A Spectrogram Image-Based Network Anomaly Detection System Using Deep Convolutional Neural Network

Khan, A. S., Ahmad, Z., Abdullah, J. & Ahmad, F.

DOI 10.1109/ACCESS.2021.3088149
ESSN 2169-3536

Publisher: IEEE

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 15, 2021, accepted May 31, 2021, date of publication June 11, 2021, date of current version June 23, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3088149

A Spectrogram Image-Based Network Anomaly Detection System Using Deep Convolutional Neural Network

ADNAN SHAHID KHAN[1], (Senior Member, IEEE), ZEESHAN AHMAD[1,2], (Member, IEEE), JOHARI ABDULLAH[1], AND FARHAN AHMAD[3], (Member, IEEE)

[1]Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak, Sarawak 94300, Malaysia
[2]Department of Electrical Engineering, College of Engineering, King Khalid University, Abha 62529, Saudi Arabia
[3]Institute for Future Transport and Cities, Coventry University, Coventry CV1 5FB, U.K.

Corresponding authors: Adnan Shahid Khan (skadnan@unimas.my) and Zeeshan Ahmad (zayshan@kku.edu.sa)

This work was supported by the Research, Innovation and Enterprise Centre (RIEC), Universiti Malaysia Sarawak, Sarawak, Malaysia, under Grant F08/PGRG/2058/2021.

ABSTRACT The dynamics of computer networks have changed rapidly over the past few years due to a tremendous increase in the volume of the connected devices and the corresponding applications. This growth in the network's size and our dependence on it for all aspects of our life have therefore resulted in the generation of many attacks on the network by malicious parties that are either novel or the mutations of the older attacks. These attacks pose many challenges for network security personnel to protect the computer and network nodes and corresponding data from possible intrusions. A network intrusion detection system (NIDS) can act as one of the efficient security solutions by constantly monitoring the network traffic to secure the entry points of a network. Despite enormous efforts by researchers, NIDS still suffers from a high false alarm rate (FAR) in detecting novel attacks. In this paper, we propose a novel NIDS framework based on a deep convolution neural network that utilizes network spectrogram images generated using the short-time Fourier transform. To test the efficiency of our proposed solution, we evaluated it using the CIC-IDS2017 dataset. The experimental results have shown about $2.5\% - 4\%$ improvement in accurately detecting intrusions compared to other deep learning (DL) algorithms while at the same time reducing the FAR by $4.3\% - 6.7\%$ considering binary classification scenario. We also observed its efficiency for a 7-class classification scenario by achieving almost 98.75% accuracy with $0.56\% - 3.72\%$ improvement compared to other DL methodologies.

INDEX TERMS Convolutional neural network, deep learning, network intrusion detection system, spectrogram.

I. INTRODUCTION

Rapid advancements and growth in Internet and communication technologies have resulted in an enormous expansion in terms of the network size and the associated applications [1]. As a result, a very huge amount of data is being generated and shared across different network nodes that needed to be secured from possible intrusions in the network [2]. To provide the required security to the network different tools like firewall, authentication schemes, encryption mechanisms, and anti-viruses, etc. are used [3]. All these mechanisms are called the first line of defense to provide the needed security to the network. But these tools are not enough to secure the network nodes and data due to the generation of a large number of new attacks either through the mutation of an old attack or a novel attack. To provide extra security, an intrusion detection system (IDS) can be employed in the network which will act as the second line of defense to improve its security [4].

An IDS can be either host-based or network-based depending upon its deployment strategy and signature-based or anomaly detection-based depending on its detection strategy [5]. In this study, our emphasis is on the network-based intrusion detection system (NIDS) using the anomaly detection-based detection mechanism. A NIDS is an anomaly detection mechanism that can be deployed at the entry

The associate editor coordinating the review of this manuscript and approving it for publication was Tyson Brooks.

point e.g., edge router of a network to allow the constant monitoring of the network traffic for any malicious activity. A network administrator is notified once it detects suspicious traffic behavior. Studies have shown that the current NIDSs have shown its inefficiency in detecting novel and zero-day attacks, which have increased the false alarm rates (FAR) [6], which is still the research gap as the network dynamics have changed very rapidly recently due to technology development. So, there is a need for an effective NIDS which can protect a network from all the new and old attacks with low FAR.

Recently, the researchers have explored and proposed the efficient NIDS based on machine learning (ML) and deep learning (DL) methodologies. Both ML and DL algorithms are quite powerful in learning from the network flows and then predicting the benign and anomaly traffic based on learned patterns [7]. ML requires feature engineering to learn patterns while DL is good at learning from raw data automatically by deploying deep architecture. Recent advancements in Graphical Processing Units (GPUs) design have opened ways to use DL methodologies in different fields including network security [8]. The possibility of using DL methods for NIDS is still in the early research phase and there is still an enormous room to explore this technology within NIDS to efficiently detect intruders within the network. Researchers are exploring different DL based methodologies to improve on the detection accuracy while keeping the FAR as lower as possible to make it more effective.

Among the DL algorithms, Convolutional neural networks (CNNs) have performed excellently in the field of computer vision for processing images [9] and spectrograms of the audio [10]. It is more suitable for the data stored in arrays. A CNN can be used to process 1D, 2D, or 3D (1, 2, or 3-dimensional) arrays of data depending upon its application and nature of data. In the field of IDS, the researchers have used CNN for processing 1D and 2D arrays. The 1D data needs to be converted into a 2D data array before being processed by the CNN for learning useful patterns to perform prediction tasks efficiently [11]–[14].

Based on our literature review on NIDS [3], we observed that although researchers have used the images for proposing DL-based NIDS. But we could not find any noticeable literature that has explored the possibility of using spectrogram images together with the DL methods for the IDS domain. Spectrogram images are the visual representation of a signal providing the frequency and energy details in the form of a color map [15]. This motivated us to perform this research work presented in this paper by proposing a NIDS using the idea of transforming data into spectrogram images before training the CNN-based NIDS model [16]. Results have shown the efficiency of our proposed methodology to prove the effectiveness of the usage of the spectrogram images in the IDS domain. This study is an effort in this direction, and we accordingly propose a methodology of using spectrograms with the CNN for the IDS domain. The main motivation of this work is to study the effectiveness of frequency information of the traffic to check if the frequency details can be helpful in the prediction of network traffic data. The results have shown the effectiveness of our methodology to detect anomalies more efficiently by reducing the FAR.

This paper aims at implementing the DL-based methodology for proposing an efficient NIDS solution. The main contributions of this paper are 3-fold.

(i) We extensively discuss the state-of-the-art works on NIDS that are being proposed using DL methodologies.
(ii) We propose a novel spectrogram-based NIDS using deep CNN (SDCNN). To the best of our knowledge, this is the first such study in the domain of IDS, that utilizes the network flows as spectrogram images and then using these spectrograms to train and test the CNN model.
(iii) To test the effectiveness and efficiency of the SDCNN against different DL models on the CIC-IDS2017 dataset. Also, we have directly compared our proposed methodology against some of the recent state-of-the-art works on the DL-based NIDS to test the efficiency of SDCNN.

The rest of the paper is organized as follows: Section II describes the related literature on ML and DL-based NIDS solutions. Section III provides the details about the proposed solution and methodology adopted in this study. Section IV extensively provides details on the dataset, experimental setup, results, and its discussion for both binary classification and multiclass classification. Finally, Section V concludes this research article. The abbreviations used in this article are summarized in Table 13.

II. RELATED WORKS

Over the last decade, researchers have extensively incorporated artificial intelligence in the NIDS to make it more effective in terms of learning efficient features and improving the predictions while at the same time minimizing the FAR. The recent trends on the NIDS showed that most of the proposed NIDS solutions were based on the ML algorithms till 2018 [3]. From the last 2-3 years, DL has become the first choice of researchers for designing efficient intrusion detection mechanisms. Since DL algorithms require a huge amount of data to learn from, utilizing its deep architecture with multiple hidden layers to explore complex patterns and features for efficient predictions. So, its complexity is very high with a high requirement for extensive computation. This fast computational requirement is solved with the advancements in GPU development, which have motivated researchers to utilize DL algorithms in designing efficient solutions in almost all the fields including the IDS [17].

Different ML algorithms are explored by researchers for IDS improvement as summarized in Table 1. For instance, Yao *et al.* [18] proposed a multilevel semi-supervised ML framework for IDS by utilizing the clustering concept along with the random forest (RF). The model performed well in detecting the attack instances even with lower records due to

TABLE 1. Summary of the related works.

Study	ML	DL	Methodology	Dataset
Yao *et al.* [18]	✓		Semi-supervised clustering concept along with the Random Forest	KDD Cup'99
Ali *et al.* [19]	✓		Particle Swarm optimization with a Fast-Learning network	KDD Cup'99
Shen *et al.* [20]	✓		Ensemble method considering many Extreme Learning machines.	KDD Cup'99, NSL KDD, Kyoto 2006+
Shone *et al.* [21]	✓	✓	Autoencoder (AE) and Random Forest	KDD Cup'99, NSL KDD
Yan *et al.* [22]	✓	✓	Stacked sparse AE and support vector machine (SVM)	NSL KDD
Marir *et al.* [23]	✓	✓	Deep Belief Network (DBN) and SVM in an ensemble fashion	NSL KDD, UNSW-NB15, ICIDS2017
Xu *et al.* [24]		✓	Gated Recurrent Unit (GRU) along with the multilayer perceptron and softmax layer for classification	KDD Cup'99, NSL KDD
Yang *et al.* [25]		✓	Supervised adversarial variational AE using regularization and Deep Neural Network (DNN)	NSL KDD, UNSW-NB15
Wei *et al.* [26]		✓	DBN combined with optimization algorithms such as particle swarm, fish swarm, and genetic algorithms.	NSL KDD
Xiao *et al.* [27]	✓	✓	Principal Component Analysis and AE for feature extraction followed by CNN	KDD Cup'99
Zhang *et al.* [28]	✓	✓	CNN and gcForest by proposing a P-Zigzag algorithm for 2D greyscale image conversion.	UNSW-NB15, CICIDS2017
Jiang *et al.* [29]		✓	CNN together with bidirectional Long short-term memory (LSTM)	KDD Cup'99, UNSW-NB15
Andresini *et al.* [30]		✓	Two AEs with the 1D CNN	NSL KDD, UNSW-NB15, ICIDS2017
Riyaz *et al.* [31]		✓	Feature selection using conditional random field and linear correlation coefficient algorithms followed by CNN	KDD Cup'99
Ganapathi *et al.* [32]	✓		Rule-based attribute selection and classification based SVM	KDD Cup'99
Nancy *et al.* [33]	✓	✓	Decision tree with the CNN	KDD Cup'99
Mirsky *et al.* [36]		✓	Ensemble of Autoencoders	Emerging Threats Rules
Bovenzi *et al.* [37]	✓	✓	Deep autoencoder (DAE) followed by soft-output classifier	Bot-IoT
This Study		✓	**Spectrogram Image based Deep CNN**	**CIC-IDS2017**

the multilevel detection process. Similarly, another notable work using ML is proposed by Ali *et al.* [19], who combined the particle swarm optimization with a fast learning network (PSO-FLN) to propose an efficient IDS solution. However, their model did not show promising results in predicting the minority class labels. Similarly, Shen *et al.* [20] used the ensemble method to propose their IDS solution by considering many extreme learning machines. They used the BAT optimization algorithm during the ensemble pruning stage. The detection accuracy exhibited by their proposed solution for the User to Root (U2R) attack was also on the lower side.

Researchers have also proposed hybrid solutions by utilizing both the ML and DL algorithms. The DL algorithms are mostly used in such cases for the feature reductions followed by the ML classifier for prediction. For instance, Shone *et al.* [21] proposed a hybrid solution using autoencoder (AE) and RF algorithms by considering the encoder part of the AE only in a nonsymmetric manner. The main limitation of their methodology was the model's lower detection accuracy in detecting minority attack classes. Another hybrid approach is proposed by Yan and Han [22] using stacked sparse AE and support vector machine (SVM). Their model also did not perform well for minority attacks. Marir *et al.* [23] also used the distributed hybrid approach by using deep belief network (DBN) and SVM in an ensemble fashion to propose a highly efficient IDS model by adopting a voting approach. But the high efficiency for their model is obtained at the cost of high model complexity, which increased the training model training time.

NIDS are also being proposed using the different DL algorithms such as AE, DBN, RNN (Recurrent Neural Network) and CNN, etc. Xu *et al.* [24] proposed an efficient IDS model using RNN with Gated Recurrent Unit (GRU) as the main memory unit along with the multilayer perceptron and softmax layer for classification. For their proposed solution, lower detection rates for Root to Local (R2L) and U2R

attack classes are recorded. Also, Yang *et al.* [25] proposed an IDS solution using supervised adversarial variational AE using regularization and Deep Neural Network (DNN) (SAVER-DNN), which showed improvement in detecting even lower frequency and new attacks. Similarly, Wei *et al.* [26] used DBN to propose their highly complex IDS framework by combining it with different optimization algorithms such as particle swarm, fish swarm, and genetic algorithms.

CNN is also being studied by the researcher in the domain of IDS. For instance, Xiao *et al.* [27] proposed their solution by combining principal component analysis and AE for feature extraction followed by CNN for classification and prediction tasks. Their proposed solution showed incapability in detecting the minority attack classes. Another notable work using CNN is by Zhang *et al.* [28] who proposed a very complex IDS solution based using the CNN and gcForest by proposing a P-Zigzag algorithm for 2D greyscale image conversion. Their model performed well in detecting the anomalies with low FAR. Similarly, Jiang *et al.* [29] used the CNN combined with bidirectional Long short-term memory (LSTM) to propose their complex IDS model. They use SMOTE to improve the ratio of minority class samples. Also, Andresini *et al.* [30] combined two AEs with the 1D CNN to proposed an effective multistage IDS solution.

Similarly, Riyaz and Ganapathy [31] proposed a CNN-based IDS by first performing the feature selection using conditional random field and linear correlation coefficient algorithms followed by classifying them using CNN. They evaluated their solution on an older dataset KDD cup'99 to show the effectiveness of their proposed methodology. The results showed the improvement in detecting the minority class attacks, with the reduced complexity due to feature selection. Also, Ganapathy *et al.* [32] proposed their IDS detection methodology based on intelligent rule-based attribute selection and classification based on a multiclass support vector machine. Another promising work to propose an IDS for the wireless sensor networks is done by Nancy *et al.* [33] that proposed a dynamic recursive feature selection methodology followed by the classification by combining the decision tree with the CNN for efficient intrusion detection. Their model exhibited a lower detection accuracy for the U2R and R2L attacks.

In [34], Wang *et al*. make use of the representation learning approach for classifying the malware traffic using CNN. However, their proposed methodology is tested on the known attacks using only temporal features without tuning the CNN parameters. Similarly, Aceto *et al.* [35], utilizes the DL to propose a mobile traffic classifier that can work with encrypted traffic. Their study considered different DL architectures such as stacked AE, CNN, and LSTM, etc., to study the model's performance for encrypted traffic data to discuss the input size to be fed into DL classifier, the traffic classifier object adopted.

In [36], Mirsky *et al.* make use of an ensemble of autoencoders to propose an efficient plug-and-play and lightweight online NIDS solution (Kitsune), that can learn in an unsupervised manner. The model is evaluated on the Raspberry PI to show its ability to run on the router. Similarly, Bovenzi *et al.* [37] proposed an efficient two-stage hybrid NIDS model using multimodal Deep autoencoder (DAE) as the first stage followed by soft output classifier. They evaluated the model on the Bot-IoT dataset to show its effectiveness for the IoT network. Stage 1 performed the binary classification followed by the multiclass classification. The use of DAE helped to reduce the dimensionality to make it a lightweight approach for IoT networks.

To sum the discussion up, although many of the proposed solutions were very commanding in detecting most of the considered attacks. But at the same time, these models showed limitations in detecting a few of the considered attacks. Most of the proposed methodologies struggled to detect the attacks with the lower training samples. It is also observed that the complexity is also related to detection accuracy. Few studies also showed that the detection accuracy is although improved at the cost of increased model complexity in terms of training time and the resource consumed. Also, DL research is still in its early phase for the IDS domain with plenty of research room available, that needed to be explored by researchers to improve IDS efficiency in detecting both new and old attacks efficiently and correctly. To this end, we explore the use of a CNN to propose our NIDS multistage solution. We make use of a novel idea of using the spectrogram images for the IDS domain. We propose a CNN-based NIDS framework by improving its efficiency utilizing spectrogram images.

The following section details the important concepts and methodology followed to propose our SDCNN model.

III. PROPOSED METHODOLOGY

In this section, we describe the important concepts followed by the details about the methodology adopted for proposing the SDCNN framework.

A. COVOLUTIONAL NEURAL NETWORK

CNN is one of the popular feedforward DL techniques that have performed well in processing the data arranged as arrays or grids such as images. A simple CNN consists of an input layer, followed by a stack of a convolutional layer with a certain activation function (CL) and a pooling layer (PL), the fully connected layer, and a final classification activation layer as shown in Figure 1. The different layers of the CL and PL stack perform the feature extraction tasks. While classification task is performed by a combination of the fully connected layers and a classification output layer. Studies have shown that CNN learns in a supervised manner for the IDS to efficiently perform classification and prediction. The different layers of CNN are explained below.

1) CONVOLUTIONAL LAYER (CL)

A CL incorporates a convolutional operation and constitutes the core of the CNN. This layer works by operating a series of convolutional kernels (filters) for learning different features

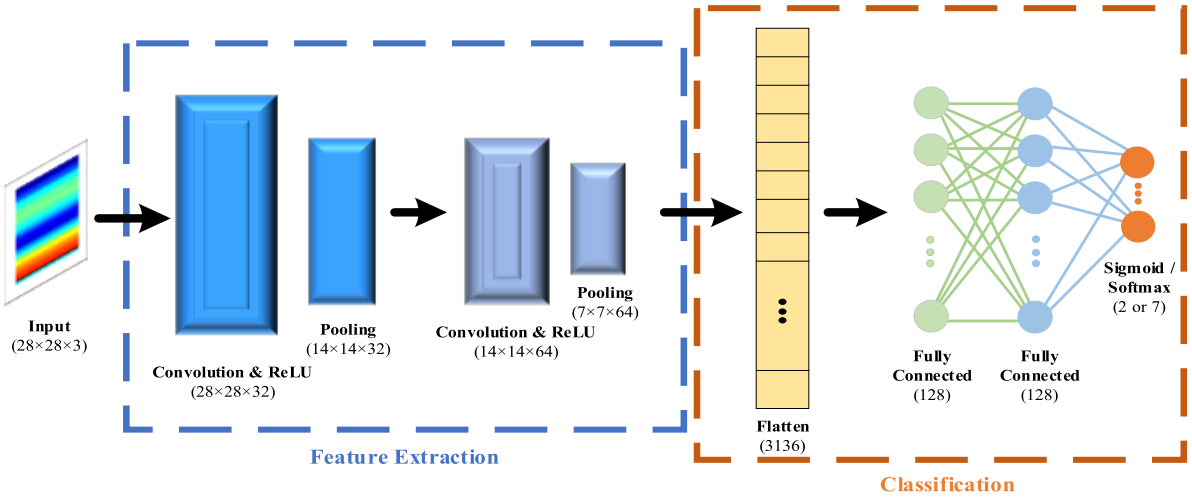

FIGURE 1. CNN architecture used to propose SDCNN.

of the input image and produces the feature maps. The convolution operation is accomplished by using the dot product between the image and a filter. The operations performed in this layer can be shown mathematically as [38],

$$f_m = b_m + \sum_n (X_n \otimes K_{nm}) \tag{1}$$

where f_m is the m-th feature map, b_m is the m-th bias function, and X_n is the n-th input. K_{nm} is convolutional kernel connecting n-th input with m-th output. Symbol $\otimes$ represents the convolution operation. The feature maps are then passed through an activation function (*ReLU* in this study) to generate layer output y_m.

$$y_m = max\,(0, f_m) \tag{2}$$

Activation Functions

The activation functions are the mathematical functions used in the neural networks to control the output. It can be linear or nonlinear depending upon the type of application used. In this study, the used activation functions are *ReLU*, *sigmoid,* and *softmax* which are calculated using the mathematical formulas given as,

$$ReLU\,(x) = \max\,(0, x) \tag{3}$$

$$sigmoid\,(x) = \frac{1}{1 + e^{-x}} \tag{4}$$

$$softmax\,(x_k) = \frac{e^{x_k}}{\sum_{i=1}^{n} e^{x_i}} \tag{5}$$

where x represents the input provided to one of the activation functions given in equation 3, 4, or 5.

2) POOLING LAYER (PL)

The PL normally follows the CL and it performs the task of reducing the size of feature maps (output of CL) by employing non-linear down-sampling such as maximum over nonoverlapping subsets of the feature map. This eventually improves memory usage by reducing image size and the number of parameters that avoid overfitting. Additionally, the regularization technique such as dropout can be added to avoid overfitting, which eventually results in improving the model accuracy.

3) FULLY CONNECTED LAYERS (FCLs)

FCLs normally follow the different layers of CL and PL stack, first convert the 2D image matrix to 1D and then pass it through the network of dense layers for preparing it for the classification. The last layer of the CNN is generally a classification layer that employs a certain activation function. This activation function depends upon the type of classification performed. For this study, the *sigmoid* and the *softmax* activation functions are used for binary and multiclass classification, respectively.

The details about the CNN used in this study are shown in Figure 1. As shown, 2 layers of the stacked CL and PL are used. The input to the CNN is a spectrogram image of a 28 × 28 matrix with 3 channels. The feature extraction block transforms the matrix into a 7 × 7 × 64 matrix which is then input for the classification block. The classification block first flattens the matrix into 3136 and passes it from the two fully connected layers with 128 neurons each. Finally, for the classification, either the *sigmoid* or *softmax* activation layer is used for binary and multiclass classification scenarios.

B. SPECTROGRAM

A spectrogram represents the image that provides the visual details of a signal. It provides information about the frequency and energy within the signal by representing the frequencies across the vertical axis and the energy by varying the color maps. Spectrograms are quite effectively used in different domains like speech analysis [15] and the medical field for ECG analysis [39]. To generate spectrogram images from the feature data available, we used the Short-Time Fourier Transform (STFT) technique for performing the time-frequency analysis. The STFT $STFT\,\{x\,[n]\}$ of a discrete-time

signal $x[n]$ is mathematically given as [39], [40],

$$STFT\{x[n]\} = X(m,\omega) = \sum_{n=-\infty}^{\infty} x[n]\, w_{hn}[n-m]\, e^{-j\omega n} \tag{6}$$

where, $x[n] = \{f_1, f_2, \cdots, f_{78}\}$ is the input vector with features f in a dataset for each record. Also, m is the reference time while ω is the angular frequency. The $w_{hn}[n]$ is the Hanning window function defined as,

$$w_{hn}[n] = \frac{1}{2}\left(1 - cos\left(2\pi \frac{n}{N}\right)\right), \quad 0 \le n \le N-1 \tag{7}$$

where N indicates the time observation length. The spectrogram is then calculated as the

$$Spectrogram(m,\omega) = |STFT\{x[n]\}|^2 = |X(m,\omega)|^2 \tag{8}$$

The use of converting network flows into 2D images is studied by many researchers for the ML and DL-based NIDS [27]–[29]. The images generated in all those studies are generated directly by converting a 1D flow into 2D. Our work is different from all those due to a different approach adopted in generating the images. For this study, we generated the spectrogram images, that are made from the frequency and energy images. Our main objective in this work is to find out the importance of using the frequency information for the NIDS. We want to study if the frequency details can improve the performance of the NIDS. To this end, we use the spectrogram images, that are obtained from the network flows using the STFT analysis.

C. METHODOLOGY

To protect the network from intrusions and anomalies, we proposed an efficient multistage NIDS by proposing a novel idea of utilizing the spectrogram images for the Deep convolutional neural network (SDCNN). The suitable place for the deployment of SDCNN is the entry points of the network to immediately start capturing and analyzing the network flows. The different stages of our proposed SDCNN are shown in Figure 2 and are,

1. Data Capturing and Preparation stage
2. Spectrogram images generation and storage stage
3. Deep CNN model training stage
4. Model testing stage

1) DATA CAPTURING AND PREPARATION STAGE

This is the first stage of our SDCNN framework that is responsible for acquiring the network flows, storing in the database and then preprocess it to bring it in the format needed for processing by the CNN model. The different steps performed are,

Step-1: The network packets are first captured using packet sniffing tools (such as TCPdump, Ethereal, etc.). Then the useful and meaningful features are extracted and then are stored in a dataset [41].

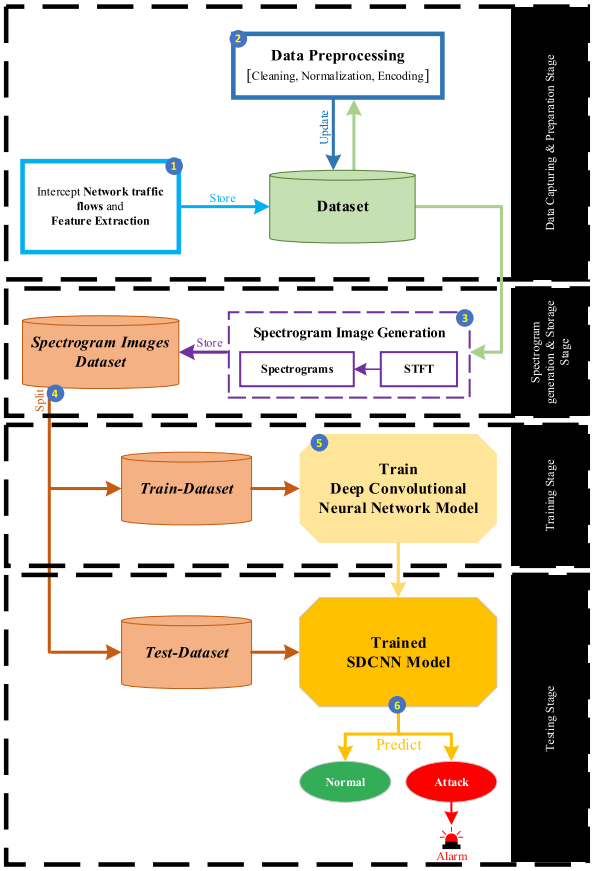

FIGURE 2. Proposed SDCNN framework.

Step-2: The stored records within the dataset are then first cleaned to remove the infinite and empty entries. After that the dataset is normalized and encoded (one hot encoding for categorical feature) to bring it in a format that can be used for the deep learning stage.

In this study, our processing starts from the step-2 as we are using an openly available dataset CICIDS 2017. Step-1 will be needed once the SDCNN is deployed at the edge router.

2) SPECTROGRAM IMAGES GENERATION AND STORAGE STAGE

This main task performed in this stage is the generation and storage of the spectrogram images in another dataset. The steps performed are,

Step-3: The updated cleaned, normalized, and encoded dataset is then used for spectrogram generation by first calculating the STFT, followed by spectrogram images using eq (6) – eq (8). The generated spectrograms are then saved in another new dataset called *Spectrogram Images Dataset*.

Step-4: This *Spectrogram Images Dataset* is then divided into a train-dataset and a test-dataset with a split ratio of 75% and 25% respectively for training and testing purposes.

3) DEEP CNN MODEL TRAINING STAGE

Step-5: The main task performed in this stage is to train the deep CNN model detailed in Figure 1, using the train-dataset. For this study, the model will be trained for 100 epochs and the best model is saved as ''Trained SDCNN Model'' that will be used for the testing stage.

4) MODEL TESTING STAGE

Step-6: This stage will be the prediction stage which will test the proposed SDCNN model using the test-dataset. If a Normal flow is detected, it will be passed as it is without taking any action. If an attack is detected, an Alarm signal will be generated to notify the administrator to take further actions.

Based on the model's performance in detecting and classifying normal and attack traffic, we calculated the effectiveness of our proposed framework. The ideal place to deploy our proposed solution is the edge router, which is the entry point of the traffic to the network. The model should be trained regularly with the updated database of unknown attacks to have efficient network protection. Once any unknown attack is detected, it will be stored in the database. When enough records are gathered, the SDCNN model can be trained during the offline time and the previous model can be replaced with the newly trained model to have better detection accuracy and performance.

D. MODEL CONFIGURATION

The detailed configuration of our proposed SDCNN model is provided in Figure 3 in terms of the input and output of each layer to predict the network record. As observed, the first stage involves the generation of the spectrogram images and a network flow record containing all the 78 features except the Label feature is input to this block. The spectrogram image based on STFT is generated as a 28×28 matrix with 3 channels that is then input to the first layer of DCNN. The DCNN contains the two layers of the CL and PL stack. We used the Batch normalization technique along with the dropout in the range of [0-1] between different layers to speed up the model and avoid overfitting during the training [30], [42]. The output of the feature extraction block (2-layer stack of CL and PL along with the batch normalization and dropout) results in a matrix of dimension $7 \times 7 \times 64$. In the classification block, this matrix is first flattened to generate into a set of 3136 neurons followed by two dense layers as fully connected layers of 128 neurons each. The output of dense layers is finally passed through the classification layer to predict the record. The decision of this layer is made based on either the *sigmoid* or *softmax* activation layer depending upon the binary and multiclass classification scenarios. In the case of binary classification, the result will be either Benign or Attack. On the other hand, the result of the multiclass classification result into one of the Benign, Attack, Brute force, DDoS, DoS, Portscan, or web attack.

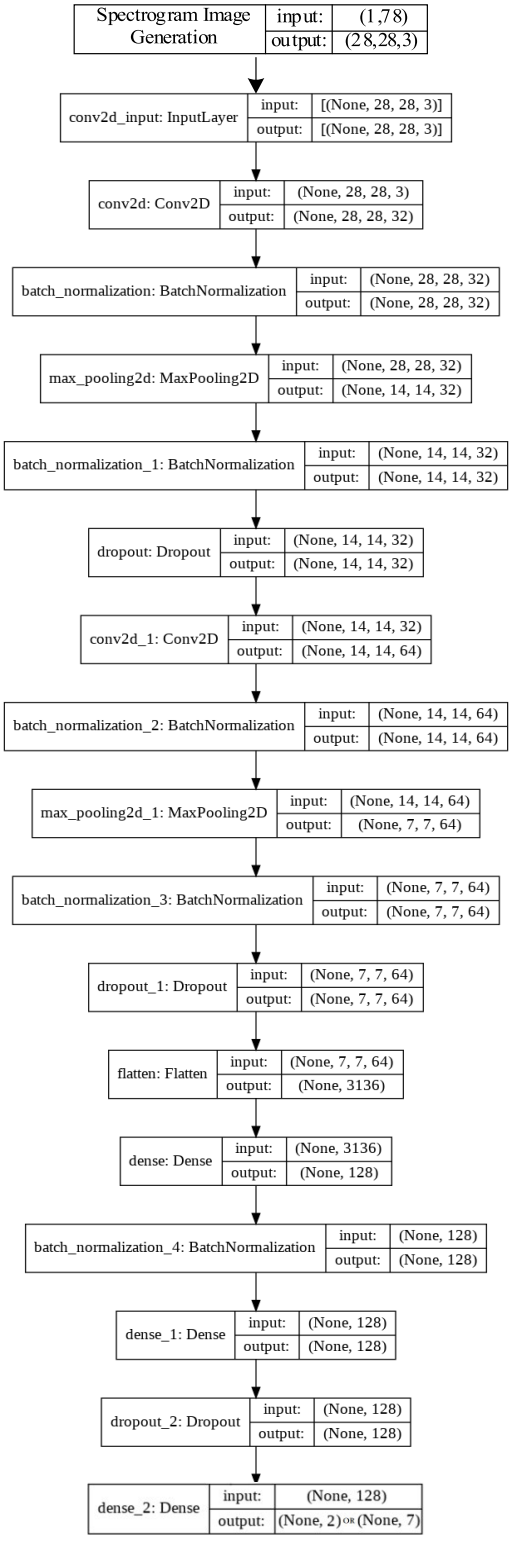

FIGURE 3. SDCNN – model configuration.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

A. DATASET DESCRIPTION

Many datasets are publicly available to evaluate the performance of the ML or DL-based NIDS systems. Some of the well-known datasets are KDD Cup'99, NSL-KDD,

TABLE 2. Actual CICIDS2017 dataset class wise distribution.

Category	No. of Records
Benign	2272688
DoS	251,712
DDoS	128027
Port scan	158930
Brute Force	13832
Web Attack	2180
Bot	1956
Infiltration	36
Heartbleed	11

UNSW-NB15, and CIC-IDS2017, etc. KDD Cup'99 is an imbalanced dataset with a very large number of redundant records causing biased results. NSL-KDD dataset is adopted from KDD Cup'99 by removing the problems within it [43]. Both KDD Cup'99 and NSL-KDD are although widely used datasets, but they are outdated to represent the current network architecture which is much different than what was 20 years ago [44]. UNSW-NB-15 dataset is generated by the University of New South Wales in 2015 [45], while CICIDS2017 is created by Canadian Institute for Cybersecurity in 2017. Both these datasets are newer representing the benign and anomaly flows that represent the current network architecture.

For this study, we considered using the publicly available intrusion detection dataset CIC-IDS2017 (Canadian Cyber Security Institute Dataset) to evaluate our proposed solution. This dataset contains records of the most common benign and different types of attack network flows [46]. The different attacks within the dataset are Port scan, Brute Force, DoS, DDoS, Web attack, Botnet, Infiltration, and Heartbleed attacks and the number of records after cleaning are mentioned in Table 2. As observed from the table, some of the attacks have very low records. Hence, we combined the attacks Botnet, Infiltration, and Heartbleed into a single class named Attack to let the model get trained with enough flows. The complete set of the features for each record within CIC-IDS2017 data is given in Table 3. Each record contains 79 features, with 78 numerical features and one categorical feature for labeling each record. We considered using a complete feature set (78 features of a record) for the generation of the spectrogram images in this study.

For this study, experiments were performed considering both binary and multiclass classification. To prepare the dataset for the binary classification, firstly the labels of all the different attacks within the dataset are changed as "Attack" and then randomly the records are chosen with approximately 22000 entries of each class. The distribution considered for the binary classification is given in Table 4. While for the multiclass classification, we combined the minority class categories Botnet, Infiltration, Heartbleed as an "Attack" class to have the maximum available number of samples for training and testing. The distribution of the samples per class considered for multiclass classification is detailed in Table 5. The number of labels considered for the multiclass classification is 7, with 1 class as benign and the remaining 6 representing different attacks.

B. EVALUATION METRICS

For evaluating the efficiency of the SDCNN model, Accuracy, Precision, Recall, F1-Score, FAR, and True Negative Rate (TNR) are considered as performance evaluation metrics. All these evaluation metrics are calculated from the different attributes within the confusion matrix shown in Table 6. In the confusion matrix, TP and TN are the correctly predicted Attack and Benign instances respectively while FN and FP are wrongly predicted instances as Benign and Attack, respectively. The different evaluation metrics considered in this study are [3], [47],

1. *Accuracy:* It is the ratio of correctly classified instances to the total number of instances.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

2. *Precision:* It denotes the ratio of correctly predicted Attacks to all the samples predicted as Attacks.

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

3. *Recall:* It is a ratio of all the correctly classified Attack samples to all the samples that are actually Attacks.

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

4. *F1 Score*: It is the harmonic mean of the *Precision* and *Recall* and provides a statistical technique for examining the accuracy of a system.

$$F1Score = 2\left(\frac{Precision \times Recall}{Precision + Recall}\right) \tag{12}$$

5. *False alarm rate:* It is the ratio of wrongly predicted Attack samples to all the samples that are labeled as Normal.

$$FAR = \frac{FP}{FP + TN} \tag{13}$$

6. *True negative rate:* It is defined as the ratio of the number of correctly classified Normal samples to all the samples labeled Normal.

$$TNR = \frac{TN}{FP + TN} \tag{14}$$

C. EXPERIMENTAL SETUP

All the experiments are performed on an HP laptop with 8GB of RAM, Intel Core I7-8550U, and Nvidia GeForce MX150 with a 64-bit Windows 10 operating system. Matlab 2019a and Python (version 3.6.9) are the used tools to implement the proposed solution. Matlab 2019a is used to generate a spectrogram of the considered dataset while Python is used as the main programming language to implement and evaluate the SDCNN and other DL methodologies in the Google Colab environment with GPU selected as Hardware accelerator [48].

TABLE 3. CICIDS2017 dataset features.

No.	Feature	No.	Feature	No.	Feature	No.	Feature
1	Destination Port	21	Fwd IAT Total	41	Packet Length Mean	61	Bwd Avg Packets/Bulk
2	Flow Duration	22	Fwd IAT Mean	42	Packet Length Std	62	Bwd Avg Bulk Rate
3	Total Fwd Packets	23	Fwd IAT Std	43	Packet Length Variance	63	Subflow Fwd Packets
4	Total Backward Packets	24	Fwd IAT Max	44	FIN Flag Count	64	Subflow Fwd Bytes
5	Total Length of Fwd Packets	25	Fwd IAT Min	45	SYN Flag Count	65	Subflow Bwd Packets
6	Total Length of Bwd Packets	26	Bwd IAT Total	46	RST Flag Count	66	Subflow Bwd Bytes
7	Fwd Packet Length Max	27	Bwd IAT Mean	47	PSH Flag Count	67	Init_Win_bytes_forward
8	Fwd Packet Length Min	28	Bwd IAT Std	48	ACK Flag Count	68	Init_Win_bytes_backward
9	Fwd Packet Length Mean	29	Bwd IAT Max	49	URG Flag Count	69	act_data_pkt_fwd
10	Fwd Packet Length Std	30	Fwd IAT Max	50	CWE Flag Count	70	min_seg_size_forward
11	Bwd Packet Length Max	31	Fwd PSH Flags	51	ECE Flag Count	71	Active Mean
12	Bwd Packet Length Min	32	Bwd PSH Flags	52	Down/Up Ratio	72	Active Std
13	Bwd Packet Length Mean	33	Fwd URG Flags	53	Average Packet Size	73	Active Max
14	Bwd Packet Length Std	34	Bwd URG Flags	54	Avg Fwd Segment Size	74	Active Min
15	Flow Bytes/s	35	Fwd Header Length	55	Avg Bwd Segment Size	75	Idle Mean
16	Flow Packets/s	36	Bwd Header Length	56	Fwd Header Length.1	76	Idle Std
17	Flow IAT Mean	37	Fwd Packets/s	57	Fwd Avg Bytes/Bulk	77	Idle Max
18	Flow IAT Std	38	Bwd Packets/s	58	Fwd Avg Packets/Bulk	78	Idle Min
19	Flow IAT Max	39	Min Packet Length	59	Fwd Avg Bulk Rate	79	Label
20	Flow IAT Min	40	Max Packet Length	60	Bwd Avg Bytes/Bulk		

TABLE 4. Binary classification: CICIDS2017 dataset distribution.

Category	No. of Records	Training (75%)	Testing (25%)
Benign	22325	16744	5581
Attack	25000	18750	6250
Total Records	**47325**	**35494**	**11831**

TABLE 5. Multiclass classification: CICIDS2017 dataset distribution.

Category	No. of Records	Training (75%)	Testing (25%)
Benign	25,000	18750	6250
DoS	25000	18750	6250
DDoS	25000	18750	6250
Port scan	25000	18750	6250
Brute force	13832	10374	3458
Web Attack	2180	1635	545
Attack	2003	1503	500
Total Records	**118015**	**88512**	**29503**

TABLE 6. A confusion matrix.

		PREDICTED	
		Attack	**Benign**
ACTUAL	**Attack**	True Positive (TP)	False Negative (FN)
	Benign	False Positive (FP)	True Negative (TN)

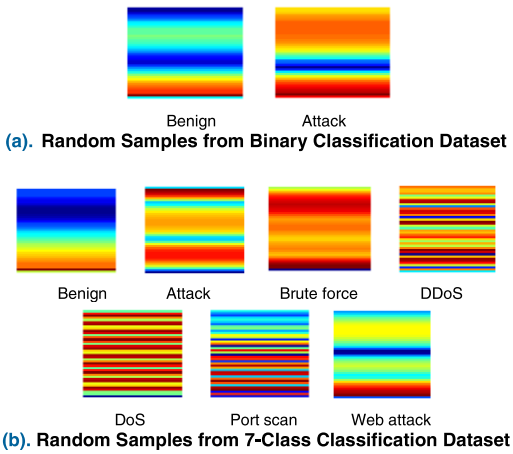

FIGURE 4. Sample spectrogram images.

Spectrogram images are generated using Matlab by first importing the dataset which was already cleaned, normalized, and encoded. Then STFT technique was adopted to generate the spectrograms that are saved as 28 × 28 images with 3 channels. Figure 4 depicts a random sample per label from the spectrogram datasets considered in this study.

For this study, we calculated the conversion overhead for the spectrogram images in terms of the spectrogram generation time. We observe that the average time taken for one spectrogram image generation is 0.150 seconds, while the average time recorded for the conversion of a 1D vector into 2D Matrix followed by black and white image generation is 0.450 seconds. The average is calculated for the 500 random

TABLE 7. Different parameters used in experiments.

Parameter	Value
Batch size	$\{2^6, 2^7, 2^8, 2^9\}$
Learning Rate	0.01
Drop out	[0,1]
Optimizer	Adam
Activation	*ReLU, Sigmoid, Softmax*

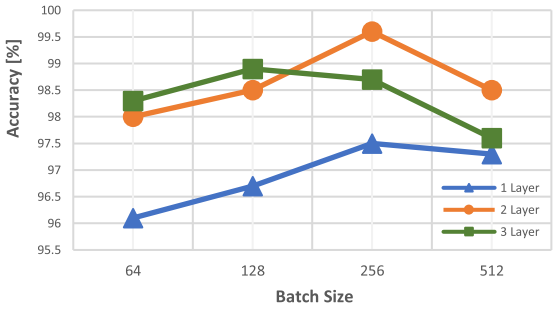

FIGURE 5. Binary classification: accuracy percentage per batch size for different layers of CNN.

samples from the dataset. Based on this analysis we believe that the overall time take by our proposed methodology SDCNN will be less comparing the CNN based methodologies adopted in literature.

D. RESULTS - BINARY CLASSIFICATION

Table 7 details the list of the parameter used in the study to find the optimal parameters and the number of hidden layers for all the considered DL models including the SDCNN. To find the optimum number of the hidden layer (a stack of CL and PL) using the batch size, different experiments are performed by varying the number of hidden layers and batch sizes. Figure 5 depicts the percentage accuracy scores for the different batch sizes by varying the number of hidden layer pairs. A higher accuracy percentage is observed for two layers of CL and PL stack at the batch size of 256. So, for a fair comparison with other DL methodologies, we used the 2 hidden layers and the batch size of 2^8. The model is trained for 100 epochs and the best model is saved to be used for the testing phase.

The overall experiment scores for the binary classification are summarized in Table 8. The proposed methodology is compared against 5 different DL methodologies such as DNN, CNN-1D, RNN, LSTM, and GRU. The detailed comparison of the proposed methodology is depicted in Figure 6. It is observed that our proposed SDCNN outperformed other DL methodologies in terms of all the evaluation metrics such as accuracy, recall, precision, F1 Score, FAR, and TNR. An improvement of almost $2.5\% - 4\%$ in the accuracy is observed by SDCNN comparing the other DL methodologies. At the same time, a reduction of $4.3\% - 6.7\%$ in the FAR rate and a similar improvement in terms of TNR is observed for the SDCNN model.

Table 9 provides the details about the percentage precision, recall, and F1 score of the individual classes. It is observed

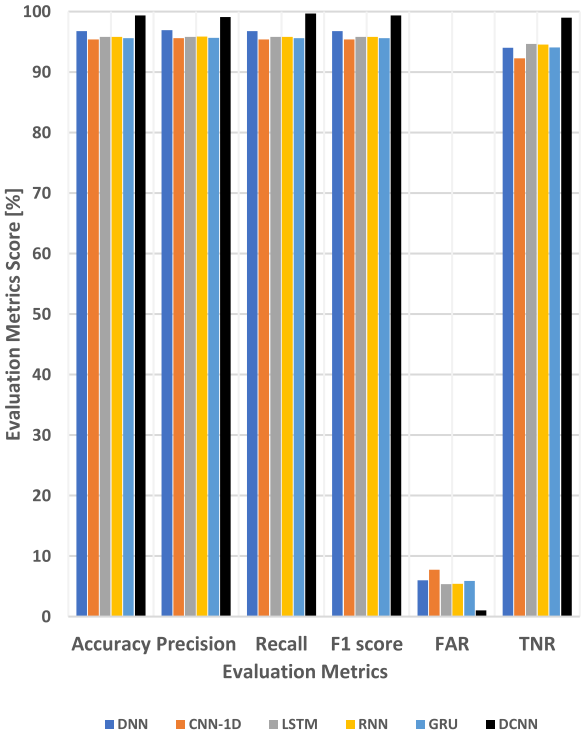

FIGURE 6. Binary classification: Performance evaluation metrics.

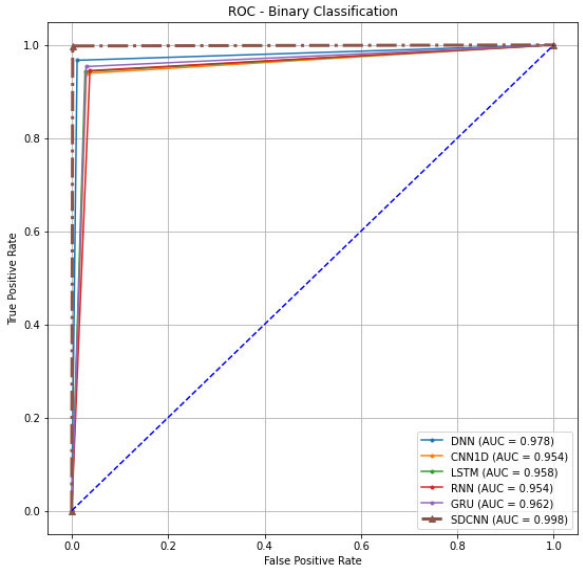

FIGURE 7. Binary classification: ROC curves.

that the SDCNN outperformed other DL-based NIDS models with a higher correct prediction rate. We also observed the lower detection rate for the LSTM model for the Attack class while CNN-1D exhibited a lower detection rate for Benign traffic. The CNN-1D model exhibited a lower percentage of predicting the Attack class while LSTM, RNN, and GRU all performed worst among other considered DL methodologies to predict the Benign class.

The receiver operating characteristic curve (ROC) for the binary classification is plotted in Figure 7. The SDCNN

TABLE 8. Binary classification: Performance evaluation metrics [%].

Algorithms	EVALUATION METRICS [%]					
	Accuracy	Precision	Recall	F1 score	FAR	TNR
DNN	96.766	96.919	96.747	96.762	5.977	94.023
CNN-1D	95.399	95.588	95.377	95.392	7.723	92.277
LSTM	95.799	95.829	95.791	95.797	5.373	94.627
RNN	95.832	95.867	95.823	95.830	5.440	94.560
GRU	95.599	95.646	95.588	95.597	5.910	94.090
SDCNN	**99.349**	**99.073**	**99.694**	**99.382**	**1.033**	**98.966**

TABLE 9. Binary classification: Performance evaluation metrics [%] of the individual classes.

Algorithms	Attacks			Benign		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
DNN	94.406	99.470	96.872	99.431	94.022	96.651
CNN-1D	92.822	98.477	95.566	98.354	92.277	95.218
LSTM	94.819	96.954	95.874	96.839	94.627	95.720
RNN	94.764	97.086	95.911	96.970	94.560	95.750
GRU	94.337	97.086	95.692	96.955	94.090	95.501
SDCNN	**99.807**	**99.791**	**99.799**	**99.768**	**99.786**	**99.777**

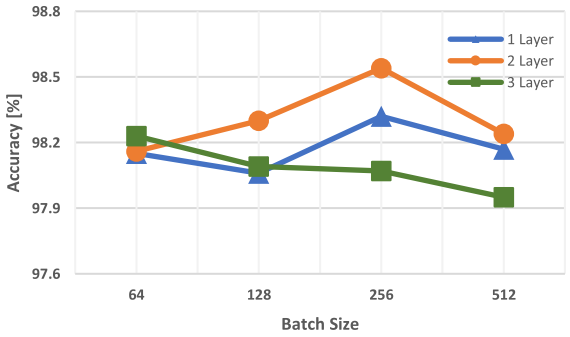

FIGURE 8. Multiclass classification: Accuracy percentage per batch size for different layers of CNN.

performed much better than other DL methodologies with Area under the ROC curve (AUC) considered as the metric. We observed a higher AUC score of 0.998 which is $2\% - 4.5\%$ better than other considered DL methodologies. Also, the SDCNN curve is much closer on the upper left corner of the ROC curve depicting a robust model with higher correct predictions.

E. RESULTS - MULTICLASS CLASSIFICATION

We repeated the same steps as discussed for binary classification in subsection *D*. We used the same parameters and the method adopted to find the optimal parameters and the number of hidden layers for 7-class classification experiments. Figure 8 shows the percentage accuracy scores of the different layer stack of CL and PL against the different batch sizes. Interestingly, we observed that the multiclass model of SDCNN also achieved a higher accuracy score for the 2 layers using the batch size of 256. Hence, we fixed the same setting of 2 hidden layers for implementing all the considered DL-based NIDS models for a fair comparison.

TABLE 10. Multiclass classification: Performance evaluation metrics [%].

Algorithms	EVALUATION METRICS [%]			
	Accuracy	Precision	Recall	F1 score
DNN	98.183	97.248	96.328	96.762
CNN-1D	95.034	91.502	92.834	92.109
LSTM	96.668	92.940	94.703	93.766
RNN	96.518	94.097	92.521	93.111
GRU	96.783	92.880	95.589	94.112
SDCNN	**98.758**	**98.653**	**98.691**	**98.670**

The experimental results for the 7-class classification experiments are detailed in Table 10. The performance is evaluated using accuracy, precision, recall, and F1 score for SDCNN and other DL methodologies. We observed that our proposed NIDS methodology based on the spectrogram (SDCNN) performed better than other methodologies. The performance evaluation for the 7-class classification experiments is depicted in Figure 9. It is observed that both SDCNN and DNN performed well in terms of accuracy with the former performed slightly better. But on the other hand, SDCNN performed much better in terms of precision, recall, and F1 score from DNN and other DL-based algorithms. We also find out that the CNN-1D performed worst among the DL methodologies in terms of accuracy, precision, and F1 score.

Table 11 provides the details about the percentage performance scores of an individual class, exhibited by all the considered DL methodologies. It is observed that SDCNN

TABLE 11. Multiclass classification: Performance evaluation metrics [%] of the individual classes.

Algorithms	Attack						Benign					
	Accuracy	Precision	Recall	F1 Score	FAR	TNR	Accuracy	Precision	Recall	F1 Score	FAR	TNR
DNN	99.573	90.315	82.851	86.422	0.148	99.852	98.586	97.648	95.589	96.607	0.614	99.386
CNN-1D	99.369	78.765	84.298	81.437	0.379	99.621	96.871	95.681	89.164	92.308	1.074	98.926
LSTM	99.364	78.072	85.331	81.540	0.401	99.599	97.394	96.610	90.855	93.644	0.854	99.146
RNN	99.349	88.624	69.215	77.726	0.148	99.852	97.352	94.861	92.433	93.631	1.336	98.664
GRU	99.417	77.083	91.736	83.774	0.455	99.545	97.498	97.076	90.855	93.862	0.730	99.270
SDCNN	**99.941**	**99.586**	**98.163**	**98.870**	**0.014**	**99.993**	**99.816**	**99.548**	**99.494**	**99.521**	**0.124**	**99.893**

	Brute Force						DDoS					
	Accuracy	Precision	Recall	F1 Score	FAR	TNR	Accuracy	Precision	Recall	F1 Score	FAR	TNR
DNN	99.929	99.770	99.626	99.698	0.040	99.969	**99.630**	**98.878**	**99.372**	**99.124**	**0.301**	**99.699**
CNN-1D	99.525	97.253	98.763	98.002	0.373	99.627	98.078	92.586	98.776	95.581	2.108	97.892
LSTM	99.721	98.705	98.677	98.691	0.174	99.826	98.820	96.766	98.792	97.768	0.887	99.113
RNN	99.729	98.404	99.310	98.855	0.215	99.785	98.963	96.081	99.114	97.574	1.078	98.922
GRU	99.742	98.627	99.195	98.910	0.184	99.816	99.051	97.093	98.438	97.760	0.786	99.214
SDCNN	**99.966**	**99.971**	**99.914**	**99.942**	**0.032**	**99.996**	98.953	97.459	97.767	97.613	0.715	99.285

	DoS						Port Scan					
	Accuracy	Precision	Recall	F1 Score	FAR	TNR	Accuracy	Precision	Recall	F1 Score	FAR	TNR
DNN	**99.013**	97.086	**98.309**	**97.694**	0.796	99.204	99.692	98.786	99.777	99.279	0.332	99.668
CNN-1D	97.342	95.428	91.897	93.629	1.188	98.812	99.475	97.829	99.745	98.778	0.599	99.401
LSTM	78.082	97.044	97.416	97.230	1.092	98.908	78.521	97.737	99.745	98.731	21.442	78.558
RNN	98.556	97.435	95.725	96.572	**0.680**	99.320	99.553	98.182	99.745	98.957	0.500	99.500
GRU	98.769	96.666	97.575	97.118	0.908	99.092	99.553	98.182	99.745	98.957	0.500	99.500
SDCNN	98.750	**97.623**	97.359	97.491	1.189	**99.339**	**99.954**	**99.999**	**99.984**	**99.992**	**0.035**	**100.000**

	Web Attack					
	Accuracy	Precision	Recall	F1 Score	FAR	TNR
DNN	**99.942**	**98.255**	**98.772**	**98.513**	**0.035**	**99.965**
CNN-1D	99.407	82.972	87.193	85.030	0.353	99.647
LSTM	99.548	85.644	92.105	88.757	0.305	99.695
RNN	99.536	85.089	92.105	88.458	0.318	99.682
GRU	99.536	85.434	91.579	88.400	0.308	99.692
SDCNN	99.882	96.383	98.158	97.263	0.085	99.929

performed very well to detect the Benign, Attack, Brute force, and Port scan classes while DNN performed well in detecting the DoS, DDoS, and web attacks. It is also observed that LSTM performed poorly in detecting the DoS and Port scan attacks.

The Confusion matrix obtained for the best model (SDCNN) considering the binary and 7-class classification scenario is shown in Figures 10 and 11, respectively. It is observed from the confusion matrix of Figure 10 that SDCNN performed equally well in detecting both the classes for binary classification scenarios with slightly more wrong predictions for Benign traffic. While it is observed from Figure 11 that the proposed model had more wrong predictions for DoS and DDoS classes by predicting a DoS instance as DDoS and vice versa. It is also observed that SDCNN showed an improvement in reducing the FAR.

The SDCNN model is compared directly with some of the notable recent DL-based NIDS solutions in Table 12, considering accuracy as a performance evaluation metric. The accuracy results obtained by those methodologies are directly collected from the respective reference. It is observed that the SDCNN model performed quite well comparing these considered works. This result demonstrates the effectiveness of our proposed model in terms of detecting the intrusions within the network.

The proposed SDCNN complexity is measured in terms of the training and testing time. Since all the models including

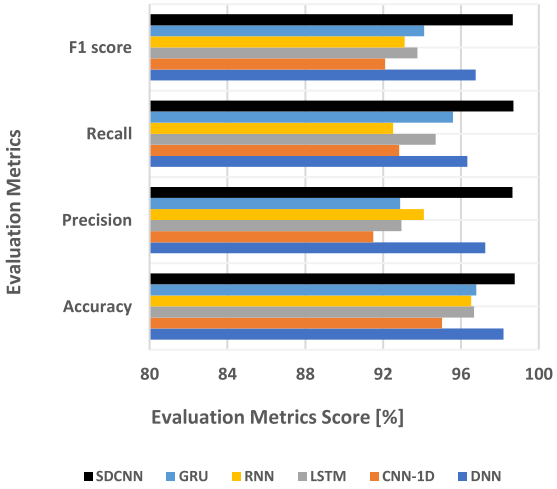

FIGURE 9. Multiclass classification: Performance evaluation metrics.

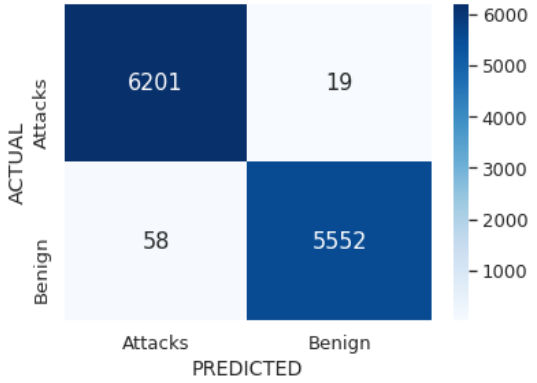

FIGURE 10. Confusion matrix for SDCNN - Binary classification.

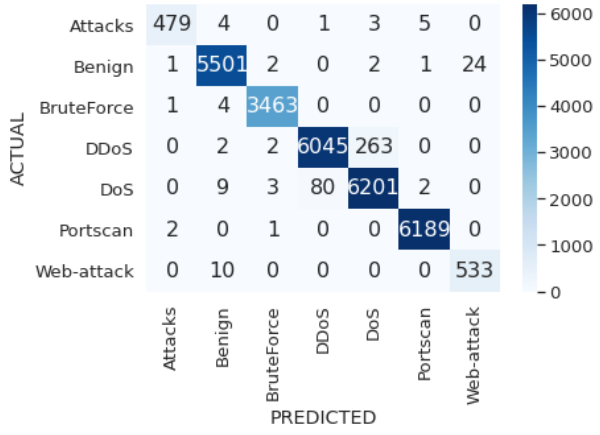

FIGURE 11. Confusion matrix for SDCNN - 7-class classification.

SDCNN are trained for 100 epochs, the training time of 350*s* (*s* denotes the seconds) is recorded for the SDNN, with the testing time of 0.669*s*. We also observed that one network record is converted into the spectrogram image in average 0.150*s*, which then takes approximately 0.30*s* for preprocessing task before the training phase. It means if a trained model is deployed in the network edge routers, one network flow will take almost 1.12*s* to be predict as a benign or attack behavior. The training time depends upon the DL model parameters such as the batch size, feature set size, number of epochs, etc. along with the size of the dataset considered. In this study, our main focus was to improve the accuracy and reduce the FAR considering the frequency details of the network flows.

TABLE 12. Direct comparison with the recent works accuracy values are collected from the reference paper.

Reference	Methodology	Accuracy [%]
Andresini *et al.*[30]	Autoencoder and 1D CNN	97.90
Vinayakumar *et al.* [42]	DNN	96.20
Roopak *et al.*[49]	CNN and LSTM	97.16
Zhu *et al.* [50]	Attention-based Multi-Flow LSTM	91.00
This Study	**SDCNN**	**98.76**

Based on the performance results discussed in this section for both binary and multiclass classification, we conclude that our proposed SDCNN will perform efficiently to detect anomalies once deployed in the network.

V. CONCLUSION

This paper proposes a novel idea of transforming the network flows into the spectrogram images using STFT and then using the CNN to process these images to learn the deep and useful patterns and features for efficient predictions. The proposed model was tested using the relatively new dataset CIC-IDS 2017 considering both the binary and 7-class classification scenarios. The results show that the proposed model performed equally well for both the scenarios by detecting the intrusions with high accuracy and at the same reducing the FAR to show its effectiveness. Results show that SDCNN achieved the accuracy of 99.35% and 98.76% for the binary and multiclass classification experiments. Also, it is observed that the model was very accurate in predicting the benign traffic with an accuracy score of 99.8% for binary and multiclass classification with a very low FAR of less than 1. Results also showed that SDCNN performed very well to detect Attack, Brute force, and Port scan intrusion classes comparing DoS, DDoS, and web attacks.

For future research, (i) we will first extend our proposed solution by studying the computational complexity. Then we try to reduce computational complexity of the proposed solution without compromising on the detection accuracy and FAR. (ii) Next, our aim is to test our proposed solution on the different older and newer datasets such as KDD Cup'99, NSL KDD, UNSW-NB15 and Bot-IoT etc. This will allow us to check the usefulness and efficiency of our proposed solution under different datasets and network conditions. (iii) Further, we will also extend this work to check its effectiveness in the unsupervised learning scenarios that will help us to deploy our proposed solution in the real work environment.

APPENDIX

TABLE 13. Summary of important abbreviations.

Abbreviation	Definition
CNN	Convolutional Neural Network
DDoS	Distributed Denial of Service
DL	Deep Learning
DoS	Denial of Service
FAR	False Alarm Rate
GPU	Graphical Processing Unit
GRU	Gated Recurrent Unit
LSTM	Long Short-Term Memory
NIDS	Network Intrusion Detection System
RNN	Recurrent Neural Network
SDCNN	Spectrogram image based Deep CNN
TNR	True Negative Rate

ACKNOWLEDGMENT

The authors would like to thank the Research, Innovation and Enterprise Centre (RIEC), Universiti Malaysia Sarawak, Sarawak, Malaysia, for encouraging this research work.

REFERENCES

[1] F. Ahmad, Z. Ahmad, C. A. Kerrache, F. Kurugollu, A. Adnane, and E. Barka, "Blockchain in Internet-of-Things: Architecture, applications and research directions," in *Proc. Int. Conf. Comput. Inf. Sci. (ICCIS)*, Apr. 2019, pp. 1–6, doi: 10.1109/ICCISci.2019.8716450.

[2] A. Tarter, "Importance of cyber security," in *Community Policing—A European Perspective* (Advanced Sciences and Technologies for Security Applications), P. Bayerl, R. Karlović, B. Akhgar, and G. Markarian, Eds. Cham, Switzerland: Springer, 2017, pp. 213–230, doi: 10.1007/978-3-319-53396-4_15.

[3] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 1, Jan. 2021, Art. no. e4150, doi: 10.1002/ett.4150.

[4] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2671–2701, 3rd Quart., 2019, doi: 10.1109/COMST.2019.2896380.

[5] T. Verwoerd and R. Hunt, "Intrusion detection techniques and approaches," *Comput. Commun.*, vol. 25, no. 15, pp. 1356–1365, Sep. 2002, doi: 10.1016/S0140-3664(02)00037-3.

[6] J. Li, Y. Qu, F. Chao, H. P. H. Shum, E. S. L. Ho, and L. Yang, "Machine learning algorithms for network intrusion detection," in *AI in Cybersecurity* (Intelligent Systems Reference Library), vol. 151, L. Sikos, Ed. Cham, Switzerland: Springer, 2019, pp. 151–179, doi: 10.1007/978-3-319-98842-9_6.

[7] R. Prasad and V. Rohokale, *Artificial Intelligence and Machine Learning in Cyber Security*. Cham, Switzerland: Springer, 2020, pp. 231–247.

[8] J. Lew, D. A. Shah, S. Pati, S. Cattell, M. Zhang, A. Sandhupatla, C. Ng, N. Goli, M. D. Sinclair, T. G. Rogers, and T. M. Aamodt, "Analyzing machine learning workloads using a detailed GPU simulator," in *Proc. IEEE Int. Symp. Perform. Anal. Syst. Softw. (ISPASS)*, Mar. 2019, pp. 151–152, doi: 10.1109/ISPASS.2019.00028.

[9] S. Khan, H. Rahmani, S. A. A. Shah, and M. Bennamoun, "A guide to convolutional neural networks for computer vision," *Synth. Lectures Comput. Vis.*, vol. 8, no. 1, pp. 1–207, Feb. 2018, doi: 10.2200/s00822ed1v01y201712cov015.

[10] Z. Ouyang, H. Yu, W.-P. Zhu, and B. Champagne, "A fully convolutional neural network for complex spectrogram processing in speech enhancement," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 5756–5760, doi: 10.1109/ICASSP.2019.8683423.

[11] M. T. Nguyen and K. Kim, "Genetic convolutional neural network for intrusion detection systems," *Future Gener. Comput. Syst.*, vol. 113, pp. 418–427, Dec. 2020, doi: 10.1016/j.future.2020.07.042.

[12] Y. Liu, S. Liu, and X. Zhao, "Intrusion detection algorithm based on convolutional neural network," *Beijing Ligong Daxue Xuebao/Trans. Beijing Inst. Technol.*, vol. 37, no. 12, pp. 1271–1275, 2017, doi: 10.12783/dtetr/iceta2017/19916.

[13] R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Applying convolutional neural network for network intrusion detection," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2017, pp. 1222–1228, doi: 10.1109/ICACCI.2017.8126009.

[14] G. Andresini, A. Appice, and D. Malerba, "Nearest cluster-based intrusion detection through convolutional neural networks," *Knowl. Based Syst.*, vol. 216, Mar. 2021, Art. no. 106798, doi: 10.1016/j.knosys.2021.106798.

[15] A. Satt, S. Rozenberg, and R. Hoory, "Efficient emotion recognition from speech using deep learning on spectrograms," in *Proc. Interspeech*, Aug. 2017, pp. 1089–1093, doi: 10.21437/Interspeech.2017-200.

[16] A. Azab and M. Khasawneh, "MSIC: Malware spectrogram image classification," *IEEE Access*, vol. 8, pp. 102007–102021, 2020, doi: 10.1109/ACCESS.2020.2999320.

[17] M. J. Barney, "GPU advancements reduce simulation times for 25 GHz automotive radar models," in *Proc. 9th Eur. Conf. Antennas Propag., (EuCAP)*, Apr. 2015, pp. 1–3.

[18] H. Yao, D. Fu, P. Zhang, M. Li, and Y. Liu, "MSML: A novel multilevel semi-supervised machine learning framework for intrusion detection system," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1949–1959, Apr. 2019, doi: 10.1109/JIOT.2018.2873125.

[19] M. H. Ali, B. A. D. Al Mohammed, A. Ismail, and M. F. Zolkipli, "A new intrusion detection system based on fast learning network and particle swarm optimization," *IEEE Access*, vol. 6, pp. 20255–20261, 2018, doi: 10.1109/ACCESS.2018.2820092.

[20] Y. Shen, K. Zheng, C. Wu, M. Zhang, X. Niu, and Y. Yang, "An ensemble method based on selection using bat algorithm for intrusion detection," *Comput. J.*, vol. 61, no. 4, pp. 526–538, Apr. 2018, doi: 10.1093/comjnl/bxx101.

[21] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 41–50, Feb. 2018, doi: 10.1109/TETCI.2017.2772792.

[22] B. Yan and G. Han, "Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system," *IEEE Access*, vol. 6, pp. 41238–41248, 2018, doi: 10.1109/ACCESS.2018.2858277.

[23] N. Marir, H. Wang, G. Feng, B. Li, and M. Jia, "Distributed abnormal behavior detection approach based on deep belief network and ensemble SVM using spark," *IEEE Access*, vol. 6, pp. 59657–59671, 2018, doi: 10.1109/ACCESS.2018.2875045.

[24] C. Xu, J. Shen, X. Du, and F. Zhang, "An intrusion detection system using a deep neural network with gated recurrent units," *IEEE Access*, vol. 6, pp. 48697–48707, 2018, doi: 10.1109/ACCESS.2018.2867564.

[25] Y. Yang, K. Zheng, B. Wu, Y. Yang, and X. Wang, "Network intrusion detection based on supervised adversarial variational auto-encoder with regularization," *IEEE Access*, vol. 8, pp. 42169–42184, 2020, doi: 10.1109/ACCESS.2020.2977007.

[26] P. Wei, Y. Li, Z. Zhang, T. Hu, Z. Li, and D. Liu, "An optimization method for intrusion detection classification model based on deep belief network," *IEEE Access*, vol. 7, pp. 87593–87605, 2019, doi: 10.1109/ACCESS.2019.2925828.

[27] Y. Xiao, C. Xing, T. Zhang, and Z. Zhao, "An intrusion detection model based on feature reduction and convolutional neural networks," *IEEE Access*, vol. 7, pp. 42210–42219, 2019, doi: 10.1109/ACCESS.2019.2904620.

[28] X. Zhang, J. Chen, Y. Zhou, L. Han, and J. Lin, "A multiple-layer representation learning model for network-based attack detection," *IEEE Access*, vol. 7, pp. 91992–92008, 2019, doi: 10.1109/ACCESS.2019.2927465.

[29] K. Jiang, W. Wang, A. Wang, and H. Wu, "Network intrusion detection combined hybrid sampling with deep hierarchical network," *IEEE Access*, vol. 8, pp. 32464–32476, 2020, doi: 10.1109/ACCESS.2020.2973730.

[30] G. Andresini, A. Appice, N. D. Mauro, C. Loglisci, and D. Malerba, "Multi-channel deep feature learning for intrusion detection," *IEEE Access*, vol. 8, pp. 53346–53359, 2020, doi: 10.1109/ACCESS.2020.2980937.

[31] B. Riyaz and S. Ganapathy, "A deep learning approach for effective intrusion detection in wireless networks using CNN," *Soft Comput.*, vol. 24, no. 22, pp. 17265–17278, Nov. 2020, doi: 10.1007/s00500-020-05017-0.

[32] S. Ganapathy, K. Kulothungan, S. Muthurajkumar, M. Vijayalakshmi, P. Yogesh, and A. Kannan, "Intelligent feature selection and classification techniques for intrusion detection in networks: A survey," *EURASIP J. Wireless Commun. Netw.*, vol. 2013, no. 1, pp. 1–16, Dec. 2013, doi: 10.1186/1687-1499-2013-271.

[33] P. Nancy, S. Muthurajkumar, S. Ganapathy, S. V. N. S. Kumar, M. Selvi, and K. Arputharaj, "Intrusion detection using dynamic feature selection and fuzzy temporal decision tree classification for wireless sensor networks," *IET Commun.*, vol. 14, no. 5, pp. 888–895, Mar. 2020, doi: 10.1049/iet-com.2019.0172.

[34] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Apr. 2017, pp. 712–717, doi: 10.1109/ICOIN.2017.7899588.

[35] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescape, "Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges," *IEEE Trans. Netw. Service Manage.*, vol. 16, no. 2, pp. 445–458, Jun. 2019, doi: 10.1109/TNSM.2019.2899085.

[36] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," 2018, *arXiv:1802.09089*. [Online]. Available: http://arxiv.org/abs/1802.09089

[37] G. Bovenzi, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescape, "A hierarchical hybrid intrusion detection approach in IoT scenarios," in *Proc. GLOBECOM IEEE Global Commun. Conf.*, Dec. 2020, pp. 1–7, doi: 10.1109/GLOBECOM42002.2020.9348167.

[38] Y. Zhu, X. Yin, and J. Hu, "Robust fingerprint matching based on convolutional neural networks," in *Mobile Networks and Management* (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering), vol. 235, J. Hu, I. Khalil, Z. Tari, and S. Wen, Eds. Cham, Switzerland: Springer, 2018, pp. 56–65, doi: 10.1007/978-3-319-90775-8_5.

[39] J. Huang, B. Chen, B. Yao, and W. He, "ECG arrhythmia classification using STFT-based spectrogram and convolutional neural network," *IEEE Access*, vol. 7, pp. 92871–92880, 2019, doi: 10.1109/ACCESS.2019.2928017.

[40] L. Yuan and J. Cao, "Patients' EEG data analysis via spectrogram image with a convolution neural network," in *Intelligent Decision Technologies 2017* (Smart Innovation, Systems and Technologies), vol. 72, I. Czarnowski, R. Howlett, and L. Jain, Eds. Cham, Switzerland: Springer, 2018, pp. 13–21, doi: 10.1007/978-3-319-59421-7_2.

[41] F. Fuentes and D. C. Kar, "Ethereal vs. Tcpdump: A comparative study on packet sniffing tools for educational purpose," *J. Comput. Sci. Colleges*, vol. 204, no. 4, pp. 169–176, 2005, doi: 10.5555/1047846.1047873.

[42] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41525–41550, 2019, doi: 10.1109/ACCESS.2019.2895334.

[43] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, Jul. 2009, pp. 1–6, doi: 10.1109/CISDA.2009.5356528.

[44] K. Siddique, Z. Akhtar, F. Aslam Khan, and Y. Kim, "KDD cup 99 data sets: A perspective on the role of data sets in network intrusion detection research," *Computer*, vol. 52, no. 2, pp. 41–51, Feb. 2019, doi: 10.1109/MC.2018.2888764.

[45] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6, doi: 10.1109/MilCIS.2015.7348942.

[46] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. ICISSp*, Jan. 2018, pp. 108–116, doi: 10.5220/0006639801080116.

[47] M. Usman, M. Zubair, Z. Ahmad, M. Zaidi, T. Ijyas, M. Parayangat, M. Wajid, M. Shiblee, and S. J. Ali, "Heart rate detection and classification from speech spectral features using machine learning," *Arch. Acoust.*, vol. 46, no. 1, pp. 41–53, Mar. 2021, doi: 10.24425/aoa.2021.136559.

[48] E. Bisong and E. Bisong, "Google colaboratory," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. New York, NY, USA: Apress, 2019, pp. 59–64.

[49] M. Roopak, G. Yun Tian, and J. Chambers, "Deep learning models for cyber security in IoT networks," in *Proc. IEEE 9th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2019, pp. 452–457, doi: 10.1109/CCWC.2019.8666588.

[50] M. Zhu, K. Ye, Y. Wang, and C. Z. Xu, "A deep learning approach for network anomaly detection based on AMF-LSTM," in *Network and Parallel Computing* (Lecture Notes in Computer Science), vol. 11276, F. Zhang, J. Zhai, M. Snir, H. Jin, H. Kasahara, and M. Valero, Eds. Cham, Switzerland: Springer, 2018, pp. 137–141, doi: 10.1007/978-3-030-05677-3_13.

ADNAN SHAHID KHAN (Senior Member, IEEE) received the B.Sc. degree (Hons.) in computer science from the University of the Punjab, Lahore, Pakistan, in 2005, and the master's, Ph.D., and Postdoctoral degrees in networks and information security from Universiti Teknologi Malaysia, Johor Bahru, Malaysia, in 2008, 2012, and 2013, respectively. He is currently a Senior Lecturer with the Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak. His research interests include wireless communication, cloud computing, the Internet of Things, software-defined networking, cryptography, networks, and information security.

ZEESHAN AHMAD (Member, IEEE) received the B.S. degree in computer engineering from COMSATS University Islamabad (previously COMSATS Institute of Information Technology), Pakistan, in 2005, the M.S. degree in computer science (networks) from the Virtual University of Pakistan, in 2019, and the M.S. degree in electrical engineering (telecommunication) from Kalmar University College, Sweden, in 2008. He is currently pursuing the Ph.D. degree with Universiti Malaysia Sarawak (UNIMAS), Sarawak, Malaysia. He worked with the Centre Tecnològic de Telecomunicacions de Catalunya (CTTC), Barcelona, Spain, for his M.S. degree thesis for six months, from 2007 to 2008. He has also worked as a Research Assistant with the Institute of Theoretical Information Technology, RWTH Aachen University, Germany, from 2008 to 2010. He is currently affiliated with the Department of Electrical Engineering, King Khalid University, Abha, Saudi Arabia, as a Lecturer. His research interests include machine learning and deep learning methods for network and information security in the Internet-of-Things networks.

JOHARI ABDULLAH received the Bachelor's degree in Computer Science (Networking) from Universiti Putra Malaysia, the Master's degree in IT from the Queensland University of Technology, Brisbane, Australia, and the Ph.D. degree in Computing Science from Newcastle University, U.K. He is currently serving as an Associate Professor with the Faculty of Computer Science and IT, UNIMAS, Sarawak. His research interests include ICT is wide and ranging from trusted systems, blockchain technology, web system design and development, system architecture, problem-solving using tools, such as TRIZ, ICT education for children and youth through computational thinking, scratch, computer science unplugged, and open-source system and software.

FARHAN AHMAD (Member, IEEE) received the B.Sc. degree in Electronics Engineering from COMSATS University Islamabad (previously COMSATS Institute of Information Technology), Pakistan, the M.Sc. degree in Communication and Information Technology from the University of Bremen, Germany, and the Ph.D. degree in Computer Science from the University of Derby, U.K., in 2019. He is currently working as an Assistant Professor with the Systems Security Group, Institute for Future Transport and Cities, Coventry University, U.K. Before joining Coventry University, he was associated with the Cyber Security Research Group, University of Derby, where he worked as a Postdoctoral Research Fellow for two years. He worked as a Post-Graduate Teaching Assistant during his Ph.D. degree, where he conducted tutorials for Cybersecurity modules at both undergraduate and postgraduate levels. He also worked as a Student Research Assistant at the Fraunhofer Institute for Wind Energy and Energy System Technology, Bremerhaven, Germany, from 2013 to 2014. His research interests include vehicular network security and the Industrial Internet of Things security from a trust management perspective.