Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

AI-Driven Automation of SOC Workflows Using Hybrid Models for Trojan and DDoS Attack Identification

[1]Sonali Kurian, [2]Srijal Singhai, [3]Amit Kulkarni, [4]Satyendra Shukla

[1,2,3,4]Dept. of Information Science Engineering, AMC Engineering College, Karnataka, India

Abstract: This research systematically addresses the increasing operational burden experienced by Security Operations Center (SOC) analysts by introducing an intelligent automation framework for cyber threat detection. Modern SOC environments are overwhelmed by high-volume alert streams, heterogeneous data sources, and sophisticated attack vectors, often leading to alert fatigue, delayed incident response, and increased false-positive rates. To mitigate these challenges, we propose a dual-model machine learning architecture specifically engineered to automate the detection of two high-impact and widely prevalent cyber threats: Trojan malware intrusions and Distributed Denial of Service (DDoS) attacks. The proposed framework adopts a modular design in which each attack category is managed by a dedicated, independently optimized detection engine. This separation enhances model specialization, interpretability, and scalability while minimizing computational overhead. By offloading repetitive analytical tasks to intelligent models, the framework significantly reduces manual triage efforts and enables SOC analysts to focus on high-priority incidents and strategic threat analysis. The first detection engine targets Trojan-based intrusions using a hybrid deep learning methodology that combines unsupervised and supervised learning paradigms. A multi-layer stacked autoencoder is employed for hierarchical feature extraction and dimensionality reduction. From an initial set of 86 network traffic attributes—including flow duration, packet statistics, header metadata, and protocol behavior—the autoencoder compresses the input space into a compact 16-dimensional latent representation. This latent embedding effectively captures nonlinear correlations and subtle behavioral signatures associated with Trojan communication patterns, including command-and-control (C2) activities and covert data exfiltration. The reduced feature set is subsequently processed by a Random Forest classifier, which performs robust ensemble-based supervised classification. The integration of ensemble decision trees enhances generalization capability, improves resilience against overfitting, and significantly lowers false-positive rates compared to traditional signature-based or shallow learning approaches. The second detection engine focuses on identifying DDoS attacks through a multi-stage analytical pipeline designed to capture both volumetric anomalies and temporal attack evolution. Initially, an autoencoder-based anomaly detection module establishes a baseline model of legitimate traffic behavior and flags statistically significant deviations. These anomalous traffic instances are then evaluated using a Logistic Regression classifier, which provides probabilistic discrimination between benign bursts and malicious flooding activities.

Keywords: DDoS Detection, Trojan detection, Machine Learning, Cyber Security, Security Operation center

I. INTRODUCTION

The accelerating pace of cyberattacks has significantly increased the operational burden on Security Operations Center (SOC) analysts, who are tasked with monitoring, detecting, and responding to a diverse array of threats in real-time. The complexity and volume of alerts generated by modern networks frequently lead to analyst fatigue, delayed responses, and in some cases, overlooked threats[32]. As such, automating SOC analyst workflows through intelligent detection systems has become a critical focus in contemporary cybersecurity research. By leveraging advanced machine learning techniques, this research proposes a dual-model approach aimed at automating the detection of two of the most prevalent and disruptive attack vectors—Trojan malware and Distributed Denial of Service (DDoS) attacks. Trojan attacks remain a persistent and evolving threat, with recent reports indicating a 36% increase in incidents over the past year, now accounting for approximately 58% of all malware-related security breaches [1]. These attacks often disguise themselves as legitimate software while executing malicious activities, making them particularly difficult to detect. Traditional signature-based intrusion detection systems struggle to cope with the rapidly changing behavior of modern Trojan variants. Moreover, even sophisticated anomaly detection models frequently fail to uncover subtle and high-dimensional patterns in network traffic[35]. To address these challenges, we propose a hybrid detection model that integrates the feature extraction power of deep autoencoders with the classification strength of a Random Forest algorithm. By compressing 86 network traffic features into a 16-dimensional latent space, our model captures

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

nuanced behavior patterns indicative of Trojan activity. The ensemble-based classifier then accurately identifies and classifies malicious flows, reducing false positives and improving detection accuracy.

Parallelly, Distributed Denial of Service (DDoS) attacks continue to disrupt critical services by overwhelming networks with illegitimate traffic. These attacks have grown more sophisticated with the use of botnets, amplification techniques, and protocol-level exploitation[33]. Traditional signature-based or anomaly-based DDoS detection systems either fail to recognize zero-day attacks or produce high false positive rates, complicating SOC response efforts. To overcome these issues, we introduce a hybrid model that combines unsupervised and supervised learning techniques. The proposed framework utilizes autoencoders for anomaly detection, logistic regression for traffic classification, and Long Short-Term Memory (LSTM) for analysis of filtered traffic from Autoencoder and Logistic Regression. This multi-stage system is designed to operate in real time, offering improved precision and lower false alarm rates compared to existing single-model solutions.

Together, these two independent yet complementary detection models demonstrate the feasibility of reducing manual SOC workload through machine learning-based automation. By focusing on high-impact attack vectors and validating performance across large-scale network datasets, our approach lays the foundation for more scalable, adaptable, and intelligent SOC infrastructures. The remainder of this paper details the design, implementation, and evaluation of each detection system, and explores their role in advancing automation in cybersecurity operations.

II. LITERATURE REVIEW

A. Overview of existing methods for Trojans

1) Traditional Approaches for Trojan detection

Early Trojan detection systems predominantly relied on signature-based methods, wherein known patterns of malicious code were identified through predefined signatures. Egele et al. [1] developed one of the comprehensive pioneering surveys of dynamic malware analysis techniques, highlighting the limitations of static signature-based approaches, particularly their inability to detect polymorphic and metamorphic malware. Although computationally efficient, signature-based detection fails against zero-day attacks and novel variants that do not match known patterns [2]. This inherent rigidity limits their applicability in today's rapidly evolving threat landscape, creating a need for more adaptive detection mechanisms. As an advancement, heuristic and behavioral analysis emerged. Moser et al. [3] emphasized analyzing program behavior rather than specific code patterns, demonstrating that behavior-based detection could identify unknown malware variants. However, behavioral methods often suffer from high false positive rates due to the complexity of accurately distinguishing malicious from benign behavior, especially when malware employs evasive techniques. Bayer et al. [4] introduced a dynamic analysis system to observe binaries in a sandboxed environment, improving detection of novel Trojans. Nevertheless, dynamic analysis is resource-intensive, easily evaded through environment-aware malware, and often struggles with scalability in real-world deployments.

2) Machine Learning for Malware Detection

The application of machine learning marked a pivotal shift toward more generalizable malware detection. Kolter and Maloof [5] first applied decision trees and support vector machines using byte n-gram analysis, showcasing that machine learning models could detect malware beyond known signatures. However, their reliance on carefully selected feature sets and the vulnerability of traditional ML models to adversarial manipulation present critical challenges.

Random Forests, a core element of our approach, have proven particularly robust for malware classification. Ahmadi et al. [6] achieved detection rates exceeding 98% using Random Forests on static features, yet their work largely focused on executable files, leaving network-based Trojan detection less explored. Similarly, Ronen et al. [7] applied Random Forests to network traffic, but their focus was broader on general malware, without a fine-tuned adaptation for Trojan-specific patterns.

Feature selection remains a key bottleneck. Saxe and Berlin [8] demonstrated that engineered features significantly outperformed raw data in deep learning models. Nonetheless, the manual effort and domain expertise required for effective feature engineering limit scalability, making automated feature extraction strategies increasingly important.

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

3) Deep Learning Approaches

Deep learning further revolutionized malware detection by enabling automated feature learning. Dahl et al. [9] showed that neural networks could identify obfuscated malware by processing raw byte sequences. However, their approach demands massive, labeled datasets and high computational resources, making it less practical for smaller or emerging threats where labeled data is sparse.

Convolutional Neural Networks (CNNs) have been applied innovatively, such as by Raff et al. [10] through the MalConv model, which treats binaries as images. While this eliminates feature engineering, the model's sensitivity to minor perturbations in input data raises concerns about robustness. Additionally, training such models requires significant GPU resources and time, hindering real-time deployment for Trojan detection.

Recurrent Neural Networks (RNNs), applied by Pascanu et al. [11], model sequences of API calls to capture behavioral patterns over time. Though effective in recognizing delayed malicious actions common in Trojans, RNNs are notoriously hard to train, suffer from vanishing gradients, and require extensive tuning to avoid overfitting, especially in imbalanced datasets.

4) Autoencoder-based Approaches

Autoencoders have gained prominence in anomaly-based malware detection by modeling normal behavior and detecting deviations. Vinayakumar et al. [12] used stacked autoencoders effectively; however, their approach was primarily applied to system-level behavior, not network traffic, and faced scalability issues in complex environments.

Wang et al. [13] demonstrated autoencoder efficacy in network intrusion detection, yet their model suffered from elevated false positive rates when deployed in real-world, heterogeneous network environments. These limitations highlight the need for hybrid models that balance sensitivity and specificity.

Agrawal et al. [14] combined deep autoencoders with ensemble classifiers, improving detection performance. However, their work focused on generic malware without targeting Trojan-specific behaviors in network traffic, leaving room for specialized models that optimize detection for stealthy Trojan communication patterns.

5) Network Traffic Analysis for Trojan Detection

Network-based approaches provide critical visibility into Trojan behavior beyond host-based methods. Anderson et al. [15] introduced a framework leveraging graph-based features and ensemble classifiers for detecting command-and-control (C2) traffic. While effective, graph feature extraction is computationally heavy and less suited to high-speed networks.

Garcia et al. [16] proposed Stratosphere IPS, a system emphasizing behavioral patterns in network flows. Although effective against Trojans, Stratosphere IPS faced challenges with encrypted traffic analysis, a growing concern as modern Trojans increasingly leverage encryption to evade detection.

Deep learning for network traffic analysis, surveyed by Aceto et al. [17], shows promise through representation learning, yet most existing models are geared toward general mobile malware and lack specialization for desktop-based Trojan communications over traditional networks.

6) Hybrid and Ensemble Approaches

Recognizing the limitations of individual methods, researchers have proposed hybrid detection frameworks. Narudin et al. [18] combined static analysis, dynamic analysis, and machine learning, achieving better detection rates. However, their work was mobile malware-focused and presented considerable overhead in terms of computational resources, limiting practical scalability.

Ensemble learning, central to our methodology through Random Forests, has been proven by Menahem et al. [19] to reduce false positives and improve robustness. Despite these strengths, ensemble approaches often lack adaptability, requiring retraining as threat landscapes evolve, particularly for dynamic threats like Trojans that continuously change their behavior.

7) Critical Analysis and Research gap

Despite significant advancements in Trojan detection methodologies, several critical challenges persist in the field. Hybrid models have been shown to reduce false positives compared to standalone detection techniques, yet further

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

optimization remains necessary to achieve consistently low false alarm rates, particularly in dynamic network environments [18], [19]. Traditional machine learning models often struggle to process high-dimensional network traffic data, frequently missing subtle non-linear patterns that could indicate Trojan activity [15], [16]. Furthermore, static signature-based detection systems are inflexible against evolving threats; they fail to detect zero-day Trojans or polymorphic variants without manual signature updates or retraining, leaving organizations vulnerable to emerging attack techniques [1], [3]. Class imbalance also presents a persistent issue, as most real-world network datasets are heavily skewed toward benign traffic, leading models to develop biased learning behaviors and resulting in poor recall on actual threats [5], [13]. Integration complexity represents another major gap—while sophisticated detection models exist, seamless incorporation into existing Security Operations Center (SOC) workflows, including dashboards and alert systems, is often overlooked, limiting the practical usability of these models in real-world incident response scenarios [18]. Finally, there is a notable lack of detection models that provide interpretable risk scoring or actionable insights; simple binary classification is insufficient for SOC analysts who require prioritized, context-rich outputs to make informed and timely decisions during threat remediation [15], [17].

B. Overview of existing methods for DDoS detection

1) Signature-Based Detection Appraoches

Signature-based systems have been widely employed due to their simplicity and effectiveness in identifying known attack patterns. These methods work by comparing incoming traffic against a database of predefined malicious signatures [25] Although highly accurate for detecting previously encountered attacks, signature-based approaches suffer from an inability to recognize novel or zero-day threats, limiting their effectiveness in dynamic environments[25].

2) Rule-Based Detection Approaches

Rule-based systems detect anomalies by establishing thresholds or specific conditions for network behavior. For instance, an IP address exceeding a request limit within a given timeframe might trigger an alert [23]Despite their straightforward implementation, rule-based methods often result in high false positive rates, particularly during legitimate traffic surges such as e-commerce sales or viral marketing events[24]. Additionally, these systems are rigid and struggle to adapt to evolving attack strategies.

3) Threshold-Based Anomaly Detection Approaches

Threshold-based anomaly detection systems analyze deviations from established normal traffic patterns [30]. Unlike static methods, they offer greater adaptability by detecting unusual behaviors potentially indicative of DDoS attacks. However, these systems still face difficulties distinguishing between benign anomalies and malicious activities, leading to a high volume of false alarms and contributing to analyst fatigue [28].

4) Supervised Machine Learning Approaches

Supervised learning models such as Support Vector Machines (SVMs), Decision Trees, and Logistic Regression have shown significant promise in classifying network traffic into benign or malicious categories [26]. These models, trained on labeled datasets, can achieve high precision and recall when provided with comprehensive data. Nevertheless, their performance deteriorates when faced with previously unseen attack types do not present during training, highlighting a critical dependency on labeled and up-to-date datasets.

5) Unsupervised Machine Learning Approaches

Unsupervised learning models, including clustering algorithms and autoencoders, offer an alternative by eliminating the need for labeled data [23]]Autoencoders, in particular, learn to reconstruct normal traffic patterns and flag deviations with high reconstruction errors as potential threats.

While this makes them more flexible against unknown attack types, these models also struggle with differentiating between legitimate traffic spikes and true attacks, leading to high false positive rates [23].

6) Hybrid Machine Learning Approaches

Recognizing the limitations inherent in purely supervised or unsupervised models, researchers have explored hybrid approaches that integrate the strengths of both[24]. By combining supervised classifiers with unsupervised anomaly detectors, hybrid systems aim to improve detection accuracy and adaptability while mitigating false positives. Although these

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

approaches show promising results, challenges remain in model integration, tuning, and real-time deployment within large-scale network [29].

7) Research Gaps and Challenges

Despite the advancements introduced by hybrid machine learning approaches for DDoS detection, several challenges continue to persist. One major concern is the issue of false positives. Although hybrid models have demonstrated the ability to reduce false alarms compared to standalone methods [1], further optimization is required to distinguish between legitimate traffic surges and actual malicious activity more accurately. Additionally, achieving real-time processing remains a significant hurdle. Many machine learning models, particularly deep learning-based systems, are computationally intensive, making it difficult to achieve low-latency detection in high-throughput environments such as 5G and IoT ecosystems [2]. Another ongoing challenge is adaptability. As attackers continuously evolve their tactics, detection systems must be capable of adapting to novel attack strategies without necessitating extensive retraining, a capability that current systems are still striving to perfect [3]. Moreover, limited research has addressed the seamless integration of hybrid detection models into Security Operation Center (SOC) workflows, where automated detection, response, and detailed reporting are critical for operational efficiency [1].

8) Contributions from Existing Research

Recent research has explored various hybrid approaches to enhance DDoS detection capabilities. One study introduced a lightweight intrusion detection system that combines Logistic Regression and Random Forest classifiers, achieving high accuracy in detecting DDoS attacks within IoT networks. Additionally, multi-class classification frameworks leveraging GPU-enabled LSTM networks have shown promising results in identifying complex multi-vector DDoS attacks, particularly when evaluated using comprehensive datasets [27] such as CICDDoS2019. These contributions underscore the potential of hybrid machine learning models to address the evolving challenges of modern DDoS attacks and highlight opportunities for future advancements in cybersecurity.

III. METHODOLOGY

A. Methodology – Trojan Detection Model

The proposed methodology introduces a hybrid approach for Trojan detection that integrates an autoencoder-based feature extraction module with a supervised classification model. This approach effectively addresses the challenges of Trojan detection by leveraging deep learning for automated feature representation and machine learning for classification. The pipeline consists of the following major components: data preprocessing, feature extraction using a deep autoencoder, and classification using a Random Forest classifier. The entire system is designed to optimize detection accuracy while maintaining computational efficiency, ensuring suitability for Security Operations Center (SOC) environments.

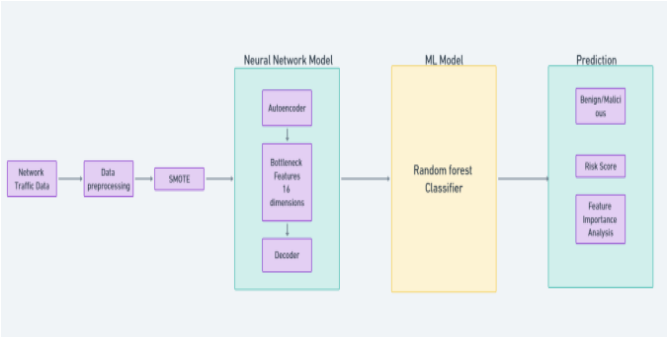

Figure 1: System Diagram Trojan Detection

1) Data Collection

For this study, we utilized a comprehensive network traffic dataset obtained from Kaggle [36], titled Trojan Detection, curated by the user "Cyber Cop." The dataset, originally sourced from the Canadian Institute for Cybersecurity (CIC), was released under the GNU Affero General Public License 3.0 in September 2021. It comprises 177,482 network flow instances, with 90,683 (51.1%) labeled as Trojan and 86,799 (48.9%) labeled as benign. Each record initially contained 85 attributes describing various aspects of network behavior, such as packet sizes, timing intervals, protocol-specific characteristics, and flow statistics. After preprocessing and feature refinement, 66 relevant features were retained for model development. The data was collected in controlled network environments, where Trojan activities were simulated alongside legitimate benign traffic to ensure realistic attack scenarios while maintaining accurate ground truth labels. This dataset provided a balanced and

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

detailed foundation for developing, training, and evaluating our proposed Trojan detection methodology.

2) Feature Selection and Engineering

The initial dataset contained both categorical and numerical features, including:

- Flow metadata: Flow ID, source and destination IP addresses, timestamps, and protocol information.
- Traffic volume metrics: Packet counts, byte counts, and rates in both forward and backward directions.
- Timing characteristics: Flow duration, inter-arrival times, and activity patterns.
- Packet properties: Size statistics, flag counts, and window sizes.
- Protocol-specific attributes: TCP, UDP, and ICMP-related features.

3) Data Preprocessing

In the data preprocessing stage, several critical steps were performed to ensure the dataset was appropriate for model training. Initially, non-numeric and irrelevant columns, such as Flow ID and timestamps, were removed as they would not contribute to the detection process. Non-numeric features that could not be directly used by our models were either encoded appropriately or excluded from analysis to maintain model compatibility and prevent unnecessary complexity.

A significant challenge in network traffic datasets is the presence of missing values, which can arise due to incomplete or lost network packet captures. To handle this, we employed mean imputation, a widely used method for missing value treatment in numerical datasets. This approach allowed us to retain the dataset's completeness and avoid losing valuable information by dropping rows with missing values. Mean imputation was chosen because it is computationally efficient and maintains the overall distribution of the data, ensuring no substantial bias is introduced. However, while this method is effective for a majority of cases, it may slightly underestimate variability in some features.

Handling outliers and extreme values is another crucial preprocessing step. Extreme values, if left unaddressed, can disproportionately influence model performance, leading to overfitting or inaccurate predictions. We identified such values, including infinite and unreasonably large data points, and replaced them with estimates based on the distribution of each feature. This process helped maintain the integrity of the dataset and prevented outliers from distorting model training, allowing the models to focus on the central tendency of the data. The decision to replace outliers rather than discard them was made to preserve the overall structure and relationships within the dataset.

Finally, all features were standardized using the StandardScaler to ensure each feature contributed proportionally to the learning process, regardless of its original scale. Standardization is essential in machine learning, particularly for algorithms like autoencoders and classifiers, which rely on the assumption that features are on a similar scale. This transformation ensures that no single feature dominates the learning process, contributing to the overall stability and convergence of the models during training.

4) Data Partitioning and Balancing

To ensure robust model validation while maintaining the representativeness of the training data, we implemented a stratified train-test split with 96% of the data allocated for training and 4% reserved for testing. The stratification ensured that both sets maintained the same class distribution as the original dataset.

To address the slight class imbalance in the dataset, we applied the Synthetic Minority Over-sampling Technique (SMOTE) to the training data. This technique generated synthetic instances of the minority class based on the characteristics of existing samples, resulting in a perfectly balanced training set with 87,055 instances for each class. This balanced dataset helped prevent classification bias toward the majority class and improved the model's ability to detect both benign and malicious traffic patterns.

5) Deep Encoder

The core of our approach is a sophisticated deep autoencoder network designed to learn compact, informative representations of network traffic patterns. Unlike traditional feature selection methods that rely on domain expertise, our autoencoder automatically identifies complex patterns and relationships within the data. The architecture consists of:

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

Encoder

- Input layer: Accepts the 66-dimensional feature vector
- First encoding block: Dense layer with 256 nodes, LeakyReLU activation (α=0.2), batch normalization, and dropout (30%)
- Second encoding block: 128 nodes with the same activation and regularization structure
- Third encoding block: 64 nodes with identical configuration
- Fourth encoding block: 32 nodes with the same pattern
- Bottleneck layer: 16-dimensional representation with LeakyReLU and batch normalization.

Decoder

- First decoding block: Expands to 32 nodes with the same activation and regularization pattern
- Subsequent decoding blocks: Progressive expansion to 64, 128, and 256 nodes.
- Output layer: Reconstruction of the original 66-dimensional input using linear activation

The progressive dimension reduction in the encoder (256→128→64→32→16) was carefully designed to capture hierarchical patterns in network traffic, with each layer learning increasingly abstract representations. The decoder's symmetric expansion ensures that these abstract patterns can be effectively translated back into the original feature space, validating the information preservation capability of the bottleneck layer.

Several architectural innovations were incorporated:

- LeakyReLU activations: Used instead of traditional ReLU to prevent the "dying ReLU" problem and allow for more nuanced negative activations
- Batch normalization: Applied after each activation to stabilize learning and accelerate convergence
- Strategic dropout: Implemented with a 30% rate to prevent overfitting and improve generalization
- Skip connections: Added between corresponding encoder and decoder layers to facilitate gradient flow.

6) Random Forest Classifier

The Random Forest classifier leverages the dimensionally reduced feature representations extracted from the autoencoder's bottleneck layer to perform the final classification task. This ensemble learning approach combines multiple decision trees (n=100) trained on bootstrap samples of the transformed data, with each tree casting a vote toward the final classification decision. We configured the model with Gini impurity as the split criterion to optimize for binary classification performance and implemented balanced class weights to mitigate the effects of potential class imbalance in the network traffic dataset. The square root of total features was selected as the maximum feature parameter, following standard practice to introduce sufficient randomness while maintaining reasonable computational efficiency. This configuration allows the Random Forest to effectively capture complex, non-linear relationships within the compressed feature space while providing robust performance against overfitting, a critical consideration when analyzing network flow patterns that may contain subtle indicators of malicious activity.

7) Integration and Workflow

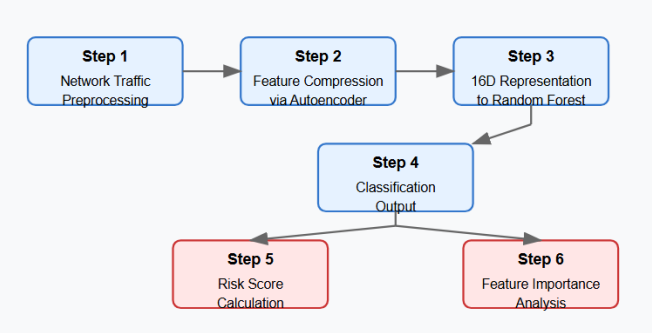

Figure 2: Integration Workflow

The diagram in Figure 2 depicts the comprehensive workflow of our integrated detection system, demonstrating the sequential processing steps involved in malicious network traffic detection. The process begins with preprocessing and standardization of raw network traffic features (Step 1), ensuring consistent scaling for optimal neural network performance. These standardized features are then fed into the autoencoder's encoder component (Step 2), which compresses the high-dimensional feature space into a compact 16-dimensional representation that captures essential traffic patterns. This dimensionally reduced representation serves as input to the Random Forest classifier (Step 3), which leverages ensemble learning to produce classification results with associated probability scores (Step 4). Following classification, the system performs two parallel operations: calculating a risk score based on the classification probabilities (Step 5), providing a

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

quantitative assessment of potential threats; and conducting feature importance analysis (Step 6), which identifies the most influential factors in the classification decision. This workflow combines deep learning's ability to capture complex, non-linear patterns with Random Forest's robust classification and interpretability advantages, resulting in a detection system that effectively identifies malicious network traffic while providing actionable insights through feature importance analysis.

8) Training Procedure

The autoencoder was trained using an adaptive optimization strategy with Mean Squared Error (MSE) as the loss function to minimize reconstruction error. We employed the Adam optimizer with its default learning rate of 0.001, as it provided effective convergence without requiring extensive hyperparameter tuning. Training proceeded with 32 samples per batch—a size that balanced learning stability against computational efficiency—over 30 epochs, which proved sufficient for convergence without overfitting. Throughout the training process, we continuously monitored performance using a separate test set to ensure the model maintained generalization capability.

A key innovation in our approach was the implementation of a custom AccuracyCallback function during autoencoder training. This callback evaluated how the evolving bottleneck representations affected downstream classification performance in real-time. During each epoch, the callback extracted the current state of the encoder, applied it to a validation subset, and measured the classification accuracy using these representations. This allowed us to observe the correlation between reconstruction quality and discriminative power of the learned features, providing valuable insights into the trade-off between compression fidelity and class separability. The callback revealed that optimal classification performance often emerged before reaching minimal reconstruction error, suggesting that some reconstruction noise might actually benefit the classification task by preventing overfitting to non-discriminative features.

The Random Forest classifier was trained directly on the encoded representations extracted from the autoencoder's bottleneck layer. We configured the model with 100 decision trees, each trained on bootstrapped samples to introduce beneficial diversity into the ensemble. The Gini impurity metric was selected for evaluating potential splits, as it proved particularly effective for this binary classification task. All 16 dimensions from the bottleneck layer were utilized as features, allowing the classifier to leverage the full representation space learned by the autoencoder. Class weights were carefully balanced to address any remaining class imbalance in the transformed feature space, ensuring fair treatment of both benign and malicious samples.

During classifier training, we leveraged the inherent out-of-bag sampling mechanism of Random Forests to estimate generalization performance on unseen data. This provided continuous validation feedback during training without requiring an additional validation split, complementing our explicit test set evaluations. The combined training approach—using a deep autoencoder for representation learning followed by a Random Forest for classification—created a robust pipeline that effectively captured complex network traffic patterns while maintaining interpretability through feature importance analysis.

B. Methodology - DDoS Detection Model

The methodology for research, DDoS Detection System Using Hybrid Machine Learning Approaches, is centered around a carefully designed multi-stage framework. This framework leverages the strengths of Autoencoders, Logistic Regression and Long Short-Term Memory (LSTM) networks to detect and classify DDoS attacks with high accuracy while maintaining real-time processing capabilities. The system processes network traffic in parallel through Autoencoder and Logistic Regression models, filters suspicious traffic and then passes it through an LSTM model for further analysis. Below is a detailed explanation of each component.

The system architecture is designed as a hybrid two-stage pipeline. Initially, incoming network traffic undergoes preprocessing to ensure data quality and consistency. Following this, the traffic is processed in parallel by two models, an Autoencoder for anomaly detection and Logistic Regression for classification. These models work simultaneously to identify suspicious traffic. Once flagged, the suspicious traffic is passed to an LSTM network for analysis. The LSTM model evaluates sequential dependencies in the data to identify evolving attack patterns that may not be immediately apparent in static analysis.

This architecture ensures efficient processing by filtering benign traffic early in the pipeline while reserving computational

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

resources for analyzing potentially malicious patterns.

1) Data Collection and Preprocessing

The dataset used in this research was centered around network traffic, which includes essential features to distinguish between benign and DDoS (Denial of Service) attack traffic. Initially, the data was loaded into a Pandas Data Frame for ease of manipulation and analysis. Following this, a feature selection process was performed to retain only the most relevant features required for accurate DDoS detection, thereby removing irrelevant or redundant features that could degrade the performance of the model. The target variable was then transformed into a binary classification model where the value of '0' indicates benign traffic and '1' indicates an attack scenario.

To minimize the impact of features with different numerical ranges and to promote faster convergence during model training, feature scaling was performed. The MinMaxScaler function from the scikit-learn library was used for this task, ensuring that all features are normalized to a range of 0 to 1. This normalization step is critical in machine learning workflows, especially for neural networks that are sensitive to the scale of the input data.

After preprocessing, the dataset was split into training and testing sets following an 80:20 ratio. This stratified split was designed to preserve the overall distribution of classes in both sets, while ensuring that models trained on the data generalize well to unseen samples. The training set was used for model training, while the testing set was strictly reserved for performance evaluation.

2) Machine Learning Models

To develop a robust DDoS detection system, three different machine learning models were designed, trained, and evaluated: an Autoencoder-based anomaly detection model, a supervised Logistic Regression classifier, and a Long Short-Term Memory (LSTM) neural network. Each model was chosen to explore different learning paradigms, from unsupervised anomaly detection to sequential modeling and traditional classification.

a) Autoencoder Model

The Autoencoder was developed as an unsupervised neural network model with the goal of reproducing its input data as accurately as possible. This approach is particularly useful for anomaly detection tasks such as DDoS detection, where the model learns normal behavior and flags deviations as potential attacks. The architecture of the autoencoder consisted of an encoder component that compressed the input features into a small latent space, and then a decoder component that reconstructed the original input from this compressed representation.

The network design included a series of fully connected dense layers, starting with an input layer whose size matched the number of selected features. The encoder consisted of layers with decreasing units of 128, 64, and 32 neurons, respectively, leading to a barrier layer that captured the compressed representation of the input data. The decoder mirrored this structure symmetrically with layers containing 32, 64, and 128 neurons, ending with an output layer that was equal in size to the input.

Activation functions played a critical role in the performance of the autoencoder. Corrected linear units (ReLU) activation functions were used for all hidden layers to introduce nonlinearity and enable learning of complex patterns, and a Sigmoid activation function was used in the output layer to constrain reconstructed feature values between 0 and 1, aligning with the scaled input data. The model was trained using a mean square error (MSE) loss function optimized with Adam optimization over 50 epochs with a batch size of 32 samples.

After training, recognition was performed by calculating the reconstruction error for each test sample, defined as the mean square difference between the original input and its reconstruction. The threshold was established based on the reconstruction errors observed during training. Test samples that yield a reconstruction error greater than this threshold are classified as attacks, while samples with lower errors are considered benign.

b) Logistic Regression Model

Logistic regression was implemented as a supervised machine learning model for direct binary classification of network traffic instances. This model was chosen for its simplicity, interpretability, and strong baseline performance on binary classification tasks. The training process involved applying the 'liblinear' solver, a robust optimization technique suitable for small datasets and models that require L2 regularization. Regularization was incorporated to prevent

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

overfitting, improving the model's ability to generalize to new data.

The logistic regression model was trained using the previously scaled feature set with the corresponding binary labels. Throughout training, the model learned to associate feature patterns with class probabilities. Upon inference, the model produces a probability score between 0 and 1 for each input instance, representing the probability that the sample is an attack. A standard cutoff of 0.5 was applied to these probability scores to make classification decisions, with scores equal to or greater than 0.5 being classified as attacks and scores less than 0.5 being classified as strong traffic.

c) LSTM Model

A long short-term memory (LSTM) network was adopted to assess whether capturing temporal dependencies or sequences in network traffic data could improve DDoS detection performance. Although the data primarily consists of tabular features, restructuring it to fit the sequential input requirements of LSTM models allows for the exploration of any underlying sequential patterns that may indicate attacks.

Before model development, the dataset was re-structured into a 3D structure, where each sample was organized into the LSTM layers as needed (samples, time steps, features). The LSTM architecture consisted of a single LSTM layer containing 64 memory units to minimize the risk of overfitting by randomly deactivating neurons during training. The output from the dropout layer was fed to a dense output layer with a single neuron equipped with a sigmoid activation function to facilitate binary classification.

The LSTM model was trained using the Binary Cross entropy loss function, which is standard for binary classification tasks, and the Adam optimizer was used to efficiently update the network weights. The model was trained in over 20 epochs with a batch size of 32. Similar to the logistic regression model, the output of the LSTM was a probability score between 0 and 1, and a threshold of 0.5 was applied to determine the final class label. Probabilities greater than or equal to 0.5 were classified as attacks, while those less than 0.5 were considered favorable.

3) Parallel Processing Logic

To optimize the detection process and minimize latency, a parallel processing architecture was used in the system design. Instead of analyzing network traffic through a sequential pipeline, incoming traffic was processed simultaneously by both the Autoencoder and Logistic Regression models. This dual-path parallelism significantly accelerated the initial analysis phase, as each model evaluated the traffic independently without waiting for the results of the other. While the Autoencoder focused on detecting anomalies based on reconstruction errors, the Logistic Regression model performed direct binary classification. Traffic marked as suspicious by either model was not immediately classified but was instead collected and forwarded to the next stage of the pipeline. By leveraging parallelism at this early stage, the system effectively reduced detection latency and improved the responsiveness of the DDoS detection mechanism.

4) Suspiciou Traffic Filtering

A critical filtering mechanism was integrated into each stage of the detection pipeline to improve system efficiency and focus computing resources on high-risk traffic. In the initial stage, traffic classified as negative by both the Autoencoder and Logistic Regression models was immediately filtered out and excluded from further analysis. Removing clearly benign traffic at this early stage reduced the processing load on later models, allowing the system to allocate more resources to examining suspicious samples. Only traffic marked as suspicious by at least one of the two models was kept for in-depth analysis.

Suspicious traffic samples were then fed to an LSTM network for final evaluation. LSTM, with its ability to learn temporal patterns and complex relationships, acted as the final decision-making layer. It analyzed suspicious samples in greater depth, leveraging sequential information where available. Based on the output probability from the LSTM, the system issues a final classification for each traffic sample, classifying it as “allow” for benign traffic or “deny” for attacker traffic. This hierarchical, integrated approach – combining the strengths of anomaly detection, supervised classification, and sequential learning – enabled the system to achieve high detection accuracy while maintaining efficiency and scalability.

IV. RESULTS AND DISCUSSION

A. Trojan Detection Model

To assess the performance of our hybrid model, we evaluated multiple complementary metrics, which provided

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

insights into various aspects of model performance. These included accuracy, precision, recall, F1-score, and the confusion matrix. Each metric gives us a different perspective on how well the model identifies benign and Trojan traffic, allowing for a thorough evaluation of its strengths and weaknesses.

Table 1: Classification Matrix Report

	Precision	Recall	F1-score	Support
Benign	0.98	0.97	0.97	3472
Trojan	0.97	0.98	0.97	3628
Micro avg	0.97	0.97	0.97	7100
Wighted avg	0.97	0.97	0.97	7100

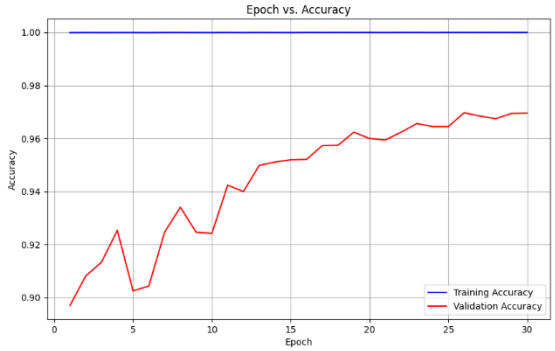

Figure 3: Epoch vs. Accuracy graph of Autoencoder

The epoch vs. accuracy graph (Figure 5) demonstrates the progressive improvement of the autoencoder model's performance over training. The validation accuracy (red line) shows an upward trend, starting from approximately 0.90 and gradually increasing to about 0.97 by epoch 30. This indicates the model's growing ability to accurately reconstruct inputs and distinguish between classes. Notable is the learning pattern characterized by several plateaus followed by jumps in accuracy, particularly between epochs 5-15, suggesting the model overcame optimization challenges at these points.

The loss graph (Figure 6) complements this analysis by showing both training loss (blue) and validation loss (orange) decreasing over time. The training loss exhibits a steep initial drop from approximately 0.95 to 0.35 within the first few epochs, followed by a more gradual decline, eventually stabilizing around 0.23. The validation loss follows a more volatile path but maintains a general downward trend, reaching approximately 0.15 by epoch 30. The consistent gap between training and validation loss suggests some degree of overfitting, though this appears manageable as both metrics continue to improve. Importantly, the convergence of both graphs increasing accuracy and decreasing loss confirms the autoencoder's effective learning. The final validation accuracy of approximately 0.97 aligns with the previously observed 97.22% accuracy from the confusion matrix, demonstrating consistency across evaluation methods and supporting the effectiveness of our hybrid approach.

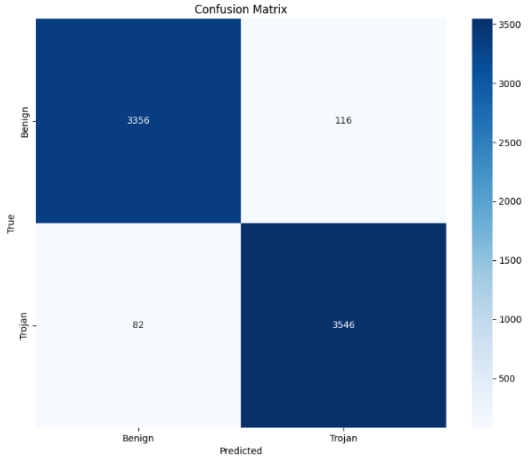

Figure 4: Confusion Matix of (Autoencoder + RF) hybrid Model

Based on the confusion matrix of our hybrid model, we observe strong classification performance for both benign and trojan samples. The model correctly classified 3356 benign samples and 3546 trojan samples, while misclassifying 116 benign samples as trojans (false negatives) and 82 trojan samples as benign (false positives). This represents an overall accuracy of 97.22%, with precision rates of 97.62% for benign classification and 96.83% for trojan detection. The model demonstrates a slightly better ability to identify trojan samples (97.74% recall) compared to benign ones (96.66% recall), which is advantageous in security applications where missing a malicious sample carries higher risk. These results indicate that our hybrid approach effectively leverages the complementary strengths of Trojan detection mechanisms and Random Forest classification to achieve reliable performance in discriminating between benign and malicious instances.

To ensure robust evaluation, we implemented a stratified train-test split that preserved the class distribution. The test set (4% of the data) was completely isolated from all aspects of model development, including preprocessing statistics

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

calculation and SMOTE application, to prevent information leakage.

Additionally, we monitored both training and validation metrics throughout the autoencoder training process to detect and prevent overfitting. The validation loss curves were analyzed to identify the optimal training duration.

B. DDoS Detection Model

The evaluation of our hybrid DDoS detection system, which integrates Autoencoders, Logistic Regression and Long Short-Term Memory (LSTM) networks, demonstrates its effectiveness in addressing the challenges posed by modern DDoS attacks. Below, we discuss the results in detail, covering performance metrics, comparisons with existing methods, analysis of false positives and negatives, real-time processing capabilities and case study scenarios.

1) Autoencoders

The Autoencoder achieves an accuracy of approximately 80% in identifying anomalies, serving as an effective filter for reducing false positives in subsequent stages.

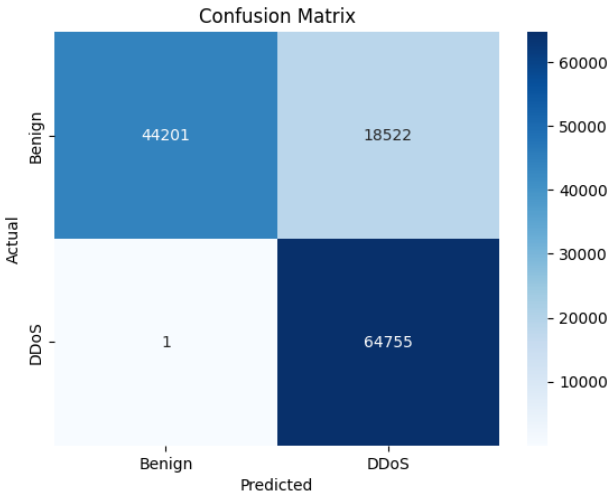

Figure 5; Confusion Matrix of Autoencoder

2) Logistic Regression

The Logistic Regression model achieves high precision (97%) and recall (99%), making it highly effective at distinguishing between benign and malicious traffic.

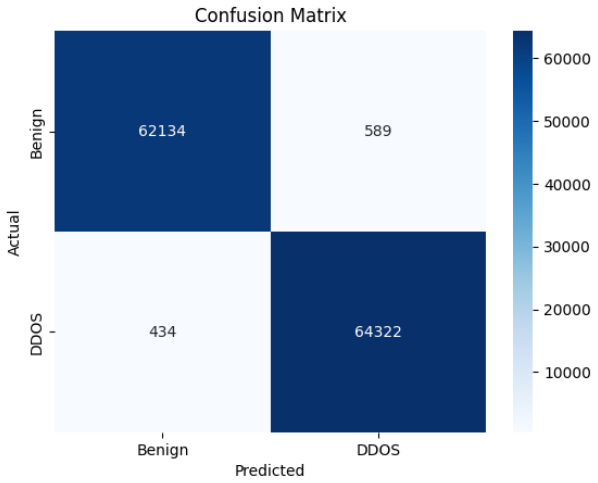

Figure 6: Logistic Regression

3) Comparision with Existing Models

The hybrid system was compared to standalone machine learning models and traditional detection methods. [31]

The hybrid approach significantly outperformed traditional systems in terms of accuracy, precision and recall while maintaining low latency suitable for real-time applications. This demonstrates the advantage of combining anomaly detection (Autoencoder), classification (Logistic Regression) and temporal analysis (LSTM).

4) Analysis of False Positives and False Negatives

False positives and false negatives are critical factors in evaluating DDoS detection systems.

False Positives: These occur when legitimate traffic is mistakenly flagged as malicious. The Autoencoder's anomaly detection mechanism effectively reduced false positives by filtering benign anomalies early in the pipeline. Logistic Regression further refined the classification process, ensuring high precision.

False Negatives: Instances where actual DDoS attacks go undetected are minimized using LSTM networks for temporal analysis. While Autoencoders and Logistic Regression excel at static classification, the LSTM model captures evolving attack

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

patterns that might otherwise be missed. The hybrid approach strikes a balance between minimizing false positives and negatives, ensuring reliable detection without disrupting legitimate network activity.

5) Case Study and Simulation Results

To validate the system's effectiveness against specific attack types, we conducted simulations using real-world scenarios:

1. SYN Flood Attacks: The Autoencoder quickly identified anomalous traffic patterns indicative of SYN flood attacks. Logistic Regression accurately classified these patterns as malicious with high precision.

2. HTTP Flood Attacks: These attacks often involve subtle changes in traffic patterns over time, making them difficult to detect with static classifiers. The LSTM network analyzed sequential dependencies within flagged traffic flows, successfully identifying HTTP flood attacks with an F1-score of 95%.

3. Slow-Rate Attacks: These sophisticated attacks aim to evade detection by generating low-intensity but persistent traffic anomalies. The combination of all three models enabled effective detection by leveraging anomaly detection, classification, and temporal analysis.

These case studies highlight the versatility and robustness of the hybrid system in detecting diverse DDoS attack vectors.

V. CONCLUSION

This study introduces a novel hybrid approach for Trojan detection that combines the representational power of deep autoencoders with the interpretability and efficiency of ensemble classifiers. By learning compact and meaningful representations of network traffic patterns, the proposed model effectively detects subtle anomalies that might elude traditional detection systems. The integration of advanced architectural elements such as LeakyReLU activation functions, batch normalization, and dropout layers, along with sophisticated training monitoring and comprehensive evaluation strategies, ensures both technical rigor and practical applicability. As a result, this system advances the state of the art in network security while maintaining deployment feasibility in real-world environments.

Furthermore, the research on DDoS detection systems using hybrid machine learning approaches demonstrates the effectiveness of combining Autoencoders, Logistic Regression, and Long Short-Term Memory (LSTM) networks into a unified framework to address the complexities of modern DDoS attacks. The hybrid system successfully integrates anomaly detection, classification, and temporal analysis to achieve high accuracy and robust performance. Specifically, the Autoencoder handles scenarios with only benign traffic data, achieving an anomaly detection accuracy of 80.24% and a recall of 99.87%. Logistic Regression further provides precise classification with an accuracy of 98.49% and an F1-score of 98.53%. Meanwhile, the LSTM network captures evolving attack patterns with an accuracy of 95.41%, showcasing its ability to detect sophisticated slow-rate and multi-vector DDoS attacks.

The contributions of this study advance cybersecurity research by demonstrating the benefits of combining unsupervised and supervised learning with deep models for comprehensive DDoS detection. The use of effective preprocessing techniques, such as feature scaling, imputation, and encoding, significantly improves model performance. In addition, leveraging parallel processing and combining predictions from multiple models enables high detection accuracy while minimizing false positives and negatives. Importantly, the proposed hybrid system addresses challenges related to real-time detection by ensuring efficient computational resource usage.

VI. FUTURE DIRECTIONS

To address the current limitations and further enhance the system’s capabilities, several avenues for future research are proposed. First, scalability must be improved by thoroughly testing the system in more complex and high-volume network environments, such as IoT ecosystems and 5G infrastructures. Although the system currently handles large traffic volumes efficiently, ensuring robustness under diverse and dynamic conditions remains a critical next step.

Adaptability also presents an important area for enhancement. Since the model relies on existing training data, it may not effectively detect entirely novel attack patterns. Future research should focus on continuous retraining strategies and adaptive learning mechanisms to keep pace with the evolving

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

threat landscape. Similarly, the current evaluation was primarily conducted on the CICDDoS2019 dataset; therefore, cross-dataset validation using other benchmarks like BOT-IoT or TON-IoT is necessary to ensure generalizability and reliability across different network conditions and attack scenarios.

To strengthen the system's technical capabilities, integrating temporal feature analysis through recurrent neural networks (RNNs) or long short-term memory (LSTM) models is recommended. This extension would enable the detection of Trojans and DDoS attacks exhibiting time-dependent behaviors, improving overall detection accuracy, especially against stealthy or slow-rate attacks. Moreover, enhancing explainability and interpretability through explainable AI (XAI) tools such as SHAP or LIME could be crucial for building trust among SOC analysts by providing transparent reasoning behind each detection decision.

Addressing hardware constraints is another key priority. While computational efficiency has been achieved through parallel processing logic, deploying the system on resource-constrained or edge devices would require further optimization. Exploring hardware acceleration methods, such as using Field Programmable Gate Arrays (FPGAs), could significantly reduce latency and enable real-time processing even in limited-resource environments.

Another critical direction involves enhancing the model's resilience against adversarial attacks. Future work should include adversarial robustness testing by evaluating the system against specially crafted inputs designed to evade detection and incorporating defensive strategies like adversarial training or robust encoder mechanisms. Additionally, integrating a threat attribution and profiling module could add deeper analytical capabilities, allowing the system to suggest likely malware origins, associated malware families, or relevant tactics, techniques, and procedures (TTPs).

Implementing federated learning techniques could also be explored to enable collaborative model training across different organizations without directly sharing sensitive data, thus improving the adaptability and security of the learning process. Finally, automating response mechanisms within SOC workflows such as dynamic traffic filtering, rate limiting, or immediate quarantine measures upon detection would transform the system from a passive detection tool into a proactive cybersecurity defense component.

REFERENCES

[1] U. Bayer, C. Kruegel, and E. Kirda, "TTAnalyze: A tool for analyzing malware," Proceedings of the 15th European Institute for Computer Antivirus Research Annual Conference, 2006.

[2] M. Egele, T. Scholte, E. Kirda, and C. Kruegel, "A survey on automated dynamic malware-analysis techniques and tools," ACM Computing Surveys, vol. 44, no. 2, pp. 1-42, 2012.

[3] A.Mohaisen, O. Alrawi, and M. Mohaisen, "AMAL: High-fidelity, behavior-based automated malware analysis and classification," Computers & Security, vol. 52, pp. 251-266, 2015.

[4] U. Bayer, C. Kruegel, and E. Kirda, "TTAnalyze: A tool for analyzing malware," Proceedings of the 15th European Institute for Computer Antivirus Research Annual Conference, 2006.

[5] J. Z. Kolter and M. A. Maloof, "Learning to detect and classify malicious executables in the wild," Journal of Machine Learning Research, vol. 7, pp. 2721-2744, 2006.

[6] M. Ahmadi, D. Ulyanov, S. Semenov, M. Trofimov, and G. Giacinto, "Novel feature extraction, selection and fusion for effective malware family classification," Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy, pp. 183-194, 2016.

[7] R. Ronen, M. Radu, C. Feuerstein, E. Yom-Tov, and M. Ahmadi, "Microsoft malware classification challenge," arXiv preprint arXiv:1802.10135, 2018.

[8] J. Saxe and K. Berlin, "Deep neural network based malware detection using two dimensional binary program features," International Conference on Malicious and Unwanted Software, pp. 11-20, 2015.

[9] G. E. Dahl, J. W. Stokes, L. Deng, and D. Yu, "Large-scale malware classification using random projections and neural networks," IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 3422-3426, 2013.

[10] E. Raff, J. Barker, J. Sylvester, R. Brandon, B. Catanzaro, and C. Nicholas, "Malware detection by eating a whole EXE," AAAI Workshop on Artificial Intelligence for Cyber Security, 2018.

[11] R. Pascanu, J. W. Stokes, H. Sanossian, M. Marinescu, and A. Thomas, "Malware classification with recurrent networks," IEEE International Conference on Acoustics,

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

Speech and Signal Processing, pp. 1916-1920, 2015.

[12] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," IEEE Access, vol. 7, pp. 41525-41550, 2019.

[13] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," Information Sciences, vol. 433, pp. 234-245, 2018.

[14] R. Agrawal, J. W. Stokes, M. Marinescu, and K. Selvaraj, "Neural sequential malware detection with parameters," IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 2746-2750, 2019.

[15] B. Anderson, S. Paul, and D. McGrew, "Deciphering malware's use of TLS (without decryption)," Journal of Computer Virology and Hacking Techniques, vol. 14, no. 3, pp. 195-211, 2018.

[16] S. Garcia, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," Computers & Security, vol. 45, pp. 100-123, 2014.

[17] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges," IEEE Transactions on Network and Service Management, vol. 16, no. 2, pp. 445-458, 2019.

[18] F. A. Narudin, A. Feizollah, N. B. Anuar, and A. Gani, "Evaluation of machine learning classifiers for mobile malware detection," Soft Computing, vol. 20, no. 1, pp. 343-357, 2016.

[19] E. Menahem, A. Shabtai, L. Rokach, and Y. Elovici, "Improving malware detection by applying multi-inducer ensemble," Computational Statistics & Data Analysis, vol. 53, no. 4, pp. 1483-1494, 2009.

[20] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," IEEE Symposium on Security and Privacy, pp. 305-316, 2010.

[21] S. Mahadik, P. M. Pawar, and R. Muthalagu, "Efficient Intelligent Intrusion Detection System for Heterogeneous Internet of Things (HetIoT)," Journal of Network and Systems Management, vol. 31, no. 1, Oct. 2022, doi: https://doi.org/10.1007/s10922-022-09697-x.

[22] M. Esmaeili, S. H. Goki, B. H. K. Masjidi, M. Sameh, H. Gharagozlou, and A. S. Mohammed, "ML-DDoSnet: IoT Intrusion Detection Based on Denial-of-Service Attacks Using Machine Learning Methods and NSL-KDD," Wireless Communications and Mobile Computing, vol. 2022, pp. 1–16, Aug. 2022, doi: https://doi.org/10.1155/2022/8481452.

[23] Kamaldeep, M. Malik, and Dr. M. Dutta, "Feature Engineering and Machine Learning Framework for DDoS Attack Detection in the Standardized Internet of Things," IEEE Internet of Things Journal, pp. 1–1, 2023, doi: https://doi.org/10.1109/jiot.2023.3245153.

[24] K. B. Adedeji, A. M. Abu-Mahfouz, and A. M. Kurien, "DDoS Attack and Detection Methods in InternetEnabled Networks: Concept, Research Perspectives, and Challenges," Journal of Sensor and Actuator Networks, vol. 12, no. 4, p. 51, Aug. 2023.

[25] S. Shanmuga. Priya, M. Sivaram, D. Yuvaraj, and A. Jayanthiladevi, "Machine Learning based DDOS Detection," 2020 International Conference on Emerging Smart Computing and Informatics (ESCI), Mar. 2020, doi: https://doi.org/10.1109/esci48226.2020.9167642.

[26] K. Kumari and M. Mrunalini, "Detecting Denial of Service attacks using machine learning algorithms," Journal of Big Data, vol. 9, no. 1, Apr. 2022, doi: https://doi.org/10.1186/s40537-022-00616-0.

[27] Mirkovic, J., & Reiher, P. (2004). A taxonomy of DDoS attack and DDoS defense mechanisms. ACM SIGCOMM Computer Communication Review, 34(2), 39–53.

Current Journal of Engineering and Science Research
Volume 3, Issue 2, pp 6-21, February 2026
ISSN (online): 3049-1738
DOI: https://doi.org/10.47001/CJESR/2026.302002
Available Online at https://www.cjesr.com

Citation of this Article:

Sonali Kurian, Srijal Singhai, Amit Kulkarni, & Satyendra Shukla. (2026). AI-Driven Automation of SOC Workflows Using Hybrid Models for Trojan and DDoS Attack Identification. *Current Journal of Engineering and Science Research*. 3(2), 6-21. Article DOI: https://doi.org/10.47001/CJESR/2026.302002

***** End of the Article *****