applied sciences

MDPI

Article

Comparative Analysis of Deep Convolutional Neural Network—Bidirectional Long Short-Term Memory and Machine Learning Methods in Intrusion Detection Systems

Miracle Udurume, Vladimir Shakhov and Insoo Koo *

Department of Electrical Electronic and Computer Engineering, Ulsan University, 93 Daehak-ro, Namgu, Ulsan 44610, Republic of Korea; udurumemiracle@kumoh.ac.kr (M.U.); shakhov@mail.ulsan.ac.kr (V.S.)
* Correspondence: iskoo@ulsan.ac.kr

Abstract: Particularly in Internet of Things (IoT) scenarios, the rapid growth and diversity of network traffic pose a growing challenge to network intrusion detection systems (NIDs). In this work, we perform a comparative analysis of lightweight machine learning models, such as logistic regression (LR) and k-nearest neighbors (KNNs), alongside other machine learning models, such as decision trees (DTs), support vector machines (SVMs), multilayer perceptron (MLP), and random forests (RFs) with deep learning architectures, specifically a convolutional neural network (CNN) coupled with bidirectional long short-term memory (BiLSTM), for intrusion detection. We assess these models' scalability, performance, and robustness using the NSL-KDD and UNSW-NB15 benchmark datasets. We evaluate important metrics, such as accuracy, precision, recall, F1-score, and false alarm rate, to offer insights into the effectiveness of each model in securing network systems within IoT deployments. Notably, the study emphasizes the utilization of lightweight machine learning models, highlighting their efficiency in achieving high detection accuracy while maintaining lower computational costs. Furthermore, standard deviation metrics have been incorporated into the accuracy evaluations, enhancing the reliability and comprehensiveness of our results. Using the CNN-BiLSTM model, we achieved noteworthy accuracies of 99.89% and 98.95% on the NSL-KDD and UNSW-NB15 datasets, respectively. However, the CNN-BiLSTM model outperforms lightweight traditional machine learning methods by a margin ranging from 1.5% to 3.5%. This study contributes to the ongoing efforts to enhance network security in IoT scenarios by exploring a trade-off between traditional machine learning and deep learning techniques.

Keywords: deep learning; intrusion detection; Internet of Things; machine learning; network security

Citation: Udurume, M.; Shakhov, V.; Koo, I. Comparative Analysis of Deep Convolutional Neural Network—Bidirectional Long Short-Term Memory and Machine Learning Methods in Intrusion Detection Systems. *Appl. Sci.* **2024**, *14*, 6967. https://doi.org/10.3390/app14166967

Academic Editors: Rongxing Lu, Sin Gee Teo and Ruitao Feng

Received: 19 July 2024
Revised: 7 August 2024
Accepted: 7 August 2024
Published: 8 August 2024

1. Introduction

Research Background

As the internet has expanded rapidly, cyberattacks on computer systems and networks have increased in frequency. To address these risks, network systems employ intrusion detection systems (IDSs) to detect and stop malicious activity. In our digital age, where dependence on technology is essential, creating secure and reliable programs, frameworks, and networks is one of the primary challenges [1]. An important deterrent to network intrusion is the ability to detect faults in a system. Intrusion detection is defined as the process of identifying and responding to malicious activity targeting computing and network resources. The rapid increase in network traffic and resulting security threats make it more difficult for IDSs to accurately detect hostile attacks. The idea is to provide up-to-date information on machine learning and deep learning-based intrusion detection systems to create a baseline for further researchers exploring this important area [2]. Studies have shown that lightweight machine learning models, such as decision trees and support vector machines, can be employed for intrusion detection. These models are computationally

efficient while maintaining acceptable detection accuracy, and they can adjust to shifting patterns by learning from past data [3].

Machine learning-based intrusion detection systems provide a learning-based method to identify attack classes by using learned normal and attack behavior. Machine learning-based intrusion detection systems use supervised learning techniques to generate a wide representation of known threats. The maintenance of the signature database is another requirement for these types of intrusion detection systems, and it increases user load [4]. Recently, there has been an outstanding advancement in intrusion detection techniques, achieved to a great extent by the remarkable capabilities of deep learning models. These deep learning models have introduced another era of accuracy and detection within various systems. Deep learning models, such as long-short-term memory (LSTM), recurrent neural networks (RNNs), convolutional neural networks (CNNs), and deep neural networks (DNNs), have demonstrated remarkable improvements in their learning behaviors and detection rates [5]. This has led to a shift in cybersecurity, with deep learning-based intrusion detection systems exhibiting high performance and resilience against emerging threats. Recent advancements in hardware architectures have facilitated the integration and deployment of sophisticated models that can ensure robust and reliable protection against cyber threats. Additionally, by utilizing deep learning models, the system can effectively detect anomalies within datasets, enabling the accurate and precise identification of security threats.

The motivation for employing lightweight machine learning models alongside deep learning models stems from the need to balance detection performance with resource efficiency. Models such as k-nearest neighbor (KNN), support vector machines (SVMs), multilayer perceptron (MLP), logistic regression (LR), and random forest (RFs) are chosen for their ability to deliver high accuracy while minimizing computational costs, making them suitable for real-time intrusion detection applications. Furthermore, deep learning and machine learning approaches have been increasingly applied to a range of Internet of Things (IoT) security solutions. Combining big data analytics with machine learning/deep learning-based methods has yielded significant outcomes in cybersecurity. Deep learning-based algorithms, in particular, have gained considerable attention due to their superior pattern-extraction capabilities, outperforming machine learning in various studies [6,7].

This paper provides a comparative analysis of deep learning and lightweight machine learning models used for intrusion detection systems. We evaluate the performance of a CNN-BiLSTM architecture alongside machine learning models (k-nearest neighbors (KNNs), support vector machines (SVMs), multilayer perceptron (MLP), logistic regression (LR), random forest (RF), and decision trees (DTs)) in identifying normal networks and detecting attacks, particularly denial of service (DoS)attacks within an IDS. The evaluation utilizes well-established datasets NSL-KDD and UNSW-NB15, assessing metrics such as accuracy, precision, recall, F1-score, and false alarm rate to determine the optimal model for IDS applications. To the best of our knowledge, no comparison has been made between using lightweight machine learning models and deep learning models. The remainder of this paper is structured as follows: Section 2 discusses related works in the field of intrusion detection, highlighting recent developments and methodologies. Section 3 presents a detailed overview of some recent intrusion detection schemes for IoT applications, as well as the methodologies employed. Section 4 presents the experimental results of both deep learning and machine learning models on the evaluated datasets. Section 5 discusses the implications of the results and compares them with state-of-the-art studies in the domain. Finally, Section 6 provides the conclusion of this study, summarizing the key findings and outlining avenues for future research in intrusion detection and IoT security.

2. Related Work

In IoT networks, cyberattack detection is typically a classification challenge. These challenges can be successfully resolved through the use of machine learning/deep learning approaches. An overview of various current machine learning/deep learning-based studies for IoT intrusion detection is provided in this section. To detect atypical attacks, Wang et al. [8] developed a flow mining-based method. They used MIT Lincoln Laboratories DDoS data and traces from the Slammer and Code Red worms to validate the effectiveness of their methodology. A deep defense network security architecture was also developed by Huang et al. [9], who suggested using data mining techniques to examine the alarms gathered by the distributed intrusion detection and prevention system (IDS/IPS). The author employed three distinct types of data mining methods to implement the prototype, and they used it in DDoS attack detection to assess the efficacy of the suggested defense architecture. In terms of the attack detection rate and FPR, it performed well. A technique based on the combination of a PCA and an optimized support vector machine (SVM) was presented by Thaseen and Kumar [10]. SVM parameters and kernels were optimized using recommended automatic parameter selection, which reduced the training time and improved accuracy for a few attacks, such as R2L and U2R. In contemporary networks, other well-liked machine learning techniques, including multilayer perceptron (MLP), random forest (RF), and Naive Bayes (NB), have also been employed to identify threats [11–13]. Unfortunately, shallow learning hinders the effectiveness of these conventional machine learning techniques, making them unable to offer a workable solution for a sizable amount of traffic data. Using several ML algorithms from the Weka Data Mining tool, Thanh and Lang [14] examined and assessed the performance of Bagging, AdaBoost, Stacking, Decorate, Voting, and random forest. Compared to single classifiers, the ensembles used in [14] that aggregate the results of their base classifiers using Stacking and Decorate procedures took longer to train and test than classifiers that employed other ensemble techniques.

A growing number of deep learning-based models have achieved exceptional performance as deep learning has advanced [15–17]. An IDS based on the recurrent neural network (RNN) was proposed by Yin et al. [18]. The design approach outperformed conventional classification techniques in terms of accuracy and detection rate for both binary and multiclass classification. To obtain more accurate detection by aggregating traffic characteristics, He et al. [19] suggested an intrusion detection model based on long short-term memory (LSTM) and a multimodal deep autoencoder. To mitigate DDoS attacks in cloud computing, Jaber et al. [20] employed principal component analysis and linear discriminant analysis in conjunction with a hybrid, nature-inspired metaheuristic algorithm called Ant Lion optimization for feature selection and artificial neural networks to classify and configure the cloud server. An IoT intrusion model leveraging CNN and grey wolf optimization (GWO) was applied to NID [21]. To address the issue of class imbalance in the dataset under consideration, Zhang et al. created a two-branch CNN and utilized feature fusion [22]. Their idea was more efficient in terms of execution time and had higher accuracy in detecting a minor class of anomalies. NID was applied to raw packet-level communication by Zhang et al. [23]. To extract significant spatial and temporal information, they combined CNNs and LSTMs, which resulted in greater detection rates than when utilizing each of these components separately. The performances of eight distinct machine learning algorithms were examined in [24] against six datasets, including KDD-99, NSL-KDD, UNSW-NB15, Kyoto2006+, and WSN-DS CICIDS2017. The algorithms include DNN, logistic regression (LR), NB, SVM, Adaptive Boosting (AB), KNN, DT, and RF. Despite having higher processing requirements, the deep learning classifier intuitively produced the greatest results when compared to the machine learning classifiers. An intrusion detection system (IDS) based on the deep LSTM algorithm, which uses recurrent neural networks (RNNs) and includes 90 hidden units spread across three hidden layers was created by Kasongo and Sun [25]. The accuracy of the model was 99.51%. There are 5 SoftMax neurons in the last layer of the DFFL structure and 29 sigmoid neurons in the first layer. This article demonstrated a notable improvement in performance by comparing the outcomes with

those of different machine learning techniques. They intend to investigate the effectiveness of every attack in the NSL-KDD dataset in future research.

Additionally, recent studies have demonstrated the effectiveness of CNN-BiLSTM architectures in different applications. For instance, Zhang et al. [26] utilized a CNN-BiLSTM-attention model for stock price prediction. In [27], Staffini applied a CNN-BiLSTM for macroeconomic time series forecasting to highlight new techniques that could be added to the set of tools available to a policymaker for forecasting macroeconomic data. Tang et al. [28] improved power load prediction using a CNN-BiLSTM model, and Cui and Xia [29] developed an EEG signal anomaly detection algorithm based on CNN-BiLSTM and compared with support vector machine to utilize the ability of CNN to automatically extract features and BiLSTM's ability to efficiently process time series data. These studies underscore the versatility and robustness of the CNN-BilSTM architecture, further motivating its application in intrusion detection systems. Other deep learning models have also been employed for intrusion detection studies. In [30], Naseer et al. employed the use of deep neural network structures, including CNN, autoencoders, and RNN, on the NSL-KDD dataset for real-world application in anomaly detection systems. In [31], Dan Dongseong Kim provided a comprehensive survey to discuss the impact of taxonomy on adversarial learning using deep learning-based network intrusion detection systems. In [32], Minshu He et al. proposed a framework for anomaly detection based on deep reinforcement learning, giving priority to outliers, state effect, and model transferability for reinforcement learning-based anomaly detection.

3. Materials and Methodology

Figure 1 depicts the system overview of the methodologies employed for the network-based IDS. Two benchmark datasets, NSL-KDD and UNSW-NB15, were used to train and test the deep learning and machine learning models for intrusion detection.

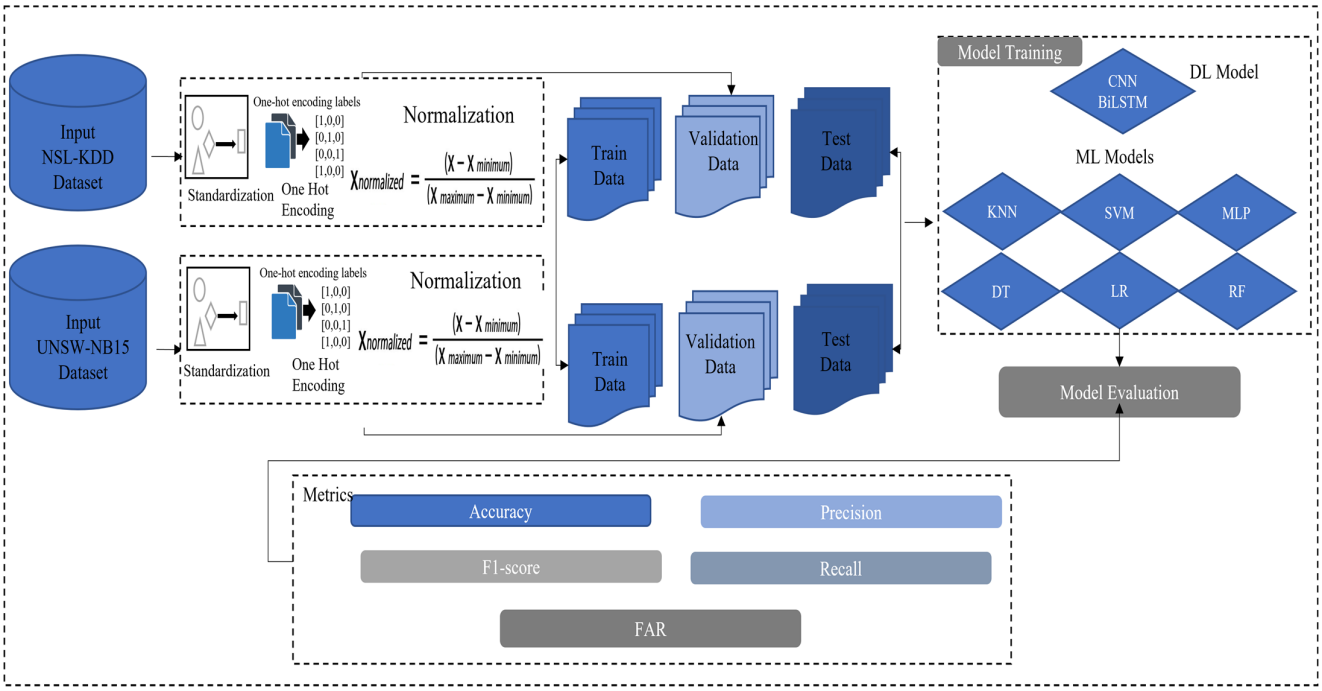


Figure 1. Overview of proposed network-based intrusion detection system.

3.1. Machine Learning Models

Machine learning has played a crucial role in intrusion detection systems and continues to do so. While K-means clustering and self-organized maps are instances of unsupervised learning, supervised learning is the foundation of machine learning algorithms, such as decision trees, support vector machines, and Naïve Bayes. The main purpose of machine learning algorithms is to increase a system's ability to detect attacks. Attacks and other dangers are identified using learned data. The most common applications of machine learning algorithms are in the areas of clustering, regression, and classification. Previous machine learning research has mostly used the NSL-KDD, DARPA, and KDD-CUP99 datasets [33]. The ideas underlying some of the traditional machine learning models in this study are presented below.

3.1.1. K-Nearest Neighbor

The k-nearest neighbor is a non-parametric supervised learning algorithm that classifies objects based on the majority class of their k-nearest neighbors in the feature space. The Euclidean is the most commonly used distance metric among all of these in KNN [34]. KNN is a slow, non-parametric learning algorithm that does not assume the underlying distribution of the data.

3.1.2. Support Vector Machine

The support vector machine is a powerful supervised learning model capable of both linear and non-linear classification [35]. It is appropriate for both classification and regression problems since each node represents a feature, and each leaf node correlates to a class label or regression value.

3.1.3. Decision Tree

Decision trees recursively partition the feature space into hierarchical structures to make decisions [36]. The method takes less time to train and can be applied to classify data that are not linearly separable.

3.1.4. Multilayer Perceptron

Multilayer perceptron (MLP) is a type of artificial neural network that uses multiple layers of nodes (perceptrons) with non-linear activation functions [37]. To maximize weights and biases, it learns using gradient descent, backpropagation of errors, and forward propagation of signals [38].

3.1.5. Logistic Regression

Logistic regression models the probability of a binary outcome using a logistic function [39]. Appropriate for binary and multiclass classification tasks, it calculates the connection between one or more independent variables and a dependent variable.

3.1.6. Random Forest

The random forest (RF) model is an ensemble learning technique that constructs multiple trees during training and outputs the class that is the mean prediction (regression) or the mode of the classes (classification) of the individual trees [40]. It uses an average of several decision trees to improve accuracy and manage overfitting [41].

3.2. Deep Learning Models

A subfield of machine learning called deep learning is focused on understanding how brain neurons function. Artificial neural networks (ANNs) are used to implement these algorithms. Several methods, such as supervised and unsupervised learning, can be used to train an algorithm for machine learning or deep learning. Supervised learning includes the task of classifying data examples that have been labeled during the training phase [42]. An LSTM network known as BiLSTM, or bidirectional LSTM, allows for better learning at

each data time step by feeding data from the beginning to the end of the model. As a result, features are learned more effectively at each time step.

Bi-LSTM layers contain two units, with identical input and output in the hidden layers, to enable learning from both forward and backward time-series data. Time-series data are handled by one unit in the forward direction and the other in the backward direction. This arrangement is thought to provide future data to the layers for improving training time with stronger feature learning, leading to improved precision for multi-dimensional time-series data. In Figure 2, an example of a bidirectional LSTM cell structure is given with input data x_{t-1}, x_{t+1}, x_t, x_{t-1}, alternately processed through a forward hidden layer h_{t-1}, h_t, h_{t+1} and backward hidden layer h_{t-1}, h_t, h_{t+1} to create y_{t-1}, y_t, y_{t+1}. A better prediction outcome can be obtained by optimizing the Bi-LSTM's feature information extraction through the forward and backward layers. The kernel size of the model doubles with each iteration according to the architecture of the BiLSTM architecture. An LSTM network with the addition of forward and backward layers forms a Bi-LSTM network.

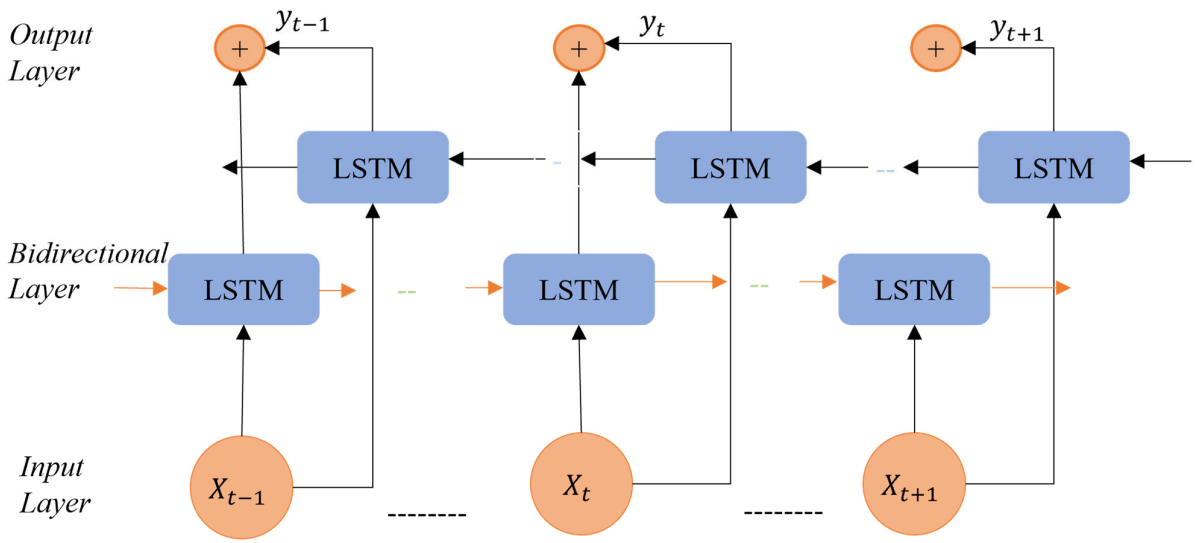


Figure 2. An example of a bidirectional LSTM cell structure.

CNN Bidirectional Long Short-Term Memory

The CNN BiLSTM model comprises multiple layers of Bi-LSTM separated by reshape and batch normalization layers, starting with a one-dimensional convolutional neural network layer. The one-dimensional convolutional neural network layer and the max pooling layer are intended to be used for local perception, spatial layout, and parameter sharing. Smaller sets of parameters and free variables are made possible via parameter sharing, which speeds up feature extraction while consuming less computing power. The grouping of features by spatial arrangement makes it possible to assess a sparse matrix, which improves the ability to identify correlations between features. Lastly, the training time is reduced greatly by using fewer parameters, which is possible with local perception. Consequently, a one-dimensional convolutional neural network allows for quick spatial learning using the provided time-series data. The max pooling layer, which comes after the one-dimensional convolutional neural network layer, enables sample-based discretization of parameters to identify important features, reducing the training time and preventing overfitting. The batch normalization layer, which comes after max pooling, enables parameter normalization between intermediate layers to avoid shortened training periods. Reshape layers, which come after batch normalization layers, reshape the output of the layer before it for the following two BiLSTM layers.

Convolutional neural networks (CNNs) are most commonly employed for image processing difficulties, but it has recently been revealed that CNNs can also be effective for natural language processing challenges. CNNs have made significant contributions to deep

learning and artificial intelligence. With a layered architecture, CNNs exhibit rapid data analysis capabilities facilitated by advanced feature extraction and reduction techniques. Operating as a spatial data filtering feed-forward network, CNNs perform two operations: convolution and pooling. Convolution involves transforming input data into output data through the application of kernels, highlighting key features, and generating a feature map. The output highlights the characteristics of the input data, which is why the output is referred to as a feature map. An activation function further analyzes the convolution result, and down-sampling uses pooling to remove extraneous data. CNNs use rounds of learning to update the kernels/filters so that the feature map can functionally represent the input data. Figure 3 depicts an example of a CNN architecture. Convolutional layers, dropout regularization, a multi-head attention mechanism, a dense layer, the max pooling layer, and the flattening layer are all included in the model. The feature maps acquired from the convolutional layer are down-sampled using max pooling to lower their dimensionality while keeping significant features. The output from the preceding layers is flattened into a vector by the flattening layer, readying it for the fully connected layer. A pair of dense layers, each consisting of 256 and 2 units, is incorporated. The ReLU activation function is employed in the first dense layer, and the SoftMax activation function is used in the final two units of the dense layer to produce probabilities for binary classification. The model is optimized using the Adam optimizer and trained with a categorical cross-entropy loss function for multiclass classification.

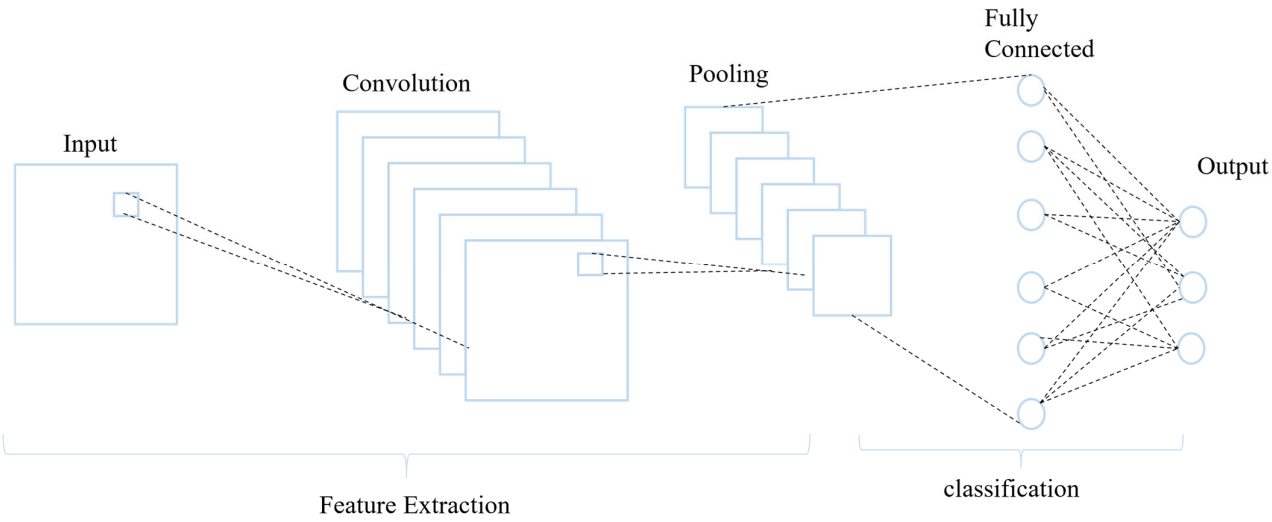


Figure 3. An example of a CNN architecture.

4. Experiments

All experiments were performed on a single server. The CPU of the server was AMD Ryzen 5 5600xU+00D7 6-Core Processor@3.70 GHz which was manufactured by Advanced Micro Devices in Santa Clara, CA, USA, and Windows 10 was installed. Python 3.9 programming language and Tensorflow were used as the deep learning framework to conduct the experiments. These resources provided a reliable and effective environment for our research and evaluation.

4.1. Description of the Datasets

4.1.1. NSL-KDD Datasets

The University of New Brunswick made the NSL-KDD dataset public. The NSL-KDD dataset is an upgrade of the KDDCup'99 dataset, which has inherent flaws, as shown by numerous analyses. NSL-KDD comprises the core records of the entire KDD dataset and is one of the most used datasets for analyzing network intrusion detection systems that can be applied as an effective benchmark to compare different intrusion detection methods, along with UNSW-NB15 and CICIDS-2017 [43]. The elimination of redundant

records, the availability of records in the training and testing datasets, and the inverse relationship between the number of selected records from each difficulty group and the percentage of records in the original KDD dataset are just a few of how NSL-KDD differs from its predecessor. The dataset contains 41 features, categorized into four groups, as listed in Table 1. Nine elements make up the first group (basic), which includes essential details, including the protocol, service, and length. Thirteen features are represented by the second category (content), which includes details about the content, including login activities. Nine time-based elements are contained in the third group (time), including the number of connections that are connected to the same host in a two-second window. Ten host-based elements are included in the fourth (host) section, and they offer details about the connection to the host, including the frequency of connections with the same destination port number attempting to be accessed by other hosts. The NSL-KDD dataset contains the KDDTrain+dataset as the training set and the KDDTest+ and KDDTest-21 datasets as the testing set. It contains normal traffic and four different attack types, namely denial of service (DoS), root to local (R2L), user to root (U2R), and probing attacks (Probe), as shown in Table 2.

Table 1. NSL-KDD dataset groups of features.

Group	Feature Numbers	Count
Basic feature	1, 2, 3, 4, 5, 6, 7, 8, 9	9
Content feature	10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21	13
Time feature	23, 24, 25, 26, 27, 28, 29, 30, 31	9
Host feature	32, 33, 34, 35, 36, 37, 38, 39, 40, 41	10

Table 2. Details of attack type for the NSL-KDD dataset.

Type	Training Dataset		Testing Dataset	
	KDD_Train	KDD_Train_20percent	KDD_Test	KDDTest-21
Normal	67,343	13,449	9711	2152
Probe	11,656	2289	2421	2402
DoS	45,927	9234	7458	4343
U2R	52	11	200	200
R2L	995	209	2751	2754
Total	125,973	25,192	22,544	11,850

4.1.2. UNSW-NB15 Datasets

UNSW-NB15 is a sophisticated dataset used in IDS research and is highly referenced in the literature. The IXIA Storm tool in the Cyber Range Laboratory of the Australian Centre for Cybersecurity (ACCS) produced the raw packets (network traces) that make up the UNSW-NB15 dataset. The dataset is simulated over 2.5 million network packets [44]. Nine different attack types, namely, exploit, reconnaissance, denial-of-service, shellcode, generic, backdoors, worms, fuzzers, and analysis attacks, as well as non-anomalous packets, are included in this dataset. The dataset is highly skewed since over 87% of the packets are non-anomalous. The dataset features and descriptions are provided in Table 3. The protocol feature, which identifies the protocols used by the hosts, such as TCP or UDP, is included in the first group (flow). The essential connection data, including the length of time and number of packets exchanged between the hosts, is represented by the second group (basic). Fourteen features are grouped. Content information, including base sequence numbers and window advertisement values, is sent by the third group (content) over the TCP. Additionally, it offers certain details about HTTP connections, including the amount of data sent through the HTTP service. There are eight characteristics in this group. Eight features, including packet arrival time and jitter, are included in the fourth group (time). Table 4 describes the attacks in the dataset.

Table 3. UNSW-NB15 groups of features.

Group	Feature Numbers	Count
Flow feature	2	1
Basic feature	1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15	14
Content feature	20, 21, 22, 23, 27, 28, 29, 30	8
Time feature	16, 17, 18, 19, 24, 25, 26, 42	8
Additional feature	31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41	11

Table 4. Details of attack type on the UNSW-NB15 dataset.

Type	Training Dataset	Testing Dataset
Normal	56,000	37,000
Generic	40,000	18,871
Exploits	33,393	11,132
Fuzzers	18,184	6062
DoS	12,264	4089
Reconnaissance	10,491	3496
Analysis	2000	677
Backdoor	1746	583
Shellcode	1133	378
Worms	130	44
Total	175,341	82,332

4.2. Data Pre-Processing

The primary goal of NIDs is to identify attack traffic. As a result, we start by filtering the two datasets, independently choosing the data samples that are marked as normal, and classifying the rest as DoS attacks. One-hot encoding of the categorical features and normalization of the numerical features are typically used to handle the pre-processing of the datasets. However, as previously mentioned, the NSL-KDD dataset contains a more precise amount of data for each attack category. Conversely, the UNSW-NB15 dataset contains a remarkably small number of records for categories such as fuzzers and worms. To address this problem, the training set employs the oversampling technique to ensure that each attack type contains the same number of records. We employ one-hot encoding and normalization for both datasets. Both datasets contain categorical features, which the deep learning model needs to translate into numerical values to produce accurate prediction results. Therefore, at the pre-processing stage, these columns were transformed into numerical values using the pandas Python Library's get dummies function. Because label encoders generate numerous numbers in a single column, the model may interpret these values incorrectly as being in a specific sequence, which could affect the classification. For this reason, one-hot encoding is preferred over label encoders.

Normalization is the process of rescaling the data into a specific range to minimize redundancy and speed up the model's training. This study also employs the use of min-max normalization [44], which rescales the data range to [0, 1].

$$X[i] = \frac{X[i] - X_{min}}{X_{max} - X_{min}} \tag{1}$$

4.3. Evaluation Metrics

Five standard classification performance measurements are adopted in this study to comprehensively estimate the machine learning and deep learning models. The classification measures are all based on four elements: True Positives (TPs), True Negatives (TNs), False Positives (FPs), and False Negatives (FNs). The representations of the utilized metrics are as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

$$Precision = \frac{\text{TP}}{TP + FP} \tag{3}$$

$$Recall = \frac{\text{TP}}{TP + FN} \tag{4}$$

$$F1\text{-}Score = \frac{2 \text{ x Precision x Recall}}{Precision + Recall} \tag{5}$$

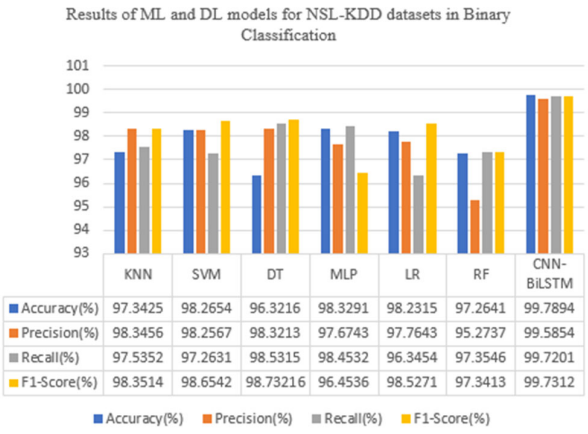
$$FAR = \frac{\text{FP}}{FP + \text{TN}} \tag{6}$$

Accuracy represents the proportion of network activities that are correctly classified, including both DoS attacks and normal traffic. Equation (2) defines its calculation. Equation (3) represents precision as the proportion of detected DoS attacks that are correctly classified. Recall measures the proportion of actual DoS attacks that are corrected and detected, as indicated by Equation (4). The F1-score is a thorough assessment metric. Its definition is given in Equation (5) as the weighted harmonic average of precision and recall. The percentage by which normal traffic is regarded as DoS attacks is known as the false alarm rate (FAR). Equation (6) illustrates how this can be quantified.

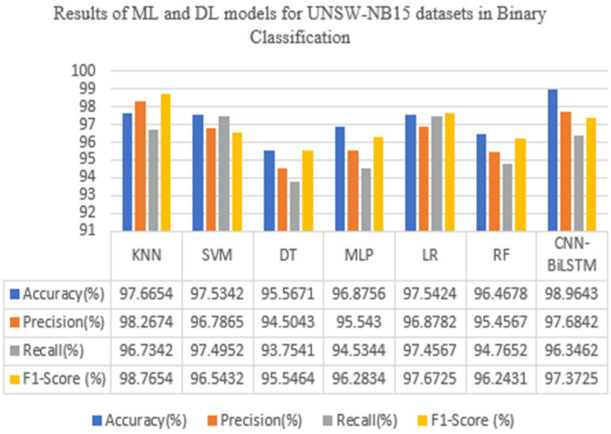
5. Discussion

5.1. Comparison with Machine Learning and Deep Learning Models

To validate the effectiveness of the intrusion detection system, we compared the deep learning model CNN-BiLSTM with the machine learning models KNN, SVM, DT, MLP, LR, and RF. The results of the NSL-KDD and UNSW-NB15 datasets in terms of binary classification are shown in Figure 4. The CNN-BiLSTM model demonstrates notable accuracy, precision, recall, and F1-score, achieving scores of 99.78%, 99.58%, 99.72%, and 99.73% for the NSL-KDD dataset and 98.96%, 97.68%, 96.34%, and 97.37% for the UNSW-NB15 dataset, respectively. Figure 5 shows the results for both datasets in multiclass classification. Here, both the deep learning and machine learning models are trained to detect a specific attack, which is the DoS attack. By combining the performance of different classifications on the two datasets, we can achieve good results in terms of DoS attack detection, accompanied by a low FAR rate. As shown in Tables 5 and 6, the deep learning model achieves the highest accuracy compared with the machine learning models, with an accuracy of 99.89% on the NSL-KDD dataset and 98.95% on the UNSW-NB15 dataset. The results in Tables 5 and 6 are based on the average performance over five experimental runs for each model. Lower values of the standard deviation imply more stable results across several runs, and they show how consistently the model performs. Moreover, it can be seen that the deep learning model performs well in terms of the FAR on both datasets, indicating very accurate predictions. Amongst the machine learning models, it can be observed that the SVM model outperforms the machine learning models, with an accuracy of 98.65% on the NSL-KDD dataset. It effectively distinguishes between normal network and attack instances, resulting in a low FAR. SVM's ability to handle complex decision boundaries makes it suitable for intricate intrusion detection scenarios. On the other hand, KNN achieved the lowest accuracy in comparison with other machine learning models on the NSL-KDD dataset. It achieved an accuracy of 97.42% and a balanced precision–recall trade-off, indicating its effectiveness in correctly classifying both normal and attack instances. However, it has a relatively false alarm rate compared to the other models, and this is because it relies on a simple majority voting scheme among its nearest neighbors. Hence, in scenarios where the nearest neighbors include noisy or misclassified instances, KNN may produce false alarms when classifying test instances, especially in regions of the feature space where the density of training instances is sparse or heterogeneous. KNN outperforms the other machine learning models, with an accuracy of 97.78% on the UNSW-NB15 dataset.



(**a**)



(**b**)

Figure 4. Results for all binary classification models: (**a**) binary classification on NSL-KDD; (**b**) binary classification on UNSW-NB15.

Table 5. Effective comparison of machine learning and deep learning models on NSL-KDD datasets.

Classifiers	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	FARnorm (%)	FARattacks (%)
KNN	97.42 ± 0.510	98.45 ± 0.655	97.67 ± 0.567	98.77 ± 0.460	0.2231	0.1320
SVM	98.65 ± 0.538	99.76 ± 0.669	97.54 ± 0.453	98.54 ± 0.577	0.2325	0.1551
DT	96.54 ± 0.610	98.54 ± 0.506	98.01 ± 0.601	98.74 ± 0.543	0.2539	0.1810
MLP	98.32 ± 0.407	97.46 ± 0.494	98.42 ± 0.422	96.28 ± 0.340	0.0215	0.0601
LR	98.44 ± 0.313	97.35 ± 0.404	96.25 ± 0.334	98.45 ± 0.422	0.1692	0.1024
RF	97.75 ± 0.461	95.56 ± 0.388	97.12 ± 0.492	97.42 ± 0.420	0.2009	0.1253
CNN-BiLSTM	99.89 ± 0.510	99.85 ± 0.532	99.83 ± 0.532	99.84 ± 0.532	0.0088	0.0017

Table 6. Effective comparison of machine learning and deep learning on UNSW-NB15 datasets.

Classifiers	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	FARnorm (%)	FARattacks (%)
KNN	97.78 ± 0.012	98.17 ± 0.027	96.61 ± 0.014	98.89 ± 0.027	0.0466	0.0439
SVM	97.42 ± 0.004	96.88 ± 0.026	97.38 ± 0.004	96.62 ± 0.026	0.0042	0.0264
DT	95.32 ± 0.044	94.60 ± 0.011	93.66 ± 0.044	95.69 ± 0.011	0.0241	0.0111
MLP	96.98 ± 0.018	95.46 ± 0.016	94.41 ± 0.018	96.19 ± 0.016	0.0177	0.0159
LR	97.32 ± 0.005	96.84 ± 0.027	97.34 ± 0.006	97.58 ± 0.027	0.0056	0.0265
RF	96.25 ± 0.021	95.36 ± 0.011	94.86 ± 0.021	94.11 ± 0.011	0.0205	0.0114
CNN-BiLSTM	98.95 ± 0.017	97.73 ± 0.018	96.20 ± 0.017	96.16 ± 0.018	0.0172	0.0183

KNN	Predicted Class	
Actual Class	DoS	Normal
DoS	9,422	289
Normal	1,573	5,887

SVM	Predicted Class	
Actual Class	DoS	Normal
DoS	9,455	256
Normal	1,359	6,101

DT	Predicted Class	
Actual Class	DoS	Normal
DoS	9,607	104
Normal	1,797	5,663

MLP	Predicted Class	
Actual Class	DoS	Normal
DoS	9,653	58
Normal	4,316	3,114

LR	Predicted Class	
Actual Class	DoS	Normal
DOS	9,702	140
Normal	1,528	5,789

RF	Predicted Class	
Actual Class	DoS	Normal
DoS	9,589	302
Normal	1,453	5,324

CNN-BiLSTM	Predicted Class	
Actual Class	DoS	Normal
DoS	8,884	13
Normal	15	15,840

(**a**)

KNN	Predicted Class	
Actual Class	DoS	Normal
DoS	12,224	102
Normal	173	3,736

SVM	Predicted Class	
Actual Class	DoS	Normal
DoS	12,311	15
Normal	334	3,657

DT	Predicted Class	
Actual Class	DoS	Normal
DoS	12,152	174
Normal	136	3,773

MLP	Predicted Class	
Actual Class	DoS	Normal
DoS	12,259	67
Normal	198	3,711

LR	Predicted Class	
Actual Class	DoS	Normal
DoS	12,306	20
Normal	337	3,572

RF	Predicted Class	
Actual Class	DoS	Normal
DoS	12,247	79
Normal	141	3,768

CNN-BiLSTM	Predicted Class	
Actual Class	DoS	Normal
DoS	14,207	1,293
Normal	1,376	26,069

(**b**)

Figure 5. Results for the confusion matrix: (**a**) confusion matrix on NSL-KDD; (**b**) confusion matrix on UNSW-NB15. By combining the performance of different classification algorithms on the two datasets, we find that both the machine learning and deep learning models can achieve good results in attack detection accompanied by a low FAR.

5.2. Comparison with State-of-the-Art Methods

To verify the obtained results, we compare both the machine learning and deep learning models with other detection attack methods. The NSL-KDD dataset was used by [45–47] since it is the most commonly used benchmark dataset, while this work considered both the NSL-KDD and UNSW-NB15 datasets. The comparison results are summarized in Table 7. Ref. [45] achieved the highest accuracy of 99.95%, while the deep learning model used in this work achieved an accuracy of 99.89% on the NSL-KDD dataset and 98.95% on the UNSW-NB15 dataset.

Table 7. Comparison with state-of-the-art methods.

Reference	Dataset	Accuracy (%)
This work	NSL-KDD UNSW-NB15	99.89 98.95
[45]	NSL-KDD	99.95
[46]	NSL-KDD	98.23
[47]	NSL-KDD	99.54

5.3. Discussion

This section discusses the insights into why certain models perform better than others and also presents the practical implications of our findings. Specifically, we address the trade-off between accuracy, latency, and computational complexity in the context of IoT environments. In terms of accuracy vs latency, while the CNN-BiLSTM model showed superior accuracy and other performance metrics, its increased computational complexity and latency might pose issues for real-time intrusion detection in IoT environments. This observation aligns with findings from [48], who also noted that deep learning models, although accurate, often require significant computational resources that may not be feasible for time-sensitive applications. The model's extensive architecture, combining convolutional layers with bidirectional LSTM layers, demands significant computational resources and time, which may not be feasible for systems requiring immediate threat response. Lightweight models such as logistic regression and k-nearest neighbors demonstrated competitive accuracy with much lower computational requirements. These models are suitable for real-time applications where latency is critical, as they can process data and generate alerts much faster than deep learning models [3]. For accuracy vs computational complexity, the CNN-BiLSTM model's complex structure leads to a higher accuracy but also demands more computational power, memory, and processing time. This complexity can be a limiting factor in resource-constrained environments like IoT. A similar conclusion was drawn by [49], who discussed the trade-offs between model complexity and feasibility in IoT contexts. A practical implication involves deploying lightweight models for preliminary detection to ensure timely responses, while more complex models like the CNN-BiLSTM models can be used for detailed analysis and configuration of threats. Models, such as decision trees and random forests, although not as accurate as CNN-BiLSTM, offer a good balance between performance and computational complexity. They are easier to implement and require fewer resources, making them suitable for environments with limited computational capacity.

6. Conclusions

This study presents a comparative analysis of deep CNN-BiLSTM and machine learning methods in intrusion detection scenarios. It evaluates the performance of deep learning and machine learning models using two benchmark datasets, NSL-KDD and UNSW-NB15, focusing on their effectiveness in detecting DoS attacks. The results demonstrate that the CNN-BiLSTM model outperforms the machine learning models in terms of accuracy, precision, recall, and F1-score and exhibits lower false alarm rates. Despite not introducing a new algorithm, this study provides valuable insights into the comparative reliability of

deep learning and lightweight machine learning models in network intrusion detection to balance the trade-off between system cost and model performance. The inclusion of standard deviation in the accuracy metrics adds robustness to the performance comparison.

In future work, we aim to propose a deep learning model that can be deployed in a real-world environment, compare the detection of specific attacks such as DDoS attacks, and assess the model's complexity and robustness by varying the training datasets. This will further enhance the practical applicability and robustness of intrusion detection systems in diverse and dynamic network environments.

Author Contributions: Conceptualization, methodology, validation, and writing, M.U.; conceptualization, and formal analysis, M.U. and V.S.; supervision, funding acquisition, and review, V.S. and I.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (RS-2023-00250720) and in part by the Regional Innovation Strategy (RIS) through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (MOE) under Grant 2021RIS-003.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All datasets are available in the mentioned references.

Acknowledgments: We thank the above projects for their support.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zavrak, S.; İskefiyeli, M. Anomaly-Based Intrusion Detection from Network Flow Features Using Variational Autoencoder. *IEEE Access* **2020**, *8*, 108346–108358. [CrossRef]
2. Kim, T.; Pak, W. Early Detection of Network Intrusions Using a GAN-Based One-Class Classifier. *IEEE Access* **2022**, *10*, 119357–119367. [CrossRef]
3. Jan, S.U.; Ahmed, S.; Shakhov, V.; Koo, I. Toward a Lightweight Intrusion Detection System for the Internet of Things. *IEEE Access* **2019**, *7*, 42450–42471.
4. Mishra, P.; Varadharajan, V.; Tupakula, U.; Pilli, E.S. A Detailed Investigation and Analysis of Using Machine Learning Techniques for Intrusion Detection. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 686–728. [CrossRef]
5. Hnamte, V.; Nhung-Nguyen, H.; Hussain, J.; Hwa-Kim, Y. A Novel Two-Stage Deep Learning Model for Network Intrusion Detection: LSTM-AE. *IEEE Access* **2023**, *11*, 37131–37148. [CrossRef]
6. Sun, P.; Liu, P.; Li, Q.; Liu, C.; Lu, X.; Hao, R.; Chen, J. DL-IDS: Extracting features using CNN-LSTM hybrid network for intrusion detection system. *Secur. Commun. Netw.* **2020**, *2020*, 8890306. [CrossRef]
7. Latif, S.; e Huma, Z.; Jamal, S.S.; Ahmed, F.; Ahmad, J.; Zahid, A.; Dashtipour, K.; Aftab, M.U.; Ahmad, M.; Abbasi, Q.H. Intrusion Detection Framework for the Internet of Things Using a Dense Random Neural Network. *IEEE Trans. Ind. Inform.* **2022**, *18*, 6435–6444. [CrossRef]
8. Wang, J.; Miller, D.J.; Kesidis, G. Efficient mining of the multidimensional traffic cluster hierarchy for digesting visualization and anomaly identification. *IEEE J. Sel. Areas Commun.* **2006**, *24*, 1929–1941. [CrossRef]
9. Huang, N.-F.; Kao, C.-N.; Hun, H.-W.; Jai, G.-Y.; Lin, C.-L. Apply data mining to defense-in-depth network security system. In Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA Papers), Taipei, Taiwan, 28–30 March 2005.
10. Thaseen, I.S.; Kumar, C.A. Intrusion detection model using fusion of PCA and optimized SVM. In Proceedings of the International Conference Contemporary Computer Informatics, Mysore, India, 27–29 November 2014.
11. Usha, M.; Kavitha, P. Anomaly-based intrusion detection for 802.11 networks with optimal features using SVM classifier. *Wirel. Netw.* **2017**, *23*, 2431–2446. [CrossRef]
12. Pajouh, H.H.; Javidan, R.; Khayami, R.; Dehghantanha, A.; Choo, K.-K.R. A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks. *IEEE Trans. Emerg. Topics Comput.* **2019**, *7*, 314–323. [CrossRef]
13. Wang, H.; Gu, J.; Wang, S. An effective intrusion detection framework based on SVM with feature augmentation. *Knowl. Based Syst.* **2017**, *139*, 130–139. [CrossRef]
14. Thanh, H.N.; Lang, T.V. Use the ensemble methods when detecting DoS attacks in network intrusion detection systems. *EAI Endorsed Trans. Context Aware Syst. Appl.* **2019**, *6*, e5. [CrossRef]

15. Xiong, Z.; Xu, H.; Li, W.; Cai, Z. Multi-source adversarial sample attack on autonomous vehicles. *IEEE Trans. Veh. Technol.* **2021**, *70*, 2822–2835. [CrossRef]
16. Cai, Z.; Xiong, Z.; Xu, H.; Wang, P.; Li, W.; Pan, Y. Generative adversarial networks: A survey toward private and secure applications. *ACM Comput. Surveys* **2021**, *54*, 1–38. [CrossRef]
17. Gui, G.; Liu, M.; Tang, F.; Kato, N.; Adachi, F. 6G: Opening new horizons for integration of comfort security and intelligence. *IEEE Wirel. Commun.* **2020**, *27*, 126–132. [CrossRef]
18. Yin, C.; Zhu, Y.; Fei, J.; He, X. A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access* **2017**, *5*, 21954–21961. [CrossRef]
19. He, H.; Sun, X.; He, H.; Zhao, G.; He, L.; Ren, J. A novel multimodal-sequential approach based on multi-view features for network intrusion detection. *IEEE Access* **2019**, *7*, 183207–183221. [CrossRef]
20. Jaber, A.N.; Zolkipli, M.F.; Shakir, H.A.; Jassim, M.R. Host-based intrusion detection and prevention model against DDoS attack in cloud computing. In *Advances on P2P, Parallel, Grid, Cloud and Internet Computing*; 3PGCIC 2017. Lecture Notes on Data Engineering and Communications Technologies; Springer: Cham, Switzerland, 2017.
21. Garg, S.; Kaur, K.; Kumar, N.; Kaddoum, G.; Zomaya, A.Y.; Ranjan, R. A hybrid deep learning-based model for anomaly detection in cloud data center networks. *IEEE Trans. Netw. Service Manag.* **2019**, *16*, 924–935. [CrossRef]
22. Vinayakumar, R.; Soman, K.P.; Poornachandran, P. Applying convolutional neural network for network intrusion detection. In Proceedings of the International Conference Advance Computer Informatics (ICACCI), Udupi, India, 13–16 September 2017.
23. Zhang, Y.; Chen, X.; Jin, L.; Wang, X.; Guo, D. Network intrusion detection: Based on deep hierarchical network and original flow data. *IEEE Access* **2019**, *7*, 37004–37016. [CrossRef]
24. Vinayakumar, R.; Alazab, M.; Soman, K.; Poornachandran, P.; Al-Nemrat, A.; Venkatraman, S. Deep learning approach for intelligent intrusion detection system. *IEEE Access* **2019**, *7*, 41525–41550. [CrossRef]
25. Kasongo, S.M.; Sun, Y. A deep long short-term memory-based classifier for wireless intrusion detection system. *ICT Exp.* **2020**, *6*, 98–103. [CrossRef]
26. Zhang, J.; Ye, L.; Lai, Y. Stock Price Prediction Using CNN-BiLSTM-Attention Model. *Mathematics* **2023**, *11*, 1985. [CrossRef]
27. Staffini, A. A CNN–BiLSTM Architecture for Macroeconomic Time Series Forecasting. *Eng. Proc.* **2023**, *39*, 33. [CrossRef]
28. Tang, C.; Zhang, Y.; Wu, F.; Tang, Z. An Improved CNN-BILSTM Model for Power Load Prediction in Uncertain Power Systems. *Energies* **2024**, *17*, 2312. [CrossRef]
29. Cui, K.X.; Xia, X.J. ECG Signal Anomaly Detection Algorithm Based on CNN-BiLSTM. In Proceedings of the 2022 11th International Conference of Information and Communication Technology (ICTech)), Wuhan, China, 4–6 February 2022; pp. 193–197.
30. Naseer, S.; Saleem, Y.; Khalid, S.; Bashir, M.K.; Han, J.; Iqbal, M.M.; Han, K. Enhanced Network Anomaly Detection Based on Deep Neural Networks. *IEEE Access* **2018**, *6*, 48231–48246. [CrossRef]
31. He, K.; Kim, D.D.; Asghar, M.R. Adversarial Machine Learning for Network Intrusion Detection Systems: A Comprehensive Survey. *IEEE Commun. Surv. Tutor.* **2022**, *25*, 538–566. [CrossRef]
32. He, M.; Wang, X.; Wei, P.; Yang, L.; Teng, Y.; Lyu, R. Reinforcement Learning Meets Network Intrusion Detection: A Transferable and Adaptable Framework for Anomaly Behavior Identification. *IEEE Trans. Netw. Serv. Manag.* **2024**, *21*, 2477–2492. [CrossRef]
33. Halbouni, A.; Gunawan, T.S.; Habaebi, M.H.; Halbouni, M.; Kartiwi, M.; Ahmad, R. CNN-LSTM: Hybrid Deep Neural Network for Network Intrusion Detection System. *IEEE Access* **2022**, *10*, 99837–99849. [CrossRef]
34. Khan, M.A.; Khan, M.A.; Jan, S.U.; Ahmad, J.; Jamal, S.S.; Shah, A.A.; Pitropakis, N.; Buchanan, W.J. A Deep Learning-Based Intrusion Detection System for MQTT Enabled IoT. *Sensors* **2021**, *21*, 7016. [CrossRef] [PubMed]
35. De Carvalho Bertoli, G.; Júnior, L.A.; Saotome, O.; Dos Santos, A.L.; Verri, F.A.; Marcondes, C.A.; Barbieri, S.; Rodrigues, M.S.; De Oliveira, J.M. An End-to-End Framework for Machine Learning-Based Network Intrusion Detection System. *IEEE Access* **2021**, *9*, 106790–106805. [CrossRef]
36. Azam, Z.; Islam, M.M.; Huda, M.N. Comparative Analysis of Intrusion Detection Systems and Machine Learning-Based Model Analysis Through Decision Tree. *IEEE Access* **2023**, *11*, 80348–80391. [CrossRef]
37. Ghanem, W.A.H.M.; Jantan, A.; Ghaleb, S.A.A.; Nasser, A.B. An Efficient Intrusion Detection Model Based on Hybridization of Artificial Bee Colony and Dragonfly Algorithms for Training Multilayer Perceptrons. *IEEE Access* **2020**, *8*, 130452–130475. [CrossRef]
38. Xu, C.; Shen, J.; Du, X.; Zhang, F. An Intrusion Detection System Using a Deep Neural Network with Gated Recurrent Units. *IEEE Access* **2018**, *6*, 48697–48707. [CrossRef]
39. Liu, C.; Gu, Z.; Wang, J. A Hybrid Intrusion Detection System Based on Scalable K-Means+ Random Forest and Deep Learning. *IEEE Access* **2021**, *9*, 75729–75740. [CrossRef]
40. Kasongo, S.M. An Advanced Intrusion Detection System for IIoT Based on GA and Tree-Based Algorithms. *IEEE Access* **2021**, *9*, 113199–113212. [CrossRef]
41. Putchala, M.K. Deep Learning Approach for Intrusion Detection System (IDS) in the Internet of Things (IoT) Network Using Gated Recurrent Neural Networks (GRU). Master's Thesis, Wright State University, Dayton, OH, USA, 2017.
42. Xu, W.; Jang-Jaccard, J.; Singh, A.; Wei, Y.; Sabrina, F. Improving Performance of Autoencoder-Based Network Anomaly Detection on NSL-KDD Dataset. *IEEE Access* **2021**, *9*, 140136–140146. [CrossRef]

43. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the IEEE Military Communication Information System Conference (MilCIS), Canberra, Australia, 10–12 November 2015.
44. Martinez-Vega, B.; Tkachenko, M.; Matkabi, M.; Ortega, S.; Fabelo, H.; Balea-Fernandez, F.; La Salvia, M.; Torti, E.; Leporati, F.; Callico, G.M.; et al. Evaluation of Preprocessing Methods on Independent Medical Hyperspectral Databases to Improve Analysis. *Sensors* **2022**, *22*, 8917. [CrossRef]
45. Gao, Y.; Wu, H.; Song, B.; Jin, Y.; Luo, X.; Zeng, X. A Distributed Network Intrusion Detection System for Distributed Denial of Service Attacks in Vehicular Ad Hoc Network. *IEEE Access* **2019**, *7*, 154560–154571. [CrossRef]
46. Idhammad, M.; Afdel, K.; Belouch, M. Semi-supervised machine learning approach for DDoS detection. *Appl. Intell.* **2018**, *48*, 3193–3208. [CrossRef]
47. Elsayed, S.; Mohamed, K.; Madkour, A. A Comparative Study of Using Deep Learning Algorithms in Network Intrusion Detection. *IEEE Access* **2024**, *10*, 1109. [CrossRef]
48. Al-Garadi, M.; Mohammed, A.; Ali, K.; Guizani, M. A Survey of Machine and Deep Learning Methods for Internet of Things Security. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1646–1685. [CrossRef]
49. Shone, N.; Phai, D.; Shi, Q. A Deep Learning Approach to Network Intrusion Detection. *IEEE Trans. Emerg. Top. Comput. Intell.* **2018**, *2*, 41–50. [CrossRef]