Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system

Article *in* Computers & Electrical Engineering · June 2022

DOI: 10.1016/j.compeleceng.2022.108156

3 authors:

Vinayakumar Ravi
Prince Mohammad University
291 PUBLICATIONS **6,263** CITATIONS

SEE PROFILE

Rajasekhar Chaganti
Expedia Group
33 PUBLICATIONS **185** CITATIONS

SEE PROFILE

Mamoun Alazab
Charles Darwin University
332 PUBLICATIONS **11,740** CITATIONS

SEE PROFILE

Noname manuscript No.
(will be inserted by the editor)

Recurrent Deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system

Vinayakumar Ravi · Rajasekhar Chaganti · Mamoun Alazab

the date of receipt and acceptance should be inserted later

Abstract This work proposes an end-to-end model for network attack detection and network attack classification using deep learning-based recurrent models. The proposed model extracts the features of hidden layers of recurrent models and further employs a kernel-based principal component analysis (KPCA) feature selection approach to identify optimal features. Finally, the optimal features of recurrent models are fused together and classification is done using an ensemble meta-classifier. Experimental analysis and results of the proposed method on more than one benchmark network intrusion dataset show that the proposed method performed better than the existing methods and other most commonly used machine learning and deep learning models. In particular, the proposed method showed maximum accuracy 99% in network attacks detection and 97% network attacks classification using the SDN-IoT dataset. Similar performances were obtained by the proposed model on other network intrusion datasets such as KDD-Cup-1999, UNSW-NB15, WSN-DS, and CICIDS-2017.

Keywords Cyber-physical systems, Cyberattacks, Cybercrime, Intrusion detection, Recurrent model, Deep learning, Feature fusion, Meta-classifier

1 Introduction

Cyber-physical systems (CPSs) enable physical devices with sensing capabilities to communicate with controllers or the internet as needed. The communication channels tend to be using either wireless or short distance communication technologies to constantly update the physical device status or physical environment

Vinayakumar Ravi
Center for Artificial Intelligence, Prince Mohammad Bin Fahd University,
Khobar, Saudi Arabia.
E-mail: vravi@pmu.edu.sa

Rajasekhar Chaganti
Dept. of Computer Science, University of Texas at San Antonio, San Antonio, Texas 78249, USA.
E-mail: Raj.chaganti2@gmail.com

Mamoun Alazab
College of Engineering, IT and Environment, Charles Darwin University, NT, Australia.
E-mail: alazab.m@ieee.org

Corresponding Author: Vinayakumar Ravi

condition to the controller or remote server. The recent advancements in sensor technologies and wireless communications make the CPS used in various application sectors such as material manufacturing with automated supply chain, smart industries including transportation, electronics, aviation and chemical industries, and many more with advanced capabilities. The emergence of CPS applications in many sectors also opens up new security issues and challenges to protect the infrastructure or confidential data from cyberattacks [23]. The attacks may not only include cyberattacks due to the internet-connected devices but also physical attacks, which can result in system failures or supply chain disruption. So, the CPS security is much more challenging than securing the conventional IT and network infrastructure [23].

The well-known CPS attack stuxnet was first observed in 2008, when it hit the Iranian nuclear plant with a malware worm [1]. Since then, there have been a number of major attacks reported targeting various industries in CPS including the recent ransomware attack on US gas and oil pipelines [2]. The attack mitigation in CPS requires a deep understanding of the underlying physical systems and embedded technology in addition to the cybersecurity controls. As more CPS systems connect to the internet day by day, the CPS systems can become a part of the bot group if compromised and then used to launch distributed denial of service (DDoS) attacks on targeted organizations. So, the public exposure of the CPS devices should be prevented to limit the attack surface. The device layer security implementation is highly unlikely in CPS and has limited control over the physical object sensing devices. The application layer security controls can be implemented to protect the CPS infrastructure. However, there is limited visibility of the device behavior, too late to detect the attacks on the physical devices and also may not cover all the security attacks in CPS.

The network layer-based security detection mechanisms such as intrusion detection systems and intrusion prevention systems are effective in conventional network settings, where the data plane and control plane are integrated within the switching and routing devices. The recent advancement in networks such as software-defined networks (SDN) possesses various advantages in terms of reliability, security, centralized control, and more. SDN separates the control plane from the data plane and can be used to have a holistic overview of the switching and routing devices in the network to monitor, control, and customize the network traffic [3]. The end devices in CPS connected to the nearest switching or routing devices can be controlled using an open flow protocol from the controller. The network communication protocol like Modbus, and CAN traffic monitoring may not be possible with conventional antivirus solutions and require specific tools, which can parse the industrial control system (ICS) network protocols. For example, Hello Flooding attacks are specific to CPS systems and require monitoring the network traffic to detect the attacks. So, SDN can benefit to improve the security by programming the network devices and monitoring the suspicious network traffic in CPS applications such as ICS. The customized applications built on top of the controller presents a unique approach to solving CPS security. The Internet of things (IoT) applications also benefits by using the SDN architecture in the network layer. The traffic generated from IoT protocols such as MQTT, CoAP etc. is also being monitored to improve the IoT security.

The CPS and IoT-based intrusion detection systems are mainly categorized as the rule and anomaly-based detection systems [24]. The rule-based intrusion detection only identifies the known security events. The zero-day vulnerability exploits are almost impossible to identify using rule-based techniques. Additionally, the rule-based database should be updated frequently with new indicators of compromise rules, which is a cumbersome process. The anomaly-based detection techniques identify the unknown security events and improve the security posture. However, the false positives are very high and require the security workforce to continuously tune the alerts. In recent times, data analytics-based solutions are proposed to improve threat detection. Although the machine learning techniques employed for CPS security achieved moderate accuracy [1], the feature selection has been a challenge and requires deep domain knowledge. CPS application features vary from industry to industry and the predefined set of features is not applicable in the CPS security context.

Deep learning-based solutions are popular for intrusion detection [2]. In these works, authors have used most well-known deep learning models such as recurrent neural network (RNN), long short-term memory (LSTM), gated recurrent unit (GRU), Convolution neural network (CNN), and a hybrid of deep learning models. These networks have employed more than one layer in the network. The multiple layers in the network learn the model and extract the high-level features in the input. There were a number of deep learning-based works focused on SDN-based network intrusion detection systems [3] [4] [5] to detect network anomalies. Although the prior art on network intrusion detection shows that the deep learning architecture shows better performance on large-scale network datasets, there is little information available on how the hidden layers represent the data, which in turn arise the question of the quality of feature representation in hidden layers. The learning process may fail, or it could not provide an optimal feature representation, especially in the case of small-and-medium-sized problems. Thus, in this work, recurrent deep learning models such as RNN, LSTM, and GRU hidden layers features are studied in detail. The main contributions of the proposed work are given below

- Recurrent models hidden layers feature fusion for network intrusion detection.
- Detailed investigations and analysis of various classical machine learning algorithms such as Naive Bayes, Logistic Regression, k-nearest neighbor (KNN), Decision Tree, and Random Forest for network intrusion detection.
- Detailed investigation and analysis of various recurrent deep learning models such as RNN, LSTM, and GRU for network intrusion detection
- Recurrent model hidden layers feature sets dimensionality reduction using KPCA.
- The performance of the proposed approach, existing machine learning and deep learning algorithms are shown on more than one network intrusion dataset.
- Ensemble meta-classifiers for network intrusion detection and classification.
- Feature fusion of recurrent models for network intrusion detection.
- Feature visualization using t-distributed stochastic neighbor embedding (t-SNE) to ensure that the learned representation of features was meaningful for network intrusion detection.

The rest of the paper is organized as follows: Section 2 discusses the related works of existing network intrusion detection systems. Section 3 includes information on the proposed recurrent deep learning-based feature fusion approach for intrusion detection. The description of network-based intrusion detection datasets is discussed in Section 4. Statistical metrics used to evaluate the performance of the proposed model are discussed in Section 5. Detailed results and their discussions are discussed in Section 6. The paper concludes with the conclusion and future works in Section 7.

2 Literature survey on Intrusion detection systems

This section includes a detailed literature survey on existing intrusion detection systems in CPSs. In contrast to IDS systems used in conventional networks, the IDS systems in CPS also monitor the physical components tagged with sensors to detect anomalies and alert the users. We firstly discuss the SDN and IoT related IDS solutions proposed in the literature and then describe IDS solutions proposed in the CPSs.

Ajaeiya et al. [6] presented a Random Forest-based intrusion detection system to detect the network attacks in SDN. The network flow records and their statistics were used as a feature to train the machine learning model. Various network attacks such as brute force attacks, Reconnaissance attacks were tested to validate the machine learning-based intrusion detection. Even though the authors reported promising results to detect network attacks, the datasets used for the experiments may not be applicable to attacks on CPS applications. Hadem et al. [7] used the combination of SVM and selective logging with IP traceback to detect attacks in SDN-based intrusion detection. The authors claim that the memory resources were saved during the implementation of the IDS. The conventional network dataset NSL-KDD was used to run the experiments and obtained 87.74% detection accuracy. These datasets may not be applicable to CPS applications, as the CPS involves the physical objects states connected through sensors to sense the activity and communicated to the receiver end and these activities may not be captured in the conventional network traffic.

Ye et al. [8] proposed a machine learning-based SVM approach for DDoS attacks detection in SDN. The network protocol TCP, UDP, and ICMP attacks were tested using network tuple characteristic features. The dataset was generated using the network traffic, which was created in a testing environment with tools like hping3. The reported results show that the detection accuracy 95.34% is achieved to identify UDP flooding attacks. Latha et al. [9] presented the comparison of various machine learning algorithms for intrusion detection in SDN. The dataset NSL-KDD was used to identify the network attacks. All these prior art solutions solely focused on detecting the network attacks in conventional networks and are not suitable to validate the attacks in SDN and IoT infrastructure. Dey et al. [10] implemented the machine learning solutions with various feature selection combinations for intrusion detection in SDN. But, the dataset NSL-KDD used to conduct the experiments was not applicable to the SDN environment. One of the future works was to create real-time traffic using the SDN testbed to validate the obtained results for the NSL-KDD dataset.

Neural network models were also used in the state of the art for attacks detection and classification in SDN and IoT networks. ElSayed et al. [2] presented a hybrid intrusion detection model in SDN using both the CNN and random forest. A new regularization technique is also proposed to improve intrusion detection. The reported results showed that the hybrid model improved the detection accuracy by 97%. However, the features did not cover the attack vector in CPS, which is necessary to validate the attacks in CPS. Hannache et al. [4] implemented a neural network-based traffic classifier in Floodlight SDN controller to classify the network attacks. They have considered TCP, UDP, and ICMP flooding attacks for simulating and classifying the attacks. The authors reported that they achieved between 96% and 96.5% accuracy for TCP SYN, UDP, and ICMP flood classification. But, the authors did not use the traffic classifier in the context of the CPS. So, we may not consider these techniques to assess the deep learning model detection efficiency in CPS. In [11], the authors proposed a Deep neural network-based model for flow-based anomaly detection in SDN. The authors leveraged the NSL-KDD dataset to perform the experiments and test the proposed model. According to the authors, DNN performed better than the SVM, Naive Bayes, and decision tree for a subset of features taken from the NSL-KDD dataset. But, the reported accuracy 75.05% for DNN is very low and there is still room for performance improvement. Overall, we see that deep learning models performed better than machine learning techniques in SDN-based intrusion detection systems. However, there were few works using SDN and IoT-based network traffic to detect and classify the IoT-based network attacks. This may be due to the lack of datasets representing the IoT network traffic captured in the SDN environment.

Literature also shows that machine learning and deep learning techniques were used specific to the CPS-based attacks. Mitchell et al. [12] proposed behavioral specification-based intrusion detection in medical CPSs. The unusual/abnormal patient behavior is monitored and detected using the proposed behavioral-based model. The authors show that their method performs better than the compared two anomaly detection techniques. Han et al. [13] reviewed the intrusion detection techniques in CPSs and discussed the open challenges in this field. The authors emphasize the nature of CPSs and how the intrusion detection types are used to detect the system intrusions in CPS. The application of artificial neural networks in CPS may face computational constraints and a lack of datasets. Belenko et al. [14] proposed GAN-based intrusion detection to identify security anomalies in CPSs. Sadreazami et al. [15] proposed a distributed blind intrusion network framework to improve the detection performance in CPS compared to conventional intrusion detection techniques. The sensor data is modelled using graph signal representation and then modelled using Gaussian

Table 1: Summary of intrsuion detection systems

Reference	Network and Environment	Dataset	Method	Comparison Parameter
Tang et al. [3]	SDN and non-IoT	NSL-KDD	DNN	Accuracy 75.75%
Ajaeiya [6]	SDN and non-IoT	Custom dataset	Random Forest	Accuracy 85.4%
Lath et al. [9]	SDN and non-IoT	NSL-KDD	Decision Tree	Accuracy 71%
Tang et al. [11]	SDN and non-IoT	NSL-KDD	GRU-RNN	Accuracy 89%
ElSayed et al. [2]	SDN and non-IoT	InSDN	CNN+random forest	Accuracy 97%
Hadem et al. [7]	SDN and non-IoT	NSL-KDD	SVM	Accuracy 95.98%
Dey et al. [1]	SDN and non-IoT	NSL-KDD	Random Forest	Accuracy 81.946%
Alzahrani et al. [22]	SDN and non-IoT	NSL-KDD	XGBoost	Attack detection: 95.5 Attack Classification: 95.95%
Hannache et al. [4]	SDN and non-IoT	Custom dataset	Neural Network	Accuracy 96.13%
Formby et al. [18]	CPS	Custom Dataset	FF-ANN	Accuracy 92%
Thakur et al. [5]	CPS	CICIDS2017	GSAE	precision(web) 0.7906
Dutta et al. [17]	CPS	SWaT	DAE	Accuracy 93.01%
Althobaiti et al. [16]	CPS	NSL-KDD, CICIDS	ANN + GRU	Accuracy 98.5%
Belenko et al. [14]	CPS	Custom Dataset	ANN	-
Proposed	CPS + SDN	SDN-IoT	RNN, GRU, LSTM, KPCA, random forest, SVM	Attack detection: 99% Attack Classification: 97%
		KDD-Cup- 1999		Attack detection: 99% Attack Classification: 89%
		UNSW-NB15		Attack detection-99% Attack Classification-99%
		WSN-DS		Attack detection-98% Attack Classification-98%
		CICIDS-2017		Attack detection-99% Attack Classification-98%

Markov random fields. In the end, the temporal analysis of the network traffic behavior is performed to detect the intrusions in CPS.

Althobaiti et al. [16] proposed an intelligent cognitive computing-based intrusion detection system for industrial CPS. The proposed method includes following all the steps in the data pipeline such as data acquisition, preprocessing and feature selection, classification, and optimization to detect anomalies. GAN model is used to classify the normal and anomalies in the industrial CPS. Dutta et al. [17] proposed a multivariate anomaly-based intrusion detection system for CPSs. Denoising autoencoder deep learning architecture is used to detect anomalies based on reconstruction error. The authors reported an accuracy of more than 90% and claim that the performance is better than other anomaly techniques in CPS. Formby et al. [18] proposed two new fingerprint attacks detection in CPS to improve intrusion detection capabilities. The datasets collected from real-time and lab environments were used to classify the fingerprinting attacks in CPS. The authors reported that they achieved 99% classification accuracy when applied to real-time datasets. Junejo et al. [19] proposed a behavioral-based supervised machine learning intrusion detection system in CPS. The authors created a testbed, which replicates all the physical and control components to train the supervised machine learning models and detect the anomalies in CPS. The reported results obtained high precision and low false positives. Haller et al. [20] proposed a three-phase intrusion detection design to address the cyber and physical dimensions of the system. The sensitivity analysis, cross association, and optimizing the IDS techniques were used to design resilient CPS. Mitechell et al. [21] proposed a hierarchical performance intrusion detection in CPSs. The authors developed two techniques positional discontinuity and enviro consistency to detect intrusion and evaluated these two techniques' performance using the hierarchical method.

The CNN-based approach is employed for smart grid intrusion detection in CPS environment [24]. The authors did a detailed experimental analysis of the proposed method on both public and private datasets and the proposed method showed better performances in both the datasets. The CPS-based model study is discussed in detail for the SCADA environment [25].

Features from the penultimate layers of deep learning models were extracted and passed into KPCA for dimensionality reduction. Further, the reduced features were fused and passed into a meta-classifier for classification. In these studies, the importance of KPCA is shown in detail and it facilitates the extraction of important features for classification. Meta-classifier has shown better performances compared to other single-level classifiers.

The details summary of intrusion detection systems is reported in Table 1. Our prior art discussion clearly shows that very limited work has been done for detecting the CPS attacks in SDN using Deep learning techniques due to the lack of an available dataset targeting the IoT attacks in SDN. Additionally, the deep learning techniques don't show the details of how the learning process happened in the middle neural network layers. So, it is very important to understand the learning process, particularly testing small and medium datasets, which are quite common in CPS-based application-specific datasets covering CPS-based attacks. We have addressed all these problems in this paper.

3 Proposed Recurrent deep learning-based feature fusion approach for intrusion detection

The proposed recurrent deep learning-based feature fusion approach for intrusion detection is shown in Figure 1 and the proposed algorithm is shown in Algorithm 1. The framework is divided into two subcategories as Network traffic data collection testbed setup and network traffic analysis. The network traffic data collection testbed setup is same as SDN-IoT [23]. It contains the network topology that includes 5 IoT devices, 4 attacker hosts, a server and 2 benign hosts to mimic the normal and attack traffic. All these devices are connected through Openflow virtual switch and Open Network Operating System (ONOS) acts as the SDN controller. The IoT services considered in this work are weather station, smart fridge, motion-activated lights, remotely activated garage door, and smart thermostat. Using these services, IoT network traffic was collected. The analysis of network traffic data, attacks detection, and attacks classification is done in network traffic analysis subsystems. This submodule contains the following:

Recurrent deep learning model feature extraction: In this work, three types of recurrent models were leveraged and they are RNN, LSTM, and GRU. Recurrent models are capable of extracting optimal features that are required for network traffic attacks detection and classification from the network traffic data. Since the network traffic flow has sequence and temporal characteristics, this work employs three different recurrent-based models. The literature survey shows that the recurrent-based models were effective for sequence and temporal data analysis. RNN is an enhanced model of a classical neural network and it contains a unit instead of a neuron. A unit acts as a short-term memory to store the information while data processing over the sequence. Mathematically, RNN is represented as follows:

$$hr_t = Sigmoid\left(w_{xhr}x_t + w_{hrhr}hr_{t-1} + b_{hr}\right) \tag{1}$$

$$Or_t = soft\max(w_{hror}hr_t + b_{or}) \tag{2}$$

Where x denotes an input, hr denotes hidden layer representation, Or denotes output layer representation, b terms denote bias, w term denotes weight matrices and t terms as subscripts in the equations denote time-step. RNN is trained using backpropagation through time (BPTT). However, the method leads to vanishing and error gradient while processing information across many time steps. To avoid this, LSTM was proposed and it has a memory instead of a simple unit. This memory has gating functions such as input gate, output

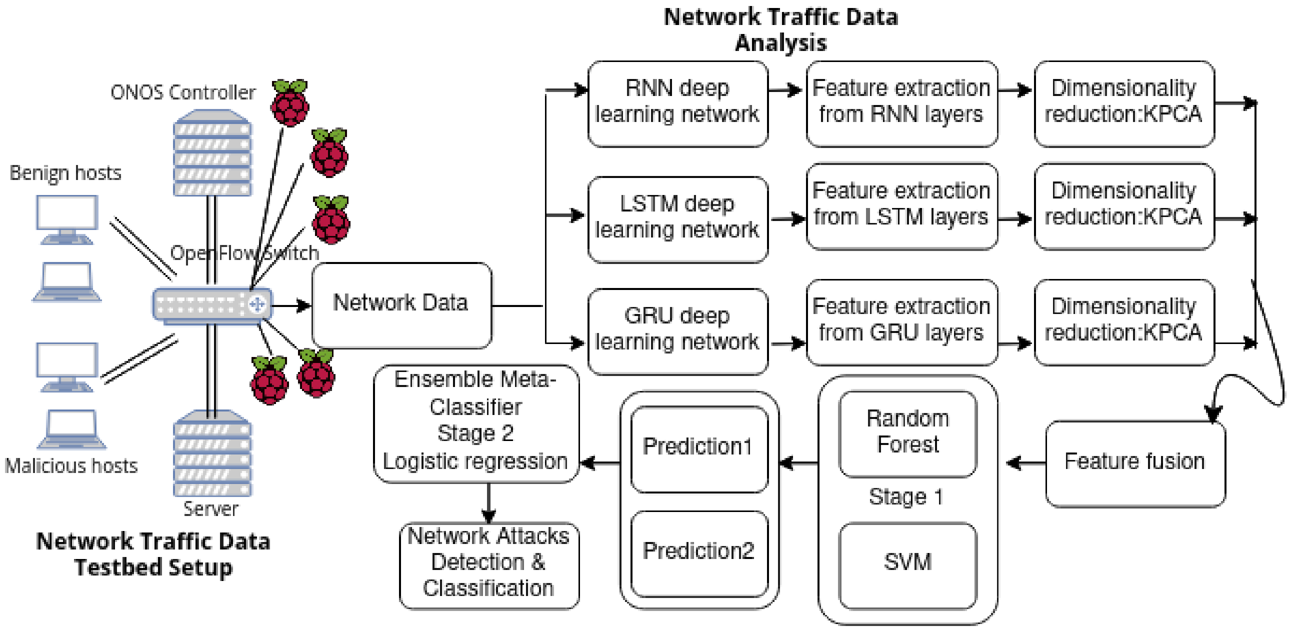

Network Traffic Data Analysis
ONOS Controller
Benign hosts
OpenFlow Switch
Network Data
Malicious hosts
Server
Network Traffic Data Testbed Setup
RNN deep learning network
LSTM deep learning network
GRU deep learning network
Feature extraction from RNN layers
Feature extraction from LSTM layers
Feature extraction from GRU layers
Dimensionality reduction:KPCA
Dimensionality reduction:KPCA
Dimensionality reduction:KPCA
Feature fusion
Random Forest
Stage 1
SVM
Prediction1
Prediction2
Ensemble Meta-Classifier Stage 2 Logistic regression
Network Attacks Detection & Classification

Fig. 1: Proposed intrusion detection system

gate, and forget gate to control the stored information in the memory across many time steps. Mathematically, LSTM is represented as follows:

$$ir_t = sigmoid\left(w_{xir}x_t + w_{hrir}hr_{t-1} + w_{crir}cr_{t-1} + b_{ir}\right) \tag{3}$$

$$fr_t = sigmoid\left(w_{xfr}x_t + w_{hrfr}hr_{t-1} + w_{crfr}cr_{t-1} + b_{fr}\right) \tag{4}$$

$$cr_t = fr_t \odot cr_{t-1} + ir_t \odot \tanh\left(w_{xcr}x_t + w_{hrcr}hr_{t-1} + b_{cr}\right) \tag{5}$$

$$or_t = sigmoid\left(w_{xor}x_t + w_{hror}h_{t-1} + w_{cror}cr_t + b_{or}\right) \tag{6}$$

$$hr_t = or_t \odot \tanh\left(cr_t\right) \tag{7}$$

Where ir_t denotes input gate representation, fr_t denotes forget gate representation, cr_t denote memory cell representation, or_t denote output layer representation, and hr_t denote hidden layer representation, b and w term denotes bias and weight matrices respectively, and t terms as subscripts in the equations denotes time-step.

The operations of LSTM are expensive and recently an improved method of LSTM typically called GRU was introduced and it contains only two gating functions such as forget gate and update gate. GRU takes less time for training and is more efficient. However, LSTM is complex but suitable to remember longer sequence information. Mathematically, GRU is represented as follows:

$$ir-fr_t = sigmoid\left(w_{xir-fr}x_t + w_{hrir-fr}hr_{t-1} + b_{ir-fr}\right) \quad (8)$$

$$fr_t = sigmoid\left(w_{xfr}x_t + w_{hrfr-fr}hr_{t-1} + b_{fr}\right) \quad (9)$$

$$cr_t = \tanh\left(w_{xcr}x_t + w_{hrcr}\left(fr \odot hr_{t-1}\right) + b_{cr}\right) \quad (10)$$

$$hr_t = fr \odot hr_{t-1} + (1-fr) \odot cr \quad (11)$$

Where $ir-fr_t$ is an update gate representation, and t terms as subscripts in the equations denotes timestep

All three recurrent layers contain two hidden layers. Each hidden layer in RNN, LSTM, and GRO model contains 32 units. From these hidden layers, we extract features for training, validation, and testing datasets. The dimension of the extracted features of training, validation and testing is *MxN*, where *M* denotes the number of network connection traffic samples and *N* is the dimension of recurrent hidden layers feature i.e. 32.

Dimensionality reduction using KPCA: The features of recurrent hidden layers are passed into KPCA. It is an extension of PCA and the dimensionality of the data can be reduced using KPCA without information loss. The kernel function helps to map data into a higher dimensional plane with the aim to form a linear decision boundary. Since the features of hidden layers in a deep learning network are highly non-linearly separable, the radial-basis function (RBF) was used as a kernel function in KPCA. Though there are many types of kernel functions available in the literature, a detailed study of these kernel functions is not studied in this work. This is important and there may be a chance that other kernel functions can perform better than RBF. This type of study can be considered one of the significant directions for future work.. In this work, the dimensions of the features of the deep learning-based recurrent models are reduced from 32 to 16.

Recurrent-based model hidden layers feature fusion: Feature fusion is an important approach in recent deep learning architectures. It combines features from different layers of the network. In this work, a simple feature fusion approach that is a concatenation of recurrent hidden layers is employed on the RNN, LSTM, and GRU hidden layer features.

Network Attacks Detection and Classification: The fused features of recurrent hidden layers are passed into an ensemble meta-classifier or stacking classifier. It combines more than one classification model. The meta-classifier is a two-stage method in which the first stage employs random forest and SVM for prediction and next, the predictions are stacked in the second stage, finally, network attacks detection and classification are done using logistic regression. This type of model has the capability to improve the performance that is shown by individual classifiers.

4 Description of network intrusion detection datasets

There are limited datasets publicly available for intrusion detection that support IoT in the SDN environment. In this work, SDN-IoT [23] database is used. IoT network simulation was done using a mininet tool. The network topology includes 4 attacker hosts, a server and 2 benign hosts to mimic the Normal and attack traffic. All these devices connected through Openflow virtual switch and Open Network Operating System (ONOS) acts as SDN controller. Two different datasets are generated by varying the number of IoT devices connected were 5 and 10. The IoT services were a weather station, smart fridge, motion-activated lights, remotely activated garage door, and smart thermostat. The simulations covered 5 different attack types

Algorithm 1: Network intrusion detection system in CPS

```
Input: Network Connections $N_1$, $N_2$,., $N_n$.
Output: Labels $y_1$, $y_2$,..., $y_n$. The labels are Normal, Attack, and types of Attack
for each network connection $N_i$ do
    // Recurrent-based model architecture
    feature $FRNN$ = RNN($N_i$);
    feature $FLSTM$ = LSTM($N_i$);
    feature $FGRU$ = GRU($N_i$);

    // Reduced feature representation using KPCA
    Dimensionality reduction $DRNN$ = KPCA($FRNN$);
    Dimensionality reduction $DLSTM$ = KPCA($FLSTM$);
    Dimensionality reduction $DGRU$ = KPCA($FGRU$);
    // Feature Fusion
    Feature concatenation $FF = DRNN + DLSTM + DGRU$;

    // Stage 1: Base-level classifiers
    prediction $P_1$ = SVM($FF$);
    prediction $P_2$ = Random Forest($FF$);

    // Stage 2: Meta-level classifier
    Compute $y_i$ = Logistic Regression($P_1$, $P_2$);
    $y_i$: Normal, Attack, and types of Attack;
end
```

such as DoS, DDoS, Port Scanning, OS Fingerprinting, and Fuzzing for attack classification. The authors leveraged hping3, nmap, and boofuzz tools to generate different types of attack traffic. The traffic flows are collected from the switches using an SDN application running on top of the controller and features are generated during traffic generation. In this work, the dataset from 5 IoT devices was used to evaluate the performances of the proposed method and other classical methods for attack detection and attack classification. The training, validation, and testing of the SDN-IoT dataset contains 63000 (10635 Normal network connections and 52365 Attack network connections), 36750 (5971 Normal network connections and 30779 Attack network connections), and 110250 (18394 Normal network connections and 91856 Attack network connections) network connections for training, validation, and testing respectively in network attacks detection. In the network attacks classification, the SDN-IoT dataset contains 63000 (10635 Normal, 10420 DoS, 10567 DDoS, 10541 Port Scanning, 10441 OS Fingerprinting, and 10396 Fuzzing), 36750 (5971 Normal, 6180 DoS, 6158 DDoS, 6097 Port Scanning, 6086 OS Fingerprinting, 6258 Fuzzing), and 110250 (18394 Normal, 18400 DoS, 18275 DDoS, 18362 Port Scanning, 18473 OS Fingerprinting, 18346 Fuzzing) network connections for training, validation, and testing respectively.

In addition to SDN-IoT, benchmark well-known network intrusion datasets such as KDD-Cup-1999[1], UNSW-NB15[2], WSN-DS[3], and CICIDS-2017[4] are used and the detailed performances of the proposed method is shown. Since KDD-Cup-1999, UNSW-NB15, WSN-DS, and CICIDS-2017 are big network-based intrusion detection datasets, in this work only a subset of the datasets are considered. The detailed performances of the proposed method were evaluated on these additional network-based datasets just to ensure that the proposed model is robust and generalizable i.e. the model is able to show similar performances on similar network-based datasets. KDD-Cup-1999 is a very old dataset and the model developed using this

[1] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[2] https://research.unsw.edu.au/projects/unsw-nb15-dataset

[3] https://sel.psu.edu.sa/Research/datasets/2016_WSN-DS.php

[4] https://www.unb.ca/cic/datasets/ids-2017.html

dataset is not accurate for today's network configuration environment. But, KDD-Cup-1999 is a standard dataset and it contains diverse attacks. Most importantly, the training and testing datasets are completely unseen and they are from a different probability distribution. Thus, the model that performs better on the testing dataset of KDD-Cup-1999 is important to show that the model has the capability to detect unseen attacks. UNSW-NB15 dataset is developed in 2015 and the authors have discussed that this dataset is more suitable for today's network configuration compared to KDDD-Cup-1999. CICIDS-2017 dataset is developed in 2017 and the authors discussed that the CICIDS-2017 dataset is considered to be good compared to UNSW-NB15 and KDD-Cup-1999. To show that the proposed method performs better in SDN-based IoT environments, network-based environments, and wireless networks, this work has considered various datasets and the detailed performances of the proposed method evaluated in this work.

The KDD-Cup-1999 dataset was prepared by a team from the 1998 DARPA intrusion detection challenge. This dataset contains 41 network features and network connections are grouped into either Normal or Attack. In addition, the network connection of Attack class is further categorized into its categories denial-of-service (DoS), DDoS, remote-2-local (R2L), and user-2-root (U2R). It is one of the old datasets and still, the database is used in several network-related research for benchmarking purposes of new algorithms. For network attacks detection, the training dataset of KDD-Cup-1999 contains 48628 Normal network connections and 198382 Attack network connections. Validation and Testing of KDD-Cup-1999 contain 12137 and 36513 Normal network connections respectively. The KDD-Cup-1999 testing dataset has 36513 Normal network connections and 148745 Attack network connections. For network attacks classification, during training, the KDD-Cup-1999 dataset contains 48628, 195682, 2100, 28, 572 network connections for Normal, DoS, Probe, U2R, and R2L respectively. The testing of KDD-Cup-1999 contains 36513, 146808, 1503, 19, 415 network connections for Normal, DoS, Probe, U2R, and R2L respectively. During validation, the total samples of Normal, DoS, Probe, U2R, and R2L are 12137, 48968, 504, 5, and 139 respectively.

The UNSW-NB15 dataset was built by the cyber security research team of Australian Centre for Cyber Security (ACCS) and they have also claimed that the dataset resolves some of the main issues that were reported by researchers on the KDD-Cup-1999 dataset. The dataset contains 42 features and the network connection contains 9 different attacks. The attacks are Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shell code, Worms. In network attacks detection, Normal network connections of UNSW-NB15 training, validation, and testing are 12340, 4710, and 13950 respectively and Attack network connections of UNSW-NB15 training, validation, and testing are 22856, 5582, and 16924 respectively. During network attacks classification, UNSW-NB15 training contains 18340, 18, 184, 1774, 9538, 3053, 5542, 2106, 269, and 342 network connections for Normal, Worms, Shell Code, Reconnaissance, Generic, Fuzzers, Exploits, DoS, Backddoors, and Analysis respectively. Testing of UNSW-NB15 contains 13950, 23, 149, 1267, 7009, 2244, 4202, 1531, 238, and 261 network connections for Normal, Worms, Shell Code, Reconnaissance, Generic, Fuzzers, Exploits, DoS, Backddoors, and Analysis respectively. The validation set of UNSW-NB15 includes 4710, 3, 45, 455, 2324, 765, 1388, 452, 76, and 74 network connections for Normal, Worms, Shell Code, Reconnaissance, Generic, Fuzzers, Exploits, DoS, Backddoors, and Analysis respectively.

WSN-DS is a wireless sensor network intrusion dataset and it contains 23 features, and 4 different attacks such as DoS, Blackhole, Grayhole, Flooding, and Scheduling. For network attacks detection, training, validation, and testing sets of WSN-DS contains 118853, 29822, 89371 Normal network connections and 12277, 2961, and 8976 Attack network connections respectively. For network attacks classification, the training dataset of WSN-DS contains 118853, 3588, 5183, 1195, and 2311 network connections for Normal, Blackhole, Grayhole, Flooding, and Scheduling respectively. Validation dataset of WSN-DS contains 29822, 826, 1280, 286, and 569 network connections for Normal, Blackhole, Grayhole, Flooding, and Scheduling respectively. The testing dataset of WSN-DS contains 89371, 2619, 3754, 837, and 1766 network connections for Normal, Blackhole, Grayhole, Flooding, and Scheduling respectively.

CICIDS-2017 is a network intrusion dataset collected by Canadian Institute for Cybersecurity. The dataset contains 7 different attacks such as SSH-Patator, FTP-Patator, DoS, Web, Bot, DDoS, and PortScan. In network attacks detection, Normal network connections of CICIDS-2017 training, validation, and testing are 29727, 7561, and 22712 respectively. Attack network connections of CICIDS-2017 training, validation, and testing are 17022, 4127, and 12350 respectively. For network attacks classification, training of CICIDS-2017 contains 29727, 2564, 3529, 3058, 1013, 782, 3057, and 3019 network connections for Normal, SSH-Patator, FTP-Patator, DoS, Web, Bot, DDoS, and PortScan respectively. Validation of CICIDS-2017 contains 7561, 587, 838, 726, 240, 170, 781, 785 network connections for Normal, SSH-Patator, FTP-Patator, DoS, Web, Bot, DDoS, and PortScan respectively. Testing of CICIDS-2017 contains 22712, 1849, 2633, 2216, 746, 548, 2162, and 2196 network connections for Normal, SSH-Patator, FTP-Patator, DoS, Web, Bot, DDoS, and PortScan respectively.

5 Statistical Measures

To evaluate the performances of the machine learning and deep learning-based models for network intrusion detection, the following metrics are considered in this work.

Accuracy is the classifier's ability to classify all positive samples as positive and all negative samples as negative.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{12}$$

Precision is a measure of the classifier's ability to not mark a negative sample as positive.

$$Precision = \frac{TP}{TP+FP} \tag{13}$$

Recall is a measure of the classifier's ability to mark all positive samples as positive.

$$Recall = \frac{TP}{TP+FN} \tag{14}$$

F1-Score is the weighted average of Precision and Recall.

$$F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{15}$$

The accuracy, precision, recall, and F1-Score are estimated from the values obtained from the confusion matrix. The confusion matrix is a $2x2$ matrix that displays and compares actual values with the predicted values. It composes of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) and details are given below

- TP: the number of Attack samples classified as Attack.
- TN: the number of Normal samples classified as Normal.
- FP: the number of Normal samples classified as Attack.
- FN: the number of Attack samples classified as Normal.

Since the dataset is imbalanced, macro precision, macro recall, and macro F1-Score are used in this work. Macro metric computes the Precision, Recall, and F1-Score for each class and returns the average without considering the proportion for each class in the network intrusion dataset.

6 Experiments, Results, and Discussions

The machine learning and deep learning algorithms implementation were done using Scikit-learn[5] and Keras[6] with TensorFlow[7] as backend. All the models were run inside the Kaggle GPU environment with K80 GPU. Backpropagation through the time gradient-based approach is used to train all the recurrent deep learning models in this work.

Recurrent-based deep learning models such as RNN, LSTM, and GRU optimal performance implicitly depends on network parameters and network architecture. In the beginning, the moderate size recurrent deep learning models were used. This contains an input layer, a hidden recurrent layer, and an output layer. The input layer contains 26 neurons and the output layer contains M neurons in network attack detection and N neurons in network attacks classification. Here the M and N denote the number of classes in the dataset. The intrusion datasets used in this study are normalized into the [0-1] range using L2 normalization. We run several trials of experiments to identify the best parameters for network parameters and network architecture for the following

- Detecting the network attacks i.e. classifying the network connection into either Normal or Attack.
- Classifying the network attacks of network connection records into their attack categories.

The network parameters are the number of units/memory cells in RNN, LSTM, and GRU. We run experiments with a number of units 16, 32, 64, and 128 in RNN, LSTM, and GRU with one hidden layer, learning rate 0.001, adam optimizer, and batch size 64. The experiments were run for 25 epochs. Recurrent models with 32 units in the hidden layer performed better than the 16, 64, and 128. Thus, units of recurrent models were set to 32. The same network was run with a learning rate of 0.2, 0.1, 0.001, and 0.0001 and experiments with lower learning rates performed better, mainly 0.001 showed better performance compared to others. So, 0.001 was set as the value for the learning rate. To identify the best parameters for batch size, the same models were run for batch sizes 16, 32, 64, and 128. The experiments with 64 batch sizes performed better compared to others. Next, the experiments were run for epochs of 25, 50, and 75. The experiments with 50 showed better training accuracy and validation accuracy and after 50 epochs the performance of training and validation declined. This may be due to overfitting and to avoid this, the experiments were stopped at epochs 50. The detailed performances of the recurrent deep learning models in terms of training accuracy and validation accuracy for attacks detection are shown in Figure 2 and train loss and validation loss for attacks detection are shown in Figure 3. Same experiments were run for network attacks classification and the detailed performances of training accuracy and validation accuracy are shown in Figure 4 and train loss and validation loss are shown in Figure 5. Overall, the figures show an increase in accuracy of training, and validation and a decrease in loss of training, and validation across 50 epochs. When the experiments were run for more than 50 epochs, there was no improvement and we decided to stop experiments at epochs 50. Most importantly, the accuracy and loss of training and validation fluctuated. During training, the training and validation datasets are completely disjoint, however, the network connection data samples were shuffled in both the training and validation. Recurrent deep learning models have shown similar performances for both the network attacks detection and network attacks classification. The training and validation performances of GRU are better than LSTM and RNN and RNN performances were less compared to LSTM and GRU. This may be due to the reason that RNN has an issue of vanishing and error gradient and a simple unit in RNN will not facilitate storing and carrying out the information for longer time steps. GRU performed better than the LSTM and in addition, GRU is computationally efficient compared to LSTM.

[5] https://scikit-learn.org/

[6] https://keras.io/

[7] https://www.tensorflow.org/

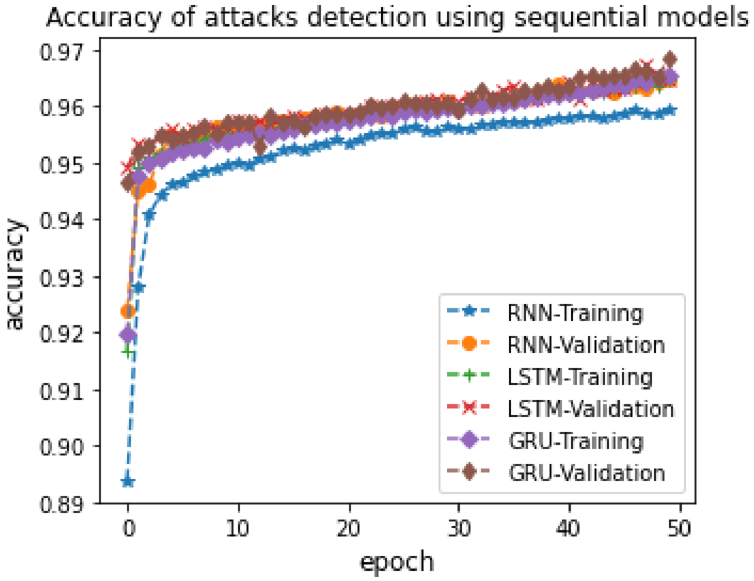


Fig. 2: Training accuracy of RNN, LSTM, and GRU models on SDN-IoT dataset for network attacks detection

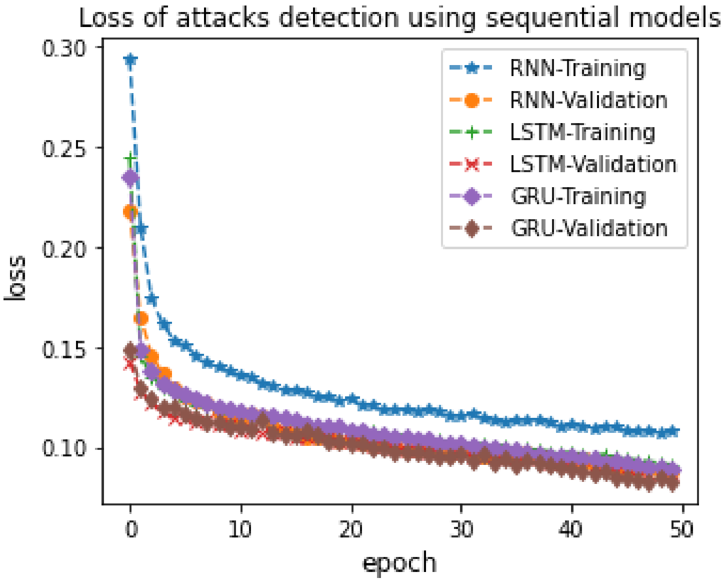


Fig. 3: Training loss of RNN, LSTM, and GRU models on SDN-IoT dataset for network attacks detection

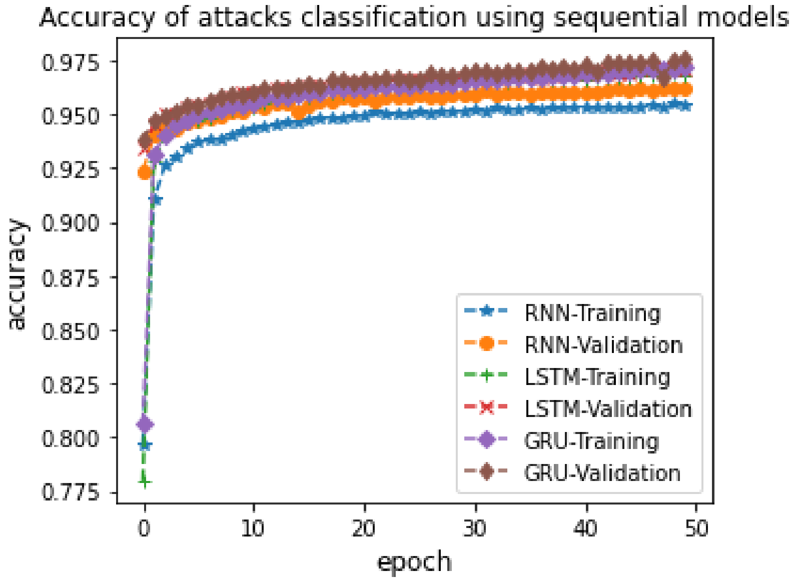


Fig. 4: Training accuracy of RNN, LSTM, and GRU models on SDN-IoT dataset for network attacks classification

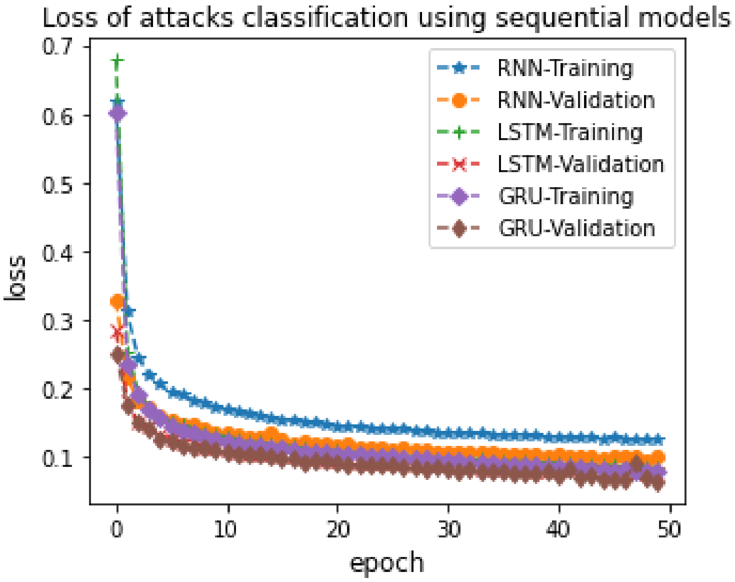


Fig. 5: Training accuracy of RNN, LSTM, and GRU models on SDN-IoT dataset for network attacks classification

To identify the network structure for recurrent deep learning models, the following network structures are considered

- RNN 1 layer with 32 units
- RNN 2 layers with 32 units
- GRU 1 layer with 32 units
- GRU 2 layers with 32 units
- LSTM 1 layer with 32 units
- LSTM 2 layers with 32 units

The performances of the RNN, GRU, and LSTM models were good with 2 layers compared to 1 layer. In addition, the layers were increased from 2 to 3, however, there was no improvement in the training accuracy, validation accuracy, and training loss, validation loss. Thus we decided to go with 2 layers for RNN, LSTM, and GRU. To avoid overfitting, a dropout of 0.0001 was used in between the recurrent layers.

All three recurrent deep learning models contain a fully connected layer with 16 neurons before the output layer. In a fully connected layer, all neurons have connections to every neuron in the succeeding layer. In a fully connected layer, ReLU activation function was used. In the output layer, the number of neurons was set to 1 in attack detection and 6 neurons for SDN-IoT datasets. The neurons are Normal, DoS, DDoS, Port Scanning, OS Fingerprinting, and Fuzzing. For KDD-Cup-1999, the models contain 5 neurons and they are Normal, DoS, Probe, R2L, U2R. In UNSW-NB15, the output layer contains 10 neurons and they are Normal, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shell code, and Worms. For WSN-DS, the output layer contains Normal, Blackhole, Grayhole, Flooding, and Scheduling as 5 neurons. For CICIDS-2017, the output layer contains Normal, SSH-Patator, FTP-Patator, DoS, Web, Bot, DDoS, and PortScan as 8 neurons. During training, binary cross-entropy loss function was used for attack detection and categorical cross-entropy was used for attack classification. Binary cross-entropy is mathematically defined as follows:

$$loss\left(e_{IDS}, p_{IDS}\right) = -\frac{1}{N}\sum_{j=1}^{N}\left[e_{IDS}\log p_{IDS} + \left[1 - e_{IDS}\right]\log\left[1 - p_{IDS}\right]\right] \tag{16}$$

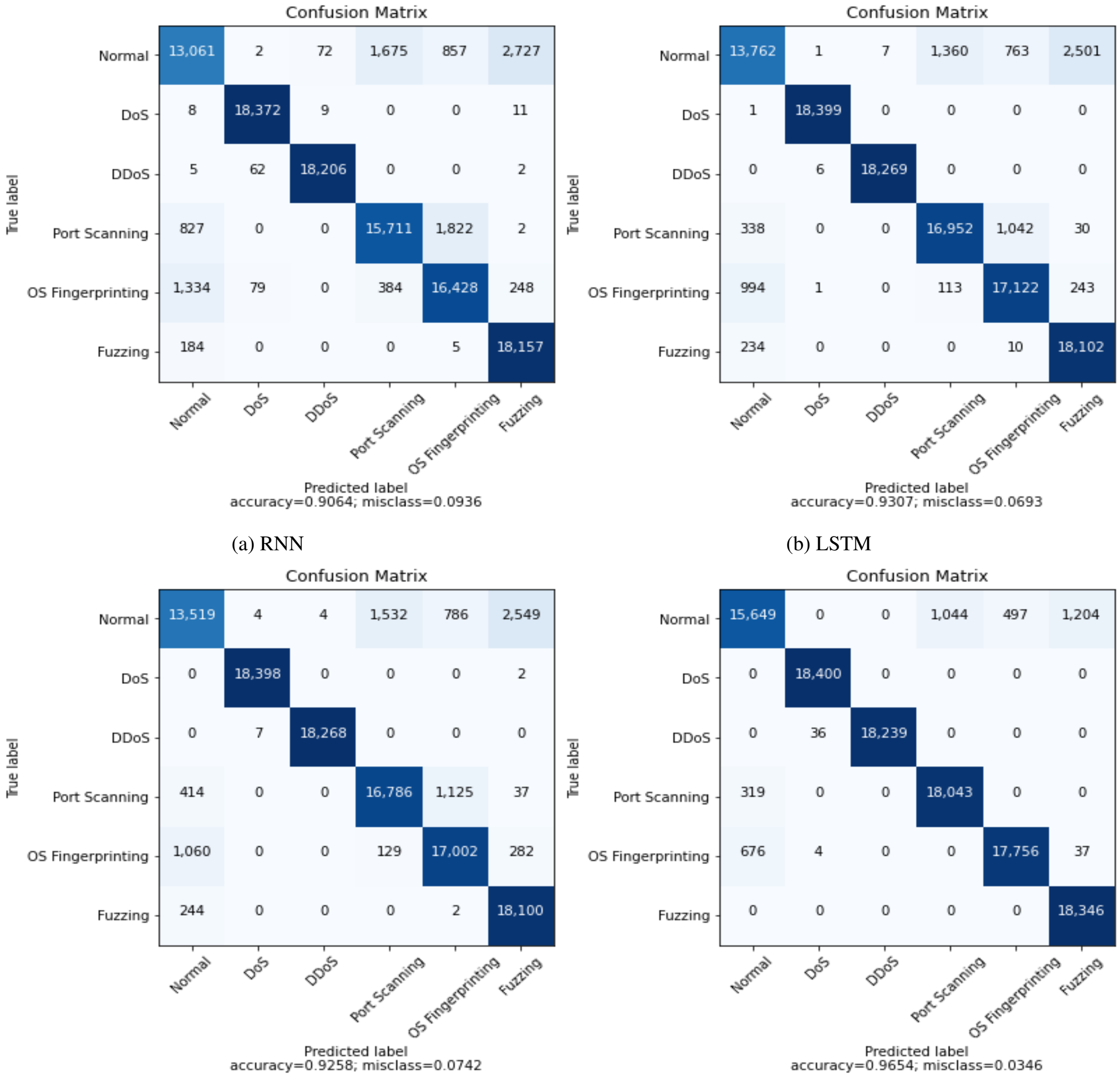
Where e_{IDS} denotes expected class label and p_{IDS} denoted predicted class label.

Categorical cross entropy is mathematically defined as follows:

$$loss\left(pe_{IDS}, pp_{IDS}\right) = -\textstyle\sum_{x} pe_{IDS}\left(x\right)\log pp_{IDS}\left(x\right) \tag{17}$$

The testing results for RNN, LSTM, and GRU recurrent deep learning models are included in Table 2 for network attacks detection and Table 3 for network attacks classification. RNN, LSTM, and GRU showed 92%, 96%, and 96% accuracy respectively for network attacks detection. For network attacks classification, the accuracy of RNN, LSTM, and GRU was 91%, 93% and 93% respectively. GRU showed a less misclassification rate of 0.0742 and the misclassification rate of RNN and GRU was 0.0936 and 0.0693 respectively. All three models have less misclassification for DoS attacks, DDoS attacks and still, there are misclassifications among Normal, Port Scanning, OS fingerprinting, and Fuzzing. A further detailed analysis has to be done to avoid this misclassification. The detailed performance of classification is shown in the confusion matrix in Figure 6.

To avoid misclassification and increase the performance of the model for network attacks detection and network attacks classification, this work extracts the features of hidden recurrent layers. The feature dimension of the extracted feature was *MxN* where *M* denotes the number of samples in training, validation, and testing datasets and *N* defines the number of features i.e. 32 for both the hidden layers in RNN, LSTM, and



Confusion Matrix

	Normal	DoS	DDoS	Port Scanning	OS Fingerprinting	Fuzzing
Normal	13,061	2	72	1,675	857	2,727
DoS	8	18,372	9	0	0	11
DDoS	5	62	18,206	0	0	2
Port Scanning	827	0	0	15,711	1,822	2
OS Fingerprinting	1,334	79	0	384	16,428	248
Fuzzing	184	0	0	0	5	18,157

(a) RNN

Confusion Matrix

	Normal	DoS	DDoS	Port Scanning	OS Fingerprinting	Fuzzing
Normal	13,762	1	7	1,360	763	2,501
DoS	1	18,399	0	0	0	0
DDoS	0	6	18,269	0	0	0
Port Scanning	338	0	0	16,952	1,042	30
OS Fingerprinting	994	1	0	113	17,122	243
Fuzzing	234	0	0	0	10	18,102

(b) LSTM

Confusion Matrix

	Normal	DoS	DDoS	Port Scanning	OS Fingerprinting	Fuzzing
Normal	13,519	4	4	1,532	786	2,549
DoS	0	18,398	0	0	0	2
DDoS	0	7	18,268	0	0	0
Port Scanning	414	0	0	16,786	1,125	37
OS Fingerprinting	1,060	0	0	129	17,002	282
Fuzzing	244	0	0	0	2	18,100

(c) GRU

Confusion Matrix

	Normal	DoS	DDoS	Port Scanning	OS Fingerprinting	Fuzzing
Normal	15,649	0	0	1,044	497	1,204
DoS	0	18,400	0	0	0	0
DDoS	0	36	18,239	0	0	0
Port Scanning	319	0	0	18,043	0	0
OS Fingerprinting	676	4	0	0	17,756	37
Fuzzing	0	0	0	0	0	18,346

(d) Proposed Approach

Fig. 6: Confusion matrix for network attacks classification.

GRU models. The penultimate layer features of RNN, LSTM, and GRU were shown in Figure 7 for network attacks detection and Figure 8 for network attacks classification. As the figure shows that the data samples were non-linear. Next, the features were passed into KPCA. Since the data distribution of penultimate layers of recurrent deep learning models is nonlinear, KPCA is preferred over PCA. In KPCA, the kernel parameter was set to RBF and other parameters of KPCA use the default values initialized by sklearn implementation. The dimension of the features was reduced from 32 to 16. Further, the reduced features were concatenated and the dimension of the feature set for all three recurrent models is $Mx32$. The final dimension of concate-

Table 2: Results for SDN-IoT attacks detection

Model	Accuracy	Precision	Recall	F1 score	Confusion Matrix
Naive Bayes	0.87	0.86	0.63	0.66	TN=4840 FP=13554 FN=841 TP=91015
Logistic Regression	0.88	0.86	0.65	0.70	TN=5919 FP=12475 FN=1159 TP=90697
KNN	0.89	0.87	0.70	0.74	TN=7486 FP=10908 FN=1378 TP=90478
Decision Tree	0.85	0.73	0.64	0.66	TN=5991 FP=12403 FN=4389 TP=87467
Random Forest	0.90	0.83	0.80	0.81	TN=12146 FP=6248 FN=4767 TP=87089
Recurrent Neural Network	0.92	0.94	0.76	0.82	TN=9808 FP=8586 FN=395 TP=91461
Long Short-term Memory	0.96	0.95	0.92	0.93	TN=15599 FP=2795 FN=1253 TP=90603
Gated Recurrent Unit	0.96	0.95	0.92	0.93	TN=15747 FP=2647 FN=1294 TP=90562
Proposed Approach	0.99	0.99	0.99	0.99	TN=17962 FP=432 FN=413 TP=91443

Table 3: Results for SDN-IoT attacks classification

Model	Accuracy	Precision	Recall	F1 score
Naive Bayes	0.58	0.53	0.57	0.52
Logistic Regression	0.67	0.68	0.67	0.66
Decision Tree	0.73	0.75	0.73	0.72
Random Forest	0.76	0.77	0.76	0.75
Recurrent Neural Network	0.91	0.91	0.91	0.90
Long Short-term Memory	0.93	0.93	0.93	0.93
Gated Recurrent Unit	0.93	0.93	0.93	0.92
Proposed Approach	0.97	0.97	0.97	0.96

Table 4: Detailed results of proposed approach for attacks classification

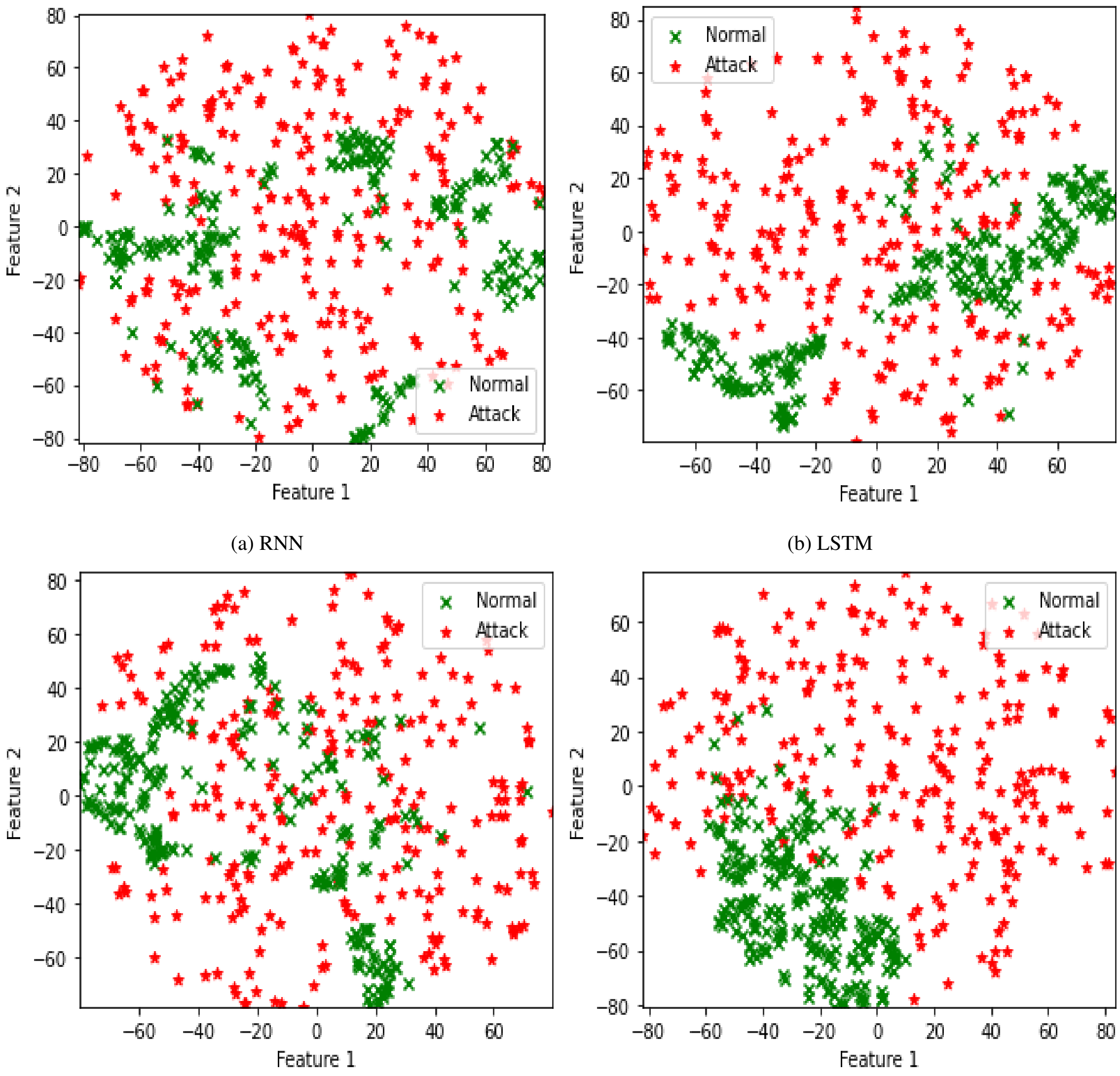
Category	Precision	Recall	F1 score
Normal	0.94	0.85	0.89
DoS	1.0	1.0	1.0
DDoS	1.0	1.0	1.0
Port Scanning	0.95	0.98	0.96
OS Fingerprinting	0.97	0.96	0.97
Fuzzing	0.94	1.00	0.97

nated features that were from RNN, LSTM, and GRU is Mx96. Further, the Mx96 feature set was passed into ensemble meta-classifiers. The first stage contains random forest and SVM. In random forest, the number of estimators is set to 50 and c and kernel set to 1.0 and RBF respectively. The predictions of stage 1 are passed into stage 2 for classification in ensemble meta-classifiers. The detailed results of the proposed method is shown in Table 2 for network attacks detection and Table 3 for network attacks classification. The proposed approach accuracy is 99% and 97% for network attacks detection and network attacks classification respectively. In network attacks detection, the proposed approach performance is good and improved 7% accuracy compared to RNN, improved 3% accuracy compared to LSTM, and improved 3% accuracy compared to GRU. For network attacks classification, the proposed approach performed better than the recurrent deep learning models and improved 6% accuracy compared to RNN, and improved 3% accuracy for both LSTM and GRU. The misclassification rate of RNN, LSTM, GRU, and the proposed approach was 0.0936, 0.0693, 0.0742, and 0.0346 respectively. The proposed approach completely classified all the DoS and Fuzzing attack samples and 36 misclassifications for DDoS. However, still, the model has more misclassification for Normal, Port Scanning, OS Fingerprinting. Most importantly, the patterns of Normal and attacks of Port Scanning, OS Fingerprinting, and Fuzzing are not completely learnt by the model. Overall, the proposed approach performed better than RNN, LSTM, and GRU. Moreover, the proposed approach has tried to make the model interpretable and explainable. However, a detailed investigation and analysis are required to understand the robustness of the model.

The detailed performance of class-wise for the SDN-IoT dataset is shown in Table 4. DoS and DDoS showed 100% performances as macro precision, macro recall, and macro F1 score. Above 95% as macro precision, macro recall, and macro F1 score for Port Scanning, OS Fingerprinting, and Fuzzing respectively. However, 94% as macro precision, 85% as macro recall, and 89% as macro F1 score for Normal network connection samples of the SDN-IoT dataset. This indicates that further improvement is required to enhance the performances of Normal, Port Scanning, OS Fingerprinting, and Fuzzing network connection samples. This can be considered one of the significant directions toward future work.

6.1 Feature visualization using t-SNE

Recurrent deep learning models are well-known in recent days as they are showing better performances in various sequence and time series related problems. However, the hidden layers in these models are deep and there is no information on how the model extracts only the optimal features and it is considered to be a black box. Thus, interpretation and explainable models are considered to be very important in recurrent deep learning models. One such approach is feature visualization and it can be done using t-SNE. It is a nonlinear dimensionality reduction approach that maps the higher dimensional data into lower dimensional data by performing different transformations on different regions and finally, displays the data in a two-dimensional or three-dimensional map. t-SNE has several parameters and the most important parameters are n_components, perplexity, learning rate, iterations, and embedding initialization. In this work, n_components is set to 2, perplexity is set to 30.0, the learning rate is set to 200.0, and embedding initialization is set to PCA. No parameter tuning is done for these parameters and it is very important to achieve optimal performances. This work tries to see how the distribution of the features from RNN, LSTM, GRU, and the proposed approach using t-SNE. The figure shows that the penultimate feature visualization shown by the proposed approach is better compared to others such as RNN, LSTM, and GRU as the plot shows a kind of cluster formation for different classes in the case of both network attacks detection and network attacks classification. The t-SNE plots of RNN, LSTM, and GRU show that the data points are highly nonlinearly separable compared to the proposed method of t-SNE representation. This indicates that the concept of KPCA and meta-classifier in the proposed method facilitates extracting the important features that are efficient for network attacks detection and network attacks classification. Still, the proposed approach distribution is not optimal and further study is required to get a separate good distribution among network attacks. This has remained one of the significant directions toward future work.



(a) RNN

(b) LSTM

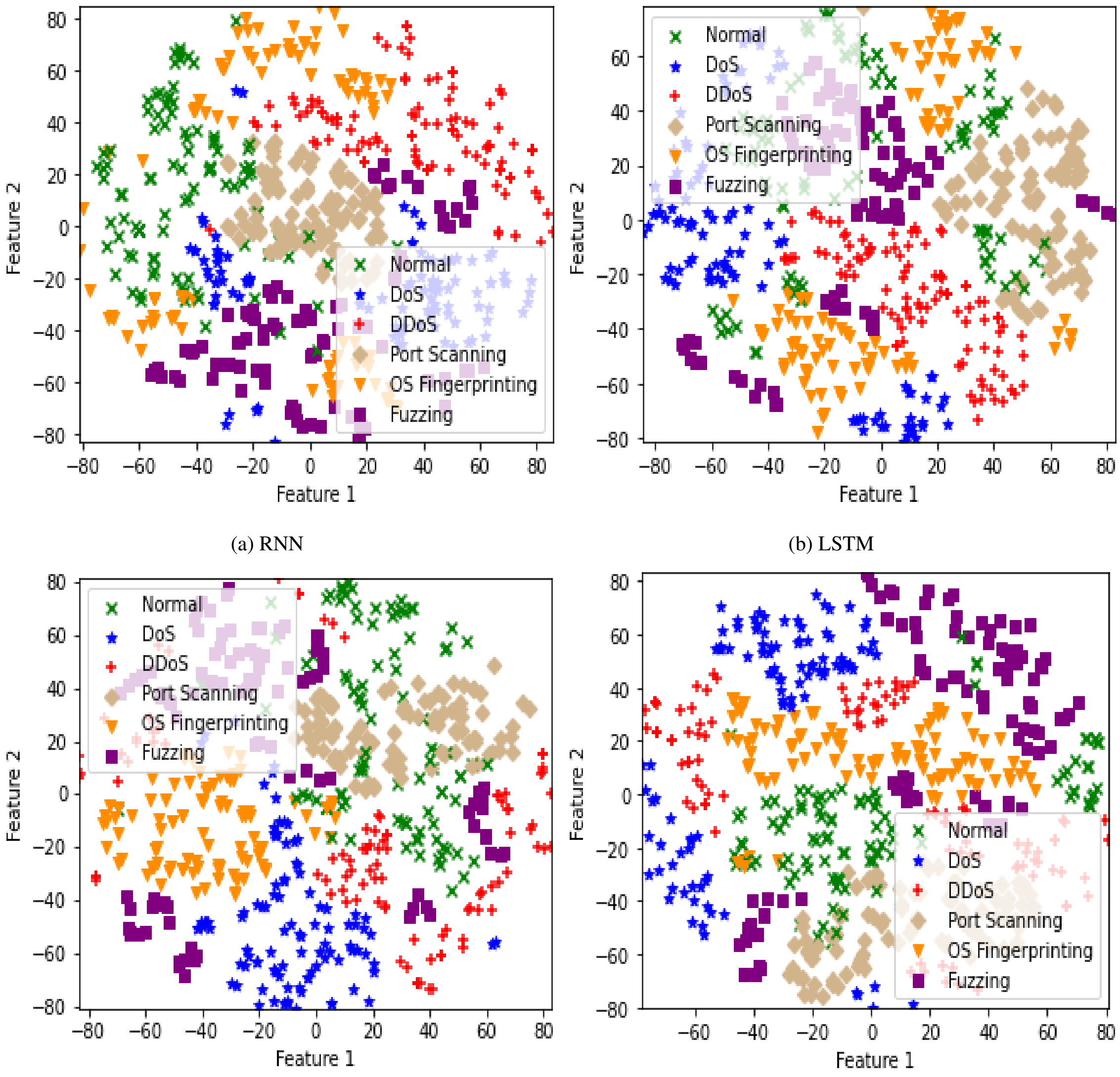
(c) GRU

(d) Proposed Approach

Fig. 7: Penultimate layer feature visualization using t-SNE.

6.2 Comparison of proposed method with deep learning and machine learning for network intrusion detection

In this work, the proposed approach is compared with the recurrent deep learning models such as RNN, LSTM, and GRU and other classical machine learning classifiers such as Naive Bayes, Logistic Regression, KNN, Decision Tree, and Random Forest. In logistic regression, the penalty parameter is set to L2 penalty. For KNN, the n_neighbors is set to 5, and leaf-size to 30. The decision tree important parameter criterion is



(a) RNN

(b) LSTM

(c) GRU

(d) Proposed Approach

Fig. 8: Penultimate layer feature visualization using t-SNE.

set to gini impurity. In Random Forest, n_estimators and criterion are set to 100 and gini impurity respectively. The detailed results of the proposed method, recurrent deep learning models such as RNN, LSTM, and GRU, and classical machine learning algorithms such as Naive Bayes, Logistic Regression, KNN, Decision Tree, and Random Forest are included in Table 2 for network attacks detection and Table 3 for network attacks classification. The table shows clearly that the proposed method performed better than the recurrent deep learning models and classical machine learning algorithms for both network attacks detection and network attacks classification in all the experiments of SDN-IoT dataset. Also, recurrent deep learning

Table 5: Results for intrusion detection using proposed approach on computer network-based datasets

Dataset	Accuracy	Precision	Recall	F1 score	Confusion Matrix
KDD-Cup-1999	0.99	0.97	0.99	0.98	TN=36096 FP=417 FN=2183 TP=146562
UNSW-NB15	0.99	0.99	0.99	0.99	TN=13886 FP=64 FN=95 TP=16829
WSN-DS	0.98	0.96	0.93	0.95	TN=88844 FP=527 FN=1149 TP=7827
CICIDS-2017	0.99	0.99	0.99	0.99	TN=22443 FP=269 FN=166 TP=12184

Table 6: Results for attack classification using proposed approachon computer network-based datasets

Dataset	Accuracy	Precision	Recall	F1 score
KDD-Cup-1999	0.89	0.42	0.50	0.35
UNSW-NB15	0.99	0.95	0.87	0.90
WSN-DS	0.98	0.91	0.84	0.87
CICIDS-2017	0.98	0.96	0.97	0.96

models performed better than the classical machine learning algorithms. The proposed method performance improved 3% accuracy for network attacks detection and 4% accuracy for network attacks classification compared to the recurrent deep learning models. Also, the proposed method has improved 9% accuracy for network attacks detection and 21% accuracy for network attacks classification compared to classical machine learning algorithms. The results clearly show that the proposed method performs better for multiclass classification i.e. network attacks classification compared to binary classification i.e. network attacks detection. This indicates that the hidden layer features of recurrent deep learning models are important for final classification and a detailed study and experiments are shown in this work.

6.3 Robustness and generalization of proposed method on network intrusion datasets

Robustness and generalization are very important in machine learning and deep learning. To show that the proposed method is robust and generalizable, the proposed method performance is evaluated on additional network intrusion detection datasets such as KDD-Cup-1999, UNSW-NB15, WSN-DS, and CICIDS-2017 for both network attacks detection and network attacks classification. The detailed performances for network attacks detection and network attacks classification are shown in Table 5 and Table 6 respectively. For network attacks detection, the proposed model shows 99% accuracy on KDD-Cup-1999, 99% accuracy on UNSW-NB15, 98% accuracy on WSN-DS, and 99% accuracy on CICIDS-2017 and for network attacks classification, the proposed model shows 89% accuracy on KDD-Cup-1999, 99% accuracy on UNSW-NB15, 98% accuracy on WSN-DS, and 98% accuracy on CICIDS-2017. The network attacks detection and network attacks classification results on additional 4 well-known benchmark network intrusion detection show that the proposed method has shown better performance similar to the SDN-IoT dataset. Thus the proposed model is to be considered robust and generalizable. However, the proposed model may not be robust and generalizable on an adversarial modified dataset or adversarial environment and the case study is not shown in this work. In addition, the classes in these 4 datasets don't have an equal number of data samples in training, validation, and testing. This is one of the main reasons the macro precision, macro recall, and macro F1 Score is less in some of the test cases. This indicates that the proposed method is sensitive to imbalanced data and some further improvement has to be done to handle the imbalanced classes in the network connection dataset. This can be another significant direction of research. This can be done by proposing a cost-sensitive approach such as during backpropagation, larger weights can be initialized to the classes that have fewer network connections and fewer weights to the classes that contain more network connections.

The proposed intrusion detection system based on feature fusion of all the hidden layers in the recurrent deep learning model showed better performances compared to the penultimate layer features of recurrent deep learning models and models based on classical machine learning. Most importantly, the proposed method has shown similar performances on datasets from the different network environments. It indicates that the proposed approach is platform-independent and the approach is robust and generalizable in any network configuration environment. A recent literature survey shows that the deep learning models are not robust in an adversarial environment [2]. However, the performance of the proposed method is not evaluated in an adversarial environment and this is considered to be an important case study. This type of study can be considered one of the significant directions for future work. Dimensionality reduction of features extracted from hidden layers of recurrent model and feature fusion employed in this work is not optimal. Instead of selecting all the features, kernel-based approaches can be employed to select the optimal features. This type of approach can further enhance the network attacks detection and classification performances and reduces the learnable parameters of the model. Further, a detailed investigation and analysis of the proposed intrusion detection system in an adversarial environment are essential as recurrent deep learning-based models are not robust in an adversarial environment. This type of research study is important to know that the system is robust against adversarial attacks. Though the ensemble meta-classifier has shown better performances with the features of recurrent deep learning-based models, no detailed experiments and investigation is shown in this work. Ensemble meta-classifiers achieve large-scale learning and further work is required to understand the importance of ensemble meta-classifiers for classification instead of a single fully connected layer.

7 Conclusion and Future works

Network intrusion detection is an essential and important component of a cyber-physical systems environment to secure the system from attackers. In this work, an end-to-end model is proposed for cyber-physical systems network intrusion detection. The model employs random forest and support vector machine in the first level on features extracted from deep learning and Kernel principal component analysis for prediction and followed by logistic regression in the second level for network attack detection and network attack classification. The proposed model performed better than the existing methods in all experiments on more than one network dataset. This indicates that the proposed model is robust and generalizable for new network traffic datasets. The proposed method can be used in real-time to effectively monitor the network traffic to proactively alert possible attacks and classify them into their attack categories.

Funding

Not applicable.

Conflicts of interest/Competing interests

The authors declare no conflict of interest.

Availability of data and material

Data used in this study are available from the first author upon request.

Code availability

The Programming codes implemented in this study are available from the first author upon request.

Ethical Approval

None

References

1. Dey, S. K., Uddin, M. R., & Rahman, M. M. (2020). Performance analysis of SDN-based intrusion detection model with feature selection approach. In Proceedings of international joint conference on computational intelligence (pp. 483-494). Springer, Singapore.
2. ElSayed, M. S., Le-Khac, N. A., Albahar, M. A., & Jurcut, A. (2021). A novel hybrid model for intrusion detection systems in SDNs based on CNN and a new regularization technique. Journal of Network and Computer Applications, 191, 103160.
3. Tang, T. A., McLernon, D., Mhamdi, L., Zaidi, S. A. R., & Ghogho, M. (2019). Intrusion detection in sdn-based networks: Deep recurrent neural network approach. In Deep Learning Applications for Cyber Security (pp. 175-195). Springer, Cham.
4. Hannache, O., & Batouche, M. C. (2020). Neural network-based approach for detection and mitigation of DDoS attacks in SDN environments. International Journal of Information Security and Privacy (IJISP), 14(3), 50-71.
5. Thakur, S., Chakraborty, A., De, R., Kumar, N., & Sarkar, R. (2021). Intrusion detection in cyber-physical systems using a generic and domain specific deep autoencoder model. Computers & Electrical Engineering, 91, 107044.
6. Ajaeiya, G. A., Adalian, N., Elhajj, I. H., Kayssi, A., & Chehab, A. (2017, July). Flow-based intrusion detection system for SDN. In 2017 IEEE Symposium on Computers and Communications (ISCC) (pp. 787-793). IEEE.
7. Hadem, P., Saikia, D. K., & Moulik, S. (2021). An SDN-based Intrusion Detection System using SVM with Selective Logging for IP Traceback. Computer Networks, 191, 108015.
8. Ye, J., Cheng, X., Zhu, J., Feng, L., & Song, L. (2018). A DDoS attack detection method based on SVM in software defined network. Security and Communication Networks, 2018.
9. Latah, M., & Toker, L. (2018). Towards an efficient anomaly-based intrusion detection for software-defined networks. IET networks, 7(6), 453-459.
10. Dey, S. K., Uddin, M. R., & Rahman, M. M. (2020). Performance analysis of SDN-based intrusion detection model with feature selection approach. In Proceedings of international joint conference on computational intelligence (pp. 483-494). Springer, Singapore.
11. Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A. R., & Ghogho, M. (2016, October). Deep learning approach for network intrusion detection in software defined networking. In 2016 international conference on wireless networks and mobile communications (WINCOM) (pp. 258-263). IEEE.
12. Mitchell, R., & Chen, R. (2014). Behavior rule specification-based intrusion detection for safety critical medical cyber physical systems. IEEE Transactions on Dependable and Secure Computing, 12(1), 16-30.
13. Han, S., Xie, M., Chen, H. H., & Ling, Y. (2014). Intrusion detection in cyber-physical systems: Techniques and challenges. IEEE systems journal, 8(4), 1052-1062.
14. Belenko, V., Chernenko, V., Kalinin, M., & Krundyshev, V. (2018, September). Evaluation of GAN applicability for intrusion detection in self-organizing networks of cyber physical systems. In 2018 International Russian Automation Conference (RusAutoCon) (pp. 1-7). IEEE.
15. Sadreazami, H., Mohammadi, A., Asif, A., & Plataniotis, K. N. (2017). Distributed-graph-based statistical approach for intrusion detection in cyber-physical systems. IEEE Transactions on Signal and Information Processing over Networks, 4(1), 137-147.
16. Althobaiti, M. M., Kumar, K. P. M., Gupta, D., Kumar, S., & Mansour, R. F. (2021). An intelligent cognitive computing-based intrusion detection for industrial cyber-physical systems. Measurement, 186, 110145.
17. Dutta, A. K., Negi, R., & Shukla, S. K. (2021, July). Robust multivariate anomaly-based intrusion detection system for cyber-physical systems. In International Symposium on Cyber Security Cryptography and Machine Learning (pp. 86-93). Springer, Cham.
18. Formby, D., Srinivasan, P., Leonard, A. M., Rogers, J. D., & Beyah, R. A. (2016, February). Who's in Control of Your Control System? Device Fingerprinting for Cyber-Physical Systems. In NDSS.

19. Junejo, K. N., & Goh, J. (2016, May). Behaviour-based attack detection and classification in cyber physical systems using machine learning. In Proceedings of the 2nd ACM international workshop on cyber-physical system security (pp. 34-43).
20. Haller, P., & Genge, B. (2017). Using sensitivity analysis and cross-association for the design of intrusion detection systems in industrial cyber-physical systems. IEEE Access, 5, 9336-9347.
21. Mitchell, R., & Chen, R. (2011, March). A hierarchical performance model for intrusion detection in cyber-physical systems. In 2011 IEEE Wireless Communications and Networking Conference (pp. 2095-2100). IEEE.
22. Alzahrani, A. O., & Alenazi, M. J. (2021). Designing a Network Intrusion Detection System Based on Machine Learning for Software Defined Networks. Future Internet, 13(5), 111.
23. Sarica, A. K., & Angin, P. (2020, November). A Novel SDN Dataset for Intrusion Detection in IoT Networks. In 2020 16th International Conference on Network and Service Management (CNSM) (pp. 1-5). IEEE.
24. Nedeljkovic, D., & Jakovljevic, Z. (2022). CNN based method for the development of cyber-attacks detection algorithms in industrial control systems. Computers & Security, 114, 102585.
25. Sheng, C., Yao, Y., Fu, Q., & Yang, W. (2021). A cyber-physical model for SCADA system and its intrusion detection. Computer Networks, 185, 107677.

Vinayakumar Ravi

Vinayakumar Ravi is an Assistant Research Professor at Center for Artificial Intelligence, Prince Mohammad Bin Fahd University, Khobar, Saudi Arabia. His current research interests include applications of data mining, Artificial Intelligence, machine learning (including deep learning) for biomedical informatics, Cyber Security, image processing, and natural language processing. More details available at `https://vinayakumarr.github.io/`

Rajasekhar Chaganti

Rajasekhar Chaganti works as a security engineer in security team at Expedia group Inc and he is also involved in security research and working towards PhD degree from University of Texas San Antonio. His research interests are in machine learning/AI for security applications, network security, threat detection in IoT, cloud and blockchain environments and social engineering scams.

Mamoun Alazab

Mamoun Alazab is an Associate Professor at the College of Engineering, IT and Environment at Charles Darwin University, Australia. He received his PhD degree in Computer Science from the Federation University of Australia, School of Science, Information Technology and Engineering. He is a cyber security researcher and practitioner with industry and academic experience.