Journal of Techniques, ISSN: 2708-8383, Vol. 8, No. 1, 2026, Pages 62-70
https://doi.org/10.51173/jt.v8i1.2748

JOURNAL OF TECHNIQUES

Journal homepage: http://journal.mtu.edu.iq

RESEARCH ARTICLE - ENGINEERING (MISCELLANEOUS)

A Deep Learning–Based Hybrid Neural Network Model for Malware Detection

Muthana S. Mahdi[1*]

[1]Department of Computer Science, College of Science, Mustansiriyah University, Baghdad, Iraq

[*] Corresponding author E-mail: muthanasalih@uomustansiriyah.edu.iq

Article Info.

Article history:

Received
12 September 2025

Revised
02 February 2026

Accepted
25 February 2026

Published
31 March 2026

Abstract

Malware remains a significant threat to modern computing systems and networks worldwide. Evolving malware utilizes polymorphism, metamorphism, and zero-day exploits to bypass defenses. Traditional signature-based and heuristic detection methods are now struggling with the increasing complexity of malware. In this paper, a hybrid neural network model is proposed that combines a convolutional neural network (CNN) to detect spatial malware patterns, recurrent neural networks (RNNs) to analyses temporal behaviors, and attention mechanisms to select crucial features for accurate and reliable threat classification. Multi-scale convolutional layers and residual connections improve dataset generalization and reduce overfitting. Focal loss functionality addresses the class imbalance in real-world malware detection scenarios. Experimental results on EMBER, EMBER Sim, and SoReL-20M datasets show exceptional accuracy and precision. This interpretable, scalable deep learning (DL) model bridges traditional methods with modern cybersecurity challenges. The model excels in zero-day detection and produces a few false positives, achieving 96.7% accuracy, 96.1% precision, 96.8% recall, and 96.4% F1-score. Additionally, the findings demonstrate clear improvements over previous methods, achieving a 1.1–2.6% increase in accuracy, confirming the model's superior detection capability. This advanced deep-learning approach sets a new benchmark in cybersecurity.

Publisher: Middle Technical University

Keywords: Malware Detection; Cybersecurity; Malware Classification; Residual Connections; Attention Mechanism; Recurrent Neural Networks (RNNs).

1. Introduction

Modern computing systems and networks face their most perilous threat from malware, which shortens "malicious software" into its current form. The suite of destructive software programs includes viruses, ransomware, worms, trojans, and spyware [1]. These types of software have specific functions to either disrupt system operations, steal sensitive information, or alter system behavior to grant unauthorized access to cyber attackers [2]. The development of digital infrastructure has allowed malware to become more sophisticated through advanced techniques that include polymorphism and metamorphism, as well as fileless execution to avoid detection [3, 4]. The growing sophistication of malware threatens digital resources because it creates an essential vulnerability for cybersecurity systems that compromises confidentiality and availability while attacking system integrity [5].

The foundations of cybersecurity protection are signature-based detection and heuristic, although it has been a long time since they were first discovered as a method of standard use [6, 7]. Detection methods based on signatures rely on pre-known patterns and fingerprints of known malware to identify security threats [8]. Such detection techniques exhibited limited attributes in their ability to identify any unknown malware since no signature patterns are available to identify the zero-day attacks and other novel malware [9]. The heuristic-based detection applies behavioral rules and statistical models that are used to identify unusual patterns that are signs of potential malicious behavior [10,11]. False detection alerts and limited scalability performance in high-throughput systems with a large number of unexpected threats are caused by the detection of these strategies [12].

This rate of malware development establishes an urgent need to have detection systems that are intelligent and can develop and detect threats in real time [13]. DL has made considerable advancements in the field of artificial intelligence (AI) technology, offering good prospects of detection [14, 15]. The great ability of DL models to identify complex data patterns allows them to carry out malware detection tasks effectively [16]. Such systems rely on neural networks to detect malicious behavior based on static metadata and dynamic runtime behavior, and Application Programming Interface (API) calls. The DL models have higher accuracy in recognizing an unknown malware since they get trained to learn generalization abilities beyond what they are trained to do [17- 18].

In 2020, Kabanda [19] conducted research on utilizing machine learning (ML) and DL methods in cybersecurity to deal with fast-evolving cyber threats. The proposed techniques encountered problems when trying to detect new malware strains. In the same year, Latif et al. [20] presented a lightweight random neural network that detected industrial Internet of Things (IoT) system attacks with high precision levels. However, this method's successful implementation failed to scale to larger applications beyond industrial environments.

Nomenclature & Symbols			
API	Application Programming Interface	AI	Artificial Intelligence
CNN	Convolutional Neural Network	SPC	Specificity
DL	Deep Learning	TP	True Positive
RNN	Recurrent Neural Network	TN	True Negative
ReLU	Rectified Linear Unit	DE	Diagnostic Efficiency
LSTM	Long Short-Term Memory	FP	False Positive
ML	Machine Learning	FN	False Negative
IoT	Internet of Things	NPV	Negative Predictive Value
EMBER	Endgame Malware Benchmark for Research	LeakyReLU	Leaky Rectified Linear Unit
EMBER-Sim	Simulated Version of EMBER Dataset	SoReL-20M	Sophos–Released Large-Scale Malware Dataset

Sarker [21] (2021) carried out a study with CNNs to detect intrusions and identify malware and obtained positive results in feature extraction. The study placed limitations on addressing time-sensitive security threats. Also, Yamcharoen et al. [22] (2021) proposed the use of an artificial neural network model that is reactive to the detection of malware based on user behavior. The predictive capability of this method is still high since it is an effective approach though it cannot identify zero-day attacks. In addition, Alharbi et al. [23] described the speed at which intrusion detection systems using ML process cybersecurity threats. The system worked well, but there were challenges in identifying any type of IoT information because of its various natures.

Abdullahi et al. [24] (2022) looked at the role of AI methods in securing IoT devices using advanced systems to handle large amounts of data. The study showed a high level of performance of the DL but revealed the challenges of scalability and issues with the design of the system. Algorain [25] investigated ML of data analytical procedures in cybersecurity due to his discovery that it would maximize the capabilities of detection systems. The researchers applied optimization of Bayesian hyperparameters to improve malware detection methods, achieving enhanced accuracy.

In 2023, Komarudin et al. [26] discussed the possibility of AI to detect malware in various network setups. The detection system showed good performance but was not effective in managing the current changes in the cyber threat environments. Furthermore, Rizvi [27] tested the performance of an AI system to identify advanced threats through the success of the analysis of great volumes of data. The efficiency of the current systems revealed the necessity to have better data potential since the sophistication of cyber threats kept increasing.

Ojha [28] (2024) examined the use of ANNs, including CNNs and RNNs in controlling cyber threat detection and prevention. Even though the suggested methods are doing well, they contain high rates of false positives. Moreover, Shahana [29] (2024) noted the ability of DL models to detect serious malware on their own, which is necessary in contemporary cybersecurity. Last, Syeda and Asghar [30] (2024) introduced an advanced dynamic malware classification using categorizing API behaviors in Windows files. The focus of the research was on improving the rate of implementation and identification of a hostile type of sample. The technique achieved detection speed and sensitivity, but problems were faced because of its computational complexity. Although AI proved to be very advanced, the inquiry cast ethical doubts on the dangers of false positives and misappropriation of resources in automated systems.

In contrast to previous works that focused on either spatial or temporal aspects of malware detection, the current study integrates both through a hybrid CNN–RNN structure enhanced with attention and also residual connections. Unlike conventional DL methods, the models that rely solely on static features or handcrafted attributes, the approach jointly analyzes spatial and sequential behaviors, improving adaptability to unseen malware. Furthermore, the integration of focal loss addresses the class imbalance problem, which is not adequately considered in prior research. This combination distinguishes the proposed framework as a more interpretable, scalable, and comprehensive DL solution for modern malware detection. Table 1 provides a brief comparison of the related works discussed, with emphasis on methodologies, strengths, and limitations.

2. Proposed Methodology

This work proposes a hybrid malware detection model to enhance the detection rate. This work merges CNNs for feature extraction, RNNs for temporal analysis, and an attention mechanism to improve interpretability and focus on critical features. The framework follows a systematic approach. Data collection and preprocessing are followed by constructing a developed neural network model and also culminating in rigorous evaluation to validate the performance metrics. Fig. 1 shows the general stages of the proposed method while Fig. 2 depicts the layers and components of the proposed model.

2.1. Dataset collection

The proposed model was evaluated using three well-established datasets. The first dataset is EMBER [31], the second dataset is EMBER-Sim [32], and the third dataset is SoReL-20M [33]. For consistency and balanced evaluation, a total of 10,000 samples were utilised from each dataset, equally divided into 5,000 malware and 5,000 benign files. This design ensured that the model learned from a diverse yet balanced distribution of samples representing both malicious and legitimate software behaviours. The datasets included a mixture of static and dynamic features such as file metadata, byte-level characteristics, and API call sequences, which collectively provided comprehensive input for training and validation. All samples were pre-processed, normalized, and encoded into uniform feature representations before being split into 70% training set, 15% validation set, and also 15% testing set. This structure provided a stable foundation for evaluating accuracy, recall, precision, and other diagnostic metrics reported in the study.

2.2. Data preparation

Preparation of the data involved cleaning, decoding, and standardization of data in order to deal with inconsistencies and missing values. Statics (like file metadata) and dynamic behaviours (such as API call sequences) are learned. These characteristics were then normalized, coded and also tokenized to create numerical representations that can be utilised in DL. Lastly, the model was evaluated using a processed dataset by

dividing the dataset into three parts (70/15/15) of which training was done on 70% and the remaining 30% was divided equally between the validation and testing set. The sampling technique was stratified to bring the same classes of the subsets to equal status.

Table 1. Summary of related work

Ref.	Approach (Method Used)	Strength Points	Weak Points or Restrictions
19	Machine Learning and Deep Learning paradigms	Effective for zero-day attack detection and evolving threats	Struggles with adapting to rapidly changing threats
20	Lightweight random neural network for IoT attack detection	High accuracy in industrial IoT settings	Limited in scalability for broader applications
21	Convolutional Neural Networks (CNNs) for malware detection	Effective feature extraction for malware identification	Limited performance to time-dependent threats
22	Reactive Artificial Neural Network for predicting malicious activities	Proactive threat detection using user behavior patterns	Limited detection of Zero-Day attacks
23	ML-based intrusion detection systems for cybersecurity	Fast analysis of the cybersecurity threats	Challenges in achieving comprehensive detection across diverse datasets
24	AI techniques for IoT security using intelligent frameworks	Handles large IoT datasets efficiently	Scalability and architectural challenges
25	Bayesian hyperparameter to improve data analysis	Improves detection work	It faces issues with time-dependent malware patterns
26	AI for improving malware detection across networks	High detection rates in the complex networks	Struggles in adapting to a rapidly changing threat
27	AI systems for analyzing large datasets and detecting threats	Efficient detection of advanced threats	Requires enhanced data-handling capabilities
28	Artificial Neural Networks, including CNNs and RNNs	Adaptability to new malware variants	High rates of false positives
29	Deep Learning for autonomously identifying sophisticated malware	Autonomous identification for sophisticated threats	Ethical concerns about false positives and automation
30	Malware classification method by categorizing API behaviours.	Enhances detection capabilities	May struggle against advanced obfuscation techniques

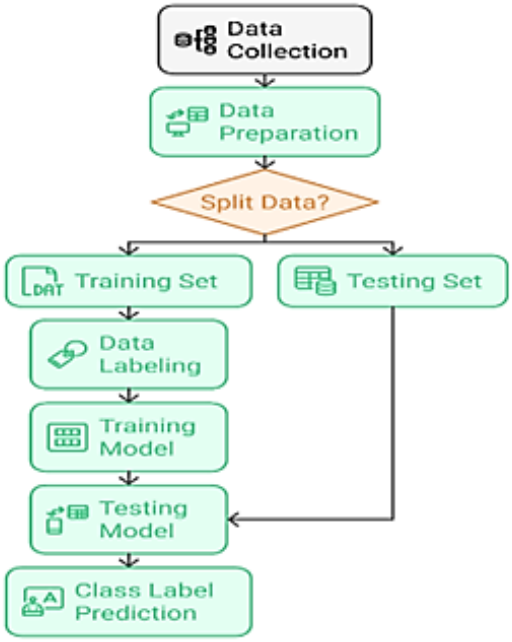

Fig. 1. General stages of the proposed method

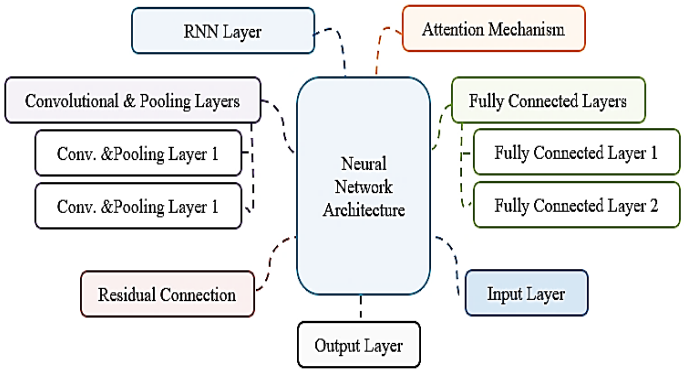

Fig. 2. The components and layers of the proposed model

2.3. The proposed model

The proposed neural network utilises innovative elements, including multi-scale convolutional layers for spatial feature extraction and recurrent layers (Long Short-Term Memory (LSTM)) to identify temporal patterns, supported by an attention mechanism for better interpretability and residual connections for efficient training. Both static elements and dynamic aspects of malware systems are effectively extracted through this joint network configuration for classification purposes. Early stopping alongside learning rate scheduling has been integrated into the training process to enhance performance and prevent overfitting. The methodology provides a systematic method to develop and validate the proposed hybrid neural network, proving its practical use for detecting malware in real-world environments. The graphic representation of the malware detection process neural network architecture can be found in Fig. 3. Algorithm 1 displays the steps for the hybrid neural network architecture for malware detection.

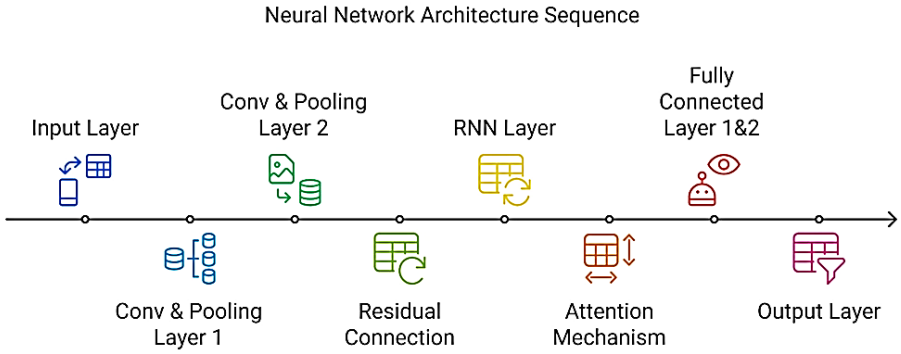

Fig. 3. The proposed neural network architecture

Algorithm 1: General steps for the hybrid neural network architecture based on hyperparameters defined in Table 2.

Input: Pre-processed data

Output: Classified data (malware vs. benign)

- The proposed hybrid neural network architecture for malware detection receives pre-processed data, merges the static features (e.g., file size, metadata) and dynamic behavioural features (such as API calls, network activity) as sequential feature vectors in order to preserve temporal relationships.
- A CNN block extracts spatial patterns using parallel convolutional layers with varying kernel sizes (3x3, 5x5), followed by batch normalization and LeakyReLU activation.
- The extracted features are passed to the RNN block (LSTM) to capture the temporal dependencies.
- The attention mechanism assigns importance weights to highlight the most relevant temporal patterns.
- Residual connections link the CNN and the RNN blocks to enable efficient training and better flow of features.
- The output is flattened and processed by fully connected layers with dropout to prevent overfitting.
- The final layer uses SoftMax activation for classification (malware vs. benign).

The proposed hybrid neural network architecture is as follows:

- Input Layer: The input layer takes pre-processed malware data as input. This includes static features (such as file size and metadata) and also dynamic behavioural features (like API call sequences and network activity logs). The features are represented as a sequence of vectors to preserve temporal relationships.
- Feature Extraction with CNNs: A few convolutional layers are used to process the input data to identify spatial patterns and local features. The model consists of batch normalization as a step to training stability and LeakyReLU activation as a step to eliminate gradient vanishing during training. The kernel scale, such as 3x3 and 5x5, is run concurrently to develop features at different levels of resolution and increase the diversity of features.

- Temporal Analysis with RNNs: The extracted features from the CNN layers are fed into a recurrent layer (LSTM) to analyse temporal dependencies within the sequence of features. This allows the model to capture the sequential behaviour of malware over time, such as patterns in API calls or network traffic.
- Attention Mechanism: The recurrent layer output passes through the attention mechanism to enhance the interpretability while allowing the model to focus on the key features. The mechanism utilises weights for different parts of the sequence structure, and also makes the model more effective at detecting important temporal patterns.
- Residual Connections: Residual connections are added between convolutional and recurrent layers to facilitate the flow of information and mitigate the degradation problem in deep networks. This also allows the network to learn low-level and high-level feature representations simultaneously.
- Fully Connected Layers with Dropout: After flattening the representation, the output of the attention mechanism passes through one or more fully connected layers. Multiple dropout layers become part of the network structure to minimise overfitting problems while enhancing model generalization. These layers abstract the features into higher-level representations.
- Output Layer: The final layer is fully connected with a SoftMax activation function for malware classification (malware vs. benign).
- Optimization and Training: A training schedule used AdamW as an optimiser and ReduceLROnPlateau as the learning rate scheduler.
- Focal loss corrects the common class imbalance in malware datasets, so the model directs more attention toward minority classes.

This network architecture utilises both RNNs and CNNs to perform spatial pattern detection through CNNs while processing time-dependent sequences through RNNs. The model contains interpretive attention modules and multiple-scale feature identification via parallel convolutional kernels with different sizes and employs residual learning with skip connections for efficient training of deep models. The method implements focal loss alongside its architecture to resolve class imbalances, enabling better rare and sophisticated malware discovery. The innovative approach fits perfectly for identifying complicated malware actions.

2.4. Training model

The training phase involved fine-tuning hyperparameters such as the learning rate and number of epochs to optimise the model's performance. Table 2 summarises the hyperparameter values used in the experiments.

Table 2. Hyperparameter values used in the proposed method

Hyperparameter	Value
Learning Rate	0.001
Batch Size	32
Number of Epochs	55
Optimizer	AdamW
Dropout Rate	0.3

The proposed network structure, balanced complexity, and computational efficiency according to the specifications, are presented in Table 3.

Table 3. Summary of network structural parameters

Layer Type	Details
Input Layer	Preprocessed malware data
Convolutional Layers	3x3 and 5x5 kernels, ReLU activation
Pooling Layers	Max Pooling
Recurrent Layers	LSTM with 128 units
Attention Mechanism	Self-attention

To reduce the overfitting, several regularization and optimisation strategies were applied. The network incorporated dropout layers (rate = 0.3) after fully connected blocks to prevent co-adaptation of neurons. Early stopping was also utilised to halt training once the validation loss stopped improving, while learning rate scheduling helped stabilize convergence. Additionally, focal loss was implemented to handle class imbalance, indirectly improving generalization. These measures ensured that the model maintained strong performance on unseen data without excessive fitting to the training samples.

3. Results and Discussion

A series of experiments was conducted to validate the effectiveness and robustness of the proposed hybrid neural network model for malware detection. The performance of the proposed model was evaluated using several standard classification metrics across multiple datasets. These evaluation criteria allow a complete assessment of the model's performance by defining its ability to detect malware correctly and its capacity to prevent incorrect classifications.

There are four major metrics used to assess the performance. True Positives (TP) are samples of malware that the system correctly detects as malware. False Positives (FP) are related to the malware not being detected in healthy samples. True Negatives (TN) represent harmless samples that are rightly identified as safe. False Negatives (FN) occur when a real malware is not detected and considered harmless. These four results are the basis of the computation of different performance measures regarding malware detection systems. According to them, the following metrics have been calculated as presented in Eqs. 1–8 [34].

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN) \quad (1)$$

Eq.1: measures overall correctness.

$$\text{Recall (Sensitivity)} = TP / (TP + FN) \quad (2)$$

Eq. 2: measures the model’s ability to detect actual malware.

Precision = TP / (TP + FP) (3)

Eq. 3: measures how many predicted malware are correct.

F1-Score = 2 × (Precision × Recall) / (Precision + Recall) (4)

Eq. 4: balances between recall and precision.

False Positive Rate (FPR) = FP / (FP + TN) (5)

Eq. 5: indicates the proportion of benign samples misclassified as malware.

Specificity (SPC) = TN / (TN + FP) (6)

Eq. 6: reflects the model's ability to correctly identify benign samples.

Negative Predictive Value (NPV) = TN / (TN + FN) (7)

Eq. 7: measures the probability that a sample predicted as benign is truly benign.

Diagnostic Efficiency (DE) = (TP + TN) / (TP + TN + FP + FN (8)

Eq. 8: measures the model's diagnostic reliability across all classes.

Training Time indicates the computational efficiency of the model during learning. These metrics together provide a comprehensive evaluation for detection accuracy and also robustness of the proposed model [34]. The model was trained and tested by using well-established DL frameworks, including TensorFlow and PyTorch. The accuracy, performance, and loss metrics show their evolution patterns, as presented in Fig. 4.

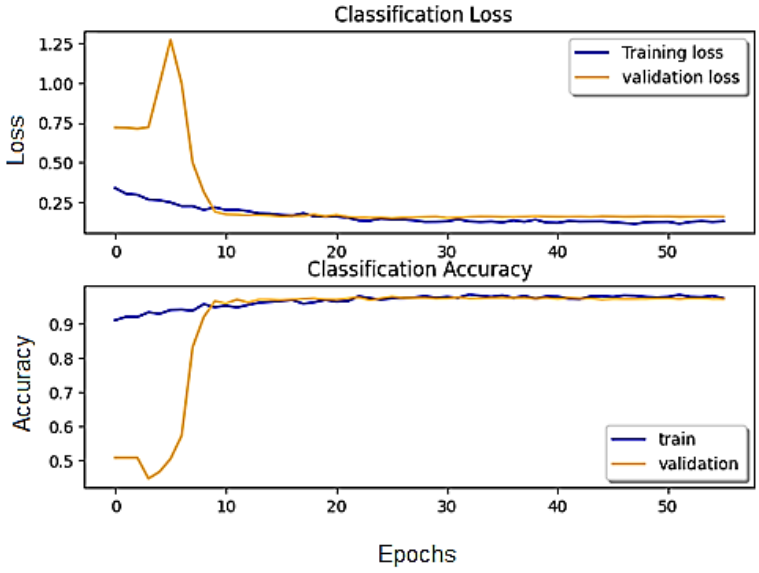

Fig. 4. The training curves for accuracy and loss

The experiments helped to prove that the proposed model provides better performance compared to traditional malware detection methods and various state-of-the-art DL solutions. The model exhibited superior detection abilities on various data and extremely low rates of false positives and negatives. Table 4 shows the full assessment measures of the proposed hybrid neural network on three benchmark datasets. The findings ascertain that the proposed model has high diagnostic reliability and constant computational efficiency in any dataset. These figures can be analysed entirely in terms of the performance of the various datasets, as shown in Fig. 5.

Table 4. Performance metrics on different datasets

Metrics	Dataset A (EMBER)	Dataset B (EMBER Sim)	Dataset C (SoReL-20M)
Accuracy	96.7%	96.5%	95.8%
Precision	96.1%	96.1%	95.3%
Recall	96.8%	96.8%	96.0%
F1-Score	96.4%	96.4%	95.6%
FPR	3.1%	3.3%	3.9%
SPC	96.9%	96.7%	96.1%
NPV	97.2%	97.0%	96.3%
DE	96.8%	96.6%	95.9%
Average Training Time per epoch	48 sec	52 sec	57 sec

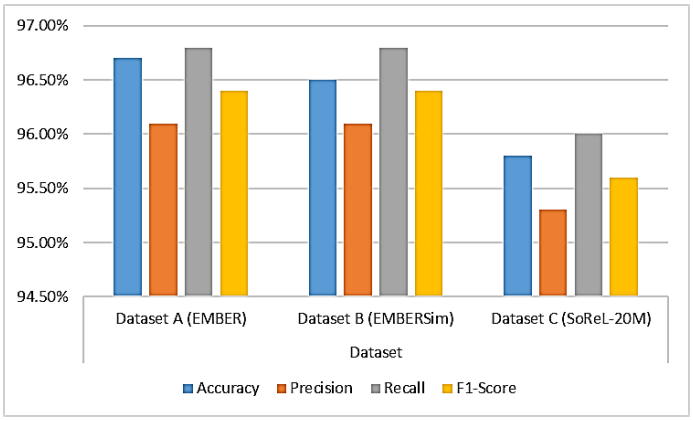

Fig. 5. Performance metrics on different datasets

To further assess classification performance, the confusion matrix was computed to illustrate the distribution of correct and incorrect classifications of malware and benign samples. As shown in Fig. 6, the proposed model achieved a high number of True Positives (TP) and also True Negatives (TN) with very few False Positives (FP) and False Negatives (FN). These would assure the strong discriminative capability of the proposed approach.

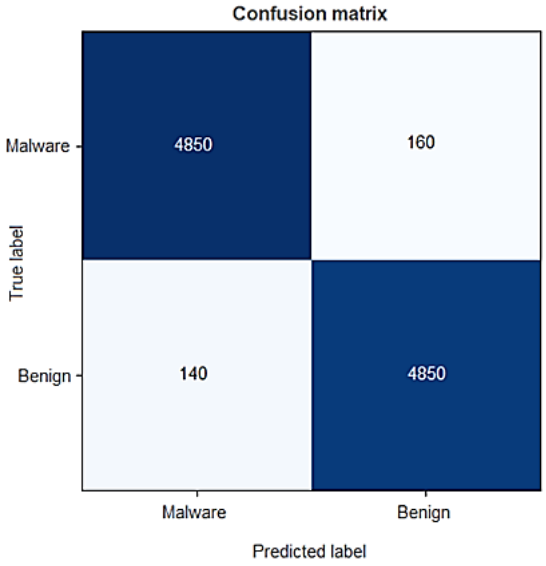

Fig. 6. Confusion matrix of the proposed hybrid neural network model

The proposed model demonstrated efficient training performance. The average training time per epoch was approximately 48 seconds for the EMBER dataset, 52 seconds for EMBER-Sim, and 57 seconds for SoReL-20M when trained on an NVIDIA T4 GPU with 16 GB VRAM and an Intel Xeon 2.3 GHz CPU. This indicates that the proposed framework achieves a favourable balance between computational cost and classification performance. The findings also show a direct correlation between the complexity of the dataset and the time of computation. Although the time of training on the bigger SoReL-20M was more time-consuming and the dataset was less diverse, the model was still highly accurate on all datasets. This shows that the proposed architecture is able to perform well without serious trade-offs in computations and hence has an ideal balance between accuracy and efficiency. Table 5 depicts that the proposed method has better performance compared to other methods on the EMBER dataset on which the proposed model achieved the best results. In this regard, the contrasted methods were selected because they follow similar deep-learning–based approaches for malware detection. All baseline models were re-implemented and evaluated on the same EMBER dataset to ensure a fair, transparent, and fully replicable comparison across accuracy and other standardized metrics. In addition, the model was effective in processing both the static and dynamic patterns of malware utilising the convolutional layers with recurrent layers and also attention mechanisms to identify the necessary patterns in the data. The attention mechanism in the model provided priority settings for the critical input characteristics and improving the detection results.

Table 5. Comparison with existing methods on the EMBER dataset

Ref.	Method	Accuracy	Precision	Recall	F1-Score
[22]	Machine Learning–based detection for predicting malicious activities.	94.1%	93.6%	94.4%	94.0%
[26]	Deep Learning model for autonomous malware classification.	94.2%	93.9%	94.3%	94.1%
[30]	Malware classification based on categorizing API behaviours.	96.0%	95.7%	96.2%	95.9%
Proposed Hybrid Model	CNN–RNN–Attention integrated neural network with a residual connection.	96.7%	96.1%	96.8%	96.4%

4. Conclusions

The current study introduced a hybrid DL system that uses CNNs, RNNs, and an attention mechanism to enhance the accuracy and reliability of malware identification. The suggested solution is effective in merging spatial and temporal feature acquisition and is computationally efficient due to residual connectivity and efficient training methods. Demonstrations on three benchmark datasets, EMBER, EMBER-Sim, and SoReL-20M, showed that the experimental evaluation can introduce uniform and better results than the available methodologies. This model has a high precision (96.1), recall (96.8), and F1-score (96.4), along with a high accuracy (96.7), which suggests consistent and balanced detection properties. In addition, a low false positive rate (0.3), high specificity (0.97), and negative predictive value (0.97) were the indicators that the model was reliable in differentiating between benign and malicious files. The diagnostic efficiency of 96.8%, which was achieved, also ascertained its strength. Besides its good predictive accuracy, the model also exhibited a realistic computational behavior, where each training cycle took less than one minute on regular hardware (NVIDIA T4 GPU). These findings indicated that the suggested framework delivers the best balance between the accuracy and efficiency of any detection mechanism, and it applies to real-life systems in the domain of cybersecurity, where the speed and the quality of any analysis have to be extremely high. Overall, the results proved that the combination of CNN, RNN, and attention mechanisms in one DL network has real effects in malware detection. The suggested model provided quantifiable benefits in a variety of measurement indicators, which makes it a reliable basis to continue studying the topic of adaptive and interpretable cybersecurity solutions.

Acknowledgment

The author thanks the Department of Computer Science in the College of Science at Mustansiriyah University for supporting this work.

References

[1] J. Jeon, J. H. Park and Y. S. Jeong, “Dynamic Analysis for IoT Malware Detection with Convolution Neural Network Model,” IEEE Access, vol. 8, pp. 96899–96911, 2020, doi.org/10.1109/ACCESS.2020.2995887.
[2] M. Woźniak, J. Siłka, M. Wieczorek and M. Alrashoud, “Recurrent Neural Network Model for IoT and Networking Malware Threat Detection,” IEEE Trans. Ind. Inform., vol. 17, pp. 5583–5594, 2020, doi.org/10.1109/TII.2020.3021689.
[3] P. Yadav, N. Menon, V. Ravi, S. Vishvanathan and T. D. Pham, “EfficientNet Convolutional Neural Networks-Based Android Malware Detection,” Computers & Security, vol. 115, p. 102622, 2022, doi.org/10.1016/j.cose.2022.102622.
[4] S. Jha, D. Prashar, H. V. Long and D. Taniar, “Recurrent Neural Network for Detecting Malware,” Computers & Security, vol. 99, p. 102037, 2020, doi.org/10.1016/j.cose.2020.102037.
[5] Y. Jian, H. Kuang, C. Ren, Z. Ma and H. Wang, “A Novel Framework for Image-Based Malware Detection with a Deep Neural Network,” Computers & Security, vol. 109, p. 102400, 2021, doi.org/10.1016/j.cose.2021.102400.
[6] J. Qiu, J. Zhang, W. Luo, L. Pan, S. Nepal and Y. Xiang, “A Survey of Android Malware Detection with Deep Neural Models,” ACM Comput. Surv., vol. 53, pp. 1–36, 2020, doi.org/10.1145/3417978.
[7] M. S. Mahdi and Z. L. Ali, “A Lightweight Algorithm to Protect the Web of Things in IoT,” in Proc. Int. Conf. Emerging Technol. Trends Internet Things Comput., pp. 46–60, Springer, Cham, 2021, doi.org/10.1007/978-3-030-97255-4_4.
[8] S. I. Imtiaz, S. ur Rehman, A. R. Javed, Z. Jalil, X. Liu and W. S. Alnumay, “DeepAMD: Detection and Identification of Android Malware Using High-Efficient Deep Artificial Neural Network,” Future Gener. Comput. Syst., vol. 115, pp. 844–856, 2021, doi.org/10.1016/j.future.2020.10.008.
[9] S. Jeon and J. Moon, “Malware-Detection Method with a Convolutional Recurrent Neural Network Using Opcode Sequences,” Information Sci., vol. 535, pp. 1–15, 2020,doi.org/10.1016/j.ins.2020.05.026.
[10] S. Berrios, D. Leiva, B. Olivares, H. Allende-Cid and P. Hermosilla, “Systematic Review: Malware Detection and Classification in Cybersecurity,” Applied Sciences, vol. 15, no. 14, p. 7747, 2025, doi.org/10.3390/app15147747.
[11] U. E. H. Tayyab, F. B. Khan, M. H. Durad, A. Khan and Y. S. Lee, “A Survey of the Recent Trends in Deep Learning Based Malware Detection,” J. Cybersecurity Privacy, vol. 2, no. 4, pp. 800–829, 2022., http://doi.org/10.3390/jcp2040041.
[12] W. Qiang, L. Yang and H. Jin, “Efficient and Robust Malware Detection Based on Control Flow Traces Using Deep Neural Networks,” Computers & Security, vol. 122, p. 102871, 2022, doi.org/10.1016/j.cose.2022.102871.
[13] F. A. Aboaoja, A. Zainal, F. A. Ghaleb, B. A. S. Al-Rimy, T. A. E. Eisa and A. A. H. Elnour, “Malware Detection Issues, Challenges, and Future Directions: A Survey,” Applied Sciences, vol. 12, no. 17, p. 8482, 2022, doi.org/10.3390/app12178482.
[14] Y. M. Mohialden et al., “Enhancing security and privacy in healthcare with generative artificial intelligence-based detection and mitigation of data poisoning attacks software,” Jordan Med. J., vol. 58, no. 4, 2024, doi.org/10.35516/jmj.v58i3.2712.
[15] M. S. Mahdi, “Innovative Neural Network Architecture for Progressive Windows Malware Detection via Adaptive Feature Fusion and Multi-stage Learning,” in Proc. Int. Conf. Cybersecurity and Artificial Intelligence Strategies, Springer Nature Switzerland, Cham, pp. 107–121, 2025, doi.org/10.1007/978-3-032-07244-3_7.
[16] S. Wang, Z. Chen, Q. Yan, K. Ji, L. Peng, B. Yang and M. Conti, “Deep and Broad URL Feature Mining for Android Malware Detection,” Information Sci., vol. 513, pp. 600–613, 2020, doi.org/10.1016/j.ins.2019.11.008.
[17] F. A. Abdulazeez, I. T. Ahmed and B. T. Hammad, “Examining the Performance of Various Pretrained Convolutional Neural Network Models in Malware Detection,” Appl. Sci., vol. 14, no. 6, p. 2614, 2024, doi.org/10.3390/app14062614.
[18] H. Babbar, S. Rani and W. Boulila, “NGMD: Next Generation Malware Detection in Federated Server with Deep Neural Network Model for Autonomous Networks,” Sci. Rep., vol. 14, p. 10898, 2024, doi.org/10.1038/s41598-024-61298-7.
[19] E. Kabanda, “Performance of Machine Learning and Other Artificial Intelligence Paradigms in Cybersecurity,” Oriental J. Comput. Sci. Technol., vol. 13, pp. 1–9, 2020,doi.org/10.13005/ojcst13.01.01.
[20] S. Latif, N. Ben Said, Z. Idrees, M. Frikha and H. Chaieb, “A Novel Attack Detection Scheme for the Industrial Internet of Things Using a Lightweight Random Neural Network,” IEEE Access, vol. 8, pp. 91065–91074, 2020, doi.org/10.1109/ACCESS.2020.2994079.
[21] I. H. Sarker, “Deep Cybersecurity: A Comprehensive Overview from Neural Network and Deep Learning Perspective,” SN Comput. Sci., vol. 2, pp. 1–14, 2021, doi.org/10.1007/s42979-021-00535- .
[22] P. Yamcharoen, O. Folorunsho, and A. Bayewu , “Application of Reactive Artificial Intelligence Model to Predict Malicious Activities,”

Adv. Multidiscip. Sci. Res. J., vol. 9, pp. 45–60, 2021, doi.org/10.22624/AIMS/MATHS/V9N2P5.
[23] M. Alharbi, A. M. El-Sherbeeny and M. A. El-Meligy, "Analyzing the Impact of Cybersecurity-Related Attributes for Intrusion Detection Systems," Sustainability, vol. 13, pp. 1–15, 2021, https://doi.org/10.3390/su132212337.
[24] M. Abdullahi, A. Bustamam, S. Muhsin and B. M. Ali, "Detecting Cybersecurity Attacks in Internet of Things Using Artificial Intelligence Methods: A Systematic Literature Review," Electronics, vol. 11, pp. 1–20, 2022, https://doi.org/10.3390/electronics11020198.
[25] F. Algorain and J. Clark, "Bayesian Hyper-Parameter Optimisation for Malware Detection," Electronics, vol. 11, no. 10, p. 1640, 2022, doi.org/10.3390/electronics11101640.
[26] K. Komarudin, S. Sunardi and Y. Wihardi, "Exploring the Effectiveness of Artificial Intelligence in Detecting Malware and Improving Cybersecurity in Computer Networks," Eduvest - J. Univers. Stud., vol. 3, pp. 1–12, 2023, http://doi.org/10.59188/eduvest.v3i4.793.
[27] S. M. A. Rizvi, "Enhancing Cybersecurity: The Power of Artificial Intelligence in Threat Detection and Prevention," Int. J. Adv. Eng. Res. Sci., vol. 10, pp. 115–123, 2023, http://doi.org/10.22161/ijaers.105.8.
[28] R. Ojha, "Use of Artificial Neural Networks to Detect and Prevent Cybersecurity Threats," NPRC J. Multidiscip. Res., vol. 1, pp. 32–45, 2024, http://doi.org/10.3126/nprcjmr.v1i6.71754.
[29] S. Shahana, "AI-Driven Cybersecurity: Balancing Advancements and Safeguards," J. Comput. Sci. Technol. Stud., vol. 6, pp. 129–140, 2024, http://doi.org/10.32996/jcsts.2024.6.2.9.
[30] D. Syeda and M. Asghar, "Dynamic Malware Classification and API Categorisation of Windows Portable Executable Files Using Machine Learning," Applied Sciences, vol. 14, no. 3, p. 1015, 2024, doi.org/10.3390/app14031015
[31] D. Hoogla, "EMBER: 2018 Dataset for Training Static PE Malware Machine Learning Models," Kaggle, 2018. [Online]. Available: https://www.kaggle.com/datasets/dhoogla/ember-2018-v2-features.
[32] D. G. Corlatescu, A. Dinu, M. P. Gaman and P. Sumedrea, "EMBERSim: A Large-Scale Databank for Boosting Similarity Search in Malware Analysis," in Adv. Neural Inf. Process. Syst., vol. 36, pp. 26722–26743, 2023.
[33] R. Harang, E. M. Rudd et al., "SOREL-20M: A Large-Scale Benchmark Dataset for Malicious PE Detection," GitHub, 2020. [Online]. Available: https://github.com/sophos/SOREL-20M.
[34] M. S. Mahdi, Z. L. Ali, A. R. Rashid, N. K. Ibrahim and A. W. Abdulghafour, "A Hybrid Deep Learning Model for Facial Emotion Recognition: Combining Multi-Scale Features, Dynamic Attention, and Residual Connections," in Proc. 13th Int. Conf. Appl. Innov. IT (ICAIIT), vol. 13, no. 2, pp. 69–77, Jun. 2025, doi.org/10.25673/120395.