Date of publication JUN-30, 2024, date of current version APR-23, 2024.
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.23.2.3538

Deep Learning-Based Echo State Neural Network for Cyber Threat Detection in IoT-Driven IICS Networks

S. SINGARAVELAN[1], P. VELAYUTHA PERUMAL[1], R. ARUN[1], V. SELVAKUMAR[1], D. MURUGAN[2]
[1]Department of Computer Science and Engineering, P.S.R Engineering College, Sivakasi, 62614, India.
[2]Computer Science and Engineering, Manonmaniam Sundaranar University, Tirunelveli, India.

Corresponding author: S. Singaravelan (e-mail: singaravelan.msu@gmail.com).

ABSTRACT The advent of Software-Defined Networking (SDN) has ushered in a new era in network architecture, providing unprecedented levels of flexibility and adaptability. However, this advanced flexibility exposes SDN to security risks, particularly Distributed Denial of Service (DDoS) attacks. Detecting and mitigating DDoS attacks in SDN environments poses a critical challenge. This study introduces an innovative DDoS detection approach leveraging Echo State Networks (ESN) tailored specifically for SDN. This approach is based on two core assumptions: firstly, routine network operations primarily exhibit normal behavior, and secondly, there are discernible differences in data characteristics between normal and abnormal network conditions. These assumptions hold true in the realm of everyday network dynamics. To validate the efficacy of the ESN algorithm, we augment this approach by incorporating flow features to enhance DDoS detection capabilities. This study underscores the effectiveness of ESN in identifying and mitigating Distributed Denial of Service (DDoS) attacks, DDoS threats, achieving an impressive average success rate of 97.78%. By harnessing the potential of Echo State Networks, this work makes a substantial contribution to ongoing efforts in fortifying network security, providing a proactive defense against disruptive DDoS attacks.

KEYWORDS Software-Defined Networking; Distributed Denial of Service; EtherCAT; Netlink; SoftRouter; Inter Planetary File System; Mobile Edge Computing; NSLKDD dataset.

I. INTRODUCTION

SOFTWARE-Defined Networking (SDN) technology revolutionizes network management by enabling dynamic, programmable configurations that enhance performance and monitoring, resembling cloud computing more than traditional approaches [1]. It addresses the limitations of static, decentralized architectures in traditional networks, providing greater flexibility and simplified troubleshooting. By dividing packet forwarding (data plane) and routing (control plane), SDN accomplishes this by centralizing network intelligence in a single component. Core intelligence of the SDN network is centered at the control plane, which is managed by one or more controllers [2]. However, while this centralization offers benefits, it also introduces challenges in security, scalability, and elasticity, constituting a primary concern in SDN.

The separation of control and data planes, which was first used in the public switched telephone network for easier provisioning and maintenance, is where the foundation of SDN principles lies. This separation had been adopted before it was used in data networks.

Following years of research, the Internet Engineering Task Force (IETF) published the "Forwarding and Control Element Separation" (ForCES) interface standard in 2004. This technology allowed for the separation of control and forwarding tasks [3]. Additionally, a supplementary SoftRouter Architecture was suggested by the ForCES Working Group. Similar goals were sought by early IETF standards such Linux Netlink and Path Computation Element (PCE)-Based Architecture, but they encountered difficulties because of growing rivalry brought on by standardized APIs between the control and data planes and worries about possible control plane failures.

Traffic patterns in enterprise data centers have changed dramatically. Modern applications feature a lot of "east-west" machine-to-machine traffic, in contrast to traditional client-server applications, where communication mostly happens between one client and one server [4]. This is followed by the typical "north-south" traffic flow, which returns data to the end-user device. Users are simultaneously changing the patterns of network traffic by connecting from different locations and time, using different devices, and seeking access to business applications and content. Furthermore, a lot of administrators in enterprise data centers are thinking about implementing a utility computing model, which can involve both public and private clouds and boost traffic on wide area networks.

Leveraging the controller's centralized view of the network and its capacity to reprogram the data plane as needed, the SDN architecture can enable, facilitate, or improve network-related security applications. Although there is still much to learn about the security of the SDN architecture, the sections that follow concentrate on security applications that SDN makes possible or revisits.

Several SDN research projects have explored security applications based on the SDN controller, pursuing different goals. Specific cases of this use include mitigating and detecting Distributed Denial of Service (DDoS) attacks as well as controlling the spread of worms and botnets. These programs basically collect standardized network statistics from the forwarding plane on a regular basis and use classification techniques to identify abnormalities. The application notifies the controller to reconfigure the data plane in order to lessen the hazard when it has been detected.

SDN controllers are used by different classes of security applications to develop moving target defense (MTD) algorithms. By periodically changing important system or network attributes, these algorithms seek to increase the difficulty of any assault on a particular system or network. MTD algorithms are difficult to deploy in traditional networks because there is no central authority that can decide which attributes to change for each component of the system [5]. This task becomes easier to handle in an SDN network because of the centralized control. For example, one application might map virtual IP addresses to hosts on a regular basis, and the controller would manage the mapping from virtual to actual IP addresses [6]. A different program might imitate fictitious open, closed, or filtered ports on distinct hosts in order to significantly increase noise when an attacker is conducting reconnaissance (such as scanning).

Software Defined Networking's Effects on Security in Industrial Control Systems. With sensor networks being the only exception now, industrial Ethernet is expected to become the dominant technology in distributed control systems and take over the whole communication network from the office to the field level. Ethernet's performance has been questioned since it was introduced in time-sensitive industrial applications, primarily due to the outdated coax networks. Automation networks are constructed using switches, have a lot of capacity, and are designed for more demanding applications. Current networks are constructed with full duplex solutions. These solutions attempt to incorporate enhancements to the Ethernet standards, such as resource reservation efforts like those of the IEEE 802.1 Time-Sensitive Networking Task Group, or they provide inherent Quality of Service (QoS), such as EtherCAT. Numerous problems that control system engineers are dealing with are not brand-new [7]. QoS and resilience have been issues since packet switching networks first appeared.

An intentional attempt to obstruct regular activity on a server, service, or network by flooding the target or its surrounding infrastructure with excessive amounts of Internet traffic is known as a distributed denial-of-service (DDoS) assault. The efficacy of DDoS attacks arises from their ability to use numerous compromised computer systems as sources of attack traffic. Computers and other networked resources, such as Internet of Things devices, might be considered exploited machines. A DDoS attack can be compared, at a high level, to an unforeseen traffic congestion that blocks the highway and keeps ordinary traffic from reaching its destination. DDoS attacks use networks of computers connected to the Internet [8]. These networks are made up of computers and other devices (such as Internet of Things devices) that have been infected with malware, enabling an attacker to remotely manipulate them. These standalone devices are known as bots (sometimes called zombies), and a collection of bots is known as a botnet. Different components of a network connection are the target of different kinds of DDoS assaults [9]. It is important to comprehend how a network connection is established in order to comprehend how various DDoS attacks operate.

II. RELATED WORK

Herrera et al. addressed the fog node placement problem, employing both optimal and approximated methods, and compared their results with state-of-the-art benchmarks.

Z. Li et al. introduced a behavior-based verification method called Crowd-Learning for software-defined vehicular networks within a Mobile Edge Computing (MEC) framework. This approach incentivizes MEC infrastructures to provide accurate data for behavior estimation without prior knowledge of the dynamic environment.

Orozco-Santos et al. investigated Mobile Multicast Forwarding with Software Defined Network (MMF-SDN), a solution utilizing Software Defined Networking for Wireless Sensor Network (SDN WISE) protocol, which leverages SDN and Time Slotted Channel Hopping (TSCH) synchronism. This approach manages mobile nodes as multicast sources through resource allocation from the controller.

Sangodoyin et al. carried out a case study in which they developed classification models for precisely detecting and categorizing DDoS flooding attacks based on parameters like jitter, throughput, and reaction time using experimental data from a sample SDN architecture.

In order to guarantee Service Level Agreement (SLA) compliance, Okwuibe et al. introduced SDRM, an SDN-enabled Resource Management scheme that dynamically allocates resources for Industrial Internet of Things (IIoT) network models. With the use of the Satisfiability (SAT) problem, resource allocation is modeled as a Constraint Satisfaction Problem (CSP) in this method.

In a survey, Amin et al. divided machine learning (ML) approaches into three categories: reinforcement learning, unsupervised learning, and supervised learning. These techniques are used for routing optimization in SDN. The survey offers advice on selecting the best machine learning technique depending on goals and available resources. It does this by providing a thorough overview and comparison of pertinent papers.

The Federated Forest Software-Defined Networking (SDN)-enabled IDS (BFF-IDS) with Blockchain technology was created by Aliyu et al. to handle sensitive data sharing in Controller Area Network (CAN) connections. They used blockchain for safe model exchanges and dynamic packet routing, and they used InterPlanetary File System (IPFS) for model hosting.

The Open Broadband trial in Brazil was examined by Montalvo et al., who highlighted the move away from traditional black box passive optical network (PON) solutions and toward open software and whitebox hardware in fiber access networks.

Nam et al. optimized stream reservation through centralized procedures by introducing a simpler Stream Reservation Protocol (SRP),which can be used over SDN in in-vehicle bridging networks.

Building on CloudSimSDN-NFV, Nyanteh et al. created the CloudSimHypervisor framework and described its characteristics, design, and applications.

Knowledge graphs (KGs) are incorporated into the alert analysis process for network irregularities in the knowledge-guided fault localization method proposed by Z. Li et al.

A centralized route optimization and service assurance technique called ROSA was introduced by Njah et al. It is designed for multi-layer programmable industrial architecture and supports a variety of flows, such as bandwidth-sensitive services and ultra-reliable low-latency communications (URLLC).

An SD-IoT framework was presented by Njah et al. to offer security services to IoT networks. In order to successfully detect DDoS attacks, they created a dynamic and programmable Counter-based DDoS Attack Detection (C-DAD) application and thoroughly tested it with a range of network parameters.

Masdari et al. studied DDoS attack types with new attacks on virtual machines and hypervisors in the cloud computing environment. The authors also include popular network defensive strategies and cloud computing defenses against DDoS attacks.

Akbar, et al. proposed a novel scheme based on Hellinger distance (HD) to detect low-rate and multi-attribute DDoS attacks. Leveraging the SIP load balancer for detecting and mitigating DDoS attacks is proposed. Usually DDoS detection and mitigations schemes are implemented in SIP proxy, however leveraging the SIP load balancer to fight against DDoS by using existing load balancing features is done with the proposed scheme implemented by modifying leading open source Kamailio SIP proxy server. The scheme is evaluated by experimental test setup and found results are outperforming the existing prevention schemes in use against DDoS for system overhead, detection rate and false-positive alarm

III. PROPOSED WORK

Network management has been completely transformed by Software-Defined Networking (SDN), which offers flexible and dynamic control. However, SDN is vulnerable to possible Distributed Denial of Service (DDoS) assaults because of its open nature.

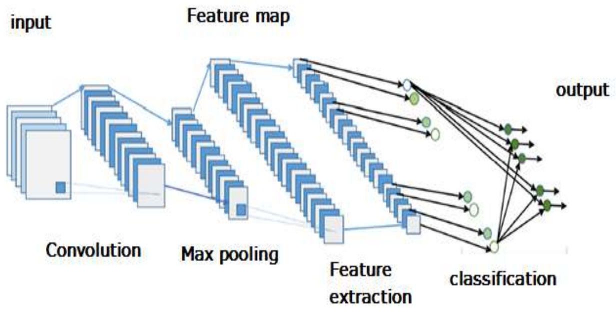


Figure 1. Overall architecture of the Convolutional Neural Network (CNN)

It is critical to identify and counteract DDoS assaults in SDN setups [10]. This paper presents a novel DDoS detection system that makes use of Echo State Networks (ESN) customized for SDN. For efficient DDoS detection, the emphasis is on feature extraction, preprocessing, robust model training, and data gathering.

A. MODULES

1. Data Set Collection:

Kaggle Data Set:

For training and evaluating the DDoS detection model, network traffic data is collected from publicly available datasets on Kaggle [11]. This encompasses both normal network traffic and DDoS attack scenarios, ensuring diverse training data.

Data Preprocessing:

The collected dataset may undergo preprocessing steps, including handling missing values, data normalization, and transformation, to create a structured dataset suitable for training and testing.

2. Feature Extraction:

Traffic Features:

This module extracts pertinent features from network traffic data, such as packet sizes, traffic patterns, and protocol usage [12]. These features serve as inputs to the ESN model for DDoS detection.

Statistical Analysis:

Statistical analyses are performed to identify patterns and anomalies indicative of DDoS attacks [13]. Feature selection techniques are applied to choose the most discriminative features.

3. Echo State Network (ESN) Model:

ESN Architecture: The ESN architecture is designed to process network traffic data, comprising an input layer, hidden reservoir layer, and output layer [14]. ESN, known for its capabilities in sequential data analysis, is employed.

Model Training:

Using the preprocessed dataset and extracted features, the ESN model is trained to differentiate normal from abnormal network behavior. This supervised learning process aims to build a robust DDoS detection system.

4. DDoS Detection:

Real-time Analysis: The trained ESN model is deployed in real-time to analyze incoming network traffic [15]. It evaluates traffic patterns and identifies deviations indicating potential DDoS attacks.

Anomaly Detection: The system focuses on detecting anomalies in network behavior, including unusual traffic spikes, protocol violations, and other DDoS attack signatures.

5. Evaluation and Testing:

Performance Metrics: Various performance metrics such as precision, recall, F1-score, and ROC curves are employed to assess the effectiveness of the DDoS detection system [16]. These metrics gauge the system accuracy and efficiency.

Validation: The system is rigorously tested against a diverse set of network traffic scenarios, including known DDoS attacks, to validate its performance in real-world conditions.

6. User Interface (Optional):

Dashboard: An optional user interface may be developed to provide real-time insights into network traffic and detected anomalies, aiding network administrators in taking prompt actions.

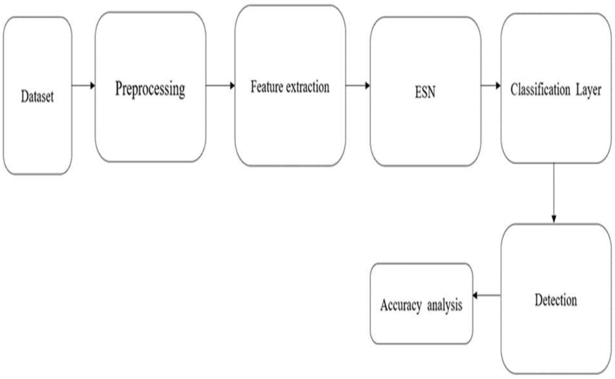


Figure 2. Proposed System

B. Echo State Network (ESN)

One particular kind of recurrent neural network (RNN) that is well-known for its ability to interpret sequential input is the echo state network (ESN) [17]. An overview of the key layers that make up the ESN and its architecture is given in this section.

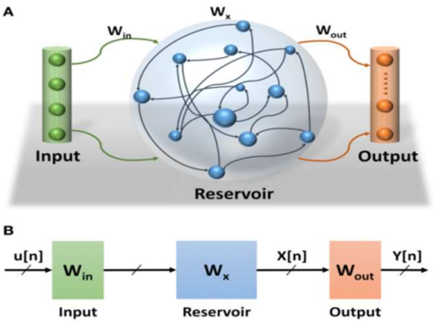


Figure 3. ESN Architecture

ESN Architecture and Layers:

1. Input Layer:

- The Input Layer receives external input data and transforms it into a format suitable for processing within the network. It is connected directly to the reservoir layer.

2. Reservoir Layer:

- The Reservoir Layer, different from ESN, comprises a multitude of recurrent neurons forming a dynamic, fixed-weight network. The random and unaltered nature of its internal connections allows it to capture complex temporal dependencies in data.

3. Output Layer:

- The Output Layer receives the reservoir state as input and generates the network output. It is a trainable layer with adjustable weights. The learning process primarily occurs in this layer.

Echo State Networks excel in capturing extensive temporal dependencies and nonlinear relationships in data [18]. Their straightforward yet powerful architecture makes them invaluable for various applications, including time series analysis and sequential data processing [19]. The success of ESN lies in its ability to leverage the intrinsic dynamics of the reservoir layer, enabling efficient learning and generalization.

The dataset is divided into training and testing sets, a standard practice in machine learning [20]. This allows for training the model on one portion of the data and evaluating its performance on another.

Before training the Echo State Network (ESN), the data is standardized using the StandardScaler from the scikit-learn library [21]. Standardization ensures that the data has a mean of zero and a standard deviation of one, which prepares it for the effective use in the model.

The ESN, being a type of recurrent neural network, is initialized with specific parameters such as the number of reservoir neurons, spectral radius, and a random seed [22]. These parameters influence the ESN dynamics and its ability to capture temporal patterns.

The ESN model undergoes training using the training data, where it learns to map the input features (X_train) to the corresponding output labels (y_train) [23]. The ESN unique characteristic is its proficiency in capturing complex temporal dependencies in the data.

Post-training, the ESN is employed to make predictions on the testing data (X_test). These predictions are then compared with the actual labels to evaluate the model's accuracy [24].

The code computes and displays the accuracy of the ESN model, representing its ability to distinguish between normal and attacker instances [25]. Additionally, a confusion matrix is presented, illustrating true positive, true negative, false positive, and false negative predictions, providing a comprehensive assessment of the model's performance.

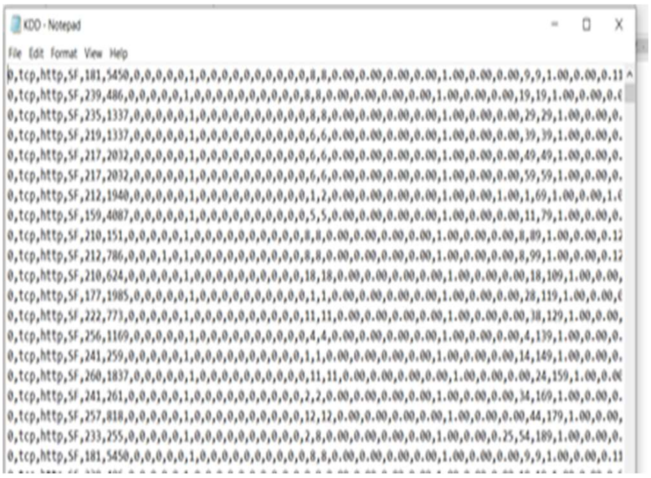
IV. RESULTS ANALYSIS

A. DATASET AND PREPROCESSING

The proposed model is evaluated on two widely used public datasets: the 10% KDD99 and the full NSLKDD, both commonly utilized for assessing intrusion detection schemes. In the KDD99 dataset, all legitimate traffic samples are utilized along with 14 types of prevalent attacks. For the NSLKDD dataset, all legitimate traffic samples are used alongside 16 types of prevalent attacks. Each data sample's features in the datasets form a feature vector. The dataset is randomly split into training, validation, and testing subsets with a ratio of 0.7:0.1:0.2.

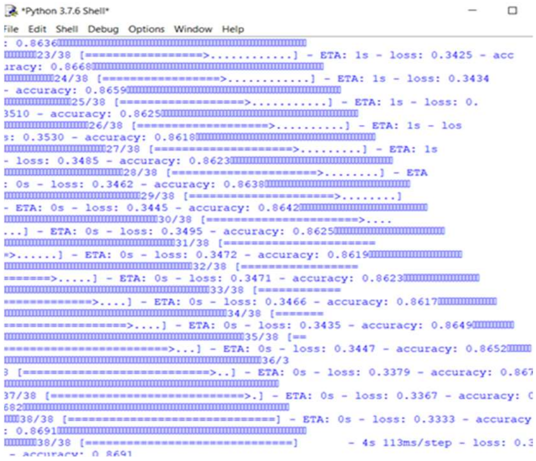
Model Training:

For each predictive model, training is conducted in mini-batches with 1,024 sequences per epoch over 100 iterations. To enhance the models' generalization performance, the data is independently divided, and each model is trained and tested 10 times. The mean evaluation metrics from these 10 testing results are reported.



```
0,tcp,http,SF,181,5450,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,0.00,9,9,1.00,0.00,0.11
0,tcp,http,SF,239,486,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,0.00,19,19,1.00,0.00,0.0
0,tcp,http,SF,235,1337,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,0.00,29,29,1.00,0.00,0.
0,tcp,http,SF,219,1337,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,6,6,0.00,0.00,0.00,0.00,1.00,0.00,0.00,39,39,1.00,0.00,0.
0,tcp,http,SF,217,2032,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,6,6,0.00,0.00,0.00,0.00,1.00,0.00,0.00,49,49,1.00,0.00,0.
0,tcp,http,SF,217,2032,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,6,6,0.00,0.00,0.00,0.00,1.00,0.00,0.00,59,59,1.00,0.00,0.
0,tcp,http,SF,212,1940,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,2,0.00,0.00,0.00,0.00,1.00,0.00,1.00,1,69,1.00,0.00,1.0
0,tcp,http,SF,159,4087,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,5,0.00,0.00,0.00,0.00,1.00,0.00,0.00,11,79,1.00,0.00,0.
0,tcp,http,SF,210,151,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,0.00,8,89,1.00,0.00,0.12
0,tcp,http,SF,212,786,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,0.00,8,99,1.00,0.00,0.12
0,tcp,http,SF,210,624,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,18,18,0.00,0.00,0.00,0.00,1.00,0.00,0.00,18,109,1.00,0.00,
0,tcp,http,SF,177,1985,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,28,119,1.00,0.00,0
0,tcp,http,SF,222,773,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,11,11,0.00,0.00,0.00,0.00,1.00,0.00,0.00,38,129,1.00,0.00,
0,tcp,http,SF,256,1169,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,1.00,0.00,0.00,4,139,1.00,0.00,0.
0,tcp,http,SF,241,259,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,14,149,1.00,0.00,0.
0,tcp,http,SF,260,1837,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,11,11,0.00,0.00,0.00,0.00,1.00,0.00,0.00,24,159,1.00,0.00
0,tcp,http,SF,241,261,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,34,169,1.00,0.00,0.
0,tcp,http,SF,257,818,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,12,12,0.00,0.00,0.00,0.00,1.00,0.00,0.00,44,179,1.00,0.00,
0,tcp,http,SF,233,255,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,8,0.00,0.00,0.00,0.00,1.00,0.00,0.25,54,189,1.00,0.00,0.
0,tcp,http,SF,181,5450,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,0.00,9,9,1.00,0.00,0.11
```

Figure 4. Data set description



```
: 0.8636
23/38 [=================>............] - ETA: 1s - loss: 0.3425 - acc
uracy: 0.8668
24/38 [=================>............] - ETA: 1s - loss: 0.3434
- accuracy: 0.8659
25/38 [==================>...........] - ETA: 1s - loss: 0.
3510 - accuracy: 0.8625
26/38 [===================>..........] - ETA: 1s - los
s: 0.3530 - accuracy: 0.8618
27/38 [====================>.........] - ETA: 1s
- loss: 0.3485 - accuracy: 0.8623
28/38 [=====================>........] - ETA
: 0s - loss: 0.3462 - accuracy: 0.8638
29/38 [======================>........]
- ETA: 0s - loss: 0.3445 - accuracy: 0.8642
30/38 [=======================>....
...] - ETA: 0s - loss: 0.3495 - accuracy: 0.8625
31/38 [========================
=>......] - ETA: 0s - loss: 0.3472 - accuracy: 0.8619
32/38 [==================
=======>.....] - ETA: 0s - loss: 0.3471 - accuracy: 0.8623
33/38 [=============
=============>....] - ETA: 0s - loss: 0.3466 - accuracy: 0.8617
34/38 [========
==================>....] - ETA: 0s - loss: 0.3435 - accuracy: 0.8649
35/38 [==
===========================>...] - ETA: 0s - loss: 0.3447 - accuracy: 0.8652
36/3
8 [============================>..] - ETA: 0s - loss: 0.3379 - accuracy: 0.867

37/38 [=============================>.] - ETA: 0s - loss: 0.3367 - accuracy: 0
682
38/38 [==============================] - ETA: 0s - loss: 0.3333 - accuracy
: 0.8691
38/38 [==============================]       - 4s 113ms/step - loss: 0.3
- accuracy: 0.8691
```

Figure 5. LSTM Training

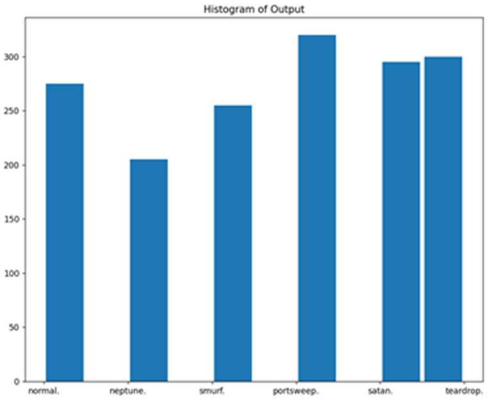
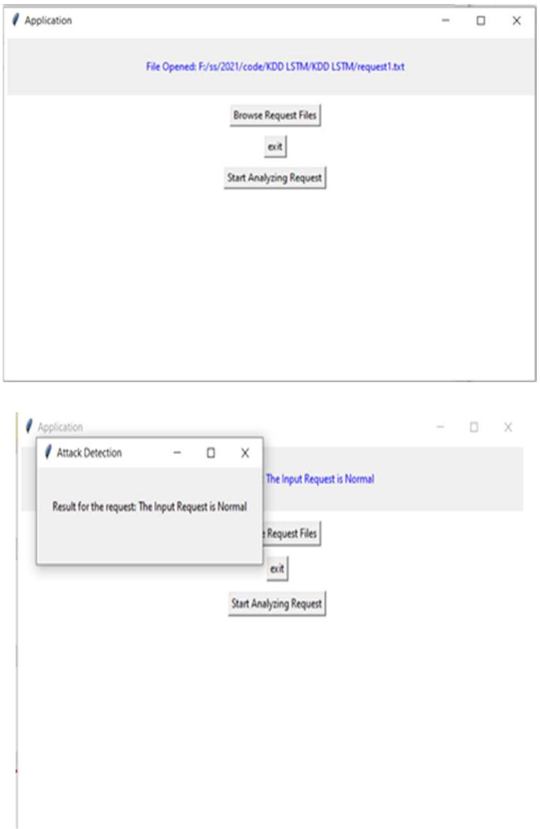


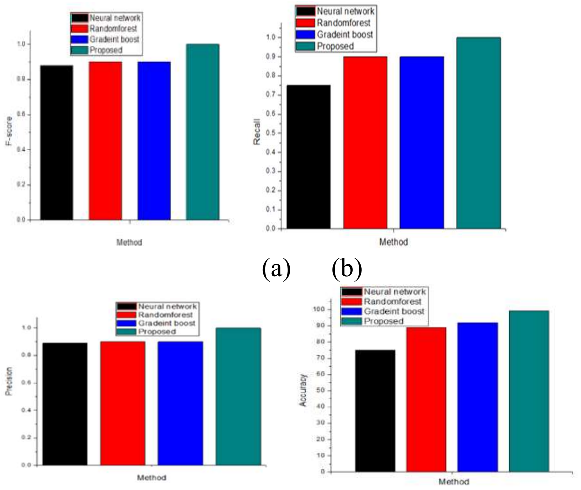
Figure 6. Histogram of attacker variation



File Opened: F:/ss/2021/code/KDD LSTM/KDD LSTM/request1.txt

Result for the request: The Input Request is Normal

Figure 7. Attacker classification



(a) (b)

(c) (d)

Figure 8. Different methods Performance analysis of a – d

IV. CONCLUSION

The proposed DDoS detection scheme, employing Echo State Networks (ESN) within the context of Software-Defined Networking (SDN), represents a substantial advancement in fortifying the security and resilience of modern network infrastructures. Built on the foundational hypotheses that routine network operations are predominantly normal and anomalies exhibit distinct data characteristics, this study underscores the effectiveness of ESN in identifying and mitigating Distributed Denial of Service (DDoS) attacks.

Through rigorous evaluation in a simulated environment, the research solidifies the scheme proficiency in detecting and responding to DDoS threats, achieving an impressive average success rate of 97.78%. This accomplishment holds great promise for the cybersecurity landscape, especially in the realm of SDN, where flexibility and adaptability are paramount.

By harnessing the potential of Echo State Networks, this work makes a substantial contribution to ongoing efforts in fortifying network security, providing a proactive defense against disruptive DDoS attacks. The findings emphasize the viability of ESN as a valuable tool in the arsenal of cybersecurity measures, further enhancing our ability to safeguard critical network infrastructures in an ever-evolving digital landscape.

References

[1] Y. Njah and M. Cheriet, “Parallel route optimization and service assurance in energy-efficient software-defined industrial IoT networks,” *IEEE Access*, vol. 9, pp. 24682-24696, 2021, https://doi.org/10.1109/ACCESS.2021.3056931.

[2] Z. Li et al., “Fault localization based on knowledge graph in software-defined optical networks,” *Journal of Lightwave Technology*, vol. 39, no. 13, pp. 4236-4246, https://doi.org/10.1109/JLT.2021.3071868.

[3] A. O. Nyanteh, M. Li, M. F. Abbod and H. Al-Raweshidy, “CloudSimHypervisor: Modeling and simulating network slicing in software-defined cloud networks,” *IEEE Access*, vol. 9, pp. 72484-72498, 2021, https://doi.org/10.1109/ACCESS.2021.3079501.

[4] S. Nam, H. Kim and S.-G. Min, “Simplified stream reservation protocol over software-defined networks for in-vehicle time-sensitive networking,” *IEEE Access*, vol. 9, pp. 84700-84711, 2021, https://doi.org/10.1109/ACCESS.2021.3088288.

[5] J. Montalvo, J. Torrijos, D. Cortes, R. Chundury and M. St. Peter, “Journey toward software-defined passive optical networks with multi-PON technology: an industry view [Invited],” *Journal of Optical Communications and Networking*, vol. 13, no. 8, pp. D22-D31, 2021, https://doi.org/10.1364/JOCN.423034.

[6] I. Aliyu, M. C. Feliciano, S. Van Engelenburg, D. O. Kim and C. G. Lim, “A blockchain-based federated forest for SDN-enabled in-vehicle

network intrusion detection system," *IEEE Access*, vol. 9, pp. 102593-102608, 2021, https://doi.org/10.1109/ACCESS.2021.3094365.

[7] R. Amin, E. Rojas, A. Aqdus, S. Ramzan, D. Casillas-Perez and J. M. Arco, "A survey on machine learning techniques for routing optimization in SDN," *IEEE Access*, vol. 9, pp. 104582-104611, 2021, https://doi.org/10.1109/ACCESS.2021.3099092.

[8] J. Okwuibe et al., "SDN-enabled resource orchestration for industrial iot in collaborative edge-cloud networks," *IEEE Access*, vol. 9, pp. 115839-115854, 2021, https://doi.org/10.1109/ACCESS.2021.3105944.

[9] A. O. Sangodoyin, M. O. Akinsolu, P. Pillai and V. Grout, "Detection and classification of DDoS flooding attacks on software-defined networks: A case study for the application of machine learning," *IEEE Access*, vol. 9, pp. 122495-122508, 2021, https://doi.org/10.1109/ACCESS.2021.3109490.

[10] F. Orozco-Santos, V. Sempere-Payá, J. Silvestre-Blanes and T. Albero-Albero, "Multicast scheduling in SDN WISE to support mobile nodes in industrial wireless sensor networks," *IEEE Access*, vol. 9, pp. 141651-141666, 2021, https://doi.org/10.1109/ACCESS.2021.3120917.

[11] Z. Li, X. Yang, C. Wang, K. Ma and C. Jiang, "Crowd-learning: A behavior-based verification method in software-defined vehicular networks with MEC framework," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1622-1639, 2022, https://doi.org/10.1109/JIOT.2021.3107581.

[12] J. L. Herrera, J. Galán-Jiménez, L. Foschini, P. Bellavista, J. Berrocal and J. M. Murillo, "QoS-aware fog node placement for intensive IoT applications in SDN-fog scenarios," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 13725-13739, 2022, https://doi.org/10.1109/JIOT.2022.3143948.

[13] A. B. Haque, B. Bhushan, & G. Dhiman, "Conceptualizing smart city applications: Requirements, architecture, security issues, and emerging trends," *Expert Systems*, vol. 39, issue 5, e12753, 2022. https://doi.org/10.1111/exsy.12753.

[14] J. Mirkovic, & P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Computer Communication Review*, vol. 34, issue 2, pp. 39-53, 2004. https://doi.org/10.1145/997150.997156.

[15] C. Douligeris, & A. Mitrokotsa, "DDoS attacks and defense mechanisms: Classification and state-of-the-art," *Computer Networks*, vol. 44, issue 5, pp. 643-666, 2004. https://doi.org/10.1016/j.comnet.2003.10.003.

[16] M. T. Manavi, "Defense mechanisms against distributed denial of service attacks: A survey," *Computers & Electrical Engineering*, vol. 72, pp. 26-38, 2018. https://doi.org/10.1016/j.compeleceng.2018.09.001.

[17] X. Zhang, Y. Zhang, R. Altaf, X. Feng, "A multi-agent system-based method of detecting DDoS attacks," *I. J. Computer Network and Information Security*, vol. 2, pp. 52-64, 2018. https://doi.org/10.5815/ijcnis.2018.02.07.

[18] N. A. Ignatev, & E. R. Navruzov, "Estimates of the complexity of detecting types of DDOS attacks," *International Journal of Computing*, vol. 21, issue 4, pp. 443-449, 2022. https://doi.org/10.47839/ijc.21.4.2779.

[19] J. Nazario, "DDoS attack evolution," *Network Security*, vol. 2008, issue 7, pp. 7-10, 2008. https://doi.org/10.1016/S1353-4858(08)70086-2.

[20] Q. Yan, F. R. Yu, Q. Gong, & J. Li, "Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, issue 1, pp. 602-622, 2015. https://doi.org/10.1109/COMST.2015.2487361.

[21] G. A. Jaafar, S. M. Abdullah, & S. Ismail, "Review of recent detection methods for HTTP DDoS attack," *Journal of Computer Networks and Communications*, vol. 2019, article ID 1283472, pp. 1-10, 2019. https://doi.org/10.1155/2019/1283472.

[22] K. Sonar, & H. Upadhyay, "A survey: DDOS attack on Internet of Things," *International Journal of Engineering Research and Development*, vol. 10, issue 11, pp. 58-63, 2024.

[23] J. Mirkovic, M. Robinson, & P. Reiher, "Alliance formation for DDoS defense," *Proceedings of the 2003 Workshop on New Security Paradigms*, 2003, pp. 11-18. https://doi.org/10.1145/986655.986658.

[24] Y. Li, Q. Liu, "A comprehensive review study of cyber-attacks and cyber security: Emerging trends and recent developments," *Energy Reports*, vol. 7, pp. 8176-8186, 2021. https://doi.org/10.1016/j.egyr.2021.08.126.

[25] S. Yu, J. Zhang, J. Liu, X. Zhang, Y. Li, & T. Xu, "A cooperative DDoS attack detection scheme based on entropy and ensemble learning in SDN," EURASIP Journal on Wireless Communications and Networking, vol. 2021, article ID 90, 2021. https://doi.org/10.1186/s13638-021-01957-9.

S. SINGARAVELAN, a Professor in the department of Computer Science and Engineering, P.S.R Engineering College, Sivakasi, Tamilnadu. He published more than 25+ research papers in indexed journal publications. He also acts as a technical reviewer in journals and conferences. Research interest in the area of Image processing, Data mining and soft Computing.

P. VELAYUTHA PERUMAL, Post Graduate Scholar in the department of Computer Science and Engineering, P.S.R Engineering College, Sivakasi, Tamilnadu. Research interest in the area of Image processing, Networks Security.

R. ARUN, an Associate Professor in the department of Computer Science and Engineering, P.S.R Engineering College, Sivakasi, Tamilnadu. He published more than 10+ research papers in indexed journal publications. Research interest in the area of Image processing, Networks.

V. SELVAKUMAR, an Associate Professor in the department of Computer Science and Engineering, P.S.R Engineering College, Sivakasi, Tamilnadu. He published more than 5+ research papers in indexed journal publications. Research interest in the area of network security , wireless networks.

D MURUGAN, a Professor in the department of Computer Science and Engineering, Manonmaniam Sundaranar University, Tirunelveli, Tamilnadu. He published more than 50+ research papers in indexed journal publications. He also acts as a technical reviewer in journals and conferences. Research interest in the area of Image processing, Data mining and Computer Networks.