information

*Article*

# Deep Learning Approach for SDN-Enabled Intrusion Detection System in IoT Networks

**Rajasekhar Chaganti [1], Wael Suliman [2], Vinayakumar Ravi [2,*] and Amit Dua [3]**

[1] Department of Computer Science, University of Texas at San Antonio, San Antonio, TX 78249, USA
[2] Center for Artificial Intelligence, Prince Mohammad Bin Fahd University, Khobar 34754, Saudi Arabia
[3] Department of Algorithmics and Software, Silesian University of Technology, 44-100 Gliwice, Poland
* Correspondence: vravi@pmu.edu.sa

**Abstract:** Owing to the prevalence of the Internet of things (IoT) devices connected to the Internet, the number of IoT-based attacks has been growing yearly. The existing solutions may not effectively mitigate IoT attacks. In particular, the advanced network-based attack detection solutions using traditional Intrusion detection systems are challenging when the network environment supports traditional as well as IoT protocols and uses a centralized network architecture such as a software defined network (SDN). In this paper, we propose a long short-term memory (LSTM) based approach to detect network attacks using SDN supported intrusion detection system in IoT networks. We present an extensive performance evaluation of the machine learning (ML) and deep learning (DL) model in two SDNIoT-focused datasets. We also propose an LSTM-based architecture for the effective multiclass classification of network attacks in IoT networks. Our evaluation of the proposed model shows that our model effectively identifies the attacks and classifies the attack types with an accuracy of 0.971. In addition, various visualization methods are shown to understand the dataset's characteristics and visualize the embedding features.

**Keywords:** intrusion detection; software defined networks; Internet of Things; deep learning; LSTM; support vector machine; denial of service; network attacks



**Citation:** Chaganti, R.; Suliman, W.; Ravi, V.; Dua, A. Deep Learning Approach for SDN-Enabled Intrusion Detection System in IoT Networks. *Information* **2023**, *14*, 41. https://doi.org/10.3390/info14010041

Academic Editor: Abderrezak Rachedi

Received: 5 December 2022
Revised: 27 December 2022
Accepted: 4 January 2023
Published: 9 January 2023



## 1. Introduction

The IoT technology has progressed to the extent that many people use or interact with smart devices in some form in everyday life. The advantages of the IoT include automation, improved productivity and effective utilization of resources, and more [1]. The number of sensing and remote communication-capable physical devices connected to the internet has increased over the past few years. The IoT device's internet connection growth pattern is the culmination of the research progress in wireless communications, cloud computing, and data analytics in recent years and also the number of applications that IoT offers, including smart homes, smart healthcare, smart transportation systems, smart grid, and many more [2]. Nevertheless, the security and privacy offered by the IoT are still a concern because these devices are more prone to security attacks and vulnerabilities, and adversaries are constantly looking for new ways to compromise the unsecured and vulnerable devices opened in public. Furthermore, the IoT devices are connected to the cloud services through the internet to send updates and receive suggestions or recommendations for taking intelligent actions on behalf of the IoT device users. Therefore, the presence of IoT devices adds more complexity to managing and maintaining networks. The IoT attack vector can lead to successful security breaches, and proactive network security defenses are necessary to protect critical assets and data.

The centralized network resource management capability is an advantage of SDN. The network resources can be managed, and the network traffic can be monitored using the SDN controller to improve the security [3]. Nowadays, SDN is well-equipped to

process not only the legacy protocol network traffic but also IoT devices generated network traffic with OpenFlow protocol and customized SDN controller applications. Although SDN can be used for routing management, resource management, traffic monitoring and management, security detection, and mitigation in IoT environments, SDN is also prone to new attacks due to the presence of IoT devices. For example, the default configured IoT devices are being compromised using Mirai botnet malware and initiate command and control communication to the remote attacker server for becoming a part of the bot army. These compromised bots can be used to generate malicious network traffic and flood the SDN controller. The controller resources or network resources saturation can lead to the shut down of the whole network with distributed denial of service attacks [4]. Numerous other attack possibilities exist, such as topology poisoning [5], compromising the controller or switches with known vulnerabilities, or zero-day attacks in SDN when the IoT devices are in the network. Some of the known existing solutions are Anomaly-based detection, Signature-based detection. Signature-based solutions cannot detect the new attack variants and need to constantly update the malware signatures for up-to-date detection rules. On the other hand, Anomaly-based detection generates many false positives and requires human resources to tune the alerts. These techniques do not have behavioral or pattern learning capability. Therefore, the adaption of Anomaly-based techniques in the network environment is challenging [6,7].

Since computing capabilities have tremendously increased recently, data analytics have become more popular and widely used in many applications. Intrusion detection using ML and DL has received more attention for advanced combat attacks and improved security due to the limitations of traditional intrusion detection systems [8,9] Research community explored the usage of ML techniques for malware detection [10], network intrusion detection [11–13] and botnet attack detection [14]. However, the data analysts should have the domain knowledge to select the features effectively, and feature engineering may consume the time and effort of an analyst. On the other hand, DL techniques can learn the patterns from historical data and adapt the features to predict network intrusions. DL has also been employed in security applications such as intrusion detection [15], malware detection [16,17] and botnet detection [18,19] and showed that DL models performed well. Nevertheless, finding the relevant dataset in a specific domain is challenging, particularly generating datasets in emerging technology cross domains due to a lack of technical expertise and lack of resources or tools to create a testbed. Therefore, the existing or old datasets have been employed to evaluate the performance of the new environments such as SDN-assisted IoT networks [14,20]. In this work, we performed a thorough performance analysis of the DL architectures and traditional ML classifiers on the datasets generated using real SDN and IoT environments. In addition, we proposed a DL approach to detect the attacks using SDN intrusion detection in IoT networks.

The main contributions of this paper are as follows.

- Utilize various DL architectures for detecting network attacks with SDN-based intrusion detection systems in IoT networks.
- Comparative performance analysis of DL architectures along with classical approach support vector machine (SVM) is performed for network attack detection.
- Detailed investigation and analysis of the proposed approach for effective detection and classification of IoT attacks.
- t-SNE feature visualization for the hidden layer of the proposed approach is conducted to ensure that the learned features are meaningful for the detection and classification of SDN IoT attacks.
- To show that the proposed method is generalizable to handle various SDN attacks and robust, the performance analysis was conducted on two different SDN IoT datasets.

The rest of the paper is organized as follows. Section 2 discusses the literature survey of DL techniques in SDN and IoT networks. Section 3 illustrates our proposed DL-based approach for SDN intrusion detection in IoT networks. Section 4 presents the two datasets used for our performance analysis and investigation and explains the dataset generation

methods and features obtained. Section 5 explains the experiments performed on different DL models and the proposed approach, a description of the results obtained, and further discussion on our approach. Section 6 concludes the paper and discusses future work.

## 2. Literature Survey

The centralized control of the network devices in SDN is leveraged to collect the network data and perform data analytics. Zhao et al. [21] have surveyed the utilization of ML algorithms in SDN for network applications. The survey classified the prior artwork in two aspects, ML algorithms used in SDN and Networking applications used in SDN. From the ML perspective, ML algorithm types are discussed first and then mapped with the existing work in SDN using those ML algorithms. The SDN network applications such as resource management, routing management, network flow monitoring, and network security are identified, and the ML algorithms are mentioned in each application. One of the notable challenges mentioned is that ML failed to show the expected performance due to the lack of available training datasets and complex network traffic patterns introduced by SDN architecture. Sultana et al. [22] performed a survey on implementing Network intrusion detection using ML approaches in SDN. The survey mainly discussed the various ML/DL techniques used for network intrusion detection in an SDN environment. Mohammed et al. [23] presented a survey using ML and DL techniques for network traffic management specific to traffic classification and prediction in SDN. The survey categorization is based on the traffic features, network topology, ML/DL techniques, and the simulator used to perform experiments. The authors described the lack of labeled data availability is a significant concern. They also suggested that a Recursive Neural Network (RNN) can be applied as a potential solution for traffic prediction, as the SDN network experiences elephant flows abruptly and needs a technique such as RNN to use historical data to make decisions.

The decoupling nature of the control plane from the data plane in SDN introduces new vulnerabilities and is prone to a new denial of service attacks such as controller saturation attacks. Boppana et al. [4] identified several DDoS vulnerabilities in SDN networks and determined that the existing mitigation solutions introduce new vulnerabilities. Therefore, ML or DL based solutions to detect network attacks are highly desired. Dey et al. [24] studied SDN-based Intrusion detection systems performance by applying various ML techniques and varying the feature selection set. However, the experiments are performed on the NSL-KDD dataset and may not be suitable in SDN and IoT environments. Nguyen et al. [25] proposed an intelligent and collaborative NIDS architecture for SDN-based IoT networks. The authors leveraged three well-known network datasets to apply ML/DL techniques for detecting the attacks in the SDN-based proposed architecture. Alzahrani et al. [26] used classical and advanced tree-based ML techniques such as decision tree, random forest, and XGBoost for traffic monitoring in the network intrusion detection deployed an SDN controller. The dataset NSL-KDD was considered, and chosen 5 out of the 41 features for conducting the experiments and achieved an accuracy of 95.95%. Birkinshaw et al. [27] implemented Credit-Based Threshold Random Walk (CB-TRW) and Rate Limiting (RL) techniques as part of the Intrusion detection and prevention system to defend against port scanning attacks and considered Quality of Service (QoS) as a DDoS mitigation scheme as part of the mitigation strategy. However, manual tuning is required per the network environment to keep the false positives low. Sebbar et al. [28] built a context-based node acceptance based-random forest model (CBNA-RF) for setting up security policies and automation in large-scale SDN infrastructure so that the Man in the Middle (MitM) attacks can be quickly detected without affecting the performance. This work is limited to only detecting MitM attacks.

DL techniques have received more popularity recently owing to the ability to produce high-performance results and the availability of computing resources to run DL-based models. Tang et al. [29] applied the DNN technique for flow-based anomaly detection in an SDN environment. The NSL-KDD dataset was chosen, and six features were selected out of 41 for performing the experiments. The accuracy of 75.75% was achieved when

using the DNN, which is not optimal for DL models. Hannache et al. [30] presented a Neural Network-based Traffic Flow Classifier (TFC-NN) for live DDoS detection in an SDN environment. Live migration is applied to defend the attacks when the DDoS attack is detected. This work is the only application for detecting and mitigating DDoS attacks. Hande et al. [31] proposed Convolution Neural Network (CNN) based network intrusion detection in SDN. The NIDS module is implemented in the SDN controller, and six features are considered for training the KDD99 dataset. However, the performance of the proposed model has not been tested. Tang et al. [32] presented a DL approach for flow-based anomaly detection in SDN. The authors used DNN and Gated Recurrent Neural Network (GRU-RNN) for classifying the network attacks. NSL-KDD dataset was chosen for evaluating the DL models, and the features have been categorized into three sets. These are the Basic feature set, traffic feature set, and mixed feature sets and are used for evaluating the role of each set of features in SDN. The performance results show that 80.7% and 90% accuracy were achieved for the model DNN and GRU-RNN, respectively. Overall, our analysis of using ML and DL in an SDN network shows that the used datasets NSL-KDD, CAIDA, and UNSW-NB15 were originally generated using a traditional network setup and the performance of the applied few DL models in [29,32] still need to be improved.

The number of IoT devices connected to the internet is expected to be 30 billion units by the end of 2025 [33]. As the Software Defined-Wide Area Networks (SD-WAN) are deployed in enterprise organizations, it is not uncommon that the IoT device traffic needs to be processed and passed through SDN infrastructure. Wani et al. [34] used the DL classifier to perform anomaly-based SDN intrusion detection for IoT. The proposed intrusion detection system comprises three components Activity monitor, Activity analyzer, and Classifier for traffic capturing, feature extraction, feature reduction, and apply DL algorithms for anomaly detection. Li et al. [35] described an Artificial Intelligence-based two-stage intrusion detection system for Software-defined IoT networks. The network flow traffic is collected from the SDN controller, applied Bat algorithm with swarm division and binary differential mutation to select the best features, and the flow classification is conducted using a modified random forest with the weighted voting mechanism. However, the selected KDD99 dataset for the evaluation of the proposed approach could not cover attacks targeted in SDN IoT networks. In [36], the authors presented another two-stage intrusion detection system in SDN IoT networks. The feature selection mechanism includes an improved firefly algorithm for search strategy and the ensemble learning algorithms C4.5 decision tree, multi-layer perception (MLP), and Instance-Based learning for the wrapper feature selection. The final feature subset is selected based on the principle minority obeying the majority. The traffic classification prediction is made using the weighted voting method. Nevertheless, the datasets NSL-KDD and UNSW-NB15 were generated using traditional network topology targeting the well-known network attacks. The applicability of this data in SDN and IoT environments is questionable.

Sriram et al. [19] proposed DL based bot detection framework to detect IoT Botnet attacks in network flows. The datasets were generated based on the Mirai and BASHLITE IoT attack tools to mimic the attacks in the IoT environment and used for the experiments. They determined that the DNN technique with four hidden layers outperformed the conventional ML techniques. However, this work is mainly focused on detecting IoT Botnet attacks. In [37], a two-stage DL framework has been proposed to detect botnet attacks using Domain Name System (DNS) queries specific to IoT smart city applications. Siamese networks are applied to find the similarities between the domain names in the first stage for early homoglyph attack detection, and the dissimilar domain names are passed to the cost and performance-effective DL architecture in the second stage to classify the domain name as part of the Domain Generation Algorithm (DGA) family or not. This work is focused on bot attack detection using DNS protocol. Vinayakumara et al. [15] implemented a hybrid intrusion detection alert system to analyze the network and host activities using a highly scalable DNN framework. The TCP connection records are extracted from network traffic used for NIDS attack classification. The system calls are collected for each host process used

for Host Intrusion Detection System (HIDS) attack classification in the proposed hybrid intrusion detection system. These works are not applicable to intrusion detection in SDN based IoT networks.

Table 1 shows the state-of-the-art work leveraged ML and DL techniques for intrusion Detection systems in SDN and IoT network environments. As shown in Table 1, most works considered the analytic data models to detect network intrusion in SDN and non-IoT environments. Most of the works did not include IoT-based network attacks and did not generate the datasets in the presence of IoT network traffic. Most of those works considered the non-IoT Datasets such as NSL-KDD, KDDCup'99, and UNSW-NB15 to train and evaluate the performance of network attack detection and classification. These datasets were created in a traditional network environment setup and did not include the latest attack trends. The IoT device uses a different set of protocols such as Message Queuing Telemetry Transport (MQTT) and Advanced Message Queuing Protocol (AMQP), and the datasets generated in the presence of the IoT network traffic are needed to test and evaluate the attacks in IoT networks correctly. A few works proposed in the SDN and IoT network environment used the non-IoT datasets, such as CSE-CIC-IDS2018, SDN-NF-TJ, and SDN-IoT. We have been inspired to utilize the SDN and IoT environment-based datasets and proposed a deep learning model to improve the network attack detection and classification in the SDN IoT networks.

**Table 1.** State-of-the art ML and DL based IDS in SDN IoT networks.

| Network Environment | Dataset | Technique | Accuracy |
|---|---|---|---|
| SDN-based non-IoT [24] | NSL-KDD | Gain ratio, Random Forest | 81.9% |
| SDN-based non-IoT [26] | NSL-KDD | XGBoost | 95.5% |
| SDN-based noN-IoT [29] | NSL-KDD | DNN | 75.75% |
| SDN-based non-IoT [38] | InSDN | CNN+RF | 97% |
| SDN-based non-IoT [39] | Custom dataset | SVM | 95.24% |
| SDN-based non-IoT [40] | NSL-KDD | SVM | 95.98% |
| SDN-based non-IoT [30] | Custom dataset | Neural Network | 96.13% |
| SDN-based non-IoT [34] | CSE-CIC-IDS2018 | IDSIoT-SDL | 96.05% |
| SDN-based non-IoT [35] | KDDCup-99 | Bat Algorithm, Random Forest | 96.03% |
| SDN-based non-IoT [36] | UNSW-NB15 | Improved firefly Algorithm, ensemble classifier | 88.46% |
| SDN-based IoT (Proposed) | SDN-IoT, SDN-NF-TJ | LSTM | 97.1% |

## 3. Proposed DL Based Approach for SDNIoT Environment

The proposed DL-based approach in SDN intrusion detection for IoT attacks collects the network flows from the OpenFlow virtual switches, which are connected to the SDN controller. The network connection statistics are extracted using an SDN application on top of the SDN controller. The network connection records are ground truth labeled while generating the network traffic. The datasets are thoroughly evaluated to select the optimum model used for IoT attack detection.

### *3.1. Selection of the Deep Learning Model*

Firstly, the classical ML model SVM is applied to the network dataset for performance evaluation because the model is extensively studied for network intrusion detection in traditional networks and is relatively memory efficient. In order to identify the best performance model and also the prevalence of DL techniques to eliminate the feature engineering and feature extraction process, we have considered three artificial neural network models for our intrusion detection evaluation, and those are DNN, CNN, and LSTM. DNN is a feed-forward network in which the information always moves in the forward direction from neurons in one layer to another, and connections between the neurons do not form a cycle. Each neuron will compute an activation function. If the incoming neuron value exceeds a threshold, the output is passed through the next connected layer by activate function; otherwise, the value is ignored. The two connected neurons

are associated with weight, and the weight values influence the input to the output for the next neuron. CNN is also a feed-forward network and addresses the regularization problem by considering the hierarchical patterns of the data and assembling the complex patterns using self-learning filters. CNN is mainly used in image classification because less preprocessing is required compared to other classification algorithm. We have chosen a single hidden layer convolution model for our comparative analysis. LSTM is an artificial recurrent neural network and comprises feedback connections, unlike DNN. LSTM is generally well suited to classification, process, and predicting the time series data because LSTM cells have memory. Therefore, LSTM can also be well suited for intrusion detection applications. These three models are tested by varying the number of hidden layers and learning parameters for evaluation and comparative analysis.

### *3.2. Description of Proposed Deep Learning Model*

After the extensive performance analysis of the considered models for this study and the natural tendency of the LSTM properties suited for the time series data classification, we select the LSTM four hidden layer architecture for the network attacks classification in the SDN IoT network.

A standard LSTM memory cell contains input, output, and forget gates. Read and write access to the cell is controlled by an input gate and output gate. The cell reset is handled by forget gate. These cells are the building blocks of the LSTM layer. Figure 1 illustrates all the layers, input, and output shapes of the layers in the proposed LSTM architecture. It contains 4 LSTM layers with 64 units, and four dropout layers to process the input dataset with $n$ records and $c$ features. The last dropout layer output is fed into the dense layer. The dense layer is a densely connected layer in which every neuron is connected to all the neurons in the previous layer. This layer helps to change the dimensions and also does not lose the learning information from the previous layers. The dense layer connected to the last dropout layer has five neurons representing the network attack types. Finally, softmax is used for multiclass classification of the outputs. This model is also applied for binary classification of attacks by changing the last layer to use *sigmoid* function.

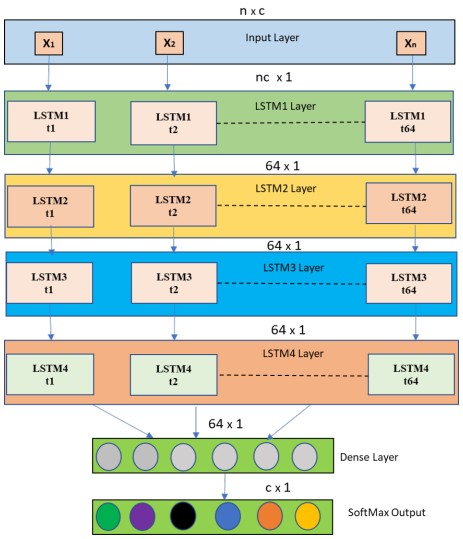



**Figure 1.** Proposed LSTM architecture for network attacks classification in SDN IoT Network.

The pseudocode for the proposed LSTM four-layer architecture is shown in Algorithm 1. The dataset rows $n$ is divided into $x$ batches with each batch size $b$. The model weights are updated for each batch size processing, and the training process ends when the number of batches reaches epoch $s$ multiplied by several batches $x$.

**Algorithm 1:** Network traffic multiclass classification in IoT network

**Input:** Network traffic dataset $\{r_1, r_2, \ldots, r_t\}$; batchsize:b; number of batches:n; total dataset batches = b*n; epoch:s
**Output:** Labels $\{y_1, y_2, \ldots, y_6\}$
**for** $i \leftarrow 1$ **to** $n$ **do**
  **for** $j \leftarrow 1$ **to** $b$ **do**
    $Standardized_{ij} \leftarrow$ Preprocessing (ij);
    $L1_{i1}, \ldots, L1_{ib} \leftarrow$ LSTM ($Standardized_{ij}$);
    $D1_{i1}, \ldots, D1_{ib} \leftarrow$ Dropout ($L1_{i1}, \ldots, L1_{ib}$);
    $L2_{i1}, \ldots, L2_{ib} \leftarrow$ LSTM ($D1_{i1}, \ldots, D1_{ib}$);
    $D2_{i1}, \ldots, D2_{ib} \leftarrow$ Dropout ($L2_{i1}, \ldots, L2_{ib}$);
    $L3_{i1}, \ldots, L3_{ib} \leftarrow$ LSTM ($D2_{i1}, \ldots, D2_{ib}$);
    $D3_{i1}, \ldots, D3_{ib} \leftarrow$ Dropout ($L3_{i1}, \ldots, L3_{ib}$);
    $L4_{i1}, \ldots, L4_{ib} \leftarrow$ LSTM ($D3_{i1}, \ldots, D3_{ib}$);
    $D4_{i1}, \ldots, D4_{ib} \leftarrow$ Dropout ($L4_{i1}, \ldots, L4_{ib}$);
    $D5_1, \ldots, D5_6 \leftarrow$ Dense ($D4_{i1}, \ldots, D4_{ib}$);
    $y_1, \ldots, y_6 \leftarrow$ Softmax ($D5_1, \ldots, D5_6$);
  **end**
**end**

## 4. Datasets Description

Two SDN environment-specific data sets were considered to perform the experiments and evaluate the proposed LSTM approach. The first dataset, named "DS1", is extensively used to determine various deep learning models performance for classifying the network attacks in the SDN-based IoT network. The dataset DS1 contains the binary labeled data for attack detection and multiclass labeled data for network attacks classification. The DS1 dataset contains 33 features, and each record in the dataset represents the network connection stats along with attack classification label. The feature details of the DS1 are shown in Figure 2. Most of these features is extracted from the characteristic of a network flow, which is a combination of 5-tuple values (source IP, destination IP, source port, destination port, protocol). The DS1 dataset is split into train and test datasets with 70% and 30%, respectively. Table 2 shows the total number of samples in the training and test category for datasets DS1 and DS2. We also applied the ML/DL models on the second dataset named "DS2" for validation and assess the generalization of our model to other SDN IoT datasets. DS2 contains binary labeled SDN network connection records captured for distributed denial of service (DDoS) attack detection in the network.

The DS1 dataset is generated using the mininet SDN simulation environment [41]. The environment is composed of legitimate and non-legitimate devices, in particular, 5 IoT devices with 2 hosts for legitimate users and 4 hosts for attackers. An open flow switch was used in the simulated network environment to connect all the devices with SDN controller open network operating system. The network environment used the Node-red tool to simulate the traffic from IoT devices such as smart thermostat, motion-activated lights, weather station, remotely activated garage door, and smart fridge. USing the network environment, the authors generated two types of datasets such as one dataset with 5 IoT devices and another dataset with 10 IoT devices. The experiments reported in this paper consider the dataset collected from 5 IoT devices. However, to evaluate the robustness and generalization of the model in machine learning and deep learning, the model effectiveness of the 5 IoT devices can be tested on the dataset from 10 IoT devices. The 5 IoT devices datasets such as DS1 contains DoS, port scanning, DDoS, fuzzing and OS fingerprinting attacks. These attack types are generated using the Nmap, boofuzz, and fuzzing tools. Using the SDN applications, the authors collected the traffic flows from the switches with an SDN-based controller. The five-node dataset initially contains 27.9 million entries, which covers all the five attacks simulated in the network. The final dataset is made up of

combining the 35,000 records from each attack category and normal traffic. Table 3 depicts the data samples for normal and each attack category in dataset DS1.

| Feature | Description |
|---|---|
| srcMAC | Source physical address |
| dstMAC | Destination physical address |
| srcIP | Source IP |
| dstIP | Destination IP |
| srcPort | Source port |
| dstPort | Destination port |
| last_seen | Last time the network flow traffic seen |
| Protocol | Transaction protocol name |
| proto_number | Transaction protocol number |
| Dur | Network flow duration |
| Mean | Average duration of network flows |
| Stddev | Standard deviation of network flows |
| Min | Min duration of network flows |
| Max | Maximum duration of network flows |
| Pkts | Total number of packets for the network flow |
| Bytes | Total number of bytes for the network flow |
| Spkts | Source initiated packet count |
| Dpkts | Destination initiated packet count |
| Sbytes | Source initiated packet byte count |
| Dbytes | Destination initiated packet byte count |
| Srate | Source initiated packets per second |
| Drate | Destination initiated packets per second |
| TnBPSrcIP | Total Number of bytes per source IP |
| TnBPDstIP | Total Number of bytes per Destination IP |
| TnP_PSrcIP | Total Number of packets per source IP |
| TnP_PDstIP | Total Number of packets per Destination IP |
| TnP_PerProto | Total Number of packets per protocol |
| TnP_Per_Dport | Total Number of packets per destination port |
| N_IN_Conn_P_SrcIP | Number of incoming connections per source IP. |
| N_IN_Conn_P_DstIP | Number of incoming connections per destination IP. |
| Attack | 0:Normal traffic, 1:Attack Traffic |
| Category | 1:DoS, 2:DDoS, 3:Port Scanning, 4:OS Fingerprinting, 5:Fuzzing |

**Figure 2.** DS1 Features [41].

The DS2 dataset is captured by enabling the Netflow protocol in the SDN controller [42]. When the client sends traffic through the SDN network, all the new connection flows will be sent to the controller. This dataset covers the malicious flows performing DDoS attempts on the SDN controller. The training and test dataset contains 126,000 and 31,500 samples, respectively in the dataset DS2, as shown in Table 2. The dataset DS1 and DS2 attack and normal sample proportions are listed in Table 2.

The datasets have been preprocessed prior to applying the DL models. The preprocessing steps include removing the network connection IP address and MAC identifiers, labeling the network connection protocol type, removing the trivial features such as timestamp, and normalizing the features by removing the mean and scaling to unit variance. The object types "srcMAC", "dstMAC", "srcIP", "dstIP" columns are deleted from the DS1. The one

hot encoding is used to categorize the protocol "TCP", "UDP" and "ICMP". The "last_seen" feature is removed, as this timestamp will not be helpful for attack detection. The "Nan" values are replaced with the median values as part of the sanitation. The Standard scalarT function is used for normalizing the features.

**Table 2.** DS1 and DS2 normal and attack label statistics.

| Dataset | Attack | Normal |
|---|---|---|
| DS1 | 175,000 | 35,000 |
| DS2 | 94,500 | 63,000 |

**Table 3.** Dataset DS1 multiclass statistics.

| Label | Sample Count |
|---|---|
| Normal | 35,000 |
| DoS | 35,000 |
| DDoS | 35,000 |
| Port Scanning | 35,000 |
| OS Fingerprinting | 35,000 |
| Fuzzing | 35,000 |

## 5. Experimental Evaluation and Discussion

### *5.1. Software and Hardware Preliminaries*

The ML library scikit-learn (https://scikit-learn.org/stable/ accessed on 15 February 2022) and the deep learning API keras (https://keras.io/ accessed on 15 February 2022) with TensorFlow (https://www.tensorflow.org/ accessed on 15 February 2022) as backend in python are used to conduct the experiments. The python scripts run in virtual machine 64-bit Ubuntu operating system 20.04 LTS with a 4 GB RAM memory configuration and Intel Core i5-4210U CPU@1.70 GHz.

### *5.2. Evaluation Metrics*

The current work evaluates the performance of machine learning and deep learning models using the following statistical metrics.

Accuracy: it is estimated by dividing the sum of TP and TN by the sum of TP, TN, FP, and FN.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Precision: it is estimated by dividing the TP by the sum of TP and FP.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Recall or true positive rate (TPR): it is estimated by dividing the TP by the sum of TP and FN.

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

F1-Score: it is the harmonic mean of Precision and Recall.

$$F1\text{-}Score = \frac{2 * Recall * Precision}{Recall + Precision} \tag{4}$$

False Positive Rate FPR)—it is calculated by dividing the total of incorrect classification of the attack class by the sum of incorrect classification of the attack class and the correct classification of the normal class.

$$FPR = \frac{FP}{FP + TN} \tag{5}$$

The terms TP, TN, FP, and FN are taken from a confusion matrix. A confusion matrix is a table of predicted and actual values and the dimension of a confusion matrix is the number of classes in the dataset X number of classes in the dataset.

- True Positive (TP)—A sample belonging to the Attack class is correctly predicted as Attack by the model
- False Positive (FP)—A sample belonging to the Attack class is predicted as Normal by the model
- True Negative (TN)—A sample belonging to the Normal class is correctly predicted as Normal by the model
- False Negative (FN)—A sample belonging to Normal traffic is predicted as an Attack by the model.

Area Under Curve (AUC)—Receiver Operative Characteristics (ROC) shows the performance of a classification model at all classification thresholds. The ROC is a plot between the True positive rate and False positive rate, and a lower classification threshold means increasing both the false positive and true positives. The Area under the curve is the size of the Area under the ROC curve and provides aggregated performance of all the possible classification thresholds.

$$AUC = \int_0^1 \frac{TP}{TP + FP} d\frac{FP}{FP + TN} \tag{6}$$

As mentioned earlier, The real SDN testbed with IoT traffic simulated dataset DS1 has been considered for the model performance experiments. The dataset DS1 is processed to remove the network device representation object type features such as MAC address and IP addresses and also removed the last_seen timestamp based on our estimation of low confidence for network attacks classification. Further, feature selection efforts are not made because our objective is to employ deep learning models. We have selected 25 features for our test experiments in dataset DS1. The attack category feature is labeled as 0:Normal, 1:DoS, 2:DDoS, 3:Port Scanning, 4:OS Fingerprinting, and 5:Fuzzing in DS1. The DS1 dataset is split into training and testing for performance prediction. The supervised model SVM with RBF kernel is chosen because SVM is used extensively for intrusion detection in traditional networks [43]. RBF kernel is chosen over linear kernel because training time and testing time for SVM model with RBF kernel is much lower than linear kernel in our case; no significant performance differences are observed in RBF and linear kernel usage. Another reason for selecting the RBF over the linear kernel is that the number of features is smaller than the sample size in our dataset. The SVM parameters such as regularization parameter "C" and kernel coefficient "gamma" are selected as default values one and "scale" respectively. The parameter default values were selected because no notable performance impact was seen when varying these parameters for SVM.

A simple feed-forward network DNN, CNN, and LSTM with hidden layers varying from 1 to 4 are chosen to evaluate the DL architecture performance for attack detection in IoT networks. DNN contains a dense layer as a hidden layer with 1024 neurons followed by a layer dropping out 1 in 100 output neurons. As the number of hidden layers increases in DNN, the neurons are reduced by 256 in each layer. The most commonly used *ReLU* is used as an activation function for DNN models. The loss function "loss_entropy" is chosen under the inference framework maximum likelihood. The "binary_entropy" and "categorical_entropy" parameters are selected for binary and multiclass classification experiments. The optimizer *adam* is considered for all the models. CNN model comprises a convolution layer followed by a max pool layer for downsampling the input. The input is also flattened and does not impact the batch size. Subsequently, a dense layer is applied with *ReLU* activation function. Finally, the binary or multiclass classification is based on another dense layer with an activation function as *sigmoid* or *sigmax*. We have opted for only one hidden layer case for CNN in our experiments, as the CNN is designed and best works for image classification. LSTM architectures comprise an LSTM layer with output

data size varying from 4 to 32, as the hidden layers increased from 1 to 4. Each LSTM layer is followed by a dropout layer of 10 in 100 input values. Finally, a dense layer with 1 or 6 output values size based on the binary or multiclass evaluation and consequently *sigmoid* or *sigmax* activation function applied to obtain the final binary or multiclass output prediction. The batch size for DNN architectures is chosen to be 64 for both binary and multiclass classification. On the other hand, the batch size for selected LSTM architectures is 32. After running experiments by varying the epoch value, we fixed the final epoch value 200 for all the DL models with different architectures.

The training accuracy and loss graphs for the DNN, CNN, LSTM model binary classification and multiclass classification are shown in Figures 3a, 4a, 5a and Figures 3b, 4b, 5b respectively. The naming convention of the DL models DNN and LSTM in the paper is given based on the number of layers considered in the model. DNN1 has one dense layer, whereas DNN4 includes four dense layers. Similarly, LSTM1 has one LSTM layer, and LSTM4 contains four LSTM layers. Figure 3a shows that the DNN2, DNN3, and DNN4 training accuracy performance was good compared to DNN1. As the number of layers in DNN increases, the performance accuracy of the models also increased. Nevertheless, there is no significant increase in accuracy in DNN 3 and DNN4. Thus, we decided to stop at the DNN4 model. In addition, most of the DNN models have achieved training accuracy in the range of 95 to 97% within 75 epochs, and after that there is no significant increase in the training accuracy. To understand the model performance, the model loss plots also included in the figures.

The same DNN model architectures were used to run experiments on the multiclass dataset and classified the attacks into different categories. As seen in Figure 3, the DNN has obtained almost the same accuracy, and it indicates that the DNN model was robust in handling multiclass data.

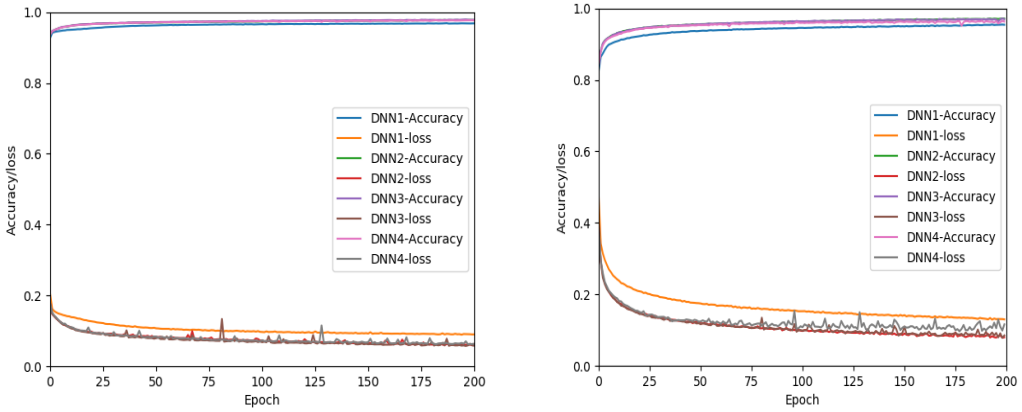



**Figure 3.** Accuracy loss plots of DDN on Dataset DS1. (**a**) DNN binary classification; (**b**) DNN multi-class classification.

Figure 4a shows that CNN has achieved training accuracy of 94 to 96% within 80 epochs and settled to 97% by the end of 200 epochs. The Figure 4a indicates that CNN training accuracy is comparable to DNN4 for binary classification. Based on Figure 4a,b, we also observe a significant difference in the training accuracy between the binary and multiclass classification for the CNN model. We have not considered adding more layers to the CNN based on the model loss values for the multiclass classification. As shown in Figure 5a, the LSTM model improved the training accuracy noticeably from 95 to 97% after 200 epochs run for the binary classification of the dataset. The same trend followed for the multiclass classification case as seen in Figure 5b except for LSTM1. It is evident that LSTM models with more than two hidden layers achieved good training accuracy in binary and multiclass cases. Overall, based on the Figures 3–5, we can also conclude that the tested

models with 200 epochs do not suffer from underfitting or overfitting problem. It is also evident that the proposed four-layer LSTM performed well in 200 epochs run for attack detection in the IoT network.

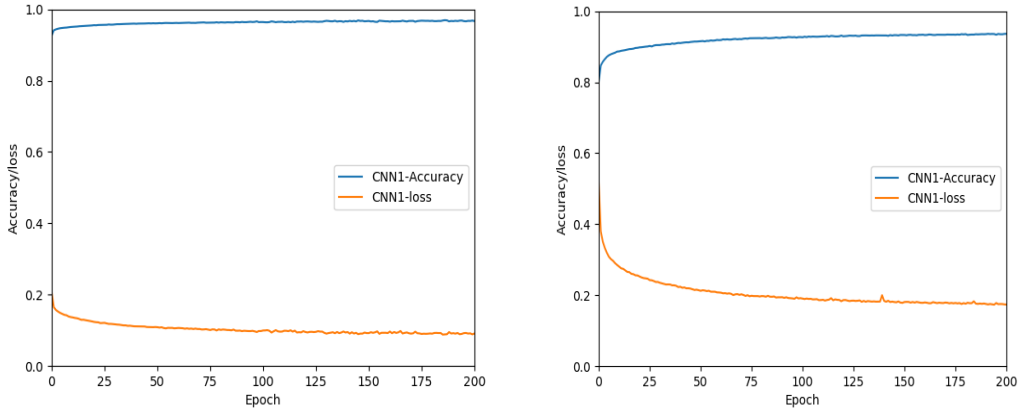



**Figure 4.** Accuracy loss plots of CNN on Dataset DS1. (**a**) CNN binary classification; (**b**) CNN multi-class classification.

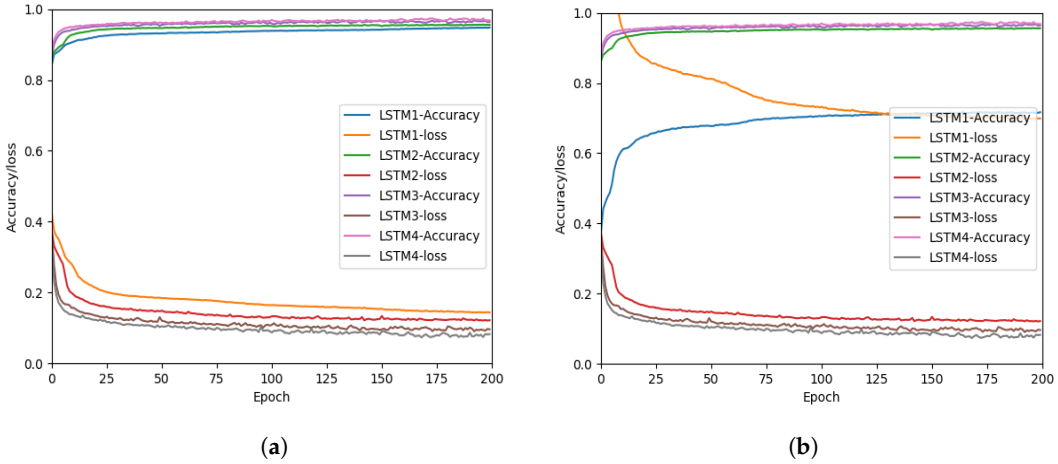
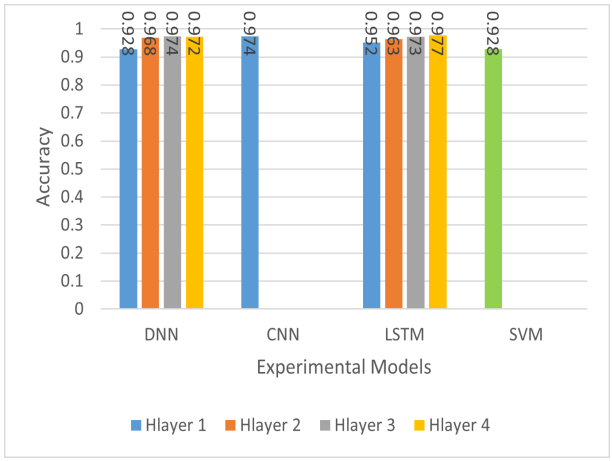
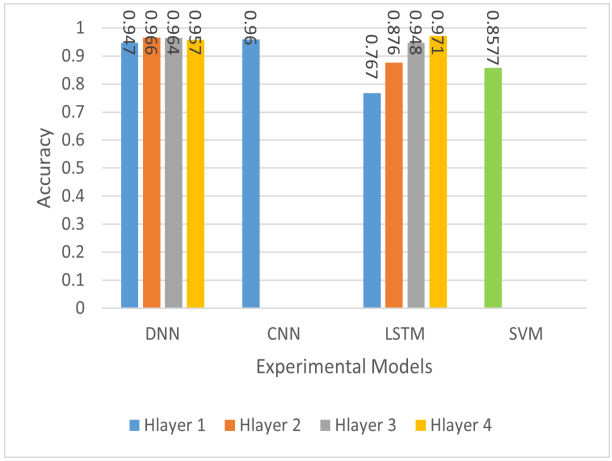



**Figure 5.** Accuracy loss plots of LSTM on dataset DS1. (**a**) LSTM binary classification; (**b**) LSTM multi-class classification.

The SVM and DL model architecture's comparative accuracy performance is shown in Figure 6 for the binary and multiclass classification of attacks in IoT networks. These plots confirm that DL architectures outperformed the supervised learning SVM except LSTM1 multiclass case. The DNN4 model has outperformed SVM because the deep neural network learns the input dataset and adjust the weights accordingly to classify attacks and attack type effectively. On the other hand, SVM divides the data space and creates a decision boundary to classify the attacks when it is trained. Since SVM cannot adapt the learning capability, it is likely to misclassify the attacks. Lack of hidden layers in LSTM1 has impacted the accuracy of the LSTM1 multiclass classification model.




(**a**)




(**b**)

**Figure 6.** Accuracy comparison of the SVM and DL models used on Dataset DS1. (**a**) Accuracy comparison of models for binary classification. (**b**) Accuracy comparison of models for multi-class classification.

Table 4 shows the performance metrics precision, recall, and F1-Score, including both macro and average scores of all the tested ML and DL models. The performance of DL models for both binary and multiclass classification is listed in Table 4. The macro average of a given metric is to compute the metric for each label, then determine the average without considering the proportion for each label in the dataset. The weighted average of a given metric has computed the metric for each label, then determined the average considering the proportion for each label in the dataset.

Table 4 reports that the LSTM4 obtained the best precision in attack detection or binary classification, whereas the CNN-LSTM is the least performed model with a precision of 0.76. The DNN's best-performed and LSTM3 models performed equally well for binary classification. However, our proposed LSTM4 model performs better than DNN's best-performed model in attack detection or binary classification.

**Table 4.** Performance results of the binary and multiclass classification.

| Architecture | Precision | Recall | F1-Score | Score |
|---|---|---|---|---|
| | | **Binary** | | |
| **CNN-LSTM** | 0.76 | 0.78 | 0.77 | Macro |
| | 0.86 | 0.86 | 0.86 | Weighted |
| **DNN** | 0.97 | 0.94 | 0.95 | Macro |
| | 0.97 | 0.97 | 0.97 | Weighted |
| **CNN** | 0.96 | 0.94 | 0.95 | Macro |
| | 0.97 | 0.97 | 0.97 | Weighted |
| **LSTM1** | 0.94 | 0.88 | 0.91 | Macro |
| | 0.95 | 0.95 | 0.95 | Weighted |
| **LSTM2** | 0.95 | 0.91 | 0.93 | Macro |
| | 0.96 | 0.96 | 0.96 | Weighted |
| **LSTM3** | 0.95 | 0.95 | 0.95 | Macro |
| | 0.97 | 0.97 | 0.97 | Weighted |
| **LSTM4** | 0.97 | 0.95 | 0.96 | Macro |
| | 0.98 | 0.98 | 0.98 | Weighted |
| | | **Multiclass** | | |
| **SVM** | 0.86 | 0.86 | 0.86 | Macro |
| | 0.86 | 0.86 | 0.86 | Weighted |
| **DNN** | 0.95 | 0.95 | 0.95 | Macro |
| | 0.95 | 0.95 | 0.95 | Weighted |
| **CNN** | 0.96 | 0.96 | 0.96 | Macro |
| | 0.96 | 0.96 | 0.96 | Weighted |
| **LSTM1** | 0.78 | 0.77 | 0.76 | Macro |
| | 0.78 | 0.77 | 0.76 | Weighted |
| **LSTM2** | 0.88 | 0.88 | 0.88 | Macro |
| | 0.88 | 0.88 | 0.88 | Weighted |
| **LSTM3** | 0.95 | 0.95 | 0.95 | Macro |
| | 0.95 | 0.95 | 0.95 | Weighted |
| **LSTM4** | 0.97 | 0.97 | 0.97 | Macro |
| | 0.97 | 0.97 | 0.97 | Weighted |

The multiclass classification of the SVM and the best performed proposed model using the confusion matrix as shown in the Tables 5 and 6. The Tables 5 and 6 shows that the confusion matrix is closer to the diagonal matrix for the ideal classification scenario in the proposed LSTM4 approach than the SVM model. The small percentage of notable misclassified data falls under normal traffic with label 0 considered as port scanning with label 3 (346) and OS fingerprinting with label 4 (282) in the LSTM4 model. These false positives do not need any tuning efforts in a real-time network intrusion detection system, as the source IP whitelisting is not a viable option. The network scanning attempts usually ignored due to the high volume of these events unless impacting or disrupting the services such as denial of service attacks.

**Table 5.** SVM Confusion Matrix.

| | | Predicted Classes | | | | | |
|---|---|---|---|---|---|---|---|
| | | **0** | **1** | **2** | **3** | **4** | **5** |
| Actual Classes | 0 | 7320 | 161 | 85 | 593 | 297 | 1956 |
| | 1 | 28 | 10,169 | 162 | 12 | 13 | 2 |
| | 2 | 109 | 146 | 10,261 | 20 | 26 | 101 |
| | 3 | 49 | 0 | 7 | 7606 | 2621 | 125 |
| | 4 | 517 | 10 | 1 | 1411 | 8348 | 348 |
| | 5 | 162 | 0 | 0 | 0 | 0 | 10,334 |

**Table 6.** LSTM4 Confusion Matrix.

| | | Predicted Classes | | | | | |
|---|---|---|---|---|---|---|---|
| | | **0** | **1** | **2** | **3** | **4** | **5** |
| Actual Classes | 0 | 9602 | 2 | 10 | 346 | 282 | 170 |
| | 1 | 0 | 10,351 | 33 | 1 | 1 | 0 |
| | 2 | 0 | 39 | 10,622 | 1 | 1 | 0 |
| | 3 | 193 | 2 | 0 | 9935 | 273 | 5 |
| | 4 | 335 | 2 | 2 | 36 | 10,251 | 9 |
| | 5 | 33 | 0 | 0 | 5 | 4 | 10,454 |

Receiver Operating Characteristic (ROC) for the proposed LSTM model architecture in binary and multiclass classification are shown in Figures 7 and 8 respectively. The proposed approach achieved 0.99 AUC in classifying the network connection records as either attack or normal. In addition, the model has shown 0.993 for Normal, 0.999 for DoS, 0.999 for DDoS, 0.998 for Port Scanning, 0.998 for OS Fingerprinting, and 0.99 for fuzzing in classifying the SDN IoT attacks into different categories. Even in the multiclass category, the proposed method has shown above 0.999 AUC for most of the attack classification. This indicates that the proposed method is robust and can achieve better performances in both binary and multiclass classification. Overall, the Figures 7 and 8 show that the performance is closer to the ideal case, with the graph closer to the reverse "L" shape in both attack detection and attack type classification.

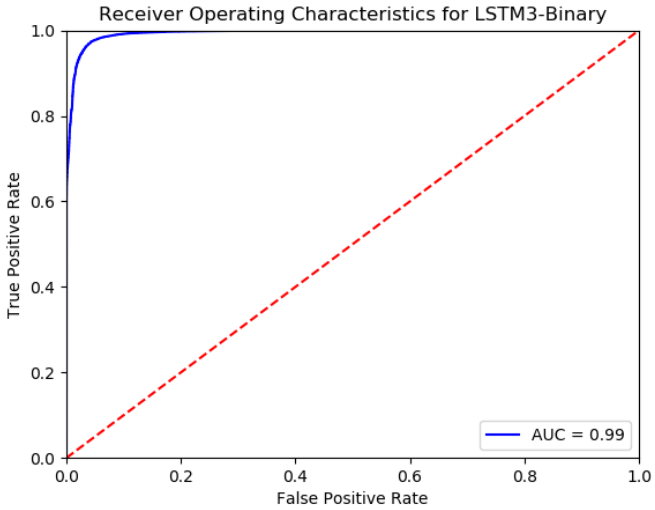



**Figure 7.** ROC Characteristics of the proposed model—binary classification.

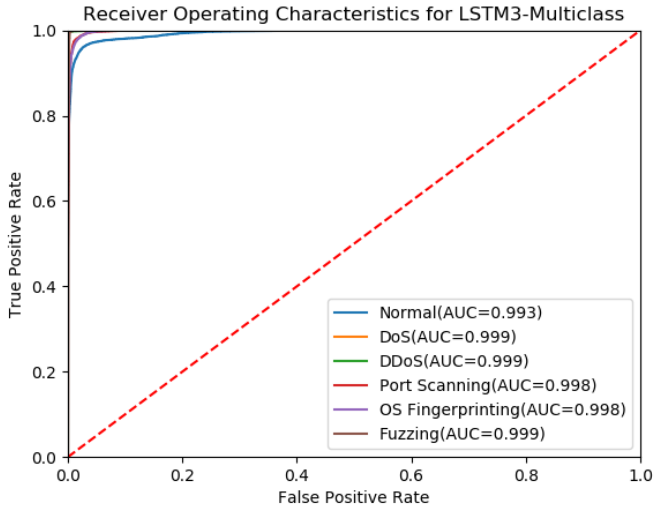



**Figure 8.** ROC Characteristics of the proposed model—multiclass classification.

### *5.3. t-SNE Feature Visualization*

The DL models may contain more than one hidden layer, and interpreting the hidden layers learning ability is challenging. These models act like a black box with minimal information on how the input data is processed to obtain optimal results. t-Distributed Stochastic Neighbor Embedding (t-SNE) is a feature visualization technique used to understand what the DL models learn across different hidden layers. The t-SNE visualization method reduces high-dimensional data into lower dimensions and represents the data in two dimensions, similar to Principal Component Analysis (PCA). We extracted the penultimate layer of the proposed model features for binary and multiclass classification. These were fed into t-SNE to represent two dimensions, and the corresponding plots are shown in Figure 9 for binary and Figure 10 for multiclass classification. Figure 9 indicates that the normal and attack traffic data appeared in different clusters with minimal overlapping. This shows that the model learned the parameters to distinguish the normal traffic or attack traffic. As shown in Figure 10, the attack types formed the clusters to distinguish each other. However, most of the attack types split the clusters into two or more may indicate that the model learns multi patterns of the features for the same attack type.

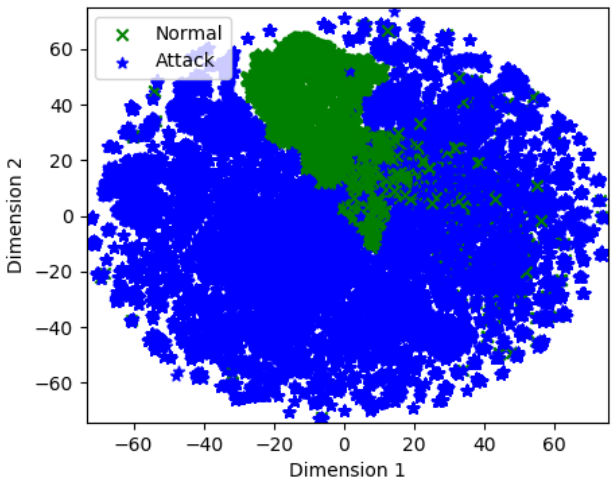



**Figure 9.** t-SNE of the proposed model for binary classification.

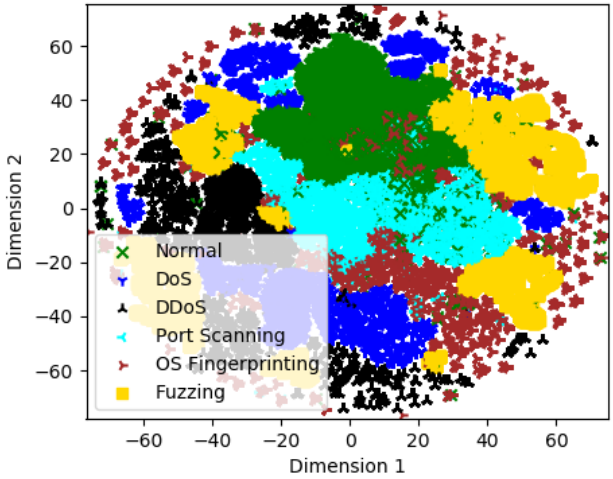



**Figure 10.** t-SNE of the proposed model for multiclass classification.

### *5.4. Generalization of Our Model*

We have also evaluated the proposed model in the dataset DS2 to test the effectiveness and generalization of our model. As mentioned in the Section 4, the dataset DS2 contains DDoS attacks and normal data samples with binary classification ground truth labeled data. Tables 7 and 8 depict the classification report for the proposed model and the classical SVM model on dataset DS2. The results show that the proposed model effectively classifies the normal and DDoS attacks in the test dataset DS2 samples. Therefore, our model is generalized and also report that LSTM model performed well compared to the SVM model.

**Table 7.** Classification Report of the proposed model for Dataset DS2 (0:Normal and 1:Attack).

| | Precision | Recall | F1-Score |
|---|---|---|---|
| 0 | 1.00 | 1.00 | 1.00 |
| 1 | 1.00 | 1.00 | 1.00 |
| macro avg | 1.00 | 1.00 | 1.00 |
| weighted avg | 1.00 | 1.00 | 1.00 |

**Table 8.** Classification Report of the SVM model for Dataset DS2 (0:Normal and 1:Attack).

| | Precision | Recall | F1-Score |
|---|---|---|---|
| 0 | 0.60 | 1.00 | 0.75 |
| 1 | 0.00 | 0.00 | 0.00 |
| macro avg | 0.30 | 0.50 | 0.37 |
| weighted avg | 0.36 | 0.60 | 0.45 |

### *5.5. Performance Comparison with Existing Works*

Table 9 compares our work with existing work on intrusion detection systems in SDN IoT networks. Various datasets are being used to apply the ML/DL models and perform the attack classification in SDN IoT networks. Although the comparison of our work is not fair in this case because of the different datasets used for each study, we have conducted the performance metrics-based comparison in this subsection. Our work obtained the accuracy of 97.1% for attack classification, whereas the others obtained around 96% when the SDN IoT network environment was used for training and testing the datasets. Additionally, the mean time to detect (MTTD) the network attack is determined as 0.35sec when the attack samples are tested using our approach.

**Table 9.** Performance comparison of ML/DL models in SDN IOT network environment.

| Article | Network Environment | Dataset | Technique | Accuracy |
|---|---|---|---|---|
| Wani et al. [34] | SDN and IoT | CSE-CIC-IDS2018 | IDSIoT-SDL | 96.05% |
| Li et al. [35] | SDN and IoT | KDD cup | Bat Algorithm, Random Forest | 96.03% |
| Our work | SDN and IoT | SDN-IoT, SDN-NF-TJ | LSTM | 97.1% |

### *5.6. Advantages and Limitations of the Proposed Approach*

The proposed LSTM architecture clearly distinguishes network attacks such as reconnaissance attacks such as port scanning, operating system fingerprinting, and the denial of service attacks such as DDoS and DoS in SDN IoT networks. In some scenarios, the denial of service attack traffic appears to be port scanning traffic originating from multiple sources. These classify them as port scanning attempts by existing IDS solutions. In addition, our proposed model achieved 97.7% accuracy for multiclass classification in the IoT network, which means our model discriminates the majority of the network attacks in the IoT network when tested with the SDNIoT-focused dataset DS1.

Network Dataset feature selection tends to be more challenging when the network handles diversified protocols such as IoT protocols MQTT, CoAP, AMQP, and traditional protocols TCP, UDP, and ICMP. In particular, if the network is SDN based, all the switches should pass the network traffic to the controller using OpenFlow. Therefore, in a complex IoT network, the proposed model achieves the best performance results with minimal effort to feature selection and engineering. However, the proposed DL architecture consumes time to train the model because of self-learning the feature and adapting the model weights. We believe that this time consumption for training is bearable in real-time network attack detection, as the models require understanding the baseline of the network traffic according to the organization's business requirements and demands. Our model can also be adapted per the type of attacks in any particular network environment.

In general, LSTM-based models are known to require large amounts of memory bandwidth for training the models. The high memory resource consumption can limits the use of LSTM for IDS in SDN IoT networks. However, with the rapid hardware advancement in recent times and the existing cloud service provider's pay-for-use models for utilizing computational and memory resources, the memory constraint should not be a stopping point.

The limitation of the LSTM model to use in IDS is that it requires a large amount of labeled data to train the models and a long training time to use trained models for network attack detection and classification in real time.

## 6. Conclusions and Future Work

This article proposed a deep learning-based approach to predict network attacks accurately in an SDN IoT network. An LSTM-based four-hidden layer model is used to detect the attacks and categorize the attack classes in the SDNIoT network dataset. Our comparative performance analysis showed that the DL model performed better than the ML classifier SVM and more effectively distinguished the network attack types in the dataset compared to the other DL models, DNN and CNN. Furthermore, we also showed that our proposed model is generalizable to detect SDN IoT network intrusions by validating our model on another dataset.

To the best of our knowledge, our work is the first to perform a comprehensive performance analysis of DL techniques on a dataset [41] generated in a real SDN testbed simulated environment. The testbed incorporates the IoT devices to generate the IoT traffic and traditional devices such as clients and servers for normal or attack traffic generation.

The ML and DL approaches are prone to various adversarial attacks, and the works [44–46] showed that the ML/DL-based intrusion detection systems are vulnerable to adversarial attack and evade the detection of DDoS attacks. One of our future works is

to validate the proposed model in adversarial environment conditions and determine the performance of the proposed model to combat adversarial attacks.

**Author Contributions:** Conceptualization, R.C., W.S., V.R., A.D.; methodology, R.C., W.S., V.R., A.D.; software, R.C., W.S., V.R., A.D.; validation, R.C., W.S., V.R., A.D. formal analysis, R.C., W.S., V.R., A.D.; investigation, R.C., W.S., V.R., A.D.; resources, V.R.; data curation, R.C., W.S., V.R., A.D.; writing—original draft preparation, R.C., W.S., V.R., A.D.; writing—review and editing, R.C., W.S., V.R., A.D.; visualization, R.C., W.S., V.R., A.D.; supervision, V.R.; project administration, V.R.; funding acquisition, V.R. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Maddikunta, P.K.R.; Gadekallu, T.R.; Kaluri, R.; Srivastava, G.; Parizi, R.M.; Khan, M.S. Green communication in IoT networks using a hybrid optimization algorithm. *Comput. Commun.* **2020**, *159*, 97–107. [CrossRef]
2. Lee, I.; Lee, K. The Internet of Things (IoT): Applications, investments, and challenges for enterprises. *Bus. Horizons* **2015**, *58*, 431–440. [CrossRef]
3. Farhady, H.; Lee, H.; Nakao, A. Software-defined networking: A survey. *Comput. Netw.* **2015**, *81*, 79–95. [CrossRef]
4. Boppana, R.V.; Chaganti, R.; Vedula, V. Analyzing the vulnerabilities introduced by ddos mitigation techniques for software-defined networks. In *National Cyber Summit*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 169–184.
5. Kaur, N.; Singh, A.K.; Kumar, N.; Srivastava, S. Performance impact of topology poisoning attack in SDN and its countermeasure. In Proceedings of the 10th International Conference on Security of Information and Networks, Jaipur, India, 13–15 October 2017; pp. 179–184.
6. Javed, A.R.; Ahmed, W.; Alazab, M.; Jalil, Z.; Kifayat, K.; Gadekallu, T.R. A Comprehensive Survey on Computer Forensics: State-of-the-Art, Tools, Techniques, Challenges, and Future Directions. *IEEE Access* **2022**, *10*, 11065–11089. [CrossRef]
7. Agrawal, S.; Sarkar, S.; Alazab, M.; Maddikunta, P.K.R.; Gadekallu, T.R.; Pham, Q.V. Genetic CFL: Hyperparameter optimization in clustered federated learning. *Comput. Intell. Neurosci.* **2021**, *2021*, 7156420. [CrossRef] [PubMed]
8. Agrawal, S.; Sarkar, S.; Aouedi, O.; Yenduri, G.; Piamrat, K.; Alazab, M.; Bhattacharya, S.; Maddikunta, P.K.R.; Gadekallu, T.R. Federated learning for intrusion detection system: Concepts, challenges and future directions. *Comput. Commun.* **2022**, *195*, 346–361. [CrossRef]
9. RM, S.P.; Maddikunta, P.K.R.; Parimala, M.; Koppu, S.; Gadekallu, T.R.; Chowdhary, C.L.; Alazab, M. An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture. *Comput. Commun.* **2020**, *160*, 139–149.
10. Rathore, H.; Agarwal, S.; Sahay, S.K.; Sewak, M. Malware detection using machine learning and deep learning. In Proceedings of the International Conference on Big Data Analytics, Warangal, India, 18–21 December 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 402–411.
11. Sangkatsanee, P.; Wattanapongsakorn, N.; Charnsripinyo, C. Practical real-time intrusion detection using machine learning approaches. *Comput. Commun.* **2011**, *34*, 2227–2235. [CrossRef]
12. Ravi, V.; Chaganti, R.; Alazab, M. Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system. *Comput. Electr. Eng.* **2022**, *102*, 108156. [CrossRef]
13. Ravi, V.; Chaganti, R.; Alazab, M. Deep Learning Feature Fusion Approach for an Intrusion Detection System in SDN-Based IoT Networks. *IEEE Internet Things Mag.* **2022**, *5*, 24–29. [CrossRef]
14. Khan, R.U.; Zhang, X.; Kumar, R.; Sharif, A.; Golilarz, N.A.; Alazab, M. An adaptive multi-layer botnet detection technique using machine learning classifiers. *Appl. Sci.* **2019**, *9*, 2375. [CrossRef]
15. Vinayakumar, R.; Alazab, M.; Soman, K.; Poornachandran, P.; Venkatraman, S. Robust intelligent malware detection using deep learning. *IEEE Access* **2019**, *7*, 46717–46738. [CrossRef]
16. Chaganti, R.; Ravi, V.; Pham, T.D. Image-based malware representation approach with EfficientNet convolutional neural networks for effective malware classification. *J. Inf. Secur. Appl.* **2022**, *69*, 103306. [CrossRef]
17. Chaganti, R.; Ravi, V.; Pham, T.D. Deep Learning based Cross Architecture Internet of Things malware Detection and Classification. *Comput. Secur.* **2022**, *120*, 102779. [CrossRef]
18. Ravi, V.; Alazab, M.; Srinivasan, S.; Arunachalam, A.; Soman, K. Adversarial defense: DGA-based botnets and DNS homographs detection through integrated deep learning. *IEEE Trans. Eng. Manag.* **2021**, *70*, 249–266. [CrossRef]

19. Sriram, S.; Vinayakumar, R.; Alazab, M.; Soman, K. Network flow based IoT botnet attack detection using deep learning. In Proceedings of the IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Toronto, ON, Canada, 6–9 July 2020; pp. 189–194.
20. Ravi, N.; Shalinie, S.M. Semisupervised-learning-based security to detect and mitigate intrusions in IoT network. *IEEE Internet Things J.* **2020**, *7*, 11041–11052. [CrossRef]
21. Zhao, Y.; Li, Y.; Zhang, X.; Geng, G.; Zhang, W.; Sun, Y. A survey of networking applications applying the software defined networking concept based on machine learning. *IEEE Access* **2019**, *7*, 95397–95417. [CrossRef]
22. Sultana, N.; Chilamkurti, N.; Peng, W.; Alhadad, R. Survey on SDN based network intrusion detection system using machine learning approaches. *Peer-to-Peer Netw. Appl.* **2019**, *12*, 493–501. [CrossRef]
23. Mohammed, A.R.; Mohammed, S.A.; Shirmohammadi, S. machine learning and deep learning based traffic classification and prediction in software defined networking. In Proceedings of the 2019 IEEE International Symposium on Measurements & Networking (M&N), Catania, Italy, 8–10 July 2019; pp. 1–6.
24. Dey, S.K.; Uddin, R.; Rahman, M. Performance analysis of SDN-based intrusion detection model with feature selection approach. In Proceedings of the International Joint Conference on Computational Intelligence, Budapest, Hungary, 2–4 November 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 483–494.
25. Nguyen, T.G.; Phan, T.V.; Nguyen, B.T.; So-In, C.; Baig, Z.A.; Sanguanpong, S. Search: A collaborative and intelligent nids architecture for sdn-based cloud iot networks. *IEEE Access* **2019**, *7*, 107678–107694. [CrossRef]
26. Alzahrani, A.O.; Alenazi, M.J. Designing a network intrusion detection system based on machine learning for software defined networks. *Future Internet* **2021**, *13*, 111. [CrossRef]
27. Birkinshaw, C.; Rouka, E.; Vassilakis, V.G. Implementing an intrusion detection and prevention system using software-defined networking: Defending against port-scanning and denial-of-service attacks. *J. Netw. Comput. Appl.* **2019**, *136*, 71–85. [CrossRef]
28. Sebbar, A.; Zkik, K.; Baddi, Y.; Boulmalf, M.; Kettani, M.D.E.C.E. MitM detection and defense mechanism CBNA-RF based on machine learning for large-scale SDN context. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *11*, 5875–5894. [CrossRef]
29. Tang, T.A.; Mhamdi, L.; McLernon, D.; Zaidi, S.A.R.; Ghogho, M. Deep learning approach for network intrusion detection in software defined networking. In Proceedings of the 2016 IEEE International Conference on Wireless Networks and Mobile Communications (WINCOM), Fez, Morocco, 26–29 October 2016; pp. 258–263.
30. Hannache, O.; Batouche, M.C. Neural network-based approach for detection and mitigation of DDoS attacks in SDN environments. *Int. J. Inf. Secur. Priv. (IJISP)* **2020**, *14*, 50–71. [CrossRef]
31. Hande, Y.; Muddana, A. Intrusion detection system using deep learning for software defined networks (SDN). In Proceedings of the 2019 IEEE International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 27–29 November 2019; pp. 1014–1018.
32. Tang, T.A.; Mhamdi, L.; McLernon, D.; Zaidi, S.A.R.; Ghogho, M.; El Moussa, F. DeepIDS: Deep learning approach for intrusion detection in software defined networking. *Electronics* **2020**, *9*, 1533. [CrossRef]
33. Vailshery, L.S. Global IoT and Non-IoT Connections 2010–2025 | Statista. 2021. Available online: https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide/ (accessed on 13 September 2022).
34. Wani, A.; Khaliq, R. SDN-based intrusion detection system for IoT using deep learning classifier (IDSIoT-SDL). *CAAI Trans. Intell. Technol.* **2021**, *6*, 281–290. [CrossRef]
35. Li, J.; Zhao, Z.; Li, R.; Zhang, H. Ai-based two-stage intrusion detection for software defined iot networks. *IEEE Internet Things J.* **2018**, *6*, 2093–2102. [CrossRef]
36. Tian, Q.; Han, D.; Hsieh, M.Y.; Li, K.C.; Castiglione, A. A two-stage intrusion detection approach for software-defined IoT networks. *Soft Comput.* **2021**, *25*, 10935–10951. [CrossRef]
37. Vinayakumar, R.; Alazab, M.; Srinivasan, S.; Pham, Q.V.; Padannayil, S.K.; Simran, K. A visualized botnet detection system based deep learning for the internet of things networks of smart cities. *IEEE Trans. Ind. Appl.* **2020**, *56*, 4436–4456. [CrossRef]
38. ElSayed, M.S.; Le-Khac, N.A.; Albahar, M.A.; Jurcut, A. A novel hybrid model for intrusion detection systems in SDNs based on CNN and a new regularization technique. *J. Netw. Comput. Appl.* **2021**, *191*, 103160. [CrossRef]
39. Ye, J.; Cheng, X.; Zhu, J.; Feng, L.; Song, L. A DDoS attack detection method based on SVM in software defined network. *Secur. Commun. Netw.* **2018**, *2018*, 9804061. [CrossRef]
40. Hadem, P.; Saikia, D.K.; Moulik, S. An SDN-based Intrusion Detection System using SVM with Selective Logging for IP Traceback. *Comput. Netw.* **2021**, *191*, 108015. [CrossRef]
41. Sarica, A.K.; Angin, P. A Novel SDN Dataset for Intrusion Detection in IoT Networks. In Proceedings of the 2020 16th IEEE International Conference on Network and Service Management (CNSM), Izmir, Turkey, 2–6 November 2020; pp. 1–5.
42. Jafarian, T. SDN-NF-TJ | IEEE DataPort. 2019. Available online: https://ieee-dataport.org/documents/sdn-nf-tj (accessed on 13 September 2022).
43. Othman, S.M.; Ba-Alwi, F.M.; Alsohybe, N.T.; Al-Hashida, A.Y. Intrusion detection model using machine learning algorithm on Big Data environment. *J. Big Data* **2018**, *5*, 34. [CrossRef]
44. Aiken, J.; Scott-Hayward, S. Investigating adversarial attacks against network intrusion detection systems in sdns. In Proceedings of the 2019 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Dallas, TX, USA, 12–14 November 2019; pp. 1–7.

45. Abusnaina, A.; Khormali, A.; Nyang, D.; Yuksel, M.; Mohaisen, A. Examining the robustness of learning-based ddos detection in software defined networks. In Proceedings of the 2019 IEEE Conference on Dependable and Secure Computing (DSC), Hangzhou, China, 18–20 November 2019; pp. 1–8.
46. Qiu, H.; Dong, T.; Zhang, T.; Lu, J.; Memmi, G.; Qiu, M. Adversarial attacks against network intrusion detection in iot systems. *IEEE Internet Things J.* **2020**, *8*, 10327–10335. [CrossRef]