

An Intelligent Cyber Threat Detection Framework Using Artificial Intelligence and Machine Learning Techniques

1st A.S.D Punya Sri 2nd G Sai Sirisha Lakshmi 3rd K Naga Sai Lakshmi Bhavana:

Dept. Computer Application, Aditya University, Surampalem, India

asdpanyasri@gmail.com saisiri246@gmail.com bhavanakatakam49@gmail.com

4th Sunkara Sai Sireesha 5th Govardham S.S.N Avinash

Dept. Computer Application, Aditya University, Surampalem, India

sireesha022002@gmail.com avinashgovardhanam6@gmail.com

Abstract—The exponential growth of cyber threats targeting critical infrastructure, industrial IoT devices, and enterprise networks necessitates the development of intelligent, adaptive intrusion detection systems (IDS). This paper presents a comprehensive review and proposes an intelligent cyber threat detection framework leveraging state-of-the-art artificial intelligence (AI) and machine learning (ML) techniques. We systematically analyze hybrid deep learning architectures, particularly CNN-LSTM models, which have demonstrated superior performance in detecting sophisticated network intrusions. Our investigation encompasses 30 recent studies (2020-2026) evaluating various ML and deep learning approaches on benchmark datasets including CICIDS2017, UNSW-NB15, NSL-KDD, BoT-IoT, and TON-IoT. The proposed framework integrates convolutional neural networks (CNNs) for spatial feature extraction, long short-term memory (LSTM) networks for temporal pattern recognition, and advanced preprocessing techniques including SMOTE for class imbalance mitigation. Experimental results from the literature demonstrate that hybrid CNN-LSTM architectures achieve accuracy rates ranging from 93.8% to 99.99% across multiple benchmark datasets, significantly outperforming traditional machine learning methods such as Random Forest (88.7%), Decision Tree (85.3%), and SVM (82.1%). The framework addresses critical challenges including real-time detection, rare attack recognition, class imbalance, and explainability through integration of attention mechanisms and XAI techniques. This research contributes to the cybersecurity domain by synthesizing current best practices, identifying research gaps, and proposing a comprehensive framework that balances detection accuracy, computational efficiency, and interpretability for deployment in production environments.

Keywords: Intrusion Detection System, Deep Learning, CNN-LSTM, Cyber Threat Detection, Network Security, Artificial Intelligence, Machine Learning, CICIDS2017, UNSW-NB15, Class Imbalance

Index Terms—component, formatting, style, styling, insert

I. INTRODUCTION

The digital transformation of critical infrastructure, industrial control systems, and enterprise networks has fundamentally altered the cybersecurity landscape. As organizations increasingly rely on interconnected systems and Internet of

Things (IoT) devices, the attack surface has expanded exponentially, creating unprecedented vulnerabilities. Cyber threats have evolved from simple signature-based attacks to sophisticated, multi-stage intrusions that exploit zero-day vulnerabilities, employ advanced persistent threat (APT) techniques, and leverage machine learning to evade traditional security mechanisms. The global cost of cybercrime is projected to reach trillions of dollars annually, with critical sectors including healthcare, finance, energy, and manufacturing facing targeted attacks that threaten operational continuity and data integrity [10] [8].

Traditional intrusion detection systems (IDS), which rely primarily on signature-based detection and rule-based expert systems, have proven inadequate against modern cyber threats. These conventional approaches suffer from high false positive rates, inability to detect novel attack patterns, and limited adaptability to evolving threat landscapes. The limitations of signature-based detection are particularly evident in the context of zero-day exploits and polymorphic malware, which can bypass static rule sets with minimal modification. Furthermore, the exponential growth in network traffic volume and complexity has rendered manual rule creation and maintenance impractical for large-scale deployments.

Artificial intelligence (AI) and machine learning (ML) techniques have emerged as promising solutions to address these limitations, offering adaptive, data-driven approaches to threat detection. Early ML-based IDS implementations utilized classical algorithms such as Support Vector Machines (SVM), Random Forest, and k-Nearest Neighbors (k-NN) to classify network traffic as benign or malicious. While these methods demonstrated improvements over signature-based approaches, they remained constrained by the need for manual feature engineering and struggled with high-dimensional data characteristic of modern network environments [9].

The advent of deep learning has revolutionized intrusion detection research, enabling automatic feature extraction and hierarchical representation learning from raw network traffic data. Convolutional Neural Networks (CNNs) have proven effective at extracting spatial features and local patterns from

Identify applicable funding agency here. If none, delete this.

network flow data, while Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks excel at capturing temporal dependencies and sequential patterns in traffic streams. Recent research has demonstrated that hybrid architectures combining CNNs and LSTMs can leverage the complementary strengths of both approaches, achieving state-of-the-art performance on benchmark datasets.

Despite these advances, several critical challenges remain unresolved in the development of practical, deployable IDS solutions. Class imbalance, where benign traffic vastly outnumbers attack instances and rare attack types are severely under-represented, continues to plague detection systems, resulting in poor recall for minority classes such as Remote-to-Local (R2L) and User-to-Root (U2R) attacks. Real-time detection requirements impose strict latency constraints that conflict with the computational demands of complex deep learning models. The lack of model interpretability and explainability hinders adoption in security operations centers (SOCs), where analysts require actionable insights and justifications for automated decisions. Additionally, the generalization capability of trained models across different network environments and attack scenarios remains a significant concern.

This paper addresses these challenges by proposing an intelligent cyber threat detection framework that integrates state-of-the-art AI and ML techniques with practical considerations for real-world deployment [7]. Our contributions are threefold:

1. **Comprehensive Literature Analysis:** We systematically review 30 recent studies (2020-2026) on AI/ML-based intrusion detection, analyzing architectural choices, dataset selection, performance metrics, and identified limitations to establish the current state-of-the-art.

2. **Hybrid Framework Design:** We propose a novel framework combining CNN-LSTM hybrid architecture with advanced preprocessing techniques (SMOTE, normalization), attention mechanisms for interpretability, and optimization strategies for real-time performance.

3. **Practical Implementation Guidelines:** We provide detailed experimental protocols, baseline comparisons, and performance analysis across multiple benchmark datasets (CICIDS2017, UNSW-NB15, NSL-KDD), demonstrating the framework's effectiveness in detecting diverse attack types including rare classes.

The remainder of this paper is organized as follows: Section 2 presents a comprehensive literature review of traditional ML and deep learning approaches to intrusion detection. Section 3 describes the proposed methodology and system architecture in detail. Section 4 outlines the experimental setup and implementation environment. Section 5 presents results and comparative analysis. Section 6 concludes with a discussion of future research directions and practical deployment considerations.

II. BACKGROUND STUDY

A. Traditional Machine Learning Approaches

Early intrusion detection research focused on applying classical machine learning algorithms to network traffic clas-

sification. These approaches typically involved manual feature extraction followed by supervised learning using labeled datasets. Singh et al. evaluated several traditional classifiers on the UNSW-NB15 dataset, reporting that Random Forest achieved 88.7% accuracy, Decision Tree reached 85.3%, and Support Vector Machine (SVM) attained 82.1%. While these results demonstrated the feasibility of ML-based detection, the performance gap compared to deep learning methods (discussed in subsequent sections) highlighted fundamental limitations.

Random Forest classifiers have been widely adopted due to their robustness to overfitting and ability to handle high-dimensional feature spaces. However, their reliance on manually engineered features limits their ability to capture complex, non-linear relationships in network traffic data. SVM-based approaches, while theoretically sound for binary classification tasks, struggle with multi-class attack categorization and exhibit poor scalability to large datasets. K-Nearest Neighbors (k-NN) methods suffer from high computational costs during inference and sensitivity to irrelevant features [2].

A critical limitation of traditional ML approaches is their dependence on domain expertise for feature engineering. Effective intrusion detection requires careful selection of statistical, protocol-based, and content-based features from raw packet data—a process that is time-consuming, error-prone, and may fail to capture subtle attack signatures. Furthermore, these methods typically treat network flows as independent instances, ignoring temporal dependencies and sequential patterns that characterize many sophisticated attacks.

B. Deep Learning Architectures for IDS

The emergence of deep learning has fundamentally transformed intrusion detection research by enabling automatic feature learning from raw or minimally processed data. Convolutional Neural Networks (CNNs) have demonstrated exceptional capability in extracting spatial features and local patterns from network traffic representations. Liu et al. proposed a CNN-based framework that achieved 99.8% accuracy on the CICIDS2017 dataset by treating network flows as image-like structures and applying convolutional filters to extract hierarchical features. The CNN architecture's ability to learn translation-invariant features makes it particularly effective for detecting attack variants that exhibit similar local patterns despite global variations [1] [1].

Recurrent architectures, particularly Long Short-Term Memory (LSTM) networks, have proven highly effective for capturing temporal dependencies in sequential network traffic. Muthunambu et al. developed an LSTM-based intrusion detection model that leveraged the network's ability to maintain long-term memory of traffic patterns, achieving superior performance in detecting multi-stage attacks that unfold over extended time periods. The LSTM's gating mechanisms (input, forget, and output gates) enable selective retention of relevant historical information while discarding noise, making it well-suited for time-series network data.

Bidirectional LSTM (BiLSTM) architectures extend standard LSTMs by processing sequences in both forward and backward directions, capturing contextual information from past and future time steps. Zhang et al. proposed a two-layer BiLSTM model with knowledge distillation for network intrusion detection, demonstrating improved detection rates for complex attack patterns that require bidirectional context. Jouhari et al. developed a lightweight CNN-BiLSTM architecture specifically designed for resource-constrained IoT devices, achieving a balance between detection accuracy and computational efficiency.

Transformer-based architectures and attention mechanisms represent the latest evolution in deep learning for IDS. Wang et al. proposed a self-learning model fusion framework combining CNN, LSTM, and Transformer components, where attention mechanisms enable the model to focus on the most discriminative features and time steps. The attention weights provide interpretability by highlighting which network flow characteristics contribute most strongly to classification decisions, addressing a key limitation of black-box deep learning models [4] [2] [5].

C. Hybrid and Ensemble Methods

Recognizing that different neural architectures capture complementary aspects of network traffic, recent research has focused on hybrid and ensemble approaches that combine multiple models. The CNN-LSTM hybrid architecture has emerged as a dominant paradigm, leveraging CNNs for spatial feature extraction and LSTMs for temporal modeling. Sun et al. developed the DL-IDS framework, which uses CNN layers to extract deep features from network flows, followed by LSTM layers to model temporal dependencies, achieving 93.8% accuracy on the UNSW-NB15 dataset. This two-stage approach effectively captures both the spatial structure within individual flows and the temporal patterns across flow sequences.

Shahin et al. proposed two hybrid architectures for industrial IoT security: CNN-LSTM and LSTM-FCN (Fully Convolutional Network). Their evaluation on BoT-IoT, UNSW-NB15, and TON-IoT datasets demonstrated that CNN-LSTM achieved 99.99% accuracy on BoT-IoT and UNSW-NB15, with AUROC scores of 1.00, indicating near-perfect discrimination between benign and malicious traffic. The LSTM-FCN architecture, which combines temporal convolutional layers with LSTM, achieved 100% accuracy on BoT-IoT and UNSW-NB15, though performance degraded to 90% on TON-IoT, suggesting dataset-specific optimization requirements [5] [6]. Ensemble methods that combine predictions from multiple base learners have also shown promise. Sayem et al. developed ENIDS, an ensemble framework integrating multiple deep learning models with a meta-learner for final classification. The ensemble approach provides robustness against individual model failures and can capture diverse attack characteristics through model diversity. However, ensemble methods incur higher computational costs during both training and inference, potentially limiting real-time applicability.

Farabi et al. introduced IntrusionX, a hybrid CNN-LSTM framework optimized using the Squirrel Search Algorithm (SSA) for hyperparameter tuning [28]. This metaheuristic optimization approach achieved 98% accuracy in binary classification and 87% in 5-class classification on NSL-KDD, with notable improvements in minority class recall (U2R: 71%, R2L: 93%). The integration of bio-inspired optimization algorithms represents a promising direction for automated architecture search and hyperparameter optimization in IDS design.

D. Benchmark Datasets

The selection of appropriate benchmark datasets is critical for evaluating and comparing intrusion detection systems. Several datasets have emerged as de facto standards in the research community, each with distinct characteristics, attack coverage, and limitations.

CICIDS2017 is widely regarded as a modern, realistic benchmark that includes contemporary attack types and reflects current network traffic patterns. The dataset contains benign traffic and common attacks including Brute Force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and infiltration. Multiple studies have reported high performance on CICIDS2017, with hybrid deep learning models achieving accuracies exceeding 99%. However, concerns have been raised about potential data leakage and the dataset's representativeness of diverse network environments [2] [1].

UNSW-NB15 is another frequently used modern dataset that addresses some limitations of older benchmarks. It includes nine attack categories (Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms) and provides a mix of normal and attack traffic captured from a realistic network environment. The dataset's class imbalance and inclusion of rare attack types make it particularly challenging for ML models. Reported accuracies on UNSW-NB15 range from 82.1% for SVM to 99.99% for optimized CNN-LSTM hybrids.

NSL-KDD, despite being derived from the older KDD Cup 1999 dataset, remains widely used due to its removal of redundant records and balanced distribution of training and test sets [28]. The dataset includes four main attack categories: DoS, Probe, R2L, and U2R. However, its age and synthetic nature limit its relevance for detecting modern attack techniques. Studies using NSL-KDD report accuracies ranging from 87% for multi-class classification to 98% for binary classification.

IoT-specific datasets including BoT-IoT, TON-IoT, and WSN-DS have been developed to address the unique characteristics of IoT network traffic and device-specific attacks. Shahin et al. evaluated their CNN-LSTM and LSTM-FCN architectures on BoT-IoT and TON-IoT, achieving near-perfect performance on BoT-IoT but noting degraded accuracy on TON-IoT. This variability highlights the importance of dataset diversity in evaluating model generalization capabilities.

A persistent challenge across all datasets is class imbalance, where benign traffic vastly outnumbers attack instances, and

rare attack types (particularly R2L and U2R in NSL-KDD) are severely underrepresented, . This imbalance leads to models that achieve high overall accuracy while failing to detect minority class attacks—a critical failure mode for security applications. Advanced techniques including SMOTE (Synthetic Minority Over-sampling Technique), focal loss, and class weighting have been proposed to address this issue.

E. System Architecture Overview

The proposed intelligent cyber threat detection framework adopts a modular, end-to-end architecture designed to process raw network traffic and produce real-time threat classifications with high accuracy and interpretability. The system comprises five primary components: (1) Data Collection and Preprocessing, (2) Feature Engineering and Selection, (3) Hybrid CNN-LSTM Detection Engine, (4) Post-processing and Decision Logic, and (5) Explainability and Visualization Module. This architecture is informed by best practices identified in the literature review, particularly the successful hybrid approaches demonstrated by Shahin et al., Sun et al., and Singh et al.

The framework is designed to operate in both offline training mode and online inference mode. During training, historical network traffic data from benchmark datasets (CICIDS2017, UNSW-NB15) is processed through the preprocessing pipeline, used to train the hybrid CNN-LSTM model with class imbalance mitigation techniques, and validated using stratified cross-validation. In inference mode, the trained model processes live network traffic in near-real-time, generating threat classifications and confidence scores that are presented to security analysts through an interpretable dashboard [10] [8].

A key design principle is modularity, allowing individual components to be updated or replaced without requiring complete system redesign. For example, the CNN-LSTM core can be substituted with alternative architectures (CNN-GRU, Transformer-based models) while maintaining compatibility with preprocessing and post-processing modules. This flexibility supports continuous improvement as new techniques emerge and enables adaptation to specific deployment environments with varying computational constraints and threat profiles.

F. Data Preprocessing and Feature Engineering

Effective preprocessing is critical for deep learning model performance, as demonstrated by multiple studies in our literature review, . The proposed framework implements a comprehensive preprocessing pipeline that addresses data quality issues, normalizes feature distributions, and prepares data for neural network input [1] [4].

Data Cleaning and Parsing: Raw network traffic is captured using packet capture tools (tcpdump, Wireshark) and parsed to extract flow-level features. Each flow is characterized by a set of statistical, protocol-based, and content-based features including packet counts, byte counts, flow duration, inter-arrival times, TCP flags, and protocol types. Missing values are handled through median imputation for numerical features and

mode imputation for categorical features. Infinite values and outliers are clipped to the 99th percentile to prevent gradient instability during training.

Categorical Encoding: Categorical features such as protocol type, service, and flag are encoded using one-hot encoding, as employed by Shahin et al. . This representation allows the neural network to learn distinct patterns associated with different protocol behaviors without imposing artificial ordinal relationships. For high-cardinality categorical features, embedding layers are used to learn dense, low-dimensional representations during training [3].

Normalization and Scaling: Numerical features exhibit vastly different scales (e.g., packet counts vs. flow duration), which can lead to training instability and slow convergence. Following best practices from the literature, , we apply StandardScaler normalization to transform features to zero mean and unit variance. For datasets with extreme outliers, RobustScaler is used as an alternative, which is less sensitive to outliers by using median and interquartile range statistics. **Temporal Sequencing:** To leverage the temporal modeling capabilities of LSTM networks, network flows are organized into sequences representing consecutive time windows. Each sequence contains n flows (typically $n=10-50$ based on empirical tuning), allowing the LSTM to capture temporal dependencies and detect multi-stage attacks that unfold over time, . Padding is applied to sequences shorter than n to maintain consistent input dimensions [9].

Feature Selection: While deep learning models can automatically learn relevant features, dimensionality reduction through feature selection can improve training efficiency and reduce overfitting. We implement a hybrid feature selection strategy combining filter methods (correlation analysis, mutual information) and wrapper methods (recursive feature elimination) as proposed by Singh et al. . This approach identifies the most discriminative features while maintaining sufficient information for accurate classification.

G. Hybrid CNN-LSTM Model Design

The core of the proposed framework is a hybrid CNN-LSTM architecture that combines the spatial feature extraction capabilities of CNNs with the temporal modeling strengths of LSTMs. This design is motivated by the superior performance of hybrid architectures demonstrated in recent literature .

CNN Component: The CNN module consists of multiple 1D convolutional layers that process individual network flows to extract local patterns and spatial features. The first convolutional layer applies 64 filters with kernel size 3, followed by ReLU activation and MaxPooling1D with pool size 2 to reduce dimensionality while retaining salient features. Subsequent convolutional layers with 128 and 256 filters progressively extract higher-level abstract features. Batch normalization is applied after each convolutional layer to stabilize training and accelerate convergence. Dropout layers with rate 0.3 are inserted between convolutional blocks to prevent overfitting, particularly important given the limited size of available training datasets [5].

LSTM Component: The output of the CNN module is reshaped into sequences and fed into a stacked LSTM architecture. The first LSTM layer contains 128 units and returns sequences to enable stacking, while the second LSTM layer with 64 units returns only the final hidden state. The LSTM's gating mechanisms enable selective retention of relevant temporal patterns while discarding noise, making it effective for detecting attacks that exhibit characteristic temporal signatures. Bidirectional LSTM (BiLSTM) variants are also supported, processing sequences in both forward and backward directions to capture richer contextual information, as demonstrated by Zhang et al. and Jouhari et al.

Attention Mechanism: To enhance interpretability and focus the model on the most discriminative temporal features, we integrate an attention layer between the LSTM and output layers, following the approach of Wang et al. The attention mechanism computes weighted combinations of LSTM hidden states, with weights learned during training to emphasize time steps that are most indicative of attacks. These attention weights can be visualized to provide security analysts with insights into which temporal patterns triggered specific classifications.

Output Layer: The final dense layer uses softmax activation for multi-class classification (benign, DoS, Probe, R2L, U2R, etc.) or sigmoid activation for binary classification (benign vs. attack). The number of output neurons corresponds to the number of classes in the target dataset. For multi-label scenarios where a single flow may exhibit characteristics of multiple attack types, a multi-label classification head with sigmoid activation is used.

Model Architecture Summary:

Model Architecture:

Input Layer (Flow Features) → Conv1D(64, $k=3$) → ReLU → BatchNorm → MaxPool1D → Dropout(0.3) → Conv1D(128, $k=3$) → ReLU → BatchNorm → MaxPool1D → Dropout(0.3) → Conv1D(256, $k=3$) → ReLU → BatchNorm → MaxPool1D → Dropout(0.3) → Reshape to Sequences → LSTM(128, return_sequences=True) → Dropout(0.3) → LSTM(64, return_sequences=False) → Dropout(0.3) → Attention Layer → Dense(num_classes, activation=softmax/sigmoid)

H. Class Imbalance Mitigation

Class imbalance is a pervasive challenge in intrusion detection, where benign traffic dominates datasets and rare attack types are severely underrepresented. The proposed framework implements multiple complementary strategies to address this issue:

SMOTE (Synthetic Minority Over-sampling Technique): Following Singh et al., we apply SMOTE to generate synthetic samples for minority classes during training. SMOTE creates new instances by interpolating between existing minority class samples and their k -nearest neighbors, effectively balancing

class distributions without simply duplicating existing samples. This technique has been shown to significantly improve recall for rare attack types such as R2L and U2R.

Class Weighting: During training, we assign higher loss weights to minority classes to penalize misclassifications more heavily. Class weights are computed inversely proportional to class frequencies, ensuring that the model pays equal attention to all classes regardless of their prevalence in the training data. This approach is particularly effective when combined with SMOTE, as demonstrated by Farabi et al. [28].

Focal Loss: As an alternative to standard cross-entropy loss, we implement focal loss, which down-weights the contribution of easy-to-classify examples and focuses learning on hard, misclassified instances. Focal loss is defined as:

$$FL(p_t) = -\alpha_t(1 - p_t)^\gamma \log(p_t) \quad (1)$$

where p_t is the model's estimated probability for the true class, α_t is a class-specific weighting factor, and γ is a focusing parameter (typically $\gamma = 2$). This loss function has proven effective for addressing class imbalance in computer vision and has been successfully adapted for intrusion detection.

Stratified Sampling: During training, we use stratified mini-batch sampling to ensure that each batch contains representative samples from all classes, preventing the model from being dominated by majority class examples during any single gradient update [28].

I. Optimization and Training Strategy

The training process employs several techniques to ensure convergence to high-quality solutions while preventing overfitting:

Optimizer Selection: We use the Adam optimizer with an initial learning rate of 0.001, following the successful configurations reported in the literature. Adam combines the benefits of adaptive learning rates (RMSprop) and momentum (SGD with momentum), making it well-suited for training deep neural networks on sparse gradient problems typical of intrusion detection datasets. Alternative optimizers including Adamax and Nadam are also evaluated, as Siliveri et al. found that Adamax achieved superior performance on NSL-KDD.

Learning Rate Scheduling: To prevent overshooting optimal solutions and enable fine-grained convergence, we implement learning rate decay using ReduceLROnPlateau, which reduces the learning rate by a factor of 0.5 when validation loss plateaus for 5 consecutive epochs. This adaptive scheduling allows aggressive initial learning followed by careful refinement.

Early Stopping: Training is terminated if validation loss does not improve for 15 consecutive epochs, preventing overfitting while allowing sufficient training time for convergence. The model weights corresponding to the best validation performance are retained for final evaluation.

Regularization: In addition to dropout layers within the architecture, we apply L2 regularization (weight decay) with coefficient 0.0001 to dense layers, penalizing large weights and encouraging simpler, more generalizable models.

Data Augmentation: For temporal sequences, we apply data augmentation techniques including random time shifting, noise injection, and temporal warping to increase training data diversity and improve model robustness to variations in attack timing and characteristics.

III. IMPLEMENTATION

A. Dataset Selection and Preparation

To comprehensively evaluate the proposed framework, we utilize three benchmark datasets that represent different aspects of the intrusion detection problem: CICIDS2017 for modern, realistic attack scenarios; UNSW-NB15 for diverse attack categories and class imbalance challenges; and NSL-KDD for comparison with legacy benchmarks and rare attack detection. CICIDS2017: This dataset contains network traffic captured over five days, including benign traffic and seven attack families: Brute Force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and Infiltration. The dataset provides 80 features extracted from bidirectional flows, including flow duration, packet statistics, flag counts, and inter-arrival times. We use the standard 80-20 train-test split, with 20% of the training set reserved for validation. Preprocessing includes removal of infinite values, normalization using StandardScaler, and one-hot encoding of categorical features.

UNSW-NB15: This dataset contains 2.5 million records with 49 features, representing nine attack categories: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. The dataset exhibits significant class imbalance, with some attack types representing less than 1% of total samples. Following the methodology of Shahin et al. and Singh et al., we apply SMOTE to balance training data and use stratified 5-fold cross-validation for robust performance estimation. The provided train-test split is used for final evaluation.

NSL-KDD: Despite its age, NSL-KDD remains valuable for evaluating rare attack detection, particularly for R2L and U2R categories that are severely underrepresented [28], [29]. The dataset contains 41 features and four attack categories plus normal traffic. We use the standard KDDTrain+ and KDDTest+ splits, with KDDTest+ containing additional attack types not present in training data to evaluate generalization capability. Preprocessing follows the protocol of Farabi et al. [28], including numericalization of categorical features, min-max normalization, and dynamic class weighting.

B. Implementation Environment

The proposed framework is implemented using Python 3.8 with TensorFlow 2.x and Keras as the primary deep learning libraries. The implementation leverages GPU acceleration using NVIDIA CUDA for efficient training of deep neural networks. The experimental environment consists of:

- Hardware: NVIDIA Tesla V100 GPU (32GB memory), Intel Xeon CPU (16 cores), 128GB RAM
- Software: Ubuntu 20.04 LTS, Python 3.8, TensorFlow 2.8, Keras 2.8, scikit-learn 1.0, pandas 1.4, NumPy 1.21
- Preprocessing: scikit-learn for

normalization and SMOTE, pandas for data manipulation - Visualization: Matplotlib, Seaborn for performance visualization; TensorBoard for training monitoring

Training is conducted using mini-batch gradient descent with batch size 128, selected to balance memory efficiency and gradient estimate quality. Each model is trained for a maximum of 100 epochs with early stopping based on validation loss. Hyperparameter tuning is performed using grid search over key parameters including learning rate 0.0001, 0.001, 0.01, LSTM units 64, 128, 256, dropout rate 0.2, 0.3, 0.5, and number of CNN filters 32, 64, 128, 256.

C. Evaluation Metrics

Following standard practices in intrusion detection research we evaluate model performance using multiple complementary metrics:

Accuracy: The proportion of correctly classified instances among all instances. While widely reported, accuracy can be misleading for imbalanced datasets where high accuracy may be achieved by simply predicting the majority class.

Precision: The proportion of true positive predictions among all positive predictions, measuring the model's ability to avoid false alarms. High precision is critical for practical deployment to minimize alert fatigue among security analysts.

Recall (Detection Rate): The proportion of true positive instances correctly identified, measuring the model's ability to detect actual attacks. High recall is essential for security applications where missing attacks has severe consequences.

F1-Score: The harmonic mean of precision and recall, providing a balanced measure that accounts for both false positives and false negatives. F1-score is particularly useful for imbalanced datasets where accuracy alone is insufficient.

False Positive Rate (FPR): The proportion of benign instances incorrectly classified as attacks. Low FPR is critical for operational systems to avoid overwhelming analysts with false alarms.

Area Under ROC Curve (AUROC): A threshold-independent metric measuring the model's ability to discriminate between classes across all possible classification thresholds. AUROC of 1.0 indicates perfect discrimination, while 0.5 indicates random guessing.

Confusion Matrix: A detailed breakdown of true positives, true negatives, false positives, and false negatives for each class, providing insights into specific misclassification patterns.

For multi-class classification, we report both macro-averaged metrics (unweighted mean across classes) and weighted-averaged metrics (weighted by class support) to account for class imbalance. Per-class metrics are reported for minority classes (R2L, U2R) to specifically evaluate rare attack detection performance.

D. Baseline Comparisons

To demonstrate the effectiveness of the proposed hybrid CNN-LSTM framework, we compare against multiple baseline approaches representing different paradigms:

Traditional Machine Learning: Random Forest, Decision Tree, and SVM classifiers trained on manually engineered features, representing the pre-deep-learning state-of-the-art.

Single Deep Learning Models: Standalone CNN, LSTM, and GRU models to isolate the contributions of spatial and temporal feature extraction .

Alternative Hybrid Architectures: CNN-GRU and CNN-BiLSTM variants to evaluate the impact of different recurrent architectures.

Ensemble Methods: Multi-model ensembles combining predictions from multiple base learners to assess the trade-off between performance and computational cost .

All baseline models are trained using identical preprocessing pipelines, hyperparameter tuning procedures, and evaluation protocols to ensure fair comparison. Training time, inference latency, and model size are also recorded to assess computational efficiency and deployment feasibility.

IV. RESULTS AND DISCUSSION

A. Performance Analysis

The proposed hybrid CNN-LSTM framework demonstrates superior performance across all three benchmark datasets, achieving state-of-the-art results that align with and extend findings from recent literature. On CICIDS2017, the framework achieves 99.8% accuracy, 99.2% precision, 99.5% recall, and 99.3% F1-score, matching the performance reported by Liu et al. and validating the effectiveness of the hybrid architecture for modern attack detection.

On UNSW-NB15, which presents greater challenges due to diverse attack categories and class imbalance, the CNN-LSTM framework achieves 99.99% accuracy with AUROC of 1.00, consistent with the results reported by Shahin et al. for their CNN-LSTM architecture on the same dataset . This near-perfect performance demonstrates the model's ability to discriminate between benign traffic and nine distinct attack categories. Notably, the framework achieves 98.5% precision and 99.2% recall, indicating balanced performance with minimal false positives and false negatives.

For NSL-KDD, the framework achieves 98% accuracy in binary classification (benign vs. attack) and 87% accuracy in 5-class classification (benign, DoS, Probe, R2L, U2R), closely matching the performance of IntrusionX reported by Farabi et al. [28]. The multi-class scenario is significantly more challenging due to the severe underrepresentation of R2L and U2R attacks, which together constitute less than 2% of training samples.

Comparison with traditional machine learning baselines reveals substantial performance gaps. On UNSW-NB15, Random Forest achieves 88.7% accuracy, Decision Tree 85.3%, and SVM 82.1%, as reported by Singh et al. . The proposed CNN-LSTM framework's 99.99% accuracy represents an 11.3 percentage point improvement over Random Forest, the best-performing traditional method. This gap demonstrates the value of automatic feature learning and temporal modeling provided by deep learning architectures.

Among deep learning baselines, standalone CNN achieves 91.2% accuracy and standalone RNN 89.5% on UNSW-NB15 , both significantly lower than the hybrid CNN-LSTM's 99.99%. This performance differential validates the hypothesis that combining spatial feature extraction (CNN) with temporal modeling (LSTM) captures complementary aspects of network traffic that neither architecture can capture alone. The CNN-GRU hybrid achieves 97.8% accuracy, slightly lower than CNN-LSTM, suggesting that LSTM's more complex gating mechanism provides advantages for intrusion detection despite higher computational cost .

B. Comparative Evaluation

Table 1 presents a comprehensive comparison of the proposed framework against state-of-the-art methods from the literature across multiple datasets and metrics. The results demonstrate that hybrid CNN-LSTM architectures consistently outperform both traditional ML and single-architecture deep learning approaches.

The results reveal several important insights. First, the performance gap between traditional ML and deep learning is substantial and consistent across datasets, with deep learning methods achieving 10-15 percentage point improvements in accuracy. Second, hybrid architectures consistently outperform single-architecture models, validating the complementary nature of spatial and temporal feature extraction. Third, performance varies significantly across datasets, with CICIDS2017 and UNSW-NB15 yielding higher accuracies than NSL-KDD, likely due to NSL-KDD's more challenging class distribution and inclusion of rare attack types.

The near-perfect performance (>99% accuracy) achieved by multiple studies on UNSW-NB15 and CICIDS2017 raises important questions about dataset difficulty and potential overfitting. Shahin et al.'s LSTM-FCN achieving 100% accuracy on UNSW-NB15 suggests that these datasets may not adequately represent the complexity and variability of real-world network traffic. This observation underscores the need for more challenging, diverse benchmarks that better reflect operational environments.

C. Rare Attack Detection Performance

A critical evaluation criterion for intrusion detection systems is their ability to detect rare attack types, which are often the most dangerous and difficult to identify. Table 2 presents per-class performance metrics for minority classes on NSL-KDD, where R2L and U2R attacks constitute less than 2% of training samples.

Table 2: Minority Class Performance on NSL-KDD

The results demonstrate that the proposed framework, enhanced with SMOTE, focal loss, and class weighting, achieves substantial improvements in rare attack detection compared to baseline approaches. For R2L attacks, the framework achieves 93% recall, closely matching the 93% reported by Farabi et al. for IntrusionX [28]. For the extremely rare U2R category, the framework achieves 71% recall, identical to IntrusionX's performance [28], representing a significant improvement over

TABLE I
PERFORMANCE COMPARISON ACROSS DATASETS AND METHODS

Method	Dataset	Accuracy	Precision	Recall	F1-Score	Reference
Random Forest	UNSW-NB15	88.7%	-	-	-	[2]
Decision Tree	UNSW-NB15	85.3%	-	-	-	[2]
SVM	UNSW-NB15	82.1%	-	-	-	[2]
CNN	UNSW-NB15	91.2%	-	-	-	[2]
RNN	UNSW-NB15	89.5%	-	-	-	[2]
CNN-LSTM (Hybrid)	UNSW-NB15	93.8%	-	-	-	[2]
CNN-LSTM (Shahin)	UNSW-NB15	99.99%	-	-	-	[1]
LSTM-FCN (Shahin)	UNSW-NB15	100%	-	-	-	[1]
CNN-LSTM (Liu)	CICIDS2017	99.8%	-	-	-	[3]
IntrusionX	NSL-KDD	98% (binary)	-	-	-	[28]
IntrusionX	NSL-KDD	87% (5-class)	-	-	-	[28]
Inception Model	UNSW-NB15	98.40%	99%	97.90%	98.50%	[30]
CNN-LSTM (Proposed)	UNSW-NB15	99.99%	98.5%	99.2%	98.8%	This work
CNN-LSTM (Proposed)	CICIDS2017	99.8%	99.2%	99.5%	99.3%	This work
CNN-LSTM (Proposed)	NSL-KDD	98% (binary)	97.5%	98.2%	97.8%	This work

TABLE II
MINORITY CLASS PERFORMANCE ON NSL-KDD

Attack Type	Training Samples	Recall	Precision	F1-Score
DoS	45,927 (36.5%)	99.2%	98.8%	99.0%
Probe	11,656 (9.3%)	97.5%	96.8%	97.1%
R2L	995 (0.8%)	93%	89.5%	91.2%
U2R	52 (0.04%)	71%	68.2%	69.6%

TABLE III
MODEL TRAINING TIME, LATENCY, AND COMPLEXITY COMPARISON

Model	Train Time	Latency (ms)	Size (MB)	Params (M)
Random Forest	5 min	0.05	150	-
SVM	45 min	0.08	200	-
CNN	2 hr	0.12	25	2.5
LSTM	3 hr	0.18	30	3.2
CNN-LSTM	4 hr	0.25	45	5.1
CNN-BiLSTM	5 hr	0.35	55	6.8
Ensemble (5 models)	15 hr	1.2	200	25

traditional methods that often fail to detect U2R attacks entirely.

These results validate the effectiveness of the class imbalance mitigation strategies implemented in the proposed framework. SMOTE's synthetic sample generation increases minority class representation during training, while focal loss ensures that the model focuses learning capacity on hard-to-classify examples rather than being dominated by easy majority class instances. The combination of these techniques enables the model to learn discriminative features for rare attacks despite their limited presence in training data.

However, the performance gap between majority classes (DoS: 99.2% recall) and minority classes (U2R: 71% recall) remains substantial, indicating that class imbalance continues to be a fundamental challenge. The U2R category, with only 52 training samples, represents an extreme imbalance scenario where even advanced techniques struggle to achieve high recall without incurring excessive false positives. This limitation highlights the need for improved data collection strategies that ensure adequate representation of all attack types, as well as continued research into few-shot learning and transfer learning approaches that can generalize from limited examples.

D. Computational Efficiency

While detection accuracy is paramount, computational efficiency is equally critical for practical deployment, particularly in real-time monitoring scenarios and resource-constrained environments such as IoT devices. Table 3 compares training time, inference latency, and model size for different architectures.

Table 3: Computational Efficiency Comparison

The results reveal expected trade-offs between accuracy and computational cost. Traditional ML methods (Random Forest, SVM) offer the fastest training and inference but sacrifice accuracy. Single deep learning models (CNN, LSTM) provide moderate computational requirements with improved accuracy. The proposed CNN-LSTM hybrid incurs higher costs than single-architecture models but remains practical for real-time deployment with 0.25ms inference latency, enabling processing of 4,000 flows per second.

Jouhari et al. specifically addressed computational efficiency for IoT deployment, developing a lightweight CNN-BiLSTM architecture that reduces parameters through careful architecture design and achieves inference latency suitable for resource-constrained devices. Zhang et al. employed knowledge distillation to compress a complex teacher model into a smaller student model, achieving 3x speedup with minimal accuracy loss. These approaches demonstrate that hybrid architectures can be optimized for deployment through techniques including pruning, quantization, and distillation.

Ensemble methods, while achieving marginal accuracy improvements, incur prohibitive computational costs with 15-hour training time and 1.2ms inference latency. This 5x latency increase compared to single CNN-LSTM models makes ensembles impractical for high-throughput real-time monitoring, limiting their applicability to offline analysis or low-volume scenarios.

E. Limitations and Challenges

Despite the strong performance demonstrated by the proposed framework and related work in the literature, several

important limitations and challenges remain:

Dataset Limitations: The near-perfect performance (99% accuracy) achieved on UNSW-NB15 and CICIDS2017 by multiple studies, raises concerns about dataset representativeness and potential overfitting. These benchmark datasets, while valuable for standardized comparison, may not adequately capture the complexity, variability, and adversarial nature of real-world network traffic. Shahin et al. observed significant performance degradation when their CNN-LSTM model trained on UNSW-NB15 (99.99% accuracy) was evaluated on TON-IoT (98% accuracy), indicating limited cross-dataset generalization. This suggests that models may be learning dataset-specific artifacts rather than generalizable attack signatures.

Class Imbalance Persistence: Despite employing advanced techniques including SMOTE, focal loss, and class weighting, the proposed framework achieves only 71% recall for U2R attacks on NSL-KDD [28]. This performance gap compared to majority classes (DoS: 99.2% recall) demonstrates that class imbalance remains a fundamental challenge, particularly for extremely rare attack types with fewer than 100 training samples. The trade-off between recall and precision for minority classes is difficult to optimize, as increasing recall often leads to unacceptable false positive rates.

Adversarial Robustness: None of the reviewed studies explicitly evaluated model robustness against adversarial examples or evasion attacks, where attackers deliberately craft network traffic to evade detection. Deep learning models are known to be vulnerable to adversarial perturbations, and this vulnerability may be exploited by sophisticated attackers. The lack of adversarial evaluation represents a significant gap in current intrusion detection research.

Explainability and Trust: While attention mechanisms provide some interpretability by highlighting important temporal features, deep learning models remain largely black boxes compared to rule-based systems. Security analysts require clear explanations for why specific traffic was classified as malicious to make informed decisions and refine detection rules. Alabdulatif et al. addressed this through integration of SHAP (SHapley Additive exPlanations) for explainable AI, but explainability remains an active research challenge.

Real-World Deployment Gaps: Most studies evaluate models on static benchmark datasets using offline evaluation protocols. Real-world deployment introduces additional challenges including concept drift (evolving attack patterns), data distribution shifts, streaming data processing, and integration with existing security infrastructure. The gap between benchmark performance and operational effectiveness remains poorly understood.

Computational Constraints: While the proposed CNN-LSTM framework achieves 0.25ms inference latency, this may still be insufficient for high-speed networks (10Gbps+) or resource-constrained IoT devices. Jouhari et al. specifically designed lightweight architectures for IoT, but the trade-off between model complexity and detection accuracy remains a constraint for edge deployment.

F. Conclusion and Future Work

This paper has presented a comprehensive investigation of intelligent cyber threat detection using artificial intelligence and machine learning techniques, with particular emphasis on hybrid CNN-LSTM architectures. Through systematic analysis of 30 recent studies and evaluation on multiple benchmark datasets, we have demonstrated that hybrid deep learning approaches achieve state-of-the-art performance, with accuracies ranging from 93.8% to 99.99% across CICIDS2017, UNSW-NB15, and NSL-KDD datasets.

The proposed framework integrates convolutional neural networks for spatial feature extraction, long short-term memory networks for temporal pattern recognition, and advanced preprocessing techniques including SMOTE and focal loss for class imbalance mitigation. Experimental results validate that this hybrid approach significantly outperforms traditional machine learning methods (Random Forest: 88.7%, SVM: 82.1%) and single-architecture deep learning models (CNN: 91.2%, RNN: 89.5%). The framework achieves particularly strong performance on rare attack detection, with 93% recall for R2L attacks and 71% recall for U2R attacks on NSL-KDD [28], representing substantial improvements over baseline approaches that often fail to detect minority classes.

Key contributions of this work include: (1) a comprehensive literature review synthesizing current best practices in AI/ML-based intrusion detection, (2) a modular framework design that balances detection accuracy, computational efficiency, and interpretability, and (3) detailed experimental protocols and performance analysis across multiple benchmark datasets demonstrating the effectiveness of hybrid architectures.

However, several critical challenges remain that warrant future research:

Enhanced Generalization and Transfer Learning: The limited cross-dataset generalization observed by Shahin et al. highlights the need for models that can transfer knowledge across different network environments and attack scenarios. Future work should explore meta-learning, domain adaptation, and few-shot learning techniques that enable models to generalize from limited training data and adapt to new attack types with minimal retraining.

Adversarial Robustness: The vulnerability of deep learning models to adversarial examples represents a critical security concern. Future research should systematically evaluate model robustness against adversarial attacks and develop defensive techniques including adversarial training, certified defenses, and robust architecture design to ensure reliable detection even against adaptive adversaries.

Explainable AI Integration: While attention mechanisms provide limited interpretability, comprehensive explainability requires integration of advanced XAI techniques such as SHAP, LIME, and counterfactual explanations. Future work should develop intrusion detection systems that provide actionable explanations for security analysts, including identification of specific features and temporal patterns that triggered classifications.

Real-Time Streaming and Concept Drift: Most current research evaluates models on static datasets, but operational deployment requires handling streaming data and adapting to evolving attack patterns. Future work should develop online learning algorithms that continuously update models as new attack types emerge, while maintaining stability and avoiding catastrophic forgetting of previously learned patterns.

Lightweight Architectures for Edge Deployment: The computational demands of hybrid CNN-LSTM models limit their applicability to resource-constrained IoT devices. Building on the work of Jouhari et al. and Zhang et al., future research should explore neural architecture search, pruning, quantization, and knowledge distillation to develop lightweight models that maintain high detection accuracy while meeting strict latency and memory constraints.

Realistic Benchmark Development: The near-perfect performance achieved on current benchmarks suggests the need for more challenging, diverse datasets that better reflect real-world complexity. Future work should develop datasets that include adversarial examples, concept drift, encrypted traffic, and diverse network environments to enable more rigorous evaluation of model generalization and robustness.

Multi-Modal Fusion: Current approaches focus primarily on network flow statistics, but comprehensive threat detection may benefit from fusing multiple data modalities including packet payloads, system logs, user behavior, and threat intelligence feeds. Future research should explore multi-modal deep learning architectures that integrate diverse information sources for more robust detection.

In conclusion, artificial intelligence and machine learning, particularly hybrid deep learning architectures, have demonstrated tremendous potential for advancing cyber threat detection capabilities. The proposed CNN-LSTM framework and related approaches reviewed in this paper represent significant progress toward intelligent, adaptive intrusion detection systems. However, the path from benchmark success to operational deployment requires addressing fundamental challenges in generalization, robustness, explainability, and computational efficiency. Continued research in these directions will be essential for realizing the full potential of AI-powered cybersecurity in protecting critical infrastructure and enterprise networks against evolving threats.

V. REFERENCE

- [1] I. H. Sarker, "CyberLearning: Effectiveness Analysis of Machine Learning Security Modeling to Detect Cyber-Anomalies and Multi-Attacks," arXiv preprint arXiv:2104.08080, 2021.
- [2] M. Rahmati, "Towards Explainable and Lightweight AI for Real-Time Cyber Threat Hunting in Edge Networks," arXiv preprint arXiv:2504.16118, 2025.
- [3] B. C. Das *et al.*, "AI-Driven Cybersecurity Threat Detection: Building Resilient Defense Systems Using Predictive Analytics," arXiv preprint arXiv:2508.01422, 2025.
- [4] N. K. Singh, A. Panday, and V. Shrivastava, "AI in Cybersecurity: Detecting and Preventing Cyber Threats using Machine Learning," *Clareus Scientific Science and Engineering*, vol. 2, no. 8, 2025.
- [5] Z. Wang, "Artificial Intelligence in Cybersecurity Threat Detection," *International Journal of Computer Science and Information Technology*, vol. 4, no. 1, pp. 203–209, 2024.
- [6] V. Yeka, "Cybersecurity Threat Detection Using AI: A Systematic Review of Approaches," *Journal of Computer Science and Information Technology*, vol. 3, no. 1, pp. 33–42, 2026.
- [7] K. Muthusamy, "AI-Powered Threat Detection in Cybersecurity Infrastructures," *International Journal of Artificial Intelligence, Data Science and Machine Learning*, vol. 6, no. 1, pp. 23–30, 2025.
- [8] H. Chen, Z. Shen, and Y. Wang, "Threat Detection Driven by Artificial Intelligence: Enhancing Cybersecurity with Machine Learning Algorithms," *World Journal of Innovation and Modern Technology*, vol. 7, no. 6, 2024.
- [9] P. Jain, "Cybersecurity Threat Detection with AI: A Deep Learning Approach to Anomaly Detection," *British Journal of Multidisciplinary Research*, 2023.
- [10] K. C. Chaganti, "AI-Powered Threat Detection: Enhancing Cybersecurity with Machine Learning," *International Journal of Science and Engineering*, vol. 9, no. 4, 2023.