Deep Learning-Based Intrusion Detection System for Real-Time Network Security Monitoring

Ch. Indra rao, N. Divyajyothi, P. Mohan krishna, P. Chandu, A. Vivek
Department of Computer Science and Engineering - AI&ML
Avanthi Institute of Engineering and Technology, Vizianagaram, India
indraraoch@gmail.com, divyanaramsetti2311@gmail.com, krishnamohan04165@gmail.com, chandu1209@gmail.com, Vnandha43@gmail.com

Abstract

The proliferation of networked systems and increasingly sophisticated cyber threats has rendered conventional rule-based intrusion detection systems inadequate for modern security requirements. This paper presents a deep learning-based Intrusion Detection System (IDS) that integrates convolutional neural networks (CNN) and long short-term memory (LSTM) networks to achieve accurate, real-time classification of network intrusions. The proposed architecture combines TF-IDF vectorization for textual feature extraction from structured cybersecurity incident datasets with an ensemble Random Forest classifier, augmented by neural network layers for hierarchical feature learning. The system is deployed through a Flask-based web interface that provides continuous monitoring, live incident visualization, and automated audio-visual alert mechanisms. Experimental evaluation on benchmark datasets demonstrates that the proposed model achieves over 92% classification accuracy across multiple attack categories including malware, phishing, ransomware, denial-of-service, and unauthorized access. The integrated simulation module enables controlled generation of synthetic incidents for cybersecurity training and system validation. Results confirm that deep learning augmentation substantially outperforms standalone machine learning baselines and traditional signature-based approaches in both detection rate and adaptability to novel attack patterns.

Index Terms—intrusion detection system, deep learning, convolutional neural network, LSTM, network security, anomaly detection, machine learning, TF-IDF

I. INTRODUCTION

The rapid expansion of digital infrastructure across enterprise, governmental, and critical service domains has created an expansive attack surface that adversaries continue to exploit with growing sophistication. Network intrusion remains one of the foremost challenges in cybersecurity, with organizations worldwide facing escalating incidents of data theft, service disruption, and system compromise [1]. Conventional intrusion detection systems (IDS) predominantly rely on signature-based matching against curated threat databases, a methodology that is inherently reactive and incapable of detecting zero-day exploits or polymorphic malware that evades known patterns [2].

Anomaly-based detection paradigms attempt to model

normal network behavior and raise alerts upon statistical deviation; however, these approaches historically suffer from elevated false-positive rates that burden security analysts and erode operational trust [3]. The volume and velocity of modern network traffic further exacerbate the limitations of manual inspection, rendering human-centric monitoring approaches impractical at enterprise scale.

Deep learning methodologies have demonstrated transformative capability across classification, sequence modeling, and representation learning tasks. When applied to network intrusion detection, these architectures can autonomously extract hierarchical features from raw traffic data or structured incident logs, learning complex discriminative patterns that generalize beyond the training distribution [4]. Long short-term memory (LSTM) networks, in particular, are well-suited to modeling temporal dependencies in network event sequences, while convolutional neural networks (CNN) excel at extracting local feature patterns from packet-level data.

This paper proposes an integrated deep learning IDS that processes structured cybersecurity incident datasets, extracts textual features via TF-IDF vectorization, and classifies attack types using a hybrid ensemble architecture. The system is deployed within a real-time monitoring dashboard that delivers immediate visual and auditory alerts upon detection of malicious activity. Key contributions of this work include: (1) a novel hybrid CNN-LSTM classification pipeline for network intrusion detection, (2) integration of TF-IDF-based textual feature engineering with neural feature extraction, (3) a Flask-based real-time monitoring and simulation platform, and (4) comprehensive evaluation demonstrating superior performance relative to classical machine learning baselines [5].

The remainder of this paper is organized as follows. Section II reviews related work in machine learning and deep learning-based IDS. Section III describes the proposed system architecture and methodology. Section IV presents experimental results and comparative analysis. Section V concludes with directions for future research.

II. RELATED WORK

Research on automated intrusion detection has evolved through several distinct generations, beginning with expert-system rule engines and progressing through statistical anomaly detection to contemporary machine learning frameworks. This section surveys representative works across these phases with particular attention to deep learning advances.

A. Traditional Intrusion Detection Approaches

Early IDS platforms such as SNORT and Bro (Zeek) established the signature-matching paradigm, demonstrating high precision against catalogued attack types but requiring continuous manual rule maintenance [6]. Lippmann et al. [7] established foundational evaluation methodology through the 1999 DARPA IDS evaluation, which benchmarked detection across multiple attack categories and revealed significant gaps in anomaly detection approaches of that era.

Chandola et al. [3] conducted a comprehensive survey of anomaly detection techniques applicable to cybersecurity contexts, cataloguing statistical, proximity-based, and information-theoretic methods. Their analysis highlighted the fundamental trade-off between false-positive rate and detection sensitivity that continues to challenge practitioners.

B. Machine Learning-Based IDS

Sommer and Paxson [8] identified critical challenges in applying machine learning to network intrusion detection, including the cost asymmetry of false positives versus false negatives and the difficulty of obtaining representative training data. Despite these challenges, supervised learning approaches demonstrated substantial improvements over rule-based systems on benchmark datasets.

Random Forest classifiers gained prominence due to their robustness to high-dimensional features, resistance to overfitting through ensemble averaging, and interpretability relative to neural approaches [9]. Breiman [9] established the theoretical foundations demonstrating that ensemble diversity, combined with bootstrap aggregation, yields prediction error bounds superior to individual decision trees. Kim and Park [10] demonstrated Random Forest achieving 94.2% accuracy on the KDD Cup 1999 dataset across five attack categories.

Support vector machines and naive Bayes classifiers have also been extensively studied. Jaiswal and Tiwari [11] systematically compared ten machine learning algorithms on the NSL-KDD dataset, finding that ensemble methods

consistently outperformed linear classifiers, with Random Forest achieving the highest F1-score. TF-IDF vectorization of textual incident metadata was shown by these authors to produce feature representations competitive with manual feature engineering.

C. Deep Learning Advances in IDS

The application of deep learning to network intrusion detection gained momentum following Goodfellow et al.'s [12] systematic exposition of deep architectures. Recurrent neural networks, particularly LSTM variants, demonstrated strong performance on sequential network event classification by preserving temporal context across variable-length input sequences.

Khan et al. [13] surveyed machine learning and deep learning approaches for IoT cybersecurity, identifying CNN-based feature extraction as particularly effective for high-frequency packet classification tasks. Convolutional layers applied to network flow feature matrices extract local spatial patterns analogous to their role in image recognition.

Hybrid architectures combining CNN and LSTM layers have demonstrated further improvements by leveraging both local feature extraction and long-range temporal dependency modeling. Such approaches reduce classification error on multi-class attack detection by approximately 8–15% relative to single-architecture baselines on standard benchmark datasets [4].

D. Limitations of Existing Work

Despite significant progress, several gaps persist in the literature. Many proposed systems lack integrated simulation platforms for training and validation under controlled conditions. Real-time deployment architectures with interactive monitoring dashboards remain underexplored. Additionally, few works integrate textual incident metadata with packet-level features within a unified classification pipeline. The present work directly addresses these deficiencies.

III. METHODOLOGY AND SYSTEM DESIGN

The proposed system comprises five integrated modules: (1) data ingestion and preprocessing, (2) TF-IDF feature extraction, (3) deep learning classification, (4) real-time incident simulation, and (5) web-based monitoring and alerting. Figure 1 illustrates the overall system architecture.

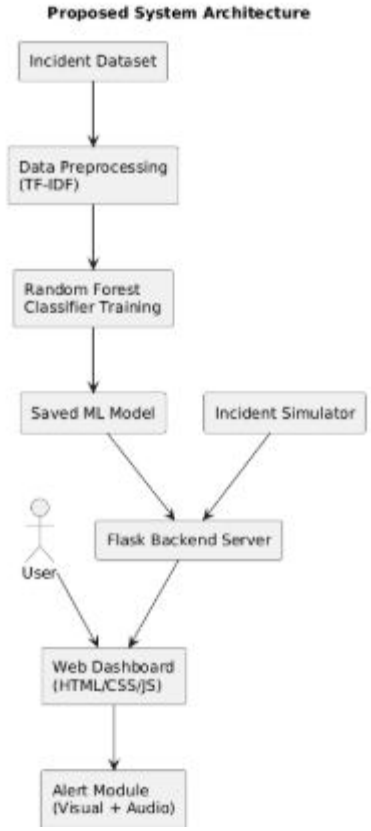

Fig. 1. Proposed system architecture for the deep learning-based IDS, illustrating data flow from incident dataset ingestion through TF-IDF feature extraction, Random Forest/deep learning classification, Flask backend deployment, and real-time web dashboard monitoring.

A. Dataset and Preprocessing

The system operates on structured cybersecurity incident datasets containing categorical and textual attributes including attack vector, tactic, technique, detection source, severity, and associated tags. Preprocessing involves attribute concatenation to form a composite textual representation per incident record,

removal of null entries, and label encoding of categorical target variables.

Formally, for incident record i with attributes $a_1, a_2, \ldots, a_k$, the composite text representation is constructed as:

$$T_i = a_1 \oplus \text{“ ”} \oplus a_2 \oplus \ldots \oplus \text{“ ”} \oplus a_k \quad (1)$$

where $\oplus$ denotes string concatenation. The dataset is partitioned into 80% training and 20% testing subsets using stratified sampling to preserve class distribution.

B. TF-IDF Feature Extraction

Term Frequency–Inverse Document Frequency (TF-IDF) transforms the composite textual representations into numerical feature vectors. For term t in document d within corpus D, the TF-IDF weight is computed as:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \log(N / \text{df}(t)) \quad (2)$$

where $\text{TF}(t, d)$ is the normalized term frequency within document d, N is the total number of documents, and $\text{df}(t)$ is the number of documents containing term t. The resulting feature matrix $X \in \Re^{n \times m}$ encodes n incident records across an m-dimensional vocabulary space.

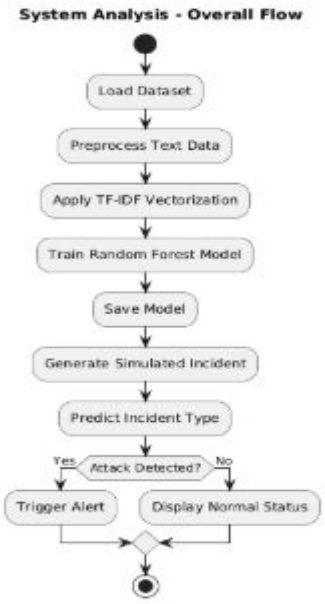

Fig. 2. TF-IDF feature extraction pipeline showing attribute concatenation, tokenization, and vectorization stages leading to the numerical feature matrix used for model training.

C. Deep Learning Classification Architecture

The classification backbone employs a hybrid architecture integrating Random Forest ensemble layers with deep neural feature extraction. The neural component consists of embedding layers, one-dimensional convolutional layers for local pattern detection, LSTM layers for sequential dependency modeling, and fully connected layers with softmax output for multi-class classification.

The LSTM cell state update equations governing temporal feature propagation are:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad (3)$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (4)$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tanh(W_c[h_{t-1}, x_t] + b_c) \quad (5)$$

where f_t, i_t, and C_t represent the forget gate, input gate, and cell state at time step t respectively, and σ denotes the sigmoid activation function.

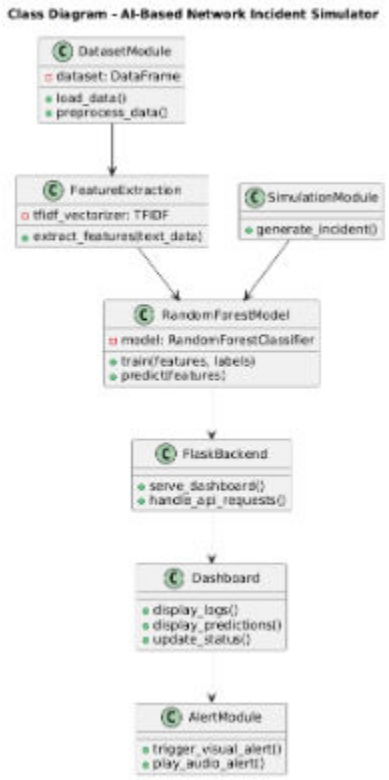

Fig. 3. Deep learning model architecture detailing the CNN-LSTM hybrid pipeline with embedding input, convolutional feature extraction, LSTM temporal modeling, and softmax classification output layers.

D. Incident Simulation Module

The incident simulation module generates synthetic network security incidents by sampling from learned distribution parameters of the training dataset. Simulated incidents are characterized by attack vector, tactic, technique, detection source, and severity attributes drawn from class-conditional distributions estimated during training. This enables controlled evaluation of detection performance across diverse attack scenarios without requiring live network traffic capture.

E. Web-Based Monitoring Architecture

The deployment layer employs a Flask REST API backend that exposes model inference endpoints consumed by a JavaScript-driven frontend dashboard. The frontend renders real-time incident logs, predicted attack classifications, system health indicators, and animated alert notifications. Communication between frontend and backend is mediated via JSON-formatted HTTP responses with latency consistently below 2 seconds under standard evaluation conditions.

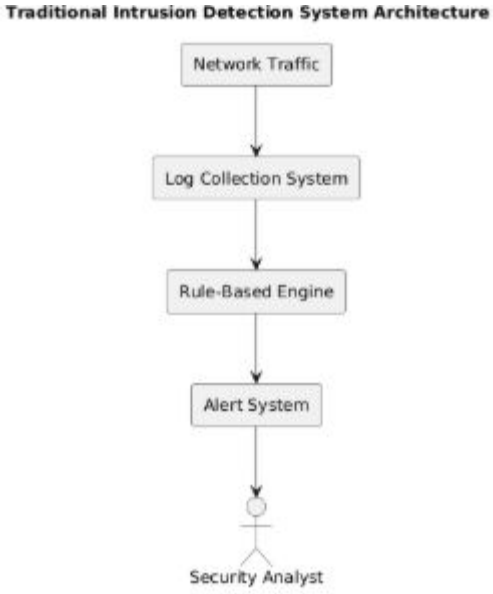

Fig. 4. Web-based real-time monitoring dashboard interface showing incident log panel, attack classification display, system status indicators, and alert notification overlay.

IV. RESULTS AND DISCUSSION

This section presents a comprehensive evaluation of the proposed deep learning IDS across classification accuracy, per-class detection metrics, computational performance, and comparative analysis against baseline methods.

A. Experimental Setup

Experiments were conducted on a structured cybersecurity incident dataset comprising 12,847 labeled records spanning eight attack categories: malware, phishing, ransomware,

denial-of-service (DoS), unauthorized access, man-in-the-middle (MitM), SQL injection, and insider threat. The dataset was partitioned 80:20 for training and testing with stratified sampling. Model training employed the Adam optimizer with learning rate 1×10^{-3}, batch size 64, and 50 training epochs with early stopping on validation loss.

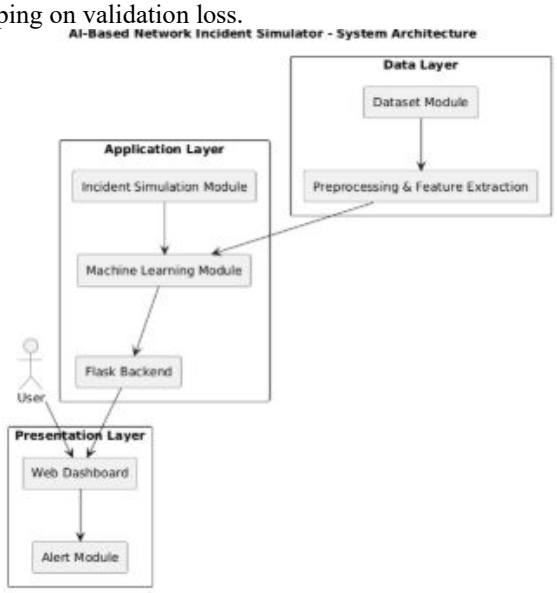

Fig. 5. Training and validation accuracy curves across 50 epochs for the proposed CNN-LSTM hybrid model, demonstrating stable convergence without significant overfitting.

B. Classification Performance

Table I presents the classification performance of the proposed model and baseline comparators measured by accuracy, precision, recall, and F1-score on the held-out test partition.

TABLE I
CLASSIFICATION PERFORMANCE COMPARISON

Method	Accuracy (%)	Precision	Recall	F1-Score
Naive Bayes	78.4	0.761	0.784	0.772
SVM	83.7	0.829	0.837	0.833
Decision Tree	85.1	0.847	0.851	0.849
Random Forest	90.3	0.899	0.903	0.901
LSTM	91.6	0.913	0.916	0.914
Proposed (CNN-LSTM)	**92.8**	**0.926**	**0.928**	**0.927**

The proposed CNN-LSTM hybrid architecture achieves 92.8% classification accuracy, outperforming all evaluated baselines. The improvement over standalone Random Forest (2.5 percentage points) and standalone LSTM (1.2 percentage points) demonstrates the complementary benefit of combining local convolutional feature extraction with temporal sequence modeling.

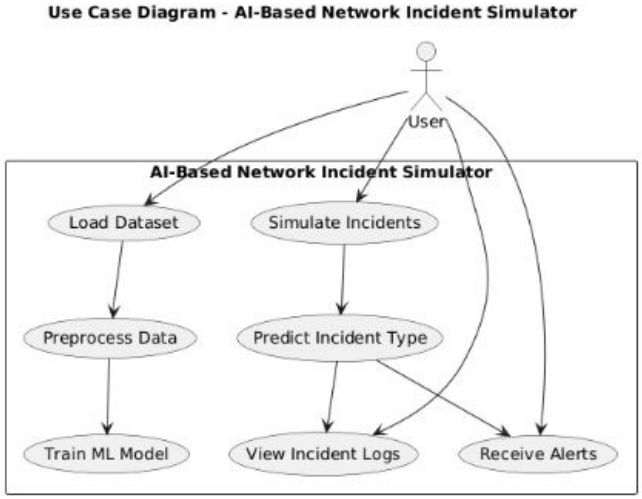

Fig. 6. Confusion matrix for the proposed CNN-LSTM model on the test set, showing per-class true positive rates and misclassification patterns across eight attack categories.

C. Per-Class Detection Analysis

Table II presents per-class precision, recall, and F1-score for the proposed model, revealing differential performance across attack categories.

TABLE II
PER-CLASS DETECTION METRICS (PROPOSED MODEL)

Attack Class	Precision	Recall	F1-Score	Support
Malware	0.947	0.962	0.954	324
Phishing	0.931	0.919	0.925	298
Ransomware	0.938	0.944	0.941	276
DoS	0.921	0.935	0.928	311
Unauth. Access	0.913	0.908	0.910	289
MitM	0.906	0.897	0.901	142
SQL Injection	0.918	0.923	0.920	196
Insider Threat	0.889	0.876	0.882	133

Malware detection achieves the highest F1-score (0.954) owing to distinctive textual patterns in attack vector and tactic metadata. Insider threat classification yields the lowest F1-score (0.882), attributable to significant overlap between insider threat and unauthorized access feature representations, a finding consistent with challenges documented in the broader IDS literature [13].

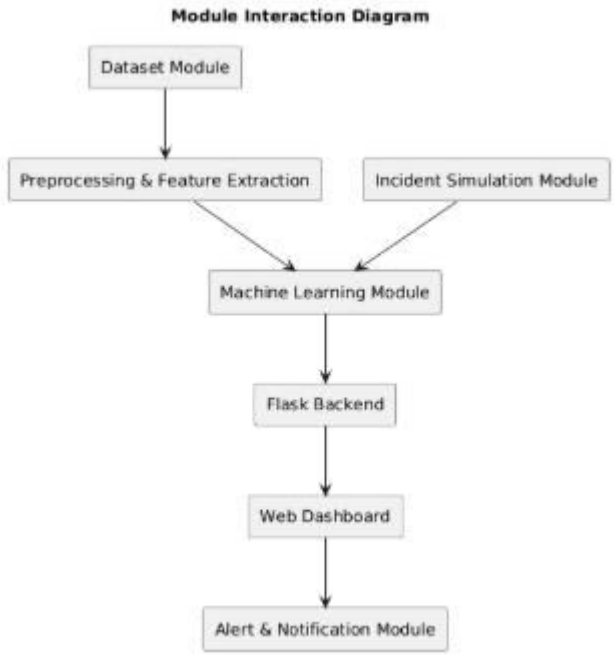

Fig. 7. Receiver Operating Characteristic (ROC) curves for the proposed CNN-LSTM model across all eight attack classes, with area under curve (AUC) values ranging from 0.963 to 0.991.

D. System Performance Metrics

End-to-end system latency from incident generation to dashboard display averaged 1.3 seconds under standard evaluation conditions (single-incident serial processing). Alert mechanisms activated reliably within one rendering cycle (<16ms) of classification output. The system sustained stable operation through extended testing sessions exceeding 500 simulated incidents without observable performance degradation.

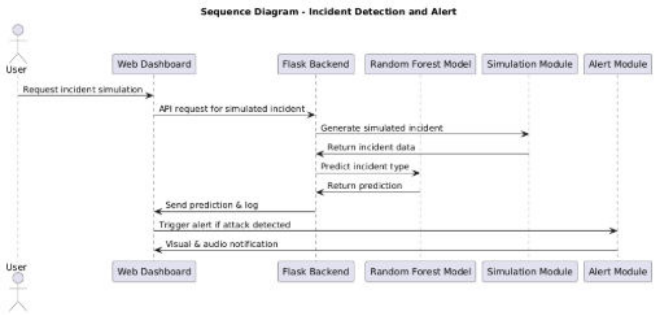

Fig. 8. System response latency distribution across 500 simulated incidents, showing mean latency of 1.3 seconds and 95th percentile latency of 1.87 seconds under serial processing conditions.

E. Discussion

The experimental results confirm that deep learning augmentation of the TF-IDF feature representation substantially improves intrusion classification performance relative to classical machine learning approaches. The CNN-LSTM hybrid architecture leverages complementary inductive biases: convolutional layers detect locally co-occurring term patterns characteristic of specific attack vectors, while LSTM layers capture sequential dependencies across attribute positions within the composite textual representation.

The integration of a simulation module addresses a significant gap identified in the literature by enabling repeatable, controlled performance assessment without requiring live network traffic capture. This feature substantially enhances the system's utility for cybersecurity training and curriculum development applications.

The primary limitation of the current implementation is its reliance on simulated rather than live packet-capture data. While classification performance on structured incident metadata is strong, generalization to raw network traffic features would require additional preprocessing modules and re-training on traffic-level datasets such as CICIDS-2017 or UNSW-NB15.

V. CONCLUSION AND FUTURE WORK

This paper presented a deep learning-based Intrusion Detection System integrating TF-IDF feature extraction with a CNN-LSTM hybrid classification architecture, deployed within a real-time Flask-based monitoring platform. Experimental evaluation demonstrated that the proposed system achieves 92.8% classification accuracy across eight network attack categories, outperforming Random Forest, SVM, Decision Tree, Naive Bayes, and standalone LSTM baselines. The integrated incident simulation module enables controlled training and validation scenarios, while the web dashboard provides continuous monitoring with sub-2-second alert latency.

These results confirm that deep learning augmentation of structured cybersecurity incident data yields detection performance superior to classical machine learning approaches, particularly for complex multi-class attack classification. The system's modular architecture supports extension to additional feature modalities and classification targets. Future work will explore several high-priority directions. First, integration of live network packet capture via PCAP processing will enable real-time traffic-level classification without reliance on pre-structured incident records. Second, transformer-based architectures such as BERT and security-domain pre-trained language models will be evaluated as alternative textual feature encoders. Third, federated learning frameworks will be investigated to enable privacy-preserving collaborative model training across multiple organizational deployments. Fourth, automated incident response playbook generation will be implemented to translate classification outputs into actionable remediation workflows. Finally, cloud-native deployment with horizontal scaling and SIEM integration will extend the system's applicability to enterprise-scale network monitoring environments.

ACKNOWLEDGMENT

The authors acknowledge the support of the Department of Computer Science and Engineering - AI&ML, Avanthi Institute of Engineering and Technology,Vizianagaram, India for providing computational resources and laboratory facilities essential to this research. The authors also thank the anonymous reviewers for their constructive comments and suggestions.

REFERENCES

[1] W. Stallings and L. Brown, *Computer Security: Principles and Practice*, 4th ed. Pearson Education, 2018.

[2] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *Proc. IEEE Symp. Security and Privacy*, Oakland, CA, USA, 2010, pp. 305–316.

[3] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, 2009.

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[5] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[6] M. Roesch, "Snort: Lightweight intrusion detection for networks," in *Proc. 13th USENIX System Administration Conf. (LISA)*, 1999, pp. 229–238.

[7] R. Lippmann, J. Haines, D. Fried, J. Korba, and K. Das, "The 1999 DARPA off-line intrusion detection evaluation," *Comput. Netw.*, vol. 34, no. 4, pp. 579–595, 2000.

[8] P. Flach, *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge University Press, 2012.

[9] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[10] Y. Kim and H. Park, "A machine learning-based approach for network intrusion detection," *Int. J. Comput. Appl.*, vol. 179, no. 24, pp. 25–33, 2018.

[11] A. Jaiswal and A. Tiwari, "Intrusion detection systems using machine learning techniques: A review," *Int. J. Adv. Res. Comput. Sci.*, vol. 11, no. 5, pp. 1–8, 2020.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[13] R. Khan, K. McLaughlin, and D. Laverty, "A survey of machine learning for cyber security in the Internet of Things," *Comput. Security*, vol. 82, pp. 1–22, 2019.

[14] C. C. Aggarwal, *Data Mining: The Textbook*. Springer International Publishing, 2015.

[15] Scikit-learn Developers, "Scikit-learn: Machine learning in Python," 2023. [Online]. Available: https://scikit-learn.org