IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 21, 2021, accepted November 13, 2021, date of publication November 16, 2021,
date of current version November 24, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3128701

A Hybrid Deep Learning Approach for Replay and DDoS Attack Detection in a Smart City

ASMAA A. ELSAEIDY[1], (Student Member, IEEE), ABBAS JAMALIPOUR[2], (Fellow, IEEE), AND KUMUDU S. MUNASINGHE[1], (Member, IEEE)

[1]Faculty of Science and Technology, University of Canberra, Bruce, ACT 2617, Australia
[2]School of Electrical and Information Engineering, The University of Sydney, Camperdown, NSW 2006, Australia

Corresponding author: Asmaa A. Elsaeidy (asmaa.elsaeidy@canberra.edu.au)

ABSTRACT Today's smart city infrastructure is predominantly dependant on Internet of Things (IoT) technologies. IoT technology essentially facilitates a platform for service automation through connections of heterogeneous objects via the Internet backbone. However, the security issues associated with IoT networks make smart city infrastructure vulnerable to cyber-attacks. For example, Distributed Denial of Service (DDoS) attack violates the authorization conditions in smart city infrastructure; whereas replay attack violates the authentication conditions in smart city infrastructure. Both attacks lead to physical disruption to smart city infrastructure, which may even lead to financial loss and/or loss of human lives. In this paper, a hybrid deep learning model is developed for detecting replay and DDoS attacks in a real life smart city platform. The performance of the proposed hybrid model is evaluated using real life smart city datasets (environmental, smart river and smart soil), where DDoS and replay attacks were simulated. The proposed model reported high accuracy rates: 98.37% for the environmental dataset, 98.13% for the smart river dataset, and 99.51% for the smart soil dataset. The results demonstrated an improved performance of the proposed model over other machine learning and deep learning models from the literature.

INDEX TERMS Intrusion detection, distributed denial of service (DDoS) attacks, replay attack, smart city, deep learning, Internet of Things (IoT).

I. INTRODUCTION

The Internet of Things (IoT) is based on the concept of connecting any device to the Internet [1]. This sort of technology has led to the creation of smart cities, in which basic infrastructure components, such as electricity health, traffic and water resources are monitored and controlled through the Internet [2], [3]. The integrity, availability and consistency of the smart city data have the potential to affect the lives of the citizens [4]. For example, the data collected from sensors about the water level in a river during heavy rains could help save lives from river flooding. The collection of such large quantities of data requires many devices to be connected to the Internet. However, this opens the backdoor for illegitimate users to threaten lives and damage the infrastructure.

Smart city security is important for the authentication and authorization conditions, which guarantee data consistency, availability and integrity. Cyber security's main aim is to secure cyber-space from cyber-attacks that could lead to network damage or service unavailability [5], [6]. Cyber-attacks threaten the ability of smart cities to supply consistent, trusted and timely services to their citizens. The huge amount of exchanged data in IoT applications increases the possibility of cyber-attacks. This threatens citizens' privacy, information integrity, confidentiality and service availability. The major problem affecting smart city security appears to be the lack of common security standards across all the organizations involved [7]. This allows the illegitimate user to interfere with the collected data in the server by cutting off or changing the service from the end user [8].

Authentication condition is represented in the consistency and integrity of the uploaded data on the server. Replay attack aims to change this data, which will violate the authentication conditions [9], [10]. This results in the wrong information being sent and leads to confusion and major damage. It happened in Texas, in 2010, when workers received false information about where to dig a hole for gas and water pipes. As a result, there was a 36 inch gas pipeline explosion and fire, where many people were killed. Floods, fires, deaths and

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

more can all happen as a result of malicious interference with the data being fed into the smart city applications [11], [12].

Authorization or access control is defined as the mechanism used to differentiate legitimate users from illegitimate users [13]. There are some attacks that target authorization conditions such as Distributed Denial of Service (DDoS) attack [14]. DDoS is the most popular attack to threaten the authorization of smart city applications [15]. DDoS attacks obscure services from end users through set of distributed agents created by attackers. DDoS attack takes many forms, for example Smurf, HTTP flood, UDP flood, SIDDOS and SYN flood [16]. The attacker attempts to send large packets of data from different sources, called zombies, to offload servers rather than to legitimate users, resulting in a termination of the service [17].

Recently, machine learning, specifically deep learning, has been shown to be very effective at detecting cyber-attacks in smart cities [18], [19]. Although simulating cyber-attacks on a real smart city dataset to use it as a benchmark is not a trivial task, our proposed model is developed to detect intrusions using a real dataset with complicated distributions. The proposed model is a hybrid deep learning model, which combines a deep restricted Boltzmann machine (RBM) model with a deep convolutional neural network (CNN) model. The RBM part of the proposed model plays an important part in learning high level features from the dataset to provide much better representation of the dataset. The RBM model has the ability to overcome the small number of input features and to model the underlying dataset distribution without the need for the associated classes. The deep CNN part of the proposed model is then trained in supervised mode derived by the associated classes. CNN is not only performing the classification task, but it is also learning the local invariance filters that detect local features from input signals.

The dataset used to evaluate the proposed model is generated by simulating DDoS and replay attacks over normal generated data from a real smart city platform in the city of Queanbeyan, Australia. This platform deployed different monitoring nodes for soil management, river monitoring and environmental traffic monitoring. For each available service node normal dataset, replay and DDoS attacks are simulated using defined models. The performance of the proposed hybrid model is compared with other machine and deep learning models from the literature. The reported results showed the ability of the proposed hybrid model to provide high accuracy rates and outperform the other models.

The main contributions of the work introduced in this paper can be summarized as below:

- The application of deep learning models for detecting replay and DDoS attacks in smart city.
- The experimental evaluation is conducted on a real life smart city dataset.
- Proposed a deep hyper model to improve attack detection accuracy.
- Handle the low number of features introduced in the datasets.
- Consider the time factor in the detection process.

The remainder of this paper is organized as follows: Section II discusses the related work. Section III describes the proposed hybrid model structure and the training procedure. Section V describes the attack models used to generate synthetic DDoS and replay attack datasets, and explains the experimental setups and deep models used for the comparative evaluation. Section VI explains the results and discussion. Section VII concludes the paper and highlights future directions.

II. RELATED WORK

Intrusion detection systems (IDSs) have a big attraction in securing smart cities, especially those using machine learning techniques. There are some surveys reporting the importance of applying machine learning approaches in building IDS models [20]–[22]. Deep learning approaches have proved their distinction over other traditional machine learning techniques at detecting cyber-attacks with high accuracy [23].

An application of deep learning in intrusion detection is the work of [24] and [25]. These works proposed a distributed deep learning schema for cyber-attack detection in fog-to-things computing. The deep learning parameters are initialized on the main node and sent to the worker nodes. Each worker node performs the training data and hyper-parameters optimization locally and aggregates its parameters to the master node. The performance of this schema is evaluated using KDD-Cup'99 intrusion dataset. This approach is similar to our proposed model regarding feature learning. However, it is based on a distributed learning approach, while ours is using a centralized learning approach with one model instance. Another relevant work is the application of self-taught deep learning for attack detection [26]. UNSW-NB15 and NSL-KDD datasets are used for evaluating the proposed model. The results showed that the model is working more efficiently for the NSL-KDD dataset.

A hybrid schema that combines deep belief network and support vector machine has been applied for intrusion detection in [27]. The deep belief network is built by training two RBM models acting as a preprocessing phase for the support vector machine classifier. The NSL-KDD dataset is used to evaluate the performance of the proposed hybrid schema. Another research is done in [28], by proposing an unsupervised learning approach and a deep learning classification model based on stacked non-symmetric autoencoders. The proposed approach is evaluated by using the NSL-KDD and KDD-Cup'99 datasets. Another work that utilized a deep learning approach is applied in [17], by combining deep RBM model with feed forward neural network for attack detection. The deep RBM model is trained in unsupervised way to learn high level features as the new representation of the original dataset features. This new representation is then used by the classifier model for attack detection. The performance of the

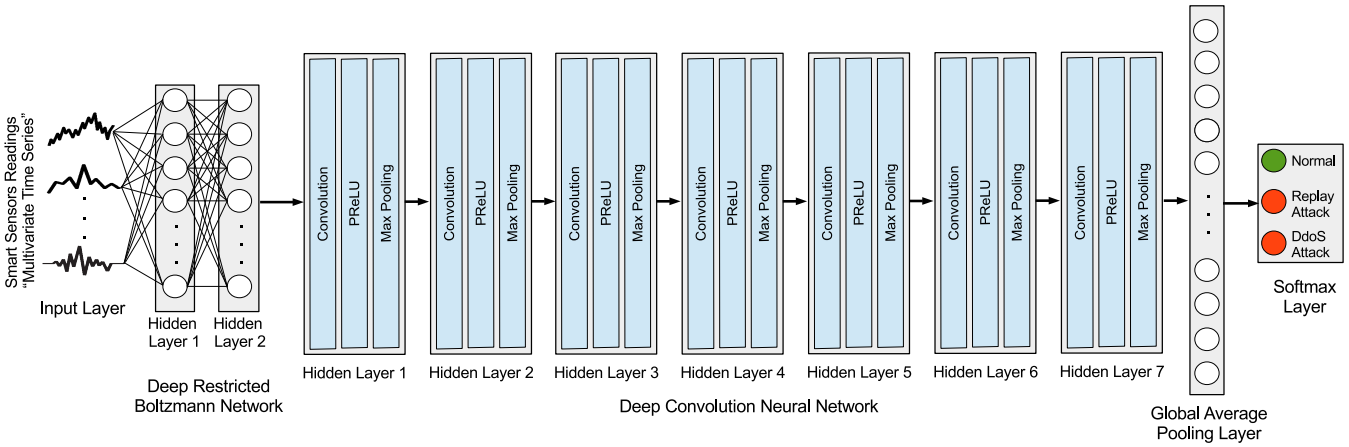

FIGURE 1. The hybrid deep learning proposed methodology for replay and DDoS attacks detection.

proposed model is evaluated using a dataset generated from a smart water distribution system.

Deep convolutional neural network model for DDoS attack detection is proposed in [29]. The dataset used for evaluating the performance of the proposed model is generated from 4G LTE-A architecture. These previous works are similar to our approach in learning high level features and use these features to train the classifier part. However, our model considers the time dependencies in the dataset. Another approach for intrusion detection in IoT architecture is proposed in [30]. A deep belief network is applied for intrusion detection, where a genetic algorithm is utilized to find an optimal network structure. The proposed model performance is evaluated using NSL-KDD dataset. In our proposed model, we used a grid method approach to find the optimal hyper-parameters. For the network structure, an incremental approach is used by keep adding more layers until no further improvement is detected in the model overall performance.

Time dependencies introduced in smart city datasets regarding the readings of smart sensors over time are considered in [12], where an intrusion detection model is applied for replay attack detection. The performance of the proposed model is evaluated on replay attack datasets that were simulated based on real smart city infrastructure. Another work that considers the time dependencies for intrusion detection is proposed in [31]. In this work a deep Long Short Term Memory (LSTM) network is proposed for DDoS attack detection. The deep LSTM network is trained to learn high level features in time domain. The deep LSTM model is trained using the backpropagation algorithm. On top of learned features, Gaussian Naive Bayes model is applied for attack detection step. The performance of the proposed model is evaluated using artificially generated normal and attack dataset. A hybrid schema for intrusion detection in smart metal packaging plant is proposed in [32]. This schema consists of three components: convolutional LSTM encoding layers, bidirectional stacked LSTM decoding layers, and time-distributed supervised learning fully connected layer. These works are similar to our proposed model in considering the time dependencies in the dataset. However, our model adds an additional step before training the CNN model by using the RBM model to learn high level features. This step is beneficial in handling small number of input features.

TABLE 1. Common notations used throughout this study.

Symbol	Definition
GAP	Global Average Pooling
N	RBM model input layer visible units number
M	Number of epochs
K	Number of batches
V	Visible layer
H	Hidden layer
$\mathcal{N}(\mu, \sigma^2)$	Normal distribution
δ	Learning rate
σ	Sigmoid activation function
$<>_{data}$	Data samples expectations
$<>_{recon}$	Reconstruction error expectations
W	Weights matrix
$p(\|)$	Conditional probability
Acc	Accuracy measure

III. THE PROPOSED HYBRID DEEP LEARNING NETWORK

In this section, the proposed hybrid deep learning model for replay and DDoS attacks detection is described. A flowchart of the proposed detection model in the context of smart city is shown in Fig. 2. The proposed model consists of an input layer, deep RBM model with two hidden layers, deep CNN with seven hidden layers, a global average pooling (GAP) layer, and a softmax output layer, as shown in Fig. 1. In the next subsections, each step of the proposed hybrid model is explained in detail.

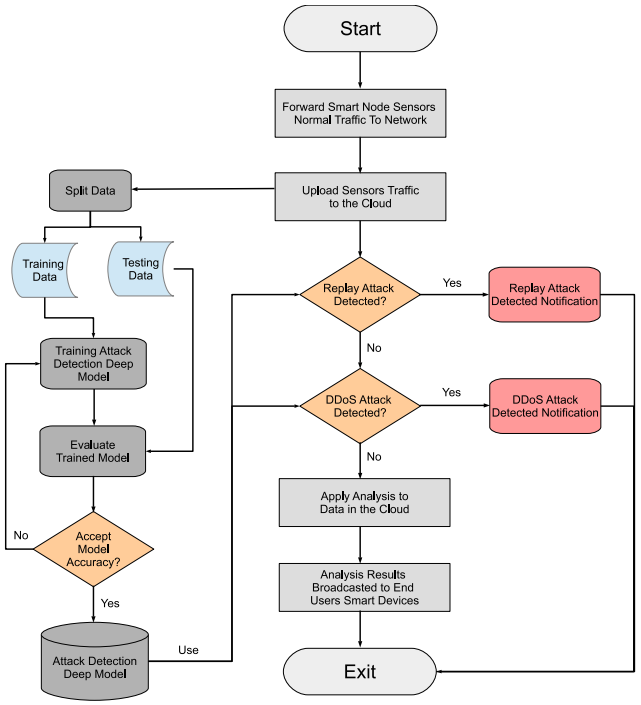

FIGURE 2. Flow chart of the proposed replay and DDoS attack detection deep model.

A. NETWORK ARCHITECTURE

1) INPUT LAYER

In smart cities, data is collected using smart meters and sensors, which provide regular readings over time. The datasets that will be used to evaluate the proposed model contain time stamped data instances. Therefore, the type of input features is multivariate time series. The input layer has N visible units equal to the number of features. The smart sensor readings are represented in real values, making Gaussian visible units in the input layer of RBM model a suitable selection for modelling the dataset. These Gaussian visible units in the input layer are connected to the hidden units in the deep restricted Boltzmann network.

2) DEEP RESTRICTED BOLTZMANN NETWORK

Each smart dataset (Section IV) has a small number of features, which could affect the classifier accuracy rate [33]. Deep RBM model is used to learn high level features from raw features in an unsupervised way. The new learned features are used as the new representation to the original datasets. In addition to the advantage of increasing the number of features, the learned features contain rich and valuable patterns. As shown in Fig. 1, the deep RBM consists of two hidden layers. The selection of two hidden layers is based on an incremental approach. This is accomplished by keep adding more layers until no further improvement is detected. Each RBM model in the deep RBM part of the proposed model consists of one visible layer and one hidden layer fully connected to each other. The first RBM model uses Gaussian visible units and stochastic binary units [34]. The second RBM model uses stochastic binary units in both visible and hidden layers. Each RBM model in the deep RBM model is trained in isolation using contrastive divergence (CD) algorithm [35]. Finally, all learned models are stacked on top of each other to form the deep RBM model [36].

3) DEEP CONVOLUTIONAL NEURAL NETWORK

The learned features by the deep RBM model are used as the new representation of the datasets. Given this new representation, a deep CNN is trained to model the time series data and to classify data instances to distinguish between normal behaviour, replay attack and DDoS attack. Time series classification is a complex task, which could be hard to model by typical machine learning approaches. CNN model is selected due to its ability to model both two-dimensional (images) data and one-dimensional (time-series) data. The deep CNN model consists of seven hidden layers. The selection of seven hidden layers is based on an incremental approach. This is accomplished by keep adding more layers until no further improvement is detected. Each hidden layer applies three subsequent operations: convolution operation that applies number of one-dimensional filters to the input signals to produce one-dimensional feature maps [37]; parametric rectified linear unit activation function [38]; and max pooling operation that introduces invariance to small disturbances in the activation result [39].

4) OUTPUT LAYER

After applying the convolution, activation and max pooling operations, the GAP layer is applied [40]. This layer serves two main purposes: first, it flattens the feature maps into a final features layer that is fully connected to the classification layer. Second, it acts as a structural regularizer that contributes to the overall training process by reducing the overfitting effect. The final part is the classification layer, where the attack detection is happening. This layer is a softmax layer with three units. One for the normal class and the other two for replay and DDoS attack classes. Softmax layer is preferred for multi-class tasks. It assigns a probability for each class, where the total probability over all classes equals one [41].

B. NETWORK TRAINING

Algorithm 1 describes the training procedure for the proposed hybrid model. The inputs are the dataset with its associated classes, an initial RBM model with one hidden layer, an initial CNN model with one hidden layer, maximum number of epochs and number of batches to divide the dataset. Initially, we fixed the CNN model with one hidden layer in order to evaluate the RBM model until we determined the best number of hidden layers to use. The initial RBM model is trained for a maximum number of epochs and batches, then used to create a new data representation. This new data representation is used to train the initial CNN model with one hidden layer to calculate the classification accuracy. If the

Algorithm 1 The Hybrid Deep Learning Proposed Methodology Training Procedure

Input: $RBM(V_1, H_1)$, $CNN(H_1)$, training data D, training targets T, number of input features N, number of epochs M, number of batches K

Output: trained RBM and CNN models

init RBM weights $W_{RBM} \leftarrow \mathcal{N}(\mu, \sigma^2)$
init RBM model visible units bias $b_{RBM} \leftarrow 0$
init RBM hidden units bias $c_{RBM} \leftarrow 0$
init overall model accuracy $Acc \leftarrow 0$
repeat
  for $i = 1$ to M **do**
    for $j = 1$ to K **do**
      load batch B_j from D
      $p(h = 1|v) = \sigma(b + vW)$
      $p(v|h) = \mathcal{N}(hW + c, \sigma^2)$
      $\delta W = \epsilon(< vh >_{data} - < vh >_{recon})$
      $W \leftarrow W + \delta W$
    end for
  end for
  create new data representation $D \leftarrow W_{RBM} D$
  train $CNN(H_1)$ using D and T
  evaluate $CNN(H_1)$ accuracy Acc_{CNN}
  $Acc \leftarrow Acc_{CNN}$
  add hidden layer to RBM model $RBM(V_1, H_1) \leftarrow H$
until $Acc_{CNN} \leq Acc$
use trained RBM model $RBM(V_1, H_1, \ldots H_k)$ to create new data representation D_{new}
repeat
  add hidden layer to CNN model $CNN(H1) \leftarrow H$
  evaluate CNN accuracy Acc_{CNN}
  $Acc \leftarrow Acc_{CNN}$
until $Acc_{CNN} \leq Acc$
return hybrid-model $\leftarrow$ $RBM(V_1, H_1 \ldots, H_k) + CNN(H_1, \ldots H_l), Acc$

classification accuracy is getting better, an additional hidden layer is added to the RBM model. Given this new updated RBM model, the training process for RBM is repeated until no further enhancement is introduced to the classification accuracy. The next part of the training procedure is to build the CNN model toward enhancing the overall classification accuracy. The CNN model is trained using the new data representation from the latest RBM deep model. The same incremental approach is used to add more hidden layers to the CNN model, while the classification accuracy is enhancing. When no further improvement in classification accuracy, the training is stopped and the trained hybrid RBM+CNN model and its overall accuracy are recorded.

IV. SMART CITY PLATFORM DATASET

In this section, the generation of the replay and DDoS attacks datasets is described. The attack datasets are public [42], which generated based on real smart city platform normal data for the Queanbeyan city [12]. To generate replay and DDoS attacks based on the normal data, we mimicked the behaviour of replay and DDoS attacks. Replay attack violates data authentication by generating misleading data based on the normal behaviour. DDoS attack violates authorization by sending large streams of data to overload the end service provider resulting in unavailability of these services.

The smart city platform that provides the normal datasets contains three nodes, soil management, environmental monitoring and river monitoring. A summary of these datasets statistics is provided in Table 2. The smart soil management platform is being designed and installed to provide information to guide the management of the city's irrigation system. It has five features: soil temperature, soil moisture at 30 cm depth, soil moisture at 60cm depth, battery levels and the number of packets that have reached the gateway successfully. The environmental monitoring node is set up to assess the impact of traffic on the city environment. It has five features: noise amplitude, air temperature, humidity, air pressure, and CO_2 levels. The river monitoring node continuously monitors number of water quality parameters including electrical conductivity, water acidity (pH), water temperature, and turbidity. It has five features: water acidity, dissolved O_2 concentration, conductivity, turbidity, and battery level. For each normal dataset generated from each service node, replay and DDoS attack are generated using Algorithms 2 and 3, respectively.

TABLE 2. Smart city platform datasets summary. Date-time stamps format is (dd/mm/yyyy - hh:mm) with readings per minute. Training and testing sizes are based on 80%-20% split.

	Environmental	River	Soil
Start date/time	03/09/2019 - 11:26	27/02/2020 - 00:00	09/07/2018 - 00:01
End date/time	04/03/2020 - 12:23	04/03/2020 - 12:29	01/05/2019 - 00:04
Number of features	5	5	5
Total records	266211	10961	126229
Normal records	94676	5358	49851
Replay records	83535	2592	37378
DDoS records	88000	3011	39000
Training size	180969	8769	100983
Testing size	85242	2192	25246

The replay attack model (Algorithm 2) assumes that the attacker already knows the normal dataset since this type of attack violates the authentication security component. The model also assumes that the attacker has the access to all dataset features. Given the normal dataset, the start and end dates are identified, and the probability distributions are estimated for each feature of the normal data. The replay attacker goes through all timestamps from start to end to add misleading data. For each timestamp, the attacker will add an input instance for all features if this timestamp does not exist in the normal dataset. This instance field is generated from the estimated probability distribution for each feature.

The DDoS attack model (Algorithm 3) assumes the attacker does not have access to the normal data. The attacker tries to violate the authorization security components by

Algorithm 2 Replay Attack Synthetic Data Generation Procedure

Input: Normal dataset D, number of features n start datetime T_1, end datetime T_m

Output: Dataset D_{synth} with replay attacks added

Estimate probability distributions for each feature $P(X_1, X2, \ldots, X_n)$ from normal dataset D Generate datetime list L from T_1 to T_m
for $T \in L$ **do**
  if $T \in D$ **then**
    Continue
  else
    Generate features vector V estimated from P
    Add V to D with time stamp T and class label "Replay"
  end if
end for
return D_{synth}

Algorithm 3 DDoS Attack Synthetic Data Generation Procedure

Input: Normal dataset D, number of features n, size of attack fields k for each datetime stamp

Output: Dataset D_{synth} with DDoS attacks added

Generate random datetime list L with size m
for $T \in L$ **do**
  Generate random feature vector V
  for i = 1 to k **do**
    Add V to D with time stamp T and class label "DDoS"
  end for
end for

overflowing the end service with a lot of sent readings. The attacker selects a range of random timestamps to send for each number of consecutive sensor readings. The generated values for feature vectors are based on random values since the attacker does not have a knowledge of the normal dataset [43].

The soil dataset features are drawn from normal probability distribution. The environmental dataset has three of its features drawn from generalized extreme value distribution, whereas the rest are drawn from normal distribution. The river dataset has all its features drawn from generalized extreme value and Weibull minimum extreme value probability distributions. The generated synthetic datasets are normalized to have zero mean and unit variance. Each dataset is divided into training and testing parts using 80%-20% split percentages. Furthermore, the training part is divided into training and validation parts with 20% for the validation part. The validation part is used to estimate model's hyper-parameters. Once the best hyper-parameters are found for each model, the model is trained using the whole training part. The trained model is then evaluated by calculating the classification accuracy using the testing part. This evaluation process using the training-testing split is applied with 30 runs using different random seeds. The reported results for each model are the average over these 30 runs.

V. EXPERIMENTAL EVALUATION

For a comparative evaluation, different machine and deep learning models from the literature are selected to compare their performance to the hybrid proposed model. All models explained in the following subsections use softmax activation function in their output layer. For all models used for the comparative study alongside with the proposed hybrid model, Adam algorithm [44] is used to optimize weights of the models using cross entropy loss function. In addition to the selected models from the literature, the deep CNN part of the proposed hybrid model is used without applying the RBM part. The reason behind this is to verify our hypothesis that the RBM model contributes to enhance the overall accuracy of the hybrid model. We called the deep CNN part of the proposed model, deep convolutional neural network (DCNN). A summary of the models is shown in Table 3.

The multi-layer perceptron (MLP) is a feed forward neural network with an input layer, one hidden layer, and an output layer [45]. Rectified linear unit activation function is used in the hidden layer units. The hidden and output layers are proceeded by a dropout operation [46]. Deep multi-layer perceptron (DMLP) is a fully connected feed forward model with four layers in total, including the input layer [47]. The three hidden layers applies rectified linear activation function. Each hidden layer and the output layer are proceeded by a dropout operation.

Fully connected convolutional neural network (FCCNN) model is a typical CNN without any local pooling layers. However, it contains a global average pooling layer at the end. This global average pooling layer is then fully connected to the output layer [47]. This model contains three convolutional layers. Each layer consists of a typical convolution operation, followed by a batch normalization operation [48] and then a rectified linear unit activation function. The results of the third convolution layer are passed to the average pooling layer. Finally, this average pooling layer is connected to the output layer.

Time convolutional neural network (TCNN) is a typical CNN model with two convolution layers [49]. Each convolution layer contains a convolution operation, sigmoid activation function operation, and a local average pooling operation. The results of the second convolution layer are flattened and then fully connected to the output classification layer. Deep residual network (DRN) is a deep CNN model with 11 layers where the first 9 layers are convolutional. These layers are followed by a global average pooling layer. This global average layer is then fully connected to the output layer [47]. Each convolution layer is followed by a batch normalization operation, where the results are passed to a rectified linear unit activation function.

Time LeNet (TLN) model has two convolutional layers, followed by a fully connected layer that finally connects

TABLE 3. Architectural properties for the deep models used for the comparative study. Number of layers (#Layers) refers to the total number of hidden layers plus any other types of layers, excluding the input and output layers that are common in all models. #Conv is the total number of convolution layers.

	#Layers	#Conv	Normalize	Pooling	Final Feature Map	Activation	Regularizer
MLP	1	0	None	None	None	ReLU	Dropout
DMLP	3	0	None	None	None	ReLU	Dropout
FCCNN	4	3	Batch	None	GAP	ReLU	None
TCNN	3	2	None	Average	Flatten	Sigmoid	None
DRN	10	9	Batch	None	GAP	ReLU	None
TLN	3	2	None	Max	Flatten	ReLU	None
CNNA	4	3	Instance	Max	GAP	ReLU	Dropout
DCNN	8	7	None	Max	GAP	PReLU	None
ESNC	2	0	None	None	None	ReLU	Dropout
Proposed	10	7	None	Max	GAP	PReLU	None

to the classification output layer [50]. Each convolution layer starts with a typical convolution operation followed by a rectified linear unit activation function. Then it is passed to a local max pooling layer. Convolutional neural network with attention (CNNA) is a deep model with three attention layers [51]. Each attention layer applies five operations: convolution operation, instance normalization, rectified linear unit activation function, dropout operation and local max pooling operation. The resulting activations from the third attention layer are passed to a global average pooling layer. This average pooling layer is then fully connected to the output layer. Instance normalization operation is applied to prevent overfitting by subtracting the mean and dividing the feature by its feature set standard deviation [52].

Echo state networks are a special type of recurrent neural networks (RNNs) [53]. Typical ESN architecture contains an input layer, a hidden layer called reservoir, and an output layer. The main advantage of ESN is its straightforward training method where all network weights, except for output weights, are fixed. Echo state network classifier (ESNC) is a typical ESN model, which usage has been extended in classification tasks instead of just time series prediction [54]. This model is used to map the input features into a higher dimension. Then the results of the dynamical reservoir are passed into a fully connected layer. Finally, the results from the fully connected layer are passed to an output classification layer. The fully connected layer applies a rectified linear unit as the activation function. Two drop operations are applied, one with the fully connected layer and the other operation with the output layer.

A. EXPERIMENTAL SETUP

All models were implemented in Python 3.7 using Keras and Tensorflow frameworks [55]. All experiments were conducted on Intel Xeon Silver 4116 processor (2.10 GHz) machine, with 128 GB of RAM and Windows 10 (64 bit) operating system. Table 4 lists the specification details.

TABLE 4. Experimental setup specification details.

CPU	Intel Xeon 4116 Processor
Number of cores	4
Number of threads	8
Processor frequency	2.10 GHz
Cache	16 MB L3 Cache
RAM	128 GB
Memory Type	DDR4
Operating system	Windows 10 (64-bit)
Python version	3.7
Machine learning libraries	Keras 2.3.1, Tensorflow 2.2.0

VI. RESULTS

This section presents the experimental results obtained by applying the proposed hybrid methodology and other models for detecting DDoS and replay attacks using the synthesized datasets. The accuracy measure is used for evaluating the models' performance:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}} \tag{1}$$

where TP, TN, FP and FN are the true positives, true negatives, false positives, and false negatives, respectively.

All reported results were based on the average accuracy over 30 experimental runs. First, we visualized the average accuracy measure for the proposed model and all other models over the three datasets using bar chart, as shown in Fig. 3. Secondly, the mean and standard deviation for measured testing accuracy of all models were reported for each dataset, as shown in Tables 5, 6, and 7. In addition to the reported average accuracy, the accuracy was measured for each class. Finally, we have drawn the corresponding critical difference diagrams for each dataset, as shown in Figs. 4, 5 and 6, and critical difference diagram for all datasets, as shown in Fig. 7. Critical difference diagrams are based on the Wilcoxon-Holm method to detect pairwise significance [49]. First the Friedman test was performed to reject the null hypothesis, which states that there is no significant difference

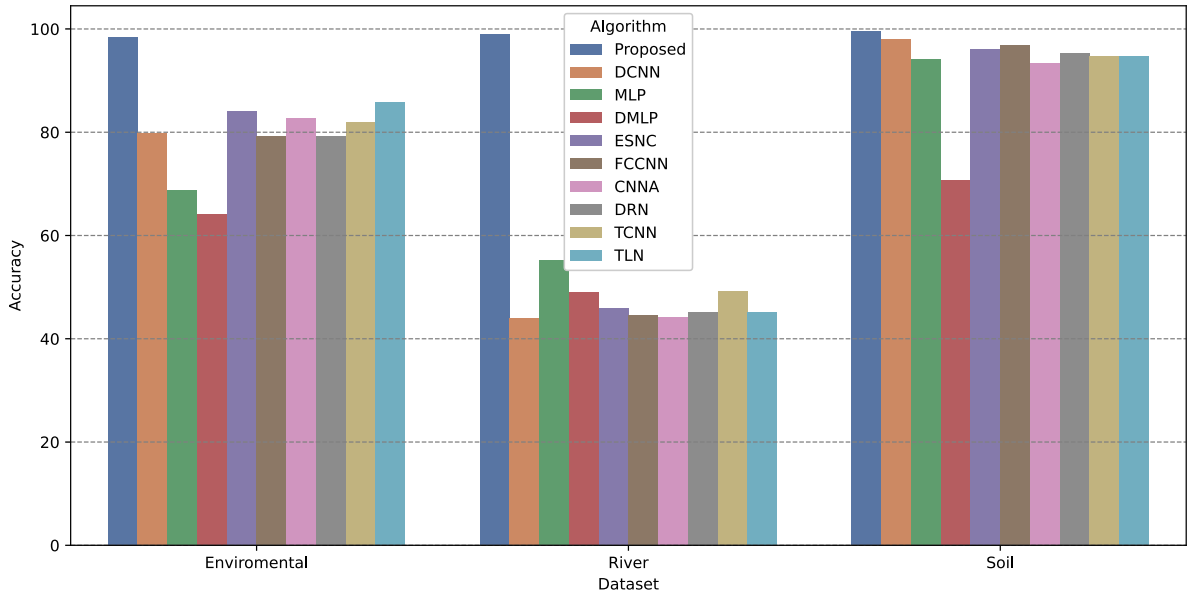

FIGURE 3. Average accuracy measures for the proposed model compared with state-of-the-art models over the smart city platform datasets.

TABLE 5. Reported overall and classes testing and training accuracies for learned models compared with the proposed hybrid model for smart environmental dataset. The mean and standard deviation are calculated for the accuracy measure (mean±std). Bold indicates the best reported model performance for each dataset.

	Proposed	DCNN	MLP	DMLP	ESNC	FCCNN	CNNA	DRN	TCNN	TLN
Normal Class (Testing)	**0.9836±0.0034**	0.7701±0.0199	0.7087±0.0231	0.6622±0.0011	0.9078±0.0199	0.8027±0.0077	0.8991±0.0397	0.8129±0.0307	0.8215±0.0001	0.9277±0.0128
Normal Class (Training)	**0.9973±0.0112**	0.8612±0.0023	0.7602±0.0061	0.7212±0.0021	0.9623±0.0021	0.8626±0.0061	0.9223±0.0312	0.8528±0.0311	0.8555±0.0023	0.9592±0.0111
Replay Class (Testing)	**0.9811±0.0019**	0.6239±0.0184	0.6411±0.0021	0.6001±0.0023	0.6174±0.0745	0.5781±0.0126	0.5859±0.0403	0.5636±0.0194	0.8198±0.0021	0.6473±0.0348
Replay Class (Training)	**0.9943±0.0021**	0.7039±0.0134	0.7411±0.0121	0.7301±0.0011	0.7023±0.0611	0.6571±0.0033	0.6678±0.0014	0.6178±0.0114	0.8712±0.0001	0.7844±0.0156
DDoS Class (Testing)	**0.9866±0.0129**	0.9987±0.0008	0.7112±0.0011	0.6622±0.0011	0.9991±0.0005	0.9986±0.0007	0.9983±0.0004	0.9991±0.0004	0.8153±0.0043	0.9975±0.0011
DDoS Class (Training)	**0.9912±0.0021**	0.9992±0.0018	0.7912±0.0211	0.6921±0.0021	0.9999±0.0013	0.9999±0.0013	0.9991±0.0011	0.9999±0.0001	0.8812±0.0002	0.9989±0.0023
Overall (Testing)	**0.9837±0.0012**	0.7976±0.0107	0.6869±0.0312	0.6415±0.0091	0.8415±0.0304	0.7931±0.0049	0.8277±0.0255	0.7919±0.0144	0.8186±0.0056	0.8575±0.0139
Overall (Training)	**0.9942±0.0011**	0.8547±0.0121	0.7641±0.0112	0.7141±0.0021	0.8881±0.0003	0.8398±0.0035	0.83617±0.0033	0.8235±0.0034	0.8693±0.0011	0.9141±0.0011

TABLE 6. Reported overall and classes testing and training accuracies for learned models compared with the proposed hybrid model for smart river dataset. The mean and standard deviation are calculated for the accuracy measure (mean±std). Bold indicates the best reported model performance for each dataset.

	Proposed	DCNN	MLP	DMLP	ESNC	FCCNN	CNNA	DRN	TCNN	TLN
Normal Class (Testing)	**0.9808±0.0134**	0.5738±0.0071	0.5872±0.0021	0.5283±0.0013	0.5813±0.0113	0.5755±0.0071	0.5676±0.0083	0.5775±0.0076	0.5413±0.0014	0.5828±0.0051
Normal Class (Training)	**0.9899±0.0012**	0.6344±0.0011	0.6512±0.0033	0.6012±0.0012	0.6599±0.0124	0.6192±0.0011	0.6167±0.0016	0.6311±0.0001	0.6011±0.0012	0.6198±0.0001
Replay Class (Testing)	**0.9721±0.0306**	0.2195±0.0106	0.4752±0.0081	0.4488±0.0002	0.2166±0.0222	0.2249±0.0118	0.2286±0.01811	0.2222±0.0154	0.4022±0.0013	0.1287±0.0245
Replay Class (Training)	**0.9878±0.0319**	0.2981±0.0006	0.5811±0.00231	0.5002±0.0011	0.2511±0.0154	0.2812±0.0012	0.2612±0.00712	0.2699±0.0103	0.4398±0.0003	0.1699±0.0112
DDoS Class (Testing)	**0.9812±0.0087**	0.5234±0.0127	0.5922±0.0033	0.4895±0.0018	0.5822±0.0296	0.5372±0.0222	0.5291±0.0259	0.5518±0.0169	0.5313±0.0014	0.6441±0.0223
DDoS Class (Training)	**0.9901±0.0012**	0.6011±0.0113	0.6389±0.0143	0.5412±0.0132	0.6387±0.0211	0.5999±0.0001	0.5712±0.0209	0.6156±0.0006	0.5602±0.0011	0.6701±0.0201
Overall (Testing)	**0.9813±0.0117**	0.4389±0.0071	0.5515±0.0001	0.4897±0.0043	0.4601±0.0146	0.4459±0.0094	0.4418±0.0121	0.4505±0.0072	0.4913±0.0064	0.4519±0.0129
Overall (Training)	**0.9892±0.0112**	0.7668±0.0061	0.6239±0.0011	0.5475±0.0021	0.5165±0.0112	0.5001±0.0019	0.4831±0.0021	0.5055±0.0011	0.5337±0.0004	0.4866±0.0019

TABLE 7. Reported overall and classes testing and training accuracies for learned models compared with the proposed hybrid model for smart soil dataset. The mean and standard deviation are calculated for the accuracy measure (mean±std). Bold indicates the best reported model performance for each dataset.

	Proposed	DCNN	MLP	DMLP	ESNC	FCCNN	CNNA	DRN	TCNN	TLN
Normal Class (Testing)	**0.9991±0.0007**	0.9952±0.0051	0.9958±0.0008	0.7353±0.0111	0.9625±0.0011	0.9958±0.0006	0.9602±0.0006	0.9962±0.0008	0.9847±0.0029	0.9949±0.0019
Normal Class (Training)	**0.9999±0.0011**	0.9998±0.0011	0.9966±0.0011	0.7598±0.0008	0.9799±0.0023	0.9973±0.0001	0.9811±0.0011	0.9981±0.0017	0.9899±0.0008	0.9984±0.0023
Replay Class (Testing)	**0.9952±0.0011**	0.9934±0.0021	0.9826±0.0012	0.6822±0.0102	0.9588±0.0004	0.9867±0.0015	0.8833±0.0116	0.9855±0.0032	0.9745±0.0021	0.9853±0.0031
Replay Class (Training)	**0.9998±0.0001**	0.9956±0.0023	0.9899±0.0045	0.7156±0.0187	0.9734±0.0013	0.9899±0.0035	0.9002±0.0003	0.9901±0.0021	0.9896±0.0098	0.9899±0.0092
DDoS Class (Testing)	**0.9932±0.0032**	0.9518±0.0216	0.8475±0.0202	0.7016±0.0005	0.9622±0.0008	0.9221±0.0244	0.9587±0.0019	0.8755±0.0927	0.8851±0.0147	0.8586±0.0436
DDoS Class (Training)	**0.9995±0.0031**	0.9702±0.0102	0.8723±0.0212	0.7302±0.0019	0.9823±0.0011	0.9523±0.0134	0.9677±0.0003	0.8976±0.0298	0.9004±0.0122	0.8879±0.0451
Overall (Testing)	**0.9951±0.0011**	0.9801±0.0075	0.9421±0.0066	0.7063±0.0702	0.9614±0.0048	0.9682±0.0082	0.9341±0.0022	0.9524±0.0316	0.9481±0.0047	0.9463±0.0155
Overall (Training)	**0.9997±0.0023**	0.9885±0.0011	0.9529±0.0204	0.7352±0.0007	0.9785±0.0011	0.9798±0.0025	0.9496±0.0032	0.9619±0.0231	0.9599±0.0051	0.9587±0.0166

between the compared pairwise models. We then proceeded with a post-hoc analysis based on the Wilcoxon-Holm method. Each model was marked with a number that indicates the model ranking over the horizontal scale line. A thick horizontal line groups a set of models that are not significantly different.

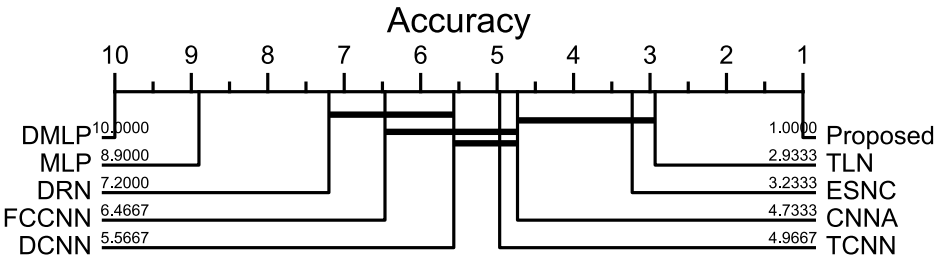

FIGURE 4. Critical difference diagram showing pairwise statistical differences between the proposed methodology and the other models using the environmental dataset.

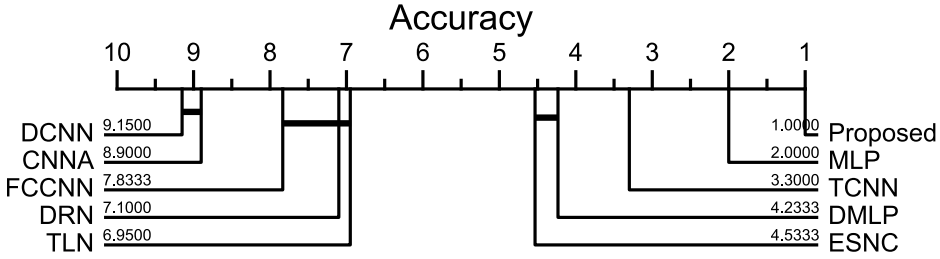

FIGURE 5. Critical difference diagram showing pairwise statistical differences between the proposed methodology and the other models using the river dataset.

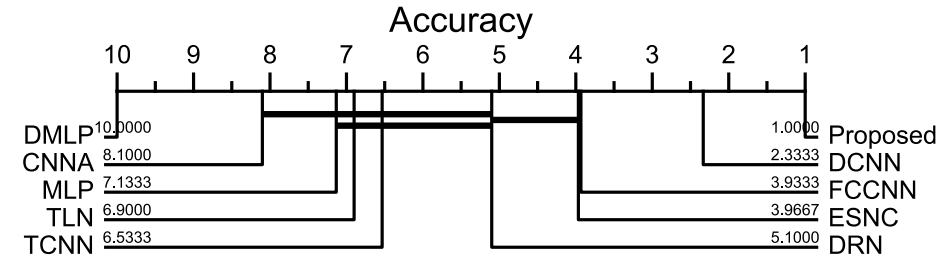

FIGURE 6. Critical difference diagram showing pairwise statistical differences between the proposed methodology and the other models using the soil dataset.

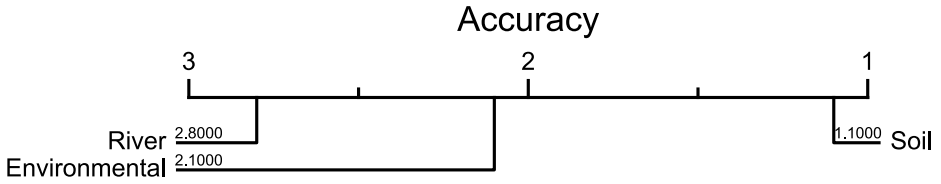

FIGURE 7. Critical difference diagram showing pairwise statistical differences between the datasets.

The proposed model reported the best accuracy and outperformed all other models in all datasets. As we mentioned earlier, our hypothesis of adding the RBM part to the proposed model was due to its ability to model the probability distribution of the data in unsupervised way without relying on the associated classes. This provides the advantage of learning the underlying data model independently. In addition to the ability to model the dataset into a new representation with more rich features. To verify our hypothesis, we added the CNN part of the proposed model as a separate model (DCNN) to compare its performance to the proposed model and other learning models. The reported performance of the proposed model compared to the DCNN showed a noticeable improvement. As shown in Fig. 3, the RBM part enhanced the accuracy of the model. This improvement is clearly shown in the river dataset (Table 6). The river dataset is challenging to model due to the small number of features and instances. However, the proposed model was able to model this dataset with relatively high accuracy.

The DMLP model reported the worst performance in the environmental and soil dataset, while the DCNN model reported its poorest performance in the river dataset. The environmental and soil datasets have a large number of instances compared to the river dataset. Deep learning models, specifically CNN, performed better with more data. This may explain why the DCNN model reported the poorest performance with the river dataset. In addition, all other CNN-based models reported poor performances when compared to the MLP and DMLP models (Fig. 5). In contrast, the MLP and DMLP models reported poor performances in the environmental and soil datasets, while the CNN-based models performed better (Figs. 4 and 6). Again, this could be due to the large number of instances available in these datasets. The ESNC model performance is in the top 5 best models for all datasets. This shows a stable and robust performance compared to other models. Its performance in each dataset is fluctuating to be either in the top 5 best models or top 5 worst models. ESNC model has an advantage over all other models in its ability to model the time factor efficiently since its originally an RNN model.

In the environmental dataset, the proposed model shows a significant difference to all other models (Fig. 4). Despite reporting the poorest performance, MLP and DMLP showed a significant difference when compared to the other models. For all CNN-based approaches, the critical difference diagram shows that there are no significant differences between any of them. For the river dataset, the proposed model shows a significant difference compared to all other models (Fig. 5. The diagram shows that there is no significant difference between FCCNN, DRN and TLN models and between DCNN and CNNA models. For the soil dataset, the proposed model shows a significant difference compared to all other models (Fig. 6). The critical difference diagram shows that there is no significant difference between all CNN-based models and MLP model. In summary, the variations in the CNN-based models did not enhanced the performance of attack detection.

At the level of the datasets, it is clear that the river dataset is the most difficult to model due to its lower accuracies (Fig. 7). As mentioned before, this is due to the small number of instances, features and complicated probability distributions. The soil dataset reported the best accuracies compared to other datasets. The soil dataset has a large number of instances with input features drawn from normal probability distributions. The critical difference diagram for the datasets (Fig. 7) shows a significant difference between all datasets, specifically the river dataset.

One of the recent proposed models for attack detection in IoT literature that is similar to our proposed model is the work that is introduced in [56]. This work proposed a deep autoencoder model, with new regularizer term added to the loss function, that is used to learn latent representation of the dataset. The conducted experimental evaluation showed that the latent representation helped the classifiers perform much better than applying them directly to original features. Another similar work to our proposed model that utilizes the deep autoencoders for attack detection is proposed in [57]. Two autoencoders models are trained, where the first model is trained on the source datasets using supervised learning mode; while the second model is trained on the target dataset using the unsupervised learning mode. The latent representation from the second autoencoder model is used for attack detection. However, in our proposed work we considered the time dependencies in the dataset. In addition, the synthesized attack dataset used to evaluate our proposed model is based on real-life smart city infrastructure.

The proposed model is built to detect replay and DDoS attacks by training the model in a variety of normal and attack instances. Despite it is trained to detect replay and DDoS attacks only, it has the capacity to detect more and different types of attacks. The unsupervised part of the proposed model has the ability to learn latent features from the dataset to enhance the performance of the attack detection classifier. In addition, the classifier part of the proposed model is based on the softmax classes representation, which proves its ability to handle classification problems with large number of classes. This provides an indication for the ability of the proposed model to detect more and different types of cyber attacks. In case of an abnormal behavior is introduced to the proposed hyper model, the model will make a decision to be either DDoS or replay attack. However, to make sure the proposed model could distinguish more different types of attacks, it needs to be trained on datasets that contain records of these attacks.

VII. CONCLUSION

The hybrid deep learning model proposed in this paper for replay and DDoS attacks detection contributes to the field of securing smart city infrastructure and services. The performance of the proposed methodology in this paper was evaluated by synthetically generating replay and DDoS attack data. Attack data was generated from real-life normal behaviour recorded in the smart city of Queanbeyan, Australia. The performance of the proposed methodology was compared with machine and deep learning models from the literature.

The experimental results showed that our proposed model outperforms all other models with high detection accuracy. The experimental results showed the importance of the RBM part of the proposed model. It overcomes the small number of features, and the complicated probability distributions presented in the datasets. The reported results showed a significant enhancement to the proposed methodology by adding the RBM part, compared to the results obtained from the deep CNN part of the proposed methodology applied alone. Modelling the river dataset was more complicated than other datasets due to the small number of data instances and the complicated probability distributions. The environmental dataset is also a challenging dataset, due to the complicated probability distributions. The soil dataset performed best overall, since all its input features are drawn from normal distributions.

The work introduced in this paper could be extended in different directions. Firstly, the proposed model could be applied to real life smart city attack dataset, instead of synthesized simulated attacks. Secondly, the proposed model could be integrated into real world security platforms to contribute to real-time attack detection. Finally, the proposed model could be compared to recent approaches of attack detection in IoT and smart city domains by conducting an experimental evaluation study using the datasets used in this paper.

REFERENCES

[1] L. Da Xu, W. He, and S. Li, "Internet of Things in industries: A survey," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2233–2243, Nov. 2014.

[2] B. Farahani, F. Firouzi, V. Chang, M. Badaroglu, N. Constant, and K. Mankodiya, "Towards fog-driven IoT eHealth: Promises and challenges of IoT in medicine and healthcare," *Future Generat. Comput. Syst.*, vol. 78, pp. 659–676, Jan. 2018.

[3] H. Kumar, M. K. Singh, M. P. Gupta, and J. Madaan, "Moving towards smart cities: Solutions that lead to the smart city transformation framework," *Technol. Forecasting Social Change*, vol. 153, Apr. 2020, Art. no. 119281.

[4] Z. Allam and Z. A. Dhunny, "On big data, artificial intelligence and smart cities," *Cities*, vol. 89, pp. 80–91, Jun. 2019.

[5] W. Wang and Z. Lu, "Cyber security in the smart grid: Survey and challenges," *Comput. Netw.*, vol. 57, no. 5, pp. 1344–1371, 2013.

[6] M. Ezuma, F. Erden, C. K. Anjinappa, O. Ozdemir, and I. Guvenc, "Detection and classification of UAVs using RF fingerprints in the presence of Wi-Fi and Bluetooth interference," *IEEE Open J. Commun. Soc.*, vol. 1, pp. 60–76, 2020.

[7] M. Vitunskaite, Y. He, T. Brandstetter, and H. Janicke, "Smart cities and cyber security: Are we there yet? A comparative study on the role of standards, third party risk management and security ownership," *Comput. Secur.*, vol. 83, pp. 313–331, Jun. 2019.

[8] R. Khatoun and S. Zeadally, "Cybersecurity and privacy solutions in smart cities," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 51–59, Mar. 2017.

[9] A. Zaman, B. Safarinejadian, and W. Birk, "Security analysis and fault detection against stealthy replay attacks," *Int. J. Control*, pp. 1–22, Dec. 2020, doi: 10.1080/00207179.2020.1862917.

[10] S. Hameed, S. A. Shah, Q. S. Saeed, S. Siddiqui, I. Ali, A. Vedeshin, and D. Draheim, "A scalable key and trust management solution for IoT sensors using SDN and blockchain technology," *IEEE Sensors J.*, vol. 21, no. 6, pp. 8716–8733, Jan. 2021.

[11] J. Pacheco, C. Tunc, and S. Hariri, "Design and evaluation of resilient infrastructures systems for smart cities," in *Proc. IEEE Int. Smart Cities Conf. (ISC)*, Trento, Italy, Sep. 2016, pp. 1–6.

[12] A. A. Elsaeidy, N. Jagannath, A. G. Sanchis, A. Jamalipour, and K. S. Munasinghe, "Replay attack detection in smart cities using deep learning," *IEEE Access*, vol. 8, pp. 137825–137837, 2020.

[13] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, "A survey on cyber security for smart grid communications," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 4, pp. 998–1010, 4th Quart., 2012.

[14] J. Bhayo, R. Jafaq, A. Ahmed, S. Hameed, and S. A. Shah, "A time-efficient approach towards DDoS attack detection in IoT network using SDN," *IEEE Internet Things J.*, early access, Jul. 19, 2021, doi: 10.1109/JIOT.2021.3098029.

[15] A. A. Jahromi, A. Kemmeugne, D. Kundur, and A. Haddadi, "Cyber-physical attacks targeting communication-assisted protection schemes," *IEEE Trans. Power Syst.*, vol. 35, no. 1, pp. 440–450, Jan. 2020.

[16] S. A. Abbas and M. S. Almhanna, "Distributed denial of service attacks detection system by machine learning based on dimensionality reduction," *J. Phys., Conf. Ser.*, vol. 1804, no. 1, Mar. 2021, Art. no. 012136.

[17] A. Elsaeidy, K. S. Munasinghe, D. Sharma, and A. Jamalipour, "A machine learning approach for intrusion detection in smart cities," in *Proc. Veh. Technol. Conf.*, Honolulu, HI, USA, Sep. 2019, pp. 1–5.

[18] M. F. Elrawy, A. I. Awad, and H. F. A. Hamed, "Intrusion detection systems for IoT-based smart environments: A survey," *J. Cloud Comput.*, vol. 7, no. 1, p. 21, Dec. 2018.

[19] S. Ho, S. A. Jufout, K. Dajani, and M. Mozumdar, "A novel intrusion detection model for detecting known and innovative cyberattacks using convolutional neural network," *IEEE Open J. Comput. Soc.*, vol. 2, pp. 14–25, 2021.

[20] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016.

[21] K. A. P. da Costa, J. P. Papa, C. O. Lisboa, R. Munoz, and V. H. C. de Albuquerque, "Internet of Things: A survey on machine learning-based intrusion detection approaches," *Comput. Netw.*, vol. 151, pp. 147–157, Mar. 2019.

[22] M. Shafiq, Z. Tian, A. K. Bashir, A. Jolfaei, and X. Yu, "Data mining and machine learning methods for sustainable smart cities traffic classification: A survey," *Sustain. Cities Soc.*, vol. 60, Sep. 2020, Art. no. 102177.

[23] S.-W. Lee, H. M. Sidqi, M. Mohammadi, S. Rashidi, A. M. Rahmani, M. Masdari, and M. Hosseinzadeh, "Towards secure intrusion detection systems using deep learning techniques: Comprehensive analysis and review," *J. Netw. Comput. Appl.*, vol. 187, Aug. 2021, Art. no. 103111.

[24] A. Abeshu and N. Chilamkurti, "Deep learning: The frontier for distributed attack detection in fog-to-things computing," *IEEE Commun. Mag.*, vol. 56, no. 2, pp. 169–175, Feb. 2018.

[25] A. A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for Internet of Things," *Future Gener. Comput. Syst.*, vol. 82, pp. 761–768, May 2018.

[26] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proc. Int. Bio-Inspired Inf. Commun. Technol. Conf.*, New York, NY, USA, Dec. 2016, pp. 21–26.

[27] M. A. Salama, H. F. Eid, R. A. Ramadan, A. Darwish, and A. E. Hassanien, "Hybrid intelligent intrusion detection scheme," in *Proc. Soft Comput. Ind. Appl. Conf.*, Berlin, Germany, 2011, pp. 293–303.

[28] 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.

[29] B. Hussain, Q. Du, B. Sun, and Z. Han, "Deep learning-based DDoS-attack detection for cyber–physical system over 5G network," *IEEE Trans. Ind. Informat.*, vol. 17, no. 2, pp. 860–870, Feb. 2021.

[30] Y. Zhang, P. Li, and X. Wang, "Intrusion detection for IoT based on improved genetic algorithm and deep belief network," *IEEE Access*, vol. 7, pp. 31711–31722, 2019.

[31] D. Wu, Z. Jiang, X. Xie, X. Wei, W. Yu, and R. Li, "LSTM learning with Bayesian and Gaussian processing for anomaly detection in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 8, pp. 5244–5253, Aug. 2020.

[32] A. Essien and C. Giannetti, "A deep learning model for smart manufacturing using convolutional LSTM neural network autoencoders," *IEEE Trans. Ind. Informat.*, vol. 16, no. 9, pp. 6069–6078, Sep. 2020.

[33] S. Nakariyakul, "A solution to the high-dimensional classification problem using an improved hybrid feature selection algorithm guided by interaction information," *IEEE Access*, vol. 8, pp. 145909–145917, 2020.

[34] J. Chu, H. Wang, H. Meng, P. Jin, and T. Li, "Restricted Boltzmann machines with Gaussian visible units guided by pairwise constraints," *IEEE Trans. Cybern.*, vol. 49, no. 12, pp. 4321–4334, Dec. 2019.

[35] A. L. Yuille, "The convergence of contrastive divergences," in *Proc. Adv. Neural Inf. Process. Syst. Conf.*, Vancouver, BC, Canada, Jul. 2005, pp. 1593–1600.

[36] N. Zhang, S. Ding, J. Zhang, and Y. Xue, "An overview on restricted Boltzmann machines," *Neurocomputing*, vol. 275, pp. 1186–1199, Jan. 2018.

[37] L. Yin and J. Xie, "Multi-temporal-spatial-scale temporal convolution network for short-term load forecasting of power systems," *Appl. Energy*, vol. 283, Feb. 2021, Art. no. 116328.

[38] M. Zhu, W. Min, Q. Wang, S. Zou, and X. Chen, "PFLU and FPFLU: Two novel non-monotonic activation functions in convolutional neural networks," *Neurocomputing*, vol. 429, pp. 110–117, Mar. 2021.

[39] H. Wu and X. Gu, "Max-pooling dropout for regularization of convolutional neural networks," in *Proc. Int. Neural Inf. Process. Conf.*, Istanbul, Turkey, Nov. 2015, pp. 46–54.

[40] Z. Li, S.-H. Wang, R.-R. Fan, G. Cao, Y.-D. Zhang, and T. Guo, "Teeth category classification via seven-layer deep convolutional neural network with max pooling and global average pooling," *Int. J. Imag. Syst. Technol.*, vol. 29, no. 4, pp. 577–583, May 2019.

[41] S. Totaro, A. Hussain, and S. Scardapane, "A non-parametric softmax for improving neural attention in time-series forecasting," *Neurocomputing*, vol. 381, pp. 177–185, Mar. 2020.

[42] A. Elsaeidy. (Nov. 2021). *Queanbeyan Smart City Platform Datasets*. [Online]. Available: https://github.com/asmaa-elsaeidy/QBN-Smart-City-Dataset.git
[43] C. G. Cordero, E. Vasilomanolakis, A. Wainakh, M. Mühlhäuser, and S. Nadjm-Tehrani, "On generating network traffic datasets with synthetic attacks for intrusion detection," *ACM Trans. Privacy Secur.*, vol. 24, no. 2, pp. 1–39, Feb. 2021.
[44] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Learn. Represent. Conf.*, San Diego, CA, USA, May 2015, pp. 1–15.
[45] M. W. Gardner and S. R. Dorling, "Artificial neural networks (the multilayer perceptron)—A review of applications in the atmospheric sciences," *Atmos. Environ.*, vol. 32, nos. 14–15, pp. 2627–2636, Aug. 1998.
[46] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.
[47] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *Proc. Int. Joint Neural Netw. Conf.*, Anchorage, AK, USA, May 2017, pp. 1578–1585.
[48] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?" in *Proc. Adv. Neural Inf. Process. Syst. Conf.*, Montreal, QC, Canada, Dec. 2018, pp. 2483–2493.
[49] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Müller, "Deep learning for time series classification: A review," *Data Mining Knowl. Discovery*, vol. 33, no. 4, pp. 917–963, Jul. 2019.
[50] A. Le Guennec, S. Malinowski, and R. Tavenard, "Data augmentation for time series classification using convolutional neural networks," in *Proc. Adv. Anal. Learn. Temporal Data Workshop*, Porto, Portugal, Sep. 2016, pp. 1–9.
[51] J. Serrà, S. Pascual, and A. Karatzoglou, "Towards a universal neural network encoder for time series," in *Proc. Catalan Assoc. Artif. Intell. Conf.*, Catalonia, Spain, Oct. 2018, pp. 120–129.
[52] H. Nam and H.-E. Kim, "Batch-instance normalization for adaptively style-invariant neural networks," in *Proc. Adv. Neural Inf. Process. Syst. Conf.*, Montreal, QC, Canada, Dec. 2018, pp. 2558–2567.
[53] H. Jaeger, *Tutorial on Training Recurrent Neural Networks, Covering BPPT, RTRL, EKF and the 'Echo State Network' Approach*. Bonn, Germany: 5 GMD-Forschungszentrum Informationstechnik, 2002.
[54] Q. Ma, L. Shen, W. Chen, J. Wang, J. Wei, and Z. Yu, "Functional echo state network for time series classification," *Inf. Sci.*, vol. 373, pp. 1–20, Dec. 2016.
[55] *Keras: The Python Deep Learning Library*, Astrophysics Source Code Library, 2018.
[56] L. Vu, V. L. Cao, Q. U. Nguyen, D. N. Nguyen, D. T. Hoang, and E. Dutkiewicz, "Learning latent representation for IoT anomaly detection," *IEEE Trans. Cybern.*, early access, Sep. 18, 2020, doi: 10.1109/TCYB.2020.3013416.
[57] L. Vu, Q. U. Nguyen, D. N. Nguyen, D. T. Hoang, and E. Dutkiewicz, "Deep transfer learning for IoT attack detection," *IEEE Access*, vol. 8, pp. 107335–107344, 2020.

ASMAA A. ELSAEIDY (Student Member, IEEE) received the Ph.D. degree in information system from the University of Canberra, Australia. She is currently working as a Tutor at the University of Canberra. She has five refereed publications with over 57 citations (H-index: three) in highly journals and conference proceedings. Her research interests include smart cities, the Internet of Things, cyber-security, machine learning, and deep learning. She is awarded with the Best Paper Award for a 2017 published conference paper from her Ph.D. work.

ABBAS JAMALIPOUR (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Nagoya University. He is currently a Professor of ubiquitous mobile networking at The University of Sydney. He has authored nine technical books, 11 book chapters, over 550 technical papers, and five patents, all in the area of wireless communications. He is the President of the IEEE Vehicular Technology Society. He is a fellow of the Institute of Electrical, Information, and Communication Engineers (IEICE) and the Institution of Engineers Australia, an ACM Professional Member, and an IEEE Distinguished Speaker. Previously, he was the Executive Vice-President and the Editor-in-Chief of VTS Mobile World and has been an Elected Member of the Board of Governors of the IEEE Vehicular Technology Society, since 2014. He was the Vice President-Conferences and a member of Board of Governors of the IEEE Communications Society. He was a recipient of a number of prestigious awards, such as the 2019 IEEE ComSoc Distinguished Technical Achievement Award in Green Communications, the 2016 IEEE ComSoc Distinguished Technical Achievement Award in Communications Switching and Routing, the 2010 IEEE ComSoc Harold Sobol Award, the 2006 IEEE ComSoc Best Tutorial Paper Award, and 15 best paper awards. He has been the General Chair or the Technical Program Chair for a number of conferences, including IEEE ICC, GLOBECOM, WCNC, and PIMRC. He was the Editor-in-Chief of IEEE Wireless Communications. He serves as an Editor for IEEE Access, IEEE Transactions on Vehicular Technology, and several other journals.

KUMUDU S. MUNASINGHE (Member, IEEE) received the Ph.D. degree in telecommunications engineering from The University of Sydney. He is currently the Head of the School of IT and System, an Associate Professor in network engineering, and the Leader of the IoT Research Group, University of Canberra. He has over 100 refereed publications with over 1100 citations (H-index 18 and FWCI 1.36) in highly prestigious journals, conference proceedings, and two books to his credit. His research interests include next generation mobile and wireless networks, the Internet of Things, green communication, smart grid communications, and cyber-physical-systems and security. He has secured over $ 2.8 million dollars in competitive research funding by winning grants from the Australian Research Council (ARC), the Commonwealth and State Governments, Department of Defence, and Industry. He has also won the highly prestigious ARC Australian Postdoctoral Fellowship. His research has been highly commended through many research awards, including two VC's Research Awards and three IEEE Best Paper Awards. He has served as the co-chair for many international conferences and served as an editorial board member for a number of journals. He is a Charted Professional Engineer, an Engineering Executive, and a Companion/Fellow of Engineers in Australia.