International Journal of Computer Applications (0975 - 8887)
Volume 186 - No.66, February 2025

An Enhanced Anomaly Detection in Networked Systems through Deep Learning Model

Chaitali V. Chaudhary
Departement of Computer Science and Engineering
PES University, Bengaluru, India

S. Vanitha
Departement of Computer Science and Engineering
PES University, Bengaluru, India

ABSTRACT

In the rapidly evolving digital landscape, the proliferation of interconnected devices and networks has introduced unprecedented security challenges. As cyber threats evolve in complexity there is a pressing need for robust intrusion detection systems (IDS) capable of safeguarding against a wide range of attacks. This paper explores the efficacy of utilizing deep learning techniques, specifically a multi-scale convolutional neural network (M-CNN) for detecting network intrusions using the CSE-CIC-IDS2018[9] dataset. The study focuses on meticulous data preprocessing techniques to enhance model performance and presents a streamlined approach for intrusion detection. Through comprehensive experimentation and evaluation, the proposed M-CNN model demonstrates high accuracy, precision, recall, and F1-score for detecting various types of network intrusions comapred to other studies, highlighting its effectiveness in mitigating cyber threats in modern networks.

Keywords

Anomaly detection, CSE-CIC-IDS2018, Deep learning, M-CNN,NIDS , Network Security

1. INTRODUCTION

In today's digital age, where technologies like IoT, cloud computing, and 5G are advancing rapidly, the internet has become an integral part of our lives. However, with this widespread connectivity comes an increasing threat from hackers. [7]Acc to CrowdStrike 2024 Global Threat Report, a significant increase in interactive intrusions, up by 60% in 2023. A key trend is the shift towards malware-free attacks, which now constitute 75% of all detections. Attackers are increasingly using stolen credentials and exploiting trusted relationships to gain access to systems, which poses new challenges for detection mechanisms. The average breakout time, the period between the initial compromise and lateral movement, decreased from 84 minutes in 2022 to 62 minutes in 2023. As we move forward, the risk of attacks on automated transactions looms large. Thus, it's crucial to bolster our network security defenses to safeguard against potential disasters in an environment where the digital world and cyber threats are evolving simultaneously.
Firewalls serve as the first line of defense, but for a more comprehensive approach, intrusions detection system (IDS) are essential. IDS are typically classified into Host-based Intrusion Detection Systems and Network-based Intrusion Detection Systems. HIDS focuses on individual hosts or endpoints, while NIDS monitors network traffic for any irregularities, acting as a second layer of defense, while. Anomaly-based intrusion detection, a subset of IDS, identifies deviations from normal behavior patterns, enabling the detection of previously unknown attacks.
Deep learning, with its sophisticated neural network architecture capable of autonomous feature learning and processing, presents an effective solution for handling the massive influx of data generated by modern networks. By designing neural networks with appropriate configurations, we can extract important features and make informed judgments on vast datasets. Numerous studies have shown the promise of deep learning in detecting network attacks, making it a suitable choice for implementing IDS.
However, the effectiveness of deep learning hinges on the quality of the datasets used for training. Traditional datasets like KDD-99[6] and NSL-KDD [4], while once reliable, are now outdated and inadequate for capturing the complexities of modern cyber threats. In contrast, the [9] dataset, which originates from actual network traffic, offers a more accurate reflection of current attack trends, providing a solid foundation for experimentation in intrusion detection.
This study focuses on using the [9] dataset for detecting network intrusions, with particular attention given to data preprocessing to enhance the quality of the input data. Instead of exploring multiple models, we streamline our approach by employing a multi-scale convolutional neural network (CNN) to detect network attacks. Our aim is to simplify the model while ensuring robust detection capabilities. Through multi-class classification task, we seek to differentiate between benign network traffic and malicious intrusions.

2. RELATED WORK

A review of previous IDS approaches, including traditional machine learning and deep learning methods, is conducted. The study compares hybrid CNN-LSTM, statistical models, and other contemporary techniques, highlighting the advancements made by our proposed method. In [19] Pakanzad and Monkaresi present a hybrid CNN-LSTM approach for enhancing Intrusion Detection System (IDS) performance. Their method achieves high classification accuracies of 98.1% on the NSL-KDD [4] dataset and 96.7% on the CICIDS2017 dataset in multiclass scenarios. Additionally, the study emphasizes the method's focus on multiclass classification, surpassing previous works that primarily focused on binary classification or specific attack classes. In [13] Karatas, Demir, and Sahingoz address the challenges of intrusion detection systems (IDSs) in detecting sophisticated and evolving network attacks. Their paper proposes and evaluates six machine learning-based IDSs on the [9]dataset. The Synthetic Minority Oversampling Technique (SMOTE) is utilized to reduce dataset imbalance, enhancing detection rates for rare intrusions. [15] Authors introduce a CNN-based intrusion detection model focused on identifying denial of service (DoS) attacks, achieving high accuracy rates surpassing RNN models.

In [22], Ratti, Nandi, and Singh introduce an unsupervised intrusion detection system that operates using a discrete-time sliding window approach, leveraging statistical feature distances to detect attacks. Their method, evaluated on the CICIDS-2018 dataset for FTP Brute Force and HTTP Distributed Denial of Service attacks, employs clustering with the DBSCAN algorithm and threshold selection for detection In [18] The Authors introduce novel methods based on convolutional neural networks (CNNs), specifically U-Net and TCN, for network attack classification. Evaluations on [6] and [9] datasets reveal promising results, with TCN-LSTM achieving 92% and 97% accuracy on KDD99 and CSE-CIC-IDS2018 respectively, while U-Net achieves 93% and 94% accuracy on the same datasets. This study underscores the importance of using modern datasets for training to avoid overfitting and achieve higher accuracy in real-world anomaly detection applications.

In [11], The Authors introduce a novel anomaly detection-based Network Intrusion Detection System (NIDS) leveraging deep neural networks (DNNs) in software-defined networking (SDN) environments. Employing the NSL-KDD dataset, a simple DNN architecture with two hidden layers and dropout regularization achieves a promising accuracy rate of 92.65%. In [14], Kavitha and Amutha explore deep intrusion detection system (IDS) approaches, emphasizing on deep learning algorithms for enhanced accuracy and precision in both binary and multi-class classification on the NSL-KDD dataset. Their results indicate that RNN achieves 99.4% accuracy in attack type classification, CNN-LSTM achieves 95.4%, and DNN achieves 91.8%. In [17], Lu discusses how artificial intelligence (AI), particularly neural networks, is revolutionizing network intrusion detection by offering improved detection efficiency and accuracy. While AI enhances audit, monitoring, and risk management capabilities, balancing rapid feedback with user privacy remains a challenge. This necessitates ongoing development to achieve optimal privacy protection and processing efficiency.

In [16], Kisanga et al. present a novel approach called Activity and Event Network (AEN), leveraging graph models to address both high-volume attacks and persistent threats. The suggested supervised Graph Convolutional Network model, based on AEN, achieves promising results on DDoS and TOR-nonTOR datasets, with accuracies of 76% and 88% respectively. In [12], The Authors introduce a deep learning system leveraging CNN and biLSTM trained on the NSL-KDD dataset, their proposed model achieves high detection rates and low false positives. While some challenges remain, particularly with U2R and Worms attacks, the proposed system outperforms many existing NIDS models in terms of false positive rate, detection rate, and accuracy. In [24], Tapu et al. address the security concerns associated with wireless networks by proposing a novel hybrid meta deep learning approach for intrusion detection. Combining Siamese and Prototypical networks based on meta learning techniques, the model achieves reliable detection with minimal data requirements. With just 3000 data samples, both binary and multi-class classifications attain over 90% accuracy.

Amol and Vikram[25] propose a deep CNN model for identifying and classifying network intrusions in cloud environment,They employ random forest algorithm to select optimal features. Experimentation on [9] datasets demonstrates the CNN model's testing accuracy of 97.07%. The study highlights the effectiveness of the CNN model in complex dataset analysis, facilitating real-time monitoring and response.

3. METHODOLOGY

Our network intrusion detection model's methodology is illustrated in Figure1. The diagram comprises two primary sections: data preprocessing and the training and evaluation phase. Prior to model training, a thorough understanding of the network data in the [9] dataset and its characteristics was essential.

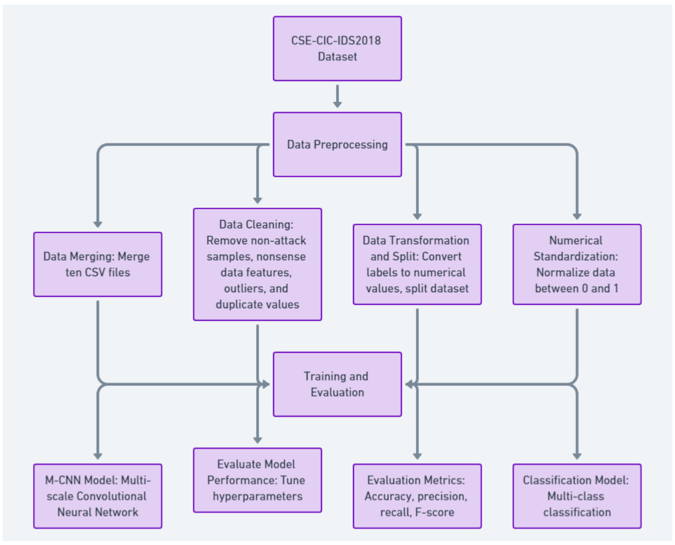

CSE-CIC-IDS2018 Dataset
Data Preprocessing
Data Merging: Merge ten CSV files
Data Cleaning: Remove non-attack samples, nonsense data features, outliers, and duplicate values
Data Transformation and Split: Convert labels to numerical values, split dataset
Numerical Standardization: Normalize data between 0 and 1
Training and Evaluation
M-CNN Model: Multi-scale Convolutional Neural Network
Evaluate Model Performance: Tune hyperparameters
Evaluation Metrics: Accuracy, precision, recall, F-score
Classification Model: Multi-class classification

Fig. 1. **Methodology**

3.1 CSE-CIC-IDS2018 Dataset

In our experiment, we utilized the [9] dataset. Developed collaboratively by the CSE[10] and the CIC[8], with support from AWS[23], this dataset stands out as one of the most extensive publicly accessible intrusion detection datasets.

The dataset spans ten days, capturing both benign and malicious network traffic in CSV format. It comprises ten files totaling 6.41 GB, containing 16,233,002 records. Due to the large volume of data and the existence of redundant entries, it is not pre-divided into training and testing sets. Previous research have employed various methods to handle this, such as randomly selecting subsets of data for experiments. In this study, we utilized the entire dataset for evaluation, which includes 83 features representing different network traffic characteristics.

Each record in the CSE-CIC-IDS2018 dataset is labeled as either benign or an attack type. The attack types are further categorized into six groups, encompassing 14 distinct attacks, as detailed in Table 1.

3.2 Data Preprocessing

Given the extensive size of the dataset, it was crucial to preprocess the data effectively to ensure the model could accurately detect various intrusion attacks. The preprocessing steps included data merging, cleaning, transformation and splitting, and numerical normalization.

Table 1. List of Attacks

Attack Category	Attack Names
DDos	DDoS-LOIC-UDP,DDoS-HOIC,DDoS-LOIC-HTTP
Bruteforce	SSH-Bruteforce,FTP-Bruteforce
Dos	DoS-Slowloris,DoS-SlowHTTPTest,DoS-Hulk,DoS-GoldenEye
Web Attack	Brute Force-XSS, Brute Force-Web, SQL Injection
Infiltration	Infiltration
Botnet	Bot

3.2.1 Data Merging. The dataset consists of ten CSV files, each containing different types of attack traffic along with benign traffic. These files were merged into a single dataset to streamline the preprocessing and analysis steps.

3.2.2 Data Cleaning. The data cleaning process for enhancing the quality and relevance of the dataset involved several key steps. Firstly, all data not classified as either benign or attack was removed to ensure focus on relevant instances. Following this, irrelevant features such as Timestamp, Flow ID, Source IP, Source Port, Dst IP, and Dst Port were excluded to streamline the dataset. Outliers were identified and appropriately managed, with NaN values filled using the mode of the respective features to prevent dataset distortion. Additionally, redundant records were eliminated to reduce the dataset's size and improve processing efficiency. These meticulous steps led to a significant reduction in the dataset, from 16,233,002 to 11763712 records, thereby enhancing its suitability for subsequent model training and analysis.

3.2.3 Data Transformation and Split. The data transformation involved converting categorical labels into numerical values to facilitate model training. This was done using one-hot encoding for multi-class classification, where benign traffic was labeled as 0, and various attack types were assigned unique integers. The dataset was divided into training set and testing set using 80-20 split. This division ensures the model's generalization capability and allows for proper evaluation of its performance.

3.2.4 Numerical Normalization. To ensure uniformity and improve the model's performance, numerical normalization was applied. Each feature was normalized to have a mean of 0 and a standard deviation of 1. The Standardization process facilitates the model's ability to learn patterns and relationships between features, as it won't be disproportionately influenced by features with larger scales. This can ultimately lead to better model performance and more reliable predictions.

3.3 M-CNN Model

Before feeding the data into the M-CNN, Principal Component Analysis (PCA) was applied for dimensionality reduction. PCA helps in transforming the high-dimensional data into a lower-dimensional space, retaining the most important features and reducing computational complexity. This step is crucial for handling the large feature set and enhancing the training efficiency of the model. Our model employs a multi-scale convolutional neural network (M-CNN) designed for efficient feature extraction and classification. The architecture integrates multiple convolutional layers with varying kernel sizes to capture diverse patterns in the data. The detailed architecture as shown in Figure 2 highlights the sequential arrangement of layers, their configurations, and the flow of data through the model for network intrusion detection.
In the M-CNN architecture, the convolutional layers are pivotal for processing input data across various scales, capturing both local and global features. The convolutional layer applies a set of learnable filters to the input, performing the convolution operation. The output of the convolutional layer is given by:

$$y_{ij}^{l} = f(\sum_{m}\sum_{n} w_{mn}^{l} \cdot x_{i+m,j+n}^{l-1} + b^{l}) \tag{1}$$

where:

—y_{ij}^l is the output of the current layer at position (i, j).
—$x_{i+m,j+n}^{l-1}$ is the input from the previous layer.
—w_{mn}^l is the learnable filter/kernel of size (m, n).
—b^l is bias term.
—f is activation function, ReLU defined as:

$$f(x) = max(0, x) \tag{2}$$

To ensure stable and expedited training, batch normalization is implemented, guaranteeing uniform data distribution across layers. Batch normalization is a technique used to normalize the inputs to a layer, helping to improve the training process.
The first M-Module begins with Conv1D layers employing 64 filters each, with kernel sizes of 1 and 3, activated by ReLU functions. Batch normalization follows each convolutional layer. Subsequently, a MaxPooling1D layer with a pool size of 3, strides of 2, and padding set to 'same' is applied. The formula for max pooling is:

$$y_{ij}^{l} = \max_{(m,n)\in R_{ij}} x_{i+m,j+n}^{l-1} \tag{3}$$

where:

—y_{ij}^l is the output of the current layer at position (i, j).
—$x_{i+m,j+n}^{l-1}$ is the input from the previous layer.
—R_{ij} is the rectangular region over which the maximum is computed.

The second M-Module mirrors the structure of the first, with Conv1D layers now utilizing 128 filters, maintaining the same kernel sizes and activation functions. In the final stages, fully connected layers aggregate extracted features, culminating in a Softmax layer for classification. This setup enables the model to produce a probability distribution across classes, facilitating precise detection of network intrusions. The process involves GlobalAveragePooling1D for feature aggregation, where the formula is:

$$y^{l} = \frac{1}{W \times H}\sum_{i=1}^{W}\sum_{j=1}^{H} x_{ij}^{l-1} \tag{4}$$

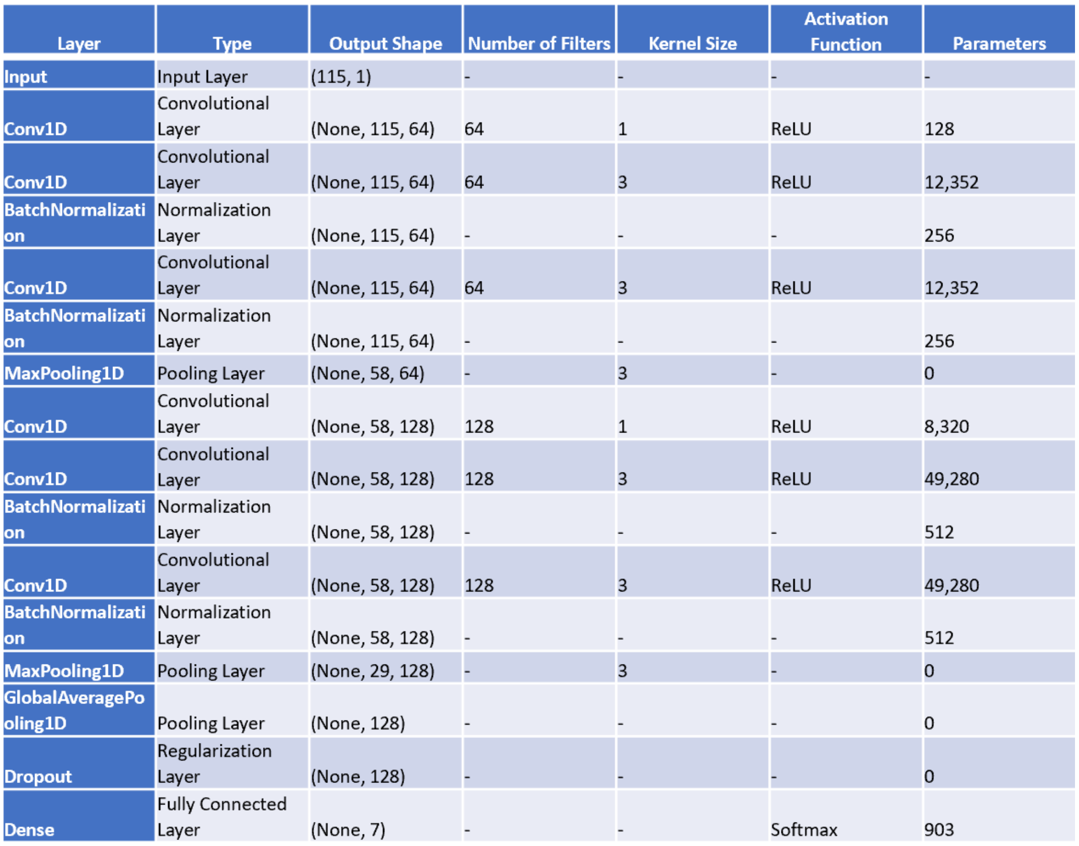
where:

—y^l is output of current layer.
—x_{ij}^{l-1} is input from previous layer.
—W and H are width and height of the input, respectively.

Next, a Dropout layer with a dropout rate of 0.5 is applied to reduce overfitting, and concludes with a Dense layer with units equal to number of classes and activated by the Softmax function, defined as:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{5}$$

where:



Layer	Type	Output Shape	Number of Filters	Kernel Size	Activation Function	Parameters
Input	Input Layer	(115, 1)	-	-	-	-
Conv1D	Convolutional Layer	(None, 115, 64)	64	1	ReLU	128
Conv1D	Convolutional Layer	(None, 115, 64)	64	3	ReLU	12,352
BatchNormalization	Normalization Layer	(None, 115, 64)	-	-	-	256
Conv1D	Convolutional Layer	(None, 115, 64)	64	3	ReLU	12,352
BatchNormalization	Normalization Layer	(None, 115, 64)	-	-	-	256
MaxPooling1D	Pooling Layer	(None, 58, 64)	-	3	-	0
Conv1D	Convolutional Layer	(None, 58, 128)	128	1	ReLU	8,320
Conv1D	Convolutional Layer	(None, 58, 128)	128	3	ReLU	49,280
BatchNormalization	Normalization Layer	(None, 58, 128)	-	-	-	512
Conv1D	Convolutional Layer	(None, 58, 128)	128	3	ReLU	49,280
BatchNormalization	Normalization Layer	(None, 58, 128)	-	-	-	512
MaxPooling1D	Pooling Layer	(None, 29, 128)	-	3	-	0
GlobalAveragePooling1D	Pooling Layer	(None, 128)	-	-	-	0
Dropout	Regularization Layer	(None, 128)	-	-	-	0
Dense	Fully Connected Layer	(None, 7)	-	-	Softmax	903

Fig. 2. **M-CNN**

—z_i is input to the softmax function for class i.

—K is number of classes.

—$\sigma(z)_i$ is the output probability for class i.

In the final stages, fully connected layers aggregate extracted features, culminating in a Softmax layer for classification. This setup enables the model to produce a probability distribution across classes, facilitating precise detection of network intrusions. The process involves GlobalAveragePooling1D for feature aggregation,The model is then supplemented by a Dropout layer featuring a 0.5 dropout rate to address overfitting, followed by a Dense layer. This Dense layer is configured with units corresponding to the number of classes and is activated using Softmax.

4. EXPERIMENT RESULTS AND EVALUATION METRICS

In this section, we provide a thorough analysis of the experimental results achieved using the proposed Multiscale Convolutional Neural Network (M-CNN) model for network intrusion detection. The experimental environment's detailed specifications included an Intel Core CPU, NVIDIA GeForce RTX 4090, and a substantial memory and storage capacity, as outlined in Table 2. Python [21] was the development language, with Pandas[2], NumPy[1], and scikit-learn used for efficient data processing and analysis. The TensorFlow [5] framework, including its Keras API, was employed for building and training the model. We evaluate the model's performance using various evaluation metrics [20] and provide a detailed interpretation of the results. These metrics are defined as follows:

—**Accuracy**: Accuracy measures the proportion of correctly classified instances, encompassing both true positives and true negatives, relative to total number of instances. It is computed as[3] :

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

—**Precision**: Precision measures the proportion of true positives (TP) relative to the sum of true positives and false positives (FP). It reflects the model's capability to minimize false alarms. It is calculated as [3] :

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

—**Recall**: Recall, also known as true positive rate, is the ratio of true positives to the sum of true positives and false negatives (FN), indicating the ability of the model to detect all positive instances. It is calculated as [3] :

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

—**F1-score**: The F1-score represents the harmonic mean of precision and recall, offering a balanced evaluation that incorporates

Table 2. Experiment Environment

Hardware	Properties
CPU	13th Gen Intel(R) Core(TM) i9-13900H
OS	Windows 11
GPU	NVIDIA GeForce RTX 4090
Memory	128GB
Disk	1TB

Table 3. Classification report for the proposed model

Class	Precision	Recall	F1-Score
Benign	0.99	1.00	0.99
Botnet	1.00	1.00	1.00
BruteForce	1.00	1.00	1.00
DDOS	1.00	1.00	1.00
Dos	1.00	1.00	1.00
Infilteration	0.51	0.03	0.06
Web Attack	0.91	0.65	0.76

both metrics. It is computed as [3] :

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{9}$$

From the classification report in Table3, we can observe that the model achieves high precision and recall for most classes, indicating its effectiveness in detecting various types of network intrusions. However, the performance for the 'Infilteration' class is relatively lower, with a precision of 0.51 and a recall of 0.03. The model achieves an accuracy rate of **99%**, demonstrating its capability to correctly classify a significant majority of instances in the test set

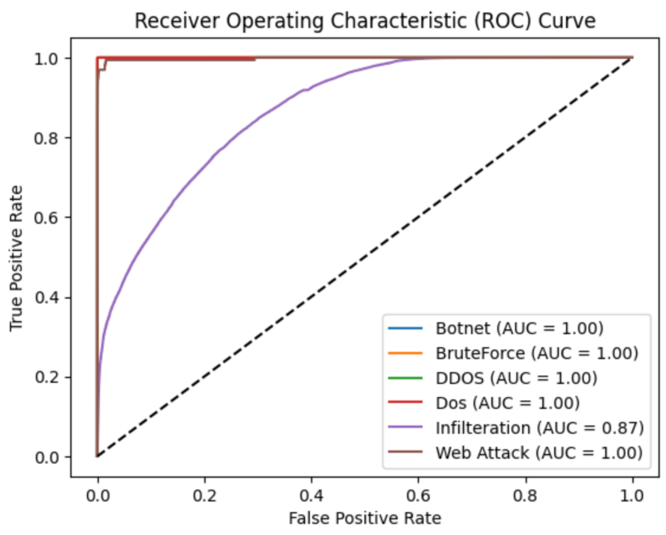


Fig. 3. **Receiver Operating Characteristic Curve**

The ROC curve in Figure 3 illustrates the performance of the M-CNN model in distinguishing between classes. Key observations reveal excellent performance, with an AUC of 1.0 for the 'Botnet', 'BruteForce', 'DDOS', and 'Dos' classes, indicating perfect classification. Additionally, the model demonstrates good performance, with an AUC of 1.0, for the 'Web Attack' class, suggesting effective detection of web attacks. However, the model exhibits relatively lower performance, with an AUC of 0.87, for the 'Infilteration' class, indicating challenges in accurately detecting infiltration attacks. The ROC curve and AUC values provide insights into the model's class discrimination ability, with higher AUC values representing better classification performance. Compared with state-of-the-art methods, the M-CNN model outperforms in terms of classification accuracy and false positive reduction. The model struggles with infiltration attack detection due to limited training instances. Future iterations will address this by augmenting dataset diversity and applying adversarial training.

5. CONCLUSION AND FUTURE WORK

In conclusion, this research underscores the significance of leveraging deep learning methodologies, particularly M-CNN, for network intrusion detection in the face of escalating cyber threats. By utilizing the rich and diverse CSE-CIC-IDS2018 dataset and employing meticulous data preprocessing steps, we have successfully developed a robust intrusion detection model. The model exhibits exceptional performance in distinguishing between benign traffic and malicious intrusions, achieving high precision and recall across multiple attack categories. Future work includes refining feature selection, enhancing infiltration attack detection, and integrating real-time adaptability. Nevertheless, The results of this study contribute to the progress and development of the field of network security and provide valuable insights for developing more resilient intrusion detection systems capable of protecting against evolving cyber threats in today's digital landscape.

6. REFERENCES

[1] Numpy, the fundamental package for scientific computing with python. 2022.

[2] Pandas, a fast, powerful, flexible, and easy-to-use open-source data analysis and data manipulation library for python. 2022.

[3] Confusion matrix, accuracy, precision, recall, f1 score. `https://medium.com/analytics-vidhya/confusion-matrix-accuracy-precision-recall-f1-score-ad`, 2023. Accessed: 2024-06-20.

[4] Datasets for data mining, 2023. Accessed: 2024-06-20.

[5] Tensorflow. `https://www.tensorflow.org/`, 2023. Accessed: 2024-06-20.

[6] KDD Cup 1999. Kdd cup 1999 data. `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`, 1999. Accessed: 2024-06-19.

[7] CrowdStrike. Crowdstrike 2024 global threat report. 2024.

[8] Communications Security Establishment. Cse. `https://www.cse-cst.gc.ca/en`, 2024.

[9] Canadian Institute for Cybersecurity. Cicids 2018 dataset. `https://www.unb.ca/cic/datasets/ids-2018.html`, 2018.

[10] Canadian Institute for Cybersecurity. Canadian institute for cybersecurity. `https://www.unb.ca/cic/`, 2024.

[11] R. Gupta, Z. C. Kumar, and V. N. Patil. Design of deep neural network based anomaly detection system. In *2022 IEEE 7th International conference for Convergence in Technology (I2CT)*, pages 1–5, Mumbai, India, 2022.

[12] S. S. Kanumalli, L. K, R. A, S. P, and T. M. A scalable network intrusion detection system using bi-lstm and cnn. In *2023 Third International Conference on Artificial Intelligence and Smart Energy (ICAIS)*, pages 1–6, Coimbatore, India, 2023.

[13] G. Karatas, O. Demir, and O. K. Sahingoz. Increasing the performance of machine learning-based idss on an imbalanced and up-to-date dataset. *IEEE Access*, 8:32150–32162, 2020.

[14] R. Kavitha and S. Amutha. Performance analysis of deep neural network and lstm models for secure network intrusion detection system. In *2022 IEEE 4th International Conference on Cybernetics, Cognition and Machine Learning Applications (ICCCMLA)*, pages 390–396, Goa, India, 2022.

[15] J. Kim, J. Kim, H. Kim, M. Shim, and E. Choi. Cnn-based network intrusion detection against denial-of-service attacks. *Electronics*, 9(6):916, 2020.

[16] P. Kisanga, I. Woungang, I. Traore, and G. H. S. Carvalho. Network anomaly detection using a graph neural network. In *2023 International Conference on Computing, Networking and Communications (ICNC)*, pages 61–65, Honolulu, HI, USA, 2023.

[17] C. Lu. Research on the technical application of artificial intelligence in network intrusion detection system. In *2022 International Conference on Electronics and Devices, Computational Science (ICEDCS)*, pages 109–112, Marseille, France, 2022.

[18] A. Mezina, R. Burget, and C. M. Travieso-González. Network anomaly detection with temporal convolutional network and u-net model. *IEEE Access*, 9:143608–143622, 2021.

[19] S. N. Pakanzad and H. Monkaresi. Providing a hybrid approach for detecting malicious traffic on the computer networks using convolutional neural networks. In *2020 28th Iranian Conference on Electrical Engineering (ICEE)*, pages 1–6, Tabriz, Iran, 2020.

[20] David M. W. Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation, 2020.

[21] Python Software Foundation. Python language reference, version 3.6. 2022.

[22] R. Ratti, S. Nandi, and S. R. Singh. Online network attack detection using statistical features. In *2021 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, pages 125–130, Hyderabad, India, 2021.

[23] Amazon Web Services. Cicids 2018 dataset on aws. `https://registry.opendata.aws/cse-cic-ids2018/`, 2024.

[24] S. U. Tapu, S. A. A. Shopnil, R. B. Tamanna, M. A. A. Dewan, and M. G. R. Alam. Malicious data classification in packet data network through hybrid meta deep learning. *IEEE Access*, 11:140609–140625, 2023.

[25] Amol D. Vibhute and Vikram Nakum. Deep learning-based network anomaly detection and classification in an imbalanced cloud environment. *Procedia Computer Science*, 232:1636–1645, 2024.