



A Hybrid Deep Learning Model for Securing Smart City Networks Against Flooding Attack

Bashar Ahmed Khalaf^{1,*}, Siti Hajar Othman¹, Shukor Abd Razak² Alexandros Konios³

¹Faculty of Computing, Universiti Teknologi Malaysia, Johor Bahru 81310, Malaysia

²Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Kuala Terengganu 21300, Malaysia

³Nottingham Trent University, Nottingham, UK

Emails: Basharalzubaidy60@gmail.com; hajar@utm.my; shukorrazak@unisza.edu.my; alexandros.konios@ntu.ac.uk

Abstract

Due to the increasing digitization of city processes, there has been a significant shift in how cities are governed and how people make their living. However, several types of attacks could target smart cities, and Flooding Attacks (FA) are the most dangerous type. It is also a major issue for many people and programs using the Internet nowadays. Security in smart cities refers to preventative measures necessary to shield the city and its residents from direct or indirect harm by attackers who try to crash the system and deny legitimate users the use of the services. Smart city security, in contrast to standard security mechanisms, necessitates new and creative approaches to protecting the systems and applications while considering characteristics like resource limitations, distributed architecture nature, and geographic distribution. Smart cities are vulnerable to several particular issues, including faulty communication, insufficient data, and privilege protection. Therefore, a hybrid CRNN model that consists of Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) algorithms is employed for the detection of Flood Attacks based on the classification of traffic data. Subsequently, the performance of the CRNN is tested and evaluated using the CIC-Bell-DNS-EXF-2021 dataset. The obtained accuracy results of the proposed CRNN model achieved in FA detection is 99.2%.

Keywords: Flooding Attack; Smart Cities; CNN; LSTM

1. Introduction

The population of metropolitan areas has been growing quickly in recent decades. Over 50% of people on the planet live in urban areas, according to United Nations Population Fund research [1]. The idea of a "smart city" has garnered excessive interest from academic and industrial communities because of its practical foundation and stringent needs in an urban setting [2]. In an effort to improve citizen services and quality of life, a number of towns have started to formulate their own approaches to the idea of smart cities. Large sums of money are being spent on smart city-related projects in several populous nations. China is currently engaged in over 200 projects aimed at implementing the smart cities concept [3]. Urban municipalities can manage their daily operations to improve the quality of life for people with the use of technologies connected to smart cities. The infrastructure of smart cities consists of several interconnected systems and gadgets that help people in many different ways, including smart parking, smart traffic systems, smart healthcare, smart transportation, smart parking, smart agriculture, and smart housing, to mention a few [4].

However, people may suffer significant security and privacy issues as a result of cities becoming smarter. This is a result of the resource-constrained devices' inherent vulnerability to various security assaults in smart cities [5]. Numerous hacks on smart cities could be the result of these vulnerabilities. For example, by manipulating sensor data, hostile attackers may generate false data, leading to the loss of control over highly intelligent systems [6].

230,000 people in Ukraine experienced a significant power outage in 2015 as a result of a denial of service (DoS) attack launched by hackers on the smart grid [7]. Malicious hackers may target numerous resource-constrained devices, such as cameras and sensors that gather and exchange sensitive data in smart cities endangering residents' safety and privacy. These assaults have made it possible to expose people's private lifestyles and possibly cause financial loss by using home area data that smart homes collect and manage. Research projects that by 2020, the smart city market would have grown to a value of \$1.5 trillion. To realize the goal of smart cities, governments must draw significant investments [5]. Thousands of sensor nodes have been installed across the city as part of this enormous development to give residents access to real-time information about a variety of services, including public transit, traffic patterns, air and water quality, and energy consumption rates, to mention a few [8]. However, handling and examining the enormous volume of private information raises a variety of security and privacy issues as well as worries about how to keep private information safe while it's being accessed by unauthorized persons [9], [10]. Cloud computing can offer affordable data processing and storage services in the Internet of Things (IoT) era and smart cities [11].

In this study, a hybrid deep learning Convolutional Recurrent Neural Network (CRNN) model by integrating Convolutional Neural Network (CNN) with Recurrent Neural Network (RNN) for detecting Flooding Attacks in the IoT environments of smart city networks has been proposed. The remainder of the paper is structured as follows: Section 1 presents the background of the study. Whereas, the most related work of smart city security is illustrated in section 2. In section 3, the research methods and materials which have been used in this study have been demonstrated. Furthermore, the design of the proposed model of this study is explained in detail in section 4. Section 5 presents the outcome of the proposed model and displays the results. Finally, section 6 concludes the study.

2. Related Work

In order to classify Flooding attack traffics in the SDN environment, Wang and Liu [12] suggested information entropy and DL method. The approach employs two levels of detection to identify attacks. The controller will first use information entropy detection to examine the suspicious traffic. A CNN model will then carry out the detection based on the fine-grained packet to distinguish between legitimate and malicious traffic. The authors' methods have been compared to those of the DNN, SVM, and DT. However, according to the obtained results it is observed that the proposed model of Wang and Liu achieved a good performance in detecting DDoS attacks with an accuracy of 98.9%.

In an earlier paper, Sabeel et al. [13] suggested a mixture of two deep learning models, DNN and LSTM, for the prediction of unknown DoS/DDoS attacks. This study's model was trained using the pre-processed DoS/DDoS samples from the CICIDS2017 dataset, and the correctness of the results was evaluated using the combined ANTS2019 dataset. The authors have integrated the CICIDS2017 dataset with the synthesized dataset in the second portion. The models are then retrained, and the effectiveness of their identification of recently synthesized unknown attacks is assessed. The models' performance significantly improved in the second phase of the experiment, where DNN and LSTM achieved detection accuracy of 96.15% and 98.72%, respectively. The DNN and LSTM had AUC values of 98.7 and 98.9, respectively. The dataset ANTS2019 was made artificially to mimic real-world attacks. Only the binary class categorization has been done; no real-time detection setup has been used.

The study's research uses a Graph CNN to detect irregular real-time traffic of RPC, which is similar to that in the study by Chen et al. [14]. The latter sought to identify cyber security flaws in the tens of thousands of RPCs produced by various microservices. A two-step procedure was used in this work to trace, log, and look for anomalies in the RPC traffic. First, a density-clustering algorithm was used to analyze the logged RPC traffic from active microservice functionality and find corresponding RPC chain patterns in the data. These chain patterns are a part of the overall functionality of the microservices. In order to tackle the irregular RPC prediction problem, a GCNN is then utilized to model each component of the RPC traffic and learn the spatiotemporal dependencies of the traffic. For each pre-existing subsystem, a series of unique predictions can be made using these GCNNs. Two case studies made up of actual malicious traffic threat models, such as batch registration of bot accounts and account cracking, were used to evaluate this strategy.

DDoS is one of the factors that contribute to service degradation in the private cloud. The study by Virupakshar et al. [15] focuses on DDoS attacks that involve bandwidth and connection flooding. For the purpose of identifying DDoS attacks in the OpenStack-based cloud, authors have utilized the DT, KNN, NB, and DNN algorithms. The classifier with the highest precision and accuracy was chosen by the authors after they compared a number of

classifier models. The DNN model was selected due to its higher accuracy and precision values when used on a dynamically created dataset. The DNN classifier achieved 96% accuracy and precision for cloud datasets, outperforming DT, KNN, and NB in these metrics. The LAN and cloud datasets are not mentioned, and the authors only used the older KDDCUP99 dataset. For the KDDCUP99 dataset, the precision value of the DNN algorithm is lower than that of other algorithms. Li et al. [16] released a deep-learning model that detects DDoS attacks in an SDN context. The input, output, forward, reverse, FC hidden layer and recursive layers make up the model. CNN, LSTM, and RNN were also used in the model. Consequently, the writers created four unique models: CNN/LSTM, 3LSTM, LSTM, and GRU. When used to forecast a DDoS attack, the ISCX dataset has a 98% accuracy rate. The Ubuntu 14.04 operating system serves as the basis for the DDoS attack detection and defense system, which is tested against active DDoS attacks. Five real-time DDoS attack variations were examined: the Ping of Death attack, the ARP flood inundation attack, the SYN flood inundation attack, the Smurf attack, and the UDP flood inundation attack [17].

3. Research Methods and Materials

As stated in earlier sections, there has been a significant shift in the way cities are governed and how people make their living as a result of the increasing digitization of city processes. Moreover, several types of attacks could target smart cities, the FA attack is the most dangerous type of them. It is also a major issue for many people and programs using the Internet nowadays. The attack is tricky to pinpoint in the first place, and it keeps shifting its tactics. However, in this section, the research and methods employed in this study are presented. Several studies have been well established to defiance against FA and secure the smart cities environment as discussed in the writings but the performance still need to be improved to

A. Convolutional Neural Network (CNN)

A convolutional neural network (CNN) is one of the most effective deep learning models that have been used to process large data; it is inspired by the structure of the visual cortex in animals. To be more precise, its primary goal is to inevitably and modifiable acquire spatial arrangements of attributes in small through upper order structures. Its use in robotics and autonomous vehicles Haider et al. [18] has shown its efficacy in a variety of applications, such as facial recognition, object recognition, and traffic sign tracking. Since the number of variables in an ANN represents the more important component of a CNN, development companies and academics have shifted their attention to bigger structures that can be applied to difficult problems that cannot be solved by conventional ANNs [19].

The main objective of a CNN is to identify the significant features of the input data. The procedure begins with a group of convolutional feature extractors that are fed into a series of learnable filters, making up the first layer. All of the input data is treated as if it were passing through a sliding window that is represented by the filters that have been applied [18]. The outputs in this scenario are called feature maps, and the overlapping distance between them is called the stride. Each CNN layer is comprised of convolutional kernels that generate unique attribute representations. In the feature map of the following layer, a neuron is linked to its neighboring neurons. A feature map's generating kernel must be consistent within all input points. Individual or more fully connected (FC) layers are used to complete the grouping once the convolutional and pooling layers have been formed. The CNN's layered structure is represented in the bellow Figure. Moreover, in Figure 1, each square represents a distinct feature map.

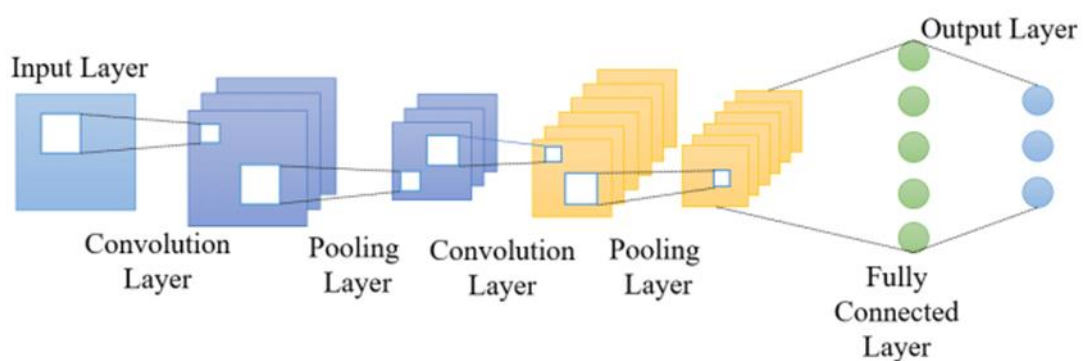


Figure1. The architecture of Deep CNN layers [18]

In CNN structures, the convolutional process throughout all input feature maps and convolutional layers is symbolized by the preceding equation:

$$h_j^{(n)} = \sum_{k=1}^k X W_{kj}^n + b_{kj}^n \quad (1)$$

In which X characterizes a 2D convolution operation, while the outcome of the j th attribute map is represented by $h_j^{(n)}$ that is located in the hidden layer (n th). Further, the k th channel of the $(n-1)$ th hidden layer is expressed by $h_k^{(n-1)}$.

Moreover, the weights are given by W_{kj}^n that are related to the k th path of the filter (j th) that is related to the n th layer. Last but not least, the related bias term is represented by b_{kj}^n . A large number of feature maps are produced when many convolutional layers are combined, hence it is common practice to add a layer on top of the convolutional layers in an attempt to lessen their complexity. A pooling layer does both of these things, reducing the mathematical cost of training a network and the likelihood of overfitting.

Then, an iterative technique is performed that cycles amongst feed-forward and backpropagation information movements to finish training the CNN. Every time the backpropagation procedure is repeated, the convolutional filters and FC layers are tweaked. According to Equation 2, the main goal is to lower the overall average (or mean) loss (E) for every one of the correct groups and the infrastructure outcomes.

$$E = \frac{1}{m} \sum_{i=1}^m \sum_{k=1}^c y_i^k \text{Log}(y_i^{(k)}) \quad (2)$$

Where $y_i^{(k)}$ signifies the true class, while y_i^k symbolizes the network production of the i th contribution in the k th label.

In addition, m stands for neurons in the hidden layer (which serves as training resources), while c represents neurons in the production layer (which serves as a representation of the model's predictions). Numerous methods have been developed for decreasing typical losses; two of the most extensively used are stochastic gradient descent (SGD) [20]. In this context, the word "stochastic" refers to any system or process that is inextricably related to the concept of chance. Therefore, in Stochastic Gradient Descent, instead of processing the entire dataset at each iteration, just a subset of it is processed. According to Shaaban et al. (2019), the term "batch" in Gradient Descent refers to the total number of samples from a dataset that are employed in the gradient calculation for each iteration. Another technique that estimates learning rates that vary with each parameter is called Adaptive Moment Estimation. Gamage et al. [21], [22] combine the benefits of multiple algorithms into a single one, such as how SGD introduces randomness to gradient descent, momentum speeds up settlement, and adaptive gradient, as its word means, adapts to varied learning rates for various parameters.

B. Recurrent Neural Network (RNN)

To overcome the difficulty of learning long-term dependencies caused by the vanishing gradient in traditional RNN, Hochreiter and Schmidhuber [23] developed the Long Short-Term Memory (LSTM) network model. When certain portions of an RNN's weights reach equilibrium throughout training, this is known as the vanishing gradient problem. For this reason, giving more weight to recent developments could mean ignoring the past. As a result, long-term dependencies (recurring relationships) cannot be properly learned. Long short-term memory is built to manage all of the data transfer between neurons. To achieve this goal, we design a gating mechanism to regulate the incorporation and removal of data during the recurrent propagation of a cell state. Consequently, forgetting can be managed, and a defined memory behavior can be realized to simulate both short- and long-term dependence [24]. Below, however, is a detailed description of how LSTMs are modeled, along with a graphical representation of the LSTM's fundamental architecture as displayed in Figure 2.

Recurrent neural networks create a weighted link between the neurons of the input layer, the hidden layer, and the output layer. The hidden layer's output from the network module is dependent on the data from the prior time period. The RNN's recurrent network module stores historical data and uses it to learn about the present [25]. However, while working with lengthy time series data, RNN is vulnerable to the problem of gradient disappearance. So, researchers recommend using an LSTM network to address this issue.

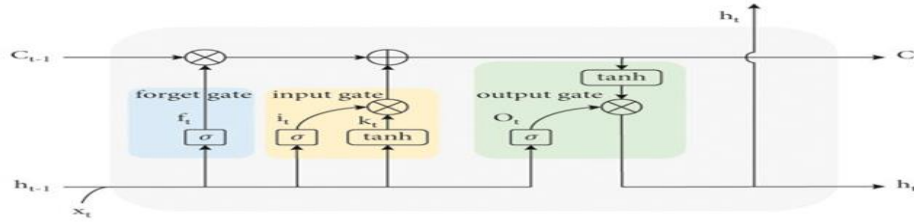


Figure 2. The basic architecture of LSTM [23], [25]

In place of the hidden layer node in the traditional RNN model is a memory unit in the LSTM network. It's the cell state that holds the key to remembering the past. Using the Sigmoid function and the point-by-point product operation, there are three gate designs that can be used to modify or remove data from the cell's state. The internal structure of an LSTM unit is depicted in Figure 1; the forget gate, input gate, and output gate can be seen in order from left to right [25]. To prevent gradient disappearance and enable long-period learning, the LSTM network employs a cumulative linear form to process sequence input. As a result, the LSTM structural may be trained to understand data over much extended periods of time. The "forget gate" is symbolized by equation 3.

$$f_t = \sigma(w_f * [h_{t-1}, x_t] + b_f) \quad (3)$$

The forget gate resultant magnitude is expressed by f_t , consequently, the resultant magnitude of the last moment is denoted by h_{t-1} , the existent magnitude input is given by x_t . The Sigmoid function of the forget gate, and its associated weight matrix and bias vector, are denoted by w_f and b_f , respectively. Also, $[h_{t-1}, x_t]$ is the joining matrix of h_{t-1} and x_t , where:

- w_f shows the weight matrix connected to the forget gate
- $[h_{t-1}, x_t]$ represents the joining of the current input with the hidden state that was previously present;
- b_f is the forget gate's bias;
- σ is the activation function of the sigmoid;

Whereas, the input gate is calculated according to equations 4 and 5,

$$i_t = \sigma(w_i * [h_{t-1}, x_t] + b_i) \quad (4)$$

and,

$$k_t = \tanh(w_k * [h_{t-1}, x_t] + b_k) \quad (5)$$

In the last two expressions, the output of the input gates is i_t and k_t . Further, w_i and b_i are defined previously. w_k and b_k are the input gate's tanh function's weight matrix and bias vector.

. Whereas, output gate are calculated by Equations 6 and 7,

$$O_t = \sigma(w_o * [h_{t-1}, x_t] + b_o) \quad (6)$$

and,

$$h_t = O_t * \tanh(C_t) \quad (7)$$

wherein O_t is the result of the outputting gate, w_o and b_o are the output value of the current instant is denoted by h_t , and the weight matrix and bias vector in the output gate's sigmoid function, respectively. However, the updated cell can be calculated as the below Equation 8,

$$C_t = f_t * C_{t-1} + i_t * k_t \quad (8)$$

wherein C_t represents the cell state of the up-to-date instant and C_{t-1} is the cell state of the last mom.

A common type of neural network for network traffic identification and classification is the CNN. It is made up of one or more convolutional layers that use a series of filters to extract features from the input dataset.

Furthermore, RNNs are a type of neural network that is capable of processing sequential data by preserving a hidden state that records details about prior inputs. As a result, long-term dependencies in sequential data can be modelled. When integrating a CNN with an RNN, the CNN is typically used as a feature extractor for the RNN. The CNN processes the input dataset and extracts a set of features, which are then fed into the RNN. The RNN then processes the sequence of features in a temporal manner, allowing for the modelling of long-term dependencies in the input image.

This model is frequently applied to the classification of network traffic, where the objective is to produce a textual description of an input image. The image's features are extracted by the CNN and put into an RNN to produce the textual description. Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN) are combined in a model for FA detection known as Convolutional Recurrent Neural Network (CRNN).

C. The CIC-Bell-DNS-EXF2021 dataset

The massive CIC-Bell-DNS-EXF-2021 Dataset is made up of 270.8 MB of DNS traffic that was produced via exfiltration of different types of files, varying in size from tiny to large. By using our created feature extractor to extract 30 attributes from the DNS packets, we were able to obtain a final structured dataset that included 323,698 heavy assault samples, 53,978 light attack samples, and 641,642 distinct benign samples [22].

Investigators need access to baseline datasets so that they can test, evaluate, and validate the suppositions behind their approach. However, the hardest issue for working on a Link Flood attack in a smart city environment is to find suitable datasets. Due to the scarcity of such data sets, this presents a significant difficulty. As was seen in the preceding chapter, there are a few distinct varieties of security standard datasets. Those certain datasets would be used for both the identification of various forms of assaults and the evaluation of computer systems [24]. Nevertheless, there is a further restriction posed by the fact that the majority of available datasets are obsolete. It is necessary to obtain a new and adequate dataset to test and assess the suggested strategy in light of the proliferation of new kinds of attacks and security solutions for identifying LFA attack traffic depending on network anomalies.

The evolution of the IT industry has also led to an evolution in the approaches and techniques used by attackers so that they can be able to achieve their goals as well. Therefore, Intrusion Detection Systems (IDSs) are constantly developed in response to the continuous development of new approaches to attacks by attackers. The key motivation for employing the CIC-Bell-DNS-EXF-2021 Dataset in this investigation is due to the fact that it is both up-to-date and adaptable. In order to recreate the individualized data sets required for this study, a supplementary Python 3.10 application is employed in a simulated setting. The following points summarize why the CIC-Bell-DNS-EXF-2021 dataset is settled on [22].

D. Evaluation Methods

Evaluation criteria including F score, accuracy, and precision are used to assess the suggested model. The elements of the confusion matrix serve as the basis for these evaluation metrics [25]. The elements that comprise the confusion matrix are false positive (FP), false negative (TN), false positive (TP), and false negative (FN). To put it simply, TN is the total number of cases that are appropriately categorized as normal. TP represents the proportion of cases that are correctly classified as attacks. FP represents the number of assault cases that were incorrectly classified as typical cases. In a similar vein, FN stands for the number of regular occurrences that were mislabeled as assault instances. The ratio of all correctly classified occurrences (TP, TN) to all instances (TP, TN, FP, and FN) is known as accuracy. Precision is the ratio of total TP and FP to TP. It is sometimes referred to as positive predictive value. The F Score is the ratio of 2TP to the total of 2TP, FP, and FN. The accuracy, precision and F Score calculation in this article is shown in Equation 9, 10, and 11:

$$\text{accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \quad (9)$$

whereby Equation 12 was used to calculate the precision.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (10)$$

Also, the sensitivity was calculated as Equation 13.

$$\text{F Score} = \frac{2TP}{2TP + FP + FN} \quad (11)$$

4. The Design of the CRNN Model

The design of experiments is one of the most important phases in the research process since it provides the researcher with a roadmap in the absence of an experimental plan. The researcher can verify the experiment outcomes when there is a plan in place. Identification of the difficulties is the initial phase in this research process, and the problems are then taken into account while formulating the study objectives. The formulation of research objectives is predicated upon an assessment of prior research. As part of the experimental plan, a suitable dataset for assessing the solution model selection is chosen. For the model to be implemented, the data must be able to show real-time processing. By doing this, the performance evaluation's quality is guaranteed.

Securing billions of connected devices in a smart city environment is an essential mission to realize the full possibility of its services. In this study, a deep learning model for securing smart cities' environments against LFA attacks has been proposed. However, several steps have been considered to design the architecture of the proposed model, in this section, these steps with its components have been displayed in Figure 3 and discussed as the following:

In the design of the proposed CRNN model, the first step presents the data pre-processing which consists of three sub-steps: Exploratory Data Analysis (EDA), coding, and standard scaling. The EDA is used for analyzing the dataset review its main characteristics. Furthermore, the EDA is used to realize what the data can tell us beyond the formal modeling and thereby contrasts traditional hypothesis testing and extract the relation between the features.

In the second step of the pre-processing stage, one hot encoding is used. For machine learning and deep learning algorithms to employ categorical data variables and improve model predictions and classification accuracy, this technique is essential. One Hot Encoding (OHC) is a widely used technique for pre-processing categorical data for machine learning models. The last step in the preprocessing stage is standard scaling, it is a method that has been employed for the normalization of the range for features or independent variables of data. It is also referred to as data normalization in the dataset preprocessing stage and is often carried out in the data preprocessing step.

According to the discussion that has been done in the above chapters, the CIC-Bell-DNS-EXF-2021 dataset has been used to test and evaluate the performance of the proposed model. It consists of three classes which are (Benign traffic, Light attack traffic, and Heavy attack traffic), the dataset is segmented into two parts; the first part takes 60% and is used for system training. Whereas, the second part is taken 40% and used for system testing and evaluation. Each class them contains the traffic types of Benign traffic, Light attack traffic and Heavy attack traffic, to train and test the system and get better accuracy, the study combines the traffic of each similar class and then export it to the deep learning model.

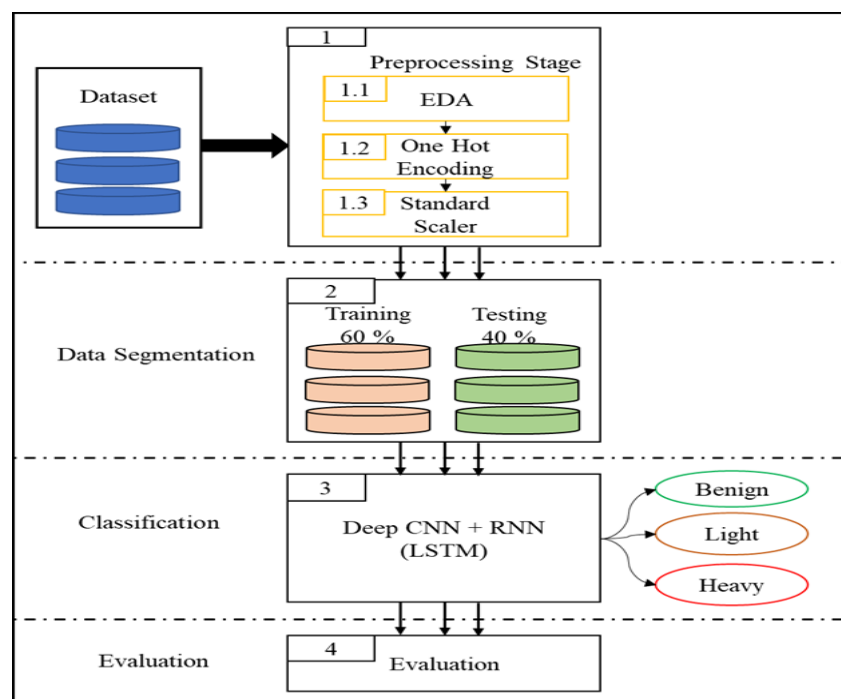


Figure 3. CRNN model architecture

In addition, the dataset is divided into training and testing subsets. The CRNN model is trained using training data. Consequently, the classes of benign traffic, mild attack traffic, and strong attack traffic must be present in the

training data. By using feature extraction and feature fusion, training and testing data are transformed into matching feature maps. During the CRNN model's training phase, feature maps of training data are learned and retained by the model, giving it the ability to recognize FA.

Likewise, the mathematical model and background theory of algorithms are presented in the section on research methods and materials, therefore, in this section, the architecture of the proposed model is detailed directly. According to the above steps, multiple features of network traffic from the CIC-Bell-DNS-EXF-2021 dataset have been extracted. Then, the CRNN model is used to detect and classify the traffic as normal or FA according to the classes of benign, light, and heavy.

In the detection of the FA step, the best non deep learning algorithms have been employed which are CNN, and RNN in the CRNN model. In steps 1-4, the H 5 model has been obtained which contains the dials of the traffic for the CIC-Bell-DNS-EXF-2021 after several steps of processing and classification. Subsequently, the testing data feature maps are classified by the thoroughly trained CRNN model. In the end, the detection system outputs the relevant detection findings based on the model's classification outcomes.

5. Results and Discussion

In this section, the simulation platform, results and discussion have been presented and discussed in details.

A. Simulation Implementation

The suggested system is put into practice using a simulation technique. This section has outlined the proposed model's simulation specification. The partitioning of the implementation specification into hardware and software is explained in the following. The simulation is conducted using a single laptop equipped with an Intel (R) Core (TM) i7-5500U CPU operating at 2.40GHz and 16 GB of RAM. Whereas, Python programming language is used in this study to simulate the proposed model. Python is a programming language that has been used to build network systems. Furthermore, it is covering a huge number of applications related to the networking including traffic models, network environment, IoT environment, smart cities and other else. Therefore, simulation is used for simulating the proposed model.

B. The Results of the CRNN model

The architecture smart cities are made up of numerous parts, which may be seen in the deep view of the Internet network. A web server processes DNS queries from web clients and forwards them to the Internet of Things server based on various configuration factors, such as URL path prefix. These queries are sent to the application server's hosted Internet of Things. An assault that targets IoT and doesn't require access to internal systems is known as an FA attack. As a result, it is difficult to identify the attack in its early stages. In order to do as much harm to the network as possible, a horde of zombies are involved in the attack. One way to carry out a critical assault is to focus a large number of nodes on one target in order to overwhelm the network and cause terrible harm to users. However, the suggested system's design comprises of multiple processes, the first of which is data preprocessing, as will be explained below:

Step 1: export the CIC-Bell-DNS-EXF2021 dataset, then the dataset is segmented into two parts; 70% percentage for training and 30% percentage for testing;

Step 2: Employ the EDA for analysing the dataset to review its main characteristics;

Step 3: using the OHC to offer a means to express categorical characteristics in a format that is appropriate for deep learning models, which often need numerical inputs;

Step 4: Standard scaling employed for normalization of the range for features;

Whereas, in the second stage of the proposed model, the CNN algorithm employed to classify the traffics of flooding attack and it is achieved an **accuracy of 96.92**, f1 score of 96.94, recall of 96.92 and precision pf 96.98 for FA detection. The obtained results are not good enough to be considered in smart cites based IoT environment. Therefore, a combination of the CRNN have been employed to classify the traffics at the FA stage of the proposed model. Furthermore, three types of traffics have been identified as an output of the classification stage which are Heavy, Light and Benign traffics. However, according to the obtained results, it is observed that the CRNN model in the FA classification stage achieved a good result with accuracy of 99.23, f1 score of 99.23, recall of 99.23 and precision pf 99.23, the multi class confusion matrix is displayed in Figure 4.

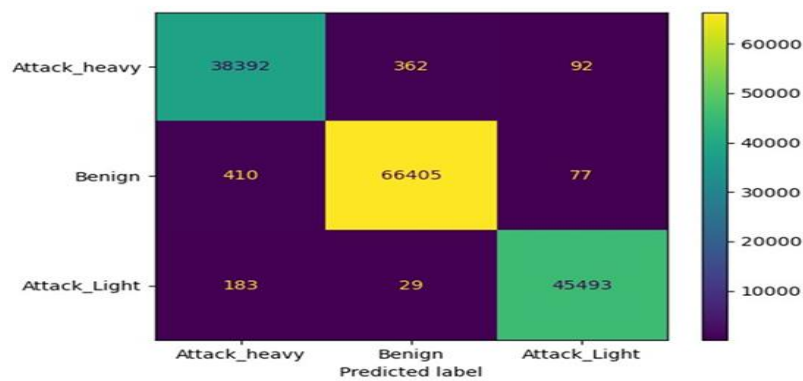


Figure 4. The confusion matrix for the CIC Bell DNS EXF 2021 dataset.

Next, the suggested model's performance was assessed using the F score, accuracy, and precision criteria. The collected results show that, with an accuracy of 99.5, precision of 99.9, and F score of 99.6, the suggested model performed exceptionally well. Figure 5 shows the accuracy and loss of our suggested model in both the training and testing stages.

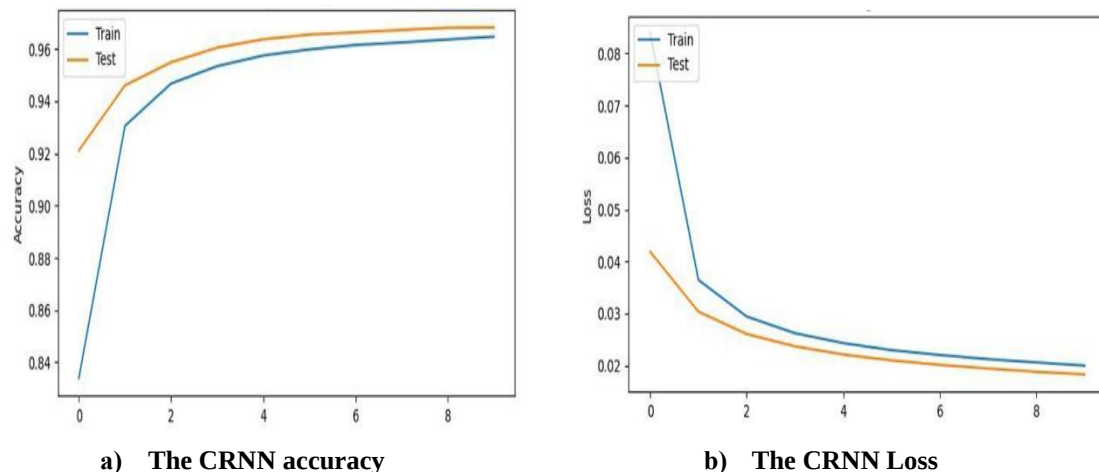


Figure 5. The precision and regression of our suggested model in the training and testing stages

Additionally, the outcomes demonstrate that the CRNN model has the highest accuracy in identifying and differentiating FA traffic. The CRNN model's daily performance results are displayed in Figure 12. As can be seen from the figure, the system's capacity to advance its adaptive behavior over time causes the CRNN model performance to increase daily. Many strategies have been created, proposed, and put into practice by researchers to protect smart cities from FA.

These defense algorithms are nonetheless hampered by hostile traffic that is concealed behind legitimate traffic. In the end, defense strategies are always changing in order to safeguard and enhance computer networks and infrastructures. Like any other model, the CRNN model is an additional attempt to offer varying efficacy against FA attacks. Given the scale of this effort, it is necessary to note three sources of constraints. First off, the processing time is not taken into account in the rating of this work. A second limitation of the evaluation might be that the model is only evaluated with the CIC Bell DNS EXF 2021 dataset. Lastly, the low-rate FA attack instances that are challenging to identify with current methods are not covered by the models testing.

However, real-world network systems are not harmed by these FA attacks. Finally, to be more confident that the CRNN model achieved the goal of the study in identifying and detecting FA, the obtained results have been compared with the most related work of Bakro et al. [26], Zhao et al. [27], and Wang et al. [28]. To get the best answers, Bakro et al. [26] suggested a hybrid feature selection strategy that combined the genetic algorithm (GA) and the grasshopper optimization algorithm (GOA), two bio-inspired algorithms. The GOA-based GA model achieved a good result in identifying the FA for both light and heavy traffic with a total accuracy of 96.51. In the

previous work of Zhao et al. [27], a future fusion based on the CNN model has been proposed to overcome FA. The proposed model achieved a good result with an accuracy of 94.27.

A DNS tunnel identification technique for Android based on isolated forests has been proposed by Wang et al. [28], the authors created a system that enables mobile devices to gather traffic over DNS tunnels. According to the obtained result, it is observed that the proposed model achieved a good performance with accuracy of accuracy of 98.1%. According to the obtained results and during the comparison of the proposed model performance with the most related work of Bakro et al. [26], Zhao et al. [27], and Wang et al. [28], it is observed that the proposed model achieved the best results in identifying FA attack. Table 1 shows the results of the comparison between the performances of the proposed model with the most related work.

Table 1. The comparison between the performance of the proposed model and the most related work

No.	Ref.	Model	Dataset	Accuracy
1	Bakro et al. [26]	GOA based GA	CIC Bell DNS EXF 2021	96.51
2	Zhao et al. [27]	CNN	CIC Bell DNS EXF 2021	94.27
3	Wang et al. [28]	KRTunnel based Isolated Forest	CIC Bell DNS EXF 2021	98.1
4	Our Proposed Model	CNN	CIC Bell DNS EXF 2021	96.92
5		CRNN	CIC Bell DNS EXF 2021	99.23

6. Conclusion

Smart cities are metropolitan regions that use data, technology, and connectivity to increase sustainability, manage resources more effectively, and improve the quality of life for its citizens. The technology and procedures used to guarantee the safety and security of a smart city's residents, data, and infrastructure are referred to as smart city security. Furthermore, several types of attack could target the smart cities, the most common and dangerous type is flooding attack. In this study, a hybrid of deep learning models has been proposed to overcome FA targeting smart cities. The proposed model consists of the CNN and RNN algorithms which have been employed for the detection of Flood Attacks based on the classification of traffic data. Subsequently, the performance of the CRNN is tested and evaluated using the CIC-Bell-DNS-EXF-2021 dataset. The proposed model is applicable in smart city environments. In the future, this study aims to be continued in the direction of cyber security attacks, and the protection system against the sophisticated that target smart cities. Test and evaluate the performance of the proposed system with different benchmarking datasets.

Funding: “This research received no external funding”

Conflicts of Interest: “The authors declare no conflict of interest.”

References

- [1] Darch Abed Dawar, A. (2024). Enhancing Wireless Security and Privacy: A 2-Way Identity Authentication Method for 5G Networks. *International Journal of Mathematics, Statistics, and Computer Science*, 2, 183–198. <https://doi.org/10.59543/ijmscs.v2i.9073>
- [2] A. Sharifi, *et al.*, “ Smart cities and sustainable development goals (SDGs): A systematic literature review of co-benefits and trade-offs,” *Cities*, vol. 146, 104659, 2024.
- [3] C. S. Lai, *et al.*, “ A review of technical standards for smart cities,” *Clean Technologies*, vol. 2, no. 2, pp. 290-310, 2020.

- [4] Chandraleka, J. Selvaraj, P. "Energy-Efficient Smart Farming with IoT-Fog- Based Dual Power Management System," Journal of Journal of Cognitive Human-Computer Interaction, vol. 5, no. 1, pp. 32-41, 2023. **DOI:** <https://doi.org/10.54216/JCHCI.050103>
- [5] F. Al-Turjman, H. Zahmatkesh and R. Shahroze, "An overview of security and privacy in smart cities," *IoT communications, Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 3, e3677, 2022.
- [6] Jain, A. "Establishing IoT Cyber Hygiene Frameworks with Continuous Monitoring and Risk Assessment in Smart City Infrastructures," *Journal of International Journal of Wireless and Ad Hoc Communication*, vol. 7, no. 2, pp. 41-55, 2023. **DOI:** <https://doi.org/10.54216/IJWAC.070203>
- [7] S. Ali and Y. Li, "Learning multilevel auto-encoders for DDoS attack detection in smart grid network," *IEEE Access*, vol. 7, pp. 108647-108659, 2019.
- [8] P. Rizwan, K. Suresh and M. R. Babu, "Real-time smart traffic management system for smart cities by using Internet of Things and big data," *In 2016 international conference on emerging technological trends (ICETT), IEEE*, pp. 1-7, 2016.
- [9] I., S. "A comparison study Big Data Analytics Methods for Selecting Suitable Method," *Journal of International Journal of Advances in Applied Computational Intelligence*, vol. 4, no. 2, pp. 08-14, 2023. **DOI:** <https://doi.org/10.54216/IJAACI.040201>
- [10] K. Ma, *et al.*, "An iot-based fog computing model," *Sensors*, vol. 19, no. 12, 2783, 2019.
- [11] J. M. Abd Elaziz, L. Abualigah and I. Attiya "Advanced optimization technique for scheduling IoT tasks in cloud-fog computing environments," *Future Generation Computer Systems*, vol. 124, pp.142-154, 2021.
- [12] L. Wang and Y. Liu, "A DDoS attack detection method based on information entropy and deep learning in SDN," *In 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), IEEE*, vol. 1, pp. 1084-1088, 2020.
- [13] U. Sabeel, *et al.*, "Evaluation of deep learning in detecting unknown network attacks," *In 2019 International Conference on Smart Applications, Communications and Networking (SmartNets), IEEE*, pp. 1-6, 2019.
- [14] J. Chen, H. Huang and H. Chen, "Informer: irregular traffic detection for containerized microservices RPC in the real world," *In Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, pp. 389-394, 2019.
- [15] K. B. Virupakshar, *et al.*, "Distributed denial of service (DDoS) attacks detection system for OpenStack-based private cloud," *Procedia Computer Science*, vol. 167, pp. 2297-2307, 2020.
- [16] C. Li, *et al.*, "Detection and defense of DDoS attack-based on deep learning in OpenFlow-based SDN," *International Journal of Communication Systems*, vol. 31, no. 5, e3497, 2018.
- [17] B. A. Khalaf, *et al.*, "A simulation study of syn flood attack in cloud computing environment," *AUS journal*, vol. 26, no. 1, pp.188-197, 2017.
- [18] Haider, S., *et al.*, "A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks," *IEEE Access*, vol. 8, pp. 53972-53983.
- [19] M. Z. Hasan, K. Z. Hasan and A. Sattar, "Burst header packet flood detection in optical burst switching network using deep learning model," *Procedia computer science*, vol. 143, pp. 970-977, 2018.
- [20] J. Chen, *et al.*, "DAD-MCNN: DDoS attack detection via multi-channel CNN," *In Proceedings of the 2019 11th International Conference on Machine Learning and Computing*, pp. 484-488, 2019.
- [21] S. Gamage and J. Samarabandu, "Deep learning methods in network intrusion detection: A survey and an objective comparison," *Journal of Network and Computer Applications*, vol. 169, p.102767, 2020.

- [22] I. Goodfellow, Y. Bengio, A. Courville, "Deep learning, MIT Press, Cambridge Google services resume after massive gmail," *youtube outage*, 2016. Available on: <https://www.livemint.com/technology/apps/google-services-youtubegmail-google-drive-face-outage-11607947475759.html>. 18 Apr 2021, Accessed on: 2021.
- [23] S. Hochreiter and J. Schmidhuber, Long short-term memory, "*Neural computation*, vol. 9, no. 8, 1735-1780, 1997.
- [24] A. Özgür and H. Erdem, "A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015," *PeerJ Preprints*, 2016.
- [25] B. A. Khalaf, *et al.*, "Comprehensive review of artificial intelligence and statistical approaches in distributed denial of service attack and defense methods," *IEEE Access*, vol. 7, pp. 51691-51713, 2019.
- [26] M. Bakro, *et al.*, "Building a Cloud-IDS by Hybrid Bio-Inspired Feature Selection Algorithms Along with Random Forest Model," *IEEE Access*, 2024.
- [27] H. Zhao, , L. Han and W. Wang, "Detection of COVID-19-related Malicious Domain Names Based on Feature Fusion," *In 2023 26th International Conference on Computer Supported Cooperative Work in Design (CSCWD, IEEE*, pp. 1251-1256, 2023.
- [28] S. Wang, *et al.*, "KRTunnel: DNS channel detector for mobile devices," *Computers & Security*, vol, 120, p. 102818, 2022.