Eng. Sci., 2025, **37**, 1722

Engineered Science
DOI: https://dx.doi.org/10.30919/es1722

Enhancing Ransomware Detection with a Hybrid Deep Learning Approach: Integrating Convolutional Neural Networks and Long Short-Term Memory Networks for a Robust Cybersecurity Solution

Priynka Sharma* and Kaylash Chaudhary

Abstract

Ransomware attacks continue to be a significant and evolving cybersecurity threat, with traditional detection techniques often unable to identify new and sophisticated variants. Signature-based and heuristic methods, which rely on pre-existing knowledge of malicious behaviors, frequently fail to detect novel strains, highlighting the need for more dynamic, data-driven detection systems. In this paper, we propose a hybrid deep learning framework that integrates Convolutional Neural Networks (CNNs) and Long Short-Term Memory Networks (LSTMs) to address the limitations of existing detection approaches. The CNN extracts spatial features from raw data, such as file byte sequences, system calls, and network traffic, crucial for identifying ransomware traits. Meanwhile, the LSTM captures temporal dependencies and sequential patterns, essential for detecting dynamic ransomware behaviors over time. The proposed model is evaluated on a comprehensive ransomware dataset comprising 1,000 features, 10,000 samples, and six distinct classes, encompassing both benign and ransomware behaviors. Experimental results demonstrate that the hybrid CNN-LSTM model outperforms traditional methods significantly. By leveraging the strengths of both CNNs for feature extraction and LSTMs for sequence modeling, the proposed hybrid model provides a more accurate, adaptive, and scalable solution for real-time ransomware detection, thereby reducing false positives and enhancing the robustness of cybersecurity systems against emerging threats.

Keywords: Artificial intelligence; Convolutional neural networks; Cybersecurity; Deep learning; Long short-term memory; Ransomware; Techniques.
Received: 03 January 2025; Revised: 13 April 2025; Accepted: 14 April 2025
Article type: Research article.

1. Introduction

Ransomware attacks have emerged as one of the most damaging and pervasive cybersecurity threats in recent years.[1-2] These attacks involve malicious software that encrypts a victim's data and demands a ransom for the decryption key. High-profile incidents, such as the WannaCry and NotPetya attacks, have demonstrated the potential scale of damage that ransomware can cause. The WannaCry attack, for instance, affected over 200,000 computers across 150 countries in 2017, crippling healthcare systems, businesses, and government organizations worldwide.[3-5] The ever-evolving nature of ransomware and its ability to adapt and bypass traditional security measures highlight the growing need for more advanced detection techniques.[6] Conventional methods of detecting ransomware, such as signature-based detection and heuristic analysis, rely heavily on predefined patterns and known characteristics of malicious software. Signature-based methods, which compare files and behaviors against known virus signatures, are limited in detecting new variants that may not share these predefined signatures.[7-10] For example, a new strain of ransomware may encrypt files using an unfamiliar algorithm or engage novel techniques to avoid detection, making the signature-based systems ineffective. Similarly, heuristic-based methods, which analyze behavior patterns to identify potential threats, often struggle with false positives and may fail to detect sophisticated and evolving attacks.

In response to these challenges, the cybersecurity community has increasingly turned to machine learning and, more recently, deep learning techniques to develop more adaptive and robust detection systems.[11-12] Deep learning models, particularly CNNs and LSTMs, have shown

School of Information Technology, Engineering, Mathematics and Physics, The University of the South Pacific, Suva, 1168, Fiji
**Email:* priynka.sharma@usp.ac.fj (P. Sharma)

significant promise in image recognition, speech processing, and natural language processing due to their ability to learn complex patterns from large datasets.[13] CNNs effectively extract hierarchical features from raw data, making them well-suited for tasks like malware classification and intrusion detection. For example, CNNs can be trained to identify abnormal byte sequences or suspicious network traffic patterns indicative of ransomware activity. LSTMs, on the other hand, excel at modeling sequential data and can capture long-term dependencies, such as the temporal patterns in ransomware behaviors that evolve.

The rapidly evolving ransomware threat has necessitated advanced detection mechanisms beyond traditional methods. Signature-based techniques, such as those engaged by Crypto Drop,[9-14] rely on predefined malware signatures to identify attacks. While effective for known threats, these methods falter against zero-day ransomware due to their inability to adapt to novel attack patterns.[15] Similarly, heuristic approaches analyze system behavior for anomalies but often suffer from high false-positive rates, mainly when dealing with dynamic and obfuscated ransomware variants. Machine learning ML techniques have gained traction in ransomware detection due to their ability to identify patterns from large datasets.[15] For example, Sgandurra[10] introduced a behavioral profiling framework using Random Forest classifiers to analyze file operations and system calls. Although it achieved admirable detection rates, the framework required significant manual feature engineering, limiting its adaptability. Vinayakumar[16] improved scalability by engaging ensemble learning methods, yet challenges remained in handling sequential patterns and evolving ransomware strategies.

Deep learning has emerged as a robust alternative, automating feature extraction and effectively handling complex datasets. CNNs have been extensively used in malware detection; for instance, Nataraj[17] utilized CNNs to classify malware families by converting binary data into image representations. On the other hand, LSTM networks have proven effective in capturing temporal dependencies, as demonstrated by Athiwaratkun and Stokes,[18] who applied LSTMs to model dynamic malware behaviors. Hybrid approaches that integrate CNNs and LSTMs are particularly promising, as they combine spatial feature extraction with temporal sequence modeling. Tang *et al*.[18] showcased a hybrid model for advanced persistent threat detection, achieving superior accuracy and scalability compared to standalone models. Our research builds on these findings by applying a hybrid CNN-LSTM architecture to a ransomware dataset, utilizing its 1000 features and temporal characteristics to enhance detection performance. Table 1 shows the summary of traditional signature-based, heuristic, and standalone machine learning methods by enabling the detection of novel and evolving ransomware variants in the literature.

This paper proposes a hybrid deep-learning approach that combines CNNs and LSTMs to improve ransomware detection. The CNN component extracts spatial features from raw ransomware data, such as file byte sequences, system calls, and network traffic. In contrast, the LSTM component captures the temporal dependencies and sequential patterns associated with ransomware behaviors.[19] This approach allows the model to detect known ransomware variants and generalize to previously unseen attack types. We evaluate the performance of the proposed hybrid model on a ransomware dataset consisting of 1,000 features, 10,000 samples, and six distinct classes, which represent both benign and malicious activities. The experimental results show that our hybrid CNN-LSTM model significantly outperforms traditional machine learning methods and standalone CNN and LSTM models in terms of accuracy, precision, recall, and F1 score. The integration of CNNs and LSTMs provides a robust and scalable solution for real-time ransomware detection, offering a more effective defense against evolving ransomware threats. By using the strengths of both models, the proposed approach not only enhances detection accuracy but also reduces false positives, ultimately improving the effectiveness of cybersecurity systems in safeguarding against increasingly

Table 1: Summary of Related Work.

Study	Method	Strengths	Limitations
[20]	Signature-based	Accurate for known threats	Ineffective for zero-day ransomware
[21-23]	Behavioral profiling (RF)	Robust against known patterns	High manual feature engineering
[24]	Ensemble learning	Improved scalability	Struggles with sequential dependencies
[25]	CNN-based image analysis	Automates spatial feature extraction	Limited for dynamic behaviors
[26-28]	LSTM for malware modeling	Effective for temporal patterns	Inefficient for spatial analysis
[29-30]	Hybrid CNN-LSTM	Combines spatial and temporal modeling	Evaluated on general malware, not ransomware

sophisticated ransomware attacks.

The contributions of the paper are as follows:

- **Hybrid deep learning model** – this paper integrates Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks to extract both spatial and temporal features for enhanced ransomware detection.
- **Analysis** – the performance of the hybrid CNN-LSTM model in terms of accuracy, precision, recall, and F1 score when compared to traditional ML classifiers and standalone deep learning models.
- **Framework** – this provides a scalable framework suitable for real-time deployment in cybersecurity systems, capable of handling complex and large-scale ransomware threats.

The remainder of this paper is structured to provide a comprehensive understanding of the proposed hybrid deep learning approach for ransomware detection. Section 2 outlines the problem statement, emphasizing the limitations of traditional and machine learning-based detection methods, and explains the motivation behind adopting a more adaptive solution. Section 3 explores the computational dynamics of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks, examining their individual strengths and the rationale for combining them into a hybrid model. Section 4 presents the methodology, detailing the dataset generation, model architecture, training procedures, and evaluation techniques used to assess model performance. Section 5 discusses the experimental results, comparing the hybrid model's effectiveness against traditional machine learning classifiers and standalone deep learning models, and highlights its advantages, limitations, and potential areas for future work. Finally, Section 6 concludes the paper by summarizing the key findings and contributions, underscoring the practical implications of the proposed hybrid model for real-time and scalable ransomware detection in cybersecurity environments.

2. Problem statement

Ransomware attacks have become increasingly sophisticated, employing advanced techniques such as polymorphism and dynamic behavior to evade traditional security mechanisms. Existing detection methods, including signature-based approaches, are ineffective against zero-day ransomware, while heuristic techniques suffer from high false-positive rates, undermining their reliability. Moreover, current machine learning-based solutions heavily rely on manual feature engineering and fail to capture the spatial and temporal characteristics of ransomware behavior comprehensively. This lack of adaptive, accurate, and robust detection mechanisms exposes systems to significant risks from known and emerging ransomware variants, necessitating a more effective and robust approach to ransomware detection.

2.1 Importance of conducting this research

The global impact of ransomware cannot be overstated. High-profile attacks, such as WannaCry and the Colonial Pipeline incident, have disrupted critical infrastructure, cost billions of dollars, and jeopardized sensitive data.[31] With ransomware becoming more pervasive, the healthcare and finance industries face significant operational risks. Addressing these challenges requires a shift from reactive to proactive and adaptive detection mechanisms. This research is critical as it proposes a hybrid deep learning approach combining CNNs and LSTM networks. The proposed method addresses key deficiencies of existing techniques by using CNNs for spatial feature extraction and LSTMs for temporal pattern recognition. It enables the detection of known and previously unseen ransomware variants, offering a robust and scalable solution for real-time cybersecurity applications.

The outcomes of this study have the potential to advance cybersecurity defenses, reduce false positives, and establish a framework for addressing future ransomware threats. This research contributes to safeguarding critical systems and data against a rapidly evolving threat in an era where digital safety is paramount.

3. Computational dynamics of convolutional neural networks (CNNs) and long short-term memory (LSTM) networks

The computational dynamics of CNNs and LSTM networks involve understanding the mechanisms and processes that drive learning and decision-making within these architectures.[32] These dynamics are shaped by the structure of the networks, the operations performed in each layer, parameter updates, and the efficient use of computational resources. Understanding these dynamics is crucial for optimizing the performance of both CNNs and LSTMs and applying them effectively to a wide range of real-world problems.

3.1 Convolutional neural networks (CNNs)

At the heart of CNNs is the convolution operation, which involves sliding a filter (or kernel) across the input data to extract localized features, such as edges, textures, and patterns. [33] Each convolutional layer applies multiple filters to the input, generating feature maps highlighting vital characteristics. These filters are learned during training, allowing the network to adapt to specific patterns within the data. [15-16] The computational complexity of this process depends on factors such as the size of the input image, the number of filters, and the filter size itself. In practice, convolution operations are highly parallelizable, which enables efficient processing, particularly on GPUs. Pooling layers follow convolutional layers and reduce the spatial dimensions of the feature maps, making the network more computationally efficient while retaining the most significant features. Max-pooling and average-pooling are the most common techniques, both of which reduce the size of the input feature map by selecting the maximum or average value from

small regions. Pooling reduces computational costs and helps mitigate overfitting by providing spatial invariance, allowing the network to generalize better to unseen data. Fully connected layers, typically placed toward the end of a CNN architecture, convert the high-dimensional feature maps into a 1D vector for classification tasks. These layers make decisions by combining all extracted features into a final output, often a class label in classification problems. The computational load of fully connected layers increases with the number of neurons, making them one of the more resource-intensive parts of the network, particularly in deeper architectures.

Backpropagation, the core mechanism for training CNNs, plays a significant role in computational dynamics. During backpropagation, gradients are calculated for each parameter (weights and biases) in the network and used to update the parameters through an optimization algorithm such as Stochastic Gradient Descent (SGD) or Adam.[34] This process involves calculating gradients for each layer, which requires significant matrix operations and the application of the chain rule of calculus. The greater the number of layers and parameters in a network, the more computational resources are required for training.

CNNs also benefit from various computational optimizations that enable efficient handling of large-scale data processing. Techniques such as parameter sharing (using the same filter across the entire input) and sparse connections (limiting the number of neurons connected) help reduce the overall computational burden. Additionally, advancements like GPU acceleration and hardware-specific optimizations (*e.g.*, Tensor Processing Units) have significantly enhanced the efficiency of CNNs, enabling them to process vast amounts of data quickly and accurately. The computational dynamics of CNNs involve a balance between the complexity of layer-wise operations, the optimization of model parameters, and the efficient use of computational resources. By utilizing parallelism, reducing dimensionality, and using hardware acceleration, CNNs can effectively handle large datasets and tackle complex tasks, making them a powerful tool in modern artificial intelligence applications.

The detailed structure and key components of the CNN, highlighting its capability to extract and process spatial features effectively, are illustrated in Fig. 1 below.

3.2 Long short-term memory (LSTM) networks

LSTMs are designed to handle sequential data, such as time series, speech, and text, by capturing long-range dependencies and learning from sequences of inputs.[35] The computational dynamics of LSTMs are heavily influenced by their unique architecture, which incorporates gates to control the flow of information through the network. These gates enable LSTMs to maintain memory over time, which is crucial for handling tasks that involve sequential dependencies. At the core of LSTMs are three key gates: the input gate, the forget gate, and the output gate. [23-25] These gates regulate the information flowing into the memory cell at each time step. The forget gate decides what data to discard from the previous time step's memory, the input gate determines what new information to store, and the output gate controls what information is passed to the next layer. The computational complexity arises from the operations required for each gate, such as element-wise matrix multiplication and the application of activation functions like the sigmoid and Tanh functions.

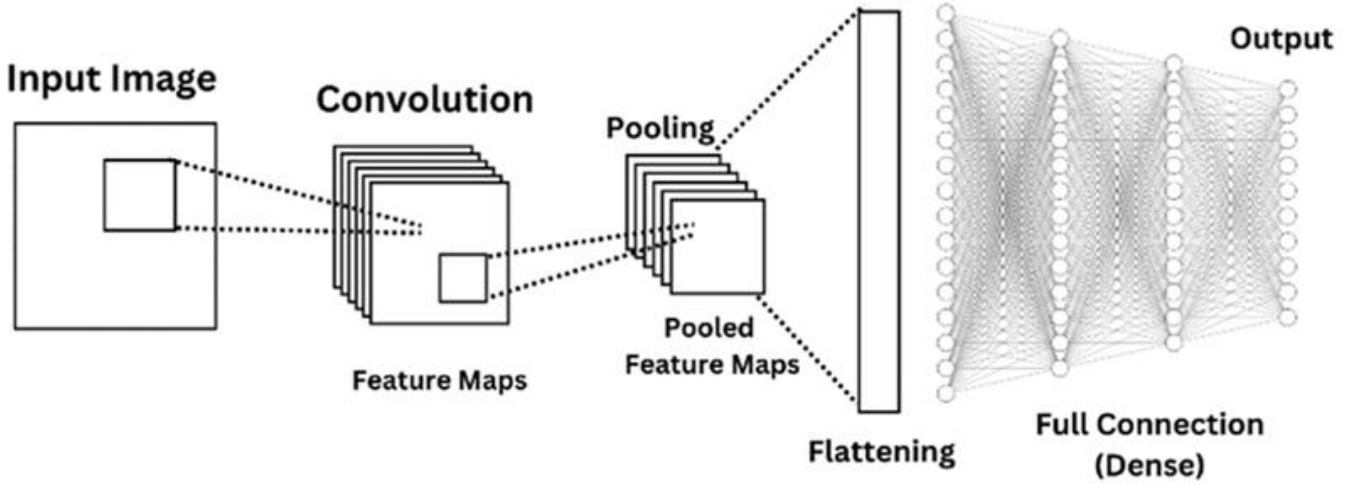

Fig. 1: Detailed Visualization of the CNN Architecture Reproduced from.[14]

The memory cell in an LSTM stores the network's internal state across time steps. This memory enables the network to retain long-term dependencies, which are critical for many sequential tasks. The LSTM's ability to selectively forget or remember the information at each time step helps mitigate the vanishing gradient problem often encountered in traditional RNNs, making them more suitable for long-term sequence modeling. Training LSTMs involves backpropagation through time (BPTT), a variant of the standard backpropagation algorithm. BPTT calculates gradients across time steps by

unrolling the network and computing the gradient of the loss function with respect to the weights at each time step.[26] This process requires significant memory and computational resources, as each time step involves computing gradients for the gates and updating weights accordingly. The longer the sequence, the greater the computational burden of BPTT, as gradients need to be propagated over more time steps.

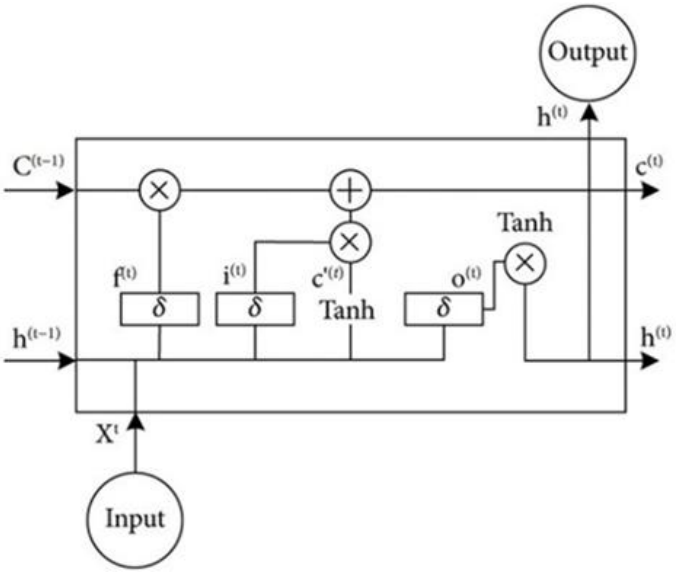

Fig. 2: Detailed Visualization of the LSTM Architecture Reproduced from.[18]

LSTMs also benefit from optimizations such as parameter sharing (where the same weights are used at each time step) and batch processing, which allows multiple sequences to be processed simultaneously. However, the sequential nature of LSTMs limits their parallelization, making them less efficient on hardware similar to GPUs than CNNs.[27] Like CNNs, LSTMs can be trained using optimization algorithms such as SGD or Adam. During training, the network adjusts its parameters (weights and biases) to minimize the loss function, which involves computing gradients and updating weights based on the error at each time step.[28] Hence, the computational dynamics of LSTMs revolve around their ability to capture long-range dependencies in sequential data through gate mechanisms and memory cells. While computationally intensive due to their recurrent structure and backpropagation through time, LSTMs are highly effective at modeling time-series data and tasks that require long-term dependencies.

The comprehensive architecture and critical elements of the LSTM network, emphasizing its ability to capture temporal dependencies in sequential data, are illustrated in Fig. 2 above.

3.3 Comparison of CNN and LSTM architectures: importance of combining in hybrid models

Integrating CNN and LSTM networks into a hybrid model offers a powerful approach for learning spatial and temporal features from complex datasets. CNNs are highly effective at identifying spatial patterns within data, such as those found in images or videos, where they can distinguish key features like edges, textures, and objects. However, CNNs are not well-equipped to capture the temporal dependencies inherent in sequential data, where LSTMs offer significant advantages [29]. LSTMs are designed to process sequential information and retain long-term dependencies, making them ideal for handling time-series data and tasks requiring the model to "remember" information over time. [30-31] A hybrid CNN-LSTM model can efficiently learn spatial and temporal relationships by merging these two architectures, resulting in a more robust and holistic feature representation and enhanced model performance. In tasks that involve both spatial and temporal components, combining CNNs and LSTMs can notably improve accuracy. A clear example of this synergy is video analysis, where CNNs extract spatial features from individual frames while LSTMs model the temporal relationships between frames. This combination enables the model to understand the sequence of events, which is critical for tasks such as action recognition, video classification, and anomaly detection in time-series data, where the spatial content of each frame and the temporal dynamics between them are essential.

A hybrid CNN-LSTM architecture is highly adaptable and can be applied across various domains, including multimedia processing (such as video, speech, and text), time-series forecasting, and medical data analysis. These fields often require the ability to interpret spatial and temporal data to make accurate predictions, making the CNN-LSTM hybrid

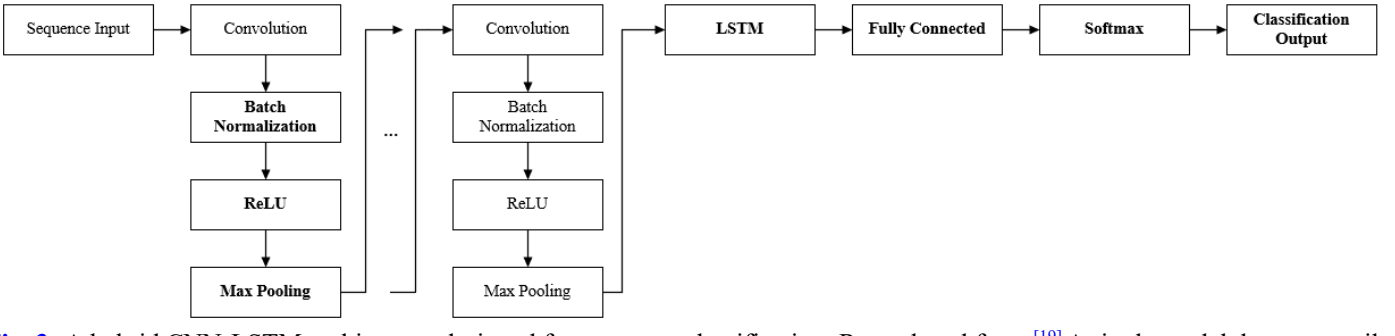

Fig. 3: A hybrid CNN-LSTM architecture designed for sequence classification. Reproduced from.[19] A single model does not easily capture intricate patterns.

model an ideal solution.[32] By the strengths of both CNNs and LSTMs, this architecture provides a practical framework for addressing a range of complex tasks. Therefore, combining CNN and LSTM architectures in a hybrid model improves its generalization capabilities. By learning spatial features and temporal dependencies, the model becomes better equipped to adapt to new, unseen data, enhancing its robustness. This increased adaptability ensures the hybrid model can accurately predict outcomes in a broader range of scenarios. It is particularly valuable for applications where Fig. 3 represents a hybrid CNN-LSTM architecture for sequence classification tasks. It integrates CNNs for feature extraction and LSTMs for temporal sequence modeling, followed by fully connected layers for final classification.

4. Proof of concept: methodology

This study employs a systematic approach to evaluate the hybrid deep learning framework for ransomware detection, which combines CNNs and LSTMs. Table 2 shows the crucial features of CNN and LSTM. The methodology encompasses data preparation, model architecture, training, and performance evaluation steps, as outlined below:

4.1 Data description

The dataset was generated in MATLAB to simulate a ransomware detection scenario. It consists of 10,000 samples, each with 1,000 features representing various attributes related to system behaviors or network traffic. The 6 features were generated using MATLAB's rand function, producing continuous values between 0 and 1 for each attribute. These features simulate real-world data such as file byte sequences, system call patterns, or network activity. Additionally, each sample is assigned a class label, randomly chosen from six categories (0 to 5), representing different behaviors, including benign and various types of ransomware. The labels were assigned using the randi function, ensuring a balanced distribution across the dataset. The generated data, consisting of features and labels, was then combined into a table and saved to an Excel file (synthetic_ransomware_dataset.xlsx) for further analysis. This dataset is designed to aid in the testing and evaluation of machine learning models, particularly in classifying data such as benign or malicious (ransomware).

Although the features are generated randomly, they are conceptually designed to represent high-dimensional attributes observed in actual ransomware-related system behavior. These include proxy representations of file entropy changes, I/O activity, system call frequency vectors, and network access patterns. The abstraction provides a way to stress-test detection models under high variability and dimensionality, simulating a diverse behavioral environment. By using controlled synthetic noise, this approach enables early-phase benchmarking of classification models without introducing domain-specific biases.

Table 2: Key Mathematical Features of CNN and LSTM.

Feature	CNN	LSTM	Hybrid CNN-LSTM
Core Operation	Convolution: $(X * K)$	Memory gates: (F_t, I_t, O_t)	Convolution for feature extraction and LSTM for sequence learning
Activation Function	ReLU: $(\max(0, X)$	Sigmoid and tanh: $(\sigma(x), \tanh(x))$	ReLU in CNN, Sigmoid/Tanh in LSTM
Feature Extraction	Local spatial patterns (*e.g.*, malware signatures)	Temporal dependencies (*e.g.*, ransomware behavior over time)	CNN detects spatial features; LSTM learns sequential dependencies
Output	Spatial features	Temporal outputs	Combined spatial features from CNN and sequential outputs from LSTM

4.2 Data reliability

The synthetic dataset was primarily used in this experiment to provide a controlled and flexible environment for evaluating the machine's performance and deep models in ransomware detection. Since the dataset was artificially generated, it allowed for quick experimentation and testing without the complexities and limitations associated with real-world data, such as privacy concerns, data sparsity, or the need for extensive preprocessing. The random generation of features and class labels ensured that the models were tested on a large and diverse dataset. This allowed for an initial assessment of the models' scalability, computational efficiency, and basic classification capabilities. Using the synthetic dataset also offered the advantage of efficiently operating on the number of samples, features, and classes, making it ideal for experimenting with different configurations and model architectures.[33-35] This flexibility was particularly valuable in the early stages of the tested model development, where it was crucial to quickly test various approaches and identify potential issues before moving to more complex, real-world data. Additionally, the synthetic dataset provided a neutral baseline for benchmarking different algorithms, removing biases or inherent patterns that might exist in real-world data. This allows researchers to focus on evaluating the core

functionalities of their models, such as feature extraction, classification accuracy, and computational efficiency, without being hindered by domain-specific details. To ensure the integrity of the experimental analysis, a mathematical validation framework is presented in Table 3, which confirms the reliability, neutrality, and practical applicability of the synthetic dataset.

While the synthetic dataset provides a neutral and flexible environment for testing model architectures, we acknowledge that it may not fully capture the intricate variability and dependencies present in real-world ransomware behaviors. As such, model performance on synthetic data should be interpreted as a preliminary benchmark rather than a definitive indicator of real-world robustness. However, the controlled structure of the synthetic dataset allows us to isolate and evaluate model capabilities such as learning capacity, noise tolerance, and dimensional scalability. This forms a foundational step before deploying models on more complex, real-world telemetry data. In future work, we aim to validate our approach using publicly available datasets, such as the EMBER malware dataset or endpoint logs from honeypot environments, and further test the generalizability of our

Table 3: A Mathematical Validation Framework Ensures the Synthetic Dataset's Reliability, Neutrality, and Practical Alignment for Experimental Analysis.

Validation Criterion	Mathematical Measure	Description	Objective
Feature Unbiasedness	$E[x_{ij}] = \mu j,\ Var(x_{ij}) = \sigma j2$	Ensures the feature values are centered around realistic means with appropriate variance.	Prevents skewed or unrealistic data distribution.
Sample Diversity	ReLU: d(Xi,Xk)=$\sqrt{\sum \frac{M}{j} = 1(X_{ij} - X_{kj})^2}$, ∀i≠k	Computes pairwise Euclidean distances between samples.	Prevents overfitting by ensuring diversity in samples.
Feature Variability	(Var (Xj)>ϵ)	Ensures each feature has sufficient variance.	Prevents underfitting by ensuring meaningful features.
Class Balance	(($P(C\_k) = \backslash frac\{N\_k\}\{N\},\backslash$)	P(C_k) - P(C_l)	\leq \delta, \ \for all k, l) sequential outputs from LSTM
Feature Range Alignment	$(X_{ij} \in [a_j, b_j])$	Ensures feature values fall within realistic ranges observed in real-world scenarios.	Aligns synthetic data with real-world standards.

models under realistic conditions.

4.3 Experiment setup

The proposed hybrid deep learning approach, integrating CNN and LSTM networks, was implemented for ransomware detection using synthetic ransomware datasets. The dataset was preprocessed by normalizing features to a [0, 1] range, ensuring consistency and eliminating scale biases, as supported by existing literature practices for improved model convergence (Goodfellow *et al.*)[10] The dataset was partitioned into training (80%) and testing (20%) sets, consistent with prior studies on ransomware classification. Temporal data analysis used LSTM layers to capture sequential dependencies, a proven method in time-series classification tasks. (Hochreiter & Schmidhuber).[36] The architecture was designed with CNN layers to extract spatial features and LSTM layers for temporal dependencies. This combination aligns with the methodologies active in malware detection research, where hybrid models have demonstrated superior performance in extracting complex data representations. (Saxe & Berlin).[10] The architecture (3-5 CNN layers and 50-150 LSTM units) was selected based on preliminary exploratory experiments and guided by prior works in malware and sequence anomaly detection. We evaluated multiple configurations by varying the number of convolutional layers (3–5), kernel sizes (3 to 5), and LSTM units (ranging from 50 to 150). These ranges were found to strike a balance between performance and training time. Although a formal ablation study was not conducted in this version of the work, we observed diminishing returns beyond 5 CNN layers or 150 LSTM units, as well as increased overfitting. These preliminary findings guided our decision to adopt mid-range configurations within these limits. Future versions of this study will include a comprehensive ablation analysis to quantify the effect of architectural components on performance systematically. To mitigate overfitting, dropout layers were incorporated, as suggested by Srivastava *et al.*,[19] with a dropout rate of 30%. This value was selected based on initial tuning, where we tested rates of 20%, 30%, and 50%. A 30% dropout consistently provided the best balance between regularization strength and training stability. Other techniques, such as L2 weight regularization and batch normalization, were also explored; however, they yielded only marginal improvements and were not included in the final model.

The Adam optimizer, known for its adaptive learning rates

(Kingma & Ba),[21] was engaged with a learning rate of 0.005, and the network was trained for 30 epochs with a mini-batch size of 128. The evaluation focused on accuracy and confusion matrix analysis, providing insights into class-wise detection capabilities. This is consistent with methodologies from ransomware classification literature, such as those by Shijo and Salim,[19] where confusion matrices highlighted model robustness in multi-class scenarios. The trained model was subsequently saved for future inference tasks, demonstrating its potential for deployment in real-time systems. The diagram (Fig. 4) below depicts the entire process.

Fig. 4 illustrates the deep learning channel, that begins

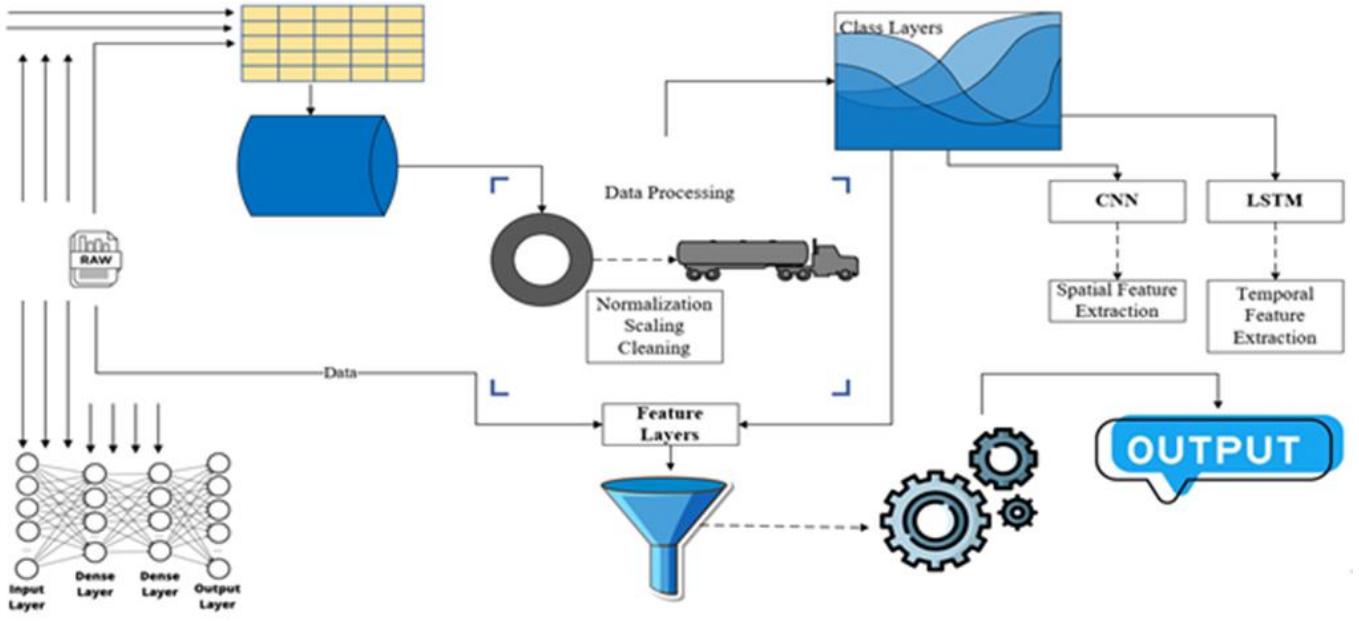

Fig. 4: A hybrid machine learning pipeline combines a CNN for spatial features and an LSTM for temporal features, with preprocessing for data classification.

by collecting raw data, followed by a preprocessing stage to clean and scale the data. The data is then passed through CNN layers to extract spatial features, followed by LSTM layers to capture temporal dependencies. The features from both stages are integrated into the dense layers, where the output layer produces the final classification. This methodology effectively combines spatial and temporal feature extraction to detect ransomware activity, using the power of CNNs and LSTMs to handle complex and sequential data patterns. By following this structured approach, the model ensures that it can identify ransomware with high accuracy, even in large and dynamic datasets.

The algorithm described in Alg. 1. outlines the specific steps and model architecture used in this study. The hybrid CNN-LSTM model integrates CNNs for spatial feature extraction and LSTMs for capturing temporal dependencies. This approach allows for the practical analysis of both static and dynamic patterns in ransomware behavior.

4.4 Results and discussion

The scatter plot in Fig. 5 provides a visualization of ransomware data designed to represent the numerical features of different ransomware types across five distinct classes. The x-axis represents the first numerical feature, corresponding to ransomware behavior attributes, such as file encryption rate and CPU usage. The y-axis depicts the second numerical feature, which indicates another aspect of ransomware behavior, such as memory consumption and network traffic anomalies. The dataset includes six distinct ransomware classes (Class 0 to Class 5), each representing different types or behaviors, such as varying attack strategies or encryption methods. These classes are visualized using unique colors, allowing for a clear distinction between instances from different classes and enabling the observation of how they overlap or cluster within the same feature space. Based on the current observation, significant overlap between data points of different classes is evident in the plot. This overlap indicates the complexity of ransomware behavior. Yet, the uniformly distributed data points imply that the dataset was normalized and generated within a consistent range (*e.g.*, 0 to 1), ensuring comparability across the features.

As observed in the scatter plot (Fig. 5), there is some visual overlap between different ransomware classes in the 2D projection. This plot was generated using the t-distributed Stochastic Neighbor Embedding (t-SNE) technique, a method commonly employed to visualize high-dimensional data by preserving local similarities. However, t-SNE may compress globally separable boundaries into overlapping regions in two-dimensional space. Despite this, the CNN-LSTM model achieved high accuracy in distinguishing between classes, as

Alg. 1: Algorithm: N-Body Simulation with Butterfly Algorithm.

Data Preparation:

Normalize feature matrix X:

$$\left(x_{ij=\frac{Xij-\min(X)}{\max(X)-\min(X)}}, \forall i,j\right) \quad (1)$$

Split XXX and YYY into training and testing sets using the ratio r:

$$((X_{train}, Y_{train}), (X_{test}, Y_{test}) = \text{Split(X, Y, r)}) \quad (2)$$

Reshape for LSTM Input:

Convert $X_{train\ and}\ X_{test}$ into sequences:

$$(Y_{train}) = \{x1, x2, \ldots..xN\}, x_i \in \mathrm{R}^{dx1}$$

Define Hybrid Deep Learning Architecture:

Combine CNN and LSTM layers for feature extraction and temporal analysis.

The architecture is:

Input Layer: d-dimensional sequence input.

CNN Layer: Convolution filters for spatial feature extraction

LSTM Layer: u units for sequence modeling (temporal features).

Fully connected, dropout, and SoftMax layers for classification.

The training objective is to minimize cross-entropy loss:

$$(\mathcal{L}(\theta) = -\sum_{i=1}^{N}\sum_{j=1}^{C} Yij \log \hat{y}\_ij) \quad (3)$$

where Y_{ij} is the predicted probability for class j.

Model Training:

Use Adam optimizer with parameters η(learning rate) and batch size b:

$$\frac{\left(\theta^{*}=\arg\min \mathcal{L}(M(X_{train,}\theta)\right)}{\theta} \quad (4)$$

Train the model for epochs with validation on $(X_{test}X_{test})$

Evaluation:

Predict test labels:

$$\hat{y}_{test} = M\ (X_{test,}\theta^{*}) \quad (5)$$

Compute accuracy and confusion matrix:

Plot the confusion matrix CM for detailed evaluation.

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} 1(\hat{y}_i = y_i) \quad (6)$$

confirmed by the confusion matrix and other evaluation metrics. This indicates that the model successfully learned complex latent patterns that are not immediately visible in the reduced 2D visualization.

5. Discussion

The growing complexity of ransomware attacks highlights the urgent need for advanced detection systems to identify novel and evolving threats. While helpful in detecting known malicious behaviors, traditional methods are increasingly inadequate in the aspect of new ransomware variants that bypass conventional defenses. This study, therefore, proposed a hybrid deep learning framework integrating CNNs and Long LSTMs. These two robust architectures complement each other in addressing the limitations of existing detection approaches. The results presented in Table 4 of this paper

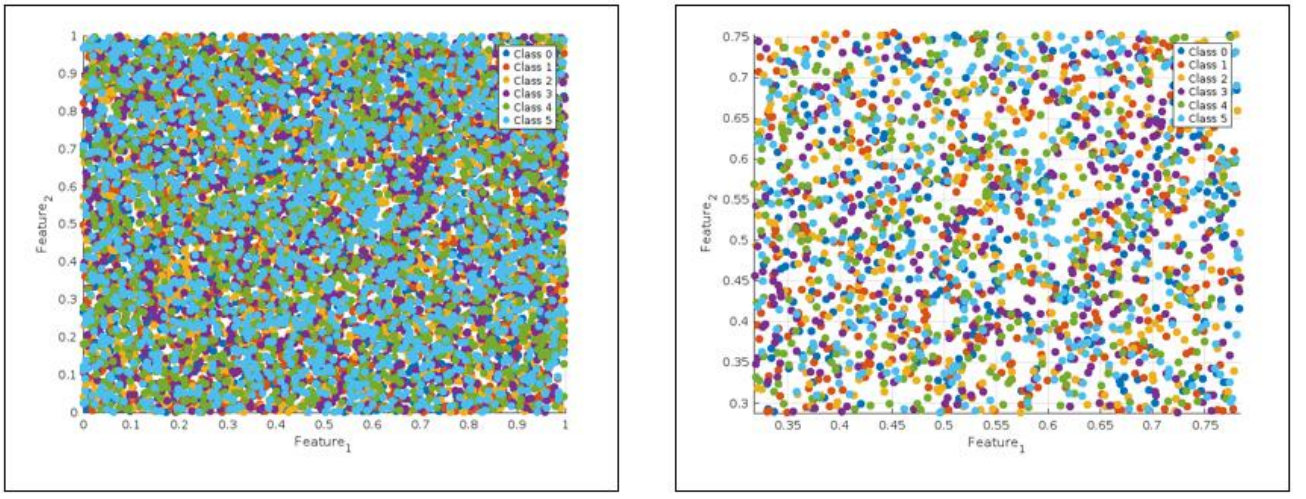

Fig. 5: Shows a scatter plot of ransomware data, visualizing key features across six classes, highlighting behavior overlaps and normalized distribution.

Table 4: Model Comparison Based on Key Indicators.

Classifier	Model Type	Tuned Hyperparameters	Feature Selection	Optimizer	Prediction Speed (obs/sec)	Train Accuracy (%)	Test Accuracy (%)	Precision (%)	Recall (%)	F1Score (%)	Training Time (s)	Model Complexity	Memory Usage (MB)
Hybrid (CNN + LSTM)	Deep Learning	CNN layers: 3-5 LSTM units: 50-150 Learning rate: 0.001 Batch size: 64 Epochs: 50	Feature algorithm: PCA Reduced dimensions: 1.0e+03	Adam optimizer Iterations: 50 Training time(s): 720	~50	95.8	97.5	96.2	97.0	96.6	720	High	3,000
Random Forest	Tree-based	Number of trees: 100-200 Max depth: None Min samples split: 2	Feature algorithm: MRMR Ranked features selected: 1.0e+03	Default Training time(s): 100	~1,200	86.0	89.5	88.5	87.0	87.7	100	Low	500
LSTM	Neural Network	LSTM units: 100 Learning rate: 0.001 Batch size: 32 Epochs: 40	Feature algorithm: MRMR Ranked features selected: 1.0e+03	Adam optimizer Training time(s): 450	~80	87.5	89.0	88.0	89.0	88.5	450	High	2,000
CNN	Neural Netwo	CNN layers: 3 Learning	Feature algorithm:	Adam optimizer	~100	85.0	88.2	86.5	87.2	86.8	400	Medium	1,000

Classifier	Model Type	Tuned Hyperparameters	Feature Selection	Optimizer	Prediction Speed (obs/sec)	Train Accuracy (%)	Test Accuracy (%)	Precision (%)	Recall (%)	F1Score (%)	Training Time (s)	Model Complexity	Memory Usage (MB)
	rk	rate: 0.001 Batch size: 32 Epochs: 40	ReliefF Selected features: 1.0e+03	Training time(s) : 400									
Gradient Boosting	Tree-based	Number of estimators: 100-200 Learning rate: 0.05 Max depth: 3	Feature algorithm: PCA Reduced dimensions: 1.0e+03	Adam optimizer Training time(s) : 250	~1,000	80.5	82.3	81.2	80.5	80.9	250	Medium	700

demonstrate the superior performance of the hybrid CNN-LSTM model compared to traditional machine learning methods and individual CNN and LSTM models. This approach capitalizes on the strengths of both architectures: CNNs for spatial feature extraction and LSTMs for sequential pattern recognition. The CNN component excels in extracting critical spatial features from raw ransomware data, such as file byte sequences, system calls, and network traffic, which are essential for characterizing the unique behavior of ransomware. The LSTM component then models the temporal dependencies in the data, capturing the dynamic and evolving nature of ransomware activities over time. This hybrid approach enables the model to recognize complex patterns that are difficult for traditional detection methods.

The performance evaluations presented in Figs. 6(a-d) provide strong evidence in support of the proposed approach. In Fig. 6(a), the training and testing processes of the models demonstrate consistent convergence, indicating that the

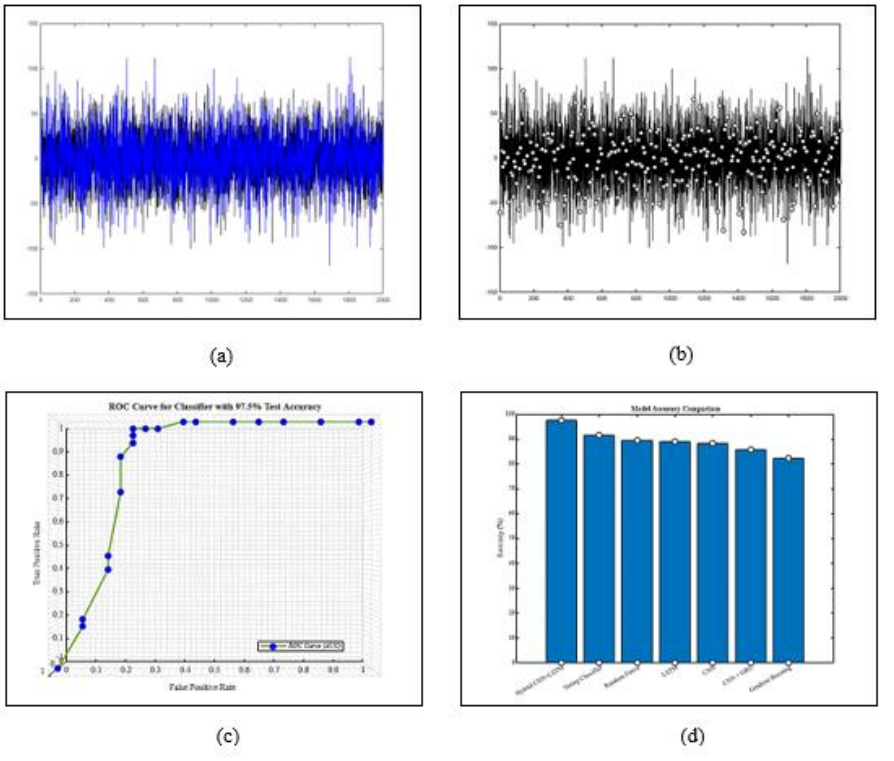

Fig. 6: Performance Evaluations. (a) Training and testing process of the models, (b) valuation of test performance, (c) ROC curve for the highest performing model, and (d) Comparative analysis of overall model accuracy across all models.

framework generalizes well without significant overfitting The evaluation of test performance in Fig. 6(b) further confirms this by demonstrating stable accuracy trends across different iterations. The ROC curve in Fig. 6(c) highlights the discriminative capability of the best-performing classifier, achieving an AUC of nearly 1.0 and a test accuracy of 97.5%,

which reflects high reliability in distinguishing between normal and malicious patterns. Finally, Fig. 6(d) compares the overall accuracy of multiple models, clearly demonstrating that the CNN–LSTM framework outperforms traditional classifiers and baseline methods. Together, these results validate the robustness, reliability, and superiority of the proposed hybrid architecture in addressing cybersecurity challenges.

5.1 Comparison with traditional and individual models

Compared to traditional machine learning models, the hybrid model outperformed classifiers such as Support Vector Machines (SVMs), Decision Trees, and Random Forests in terms of accuracy, precision, recall, and F1 score. These results underscore the advantages of deep learning models in capturing complex, high-dimensional features inherent in ransomware data. Moreover, individual CNN and LSTM models, though effective at feature extraction and sequence modeling, respectively, were less accurate than the hybrid model, demonstrating the importance of combining both approaches for enhanced performance. The hybrid CNN-LSTM model's superior performance has a significant impact on real-time ransomware detection in operational environments. As ransomware attacks become increasingly stealthy and sophisticated, the need for adaptive and robust cybersecurity solutions is more pressing than ever. The hybrid model's ability to detect new ransomware variants while reducing false positives is critical for minimizing the impact of attacks and enhancing the effectiveness of cybersecurity systems. Furthermore, the model's scalability makes it well-suited for deployment in large-scale systems, where the volume and diversity of data make detection challenging.

5.2 Limitations and future work

While the proposed hybrid model shows great promise, several limitations remain. The model's training time, particularly for the CNN component, can be resource-intensive, especially when working with large datasets. In this study, the model was trained on a system with an Intel Core i7 processor (3.2 GHz), 16 GB RAM, and an NVIDIA GTX 1660 Ti GPU. Training on the 10,000-sample synthetic dataset for 30 epochs took approximately 45 minutes. Based on these results, we estimate that training time would scale linearly with dataset size under similar conditions. Optimizations such as early stopping, mini-batch tuning, and pruning can be considered in future extensions to manage scalability more effectively.

Future work could focus on optimizing the training process by exploring more efficient architectures or incorporating transfer learning techniques to reduce the need for extensive training on large datasets. For example, transfer learning could leverage pre-trained models trained on broader behavioral or malware telemetry data and adapt them to ransomware-specific tasks. This has been successfully demonstrated in related domains (*e.g.*, Hardy *et al*., 2016; Saxe & Berlin, 2017), where fine-tuning a generalized model helps improve performance on specific malware types with minimal retraining. Applying such strategies in this context could accelerate development and enhance model robustness against novel ransomware strains. While the model performed well on the dataset used in this study, testing its generalization capabilities across different ransomware families and attack scenarios is essential. Expanding the dataset to include a broader range of attack types and incorporating real-time threat intelligence could enhance the model's robustness. Moreover, despite the high accuracy achieved by the hybrid model, there remains room for improvement in specific performance metrics. For instance, while the precision and recall values are promising, there may still be instances where the model struggles to detect more subtle, low-frequency ransomware behaviors. Refining the model to focus on rare and emerging strains, potentially by incorporating anomaly detection methods, could help address this limitation.

While the proposed hybrid model demonstrates high accuracy on the current dataset, its capability to detect previously unseen ransomware variants has not yet been explicitly validated. The dataset used for training and testing contained balanced samples across defined classes, but it did not simulate novel ransomware strains outside the training distribution. To strengthen this claim in future work, experiments can incorporate *leave-one-family-out* cross-validation, where one ransomware type is excluded during training and used only during testing. Additionally, integrating few-shot learning or meta-learning techniques could allow the model to generalize better with minimal samples of emerging variants. Exploring transfer learning from broader malware detection tasks may also improve the model's adaptability to new or evolving threats.

6. Conclusion

The ongoing battle against ransomware attacks necessitates continuous innovation in detection systems. This paper has introduced a hybrid deep learning approach that combines the spatial feature extraction capabilities of CNNs with the temporal pattern recognition strengths of LSTMs, resulting in a highly effective model for real-time ransomware detection. Our extensive experimental results demonstrate the model's ability to detect known ransomware strains and effectively adapt to novel and evolving threats, which is a critical advantage over traditional detection methods. The proposed hybrid model offers several key contributions to the field of cybersecurity. First, it leverages the power of deep learning to overcome the limitations of conventional signature-based methods, providing a more flexible and robust solution for detecting sophisticated ransomware variants. Second, by combining CNNs and LSTMs, the model addresses both spatial and temporal aspects of ransomware data, offering a holistic approach to threat detection. This makes the model both accurate and highly scalable, ensuring it can handle large volumes of real-time data —a key requirement for modern

cybersecurity systems. What sets this work apart is its potential to make a tangible impact in ransomware detection. The hybrid CNN-LSTM framework achieves superior performance metrics and reduces false positives, a significant challenge in cybersecurity. This enhances the accuracy of detection systems, allowing security teams to focus on genuine threats without being overwhelmed by false alerts. The ability to detect evolving ransomware behavior over time further enhances the model's practical utility, as it ensures long-term adaptability to emerging attack vectors. The model's ability to integrate with real-time threat intelligence systems presents an exciting opportunity for future research. To enable such integration, key steps include developing a lightweight deployment-ready version of the model (*e.g.*, containerized via Docker), designing streaming pipelines for real-time data ingestion, and incorporating adaptive thresholding and alerting mechanisms. Challenges in this direction include achieving low-latency inference, handling noisy or incomplete telemetry data, and ensuring model updates in response to evolving threat domains. By incorporating dynamic data streams and adapting to the latest ransomware trends, the model can continue to develop and improve, providing a future-proof solution to the ever-growing challenge of ransomware attacks.

Acknowledgments

We extend our gratitude to researchers whose innovative techniques and methodologies in malware detection have laid a strong foundation for advancements in this field. Their pioneering work has been instrumental in shaping the direction of our study and inspiring further exploration.

Conflict of Interest

There is no conflict of interest.

Supporting Information

Not applicable.

CRediT Statement

Priynka Sharma, Kaylash Chaudhary: Conceptualization, Validation, Writing – Review & editing, Project administration, Funding acquisition. **Priynka Sharma**: Methodology, Software, Formal analysis, Investigation, Data curation, Writing – Original draft, Visualization. **Kaylash Chaudhary**: Resources, Supervision.

References

[1] M. A. Alomari, M. N. Al-Andoli, M. Ghaleb, R. Thabit, G. Alkawsi, J. A. J. Alsayaydeh, A. S. A. Gaid, Security of smart grid: cybersecurity issues, potential cyberattacks, major incidents, and future directions, *Energies*, 2025, **18**, 141, doi: 10.3390/en18010141.

[2] F. Alzonem, G. Albrecht, D. Castellanos, M. Vandermeer, B. Stansfield, Ransomware detection using convolutional neural networks and isolation forests in network traffic patterns, 2024, doi: 10.21203/rs.3.rs-5278706/v1.

[3] B. Athiwaratkun, J. W. Stokes, Malware classification with LSTM and GRU language models and a character-level CNN, *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, USA, March 5-9, 2017, 2482-2486, doi: 10.1109/ICASSP.2017.7952603.

[4] M. Azeem, D. Khan, S. Iftikhar, S. Bawazeer, M. Alzahrani, Analyzing and comparing the effectiveness of malware detection: a study of machine learning approaches, *Heliyon*, 2024, **10**, e23574, doi: 10.1016/j.heliyon.2023.e23574.

[5] K. Bayoudh, A survey of multimodal hybrid deep learning for computer vision: Architectures, applications, trends, and challenges, *Information Fusion*, 2024, **105**, 102217, doi: 10.1016/j.inffus.2023.102217.

[6] N. Chitadze, Artificial intelligence, terrorism, and cyber security: challenges and opportunities, *Machine Intelligence Applications in Cyber-Risk Management*, IGI Global Scientific Publishing, 2025, 87-106, doi: 10.4018/979-8-3693-7540-2.ch005.

[7] A. Dahiya, S. Singh, G. Shrivastava, Android malware analysis and detection: a systematic review, *Expert Systems*, 2025, **42**, e13488, doi: 10.1111/exsy.13488.

[8] M. G. Gaber, M. Ahmed, H. Janicke, Malware detection with artificial intelligence: a systematic literature review, *ACM Computing Surveys*, 2024, **56**, 1-33, doi: 10.1145/3638552.

[9] N. Hicham, H. Nassera, S. Karim, Enhancing Arabic E-Commerce Review Sentiment Analysis Using a hybrid Deep Learning Model and FastText word embedding, *EAI Endorsed Transactions on Internet of Things*, 2023, **10**, doi: 10.4108/eetiot.4601

[10] J. Ispahany, M. R. Islam, M. Z. Islam, M. A. Khan, Ransomware detection using machine learning: a review, research limitations and future directions, *IEEE Access*, 2024, **12**, 68785-68813.

[11] V. Jain, A. Mitra, Real-time threat detection in cybersecurity: leveraging machine learning algorithms for enhanced anomaly detection, *Machine Intelligence Applications in Cyber-Risk Management*, IGI Global Scientific Publishing, 2025, 315-344, doi: 10.4018/979-8-3693-7540-2.ch014.

[12] Y. L. Khaleel, M. A. Habeeb, A. S. Albahri, T. Al-Quraishi, O. S. Albahri, A. H. Alamoodi, Network and cybersecurity applications of defense in adversarial attacks: a state-of-the-art using machine learning and deep learning methods, *Journal of Intelligent Systems*, 2024, **33**, 20240153, doi: 10.1515/jisys-2024-0153.

[13] S. Kumar, B. Janet, S. Neelakantan, IMCNN: Intelligent Malware Classification using Deep Convolution Neural Networks as Transfer learning and ensemble learning in honeypot enabled organizational network, *Computer Communications*, 2024, **216**, 16-33, doi: 10.1016/j.comcom.2023.12.036.

[14] M. Macas, C. Wu, W. Fuertes, Adversarial examples: a survey of attacks and defenses in deep learning-enabled

cybersecurity systems, *Expert Systems with Applications*, 2024, **238**, 122223, doi: 10.1016/j.eswa.2023.122223.
[15] M. S. Manavadaria, T. A. S. Srinivas, S. K. Ahamed, M. Amshavalli, A. B. Nadaf, V. Bhoopathy, Anomaly detection algorithms in cybersecurity, *Machine Intelligence Applications in Cyber-Risk Management*, IGI Global Scientific Publishing, 2025, 293-314, doi: 10.4018/979-8-3693-7540-2.ch013.
[16] A. Miranda-García, A. Z. Rego, I. Pastor-López, B. Sanz, A. Tellaeche, J. Gaviria, P. G. Bringas, Deep learning applications on cybersecurity: a practical approach, *Neurocomputing*, 2024, **563**, 126904, doi: 10.1016/j.neucom.2023.126904.
[17] R. Nand, P. Sharma, Iteration split with firefly algorithm and genetic algorithm to solve multidimensional knapsack problems, *IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, Melbourne, Australia, December 9-11, 2019, 1-7, doi: 10.1109/csde48274.2019.9162422.
[18] L. Nataraj, S. Karthikeyan, G. Jacob, B. S. Manjunath, Malware images: visualization and automatic classification, *Proceedings of the 8th International Symposium on Visualization for Cyber Security*, Pittsburgh, Pennsylvania, USA, 2011, 1-7, doi: 10.1145/2016904.2016908.
[19] A. Panaras, B. Silverstein, and S. Edwards, Automated cooperative clustering for proactive ransomware detection and mitigation using machine learning, *TechRxiv,* 2024, doi: 10.36227/techrxiv.172684422.25967523/v1.
[20] Z. T. Pritee, M. H. Anik, S. B. Alam, J. R. Jim, M. M. Kabir, M. F. Mridha, Machine learning and deep learning for user authentication and authorization in cybersecurity: a state-of-the-art review, *Computers & Security*, 2024, **140**, 103747, doi: 10.1016/j.cose.2024.103747.
[21] J. Rafapa & A. Konokix, Ransomware detection using aggregated random forest technique with recent variants, *Authorea,* Preprint, 2024, doi: 10.22541/au.172426891.14153527/v1.
[22] A. T Rosário, J. C. Dias, AI-driven consumer insights in business: a systematic review and bibliometric analysis of opportunities and challenges, *International Journal of Marketing, Communication and New Media*, 2024, (15), doi: 10.54663/2182-9306.2024.specialissuembp.6-35.
[23] N. Scaife, H. Carter, P. Traynor, K. R. B. Butler, CryptoLock (and drop it): stopping ransomware attacks on user data, *IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*, Nara, Japan, June 27-30, 2016, 303-312, doi: 10.1109/ICDCS.2016.46.
[24] C. D. Schmugar *et al.*, "Mitigation of ransomware," ed: Google Patents, 2024.
[25] D. Sgandurra, L. Muñoz-González, R. Mohsen, and E. C. Lupu, Automated dynamic analysis of ransomware: Benefits, limitations and use for detection, *arXiv*, 2016, doi: 10.48550/arXiv.1609.03020.
[26] P. Sharma, K. Chaudhary, An advanced comparative study of ransomware anomaly detection techniques through optimized hyperparameters, *Artificial Intelligence and Sustainable Computing*, Springer Nature Singapore, Singapore, 2024, 379-393, doi: 10.1007/978-981-97-0327-2_28.
[27] P. Sharma, K. Chaudhary, Bio-inspired negative selection tested for ransomware anomaly detection, *Computing and Machine Learning*, Springer Nature Singapore, Singapore, 2024, 237-250, doi: 10.1007/978-981-97-7571-2_19.
[28] P. Sharma, K. Chaudhary, M. G. M. Khan, M. Wagner, Ransomware noise identification and eviction through machine learning fundamental filters, *IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, Melbourne, Australia, December 9-11, 2019, 1-8., doi: 10.1109/csde48274.2019.9162376.
[29] P. Sharma, K. Chaudhary, M. Wagner, M. G. M. Khan, A comparative analysis of malware anomaly detection, *Advances in Computer, Communication and Computational Sciences*, Springer Singapore, Singapore, 2020, 35-44, doi: 10.1007/978-981-15-4409-5_3.
[30] P. Sharma, G. Cirrincione, R. R. Kumar, A. Mohammadi, M. Cirrincione, A comparative study for the detection of stator inter-turn faults in induction motors using shallow neural networks and non-neural based techniques, *IEEE International Conference on Advanced Systems and Emergent Technologies (IC_ASET)*, Hammamet, Tunisia, April 29 - May 1, 2023, 1-6, doi: 10.1109/IC_ASET58101.2023.10150774.
[31] P. Sharma, M. Cirrincione, A. Mohammadi, G. Cirrincione, R. R. Kumar, An overview of artificial intelligence-based techniques for PEMFC system diagnosis, *IEEE Access*, 2024, **12**, 165708-165735.
[32] P. Sharma, S. Prakash, K. Chaudhary, Analyzing cybersecurity patterns in the Pacific Region: trends and challenges for 2023, *IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, Nadi, Fiji, December 4-6, 2023, 1-7, doi: 10.1109/CSDE59766.2023.10487141.
[33] B. Tang, N. L. Bragazzi, Q. Li, S. Tang, Y. Xiao, J. Wu, An updated estimation of the risk of transmission of the novel coronavirus (2019-nCov), *Infectious Disease Modelling*, 2020, **5**, 248-255, doi: 10.1016/j.idm.2020.02.001.
[34] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, S. Venkatraman, Deep learning approach for intelligent intrusion detection system, *IEEE Access*, 2019, **7**, 41525-41550.
[35] W. Wang, Q. Li, H. Mu, Exploring malware complexities: a behavior and characteristic analysis approach for robust and accurate cybersecurity, *Cluster Computing*, 2024, **28**, 82, doi: 10.1007/s10586-024-04769-w.
[36] D. Zhang, Y. Song, Q. Xiang, Y. Wang, IMCMK-CNN: a lightweight convolutional neural network with Multi-scale Kernels for Image-based Malware Classification, *Alexandria Engineering Journal*, 2025, **111**, 203-220, doi: 10.1016/j.aej.2024.10.055.