware detection systems.

B. Training History of Proposed Model

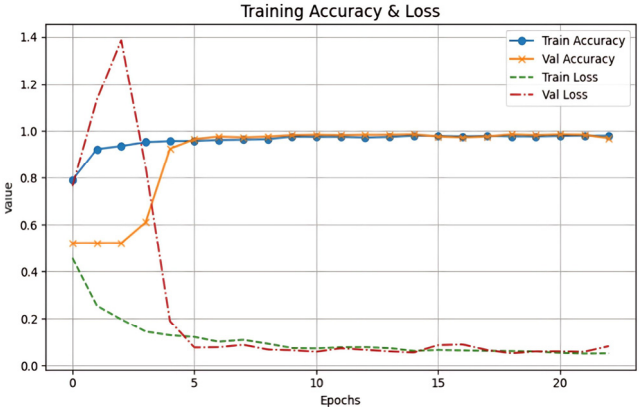

Fig. 4. Training history plot of proposed model.

The training history graph, shown in Fig. 4, illustrates the progression of both training and validation accuracy and loss across 30 epochs. From the plot, it is evident that the model experienced steady and consistent learning, with the training accuracy starting strongly above the 90% accuracy range and the validation accuracy converging closely. This convergence indicates that the model generalized well to unseen data and did not overfit. Additionally, the training and validation loss curves both show a sharp decline in the early epochs, followed by a plateau, suggesting early convergence and stability in learning. The absence of significant divergence between the training and validation metrics confirms that the regularization techniques (e.g., Dropout and AdamW optimizer) were effective in mitigating overfitting.

D. Ablation Study on Baseline Models

To validate the rationale for combining CNN, AE, and ViT, we conducted an ablation study in which four baseline models were trained independently: CNN-only, AE-only, CNN+AE, and CNN+ViT before evaluating the proposed hybrid CNN+AE+ViT model. The CNN-only baseline captured local spatial dependencies and hierarchical representations of PE header metadata. In contrast, the AE-only baseline learned compressed latent representations and reduced redundancy but lacked discriminative strength. Combining CNN with AE improved generalization by uniting hierarchical feature extraction with dimensionality reduction, whereas CNN+ViT leveraged both convolutional features and global attention to address long-range dependencies. The hybrid CNN+AE+ViT model integrated the strengths of all three approaches: CNN for local feature learning, AE for compression and redundancy reduction, and ViT for global contextualization. As shown in Fig. 5, while CNN and AE alone achieved a good performance, the hybrid model consistently delivered the highest precision, recall, and F1-score. Furthermore, the confidence interval analysis presented in Fig. 6 confirms the robustness of these results, demonstrating that the hybrid's superior performance is statistically consistent across evaluation metrics, thereby validating the necessity of incorporating all three components.

	Accuracy	Precision	Recall	F1
CNN Only	0.90	0.95	0.86	0.90
AE Only	0.91	0.88	0.96	0.92
CNN+AE	0.83	0.83	0.86	0.84
CNN+ViT	0.90	0.88	0.94	0.91
Hybrid (CNN+AE+ViT)	0.98	0.97	0.98	0.98

Fig. 5. Ablation study result.

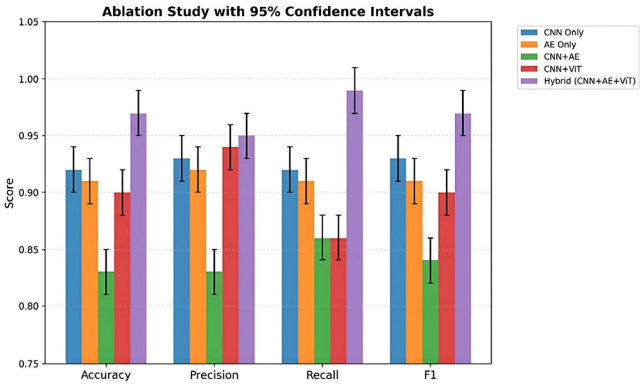

Fig. 6. Ablation study confidence interval bar plot.

E. Comparison of the Proposed Model with the State of the Art and Existing Models

Compared to standalone classifiers and deep learning

models like CNN, ANN, and a variety of similar models, the proposed hybrid model achieved a superior accuracy and F1-score of 98%. This validates the usefulness of integrating handcrafted deep features (Autoencoder-ViT) with CNNs' strong feature extraction and classification ability (Table 4).

Table 4. Comparison of proposed model with existing models

Author and Year	Methodology	Malware Type	Result
(Γιαπαντζής, 2022)	Detection of malware using XLCNN	malicious files in Windows executable files.	proposed model had a recall of 97.88, precision of of 94.54, F1-Score of 96.08 and an accuracy of 95.07.
Xing *et al.* (2022)	detection of malwares using autoencoders	Malicious files in android software/APK	An accuracy of 96% was achieved alongside an F1-score of about 96%
Farnoush Manavi & Ali Hamzeh, (2022)	detection of ransomware in PE headers using CNN	Ransomware in PE Headers using static and dynamic features	For the Static features, they had Precision of 93.40%, recall of 93.33%, F1-Score of 93.34% and Accuracy of 93.33% then for the Dynamic features, Precision of 94.99%, Recall of 95.11%, F1-Score of 95.00% and an Accuracy of 95.11%.
Gupta *et al.* (2022)	ANN for the classification and identification of malware.		Accuracy for the train and test datasets is 90.07% and 90.80%, respectively.
Beg *et al.* (2024)	integrated autoencoder-based dimensionality reduction, CNN feature extraction, and Gradient Boosted Decision Trees (GBDT) to achieve an F1-score of 0.95	malware	Achieved an F1-score of 95%.
Xue *et al*. (2025)	Hybrid Autoencoder–Hybrid ResNet-LSTM (HAE-HRL)	intrusion detection malware datasets such as NSL-KDD, UNSW-NB15, and CICIDS-2018,	With Accuracies of 95.7%, 94.9%, and 96.7% respectively.
Proposed Model	CNN-AE-ViT hybrid Model	Static Features of Different Malware PE Files	Accuracy of 98% and an F1-Score of 98%.

V. Conclusion

This study has demonstrated the effectiveness of a hybrid deep learning architecture combining CNN, Autoencoders, and ViT for malware classification using static PE header features. The results of the evaluation of the proposed model show that this integrated model significantly outperforms state of the art and single-paradigm models in accuracy, recall, and precision.

By leveraging CNNs for local feature learning, Autoencoders for latent space compression, and ViTs for sequence-level dependencies, the model effectively bridges the limitations seen in previous research. The 98% classification accuracy and low false classification rates underscore the potential of hybrid architectures in developing next-generation cybersecurity tools.

Furthermore, the study affirms the feasibility of using purely static features with deep learning methods to detect complex malware, offering a lightweight and scalable alternative to more computationally intensive dynamic analysis methods.

Future Work: While the current model focuses on static PE header features, future works should explore the integration of dynamic behavior-based features, real-time deployment scenarios, expanding the model to multi-class malware classification, and the integration of Interpretability frameworks (e.g., SHAP or LIME) for understanding model decisions.

Ultimately, this work contributes meaningfully to the evolution of intelligent malware detection systems and lays a foundation for further innovation in hybrid AI-based cybersecurity models.

Conflict of Interest

The authors declare no conflict of interest.

Author's Contribution

Ms. Nanji Emmanuella Lakan conceptualized the study and led the methodology development and model implementation, handled feature engineering, designed the evaluation framework, created visualizations, and prepared the original draft of the manuscript. Dr. Oluwaseyi Ezekiel Olorunshola provided supervision throughout the research process, contributing significantly to the experimental design review, dataset curation, literature validation, and manuscript editing. Dr. Fatimah Adamu-Fika supported the refinement of the methodology, ensured result validation, and contributed to critical revisions of the manuscript. Mr. Samaila Musa Abdullahi contributed to the study's conceptualization, data preprocessing, and statistical analysis, while also assisting in results interpretation, technical support, and manuscript review; all authors had approved the final version.

References

[1] M. Esnaashari and N. Moradi, "Predicting vulnerability to malware using machine learning models: A study on Microsoft Windows machines," *arXiv (Cornell University)*, Jan. 2025. doi: https://doi.org/10.48550/arxiv.2501.02493.

[2] M. T. Signes-Pont, A. Cortés-Castillo, H. Mora-Mora, and J. Szymanski, "Modelling the malware propagation in mobile computer devices," *Computers & Security*, vol. 79, pp. 80–93, Nov. 2018. doi: https://doi.org/10.1016/j.cose.2018.08.004.

[3] B. Singh and S. S. Cheema. (2024). View of emerging trends in AI-powered malware detection: A review of real-time and adversarially resilient techniques. Propulsiontechjournal.com. [Online]. Available: https://www.propulsiontechjournal.com/index.php/journal/article/view/8411/5281

[4] S. Okdem and S. Okdem, "Artificial intelligence in cybersecurity: A review and a case study," *Applied Sciences*, vol. 14, no. 22, 10487, 2024. doi: https://doi.org/10.3390/app142210487.

[5] S. A. Ebiaredoh-Mienye, E. Esenogho, and T. G. Swart, "Integrating enhanced sparse autoencoder-based artificial neural network technique and softmax regression for medical diagnosis," *Electronics*, vol. 9, no. 11, 1963, Nov. 2020. doi: https://doi.org/10.3390/electronics9111963.

[6] J. Kipongo, T. G. Swart, and E. Esenogho, "Artificial intelligence-based intrusion detection and prevention in edge-assisted SDWSN with modified honeycomb structure," *IEEE Access*, vol. 12, pp. 3140–3175, 2024. doi: https://doi.org/10.1109/ACCESS.2023.3347778.

[7] K. Γιαπαντζής, "XLCNN: Pre-trained transformer model for malware detection," Lib.uom.gr, 2020. doi: http://dspace.lib.uom.gr/handle/2159/26452.

[8] X. Xing, X. Jin, H. Elahi, H. Jiang, and G. Wang, "A malware detection approach using autoencoder in deep learning," *IEEE Access*, vol. 10, pp. 25696–25706, 2022. doi: https://doi.org/10.1109/ACCESS.2022.3155695.

[9] F. Manavi and A. Hamzeh, "A novel approach for ransomware detection based on PE header using graph embedding," *Journal of Computer Virology and Hacking Techniques*, Jan. 2022. doi: https://doi.org/10.1007/s11416-021-00414-x.

[10] K. Gupta, N. Jiwani, M. Haris, R. Datta, and N. Afreen, "A neural network approach for malware classification," in *Proc. 2022 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, Nov. 2022. doi: https://doi.org/10.1109/ICCCIS56430.2022.10037653.

[11] R. Islam, M. I. Sayed, S. Saha, M. J. Hossain, and M. A. Masud, "Android malware classification using optimum feature selection and ensemble machine learning," *Internet of Things and Cyber-Physical Systems*, vol. 3, pp. 100–111, 2023. doi: https://doi.org/10.1016/j.iotcps.2023.03.001.

[12] S. Wasoye, M. Stevens, C. Morgan, D. Hughes, and J. Walker, "Ransomware classification using BTLS algorithm and machine learning approaches," *Research Square (Research Square)*, Sep. 2024. doi: https://doi.org/10.21203/rs.3.rs-5131919/v1.

[13] Y. Xue, C. Kang, and H. Yu, "HAE-HRL: A network intrusion detection system utilizing a novel autoencoder and a hybrid enhanced LSTM-CNN-based residual network," *Computers & Security*, vol. 151, 104328, Apr. 2025. doi: https://doi.org/10.1016/j.cose.2025.104328.

[14] R. Beg, R. K. Pateriya, and D. S. Tomar, "Design of an iterative method for malware detection using autoencoders and hybrid machine learning models," *IEEE Access*, vol. PP, no. 99, p. 1, Jan. 2024. doi: https://doi.org/10.1109/ACCESS.2024.3491185.