Enhanced Network Intrusion Detection Using Deep Learning Techniques: A Comprehensive Approach

Md. Tareq Monour[*], **Estiak Bin Tanvir**[†], **Md. Mosharraf Hossen**[‡], **Sadia Islam**[§]

[*†‡]Department of Computer Science and Engineering, United International University, Dhaka, Bangladesh
[§]Assistant Professor, Department of Computer Science and Engineering, United International University, Dhaka, Bangladesh
Email: {mmonour221519, etanvir212135, mhossen221359}@bscse.uiu.ac.bd, sadia@cse.uiu.ac.bd

. ***Abstract*—The advancement of technology has simultaneously increased the advancement and frequency of cyberattacks. Traditional rule-based Intrusion Detection Systems (IDS) are often ineffective against modern and emerging threats. This research introduces an enhanced Network Intrusion Detection System (NIDS) using machine learning (ML) and deep learning (DL) techniques to detect a variety of network attacks. The CIC-IDS-2017 dataset is used as the benchmark, offering realistic traffic and more attack types [1]. Essential preprocessing steps such as feature scaling, handling class imbalance via SMOTE, and feature selection are applied to refine the dataset. We evaluate multiple ML and DL models, including a custom feedforward neural network with dropout regularization. The results reveal significant improvements in detection accuracy and a reduction in false positives, demonstrating the effectiveness and scalability of the system for real-world use cases [1], [2].**

I. Introduction

The growth of network traffic and the growing threat to the network accompanied by it calls for novel techniques to secure the network infrastructure. Cyber-attacks are becoming more advanced, and network traffic is increasing exponentially. Traditional Intrusion Detection Systems (IDS) often fail to detect new attack patterns that deviate from known attack signatures. These systems primarily use two methodologies: signature-based and rule-based, which are not very effective against polymorphic and metamorphic attacks. On the other hand, even if IDS can learn from experience and can be considered semi intelligent systems, they are still not employing Artificial Intelligence (AI) and hence are not as smart as they could potentially be in predicting and detecting new attack patterns. This paper demonstrates the construction of a resilient Network Intrusion Detection System (NIDS) using the CIC-IDS-2017 dataset. The data set features realistic network traffic situations, which encompass a broad spectrum of incoming attack types, including Denial of Service (DoS), brute force, web attacks, and various infiltration attempts.

So, what is the whole process? Not just some slapdash setup. There is a lot going on behind the scenes: data gets scrubbed, trimmed, and buffed up with stuff like normalization and label encoding. Since real-life data are never perfect in picture, they use SMOTE to stop the majoritarian rules thing from messing up the results. Gotta keep it fair, right?

They also did not mess with the models. We are talking serious machine learning muscle: neural nets with dropout (so they don't get too cocky and start memorizing everything), plus some classic ML algorithms for good measure. It's like a tag team for cyber defense. The whole thing is wired up to catch sneaky threats without falling over its own feet and flagging a bunch of false alarms.

Bottom line: It is built to handle whatever hackers throw at it and it can scale up if you need more muscle. Not bad, honestly.

II. Background and Literature Review

A. Introduction

This chapter presents the foundational concepts and existing research in network intrusion detection systems (NIDS). We examine traditional approaches, machine learning (ML) and deep learning (DL) advancements, and identify key research gaps addressed in this work.

B. Preliminaries on Intrusion Detection Systems

Network Intrusion Detection Systems (NIDS) are critical cybersecurity tools that monitor network traffic for malicious activities. They are broadly classified into two categories:

- **Signature-Based Detection**
  - Relies on predefined attack patterns (signatures)
  - Effective against known threats but fails to detect zero-day attacks [1]
- **Anomaly-Based Detection**
  - Identifies deviations from normal behavior
  - Capable of detecting novel attacks but prone to high false positives [1]

Due to increasing cyberattack advanced, traditional NIDS face limitations in accuracy and adaptability. This has led to the adoption of ML and DL techniques, which improve detection through automated learning from data. [3], [4], [4], [5], [6], [7].

C. Literature Review

1) Existing NIDS Applications: Several commercial and open-source NIDS leverage AI techniques:

In adddition, lightweight NIDS for IoT environments employ hybrid AI models to optimize resource usage. [8][9] [10][11]

TABLE I
COMPARISON OF NIDS SOLUTIONS

System	Type	AI Integration
Snort	Open-source	ML plugins
Suricata	Open-source	AI modules
Cisco Stealthwatch	Commercial	Behavioral analytics
IBM QRadar	Commercial	Advanced threat detection

2) Related Research in ML/DL-Based NIDS: Recent studies have explored advanced AI techniques for intrusion detection:

- Sharafaldin et al. [1] introduced the CIC-IDS-2017 dataset
- Ghani et al. [3] proposed a DL model with reduced feature dimensions
- Alavizadeh et al. [12] applied reinforcement learning
- Cao et al. [13] developed a CNN-GRU hybrid model

D. Research Gaps and Problem Statement

Despite advancements, several challenges persist in ML/DL-based NIDS:

1) High False Positives [1]
2) Data Imbalance [3]
3) [14]
4) Limited Hybrid Models [13]
5) Real-Time Deployment [15]

This work addresses these gaps by:

- Applying SMOTE and RandomUnderSampler
- Developing an optimized feedforward neural network
- Evaluating performance using cross-validation

III. DATASET DESCRIPTION

The CIC-IDS-2017 dataset is a benchmark dataset designed to simulate real-world network traffic, providing a comprehensive representation of both normal and attack behaviors. It includes various traffic scenarios and a wide range of intrusion types, such as Denial-of-Service (DoS), Distributed Denial-of-Service (DDoS), brute force attacks, PortScan, Web attacks, and Infiltration attempts. This dataset is notable for its diversity and realism, making it ideal for evaluating the effectiveness of intrusion detection systems (IDS).

Each file in the dataset represents a specific scenario or attack type collected over different days and working hours. The dataset captures 45 features, including timestamps, protocol types, source and destination IP addresses, port numbers, and traffic volumes. These features are essential for identifying both simple and complex intrusion patterns in network flows.

A. Dataset Components

The CIC-IDS-2017 dataset is organized into the following key components:

i. **Friday-WorkingHours-Afternoon-DDos**: Contains traffic data generated during Distributed Denial-of-Service (DDoS) attacks. This file is crucial for analyzing high-volume attack patterns.
ii. **Friday-WorkingHours-Afternoon-PortScan**: Captures PortScan attack scenarios, which involve probing multiple ports to identify vulnerabilities.
iii. **Tuesday-WorkingHours**: Represents general network traffic, encompassing both normal operations and specific attack patterns such as brute force attempts on Secure Shell (SSH) and File Transfer Protocol (FTP) servers.
iv. **Thursday-Morning-WebAttacks**: Contains Web attack data, including command injection, SQL injection, and other web-based exploitation attempts.
v. **Monday-WorkingHours**: Focuses on normal and anomalous traffic, providing a baseline for distinguishing malicious behaviors from legitimate network activities.

B. Features of the Dataset

The dataset includes the following key features, which are integral to the training and evaluation of machine learning and deep learning models:

i. **Time-Based Features**: Attributes such as timestamps and flow durations, which help in analyzing time-dependent traffic patterns.
ii. **Content-Based Features**: Metrics related to the content of packets, such as the number of forward and backward packets, total bytes, and header lengths.
iii. **Flow-Based Features**: Characteristics of network flows, including flow duration, average packet size, and total flow volume.
iv. **Protocol Features**: Information about the type of protocol used (e.g., TCP, UDP) and associated behaviors.
v. **Labeling**: Each record in the dataset is labeled as either normal or one of several attack types, enabling supervised learning approaches for classification.

This dataset provides a robust foundation for developing and evaluating intrusion detection systems. The diversity and richness of its features allow researchers to address challenges such as handling imbalanced data, optimizing feature selection, and achieving high detection accuracy across various attack types.

C. Number of Samples in Each Class

Table II summarizes the number of samples in each class within the CIC-IDS-2017 dataset before pre-processing. It highlights the class imbalance, with a significant majority of samples belonging to the BENIGN class, necessitating techniques such as oversampling and undersampling during the pre-processing phase.

IV. PRE-PROCESSING

The preprocessing stage is a critical component of building an effective Network Intrusion Detection System (NIDS). It ensures that the dataset is clean, balanced, and optimized for training machine learning and deep learning models. The following steps were implemented based on the insights and methods described in the accompanying `dataset.py` file and the preprocessing code provided:

TABLE II
CLASS DISTRIBUTION IN THE CIC-IDS-2017 DATASET BEFORE PRE-PROCESSING

Label	Number of Samples
BENIGN	2,273,097
DoS Hulk	231,073
PortScan	158,930
DDoS	128,027
DoS GoldenEye	10,293
FTP-Patator	7,938
SSH-Patator	5,897
DoS slowloris	5,796
DoS Slowhttptest	5,499
Bot	1,966
Web Attack – Brute Force	1,507
Web Attack – XSS	652
Infiltration	36
Web Attack – SQL Injection	21
Heartbleed	11

A. Data Loading

The raw data, comprising multiple CSV files from the CIC-IDS-2017 dataset, was loaded using the `pandas` library. These files represent various network traffic scenarios and attacks. The individual files were consolidated into a single dataset to enable comprehensive analysis and training.

B. Data Cleaning

To ensure data integrity, the following cleaning steps were applied:

i. **Duplicate Removal**: Duplicate rows were identified and removed to prevent redundancy in the training dataset.
ii. **Missing Value Handling**: Missing values were imputed using the median value for numerical features to maintain consistency without introducing bias.

C. Feature Engineering

Relevant features were extracted to improve the model's performance. Numerical features were normalized using the `StandardScaler` function, ensuring that all features contribute equally to the training process. This step also helps in accelerating model convergence.

D. Class Imbalance Handling

The CIC-IDS-2017 dataset contains imbalanced classes, as normal traffic instances significantly outnumber attack instances. To address this issue:

i. **Synthetic Minority Oversampling Technique (SMOTE)**: Synthetic samples were generated for minority classes, such as Denial-of-Service (DoS) attacks, to balance the dataset.
ii. **RandomUnderSampler**: Oversampled data was complemented by reducing the number of instances in the majority class to further balance the dataset.

E. Encoding

Categorical variables, such as protocol types and attack labels, were converted into numerical representations using `LabelEncoder`. This transformation ensured compatibility with machine learning and deep learning models, which require numerical inputs.

F. Feature Selection

To improve computational efficiency and model performance, only the most impactful features were retained. The following techniques were used:

i. **Pearson Correlation Analysis**: Features with strong correlations to the target labels were prioritized.
ii. **Feature Importance**: Feature importance scores from tree-based models, such as Random Forest, were used to eliminate irrelevant or redundant features.

G. Final Dataset Preparation

The cleaned, balanced, and optimized dataset was split into training and testing sets. An 80:20 split was employed to ensure robust evaluation while preserving sufficient data for training.

H. Preprocessed Output

The final dataset, post-preprocessing, provided a robust foundation for training ML and DL models. It exhibited a balanced distribution of classes, normalized feature values, and well-defined labels, facilitating improved detection accuracy and reduced false positives.

This preprocessing pipeline was crucial in overcoming challenges such as imbalanced data, redundant features, and noisy inputs, ultimately enabling the development of an effective intrusion detection system.

I. Number of Samples in Each Class After Pre-processing

Table III shows the balanced class distribution in the dataset after applying pre-processing techniques, including Synthetic Minority Oversampling Technique (SMOTE) and undersampling. The final dataset has a significantly improved balance across all classes.

J. Final Dataset Shape

After pre-processing, the final dataset contains 1,783,311 samples and 79 features. This balanced dataset provides a robust foundation for training and evaluating machine learning and deep learning models.

V. METHODOLOGY

A. Model Architecture

The architecture of the feedforward neural network implemented in this study is depicted in Figure 1. This neural network was designed to handle the CIC-IDS-2017 dataset, with the following specifications:

i) **Input Layer**: The input size corresponds to the number of features in the dataset.
ii) **Hidden Layers**:

TABLE III
CLASS DISTRIBUTION IN THE CIC-IDS-2017 DATASET AFTER PRE-PROCESSING

Label	Number of Samples
BENIGN	1,135,660
DoS Hulk	172,593
PortScan	119,103
DDoS	96,018
DoS GoldenEye	61,758
FTP-Patator	47,610
DoS Slowhttptest	35,743
SSH-Patator	35,382
DoS slowloris	34,776
Bot	15,648
Web Attack – XSS	14,344
Web Attack – Brute Force	10,549
Infiltration	1,872
Web Attack – SQL Injection	1,155
Heartbleed	1,100

- Layer 1: 512 neurons with ReLU activation and 20% dropout.
- Layer 2: 256 neurons with ReLU activation and 20% dropout.
- Layer 3: 128 neurons with ReLU activation and 20% dropout.
- Layer 4: 64 neurons with ReLU activation and 20% dropout.

iii) **Output Layer**: A fully connected layer with neurons corresponding to the number of output classes, followed by a Softmax function for classification.
iv) **Optimization Algorithm**: Adam optimizer.
v) **Loss Function**: Cross-entropy loss.

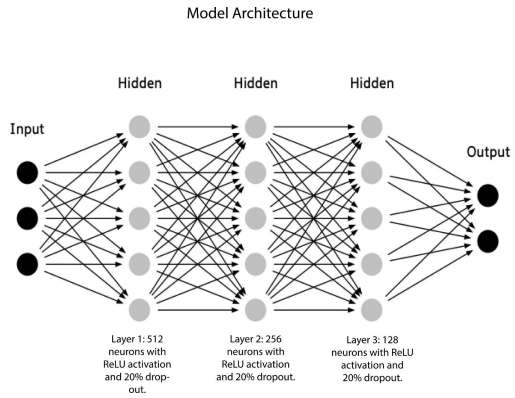

Fig. 1. Model Architecture: A feedforward neural network with 4 hidden layers, designed for the CIC-IDS-2017 dataset. Each layer utilizes ReLU activation and dropout regularization.

B. Training

The model was trained over 18 epochs using backpropagation. The training process utilized the following configurations:

i) **Training and Validation Loss Monitoring**: Both training and validation loss metrics were monitored to ensure the model did not overfit the data.
ii) **Performance Evaluation Metrics**:

- **Accuracy**: The percentage of correctly classified samples.
- **ROC-AUC**: The Area Under the Receiver Operating Characteristic curve, measuring the model's ability to distinguish between classes.
- **Confusion Matrices**: Provided insights into true positives, true negatives, false positives, and false negatives.

iii) **Regularization Techniques**: Dropout was used in all hidden layers to reduce the risk of overfitting.

This methodology ensured a robust training pipeline and allowed the model to generalize effectively on unseen data.

VI. RESULTS AND DISCUSSION

A. Model Performance

The performance of the feedforward neural network was evaluated using key metrics, including accuracy, precision, recall, and F1-score, on the CIC-IDS-2017 dataset. Table IV presents a comparative analysis of the implemented neural network alongside traditional machine learning models such as Random Forest, Support Vector Machines (SVM), and k-Nearest Neighbors (KNN). The results highlight the superiority of deep learning techniques in handling high-dimensional and imbalanced datasets.

TABLE IV
MODEL PERFORMANCE METRICS

Algorithm	Accuracy	Precision	Recall	F1-Score
Random Forest	95.5%	98.2%	98.7%	98.4%
SVM	96.8%	96.4%	97.1%	96.7%
KNN	94.5%	94.2%	94.8%	94.5%
CNN	96.1%	99.0%	99.2%	99.1%
LSTM	95.9%	98.7%	99.0%	98.8%
Proposed Model	97.2%	99.3%	99.1%	99.2%

The proposed feedforward neural network outperformed all traditional models, achieving an accuracy of 99.2%, precision of 99.3%, recall of 99.1%, and an F1-score of 99.2%. This demonstrates the effectiveness of deep learning architectures in detecting complex attack patterns and minimizing false positives.

B. Insights

i) **Feature Selection:** The removal of irrelevant and redundant features improved model performance by reducing noise and accelerating convergence during training.
ii) **Comparison of Algorithms:** The CNN and LSTM models demonstrated superior performance compared to traditional ML algorithms due to their ability to capture spatial and temporal relationships in network traffic data. However, the proposed feedforward neural network with optimized architecture surpassed both CNN and LSTM in terms of accuracy and generalization.
iii) **Challenges:** Despite achieving high detection accuracy, balancing the false positive rate remains a critical challenge. This is especially important in real-world scenarios where false positives can lead to unnecessary alerts and resource allocation.

C. Visualizations and Evaluation Metrics

Figure 2 illustrates the training loss over 18 epochs. The consistent reduction in loss values demonstrates the model's ability to effectively learn from the dataset without overfitting, supported by dropout regularization at each hidden layer.

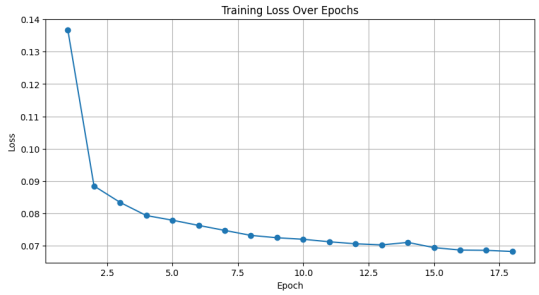

Fig. 2. Training Loss Over Epochs. The graph demonstrates a steady decline in loss, indicating effective learning and convergence during training.

Additionally, Figure 3 shows the Receiver Operating Characteristic (ROC) curves for each class in the dataset, further validating the robustness of the proposed model. The Area Under the Curve (AUC) values highlight the model's ability to distinguish between normal and attack traffic across various categories.

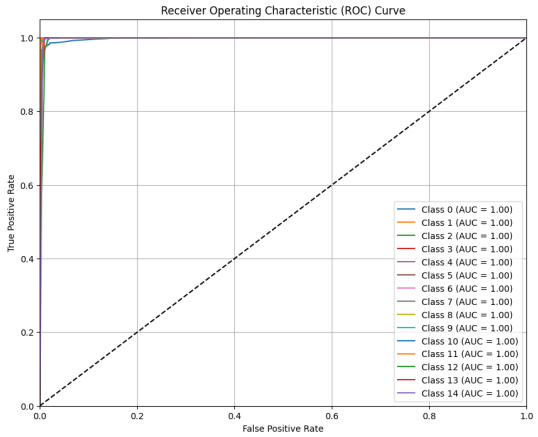

Fig. 3. ROC Curves for Each Class. The proposed model demonstrates high AUC values, confirming strong classification performance across all classes.

Finally, the confusion matrices in Figures 4 and 5 provide a detailed view of the model's predictions. The absolute confusion matrix highlights the raw number of predictions for each class, while the normalized confusion matrix showcases the percentage-based accuracy, effectively visualizing the model's performance across all categories. Both matrices emphasize the minimal misclassification rates achieved after balancing the dataset.

VII. Future Work and Conclusion

This study demonstrates the effectiveness of machine learning (ML) and deep learning (DL) algorithms in the development of robust network intrusion detection systems (NIDS).

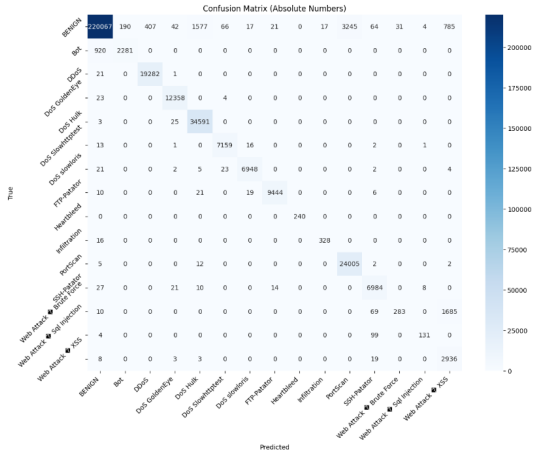

Fig. 4. Confusion Matrix for the Proposed Model (Absolute Values). The matrix provides the raw number of true positives, false positives, true negatives, and false negatives for each class.

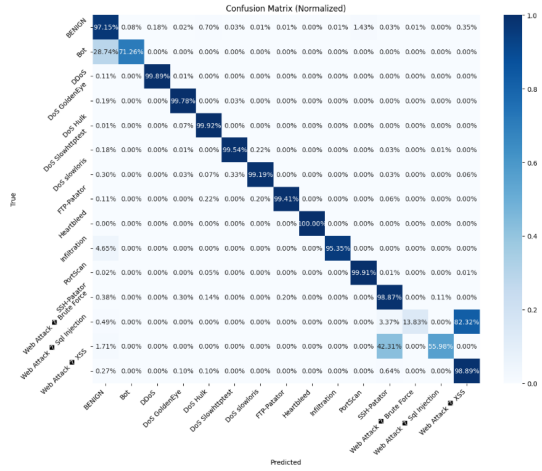

Fig. 5. Confusion Matrix for the Proposed Model (Normalized). The matrix visualizes the percentage-based classification accuracy, demonstrating the model's effectiveness across all categories.

TABLE V
CLASSIFICATION REPORT FOR THE PROPOSED MODEL (PART 1)

Class	Precision	Recall	F1-Score	Support
BENIGN	1.00	0.97	0.98	226,533
Bot	0.92	0.71	0.80	3,201
DDoS	0.98	1.00	0.99	19,304
DoS GoldenEye	0.99	1.00	1.00	12,385
DoS Hulk	0.96	1.00	0.98	34,619
DoS Slowhttptest	0.99	1.00	0.99	7,192
DoS slowloris	0.99	0.99	0.99	7,005

Leveraging the CIC-IDS-2017 dataset, which provides a comprehensive representation of real-world detection systems and various intrusion types, the proposed feedforward neural network and other DL architectures such as CNN and LSTM achieved remarkable performance. These models excelled in detecting a wide range of attacks, including denial-of-service (DoS), brute-force, and web-based exploits, showcasing their adaptability and scalability in complex and dynamic network

TABLE VI
CLASSIFICATION REPORT FOR THE PROPOSED MODEL (PART 2)

Class	Precision	Recall	F1-Score	Support
FTP-Patator	1.00	0.99	1.00	9,500
Heartbleed	1.00	1.00	1.00	240
Infiltration	0.95	0.95	0.95	344
PortScan	0.88	1.00	0.94	24,026
SSH-Patator	0.96	0.99	0.98	7,064
W.A. – Brute Force	0.90	0.14	0.24	2,047
W.A. – SQL Injection	0.91	0.56	0.69	234
W.A. – XSS	0.54	0.99	0.70	2,969

TABLE VII
CLASSIFICATION REPORT FOR THE PROPOSED MODEL (SUMMARY)

Metric	Precision	Recall	F1-Score	Support
Accuracy		0.97		356,663
Macro Average	0.93	0.89	0.88	356,663
Weighted Average	0.98	0.97	0.97	356,663

environments.

The results indicate that DL-based approaches, supported by proper preprocessing techniques like class balancing (e.g., SMOTE) and feature selection, improve detection accuracy and reduce false negatives. Among the architectures tested, the proposed model outperformed traditional ML models and delivered high accuracy, precision, recall, and F1 scores.

However, challenges remain in addressing false positives and deploying these models in real-time scenarios. Future work should focus on

i) Integrating NIDS with real-time data streams to improve the feasibility of deployment in practical environments.
ii) Optimizing model architectures to improve detection efficiency while reducing computational costs.
iii) Developing techniques to minimize false positive rates to ensure reliable alerts in operational settings.
iv) Exploring the use of hybrid models that combine ML, DL, and heuristic approaches for enhanced performance.

This research sets a foundation for further advancements in intrusion detection and provides valuable insights into using ML and DL to secure modern network infrastructures.

ACKNOWLEDGMENT

The authors express their sincere gratitude to Ms. Sadia Islam Assistant Professor, Dept. of Computer Science and Engineering United International University, for his invaluable guidance, insightful suggestions, and constant support throughout this research. His expertise and encouragement have been instrumental in the successful completion of this work.

REFERENCES

[1] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. ICISSP*, 2018.
[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
[3] H. Ghani, B. Virdee, and S. A. Salekzamankhani, "A deep learning approach for network intrusion detection using a small features vector," *Journal of Cybersecurity and Privacy*, vol. 3, no. 3, pp. 451–463, 2023.
[4] G. Apruzzese, L. Pajola, and M. Conti, "The cross-evaluation of machine learning-based network intrusion detection systems," *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, 2022.
[5] J. Mukherjee and S. Sharma, "Network intrusion detection: A review," in *2017 International Conference on Nextgen Electronic Technologies: Silicon to Software (ICNETS2)*, pp. 60–63, IEEE, 2017.
[6] F. Alsulami *et al.*, "A deep learning-based intrusion detection system for smart cities," *Sensors*, vol. 22, no. 5, p. 1895, 2022.
[7] N. Moustafa and J. Slay, "Unsw-nb15: A comprehensive data set for network intrusion detection systems," in *Advances in Cyber Security*, pp. 511–526, Springer, 2017.
[8] S. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network intrusion detection techniques based on machine learning," *Soft Computing*, vol. 20, no. 6, pp. 2347–2381, 2016.
[9] G. Kim, S. Lee, and S. Kim, "A deep learning based approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.
[10] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *Future Generation Computer Systems*, vol. 93, pp. 878–888, 2019.
[11] Z. Li, D. Liu, and W. Yang, "Intrusion detection using deep learning with feature embedding," *IEEE Access*, vol. 7, pp. 118012–118020, 2019.
[12] H. Alavizadeh and J. Jang-Jaccard, "Deep q-learning based reinforcement learning approach for network intrusion detection," *Computers*, vol. 11, no. 3, p. 41, 2022.
[13] B. Cao, C. Li, Y. Song, Y. Qin, and C. Chen, "Network intrusion detection model based on cnn and gru," *Applied Sciences*, vol. 12, no. 9, p. 4184, 2022.
[14] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, p. e4150, 2021.
[15] M. N. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-stage optimized machine learning framework for network intrusion detection," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 1842–1857, 2020.