International Journal of Informatics and Computation (IJICOM)
Vol. 6, No.1, August 2024
ISSN: 2685-8711, E-ISSN: 2714-5263 DOI: 10.35842/ijicom

Establishing CNN for Network Intrusion Detection: A Comparative Approach

M. Hizbul Wathan[1]*, Muhamad Aziz[2]

Abstract
Intrusion detection plays an important role in protecting systems from various threats. However, as intrusion techniques become more sophisticated, traditional detection methods have shown limitations in identifying new attacks. This research addresses the pressing issue of improving intrusion detection by utilizing Convolutional Neural Networks (CNN) algorithms, compared to various other machine learning techniques such as Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Gaussian Naive Bayes (GNB), Decision Trees, and Gradient Boosting (GBoost). The main objective is to evaluate and compare the performance of these algorithms using a comprehensive dataset sourced from Kaggle, which includes 25,192 data and 42 features. Using metrics such as accuracy, precision, recall, and F1-score, the results show a complex pattern in the strengths and weaknesses of each. Surprisingly, CNN achieved exceptional accuracy, raising questions that require further investigation. Notably, KNN stands out as the best-performing machine learning algorithm. Contextualized within existing research, this study advances the understanding of the role of machine learning in intrusion detection, providing valuable insights for practical implementation. The findings reinforce the relevance of adapting to the evolving network threat landscape while raising interesting questions for future research. In conclusion, this research provides a comparative analysis of intrusion detection techniques, offering a basis for making informed decisions to improve network security and mitigate evolving threats.

Keywords:
Intrusion Detection, CNN, DL Algorithms, Network Security, Performance Evaluation

1. Introduction

Intrusion detection plays a very important role in computer network environments as intrusion attacks can have a serious impact on system security and data integrity [1], [2]. Intrusion attacks such as hacking, malware, and denial of service (DoS) have evolved into more sophisticated forms and continue to evolve which are not only capable of disrupting the normal operation of systems but can also steal sensitive

information or damage network infrastructure [3]. The potential impacts include theft of confidential data, fraud, service outages, and significant financial losses [4], [5]. Therefore, intrusion detection is crucial to detect attacks quickly, respond efficiently, and prevent potential losses that could threaten business continuity and user privacy [6], [7].

Today, in the face of increasingly complex and evolving intrusion attacks, traditional detection methods may no longer be adequate. Modern attacks tend to use variations and sophisticated techniques to evade detection based on recognized attack signatures [8]. Static detection methods based on previously recognized attack signatures may fail to identify new attacks that have not been detected before. In addition, attacks increasingly

M. Hizbul Wathan (hizbulti@gmail.com)

use disguise techniques and behavioral changes that are difficult to capture by static rule-based approaches. Therefore, learning and adaptable approaches such as machine learning and deep learning are becoming more relevant in the face of increasingly complex and dynamic attack diversity [9].

Machine learning and deep learning techniques, especially CNN, play a very important role in improving intrusion detection [10], [11]. CNNs have a remarkable ability to extract complex and abstract features from network traffic data that are difficult to identify by traditional detection methods. The advantage of this approach lies in its ability to be first developed for image processing but is now able to automatically extract relevant features from raw data and cope with variations and scales in network data. The technique is also capable of learning from diverse data, enabling adaptation to new attacks, and improving detection accuracy in networked environments [12].

Assessing how CNN performs in comparison to alternative machine learning methods like SVM, KNN, GNB, Decision Tree, and GBoost has important benefits in the context of intrusion detection research [13]. This comparison makes it possible to identify the most effective techniques for addressing the challenges of intrusion detection in network traffic. Different techniques have different approaches to handling data and understanding attack patterns. By comparing the performance, the research can reveal the advantages and limitations of each method in detecting diverse and complex intrusion attacks. The results of this comparison can provide valuable insights for the selection of the best method for building reliable and effective intrusion detection solutions in the real world [14].

Hence, the primary aim of this study is to evaluate and contrast the effectiveness of CNN against various other machine learning methodologies, including SVM, KNN, GNB, Decision Tree, and GBoost. This research aims to measure the effectiveness of each method in detecting intrusion attacks on network traffic. This study will utilize a Kaggle dataset to evaluate performance indicators including accuracy, precision, recall, and F1-score. The results of this comparison are expected to provide valuable insights for choosing the best method to improve computer network security and protect against various attacks.

2. Related Works

Several previous studies have applied CNN in intrusion detection on network traffic. For example, research by Liu and Wang [15] proposed a novel online anomaly detection system based on software-defined networks that uses CNN to extract the original features of network flows for analysis. Other research such as Alabsi and Friends [16], proposed an approach that uses a combination of two CNN models to detect attacks on IoT networks with high accuracy.

Other intrusion detection research using Naive Bayes Classifier with Feature Reduction Classifier conducted by Mukherjee [17] provides better performance for designing efficient and effective IDS for network intrusion detection. Other research on network intrusion detection systems conducted by Wisanwanichthan and Thammawichai [18] is trying to combine two algorithms, namely Naive Bayes and SVM called the Double-Layered Hybrid Approach (DLHA). DLHA layer 1 uses Naïve Bayes Classifier and layer 2 uses SVM. DLHA shows good performance in detecting attacks with an R2L detection rate of 96.67% and U2R of 100%.

Another research is about intrusion detection systems using XGBoost conducted by Dhaliwal and his friends [19]. XGBoost was applied to the NSL-KDD dataset to obtain the desired modeling. This model produces an accuracy of 98.70%. As well as 98.41% precision, 99.11% recall and 98.76% f1-score.

Li Wenchao and friends [20] conducted a study that introduced a novel intrusion detection system employing the KNN classification algorithm within wireless sensor networks. The system's objective is to distinguish abnormal nodes from their normal counterparts by monitoring their anomalous conduct. As a result, the system can detect and prevent attacks in wireless sensor networks, which can seriously affect the network flow.

Earlier research has involved the evaluation of machine learning methods in intrusion detection by utilizing performance metrics like accuracy, precision, recall, and F1-score. Son and colleagues [21] conducted a comparative analysis of the effectiveness of SVM, KNN, Random Forest, and Decision Tree algorithms. in detecting DoS/DDoS attacks and found that Decision Tree had the lowest performance.

The advantages of using CNNs in intrusion detection include their ability to recognize complex and abstract patterns in traffic data and adapt to changes in attacks. However, the disadvantages of CNN can include high computational requirements and complexity in model parameter optimization [22]. SVMs have advantages in tackling high-dimensional data problems and non-linear classification such as in network traffic. However, SVM tends to be less efficient for large datasets [23]. KNN is a simple and versatile algorithm, but it can be sensitive to network noise and metric choice, which can affect its performance [24]. GNB and Decision Tree can handle diverse features but may have limitations in handling highly complex data [25], [26]. GBoost is a powerful technique in ensemble learning but requires careful parameter tuning [27].

Previous studies generally conclude that the strengths of each machine learning technique in intrusion detection are its ability to handle specific cases and its limitations are sensitivity to parameters and data size. In choosing the most suitable technique, it is important to consider the nature of the traffic data, scale, and available resources.

3. Proposed Method

A. Mathematical Concept

The proposed model is a neural network based on CNN architecture for internet network intrusion detection. The following is the mathematical concept. The Conv1D layer, also known as the first Convolutional layer, operates on one-dimensional time-series data, specifically tailored to a dataset of length “padding_value” In this particular layer, there are 128 filters utilized, each with a kernel size of 3 for scanning the input data. The employed activation function in this context is ReLU (Rectified Linear Unit). It imparts non-linearity to the model by converting negative values to zero and preserving positive values unaltered. Conv1D Layer 1 is shown in Equation 1.

$$Conv1D_1(x) = ReLU(W_1 * x + b_1) \quad (1)$$

In Equation 1, Conv1D Layer 1 uses the symbol x to denote the input to this specific layer. Additionally, it involves the weight matrix represented by W and the bias vector denoted as b. Incorporating a MaxPooling1D layer, denoted as Pooling Layer 1, this component performs maximum pooling on the data, specifically designed for one-dimensional datasets. It employs a pool size of 2, effectively reducing the dimensionality of the input data by selecting the maximum value within each pair of adjacent elements. This pooling operation aids in capturing the most significant features while reducing computational complexity, contributing to the model's ability to focus on essential patterns within the data. MaxPooling1D Layer 1 is shown in Equation 2.

$$MaxPooling1D_1(x) = max(stride(x)) \quad (2)$$

MaxPooling1D Layer 1 in Equation 2, $stride(x)$ represents the maximum pooling operation. The Dropout Layer 1 is a crucial element within the neural network architecture, primarily aimed at mitigating overfitting, a common challenge in machine learning. In this layer, a dropout rate of 30% is applied to the units from the preceding

layer. This means that during training, 30% of the units are randomly deactivated, effectively preventing the model from becoming overly reliant on specific connections and features. By doing so, Dropout Layer 1 encourages the network to generalize better and enhance its ability to perform well on unseen data, thereby improving the model's overall robustness and preventing it from memorizing the training data.

The Conv1D Layer 2, also known as the second Convolutional Layer, plays a significant role in the neural network architecture. It is equipped with 256 filters, each with a kernel size of 3, and employs the ReLU activation function. This layer is responsible for further extracting intricate patterns and features from the input data. The 256 filters act as feature detectors, scanning the data for relevant patterns. The utilization of the ReLU activation function brings non-linear characteristics to the network, which enhances its capacity to discern intricate connections within the data. This layer's output contributes to the network's ability to discern and understand the subtle details in the input, making it a crucial component in the process of detecting intrusion patterns in network traffic data. Conv1D layer 2 is shown in Equation 3.

$$Conv1D_2(x) = ReLU(W_2 * x + b_2) \quad (3)$$

The MaxPooling1D Layer 2, followed by Dropout Layer 2, constitutes a crucial part of the network architecture. In the MaxPooling1D Layer 2, maximum pooling is implemented using a pool size of 2, effectively diminishing the spatial dimensions of the data while preserving the most crucial information. This layer helps in subsampling the features extracted by the previous convolutional layers, focusing on the most relevant details while reducing computational complexity. Subsequently, Dropout Layer 2 randomly drops 30% of the units from the previous layer during training, acting as a regularization technique to prevent overfitting. This dropout mechanism enhances the network's generalization capabilities and ensures that it doesn't rely too heavily on any particular set of neurons. Together, these layers contribute to the network's ability to learn and represent intricate patterns in network traffic data efficiently, aiding in the detection of intrusion patterns.

The Conv1D Layer 3 is a pivotal component of the network architecture, characterized by its substantial number of filters, totaling 512. Employing a kernel size of 3, this layer conducts convolutional operations on the input data, focusing on capturing intricate patterns and features. The activation function used here is ReLU, which introduces non-linearity to the model by transforming negative values to zero and allowing the network to learn complex relationships within the data. This layer plays a crucial role in extracting high-level abstractions from the input, contributing significantly to the network's ability to discern subtle intrusion patterns in network traffic data. Conv1D Layer 3 is shown in Equation 4.

$$Conv1D_3(x) = ReLU(W_3 * x + b_3) \quad (4)$$

The GlobalAveragePooling1D Layer serves as an integral part of the network architecture by executing a global average pooling operation across all the features or time steps. This operation involves computing the average of all the values in the input data, and condensing the information while retaining essential insights. It plays a pivotal role in reducing the spatial dimensions of the data, ensuring that the subsequent layers receive a compact representation of the features, which can aid in efficient and effective pattern recognition. This layer's ability to distill complex information into more manageable forms contributes to the overall success of the Convolutional Neural Network in detecting intricate intrusion patterns within network traffic data. GlobalAveragePooling1D layer is shown in Equation 5.

$$GlobalAveragePooling1D(x) = \frac{1}{N}\sum_{i=1}^{N} x_i \quad (5)$$

The Dense Layer 1, also referred to as the Fully Connected Layer 1, plays a significant role in the network architecture with a total of 128 units or neurons. These units are densely connected to the previous layer, creating a fully connected structure. The activation function employed here is ReLU, which introduces non-linearity to the network by allowing only positive values to pass through while filtering out negative values. This non-linearity is crucial for the model's capacity to learn complex relationships within the data. Dense layers are essential for the final stages of feature extraction and abstraction, enabling the network to capture high-level patterns and representations from the previously processed features. These layers are a key component of the Convolutional Neural Network and contribute significantly to its overall performance in network intrusion detection. Dense Layer 1 is shown in Equation 6.

$$Dense_1(x) = ReLU(W_4 \cdot x + b_4) \quad (6)$$

The Dropout Layer 3 is incorporated into the model with a dropout rate of 30%. This dropout rate implies that during each training iteration, approximately 30% of the units or neurons from the preceding layer are randomly dropped or deactivated. This dropout mechanism serves as a regularization technique to prevent overfitting, enhancing the model's generalization capabilities. It encourages the network to rely on multiple pathways, reducing the risk of relying too heavily on specific neurons and thus improving the model's robustness.

The final layer in the network architecture is the Dense Layer 2, commonly known as the Output Layer. In this binary classification problem, a single unit is employed, and the activation function used is Sigmoid. Sigmoid activation maps the network's output to a probability score between 0 and 1, which is interpretable as the likelihood of the input data belonging to the positive class (intrusion) in binary classification scenarios. This layer is responsible for producing the final prediction or decision of whether a network traffic instance is normal (no intrusion) or malicious (intrusion detected) based on the learned features and patterns from the previous layers of the Convolutional Neural Network. Dense layer 2 is shown in Equation 7.

$$Dense_2(x) = Sigmoid(W_5 \cdot x + b_5) \quad (7)$$

This model is then compiled with the specified optimizer ("optimizer"), and the loss function used is binary cross-entropy ("binary_crossentropy"). The model undergoes training for 200 epochs using the training dataset ("X_train" and "y_train") with a batch size of 64.

B. Dataset

The dataset used in this research is Network Intrusion Detection obtained from Kaggle. This dataset consists of 25,192 data records with a total of 42 features covering various information related to network traffic. The objective of this research is to perform intrusion detection in computer networks by distinguishing between anomalies (target "yes") and normal situations (target "no"). This data has an important role in developing intrusion detection models that can improve network system security by identifying suspicious behavior and potential attacks.

C. Preprocessing

In the preprocessing stage of the network intrusion detection dataset, the main steps include factorization of text fields such as 'protocol_type', 'service', 'flag', and 'class' into numbers for processing by the model, normalization of numerical features such as 'duration' and 'src_bytes' so that their values are in the range [0, 1], and randomization of the dataset to avoid bias in the division of training and testing data. In addition, shorter data was filled with zero values through zero padding, and the data was deformed to match the input received by the CNN model. Thus, the data has been appropriately prepared for use in the training and evaluation of the CNN model for network intrusion detection.

D. Comparison of Methods

This research adopts the CNN algorithm as the main approach for network intrusion detection, while also comparing its performance with five baseline machine learning algorithms, namely SVM, Decision Tree, KNN, GNB, and GBoost. The chosen CNN approach relies on its deep understanding of network traffic data and its potential to extract complex features. Meanwhile, the comparison with baseline algorithms aims to identify the strengths and limitations of each method in the context of intrusion detection, as well as to provide greater insight into the effectiveness of CNNs in tackling this problem in comparison with conventional machine learning techniques.

E. Training and Evaluated

This research involves training the model with 200 epochs and a batch size of 64. The performance of the model is evaluated using metrics such as the Confusion Matrix and Classification Report. For the Confusion Matrix is shown in Equations 8-11.

$$TP\ (True\ Positives) = \sum_{i=1}^{N} \delta\left(y_i^{true} = 1 \cap y_i^{pred} = 1\right) \quad (8)$$

$$TN\ (True\ Negatives) = \sum_{i=1}^{N} \delta\left(y_i^{true} = 0 \cap y_i^{pred} = 0\right) \quad (9)$$

$$FP\ (False\ Positives) = \sum_{i=1}^{N} \delta\left(y_i^{true} = 0 \cap y_i^{pred} = 1\right) \quad (10)$$

$$FN\ (False\ Negatives) = \sum_{i=1}^{N} \delta\left(y_i^{true} = 1 \cap y_i^{pred} = 0\right) \quad (11)$$

Furthermore, for the Classification Report, precision, recall, F1-score, and accuracy can be calculated for each class. The mathematical formula for Classification Report is shown in Equations 12-15. These are some mathematical formulas related to model evaluation, and their results will be used to analyze the performance of the intrusion detection model in this research.

4. Results

A. Training Process

Fig. 1 is a visualization depicting the changes in accuracy during training and validation of the model. The training accuracy is represented by the blue line on the graph, whereas the orange line depicts the validation accuracy. Through the graph, it can be seen how the performance of the model changes as the number of training epochs increases. When there is a notable surge in training accuracy followed by a consistent rise in validation accuracy, it suggests that the model is proficient at learning from the training dataset and can effectively apply its knowledge to previously unseen validation data.

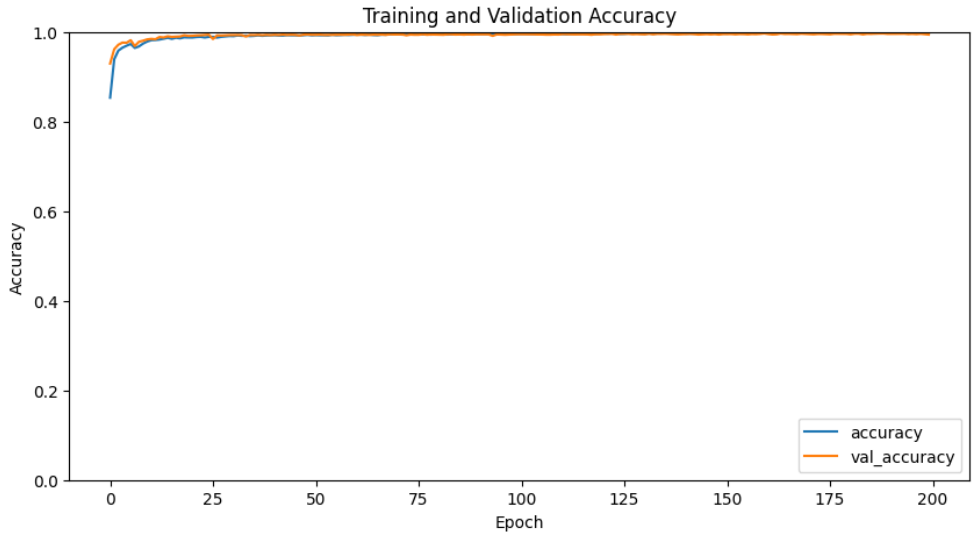

Fig. 1 Training and Validation Accuracy

Meanwhile, Fig. 2 displays the changes in the loss function during the training and validation of our model. The graph displays two lines, a blue one representing the training loss and an orange one illustrating the validation loss. This graph shows the extent to which the model can achieve convergence during training and whether there are signs of overfitting or underfitting. When the training loss and validation loss are close to or reach the same point, then our model has reached a good level of convergence, which indicates that the model training has successfully generalized the data well.

Fig 2. Training and Validation Loss

B. Model Performance

Table 1 contains the Confusion Matrix which includes TP, FP, FN, and TN of several algorithms, including CNN, SVM, Decision Tree, KNN, GNB, and GBoost. Table 1 describes the extent to which each algorithm successfully classifies the data, including in terms of detecting intrusion attacks and avoiding misclassification.

Table 1. Confusion Matrix.

Algorithms	TP	FP	FN	TN
CNN	2697	4	5	2333
SVM	2567	134	71	2267
DTree	2631	70	61	2277
KNN	2689	12	18	2320
GNB	2400	301	199	2139
Gboost	2691	10	20	2318

On the other hand, Table 2 presents the Classification Report, encompassing metrics like accuracy, precision, recall, and F1-score for the identical algorithms, namely CNN, SVM, Decision Tree, KNN, GNB, and GBoost. Table 2 goes deeper into the performance of each algorithm in classification by measuring accuracy, precision, ability to recall relevant information, and the balance between precision and recall represented by the F1-score.

Table 2. Classification Report

Algorithms	Accuracy	Class	Precision	Recall	F1-score
CNN	1	No	1	1	1
		Yes	1	1	1
SVM	0.96	No	0.97	0.95	0.96
		Yes	0.94	0.97	0.96
DTree	0.97	No	0.98	0.97	0.98
		Yes	0.97	0.97	0.97

KNN	0.99	No	0.99	1	0.99
		Yes	0.99	0.99	0.99
GNB	0.90	No	0.92	0.89	0.91
		Yes	0.88	0.91	0.90
Gboost	0.99	No	0.99	1	0.99
		Yes	1	0.99	0.99

C. Summarization of Key Findings

This research studies the important domain of network intrusion detection using a set of machine learning algorithms, namely CNN, SVM, Decision Tree, KNN, GNB, and GBoost. The main objective is to ascertain their effectiveness in distinguishing normal and intrusive network traffic, thereby improving network security. The main findings underscore the outstanding performance of KNN, which achieved the highest accuracy among the algorithms, followed by SVM and Decision Tree, all of which demonstrated their proficiency in intrusion detection. Moreover, this study emphasizes the suitability of CNN for this task, with perfect accuracy in classifying normal and intrusion traffic, highlighting its potential as a powerful solution. These findings shed light on the diverse strengths of these algorithms, providing valuable insights for network security practitioners to choose the most suitable approach to maintain network integrity.

D. Interpretation

The analysis of the results of this study includes several important aspects. Firstly, it involves identifying intricate patterns and relationships that emerge among machine learning algorithms and related performance metrics, such as accuracy, precision, recall, and F1 scores. In particular, KNN emerged as a standout algorithm, underscoring the strong interaction between accuracy and precision. The results largely met expectations, confirming the hypothesis that such algorithms exhibit varying degrees of success in network intrusion detection. However, the unexpectedly perfect accuracy achieved by CNN suggests the need for deeper investigation into potential factors driving these results, such as the characteristics of the dataset and model parameters. These findings effectively align with existing research on intrusion detection, reaffirming the potential of machine learning algorithms. Nevertheless, the revelations of this study, especially the outstanding performance of KNN and CNN, offer new insights into this field. These unexpected results warrant further exploration, perhaps by considering class imbalance, ensemble methods, and feature engineering, paving the way for future research to optimize real-world intrusion detection systems through different approaches.

E. Implication of The Research

This research carries significant relevance and implications in the realm of network intrusion detection. Evaluating and comparing multiple machine learning algorithms provides valuable insights into selecting the most suitable approach for bolstering network security. The findings resonate with prior literature on intrusion detection, aligning with the recognized effectiveness of algorithms like SVM and Decision Tree. However, the exceptional performance of KNN and the unexpected perfect accuracy of CNN offer new perspectives. These results suggest that KNN and CNN could be formidable contenders in real-world intrusion detection scenarios. The study underscores the importance of not only leveraging existing knowledge but also exploring the unique strengths of diverse algorithms, potentially paving the way for hybrid or ensemble approaches that harness the capabilities of multiple models. In essence, this research contributes novel insights by providing a comprehensive comparative analysis, empowering practitioners to make informed decisions for enhancing network security.

F. Limitation of The Result

This research provides a comprehensive view of how different machine learning algorithms perform in the context of network intrusion detection. The research highlights KNN as a robust algorithm, which demonstrates its success in real-world intrusion detection scenarios. The unexpectedly perfect accuracy of the CNN sparks curiosity and warrants further investigation. However, it is important to recognize the impact of limitations, such as dataset characteristics and specific hyperparameters chosen. Despite these constraints, the results remain valid for answering the research questions, as they provide valuable insights into the relative strengths and weaknesses of these algorithms. The findings serve as a basis for informed decision-making in the deployment of intrusion detection systems, and the unexpected results invite further exploration to refine and optimize these systems for real-world applications.

G. Future research recommendation

Practical implementation recommendations stemming from this research emphasize the adoption of KNN and CNN for real-world intrusion detection systems, potentially in a combined or ensemble approach. Ongoing fine-tuning of model hyperparameters, vigilant adaptation to evolving intrusion patterns, and the refinement of data preprocessing and feature engineering methods are essential for maintaining system effectiveness. In future research, a deeper exploration of CNN accuracy, addressing class imbalance, and investigating hybrid models is warranted. Additionally, assessing model resilience against adversarial attacks, scalability to large-scale network traffic, real-time intrusion detection, and integrating explainable AI techniques represent promising research directions, contributing to the advancement of network intrusion detection and security against emerging threats.

5. Conclusion

In conclusion, the results and discussion presented in this research offer invaluable insights into the field of network intrusion detection. The comparative analysis of machine learning algorithms, including KNN and CNN, underscores the effectiveness of these approaches in distinguishing normal from intrusive network traffic. While KNN emerged as the best-performing algorithm, the perfect accuracy achieved by CNN adds an interesting dimension to these findings. This research is in line with previous literature on intrusion detection, confirming the capabilities of existing algorithms such as SVM and Decision Tree. However, it also provides a new perspective, hinting at the potential of KNN and CNN in real-world intrusion detection scenarios. Despite some limitations, these results remain valid and relevant, providing a basis for informed decision-making in improving network security. The implications of this research extend to practical implementation and future research directions, offering valuable guidance in the ongoing quest for a robust intrusion detection system.

Data Availability

The dataset Kaggle (www.kaggle.com/datasets/sampadab17/network-intrusion-detection).

References

[1] G. Yadav, “Network Intrusion Detection - Comparison across classifiers,” vol. 7, no. 10, 2020.

[2] H. Y. Fan and S. Anjani, “IDS-GAN: Stepping up Intrusion Detection Method using GAN Algorithm,” Aug. 2023.

[3] J. Jie and P. Wanda, “A Survey of Network and Information Security,” Dec. 2019.
[4] A. Ezenwe, E. Furey, and K. Curran, “Mitigating Denial of Service Attacks with Load Balancing,” *J. Robot. Control JRC*, vol. 1, no. 4, 2020, doi: 10.18196/jrc.1427.
[5] D. J. Marchette, “Network intrusion detection”.
[6] N. B. Nanda and Dr. A. Parikh, “Classification and Technical Analysis of Network Intrusion Detection Systems,” in *2016 International Symposium on Networks, Computers and Communications (ISNCC)*, Yasmine Hammamet, Tunisia: IEEE, May 2016, pp. 1–6. doi: 10.1109/ISNCC.2016.7746067.
[7] P. Wanda and H. J. Jie, “A Survey of Intrusion Detection System,” Aug. 2019.
[8] I. V. Zhukovyts’kyy, V. M. Pakhomova, D. O. Ostapets, and O. I. Tsyhanok, “DETECTION OF ATTACKS ON A COMPUTER NETWORK BASED ON THE USE OF NEURAL NETWORKS COMPLEX,” *Sci. Transp. Prog. Bull. Dnipropetrovsk Natl. Univ. Railw. Transp.*, vol. 0, no. 5(89), pp. 68–79, Dec. 2020, doi: 10.15802/stp2020/218318.
[9] L. Zhou, Y. Zhu, Y. Xiang, and T. Zong, “A novel feature-based framework enabling multi-type DDoS attacks detection,” *World Wide Web*, vol. 26, no. 1, pp. 163–185, Jan. 2023, doi: 10.1007/s11280-022-01040-3.
[10] 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,” *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 1, Jan. 2021, doi: 10.1002/ett.4150.
[11] J. V. Shirabayashi, A. S. M. Braga, and J. da Silva, “Comparative approach to different convolutional neural network (CNN) architectures applied to human behavior detection,” *Neural Comput. Appl.*, vol. 35, no. 17, pp. 12915–12925, Jun. 2023, doi: 10.1007/s00521-023-08430-2.
[12] B. Ibrahim Hairab, H. K. Aslan, M. S. Elsayed, A. D. Jurcut, and M. A. Azer, “Anomaly Detection of Zero-Day Attacks Based on CNN and Regularization Techniques,” *Electronics*, vol. 12, no. 3, p. 573, Jan. 2023, doi: 10.3390/electronics12030573.
[13] N. Thapa, Z. Liu, D. B. Kc, B. Gokaraju, and K. Roy, “Comparison of Machine Learning and Deep Learning Models for Network Intrusion Detection Systems,” *Future Internet*, vol. 12, no. 10, p. 167, Sep. 2020, doi: 10.3390/fi12100167.
[14] Keshav A Ruia, Milap M Alondra, Sahil M Shrivatsava, Mitesh V Salunke, Dr. Geeta S Navale, and Mrs. Supriya H Lokhande, “Network Intrusion Detection by Machine Learning Techniques,” *Int. J. Adv. Res. Sci. Commun. Technol.*, pp. 514–520, May 2022, doi: 10.48175/IJARSCT-3695.
[15] H. Liu and H. Wang, “Real-Time Anomaly Detection of Network Traffic Based on CNN,” *Symmetry*, vol. 15, no. 6, p. 1205, Jun. 2023, doi: 10.3390/sym15061205.
[16] B. Alabsi, M. Anbar, and S. Rihan, “CNN-CNN: Dual Convolutional Neural Network Approach for Feature Selection and Attack Detection on Internet of Things Networks,” *Sensors*, vol. 23, no. 14, p. 6507, Jul. 2023, doi: 10.3390/s23146507.
[17] S. Mukherjee and N. Sharma, “Intrusion Detection using Naive Bayes Classifier with Feature Reduction,” *Procedia Technol.*, vol. 4, pp. 119–128, 2012, doi: 10.1016/j.protcy.2012.05.017.
[18] T. Wisanwanichthan and M. Thammawichai, “A Double-Layered Hybrid Approach for Network Intrusion Detection System Using Combined Naive Bayes and SVM,” *IEEE Access*, vol. 9, pp. 138432–138450, 2021, doi: 10.1109/ACCESS.2021.3118573.
[19] S. Dhaliwal, A.-A. Nahid, and R. Abbas, “Effective Intrusion Detection System Using XGBoost,” *Information*, vol. 9, no. 7, p. 149, Jun. 2018, doi: 10.3390/info9070149.
[20] W. Li, P. Yi, Y. Wu, L. Pan, and J. Li, “A New Intrusion Detection System Based on KNN Classification Algorithm in Wireless Sensor Network,” *J. Electr. Comput. Eng.*, vol. 2014, pp. 1–8, 2014, doi: 10.1155/2014/240217.
[21] D. T. Son, N. T. K. Tram, and T. T. Thu, “Machine learning approach detects DDoS attacks,” *J. Sci. Technol. Inf. Secur.*, vol. 1, no. 15, pp. 102–108, Jun. 2022, doi: 10.54654/isj.v1i15.850.
[22] H. Hindy, E. Hodo, E. Bayne, A. Seeam, R. Atkinson, and X. Bellekens, “A Taxonomy of Malicious Traffic for Intrusion Detection Systems,” in *2018 International Conference On Cyber Situational Awareness, Data Analytics And Assessment (Cyber SA)*, Glasgow: IEEE, Jun. 2018, pp. 1–4. doi: 10.1109/CyberSA.2018.8551386.
[23] S. Saini, S. Bhatia, and I. S. Thaseen, “sv(M)kmeans - A Hybrid Feature Selection Technique for Reducing False Positives in Network Anomaly Detection”.
[24] A. Munther, R. Razif, M. AbuAlhaj, M. Anbar, and S. Nizam, “A Preliminary Performance Evaluation of K-means, KNN and EM Unsupervised Machine Learning Methods for Network

Flow Classification," *Int. J. Electr. Comput. Eng. IJECE*, vol. 6, no. 2, p. 778, Apr. 2016, doi: 10.11591/ijece.v6i2.8909.
[25] A. Guezzaz, S. Benkirane, M. Azrour, and S. Khurram, "A Reliable Network Intrusion Detection Approach Using Decision Tree with Enhanced Data Quality," *Secur. Commun. Netw.*, vol. 2021, pp. 1–8, Aug. 2021, doi: 10.1155/2021/1230593.
[26] M. Aljanabi, M. A. Ismail, and A. H. Ali, "Intrusion Detection Systems, Issues, Challenges, and Needs:," *Int. J. Comput. Intell. Syst.*, vol. 14, no. 1, p. 560, 2021, doi: 10.2991/ijcis.d.210105.001.
[27] A. Parashar, K. S. Saggu, and A. Garg, "Machine learning based framework for network intrusion detection system using stacking ensemble technique," *Indian J. Eng. Mater. Sci.*, vol. 29, no. 04, 2022, doi: 10.56042/ijems.v29i4.46838.