

Automatika

Journal for Control, Measurement, Electronics, Computing and Communications

64
60089-7001097 02/2023

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/taut20

An adaptive multistage intrusion detection and prevention system in software defined networking environment

N Maheswaran, S Bose & Buvaneshwari Natarajan

To cite this article: N Maheswaran, S Bose & Buvaneshwari Natarajan (2024) An adaptive multistage intrusion detection and prevention system in software defined networking environment, *Automatika*, 65:4, 1364-1378, DOI: [10.1080/00051144.2024.2372749](https://doi.org/10.1080/00051144.2024.2372749)

To link to this article: <https://doi.org/10.1080/00051144.2024.2372749>



© 2024 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.



Published online: 11 Jul 2024.



Submit your article to this journal [↗](#)



Article views: 807



View related articles [↗](#)



View Crossmark data [↗](#)



Citing articles: 1 View citing articles [↗](#)



An adaptive multistage intrusion detection and prevention system in software defined networking environment

N Maheswaran^a, S Bose^a and Buvaneswari Natarajan^b

^aDepartment of Computer Science and Engineering, College of Engineering Guindy, Anna University, Chennai, India; ^bMiddlesex College, Edison, NJ, USA

ABSTRACT

The advancements made in Software-Defined Networking (SDN) technology seem quite promising, with potential wide application in managing and controlling the latest network infrastructures. SDN technology decouples the control plane from the data plane, enabling effective and flexible network management. However, this dynamic phenomenon brings new security challenges. With the increasing dynamism and programmable nature of networks, conventional security protocols may not sufficient to protect against advanced and sophisticated attacks. Although Intrusion Detection Systems (IDSs) have been extensively applied for identifying and preventing security threats in traditional network environments, IDS models designed specifically for traditional network requirements may not be adequate for SDN environments. These issues may stem from the static nature of conventional networks, contrasting with the dynamicity of advanced SDN networks, and the traditional IDS's inability to adapt to the dynamic nature of SDN. To address these challenges, the current research proposes a novel Deep Hybrid IDS model to enhance network security in SDN environments and prevent attacks using Scapy. The proposed model detects signature-based attacks by integrating Gated Recurrent Units (GRU) and Long Short-Term Memory (LSTM) for real-time simulated datasets, achieving an accuracy of 97.8%, which is comparatively better than existing models.

ARTICLE HISTORY

Received 19 December 2023
Accepted 21 June 2024

KEYWORDS

Software-defined networking; deep one-class Intrusion Detection System; open network operating system; Canadian institute for Cyber security Flow meter; Scapy



1. Introduction

SDN was proposed in 2006 at Stanford University and is a centralized model in which the complete set of requests raised by clients is received. This dynamic architecture helps achieve flexibility, scalability, and affordability compared to Physical Machines and Virtual Machines (VMs) [1]. The SDN architecture has the ability to decouple the data and control plane, thus enabling the administrator to directly program the network control and forwarding functions [2]. This feature enables the widespread application of the SDN architecture. The primary function of SDN is to transfer data from the control plane to forwarding devices, into a separate data plane [3]. In this structure, the control plane determines the path, while the data planes execute the task of data forwarding. With the rapidly growing penetration of Internet of Things-based devices, projections for the year 2050 estimate the number of devices connected to the internet to reach 100 billion [4].

The SDN architecture comprises application layer, control layer, and data layers based on their respective functions. Although OpenFlow is the extensively used protocol in SDN, there are other protocols as well, such as NETCONF. SDN controllers are located in the

control plane, providing logic to the data plane. The controller consists of a northbound interface, controller core, and southbound interface [5]. Multiple studies [6–9] have discussed the application of software-defined IoT through SDN solutions, which may provide users with multiple benefits such as global information access, efficient resource utilization, security, privacy, energy management, and network function virtualization. Despite the various benefits associated with deploying SDN in IoT communication processes, challenges are encountered, such as the demand for a huge number of entire networks in IoT, distributed controllers [10], differentiation of forwarding plane configurations, and the evolution of commuter flow charts, etc. [3].

In SDN, the application layer functions as an interface between the administrator and the network. This integrated structure enables the IDS to monitor and perform network traffic analysis in real-time to detect network anomalies [10,11]. Furthermore, in SDN architecture, implementing security policies across the network is simplified. For example, it is possible to identify malicious attacks by programming the SDN controller [12] using Convolutional Neural Network (CNN) [13] and LSTM [14].

CONTACT N Maheswaran  nmaheswaran97@gmail.com  Department of Computer Science and Engineering, College of Engineering Guindy, Anna University, Chennai 600025, TN, India

© 2024 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.

This is an Open Access article distributed under the terms of the Creative Commons Attribution-NonCommercial License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited. The terms on which this article has been published allow the posting of the Accepted Manuscript in a repository by the author(s) or with their consent.

As mentioned in the literature cited above, a wide range of studies has been conducted combining two or more approaches to develop robust models. Ensemble learning is one of the Machine Learning (ML) techniques that combine different models to achieve high prediction accuracy. An ensemble deep learning model, which combines Deep Neural Networks (DNNs) and the ensemble learning approach, can be leveraged to enhance overall prediction accuracy. Traditional IDSs have been applied in a limited number of studies in SDN environments. Since signature-based IDSs require a static configuration, they can be used in traditional networks. However, in a dynamic and programmable environment like SDN, such IDSs cannot be applied. This scenario renders conventional IDSs incapable and inadequate for application in modern network architectures. The growing adoption of SDN poses challenges in network management, including security, privacy, and distribution of controllers. In this context, the research work attempts to develop an adaptive Intrusion Detection and Prevention System (IDPS) capable of meeting the demands of the dynamic SDN environment without compromising security and privacy outcomes.

As network size and complexity grow, SDNs may struggle with processing vast amounts of data in real-time, leading to latency and reduced performance. The hybrid model leverages the strengths of GRU and LSTM networks to efficiently handle large datasets and time-series data, improving real-time processing capabilities. Traditional SDN-based IDS may face difficulties in adapting to new and evolving cyber threats due to static rule sets and limited adaptability. The combined use of GRU and LSTM allows the model to learn complex patterns and dynamically adapt to threats by continuously updating its learning parameters.

Additionally, it prevents attacks by using Scapy as an Intrusion Prevention System (IPS). The proposed model has the ability to adapt to the dynamic nature of SDN environments and detect network attacks. This research contributes to the domain by developing a highly advanced and effective network security solution for application in SDN environments. This, in turn, can improve the overall security posture of modern network infrastructures. The outcomes of the proposed model were compared with those of other models to establish its superiority.

2. Objectives

Based on the identified research gap, the research aims to develop an adaptive IDPS for application in SDN environments. The current research work is conducted with the following objectives:

- (1) To propose and develop an adaptive IDPS for detecting and alerting the network administrator regarding malicious or suspicious traffic.

- (2) To monitor the network's performance after deploying the IDPS under different measures and compare and contrast it with existing state-of-the-art models.

3. Literature survey

In the literature [15], a comprehensive survey was conducted earlier focusing on IDS for SDN networks using ML, Deep Learning (DL), Reinforcement Learning (RL), and hybrid and ensemble-based techniques. The review encompassed various types of models within all the aforementioned techniques. Additionally, the authors listed some of the prominent datasets that act as benchmarks in IDS-based research works. However, the study cited the lack of an SDN-dedicated dataset to train the models. Bhardwaj et al. [16] mentioned that despite the favourable environment for different security levels in IoT, it is not possible for SDN to resolve all types of security issues on its own. The issue with IDS is that most are based on ML techniques, which tend to generate a high false positive rate and are unable to provide results in the case of non-linear data.

A LSTM-based approach has been validated for the detection of network attacks with the help of SDN-backed IDS in IoT networks [17]. The proposed model was found to efficiently identify and classify attacks. In an earlier study [18], the authors proposed and validated a novel hybrid DL approach based on CNN for the detection and classification of network traffic as either normal or attack. The proposed SD-Reg method was found to outperform existing methods, and the study proposed a lightweight Network IDS (NIDS) by training the CNN-based models to achieve network security without compromising the model's performance.

The authors [19] proposed a novel HFS-LGBM (Hybrid Feature Selection-Light Gradient Boosting Machine)-based IDS to be applied in SDN architecture. In this approach, correlation-based feature selection and random forest recursive feature elimination algorithms were combined to reap the advantages of both algorithms. The proposed method obtained excellent results in terms of precision, accuracy, recall, and F-measure. To detect intrusions in every sub-network, Luo [20] proposed a decision tree optimized by the Black Hole Optimization (BHO) algorithm. The proposed method was validated using the NSL-KDD and NSW-NB15 databases, and the accuracy achieved with both datasets was excellent compared to existing methods.

A NIDS-DL approach was proposed in the literature [21] for application in SDN architecture. This hybrid method combined the concepts involved in NIDS and DL methods and deployed 12 features (including 5 deep learning classifiers) from the NSL-KDD dataset. The proposed method achieved high accuracy values

in binary classification as well as attack detection. In an earlier study [22], the authors combined the pattern recognition ability of ML techniques and network programmability features to safeguard the network from DoS attacks and port scanning attacks. According to the study outcomes, the Naive Bayes ML model achieved the highest accuracy for both types of attacks considered in the study.

From the literature survey, it is evident that various approaches have been proposed for IDS in an SDN environment. Although methods such as ML, DL, and hybrid techniques appear promising, they still face significant challenges. ML-based IDS often encounter issues with false positive rates and have difficulty managing non-linear data.

4. Proposed system

Figure 1 provides an overview of the IDPS, while Figure 2 illustrates the proposed IDPS architecture, which is supported by deep hybrid learning. The network traffic is captured by the proposed model using a suitable capturing module. Subsequently, the information is preprocessed and converted into training and testing datasets. The architecture comprises GRU and LSTM models that include an input layer, a convolution layer, an LSTM layer, and an output layer. The proposed model is trained using an appropriate optimizer, loss function, and evaluation metric. It is then evaluated using the testing data. Finally, the output values are fed into the Multi-Layer Perceptron classifier and combined to create the deep hybrid model. Here is the revised version of the provided text with improved grammar:

Finally, the SDN is deployed along with the data plane and control plane. In this procedure, it is important to monitor the data plane for network traffic, while

it is crucial to feed the data to the collector located at the control plane. The network traffic data is analysed using the deep hybrid model with the purpose of detecting intrusions and evaluating the proficiency of the model using appropriate metrics.

4.1. Collector

4.1.1. Network traffic capturing

Traffic is generated from kali Linux machine. It is captured using Wireshark after which the information is to be stored as a .pcap file. With the help of CICFlowmeter, the features relevant to the requirement are extracted from .pcap file saved as a .csv file. Figure 3 shows the logical setup of network traffic capturing module required to collect the network traffic data in SDN architecture. In the proposed work used the ONOS controller. Further, the proposed work used a total of four Virtual Machines (VMs) in this study such as the ONOS controller VM, OVS switch + mininet VM, Kali Linux VM (attacker machine), and metasploitable-2 server (vulnerable Linux machine). Out of these VMs, OVS switch is used for the purpose of hardware virtualization.

4.1.2. VMnet setup

The ONOS controller is installed and configured while in parallel, the OVS switch is installed along with Mininet software in the same virtual machine. Then, four adapters such as ens38, ens39, ens40, and ens41 are created in the OVS VM. The same OpenFlow switch has two more OVS bridges named br1 and br2. Every data plane interface is assigned with a corresponding appropriate bridge. The IP addresses are removed from each data plane interface and are assigned to the bridge created earlier. "Kali Linux VM" is connected with the same adapter alike br1 whereas Metasploitable2 server

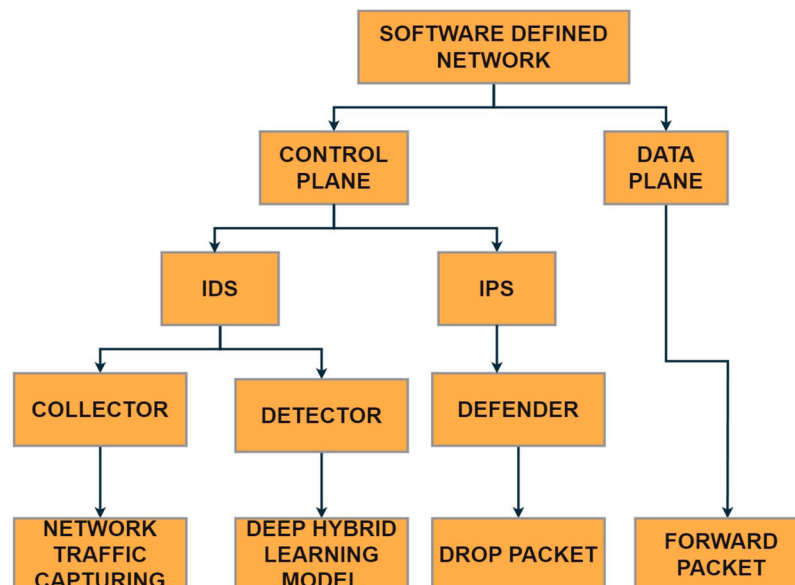


Figure 1. Overview of the proposed IDPS system.

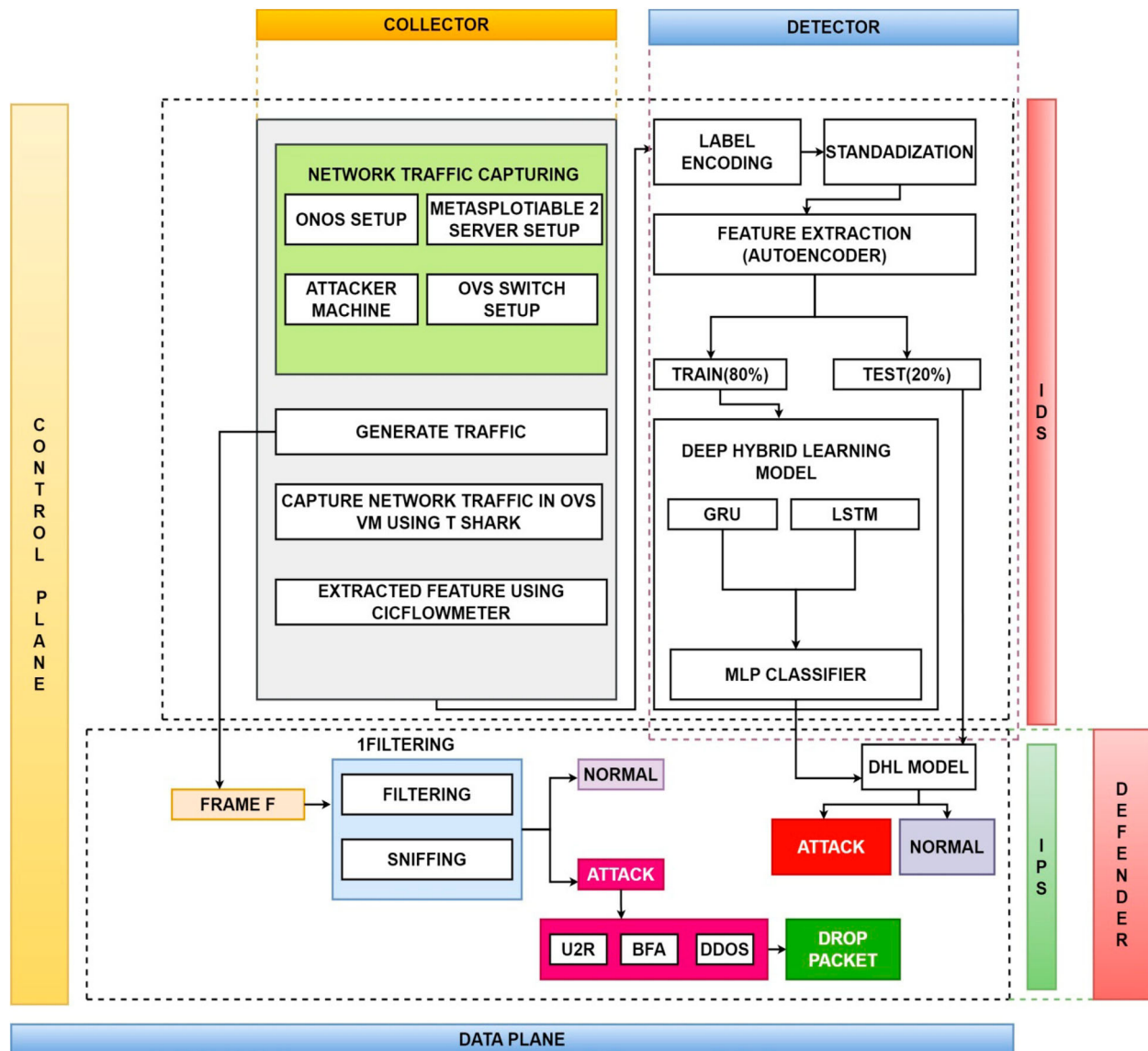


Figure 2. Proposed system architecture.

is integrated with the same adapter alike br2. Here, the IP forwarding option is enabled for the OVS Linux machine. With a total of four virtual hosts from h1 to h4 a Mininet topology is manufactured. All the four hosts are integrated with S1 bridge. Here, the IP address of the S1 bridge is supplemented in the form of a typical gateway for every virtual host that is present in the Mininet topology. The ONOS controller is connected with the bridges produced earlier such as br1, br2 and S1 after which the setup is completed.

4.1.3. Generate traffic

In this stage, the network device is informed with a network status request message. Then, the collector raises a request message to the network device with regards to the information about the current status of the network. Afterwards, the request for network status is processed. When the network device receives a request message, it processes the request and starts tshark.

4.1.4. Capture network traffic data using tshark

The collector uses tshark to capture the network traffic data after which the collected data is saved as a .pcap file.

4.1.5. Extract the required features using CICFlowmeter

The CICFlowmeter extracts the features from the captured network traffic data (.pcap file). The features are inclusive of information like the type of protocol, IP addresses of both source and the destination, and the packet length. Then, the conversion of .pcap file to .csv file occurs with the help of CICFlowmeter after which the result is transmitted to the detector.

Figure 4 shows the real-time data in which 1 represents normal, 2 corresponds to U2R, 3 represents the BFA and 4 represents the DDOS attack.

4.2. Detector

In detector phase, label encoding is conducted to provide the real-time dataset as an input to generate the

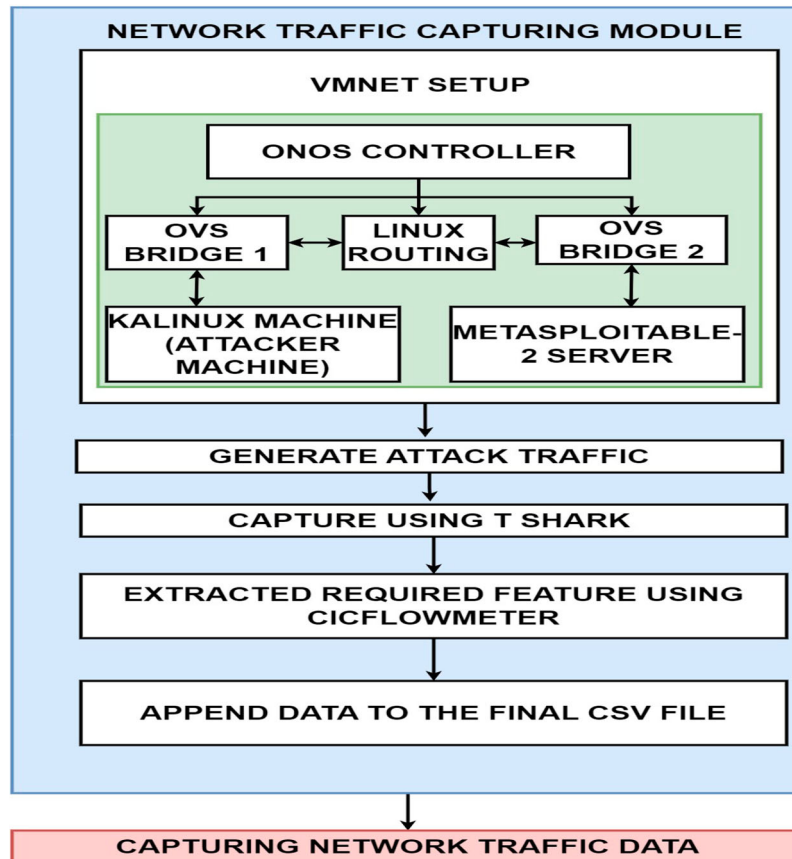


Figure 3. Logical setup of network traffic capturing module.

output as data frame with encoded labels. Further, the data is split into number arrays with standard data points in order to split the test data and train the model accordingly. Figure 5 shows the logical design used behind the development of deep hybrid learning model using GRU and LSTM models.

4.2.1. Label encoding

The real-time dataset serves as the input and it initiates the creation of a dictionary that uniquely assigns an integer to each distinct label. This is accomplished through enumeration of a set of unique labels that are present in the dataset. Subsequently, every label in the dataset is encoded with the help of its respective integer value. Then, these encoded labels are integrated into a data frame, thus facilitating their utilization in subsequent analyses or ML tasks. In this manner, the compatibility of the dataset is enhanced using various computational algorithms and methodologies.

4.2.2. Standardization

Standardization is a crucial pre-processing step in ML technique as it scales up the numerical features to a common scale, typically with 0 mean and unit variance. The process of standardization is applied methodically, when the dataset includes both numerical features as well as encoded categorical labels. At first, the categorical labels are encoded into numerical values with the help of techniques like label encoding. Thus, the model

is ensured not to have any bias towards a particular feature due to varying nature of the scales. Finally, the standardized numerical features and the encoded categorical labels are combined into NumPy array, thus creating a unified dataset, which is ready for use in ML algorithms. This standardized array provides a consistent and normalized representation of the original data, thus improving the performance of the model and its interpretability. In SDN environments, IDS addresses data imbalance using the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE is employed to alleviate skewed class distribution by generating synthetic instances of the minority class. Specifically, we applied SMOTE to augment the minority class samples, thereby achieving a more balanced dataset. This approach enhances the classifier's ability to learn from the minority class instances, leading to improved detection performance for rare network intrusions. By leveraging SMOTE, our IDS model in SDN environments demonstrates increased effectiveness in identifying and mitigating security threats.

4.2.3. feature extraction

When creating ML-based models, it is important to segregate the data into training and testing datasets. This is crucial for evaluating the generalization performance of the model on unseen data. During the incorporation of an autoencoder for feature extraction, the data is segregated according to standard procedures before being

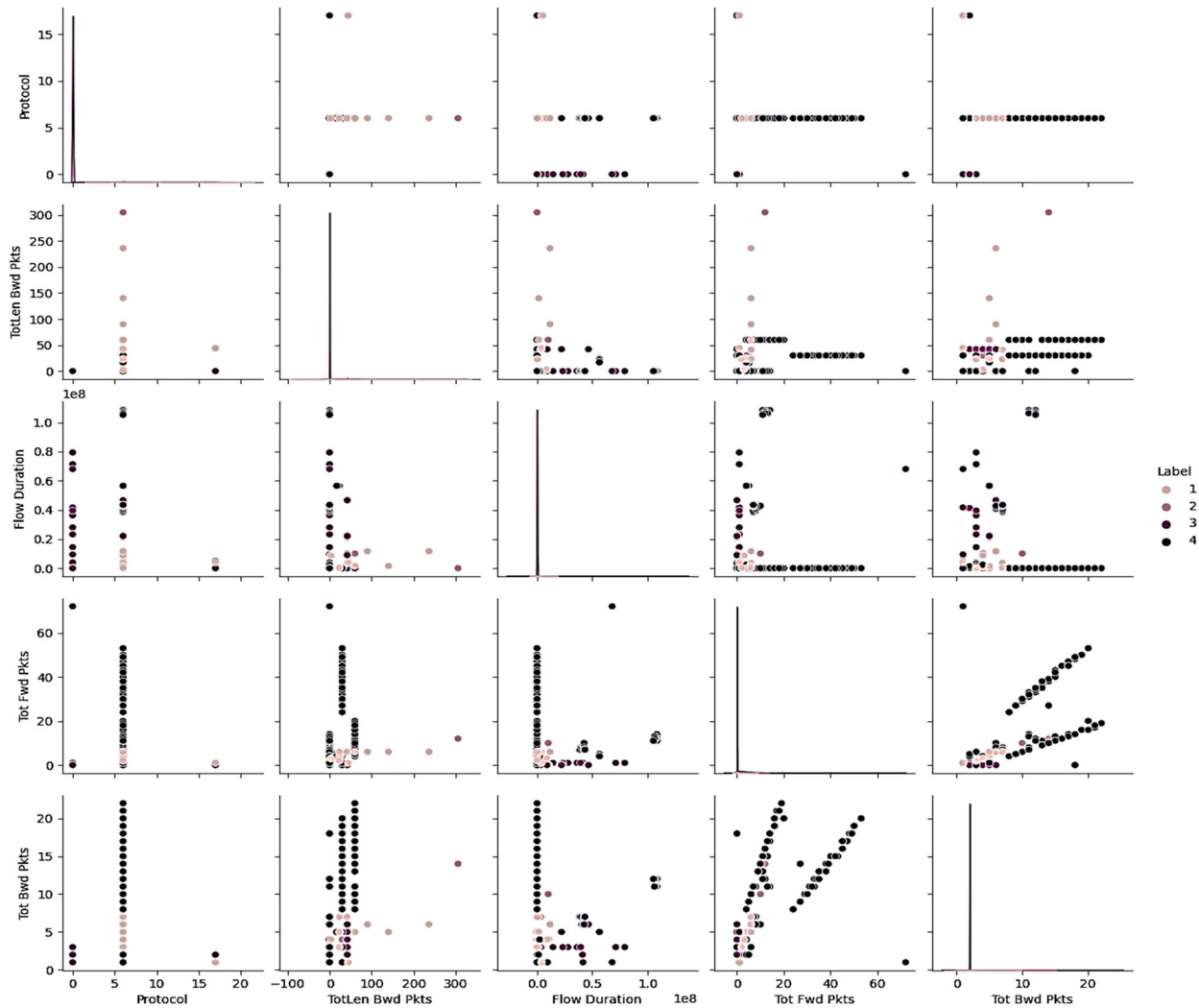


Figure 4. Extracted real time data visualization.

fed into the GRU and LSTM networks. After training the autoencoder on the entire dataset, the data is transformed by the encoder into a lower-dimensional representation. This part serves as the input for the subsequent GRU or LSTM models. Prior to feature extraction, the data is split from the original dataset, commonly into training and testing datasets. This phenomenon ensures that the model is trained on one subset of the data whereas it is evaluated on an independent subset, thus enabling the analytical outcomes remain unbiased. The resulting split dataset after feature extraction using auto encoder supports both the training and testing of GRU and LSTM models. This outcome contributes to the development of accurate and robust sequence-based predictive models. Table 1 shows the Hyper Parameter tuning in GRU, LSTM and Hybrid GRU-LSTM model for deep hybrid learning model.

4.2.4. Deep hybrid learning model

4.2.4.1. Ensemble GRU and LSTM. The model demonstrates a comprehensive approach in capturing both short-term and long-term dependencies within the sequential information. The encoded data from the

autoencoder's training split is simultaneously fed into both the GRU and LSTM layers. Generally, the GRU layer excels in capturing short-term dependencies through its update and reset gates. On the other hand, the LSTM layer specializes in maintaining long-term memory with the help of its intricate gating mechanism. The ensemble model combines the strengths of both architectures, allowing the layers to complement each other's capabilities. This collaborative processing ensures that the model effectively captures and represents intricate patterns within the sequential data by leveraging the strengths of both the GRU and LSTM components. Furthermore, this collaborative approach enhances the overall performance of the autoencoder-based system during the training phase.

Figures 6–8 show the accuracy, loss, and confusion matrices of the GRU, LSTM, and Ensemble models across different counts of epochs. The accuracy of the GRU, LSTM, and Ensemble models is depicted over time or epochs. In general, accuracy represents the number of correctly classified instances and provides an overview of the model's performance. Analysing these figures allows us to observe trends and fluctuations in the accuracy of each model throughout the

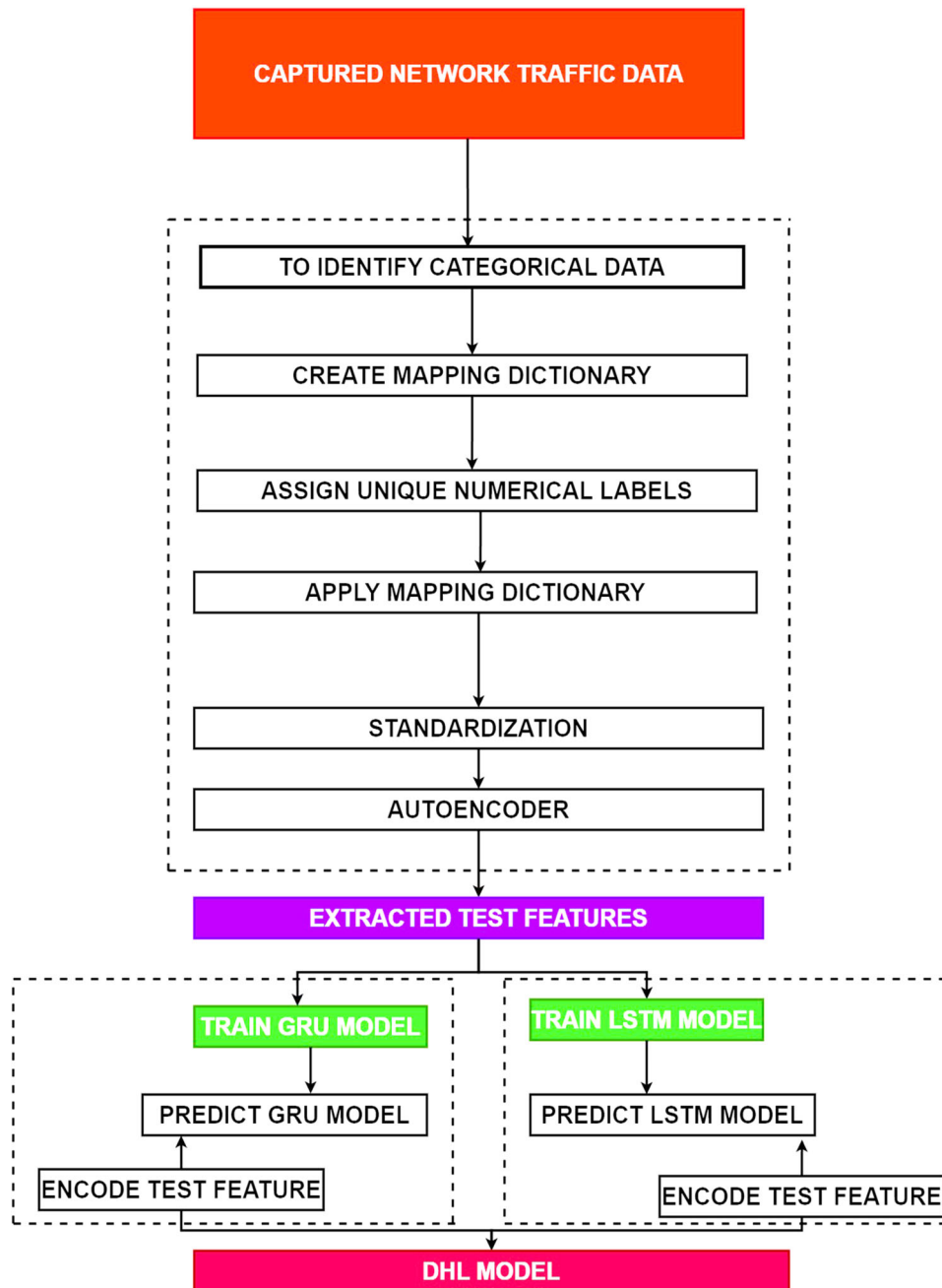


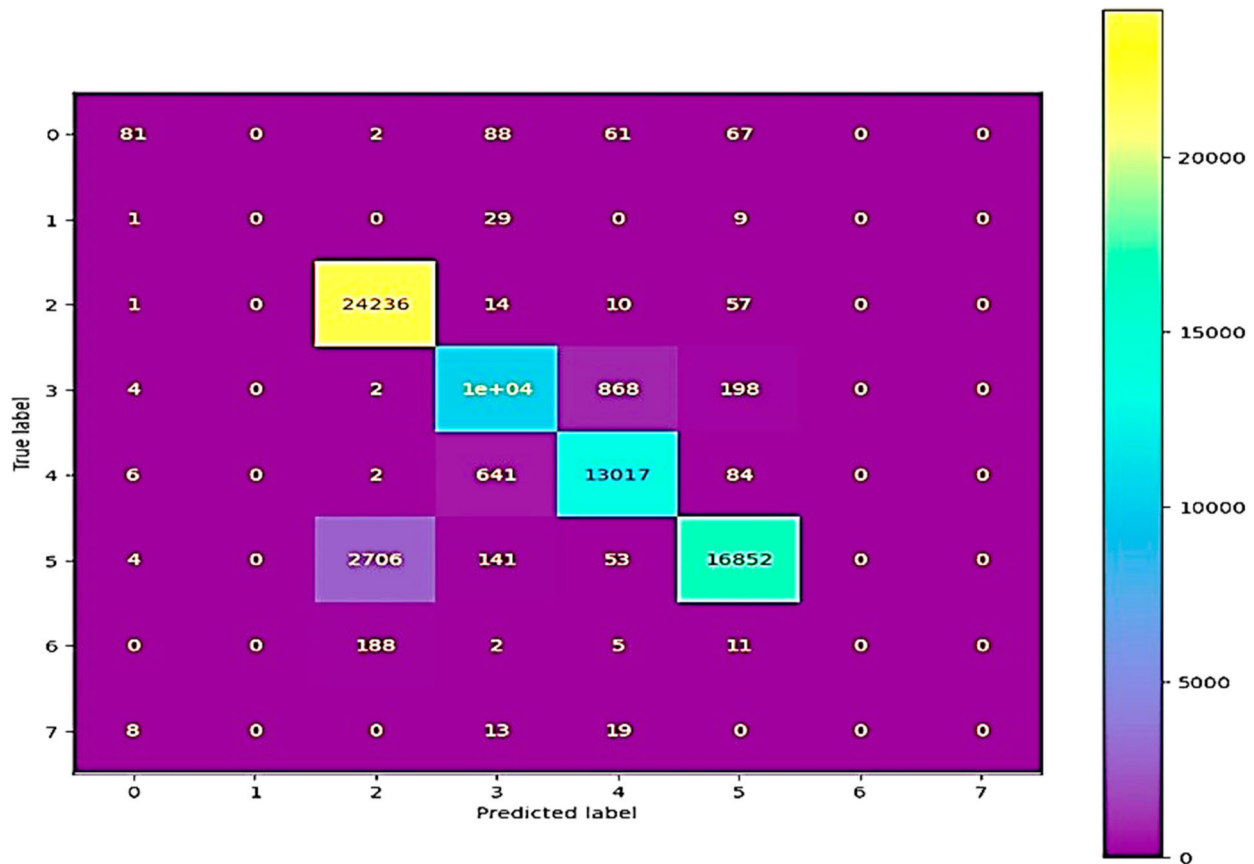
Figure 5. Logical setup of the deep hybrid learning model.

Table 1. Hyper Parameter tuning in GRU, LSTM and Hybrid GRU-LSTM model for deep hybrid learning model.

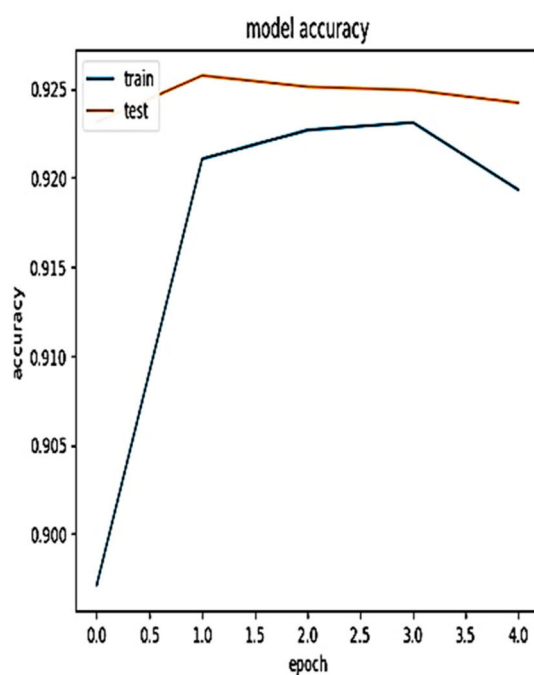
Model	Hyper parameter	Values Tested	Accuracy (%)
GRU	Hidden Units	[64, 128, 256]	[96.5, 97.1, 96.8]
	Dropout	[0.0, 0.1, 0.2]	[97.0, 96.7, 96.9]
	Learning Rate	[0.001, 0.01, 0.1]	[96.8, 97.2, 96.5]
	Activation	["relu", "tanh"]	[96.9, 97.1]
LSTM	Hidden Units	[64, 128, 256]	[97.2, 96.5, 97.0]
	Dropout	[0.0, 0.1, 0.2]	[97.1, 96.8, 96.9]
	Learning Rate	[0.001, 0.01, 0.1]	[96.5, 97.0, 96.8]
	Activation	["relu", "tanh"]	[96.8, 97.2]
Hybrid(GRU + LSTM)	GRU Hidden Units	[64, 128, 256]	[97.0, 96.9, 97.1]
	LSTM Hidden Units	[64, 128, 256]	[97.2, 96.8, 97.0]
	Dropout	[0.0, 0.1, 0.2]	[96.9, 97.1, 96.5]
	Learning Rate	[0.001, 0.01, 0.1]	[97.1, 96.5, 97.8]
	Activation	["relu", "tanh"]	[96.8, 97.8]

training process. It is essential to identify the model that exhibits high accuracy, indicating better predictive capabilities.

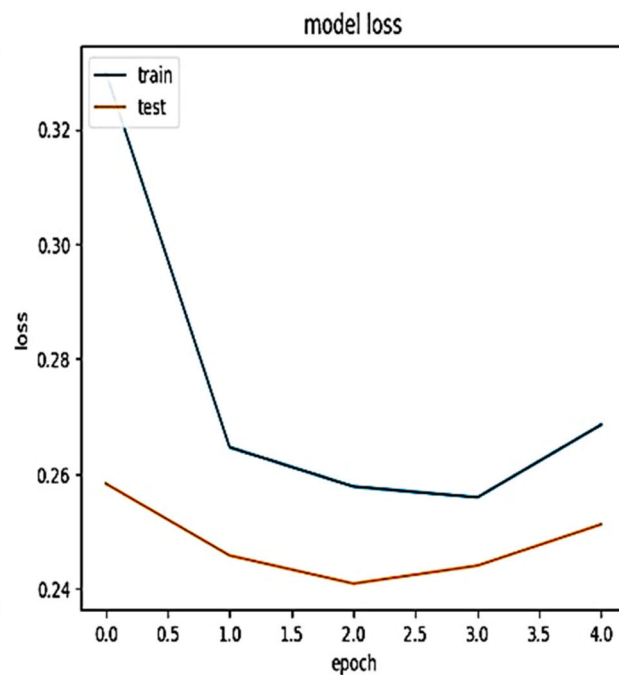
Loss quantifies the dissimilarity between the predicted values and the actual values. In general, lower loss values signify better performance of the model.



6 (a) Confusion Matrix



6 (b) Model Accuracy

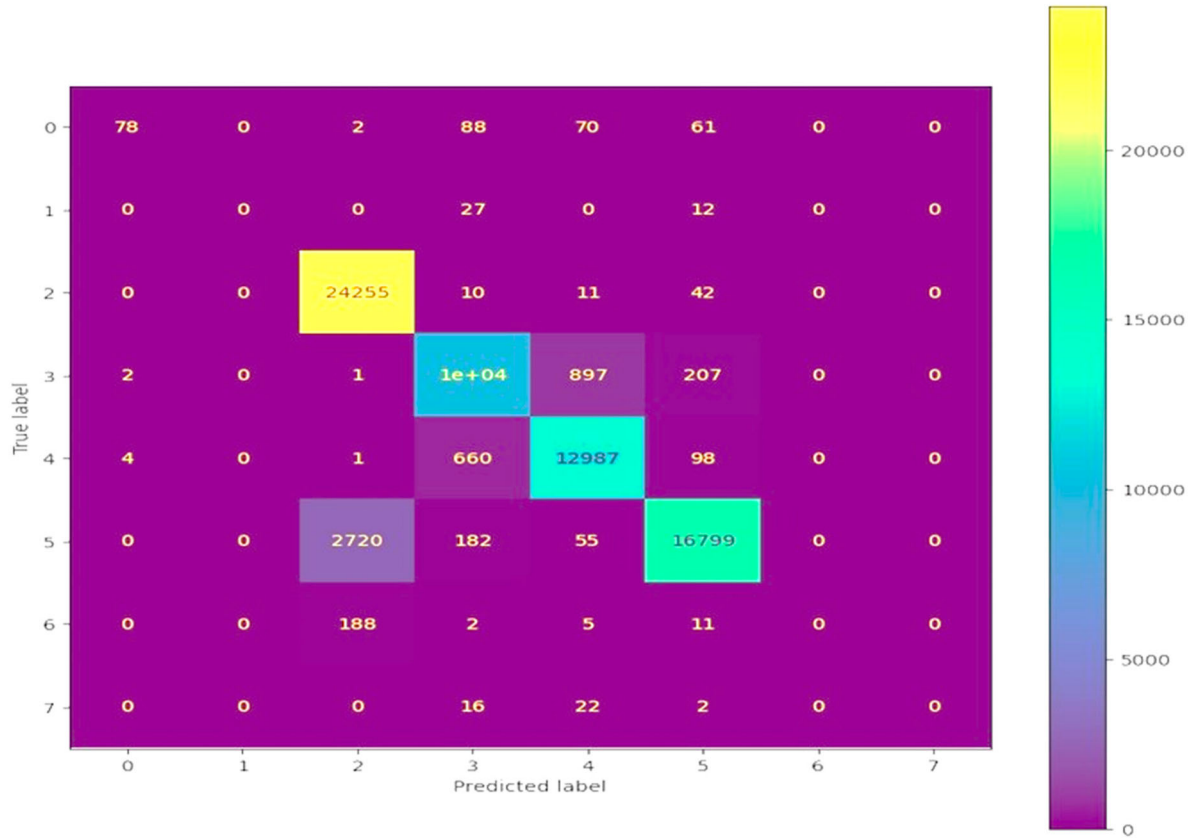


6 (c) Model Loss

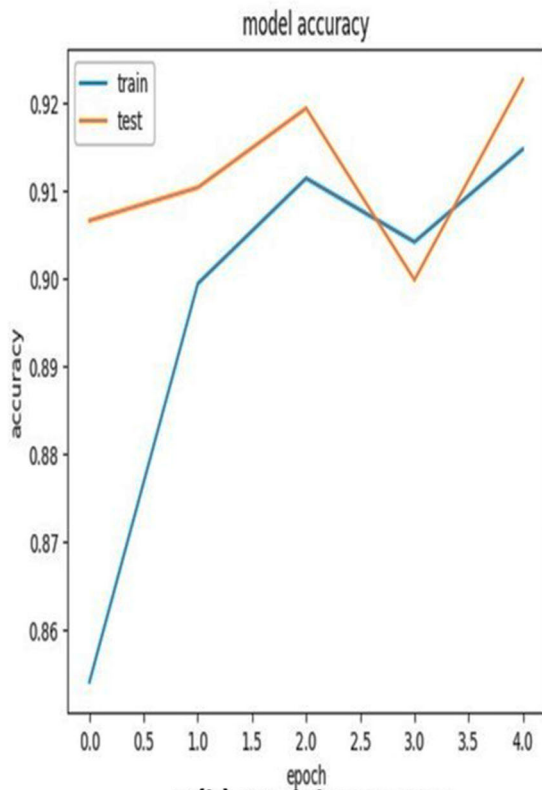
Figure 6. (a) Confusion Matrix, (b) Accuracy, (c) Loss for LSTM.

Figures 6–8 help in identifying the convergence patterns and relative stability of each model. Both lower and highly consistent loss values generally indicate that the model performs better.

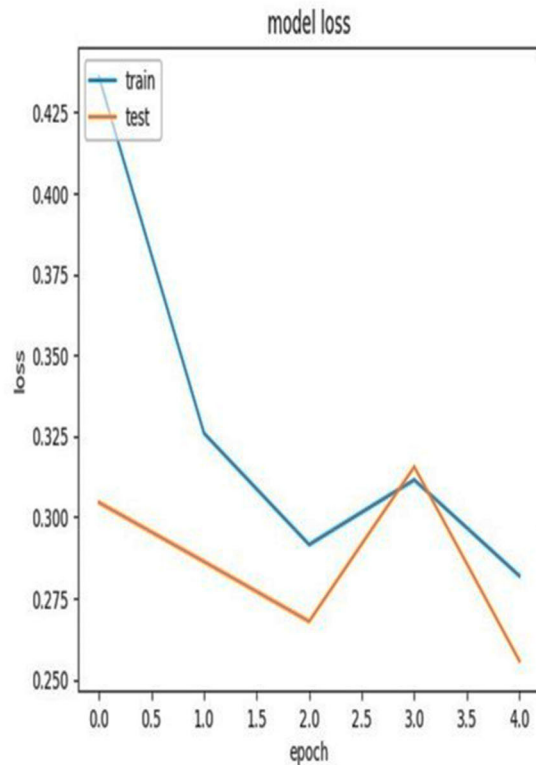
Confusion matrices provide a bird's-eye view of the classification outcomes of the model, showing the number of true positives and negatives followed by false positives and negatives. A careful analysis of these matrices



7 (a) Confusion Matrix

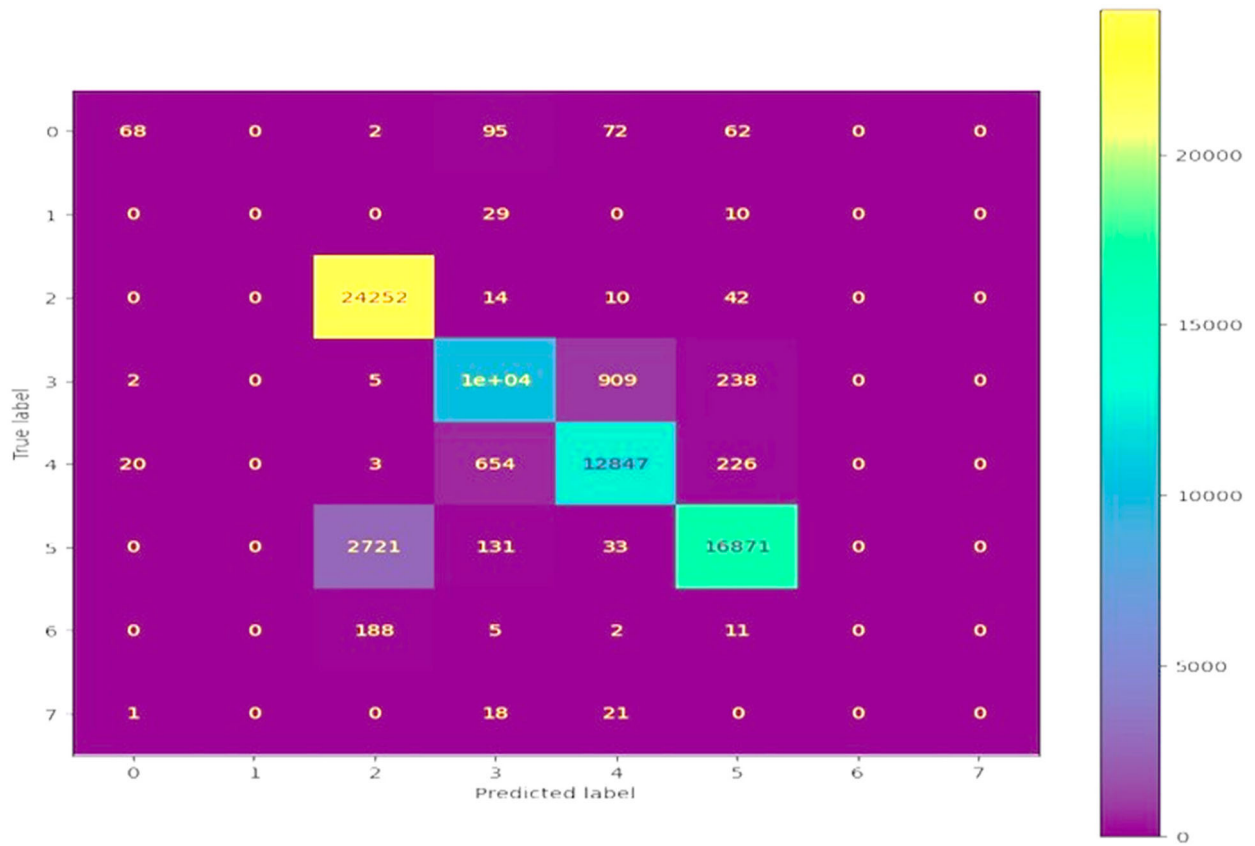


7 (b) Model Accuracy

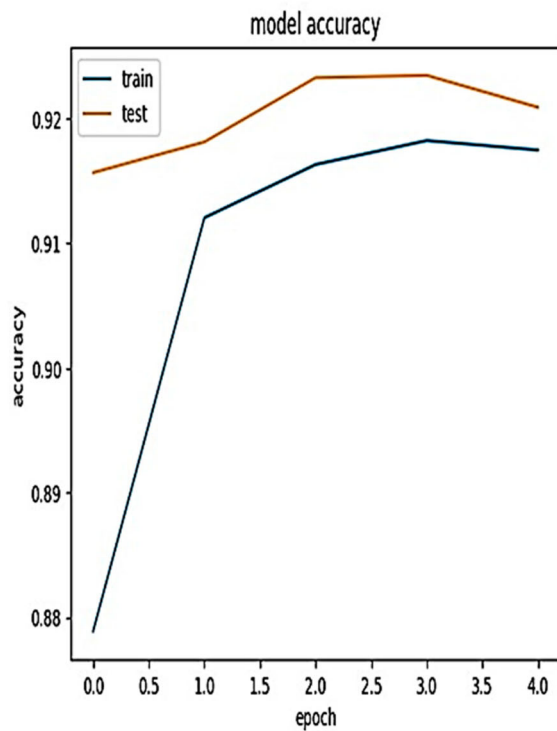


7 (c) Model Loss

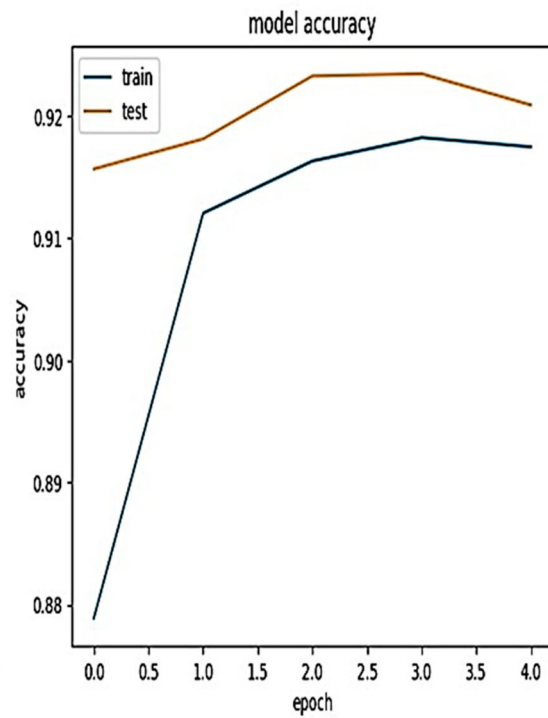
Figure 7. (a) Confusion Matrix, (b) Accuracy, (c) Loss GRU.



8 (a) Confusion Matrix



8 (b) Model Accuracy



8 (c) Model Loss

Figure 8. (a) Confusion Matrix, (b) Accuracy, (c) Loss Ensemble model.

Algorithm 1: Deep Hybrid Learning Model (GRU+LSTM)

```

1 Get X=(x1, x2, x3) from SDN Dynamic Dataset.
2 x1', x2', x3' = label encoding (x1, x2, x3)
3 x1', x2', x3' = Standardization (x1', x2', x3')
4 Conduct convolution processing.
5 for z = 1; z ≤ Z; do do
6   for y = 1; y ≤ Y; do do
7     Create backwardGRUcell and backwardLSTMcell by Sstate;
8     Create forwardGRUcell and forwardLSTMcell by Sstate;
9     Connect BLSTMnet by backwardLSTMcell and forwardLSTMcell
10    Connect BGRUnet by forwardGRUcell and backwardGRUcell
11    Initialize BLSTMnet and BGRUnet by seed.
12    Get hidden states h(z,y) of BLSTMnet and BGRUnet.
13   end
14 end
15 Add a full connection layer, whose value is 320;
16 Add a dropout, whose value is 0.1;
17 for each hidden state in 1: h(z,y); do do
18   Obtain h(z,y) implicit representation u1 through a non-linear transformation.
19   Generate a random initialization matrix vw
20   Obtain the normalized importance weight coefficient β1
21   Get fine-grained feature s via β1 and h(z,y).
22 end
23 Add a full connection layer whose value is 1024.
24 Add a full connection layer, whose value is 10;
25 Return accuracy, F1Score.

```

helps in assessing the strengths and weaknesses of the model in terms of correctly and incorrectly classified instances.

Various metrics, such as precision, recall, F1 score, and Area Under the Receiver Operating Characteristic Curve (AUC-ROC), are used to conduct a comprehensive evaluation of the performance. These metrics can be calculated from the confusion matrices and provide insights into the model's ability to create a fine balance between precision and recall for various classes. Precision reflects the accuracy of positive predictions, while recall denotes the capability of a model to capture the entire set of relevant instances. In general, the F1 score is calculated by combining the recall and precision values into a single measure. The AUC-ROC calculates the trade-off that exists between the True Positive Rate (TPR) and the False Positive Rate (FPR).

4.3. Defender – IPS

Scapy Defender is a security tool designed to enhance the network defense by intercepting and dropping specific packets based on the predefined criteria. By leveraging the capabilities of Scapy library, the Scapy Defender enables the users to create custom rules as well as filters to inspect these packets. When finding a packet that matches the defined criteria, the

Scapy Defender takes action to drop the packet. Thus, it prevents the packet from reaching the intended destination. This proactive approach towards filtering the packet is particularly valuable in network security as the phenomenon allows the administrators to thwart any sort of potential threats, malicious activities, or unwanted traffic. By offering a flexible and programmable framework, the Scapy Defender empowers the network defenders to tailor their defense mechanisms, according to specific requirements, thereby bolstering the resilience of networks against different types of cyber threats. Figure 9 provides a visual representation of the features or patterns used to filter and differentiate the attacks and normal data.

5. Performance metrics

5.1. Threshold metrics

- F-Measure: It is a measure used to calculate the accuracy of a classification model. The value corresponds to the harmonic mean between recall and precision, as shown in Equation (1).

$$F - \text{Measure} = \frac{2 * (\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})} \quad (1)$$

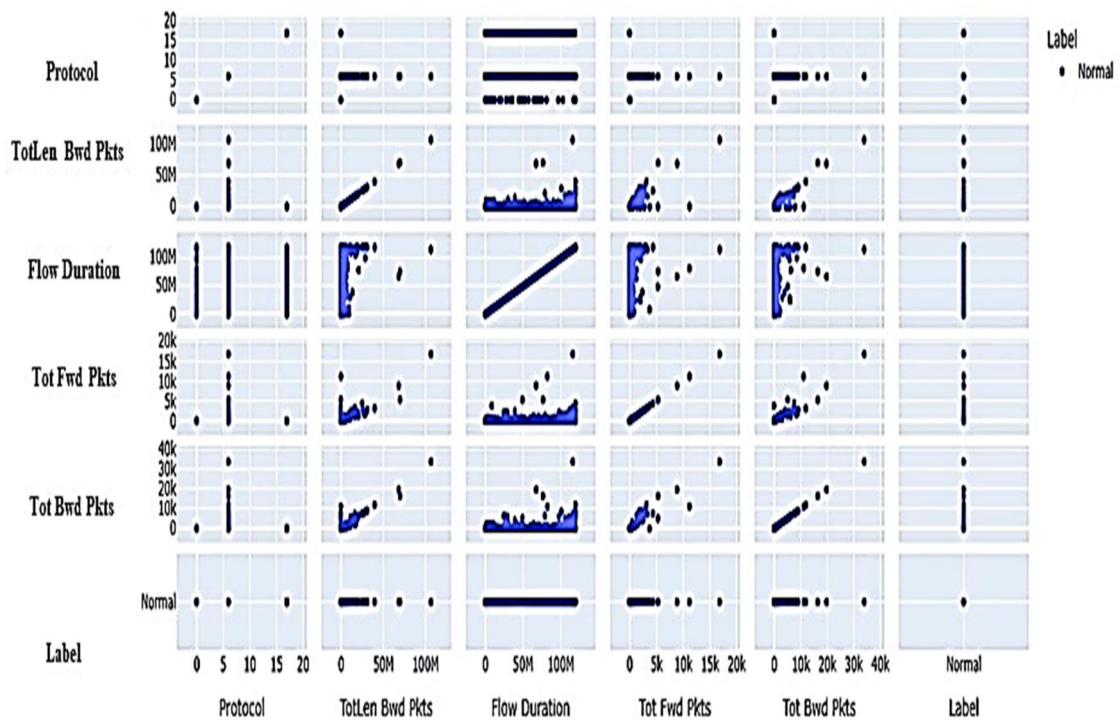
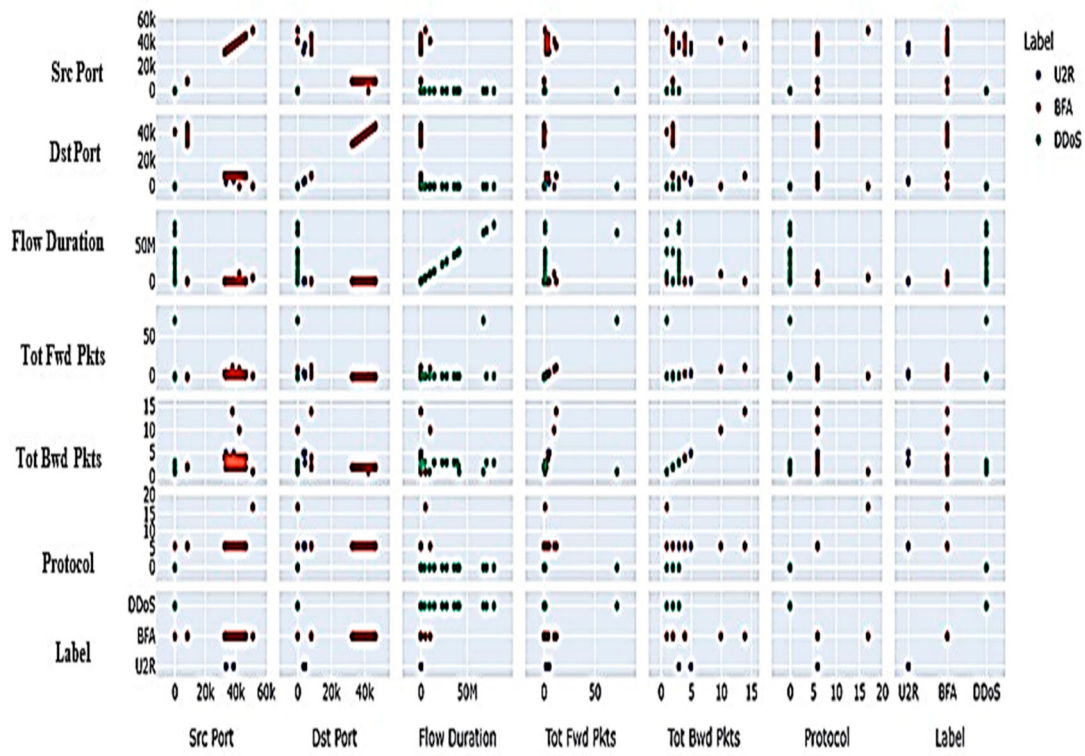


Figure 9. Filter Attack and Normal.

5.2. Ranking metrics

- Accuracy: It is determined as the percentage of correctly predicted labels from the total number of instances in a dataset, as shown in Equation (2).

$$\text{Accuracy} = \frac{\text{Number of correct Prediction}}{\text{Total number of Prediction}} \quad (2)$$

- Recall: This measure is calculated as the proportion of original positive instances that are correctly identified by the model as positive, as shown in Equation (3).

$$\text{Recall} = \frac{\text{True Positive}}{(\text{True Positive} + \text{False Negative})} \quad (3)$$

- Precision: Precision corresponds to a measure for the accuracy of a measurement or a system that produces the measurements, as shown in Equation (4).

Precision

$$= \frac{\text{Number of correctly estimated measurements}}{\text{Total number of measurements}} * 100 \quad (4)$$

Table 2 shows the accuracy, precision, recall, and F-measure for the GRU, LSTM, and Ensemble models. In Table 3, the proposed GRU + LSTM model stands out with an impressive accuracy rate of 97.8%, surpassing competing models in performance. By amalgamating the strengths of both GRU and LSTM architectures, our model effectively captures and processes sequential data, yielding highly accurate predictions. Compared to alternative approaches, which often fall short

Table 2. Accuracy, precision, recall and F-measure in GRU, LSTM and Ensemble Model.

MODEL	LSTM				GRU				ENSEMBLE MODEL			
	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support
0	0.928571	0.26087	0.4073	299	0.771429	0.271	0.401	299	0.747253	0.227	0.34872	299
1	0	0	0	39	0	0	0	39	0	0	0	39
2	0.892811	0.997409	0.9422	24318	0.893131	0.997	0.942	24318	0.892569	0.997	0.94203	24318
3	0.912932	0.903192	0.908	11435	0.917811	0.906	0.912	11435	0.915739	0.899	0.90733	11435
4	0.924539	0.944509	0.9344	13750	0.927599	0.947	0.937	13750	0.924644	0.934	0.92946	13750
5	0.974872	0.850324	0.9083	19756	0.975344	0.853	0.9101	19756	0.924644	0.854	0.90665	19756
6	0	0	0	206	0	0	0	206	0	0	0	206
7	0	0	0	40	0	0	0	40	0	0	0	40
Accuracy	0.922741	0.922741	0.9227	0.9227	0.924201	0.924	0.9242	0.9242	0.920908	0.921	0.92091	0.92091
Macro avg	0.579216	0.494538	0.5125	69843	0.560664	0.497	0.5128	69843	0.555809	0.489	0.50427	69843
Weighted Avg	0.922074	0.922741	0.9194	69843	0.923047	0.924	0.9209	69843	0.919259	0.921	0.91748	69843

Table 3. Accuracy of proposed system vs existing system.

Author	Year	Methodology	Accuracy
Almasri T et al [22]	2022	Naïve Bayes	93.50%
Bhardwaj et.al [16]	2022	Clonal Selection Model	95%
Luo K et.al [20]	2023	Optimized Decision Tree	96.80%
Sharma et.al [2]	2023	Dynamic SDN model	83.00%
Logeswari G et al. [23]	2023	CFS-LGBM Model	95%
Maheswaran N et al. [24]	2022	Random Forest	86%
L Yang et al. [25]	2022	Griffin Real Time Network	97%
Proposed Model		GRU + LSTM	97.80%

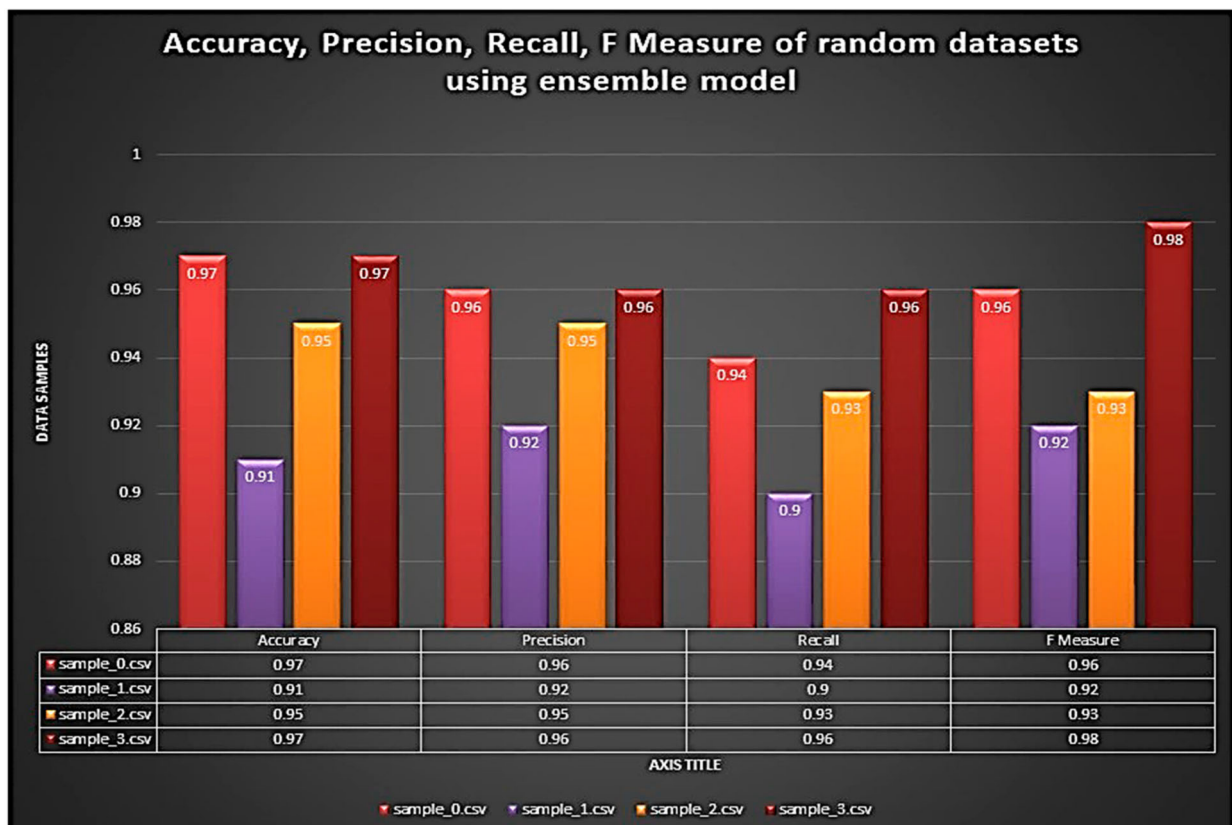


Figure 10. Performance outcome of four randomly picked samples.

in accuracy and efficiency, our model excels in handling complex patterns and long-term dependencies within the data. Figure 10 shows the outcomes from four randomly selected datasets. These datasets contain both attack data and normal data. The classification rate is similar to accuracy.

6. Conclusion

This research proposes and validates a new approach to enhance network security in SDN environments using a Deep Hybrid IDS model. The proposed model leverages both ML techniques and DNNs to detect known and unknown attacks through integrated anomaly-based intrusion detection and signature-based intrusion detection techniques. The model was tested on a network, and the results confirm that it outperformed all other conventional IDSs in terms of false positive rate and detection accuracy. Overall, the proposed deep hybrid IDS model is a promising approach to overcome the security challenges encountered in SDN environments and can contribute to the development of highly advanced and effective network security solutions.

7. Future work

Future work for enhancing network security in SDN environments using the deep hybrid IDS model should focus on improving the accuracy and scalability of the models through further experimentation and validation under different SDN environments. Another area of focus is the integration of the deep hybrid IDS model with the rest of the security solutions to develop a highly cohesive security framework for SDN networks. Additionally, other ML and DNNs can be applied to achieve refined results in terms of enhanced network security. Finally, adapting the proposed model to address emerging security threats in SDN environments, such as insider attacks, can also be a potential research area. Overall, future work should aim to improve and adapt the proposed model to ensure the security and resilience of SDN networks against evolving security threats.

Disclosure statement

No potential conflict of interest was reported by the author(s).

References

- [1] Chaudhary R, Aujla G.S, Kumar N. and Chouhan P.K. A comprehensive survey on software-defined networking for smart communities. *Int J Commun Syst.* 2022;n/a(n/a):e5296. doi:10.1002/dac.5296
- [2] Sharma A, Balasubramanian V, Kamruzzaman J. A novel dynamic software-defined networking approach to neutralize traffic burst. *Computers.* 2023. doi:10.3390/computers12070131
- [3] Al-Shareeda M, Alsadhan AA, Qasim HH, & Manickam S. Software defined networking for internet of things: review, techniques, challenges, and future directions. 2024;13:638–647. doi:10.11591/eei.v13i1.6386
- [4] Karunarathne G, Kulawansa K, Firdhous M. (2018). *Wireless communication technologies in internet of things: a critical evaluation.* doi:10.1109/ICONIC.2018.8601226
- [5] Etxezarreta X, Garitano I, Iturbe M, Zurutuza U. Software-Defined Networking approaches for intrusion response in Industrial Control Systems: a survey. *Int J Crit Infrastruct Prot.* 2023;42:100615. doi:10.1016/j.ijcip.2023.100615
- [6] Raikar MM, S M M. MullaMM Software Defined Internet of Things using lightweight protocol. *Procedia Comput Sci.* 2020;171:1409–1418. doi:10.1016/j.procs.2020.04.151
- [7] Siddiqui S, Hameed S, Shah SA, Ahmad I, Aneiba A, Draheim D, Dustdar S. Toward Software-Defined Networking-Based IoT Frameworks: a systematic literature review, taxonomy, open challenges and prospects. *IEEE Access.* 2022;10:70850–70901. doi:10.1109/ACCESS.2022.3188311
- [8] Vimal V, Muruganatham R, Prabha R, Arularasan AN, Nandal P, Chanthirasekaran K, Reddy Ranabothu G. Enhance Software-Defined Network Security with IoT for strengthen the encryption of information access control. *Comput Intell Neurosci.* 2022: 4437507. doi:10.1155/2022/4437507
- [9] Kumhar M, Bhatia J. Software-defined networks-enabled fog computing for IoT-based healthcare: security, challenges and opportunities. *Secur Priv.* 2023;6(5): e291. doi:10.1002/spy2.291
- [10] Zeleke EM, Melaku HM, Mengistu FG. Efficient Intrusion Detection System for SDN Orchestrated Internet of Things. *J Comput Netw Commun.* Edited by I. Ali. 2021: 5593214. doi:10.1155/2021/5593214
- [11] Saheed YK, Misra S. A voting gray wolf optimizer-based ensemble learning models for intrusion detection in the Internet of Things. *Int J Inf Secur.* 2024. doi:10.1007/s10207-023-00803-x
- [12] Shoab F, Chow YW, Vlahu-Gjorgievska E, and Nguyen C. Mitigating Timing Side-Channel Attacks in Software-Defined Networks: detection and response. *Telecom.* 2023: 877–900. doi:10.3390/telecom4040038
- [13] Najar AA, Manohar Naik S. Cyber-Secure SDN: a CNN-based approach for efficient detection and mitigation of DDoS attacks. *Comput Secur.* 2024;139:103716. doi:10.1016/j.cose.2024.103716
- [14] Rajan D, Aravindhar DD. Detection and mitigation of DDOS attack in SDN environment using hybrid CNN-LSTM. *Migr Lett.* 2023;20:407–419. doi:10.59670/ml.v20iS13.6472
- [15] Ahmed MR, et al. 2022. Intrusion Detection System in Software-Defined Networks using machine learning and deep learning techniques –a comprehensive survey.
- [16] Bhardwaj A, Tyagi R, Sharma N, Khare A, Punia MS, Garg VK. Network intrusion detection in software defined networking with self-organized constraint-based intelligent learning framework. *Meas: Sens.* 2022;24:100580. doi:10.1016/j.measen.2022.100580
- [17] Chaganti R, Suliman W, Ravi V, Dua A. Deep learning approach for SDN-enabled Intrusion Detection System in IoT networks. *Information.* 2023. doi:10.3390/info14010041
- [18] ElSayed MS, Le-Khac NA, Albahar MA, 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. doi:10.1016/j.jnca.2021.103160
- [19] Alzahrani AO, Alenazi MJF. Designing a Network Intrusion Detection System Based on machine learning for software defined networks. *Future Internet.* 2021. doi:10.3390/fi13050111
- [20] Luo K. A distributed SDN-based Intrusion Detection System for IoT using optimized forests. *PLoS One.* 2023;18(8):e0290694. doi:10.1371/journal.pone.0290694
- [21] Radhi Hadi M, Saher Mohammed A. (2022). 'A novel approach to Network Intrusion Detection System using Deep Learning for SDN: Futuristic Approach', in *Machine learning & applications*. Academy and Industry Research Collaboration Center (AIRCC) (CMLA 2022). doi:10.5121/csit.2022.121106
- [22] AlMasri T, Snober M, Al-Haija Q. (2022). 'IDPS-SDN-ML: an intrusion detection and prevention system using Software-Defined Networks and Machine Learning', in *1st International Conference on Smart Technology*. Surakarta, Indonesia.
- [23] Logeswari G, Bose S, Anitha T. An Intrusion Detection System for SDN using machine learning. *Intell Autom Soft Comput.* 2023;35(1):867–880. doi:10.32604/iasc.2023.026769
- [24] Maheswaran N, Bose S, Logeswari G, et al. Hybrid Intrusion Detection System Using Machine Learning Algorithm. In: Khanna A, Polkowski Z, Castillo O, editors. *Proceedings of data analytics and management. lecture notes in networks and systems*, vol. 572. Singapore: Springer; 2023. p. 333–346. doi:10.1007/978-981-19-7615-5_30.
- [25] Yang L, Song Y, Gao S, Xiao B. Griffin: real-time Network Intrusion Detection System via ensemble of autoencoder in SDN. in *IEEE Trans Netw Serv Manag.* September 2022;19(3):2269–2281. doi:10.1109/TNSM.2022.3175710