IEEE Open Journal of the Computer Society

Received 15 October 2020; accepted 6 January 2021. Date of publication 12 January 2021;
date of current version 5 February 2021. The review of this article was arranged by Guest Editors Zhou Su, Abderrahim Benslimane, David (Bong Jun) Choi, Jiliang Li, and Dongfeng Fang.

Digital Object Identifier 10.1109/OJCS.2021.3050917

A Novel Intrusion Detection Model for Detecting Known and Innovative Cyberattacks Using Convolutional Neural Network

SAMSON HO [1], SALEH AL JUFOUT [1], KHALIL DAJANI [2], AND MOHAMMAD MOZUMDAR [1]

[1] Electrical Engineering Department, California State University, Long Beach, CA 90840 USA
[2] California Aerospace Technologies Institute of Excellence, Lancaster, CA 93536 USA

CORRESPONDING AUTHOR: MOHAMMAD MOZUMDAR (e-mail: mohammad.mozumdar@csulb.edu)

This article has supplementary downloadable material available at https://doi.org/10.1109/OJCS.2021.3050917, provided by the authors.

ABSTRACT As a tremendous amount of service being streamed online to their users along with massive digital privacy information transmitted in recent years, the internet has become the backbone of most people's everyday workflow. The extending usage of the internet, however, also expands the attack surface for cyberattacks. If no effective protection mechanism is implemented, the internet will only be much vulnerable and this will raise the risk of data getting leaked or hacked. The focus of this paper is to propose an Intrusion Detection System (IDS) based on the Convolutional Neural Network (CNN) to reinforce the security of the internet. The proposed IDS model is aimed at detecting network intrusions by classifying all the packet traffic in the network as benign or malicious classes. The Canadian Institute for Cybersecurity Intrusion Detection System (CICIDS2017) dataset has been used to train and validate the proposed model. The model has been evaluated in terms of the overall accuracy, attack detection rate, false alarm rate, and training overhead. A comparative study of the proposed model's performance against nine other well-known classifiers has been presented.

INDEX TERMS CICIDS2017, Convolutional neural network, Cybersecurity, Deep learning, Intrusion detection system.

I. INTRODUCTION

In recent years, the amount of applications that stream services to their users has increased explosively. This type of service requires minimal installations and computing power on the user terminal because the applications are operating at the service carrier's cloud servers instead of the local terminal; all the inputs and outputs are streamed to the users via the internet. Seeing the obvious advantage of providing high-end service to customers, who are not able to access high-end devices, many corporations have started to develop their streaming services. For instance, entertaining service such as Google Stadia makes high-end gaming, which is typically hardware demanding, now possible on any portable devices with good internet connectivity. The game is processed and rendered at Google's cloud server with user's inputs in real time, then the video is streamed back to the user's terminal via the internet. However, the extensive data exchange at the network between the cloud servers and local user terminals also expand the attack surface for intrusions. Malicious hackers may deploy various types of attacks, such as Distributed Denial-of-Service (DDoS), Port Scan and Infiltration attack to hijack valuable data or make servers unavailable to users. To stop these cyberattacks from happening, the development of a reliable and effective Intrusion Detection System (IDS) for cybersecurity has become an urgent issue to be solved.

The idea of the IDS is not new. In 1980, James P. Anderson delineated a type of cyber-security device or application to monitor the network traffic with the intention to warn the administrators of any suspicious activities or violations of the system policy [1]. The IDS was proposed to include tools for the administrators to inspect the audit trails of the network traffic in a system. By analyzing the statistical information of

audit trails, system administrators could hunt for malicious traffic and take further actions to secure the system. Most traditional IDS use an attack signature database that is created from expert's knowledge, to cooperate with some predefined decision rules in the program to find out intrusions [2]. In [3], the author claimed that it is easy to develop and understand a signature-based IDS if the network behavior of the target anomaly activities is known. However, in recent years, cyberattacks have become more sophisticated, especially the attacks on the systems that are storing or manipulating sensitive information.

The hand-crafted attack database by experts will obsolete quickly without constant updates. Another major problem with the signature-based IDS is that they fail to be generalized to detect innovative attacks that have signatures that are not logged in the signature database. In short, the attack signature database causes a high storage overhead in order to include the signature of all the known attacks, making them hard to get implemented or distributed. Also, matching the incoming data flow with the signatures logged in the dataset could be computationally expensive.

The Artificial Neural Network (ANN) architecture is a popular solution nowadays for prediction and classification tasks. There are many advantages of ANN which make it especially suitable for network intrusion detection [4]. Firstly, ANN performs great at modelling non-linear data with a large number of input features; for example, network packets. Secondly, once an ANN is trained, its forward propagation, in other words, the predictions, are fast. This is crucial for network's speed if the IDS model is placed in-line with the network traffic. In brief, an ANN is trained by a big dataset to become a generalized solution for a specific task. The traditional signature-based IDS, on the other hand, is using manually defined and human-understandable rules for intrusion detection. The ANN approach relies on a solid mathematics backbone in how the error is backpropagated with the stochastic gradient descent method to optimize the model [5], [6]. In addition, no predefined rules are required during the ANN's training process. Meaning that the developers are not required to possess much expertise in the field of cybersecurity to train an ANN-based IDS. In addition, since the decision mechanisms of the ANN-based IDS are generalized from the features of all the known attacks, they have the potential to detect the innovative attacks sharing similar traits with the known attacks. On the other hand, the signature-based IDS will fail to detect innovative attacks due to the lack of knowledge on their specific signature.

In this paper, an IDS for cybersecurity based on the Convolutional Neural Network (CNN) architecture is proposed. Unlike some previously proposed CNN based IDS, which mostly aiming at one-class or a subset of classes classification [7]–[9]. The proposed IDS has high performance at multi-class classification, for identifying innovative and all the known attack classes in a dataset. And compared to some state-of the-art CNN based IDS's that are aiming at multi-class classification, such as Potluri's work in [10] for classifying the UNSW-NB 15 dataset [11]. The Canadian Institute for Cybersecurity Intrusion Detection System (CICIDS2017) dataset, used in this paper, is more challenging for a classifier with its larger feature set and more attack classes that are 59% and 55% more than the UNSW-NB 15 dataset. This paper is structured to include a brief review of some recent research on deep learning applications at cybersecurity. Then in the model design section, CICIDS2017 database is explored to reveal the design decisions of the custom training and test dataset of the proposed model. Also, the architecture of the CNN, which is the deep learning model on which the proposed IDS is based, is discussed and the mathematical details are provided. Finally, the architecture of the proposed IDS is presented along with its parameter selections. And the model is evaluated by the defined benchmarks in the simulation result section to provide performance validation and comparison.

II. LITERATURE REVIEW

Prior to the rise of machine and deep learning, the design of the IDS is usually based on network experts' knowledge of the attacks. In [12], Ansam Khraisat *et al.* classified various kinds of IDS models based on their detecting techniques. In short, the IDS with statistics-based techniques builds a distribution model for the benign traffic and flag the low probability events as potential attacks. The IDS with knowledge-based techniques, on the other hand, creates a knowledge base to reflect the legitimate traffic profile. Afterward, any action that differs from the standard profile is flagged as an intrusion. Finally, there are IDS with machine learning techniques. These models mine characteristics of each type of attack from large quantities of data and classify traffic based on the learned characteristics. In addition, a survey of the datasets for intrusion detection systems is presented by the authors in [12]. They explored some public datasets such as Knowledge Discovery Databases (KDD) Cup'99, Center for Applied Internet Data Analysis (CAIDA), Network Security Laboratory- Knowledge Discovery Databases (NSL-KDD), and CICIDS2017, and a comparative study of those IDS datasets, in terms of their feature selection and type of computer attacks, is presented as well. Finally, the authors provided classification results of the selected datasets based on their prior research. Specifically, on CICIDS2017, their model that combinedly uses a Multiplayer Perceptron (MLP) neural network and payload classifier reaches an accuracy of 95.2%.

Machine learning is a popular solution for classification tasks due to its relatively simple architecture and low computation overhead. Many studies that are applying machine learning techniques on the CICIDS2017 dataset for attack classification have been proposed [13]–[15]. Many of which have reached decent accuracy at one-class classifications for a certain attack class, such as DDoS, in the dataset. However, to reach a usable multi-class classification accuracy in detecting modern network intrusions, many data preprocessing methods have been proposed to improve the model's performance. In [13], Yonghao Gu *et al.* proposed a semi-supervised K-means model to detect the DDoS attacks in CICIDS2017. Besides,

they implemented a hybrid feature selection algorithm to exclude the unreasonable features from being the input of the model to prevent "the curse of dimensionality". Their feature selection algorithm takes the available features as input. The features are processed by a series of procedures including the data normalization, feature ranking, and feature subset searching, and selected features are output by the algorithm. Finally, with their proposed feature selection method, they reached a detection rate of 96.50% and a false positive rate of 30.5%.

In recent years, deep learning models such as neural networks have become more effective solutions for classification tasks because of their ability to generalize more complex patterns of features of tasks. In [16], the authors provided a study of anomaly analysis for intrusion detection with K-Nearest Neighbors (KNN) and Deep Neural Network (DNN) to compare the classification performance of a machine learning model to a deep learning model. They used CICIDS2017 as the database for the simulations of the model's performance in the study. They concluded that DNN performed significantly better than KNN. In specific, their DNN yields an accuracy of 96.427%, which is much higher than 90.913% accuracy by the KNN. In addition, they also compared the computation time overhead of the two models. The 110 (s) CPU time of DNN is shorter than the 130 (s) CPU time of KNN, therefore, indicated that DNN has a lower time overhead than KNN.

In [7], another study of using deep learning models for cybersecurity in the Internet of Things (IoT) networks is proposed by Monika Roopak, Gui Yun Tian and Jonathon Chambers. The authors used the DDoS attack samples in CICIDS2017 to evaluate the performance of MLP, Long Short-Term Memory (LSTM), CNN, and a hybrid model of LSTM and CNN. They reached a 98.44% precision by the LSTM model, followed by a 98.14% precision by the CNN model and a 97.41% precision by the hybrid model. Lastly, the MLP model reached an 88.47% accuracy in their simulation. The authors also compared their results to some machine learning models. After the simulation, they concluded that all the tested deep learning models except MLP outperform the machine learning models such as SVM, Bayes and Random forest.

In [8], Sungwoog Yeom and Kyungbeak Kim tested the performance of Naïve bayes, SVM and CNN based classifier on the CICIDS2017 dataset. In the study, it was focus more on the model's binary classification performance on each attack class in the dataset. The raw CICIDS2017 dataset, which included separate sub-datasets of the network traffic in a day, was used to train the models. The authors trained and evaluated a CNN based classifier based on each sub-dataset, which mostly included only 1∼3 attack classes. And the accuracy, precision, recall and F-measure of the models were record. After the evaluation, the authors concluded that CNN and SVM generally had high detection rates. In addition, CNN was better than SVM in term of the processing time. However, they also observed that CNN had mediocre performance with datasets with many labels. In other words, it was challenging for the CNN models if there were many labels need to be classified.

In [9], a CNN based IDS was proposed by Jiyeon Kim *et al.* In this paper, the authors employed deep learning techniques and developed a CNN model for the CICIDS2018 dataset, which was a dataset sharing the same feature set with CICIDS2017 but with larger sample counts. The training and test of the models in the study were performed on sub-datasets which included a subset of types of network traffic from CICIDS2018. Therefore, the models were simulated for multi-class classification for certain classes in the dataset, not all of them at once. In this study, the experimental results showed that the performance of the CNN based IDS could be higher than that of the recurrent neural network (RNN), which is another deep learning model that is popular in the cases of time series data being used as input. The CNN model proposed in this study reached a 96.77% accuracy in the sub-dataset which was composed by the benign and DoS samples from CICIDS2018. On the other hand, the RNN model tested in this study reached a 82.84% accuracy in the same dataset, which was significantly lower than of the CNN model.

In [17], Ahmed Ahmim *et al.* proposed a novel hierarchical IDS that is based on Decision Tree and Rules-based models. They also used CICIDS2017 as the dataset to evaluate the performance of their model. Their proposed model combines Reduced Error Pruning Tree (REP Tree) and JRip algorithm at its first stage. The input features of the dataset are used as input at this stage to classify traffic as attacks or benign. Then a Forest PA classifier takes the output of the two classifiers at the first stage, in combination with the input features of the initial dataset as input to generate the final classification result. Their model reached decent performance on almost every traffic class in CICIDS2017. They also provided a performance comparison of their proposed model with 11 well-known classifiers to validate its classification power. Within the 12 classifiers models, their model had the best classification performance at seven attack classes and the lowest false alarm rate for benign traffic. This model is competitive for its great overall classification performance on CICIDS2017. Therefore, in the result section of this paper, the proposed IDS model has been compared against their novel hierarchical IDS to assess the proposed model's performance.

III. MODEL DESIGN

This section specifies many design elements of the proposed IDS model. At first, the cybersecurity database CICIDS2017 that is used to train the proposed model is introduced and analyzed. The data preprocessing methods and the design of the training dataset are also presented. Then, the architecture and mathematical model of the CNN are introduced and discussed. Finally, the architecture, operating flow of the proposed model and input data collecting methods are presented in details.

A. CICIDS2017 SECURITY DATASET

The training, validating, and testing of the proposed model are all accomplished with the data from CICIDS2017 database. CICIDS2017 database was proposed in 2018 by Iman

TABLE 1. CIDIDS2017 With Sample Size and Class Composition

CICIDS 2017			
Traffic Class	**Label**	**Samples**	**Composition**
BENIGN	BENIGN	2273097	80.301%
Brute Force	FTP-Patator	7938	0.281%
	SSH-Patator	5897	0.209%
Botnet	Bot	1966	0.07%
DDoS	DDoS	128027	4.523%
DoS	DoS GoldenEye	10293	0.364%
	DoS Hulk	231073	8.163%
	DoS Slowhttptest	5499	0.195%
	DoS slowloris	5796	0.205%
Heartbleed	Heartbleed	11	0.001%
Infiltration	Infiltration	36	0.002%
PortScan	PortScan	158930	5.615%
Web Attack	Web Attack – Brute Force	1507	0.054%
	Web Attack – Sql Injection	21	0.001%
	Web Attack - XSS	652	0.024%
Total	N/A	2830743	100%

Sharafaldin *et al.* [18]. The database was developed to combat the sophisticated and ever-growing network attacks in the modern era. The creators of the database had the ambition to replace outdating security databases such as DARPA98, KDD99, ISC2012, and ADFA13. These databases were vastly used for the evaluation of intrusion detection and prevention models in many papers in the past decades, but were in fact obsolete and unreliable to use [19]. Therefore, for the proposed model, which aims at detecting modern cyberattacks, CICIDS2017 is a top-notch option for the training database.

The CICIDS2017 database emulates the network traffic of a real environment for one week. The network traffic contains normal network packets and a diverse set of attack scenarios that are injected by the attack profiles created by developers. In total, the database contains 2830,743 samples, and the composition of the traffic classes is presented in Table 1.

Each sample in the database contains 78 features that are extracted from the traffic of two nodes in the network, and a label to indicate the class of the traffic. A few examples of the extracted features are the destination port, flow duration, total forward packets, and total backward packets. Among the 78 features, the proposed model takes all of them as input except for the destination port. The destination port of the network traffic may be a useful audit trail for the network admin to track down the place where a cyberattack is initiated or target towards. However, it is a method of encoding the network ports into identification numbers. The destination port is not a quantitative measure of network traffic that our neural network-based IDS is expected for input. Accordingly, the inclusion of it, as an input feature, can cause unexpected issues at the training process of the proposed neural network model. The rest of the 77 features are all quantitative measure, thus, they are all valid inputs for the proposed model.

B. DESIGN OF THE TRAINING AND TEST DATASET

Albeit CICIDS2017 is an advanced modern security database with merits, it does have certain shortcomings that need to be considered and addressed to unlock its full potential. There are three main issues of the database: scattered presence, missing values and class imbalance. The network traffic is logged into eight separate files corresponding to the time window and class of the traffic sample. The scattered presence is not ideal for the training of the proposed model since the model aims to classify all the cyberattacks in the database instead of particular types of attack. Further, according to [19], CICIDS2017 has a total of 288,602 samples with an absent class label, and 203 instances of missing information. The samples with an absent class label need to be removed, and the missing information needs to be restored.

Table 1 shows the class composition of the original data in CICIDS2017. Among all the samples, the benign samples have a proportion of 80.301%, which is the highest in the database. On the opposite, the Heartbleed samples have a proportion of only 0.001%. This high imbalance of in-class sample size typically leads to biased classification results. It needs to be avoided in the proposed model to prevent a drop at overall performance.

The three issues in the database raise the difficulty for the neural network model to reach decent performance. However, they are not necessarily defects for a database, which is aimed at simulating sophisticated network traffic. In fact, the three issues, mentioned above, do have a high chance to appear in the data that is collected from a real-world environment. Therefore, by solving these shortcomings in CICIDS2017, the proposed model is also prepared to be implemented in real environments.

The first issue of missing values is solved in the data preprocessing phase by a data imputer. In this case, the missing values are all replaced by 0 to prevent value errors. In addition, the rest of the two issues, namely, class imbalance and scattered presence, are solved by building a custom database that concatenates all the separate files of CICIDS2017 with balanced class composition. This custom database is called α-Dataset.

Table 2 presents a few design decisions of the α-Dataset, such as the size of samples for each class, splitting strategy of training and test set, and the class composition of the training set. Balancing the samples for a high imbalance database is a problem with trade-offs. Most neural network models work better with databases with balanced samples in each class. However, neural network models also require a big dataset for better generalization of the task assigned to it. Therefore, it is not feasible to simply trim down the number of samples in the classes with a higher proportion to match with the minority classes. For the best of both worlds, the database is divided into two categories. Namely, the normal attack samples, and minority attack samples that are marked grey in Table 2.

The normal attack samples include the benign samples from the Monday file, DDoS, DoS, and Port Scan. Each of the four classes is split into training and test samples with a distinct

TABLE 2. α-Dataset With Sample Size, Training/Test Distribution and Class Composition in Training Dataset

α Dataset				% in Training set (%)
Label	Samples	Training (% in label)	Test (% in label)	
BENIGN	529918	105983 (20%)	423934 (80%)	24.81%
FTP-Patator	7938	6350 (80%)	1588 (20%) + Training	2.59%
SSH-Patator	5897	4717 (80%)	1180 (20%) + Training	
Bot	1966	1572 (80%)	394 (20%) + Training	0.37%
DDoS	128027	102422 (80%)	25605 (20%)	23.98%
DoS GoldenEye	10293	4118 (40%)	6175 (60%)	23.66%
DoS Hulk	231073	92430 (40%)	138643 (60%)	
DoS Slowhttptest	5499	2200 (40%)	3299 (60%)	
DoS slowloris	5796	2319 (40%)	3477 (60%)	
Heartbleed	11	9 (80%)	2 (20%) + Training	0.001%
Infiltration	36	28 (80%)	8 (20%) + Training	0.01%
PortScan	158930	103304 (65%)	55626 (35%)	24.18%
Web Attack - Brute Force	1507	1205 (80%)	302 (20%) + Training	0.41%
Web Attack - Sql Injection	21	16 (80%)	5 (20%) + Training	
Web Attack - XSS	652	521 (80%)	131 (20%) + Training	
Total	1087564	427190 (39.28%)	660374 (60.72%)	100%

ratio. These ratios are tuned to generate a training set with a balanced size of samples across the four classes. Table 2 shows that each of the four classes has roughly 102,000 samples to form a balanced training set. On the other hand, the minority samples contain the rest of the attack samples in the database. Given their small size of samples, 80% of the samples, in each class, become training samples, and the whole class is also used as test samples to provide sufficient data for the training and testing process.

C. CONVOLUTIONAL NEURAL NETWORK

CNN is a popular neural network that is typically used in image processing tasks. In recent years, due to its popularity, CNN is also extensively used in natural language processing, video analyzing, and even some models with sequential inputs. The unique convolution and pooling process of CNN allow the model to learn complicated patterns of features form a high-dimensional data space while maintaining reasonable storage and computation time overhead. This advantage is significant when a CNN model is trained with a network traffic dataset, given that these datasets typically have a large feature set. For example, there are 78 features recorded for each traffic sample in the CICIDS2017 dataset, which can lead to high computational complexity for other deep learning models such as DNN [20]. Also, for any network IDS, it is

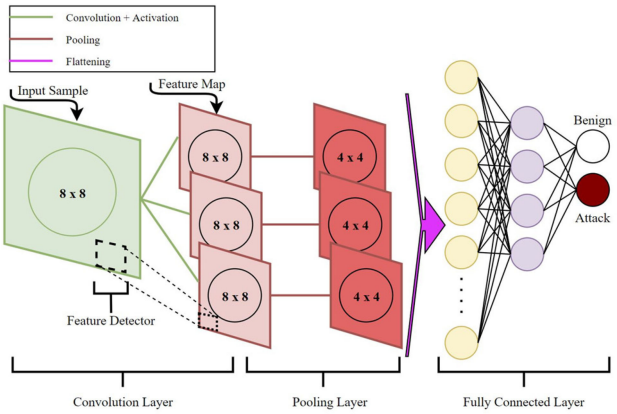


FIGURE 1. Architecture of a basic convolutional neural network.

exceptionally crucial to have the ability to capture the complicated features of network traffic within a short amount of time. Therefore, it is believed that CNN is the priority neural network model for developing the prototype of the proposed model. Fig. 1 shows the architecture of a CNN model with the input size of 8x8 matrix. The whole model could be separated into three main sections: convolution layer, pooling layer, and fully connected layer. In terms of functionality, the convolution layer and pooling layer together handle the feature extraction of the training samples, and the fully connected layer processes the final classification. At the convolution layer, CNN model convolves the input sample with some specific matrices called "feature detector" or "kernel map". The result of the convolution is called "feature map". Feature detectors are matrices used to extract, from the input sample, specific features, such as patterns, shape and lines. Their value is initially randomized and gradually updated by the optimizer during the training process. After the input sample is convolved with the feature detectors, the feature maps are generated. For CNN model with K feature detectors, this process could be mathematically denoted as:

$$feature\ map\{i\} = input \otimes feature\ detector\{i\}; i \in K. \tag{1}$$

Then, the feature map calculated by (1) is mapped to a nonlinear activation function to break the linearity of the model. In Fig. 1, the whole process of generating the feature maps and mapping them to the activation function is represented by the green connections at the convolution layer.

The second section of the model is the pooling layer. At this layer, the feature maps will be pooled to bring down their dimension. The purpose of this process is to eliminate noise and model's dependent on the spatial location of the learned patterns.

In short, pooling mitigates the effects when a specific pattern that the model is trained to recognize appears at different locations or angles at the input sample. Furthermore, to eliminate noise, pooling brings down the dimension of the feature

maps, while preserving the valid information carried by them. This also eases the calculation of any further processing to the feature maps.

After the feature maps are calculated and pooled, they will undergo a "flattening" process before being passed to the fully connected layer. The flattening process converts the feature maps from 2-D matrices to 1-D arrays in row-major order, which is the format expected by the fully connected layer. The fully connected layer is composed of input neurons, optional hidden neurons and output neurons. Each neuron is connected to every neuron at the adjacent layers with distinct weights and biases as the parameter. The output neurons, which are also the final output of the CNN model, determine to which class each input sample belongs.

In the case of cybersecurity, the model can use the output neurons to represent the benign class and every class of attack that the model is trained to classify. When a network traffic sample is processed by the model, it will be classified as the class that is represented by the output neuron that holds the highest value at the output layer.

One of the characteristics that distinguish CNN from other neural networks is weight sharing. CNN has shared weights, which means the model uses the same weight vectors to do the convolution. In other words, the feature detectors, which contain the weight vectors, are replicated at every convolution process at the convolution layer. There are some advantages to the implementation of shared weights. First, it makes the model have substantially fewer parameters to be optimized. This means the optimizer can potentially converge the model to optimal loss with less time overhead. Secondly, it slightly lowers the degrees of freedom of the model. This could help the model to avoid overfitting, which happens when the model makes too much of an effort to fit itself to a limited set of samples.

The constraint on weights also enables the model to achieve better generalization on the tasks assigned to it. Briefly, CNN has merits that are suitable for the development of IDS for cybersecurity in the modern era. CNN requires less time overhead at the testing process. Besides, with its convolution mechanism, CNN could potentially learn the much more complex characteristics of some modern cyberattacks, which other neural network models struggle to capture. Lastly, CNN can achieve better generalization on the classification of cyberattack samples. This enables the IDS to potentially detect innovative attacks that share similar traits with the known attacks.

However, from [7]–[9], it is observed that CNN, albeit with its merits at generalize complex patterns of features in the dataset, has some potential issues at multi-class classifications. The CNN models proposed in these studies are tested on classification tasks of either one-class or a subset of classes in the dataset. This is not a surprise since neural networks depend a lot on training data to reach the full potential and prevent issues such as biased classification results or over-fitting [4]. Therefore, to access the advantage of the neural network models, it is important to train the model with a dataset that is not only rich in its sample counts and feature set but also has a balance class composition.

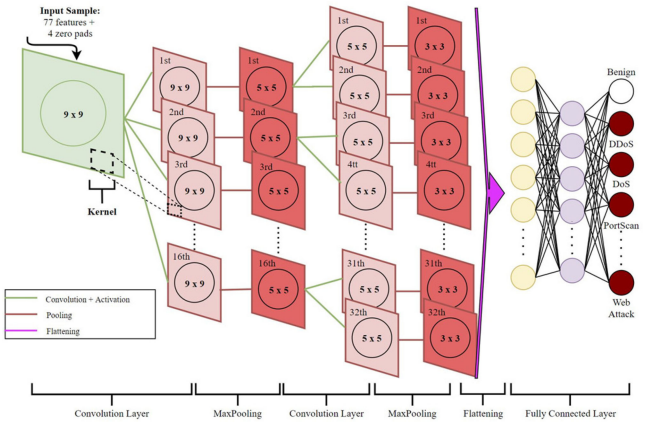


FIGURE 2. Architecture of the proposed IDS model.

In this study, a novel data-preprocessing method is proposed in the Model Design section to generate a relatively balance sub-dataset, which still includes all the available attack classes from the CICIDS2017 dataset, for the training and test of the proposed IDS model. And the proposed IDS is trained to perform multi-class classifications on every types of attacks that are known in the CICIDS2017 dataset.

D. DEVELOPMENT OF THE PROPOSED IDS MODEL

The proposed IDS model, in this paper, is developed on Python 3. PyTorch is used as the framework to build the neural network layers of the proposed model [21].

CNN has many layers to be implemented and some parameters to be selected before training one. Empirically, the number of convolution and fully connected layers for a CNN for network intrusion detection tasks could be selected from the range of 1∼3 layers each [8], [9], [22]. In general, the more complex the task assigned to a neural network, the more layers are added. For example, the ResNet-50, which is a large CNN that classify images into 1000 object classes, has 50 layers in total. However, a large network size does not always guarantee the performance of a CNN [22]. The input of the model is batches of matrices. Each matrix has a size of 9×9, which is composed of 77 extracted features from the network traffic samples and four zero pads for format concern. The model contains two convolution layers and two fully connected layers. The decision of hyperparameters, such as the number of layers, the kernel size for convolution, and the number of neurons in a fully connected layer, are all determined by exhaustive grid search. In this method, all the hyperparameters are selected from a subset of values that are manually specified, and the model will be evaluated iteratively to search for the best combination in the parameter values. The architecture of the proposed model is presented in Fig. 2. At the first convolution layer, the kernel of the convolution is designed to have a size of 3×3, and the stride and padding of

the convolution to be one. In addition, the number of output feature maps is selected to be 16, and the activation function to be a rectified linear unit (ReLU) function at this layer. Accordingly, for each matrix in a training batch, the model generates 16 feature maps at the first convolution layer. Each feature map has a size of 9x9 and is mapped to a ReLU function. Then, the feature maps are pooled at the first pooling layer by the Max Pooling algorithm with a kernel size of 2x2, stride of 2, and padding of 1. This generates 16 pooled feature maps, each has a size of 5x5. Afterward, the pooled feature maps are passed to the second convolution layer and pooling layer, which both have the same kernel size, stride, and padding configuration as the first two layers.

The second convolution layer is designed to output 32 feature maps, and they are also individually mapped to a ReLU activation function. Eventually, after the two convolution layers and two pooling layers, the proposed model generates 32 pooled 3×3 feature maps for each input matrix. At the flattening layer, all the elements of the pooled feature maps are converted into a 1-D array in row-major order, which is the format expected by the fully connected layer. This generates an array of 288 elements for each input matrix. Then the array becomes an input sample of the fully connected layer, which has 288 input neurons, a hidden layer and nine output neurons.

Consequently, the proposed model is trained to classify network traffic into nine different classes represented by the nine output neurons, namely, Benign, Brute Force, Bot, DDoS, DoS, Heartbleed, Infiltration, Port Scan, and Web Attack. The number of classes is designed this way for the model to reach a better generalization for the cyberattacks.

For a certain dataset with a limited number of samples, when the number of classes increases, the number of samples for each class decreases inevitably. It raises the difficulty of the model to generalize the characteristics of each class. Therefore, the proposed model has a custom dataset function to merge some of the samples, based on their type of attack, into a bigger class.

For instance, from CICIDS2017 database, samples with the four labels of DoS attacks, namely, DoS GoldenEye, DoS Hulk, DoS Slowhttptest, and DoS slowloris are merged into a single DoS class. The same merging strategy is also implemented at various labels of Brute Force and Web Attack samples in the database to form bigger classes of the two types of attacks.

Eventually, the samples from CICIDS2017 are assorted into nine distinct classes with some greater sizes of in-class samples to facilitate the model's generalization of cyberattacks. Hence the selection of nine output neurons at the final output layer of the model. Meanwhile, the original label is still attached to each sample for label-wise performance evaluation of the proposed model.

IV. SIMULATION RESULTS

This section presents in detail the performance of the proposed IDS model based on the simulation results with some benchmarks. The benchmarks include the Detection Rate (DR), True Negative Rate (TNR), and overall accuracy, which are all explicitly defined and calculated. In addition, a comparative study of the proposed model with nine other well-known classifiers is provided. Finally, the proposed model's performance on innovative attacks is also validated. All the simulations in this section are done on a workstation of the Networked Embedded Systems (NES) Laboratory at California State University, Long Beach. This workstation runs a 64-bit Windows 10 OS and is equipped with an Intel Xeon W-2125 processor and 64 GB of RAM.

TABLE 3. Confusion Matrix

		Predicted Class	
		Negative	**Positive**
True Class	**Negative**	True negative (TN)	False positive (FP)
	Positive	False negative (FN)	True positive (TP)

A. PERFORMANCE BENCHMARKS

To evaluate the classification performance of the proposed model on cyberattacks, it has been simulated with several benchmarks that are derived from the confusion matrix generated by the model. Confusion matrix, as shown in Table 3, is a matrix showing all the possible scenarios of classification results. In terms of cybersecurity, the positive class is defined to be the attack samples, and the negative class contains the benign samples. There are four scenarios in a confusion matrix, namely, True Negative (TN), True Positive (TP), False Negative (FN) and False Positive (FP). Among the four scenarios, TN and TP contribute to the overall accuracy of the classification outcomes. They represent the instances when a model successfully predicts a sample to be its true case. On the other hand, instances of FN and FP deteriorate the performance of a classification model. They occur when a model predicts a sample incorrectly. In a sense of cybersecurity, FP occurs when a benign sample is classified as an attack, which is also known as a false alarm.

Confusion matrix is a great tool to record the simulation results of a classification model. However, it is not necessarily easy to read nor handy enough to quickly evaluate the performance of IDS models. Thus, for comparative studies, some performance benchmarks are derived from the confusion matrix, which are universally used. The first benchmark is DR, which could be understood as the ratio of the samples in an attack class, which are correctly classified as attacks by an IDS model. Mathematically, DR could be derived from the confusion matrix and denoted as:

$$DR_{class} = \frac{TP_{class}}{TP_{class} + FN_{class}}. \tag{2}$$

In general, when an IDS model has a high DR at an attack class, it implies the model performs great at recognizing this type of attack and not confusing them with the benign samples.

TNR is a benchmark of an IDS model's classification power on the benign samples. When an IDS model yields a high TNR, it suggests that the model performs well on isolating the benign samples from the attack samples. TNR is also derived from the confusion matrix. Mathematically, it could be denoted as:

$$TNR = \frac{TN_{BENIGN}}{TN_{BENIGN} + FP_{BENIGN}}. \quad (3)$$

Another performance benchmark, which is closely related to the TNR, is the False Alarm Rate (FAR). As its name implies, FAR gives the probability of a false alarm, which occurs when a benign sample is falsely classified as an attack, initiated by the IDS model. It is very important to evaluate a classifier's performance at FAR, especially in the cases of anomaly detections. A classifier for network intrusion detections is not trustworthy when it has a high DR for attacks but also a high FAR. In such case, the classifier will perform poorly with data from the real world. Given that the benign traffic is the majority in an ordinary network environment, a high FAR means the IDS can falsely warn the network admin on benign traffic very frequently, making the IDS not usable. In other words, for an IDS model, the higher the FAR, the lower the TNR. FAR could be denoted as:

$$FAR = \frac{FP_{BENIGN}}{TN_{BENIGN} + FP_{BENIGN}} = 1 - TNR. \quad (4)$$

Finally, the proposed IDS model's overall accuracy is also provided. The overall accuracy reflects a classifier's ability of classifying each sample into its true class. In the case of network intrusion detections, it is a comprehensive performance metric of an IDS's classification power at both benign and attack samples. It is defined to be the ratio of the total correct predictions made by the classifier to the total number of the input samples. The overall accuracy is calculated as follows:

$$Accuracy = \frac{TN_{overall} + TP_{overall}}{TN_{overall} + TP_{overall} + FN_{overall} + FP_{overall}}. \quad (5)$$

B. PERFORMANCE ON THE α-DATASET

The proposed IDS model is trained on the training set from the α-Dataset. The Adam optimizer [23] is used in the training process. The learning rate is set to 0.0002 for the first 50 epochs and decreased to its 1/10 for refine tuning. Some performance benchmarks of the trained model are shown in Table 4. In Table 4, DR and TNR of the proposed model on the training set and test set are presented. The proposed model reaches over 90% in DR at 12 attack classes out of the total 14 classes. In addition, at 10 attack classes, the proposed model successfully yields over 99% at DR. Also, TNR of the benign class reaches 98.984%. The proposed IDS model performs relatively poorly at the Botnet and Sql Injection class when compared to the other attack classes. This is due to the small size of in-class samples of the two classes. Botnet and Sql Injection only make proportions of 0.07% and 0.001%, respectively, in the original CICIDS2017 database. Accordingly, the proposed IDS model struggles to capture the characteristics of these two types of network traffic due to their insufficient samples. This also happened at some other IDS models using CICIDS2017 as the training database, as shown in Table 5.

TABLE 4. Performance of the Proposed Model on α -Dataset

Traffic Class	Label	TNR - Test Set	TNR - Training Set
		98.984%	99.044%
BENIGN (Mon)	BENIGN	**FAR - Test Set** 1.015%	**FAR - Training Set** 0.955%
Traffic Class	**Label**	**DR - Test Set**	**DR - Training Set**
BENIGN (Mon)	BENIGN	99.735%	99.975%
Brute Force	FTP-Patator	99.322%	99.542%
	SSH-Patator	66.378%	66.429%
Botnet	Bot	99.941%	99.974%
DDoS	DDoS	99.922%	99.941%
DoS	DoS GoldenEye	99.965%	99.975%
	DoS Hulk	99.631%	99.852%
	DoS Slowhttptest	99.651%	99.861%
	DoS slowloris	100%	100%
Heartbleed	Heartbleed	91.667%	91.667%
Infiltration	Infiltration	99.994%	99.996%
Port Scan	Port Scan	99.536%	99.536%
Web Attack	Web Attack - Brute Force	80.952%	80.952%
	Web Attack - Sql Injection	92.8%	93.42%
	Web Attack - XSS	99.735%	99.975%
Traffic Class	**Label**	**Accuracy - Test set**	**Accuracy-Training Set**
		99.64%	99.78%
Overall	N/A	**Test Time / Sample** 5.145e-5s	**Train Time / Sample** 1.884e-4s

C. PERFORMANCE COMPARISON

The proposed IDS model is also evaluated by comparing to some well-known classifiers. In Table 5, the class-wise DR and TNR of the following nine models: Hierarchical [17], WISARD [24], Forest PA [25], J48, LIBSVM [26], FURIA [27], Random Forest, MLP, and Naive Bayes are presented. In addition, the figures in Table 5 that are bolded represent the best performer in a class. For example, the 98.984% TNR yielded by the proposed IDS is bolded to indicate it has the highest TNR performance among the competitors.

Among the selected models, the proposed IDS model has the highest DR at 10 attack classes out of the total 14 classes. These 10 classes are the Bot, DDoS, DoS GoldenEye, DoS Hulk, DoS Slowhttptest, DoS slowloris, Heartbleed, Port Scan, Web Attack-Brute Force, and Web Attack-XSS. For the rest of the four attack classes, in terms of DR, the proposed model is ranked the second at FTP-Patator, the seventh at SSH-Patator, the third at Infiltration, and the fourth at Web Attack-Sql Injection. In addition, the proposed model has the

TABLE 5. Class-Wise Performance of the Proposed IDS Model and Other Classifiers

Label	Metrics	The Proposed Model	Hierarchical	WISARD	Forest PA	J48 Consol -idated	LIBSVM	FURIA	Random Forest	MLP	Naive Bayes
BENIGN	TNR	**98.984%**	98.860%	97.135%	96.450%	93.355%	94.870%	96.835%	98.120%	92.650%	66.545%
FTP-Patator	DR	99.735%	99.636%	0.000%	99.727%	**100%**	0.000%	99.636%	99.727%	99.000%	99.455%
SSH-Patator	DR	99.322%	99.909%	0.000%	**100%**	99.727%	0.000%	**100%**	99.818%	99.727%	99.182%
Bot	DR	**66.378%**	46.474%	1.442%	48.718%	52.083%	0.000%	48.077%	49.679%	51.282%	29.968%
DDoS	DR	**99.941%**	99.879%	54.697%	99.818%	93.212%	55.970%	99.758%	99.818%	91.212%	93.879%
DoS GoldenEye	DR	**99.922%**	67.571%	48.714%	67.571%	67.143%	57.571%	65.143%	67.571%	65.429%	62.143%
DoS Hulk	DR	**99.965%**	96.782%	67.600%	93.945%	95.891%	73.709%	98.655%	95.164%	86.891%	73.782%
DoS Slowhttptest	DR	**99.631%**	93.841%	23.353%	86.826%	83.832%	76.561%	78.358%	81.352%	88.537%	70.060%
DoS slowloris	DR	**99.651%**	97.758%	78.909%	92.848%	95.030%	78.182%	93.758%	93.758%	78.485%	82.667%
Heartbleed	DR	**100%**	**100%**	80.000%	**100%**	80.000%	0.000%	40.000%	100%	0.000%	80.000%
Infiltration	DR	91.667%	**100%**	50.000%	83.333%	**100%**	0.000%	83.333%	83.333%	16.667%	83.333%
Port Scan	DR	**99.994%**	99.881%	51.407%	99.594%	99.046%	48.521%	87.118%	99.881%	48.521%	99.499%
Web Attack-Brute Force	DR	**99.536%**	73.265%	4.694%	73.469%	55.102%	80.816%	49.796%	70.408%	90.408%	5.102%
Web Attack-Sql Injection	DR	80.952%	50.000%	0.000%	50.000%	**100%**	0.000%	50.000%	**100%**	50.000%	**100%**
Web Attack- XSS	DR	**92.8%**	30.625%	1.250%	34.375%	48.750%	0.000%	38.750%	37.500%	1.875%	91.875%

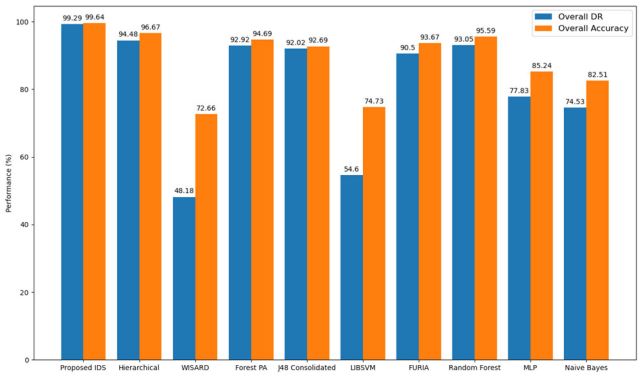


FIGURE 3. Overall detection rate and accuracy of the ten selected classifiers.

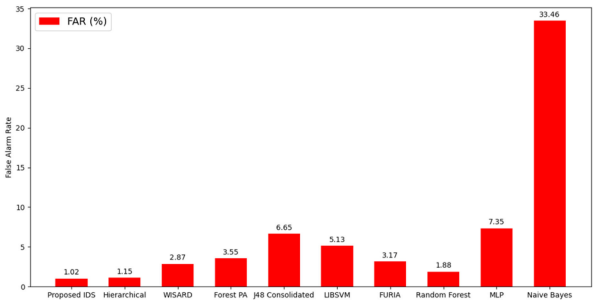


FIGURE 4. False alarm rate of the ten selected classifiers.

highest TNR for benign samples compared to the nine selected models.

This also means that this model has the lowest FAR, which can be calculated by (19). Accordingly, out of the total 15 traffic classes in CICIDS2017, the proposed model has the best classification performance at 11 classes when compared to the rest of the nine classifier models. Finally, in Figs. 3 and 4, the overall attack DR, FAR and accuracy of the ten selected classifiers are presented to provide a visually intuitive performance comparison. From the figures, it is easily observable that the proposed model has the highest overall DR and accuracy and the lowest FAR within the ten models.

In addition, it has a minimal performance gap between the overall DR and accuracy. Noted that the overall DR is calculated by the ratio of the attack samples that are not classified as benign by the model. And the overall accuracy is calculated based on the ratio of the samples that are correctly classified into its true case. Therefore, the small gap between overall DR and accuracy indicates that the proposed model is not only highly capable of detecting cyberattacks, but also precisely classifies the attacks into their true class.

In addition, the proposed IDS is compared with two other CNN based IDS that are recently published. The compared models have been introduced in the literature review chapter [8], [9]. Both studies perform intrusion detections at the CICIDS dataset with a similar CNN architecture yet different data pre-processing methods. In [8], the raw CICIDS2017 dataset, which included separate sub-datasets of the network traffic in a day, was used to train the models. The authors trained and evaluated a CNN based IDS based on each of the sub-dataset, which mostly included only 1∼3 attack classes. In [9], the authors manually separated the CICIDS2018 dataset into ten sub-datasets, and each sub-dataset included up to three attack classes. Therefore, the CNN based IDS in [8], [9] both performed one-class or a subset of class

TABLE 6. Class-Wise Performance of Other CNN Models on CICIDS2017

Label	Metrics	Proposed Model	CNN -1 [8]	CNN -2 [9]
BENIGN	TNR	98.984%	97.5%	97.1%
FTP-Patator	DR	99.735%	0%	98%
SSH-Patator	DR	99.322%	0%	96%
Bot	DR	66.378%	0%	100%
DDoS	DR	99.941%	100%	100%
DoS GoldenEye	DR	99.922%	0%	47%
DoS Hulk	DR	99.965%	65%	100%
DoS Slowhttptest	DR	99.631%	0%	100%
DoS slowloris	DR	99.651%	100%	66%
Heartbleed	DR	100%	0%	N/A
Infiltration	DR	91.667%	0%	35%
Port Scan	DR	99.994%	100%	N/A
Web Attack-Brute Force	DR	99.536%	N/A	30%
Web Attack-Sql Injection	DR	80.952%	N/A	8%
Web Attack- XSS	DR	92.8%	N/A	65%

classification, and the model's accuracy, DR, and TNR were provided. In Table 6, the DR and TNR of the proposed IDS and the compared CNN based IDS are provided. The proposed IDS is superior to the other models in many aspects. First, in the comparison, the proposed IDS is the only model performing true multi-class classification on all the 15 classes that are available in the CICIDS dataset. The other two CNN based IDS, while performing well on binary classification on one attack class, have poor performance if there are many attack classes need to be classified. Second, in term of class-wise classification performance, the proposed IDS also has the highest TNR, and a higher or comparable DR at each attack class. In summary, the proposed IDS, thanks to the innovative dataset pre-processing method, not only can classify more classes at once with one CNN classifier but also yields a competitive benchmark at each class-wise classification category compared with other previously proposed CNN based IDS.

D. PERFORMANCE ON INNOVATIVE ATTACKS

To test how the proposed model performs on innovative attacks, the data from CICIDS2017 is used to design some custom training and testing datasets for simulation. More specifically, the DoS attacks is selected to be the test subject due to its diversity.

CICIDS2017 has four DoS varieties, namely, the DoS GoldenEye, DoS Hulk, DoS Slowhttptest and DoS slowloris. During the simulation, four IDS models are generated, each has the same architecture of the proposed model, with four combinations of training and test dataset. Each time a model is trained, only one type of the DoS samples are included in the training dataset to represent the known attacks. Meanwhile, the rest of the three DoS varieties are only used in the test dataset to represent the innovative attacks. In this scenario, since the model never encounters the innovative DoS samples in the test dataset, some traditional signature-based IDS models may fail easily due to the lack of signature information in the database. On the other hand, the proposed model has the potential to recognize these innovative DoS samples.

TABLE 7. Performance of the Proposed IDS Model At Innovative DoS Attacks

Training Dataset Composition	DoS GoldenEye	✓	×	×	×
	DoS Hulk	×	✓	×	×
	DoS Slowhttptest	×	×	✓	×
	DoS slowloris	×	×	×	✓
Test Dataset DR	DoS GoldenEye	N/A	57.80%	5.916%	0.087%
	DoS Hulk	87.29%	N/A	39.29%	0.197%
	DoS Slowhttptest	9.273%	0.036%	N/A	85.73%
	DoS slowloris	3.795%	0.104%	38.94%	N/A

The proposed IDS model, which is a neural network-based model, captures the generalized characteristics of each type of cyberattacks. Therefore, if some innovative DoS samples share similar traits with the DoS samples in the training dataset, they might be successfully classified as the DoS class by the model.

The four models are trained separately and their DR on the innovative DoS attacks in the test dataset is recorded. The simulation results of all the four models are presented in Table 7. From Table 7, one can easily observe that when only the DoS GoldenEye samples are used in the model's training process, the model has the potential to also detect DoS Hulk at a decent 87.29% DR.

Besides, when the model is only trained on the DoS slowloris samples, it has the potential to also detect DoS Slowhttptest at an 85.73% DR. These simulation results confirm that the proposed model is superior to the traditional IDS at detecting innovative attacks.

Thus, the scientific significance of the proposed IDS model could be summarized into three categories, namely, the original design of the training dataset, the model's outstanding 99.78% accuracy, and its potential to detect innovative attacks.

V. CONCLUSION

In this paper, an Intrusion Detection System (IDS) for cybersecurity based on a Convolutional Neural Network (CNN) classifier is proposed. The convolution and pooling process of CNN allow the proposed IDS model to learn complicated patterns of features form network traffic, while maintaining reasonable storage and computation overhead. This feature differentiates the proposed IDS model with the traditional one, which requires a signature database that is usually handmade

by security experts to perform classification. Also, an innovative dataset preprocessing method is used to boost the multi-class classification performance of the proposed CNN based IDS. In our evaluation using CICIDS2017 security database, the proposed IDS model outperforms nine other well-known classifier models in most multi-class classification categories. More specifically, out of the 14 available attack classes in CICIDS2017, the proposed model reaches the highest Detection Rate (DR) at 10 attack classes compared to the nine selected models. This IDS model also has the highest True Negative Rate (TNR) and the lowest False Alarm Rate (FAR) for benign network traffic. Also, the proposed IDS provides better multi-class classification results than some other recently proposed CNN based IDS, which only capable of only one-class classification or classifying a few classes. In addition, it shows the potential at detecting innovative attacks, which the traditional IDS models struggle to perform. Some custom datasets were used to evaluate this model's performance at innovative DoS attacks. The simulation results indicate that the proposed model is capable of detecting some DoS samples that the model never encounters in its training process.

The proposed IDS reaches a new altitude at detecting cyberattacks, which the traditional IDS models never achieve. This model not only offers protection against the known attacks with a high DR and low FAR, but also shows potential at detecting innovative attacks. However, we also believe the proposed model has not yet arrived at perfection. There are many ways one could extend this research in the future. The class imbalance issues of the training dataset may be solved by more automated methods instead of the manually tuned method that is used in this research to prevent biased classification decisions. Besides, some sample augmenting techniques that are used in the CNN models for picture and video detection may be adapted to solve the issue of insufficient samples of minority attacks. Finally, the proposed IDS model's accuracy on innovative cyberattacks still has room for improvement. Given the constant momentum at the development of the neural network discipline, it is believed that some new revisions to the existing neural network models may be proposed in the future to further facilitate the detection of innovative attacks.

REFERENCES

[1] J. P. Anderson, *Computer Security Threat Monitoring and Surveillance*. Washington, PA, USA: James P. Anderson, 1980.

[2] W. Lee *et al.*, "Real time data mining-based intrusion detection," in *Proc. DARPA Inf. Survivability Conf. Expo. II.*, 2001, pp. 89–100.

[3] V. V. R. P. V. Jyothsna, V. V. Rama Prasad, and K. Munivara Prasad, "A review of anomaly based intrusion detection systems," *Int. J. Comput. Appl.*, vol. 28, no. 7, pp. 26–35, 2011.

[4] G. Ciaburro and B. Venkateswaran. *Neural Networks With R: Smart Models Using CNN, RNN, Deep Learning, and Artificial Intelligence Principles*. Birmingham, U.K.: Packt Publishing, 2017.

[5] J. Kiefer and J. Wolfowitz, "Stochastic estimation of the maximum of a regression function," *Ann. Math. Statist.*, vol. 23, no. 3, pp. 462–466, 1952.

[6] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *Siam Rev.*, vol. 60, no. 2, pp. 223–311, 2018.

[7] 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.*, Las Vegas, NV, USA, 2019, pp. 0452–0457.

[8] S. Yeom and K. Kim, "Detail analysis on machine learning based malicious network traffic classification." in *Proc. Int. Conf. Smart Media Appl.*, 2019, pp. 49–53.

[9] J. Kim, Y. Shin, and E. Choi, "An intrusion detection model based on a convolutional neural network," *J. Multimedia Inf. System*, vol. 6, no. 4, pp. 165–172, 2019.

[10] S. Potluri, S. Ahmed, and C. Diedrich, "Convolutional neural networks for multi-class intrusion detection system," in *Proc. Int. Conf. Mining Intell. Knowl. Exploration*, 2018, pp. 225–238.

[11] N. Moustafa and J. Slay, "The UNSW-NB15 data set description," 2015, Accessed: Apr. 6, 2018. [Online]. Available: https://www.unsw.adfa.edu.au/unsw-canberra-cyber/ cybersecurity/ADFA-NB15-Datasets/

[12] A. Khraisat *et al.*, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 20, 2019.

[13] Y. Gu, K. Li, Z. Guo, and Y. Wang, "Semi-supervised K-means DDoS detection method using hybrid feature selection algorithm," *IEEE Access*, vol. 7, pp. 64351–64365, 2019.

[14] O. Faker and E. Dogdu, "Intrusion detection using big data and deep learning techniques," in *Proc. ACM Southeast Conf.*, 2019, pp. 86–93.

[15] J. Hühn and E. Hüllermeier, "FURIA: An algorithm for unordered fuzzy rule induction," *Data Mining Knowl. Discov.*, vol. 19, no. 3, pp. 293–319, 2009.

[16] K. Atefi, H. Hashim, and M. Kassim, "Anomaly analysis for the classification purpose of intrusion detection system with K-nearest neighbors and deep neural network," in *Proc. IEEE 7th Conf. Syst., Process Control*, Dec. 2019, pp. 269–274.

[17] A. Ahmim *et al.*, "A novel hierarchical intrusion detection system based on decision tree and rules-based models," in *Proc. IEEE 15th Int. Conf. Distribution Comput. Sensor Syst.*, 2019, pp. 228–233.

[18] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy*, 2018, pp. 108–116.

[19] R. Panigrahi and S. Borah, "A detailed analysis of CICIDS2017 dataset for designing intrusion detection systems," *Int. J. Eng. Technol.*, vol. 7. no. 24, pp. 479–482, 2018.

[20] V. Sze *et al.*, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[21] A. Paszke *et al.*, "Automatic differentiation in pytorch," in *Proc. 31st Conf. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2017, pp. 1–4.

[22] R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Applying convolutional neural network for network intrusion detection," in *Proc. Int. Conf. Adv. Comput., Commun. Inform.*, 2017, pp. 1222–1228.

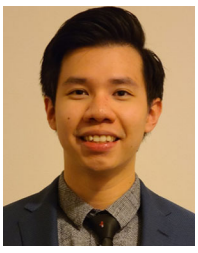
[23] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, San Diego, CA, USA, May 2015, pp. 1–15.

[24] M. D. Gregorio and M. Giordano, "An experimental evaluation of weightless neural networks for multi-class classification," *Appl. Soft Comput.*, vol. 72, pp. 338–354, 2018.

[25] M. N. Adnan and M. Z. Islam, "Forest PA: Constructing a decision forest by penalizing attributes used in previous trees," *Expert Syst. Appl.*, vol. 89, pp. 389–403, 2017.

[26] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, 2011.

[27] Z. Xueqin, C. Jiahao, Z. Yue, L. Han, and L. Jiajun, "A multiple-layer representation learning model for network-based attack detection," *IEEE Access*, vol. 7, pp. 91992–92008, 2019.



SAMSON HO received the B.S. degree in photonic electronics engineering from Yuan Zi University, Taiwan. He is currently working toward the graduate degree with Electrical Engineering Department, California State University, Long Beach, CA, USA**.** He is currently a Graduate Research Student with the Networked Embedded Systems Laboratory, California State University. His research interests include deep learning and its applications.

SALEH AL JUFOUT received the B.S., M.S. and Ph.D. degrees in electrical power engineering from Donetsk State Technical University, Donetsk Oblast, Ukraine, in 1993 and 1997, respectively. He is currently a Professor of electrical power engineering. From 1997 to 2005, he was a Faculty Member with Al-Balqa Applied University, Salt, Jordan. From 2005 to 2019, he was a Professor with the Electrical Power Engineering and Mechatronics Department, Tafila Technical University, At-Tafilah, Jordan. Since January 2020, he has been with Electrical Engineering Department, California State University, Long Beach, CA, USA. He is the author of more than 70 articles. His research interests include the simulation of transients in electrical power systems, protection system design, and modeling of electrical systems and machines. He is the Founder Editor-in-Chief of the *Jordan Journal of Electrical Engineering*. He was the recipient of the Erasmus Mundus Postdoctoral Scholarship in 2015 at the Technical University of Berlin, Berlin, Germany, and the Fulbright Postdoctoral Scholarship in 2016 at Wayne State University, Detroit, MI, USA.

KHALIL DAJANI is the Executive Director of te California Aerospace Technologies Institute of Excellence, Palmdale, CA, USA. He is currently the United States Edwards Air Force Base Honorary Commander with Edwards Air Force Base, Kern County, CA, USA, andthe Executive Manager with the Research Foundation, California State University, Long Beach, CA, USA. He held several administrative positions with academic institutions across the United States. He was also the Executive Director of the Antelope Valley Engineering Programs, Satellite Campus, California State University. He was the Chair, a Professor, and the Director of Graduate Programs with the College of Science and Engineering, Southern Arkansas University, Magnolia, Arkansas. He has authored or coauthored more than 80 articles in scientific journals and conference proceedings. He participated in multimillion-dollar grants from the National Science Foundation, NASA, Department of Defense, and other federal and state agencies. He has also chaired technical and engineering conferences at the state and national levels.

MOHAMMAD MOZUMDAR received the Ph.D. degree in electronics and communication engineering from Politecnico di Torino, Turin, Italy. He is currently an Associate Professor with Electrical Engineering Department, California State University, Long Beach, CA, USA, and an Ex-Postdoc with the University of California, Berkeley, CA, USA. His novel ideas of model-based design for sensor networks made profound impact on engineering and industrial communities. He has authored or coauthored in a book chapter, renowned journals, reputed conference proceedings, major scientific magazines, and also translated in several different languages. His current research interests include methodologies and tools for embedded system especially in the domain of sensor networks, energy-efficient building information and control system design, cloud computing, cyber physical system, and methodology for the design of distributed embedded systems typically subjected to high real time, safety, and reliability constraint.