SDN Architecture for Smart Homes Security with Machine Learning and Deep Learning

Wesam Abdulrhman Alonazi[1], Hedi HAMDI[2]
Department of Computer Science, College of Computer and Information Sciences, Jouf University, Aljouf, KSA[1, 2]

Nesrine A. Azim[3]
Department of Information Systems & Technology, Faculty of Graduate Studies for Statistical Research
Cairo University, Cairo, Egypt[3]

A. A. Abd El-Aziz[4]
Department of Information Systems
College of Computer and Information Sciences
Jouf University, Aljouf, KSA[4]
Department of Information Systems & Technology
Faculty of Graduate Studies for Statistical Research
Cairo University
Cairo, Egypt[4]

***Abstract*—In recent decades, Intelligent home systems are popular because they improve comfort and quality of life. A growing number of homes are becoming "smarter" by incorporating Internet of Things (IoT) technology to improve comfort, energy efficiency, and safety. Increases in resource-constrained IoT devices heighten security threats and vulnerabilities connected with them. Using SDN and virtualization, the IoT's size and adaptability can be managed at a lower cost than ever before. Using these intelligent security solutions, we can achieve real-time detection and automation for attack detection and prevention using artificial intelligence. Consequently, a large variety of solutions utilizing machine learning and deep learning have been developed to mitigate attacks on the IoT. Thus, the goal of this work is to use machine learning and deep learning to defend smart homes with SDN-based. We have designed smart home environments using Software-Defined Networking and Mininet that provide Instant Virtual networks for IoT in smart homes. Two datasets were used in this work: the first SDN dataset, which we acquired from smart homes by launching real attacks and creating normal traffic, and the second IoTID20 dataset, which is publicly available online. On both datasets, conducted ML and DL experiments. The best accuracy on SDN Dataset was 99.9% using Xgboost classifier, and on IoTID20 was 98.9% LSTM in binary classification, and ANN 85.7% on multiclass.**

Keywords*—SDN; smart home; security; machine learning; deep learning

I. Introduction

The Internet of Things (IoT) paradigm attracted a lot of attention from the scientific and business communities throughout the last decade [1]. Basically, such a technique is responsible for transferring massive data over a large network without the need for human intervention. The environment of the IoT consists of a huge number of smart devices that communicate with one another for sending and collecting massive sensing data, including wearables devices, sensors, actuators, mobile phones, smart home devices, etc. In 2021, the total number of installed smart devices in the world was 35.82 billion devices, while the total number of smart devices might increase to 75.44 billion by 2025 [2]. An environment will change the way we work, communicate, and interact with one another.

At the same trend, IoT devices in the Smart Home provide some facilities, such as monitoring temperature, air condition, humidity, gas leakage, and fire. The IoT devices furthermore can be used in smart Refrigerators to automatically check the available or missing foods and to automatically lock and open doors. The mentioned facilities are just some of many that can be provided by the IoT devices in the smart home environment. Therefore, the Internet of Things (IoT) is the core of Smart Homes that provides electronic, sensor, software, and network connectivity within a home. Smart home technology has revolutionized human life by allowing access to information and services from anywhere and at any time [3]. On the other hand, some limitations existed in the Smart Home environment because of using IoT devices. The IoT devices have limited resources and are vulnerable to various attacks, so much research has been investigated to lessen such attacks by taking into account the limited resources of the IoT devices. Machine and deep learning have been utilized to protect against various cyber-attacks in the Smart Home environment, such as distributed denial of service (DDoS) attacks, eavesdropping, Man in the middle attacks (MiTM), and many others. Furthermore, Software-Defined Networking (SDN) is a technique used for enhancing security in the Smart Home domain.

Large-scale IoT networks in smart homes provide new issues in areas including device management, data storage, transmission infrastructure, computation, and privacy protection due it has different connectivity methods, transmission protocols, architecture, and applications that make it a security and privacy challenge so have been extensively researched to improve it. Various types of attacks come from different resources due to the lack of security mechanisms in the architecture of smart homes such as attacks on application layer-based, attack network layer-based, attack perception layer-based, and attack on middleware layer.

Machine learning algorithms are support vector machine (SVM), Random Forest (RF), and XGboost. For deep learning algorithms, we use Artificial neural networks (ANN), Long

short-term memory (LSTM), and Convolutional neural networks (ConvNet/CNN). The main contributions of this work are as follows:

- We built a suitable framework that integrates machine learning or deep learning with SDN, which makes the framework able to detect and prevent various type of attacks like DDoS, Man-in-the-middle, Jamming, and SQL injection.
- Designed a smart home system to provide security at a low cost based on minicomputer Raspberry Pi and Zodiac-Fx OpenFlow Switch.
- The effectiveness of the proposed technique was verified utilizing a recent publicly available dataset.
- We selected an appropriate SDN controller that was compatible with our needs and implemented simulation for the proposed system and tested the result.

The remainder of this paper is formulated as follows. Section II reviews related works. Section III is the proposed methodology. Section IV presents the discussion of the experiment results. Finally, Section V presents a conclusion of our work and highlights our future research directions.

II. Related Work

In [4], X. Huang et al. have proposed a Deep Reinforcement Learning paradigm, which combines two algorithms; Deep Learning and Reinforcement Learning which is based on learning by the concept of trial and error. This paradigm is proposed to be applied in SDN-based IoT networks to control the data overflow. This paradigm consists of three components, the environment, the SDN-enabled network, and the agent, which is the central controller of the network where Deep Reinforcement Learning is applied. The third component is the customer, who provides feedback regularly, where the agent responds immediately and controls the flow of the data accordingly to meet the customer's satisfaction. Their experimentation shows that this paradigm significantly enhanced the Quality of Experience of users compared to previously proposed studies. The simulated network topology was built using OMNET++, which is consisted of 65 nodes and 108 links, and the resulting confidence interval was 95%.

Another work conducted by G. Stampa et al.[5], also utilized the same concept of applying Deep Reinforcement Learning, but for the sake of optimizing the traffic routing, by learning on the go the normal and abnormal behavior of the flow of the network and keeping reconfiguring the network settings accordingly. For simulating the network topology, OMNET++ was used, and as stated by the authors, the results indicate a better user experience within SDN-based IoT networks in terms of network performance optimization.

In [6], T. Tang et al. have proposed a deep learning approach for SDN-based IoT networks to work as flow-based anomaly detection. Their proposed approach utilized the Deep Learning Network in Open Flow Controllers as the central network controller, where they applied it to a dataset called NSL-KDD. They worked on features selection and extraction, where they selected only six out of forty-one features, namely: Duration, Protocol-type, Src-bytes, Dst-bytes, Count, and Srv-count. After that, they trained and tested the controller against common networking attacks. For the performance analysis, they used the following evaluation metrics: precision: 83, Recall: 75, and F1 score: 74, where the learning rate was set to 0.0001. for the accuracy, it was 75%, while the loss was 20%, compared to previously implemented algorithms such as Naïve bias, Random Forest, multi-layer perception, and support vector machine.

In [7], M. Ahmed et al. have proposed a mitigating DDoS attacks approach in SDN-based IoT networks, using machine learning algorithms such as Support Vector Machine, Gaussian Mixture Model, and Artificial neural networks. These algorithms are applied to the central controller of the network. The dataset used was a real dataset captured from the traces of a real network, where it contained 80,000 TCP connections, and these connections are split into three classes, 87% for HTTP, 6% for FTP, and 7% for attacks. The features of the data set are only three, which are being selected for training and testing the model that will classify the traffic of the network as either benign or malicious. The three features are the Total number of packets, Connection Duration Time, and the ratio of source and destination bytes. To prove the feasibility of their proposed approach, simulations were conducted by creating a prototype that utilizes the Dirichlet process mixture model for two web protocols, HTTP and FTP.

In [8], A. Al-Zahrani and M. Alenazi have proposed a machine learning-based Intrusion Detection system for SDN-based IoT networks. In their experimentations, they used the dataset NSL-KDD, and they selected only five features out of 41 features, namely: Duration, protocol type, Src-byte, Srv-count, Dst-host-same-src-port-rate. For the machine learning algorithms, they utilized the classical and advanced tree-based algorithms such as decision tree, random forest, and XGBoost, where they applied these algorithms at the OpenFlow switches to analyze the network traffic against network attacks such as Denial of service attacks, User to Root attacks, Remote to Local attacks, and probe attacks. The used dataset was split for training and testing, where the number of samples was 125.973 for training and 22.554 for testing. The results gave an accuracy of 95.95%, which indicates a high classification rate.

In [9], Q. Niyqz et al. have proposed a DDoS Detection System in IoT-based Software Defined Networks (SDN) using a stacked autoencoder as the deep learning algorithm used to build a classification model to analyze the network traffic against DDoS attacks. The dataset contains regular internet traffic with DDoS attacks in the protocols TCP, UDP, and ICMP. For performance analysis, all evaluation metrics were measures such as precision, recall, f1 score, and accuracy, where the last gave a value of 95%, indicating a high rate of classification that is suitable to be used in real networking environments. Moreover, an average computational time was computed where the training time took 524s, while the classification time took 0.0835s.

In [10], S. Sen et al. have proposed a DDoS detection system for SDN-based IoT networks using a Machine learning approach in the central controller of the network. For the experimentations, they applied a virtual SDN environment by

using the Adaboost decision stump to train the model on a dataset that contains private networking data. Their experimentation gave an accuracy value of 93%, with a low false-positive rate.

In [11], Y. Qin et al. have proposed an anomaly detection scheme based on machine learning algorithms to be applied in IoT-based SDN networks. An ensemble deep learning algorithm was used combining Convolutional Neural Network and Recurrent Neural Network, where this ensemble algorithm is used to deeply learn the main features of the traffic being analyzed against malicious attacks. The proposed method was implemented using Graphic Process Unit enabling a Tensor-Flow as the main controller of the SDN-based network. The experimentation was conducted on three datasets, namely: CTU-13, CSIC, and Sim_data. For performance analysis, evaluation metrics were computed, such as accuracy, recall, and F1-score, where the accuracy given was 99.86%.

In [12], S. Mohammed et al. have proposed a DDoS Detection system based on machine learning in an SDN-based IoT network to protect such networks from such threatening attacks. The machine-learning algorithm used is Naïve Bayes, and the data set was NSL-KDD. Precision, Recall, and F1 score were calculated for performance evaluation measurements.

In [13], S. Dey et al. have proposed an anomaly detection system based on machine learning to be also applied in the OpenFlow controller of the SDN-based IoT networks. The machine-learning algorithm that was used is the Random Forest Classifier, which was applied to NSL-KDD dataset that contained 41 features. These features were processed for features selection and extraction using automatic tools such as Info Gain, Gain Ration, and CFS Subset Evaluator. The resulting accuracy is 82%.

In [14], J. Wu et al. have proposed a network link congestion prediction methodology for predicting whether the links are congested or not in SDN-based networks. The proposed methodology used machine learning to learn the information that features network congestion. During their implementation, they simulated a network topology that consists of 12 hosts and 13 switches, using a Mininet Simulator, and produced 559929 records of data being recorded from this simulated network and saved as the dataset to be preprocessed and learned to be used later for congestion classification. For the learning process, four algorithms were used, SVM, MLP, KNN, and IDCNN, where the last one gave the highest accuracy of 98.3%.

In [15], T. Tang et al. have proposed an Intrusion Detection System based on Machine learning for SDN-based networks to classify the flow of data in the network into legitimate and anomaly to check against many types of attacks such as DoS, R2L, U2R, and Prob. They used the dataset NSL_KDD and two learning algorithms, namely: Gated Recurrent neural network and a Fully connected deep neural network. For performance analysis, they got 90% for GRNN. In addition, they evaluated the performance of their intrusion detection model over their simulated network in terms of resource utilization, throughput, and latency.

In [16], B. Susilo and R. F. Sari have proposed an Intrusion Detection Mechanism using Machine and Deep learning in SDN-based IoT networks, where their proposed methodology consists of three tiers, namely: Data plane that is responsible for the packet flow in a network, Control Plane that is responsible for network configuration and management, and Application plane that is responsible for the end-user applications. For implementation, they used three virtual machines: Kali Linux VM, Minimet VM, and Metasploittable VM, all installed on a Windows OS. For controlling the dataflows to classify and detect anomalies and network attacks, the Random Forest algorithm was used as a machine learning algorithm, and the Convolutional Neural network as a deep learning algorithm; both algorithms were applied to two datasets, BoT-IoT to be used for classifying Botnet Attacks, and CIC-IDS-2018 for classifying Intrusidion attacks. Experimentations gave accuracy values of 90% for Bot-IoT and 99.95% for IDS-2018 when using CNN.

In [17], M. M. Raikar et al. have proposed a data traffic classification using supervised machine learning algorithms in SDN-based IoT networks, such as Support Vector Machine, Nearest Centroid, Gaussian Naïve Bayes (NB) over a dataset containing traces from a real network. The experimentation results gave accuracy values of 92.3% for SVM, 91.02% for nearest centroid, and 96.79% for NB.

In [18], Y. Handa and A. Mudanna have proposed an intrusion detecting model using deep learning to be applied in SDN-based IoT networks. As the network is split into three tiers, the data layer, controller layer, and application layer, the proposed model will be applied in the data layer of the network to check the data flow and classify it and filter it against anomalies and malicious attacks such as DoS attack, Probe Attack, User to root attacks (U2R), and Remote to Local attacks (R2L). For deep learning, Convolutional Neural Network was used first to do the feature extraction and then to do the learning and classification. The dataset used was called KDD 99, and the learned features were six; Duration, Protocol-Type, Src_bytes, Dst_bytes, Count, and Srv_Count, where the data being examined in these features are learned to classify the data in packets flow into normal or malicious and hence detect intrusion attacks in SDN based networks.

The authors in [32] suggest a deep learning classifier-based SDN-based intrusion detection system for the Internet of Things. The suggested study is carried out in a simulated setting, with test accuracy of IDSIoT-SDL compared using a number of different matrices. From previous studies, gaps in IoT security solutions are costly and occur during a realistic attack environment. So, it is of the utmost need to develop low-cost solutions that can detect and prevent cyber-attacks in simulation environments that are carried out against or through IoT devices in real attacks.

III. Material and Methods

A secure, adaptable architecture is proposed in this section for intelligent homes and is based on where monitoring network traffic. SDN architecture is examined from a security standpoint in this study. The proposed intelligent system for detecting attacks based on machine learning deep learning and utilizing SDN for security purposes SDN will be the suitable

technique for implementing SDN-based smart homes architecture. The proposed smart system uses the recent technique to classify network traffic and prevents attacks.

The proposed system consists of software or hardware that monitors traffic in real-time to detect and prevent security breaches as shown in Fig. 1. The system architecture consists of three layers: the First Layer sensor or data flow layer, the Second is the OpenFlow layer, and the third is the SDN controller. In our system, the Raspberry Pi will be used as a gateway for connecting sensors that use different protocols to the Internet via Wi-Fi, Ethernet, or Zigbee, Bluetooth, and cellular communications via protocol translation. OpenFlow switch called Zodiac-OpenFlow FX's it represents the second layer to forward data and control IoT device: There are four physical ports on Zodiac-OpenFlow FX's switch: port 4 for connecting the controller to the switch, and ports 1 to 3 for connecting switches with each other [19]. The controller in the system will be Raspberry Pi installed inside it ubuntu OS with Ryu controller. Data processing and forwarding are both under the control of the controllers.

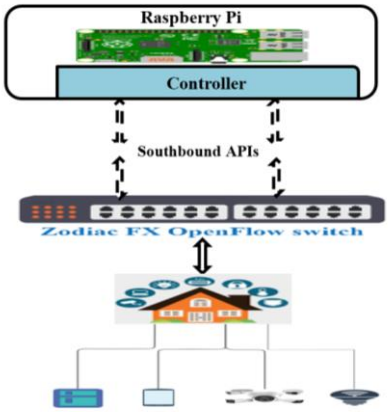

Fig. 1. Proposed System.

A. SDN Controller

SDN controller was used as middleware or as framework for our system. There are a lot of type of controllers such as ONOS, Open Daylight Controller, Ryu. In our study, we have used Ryu for high traffic and network agility, so the Ryu controller is the best choice in our study. APIs and well-defined software components were incorporated into Ryu. Python was used as the coding language of choice for the Ryu game controller. We suggested installing Ryu on Raspberry Pi 4, which will minimize the cost. A decision is made by the controller about how each flow is forwarded. The data plane's flow table stores the decision, which can be made either reactively to newly discovered flows or proactively in the flow table. The controller function as the following:

- Data collection by Information snooper based on statics features for each type of attack this snooper is a python script.
- Send a request to open for the switch to start collecting the static feature from data table entry via southbound API.
- Extract some features related to attacks like MITM DDoS attacks.
- Select the critical feature for attacks that were predefined previously, like packet size, number of hops, etc.
- Automatically call the Machine learning or deep learning classifier to identify if traffic is abnormal; if yes, add a flag to the header in the data flow entry to take any action to block the host and drop the packet.

B. Proposed System Simulation

The simulation and experimentation will be described in the following part. It is possible to prototype and conduct research quickly and easily with Ryu Controller's python-based fork of Ryu Controller, thanks to the language's ease of learning and short development cycle. We created a virtual network using the Mininet OpenFlow network to test our machine learning algorithm shown in Fig. 2.

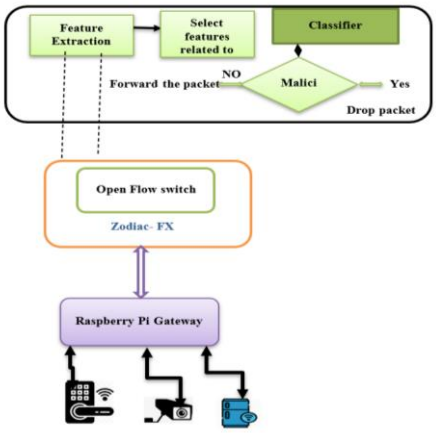

Fig. 2. System Scenario.

C. Mininet OpenFlow Emulator

Mininet is a network emulation tool used to create a network of virtual hosts, links, and switches. To facilitate software-defined networking and bespoke routing, the Linux distribution provides Networking software for Mininet hosts. Tir switches support OpenFlow [20]. Our study used this simulator to test the proposed system and its effectiveness in providing a secure environment for the Internet of Things devices that make up intelligent homes. This system can detect cyber-attacks with high accuracy and prevent them in real-time. We have created 20 virtual hosts representing the IoT devices in smart homes and 20 OpenFlow switches, as shown in Fig. 3.

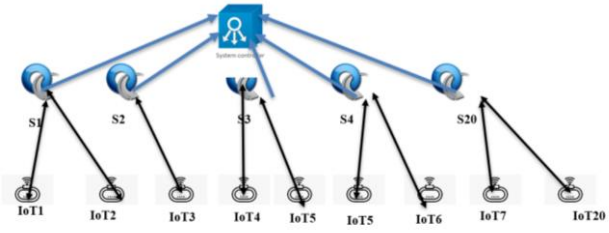

Fig. 3. Mininet Topology.

1) Triggering attack: Smurf attacks are distributed denial-of-service attacks in which huge numbers of Internet Control Message Protocol packets with the target victim's faked source IP are broadcast to a computer network using an IP broadcast address." In our experiment, we have triggered this type of attack for purpose traffic generation and collection. Parallelism is applied in traffic generation via the usage of the threading module, which allows several threads to execute at the same time. To produce faked IP addresses for non-existent nodes that are then used to flood targets, the rant function of the Random module is utilized, as is the random module. 0-255 is the range of the first, second, third, and fourth octets. It is also possible to utilize the OS module to allow. To specify the target IP address, port, and even the SYN TCP flag for the destination port. Hping tool has been used to generate this type of attack.

2) Ryu controller: Ryu is an open-source controller Python-based. Ryu can support a wide range of protocols. Ryu provides a well-defined API for software components. Network management and control apps may be quickly developed by developers [21]. RYU controller has been installed on a VMware machine that contains ubuntu OS. SDN is the system's brain to manage open flow switches and hosts. This controller works by sending a flow stats req message to the OpenFlow switch and asking the controller to send a flow stat to reply message back. Requests for data from the Ryu controller module are made every five seconds, which was considered the best time frame for detecting assaults early enough to safeguard the controller. There is a "Flow Removed" message that the switch sends the controller when a flow entry is deleted, either because of a hard timeout or because of a period of inactivity. Only if the "OFPFF SEND FLOW REM" flag is set in the Flow MoD message will the "Flow Removed" message be delivered. FlowMod messages from the controller may also remove flow entries, resulting in the "Flow Removed" message. It is possible that "Flow Removed" is caused by a timeout, which may be either an idle timeout, which occurs when there are no matching packets for some time or a hard timeout, which occurs when there is a predetermined amount of time remaining.

D. Data Processing and Models

1) Dataset collection: We have used two datasets; the first one is called the SDN dataset. It has been collected by training real attack on the SDN environment, and the second one it's available publicly on the internet for research purposes.

For SDN dataset, the data was captured using the Wireshark app as (pcap). This monitored all ports and captured the incoming and outgoing data in all ports with different protocols. The Wireshark program (pcap) captured the data by listening to all ports and capturing the incoming and outgoing data in multiple protocols. Protocols, flow duration flags, byte count per Second, and the source and destination IP port numbers. Python controller module accepts the flow monitor's data and extracts it into a format that the trained model can use for prediction. Then we utilize entropy (statistics features) to estimate the distribution of these variables and train the model using normal and atypical traffic data for smurf or DDoS assault.

In this work, we have collected by triggering attacks and normal traffic. The Attack was performed on the Mininet host that represents Internet of things devices. The Attack is implemented using the Hping tool and "pingall "command and Normal traffic is collected by ping command between each device. Then data is collected as a CSV file on the SDN controller (RYU) that enables the socket to listen on all ports on the Open flow switch that is included on Mininet topology as shown in Fig. 8. The process iterated till we reached the required size of data. the data size reached above 250 MB that size is enough to train the model. The Normal traffic collection process shown in Fig. 4 and the attack traffic shown in Fig. 5.

For IoTID20 Dataset, we have used the IoTID20 dataset, collected by Imtiaz [22]. This dataset was collected using the SKT NGU and EZVIZ Wi-Fi cameras, which are smart home devices. The dataset contains 86 features; after pre-processing the features have been reduced to 80 features. We performed pre-processing; First, we removed all nulls and invalid values, then we labelled and scaled the data using a stander scaler, label encoder, and a data splitting using K-fold to 10 groups to combat overfitting in the second experiment splitting data by ratio of 25 %for testing and 75 for training.

2) Data Pre-processing: Data pre-processing is the process of transforming unstructured data into a form that can be read, accessed, and analyzed. The raw data cannot be used in data mining; therefore, this is a critical step. Check the data quality before using machine learning or other methods. The primary purpose of data pre-processing is quality control.

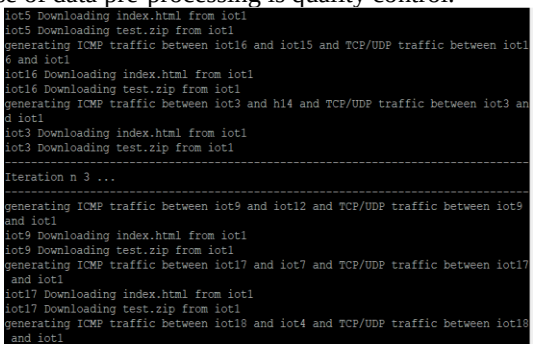

Fig. 4. Normal Traffic Collection in Mininet.

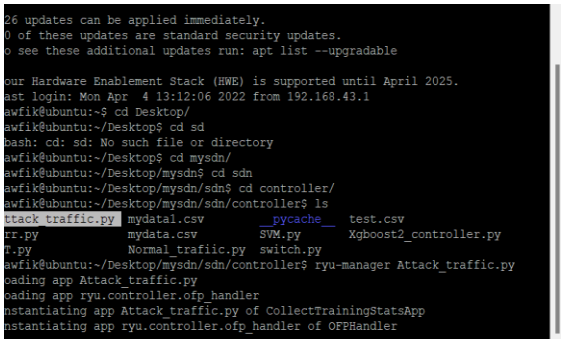

Fig. 5. Attack Traffic Collection in Mininet.

The following are ways to evaluate the level of quality to ensure that the data entered is accurate.

- Checking for completeness: determining whether the data has been captured.
- To ensure that all the data is maintained in the same location, one must ensure that all information is consistent.

3) Data normalization: Normalization is generally required when we are dealing with attributes on different scales because, if it is not done, it may lead to a dilution in the effectiveness of an important equally important attribute on a lower scale because of another attribute having values on a larger scale. Normalization is generally required when we are dealing with attributes on different scales. When there are several attributes, but those attributes have values that are measured on different scales, this can result in inadequate data models when executing data mining activities. Therefore, they must be normalized so that all the properties may be measured on the same scale.

4) Feature extraction: The goal of feature extraction is to cut down on the total number of features in a dataset by generating new features based on the ones that are already there and skipping the original features. After being trimmed down, this new collection of features ought to be capable of summarizing most of the information that was included in the initial set of features. By combining the components of the original set in this way, it is possible to generate a condensed version of the original features.

5) Feature selection: The act of selecting the features that will most significantly contribute to the prediction variable or output that you are interested in can be done either automatically or manually and is referred to as "feature selection." Feature selection reduces the amount of predictive model input variables. Reducing the amount of input variables can minimize modelling costs and increase model performance. Statistical-based feature selection approaches evaluate each input variable's relationship to the target variable and select those with the strongest association. The choice of statistical measures depends on the input and output data types.

6) Machine learning models: The project model's primary goal in the IoT-smart homes-based collection is to detect whether a traffic flow is normal or abnormal based on learning from an existing labelled data. The flow-based detection module needs a model that can distinguish an attack from a regular flow to work.

7) Xgboost: Extreme Gradient Boosting (Xgboost) Learning techniques are used to transform weak models into strong estimators that function sequentially as circular iterations [23]. As in neural networks, each one seeks to rectify a mistake in its preceding model in order to minimize the loss function. There are several ways to increase your model's performance, but Gradient Boosting is one that focuses on fitting the new predictor to fit on the pseudo-residuals (errors) of its predecessor, rather than modifying the weights for each wrong classification at each iteration. In order to attain the best possible accuracy, ensemble machine learning uses a variety of ML models. An ensemble machine learning methodology for classification, regression, and ranking problems, XGBOOST is a current ensemble machine learning method. Fast and efficient implementation of Gradient boosting structure to identify the best tree models. On August 1st, 2016, Tianqi Chen and Carlos Guestrin launched the project. Uses Gradient Boosting to discover the best and most accurate tree model, emphasizing computation speed. Parallelization, Cache Optimization, Distributed Computing, and processing a vast dataset are all supported by XGBoost.

We have used this algorithm in our study. After training the model, we tested the pre-trained model's performance on actual attacks on IoT devices in an intelligent home virtual environment using the Mininet with SDN controller. The captured data from the OpenFlow switch passes through this classifier to identify if traffic is benign or malicious. If the traffic is malicious, add a flag to the packet in SDN to take any action and send a request to open the flow switch to drop it, as shown in Fig. 6.

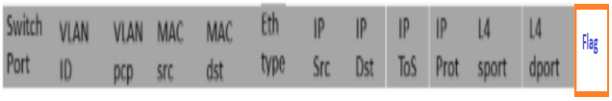

Fig. 6. Packet with Flag.

8) Support Vector Machines (SVM): SVM is a set of supervised learning techniques applied for classification or regression. SVM classifier has been adopted in our model for intelligent home security. A key objective of the SVM technique is to find the optimum decision boundary or line that can divide n-dimensional space into classes [24], allowing us to classify new data points quickly. A hyperplane denotes the border of optimal decision-making inside a problem domain. To create the hyperplane, the SVM uses the most extreme points/vectors in the dataset. So-called "support vectors" refer to these extreme circumstances, and the process is known as a Support Vector Machine.

9) Random Forest (RF): It is commonly utilized in Classification and Regression issues as a supervised Machine Learning Algorithm. It creates decision trees on various samples and uses their majority vote to classify and average the data[25, 26]. The Random Forest Algorithm's ability to handle data sets with continuous variables, such as regression, and categorical variables, such as classification, is critical. When it comes to categorization challenges, it does a better job.

10)ANN classifier: Computerized Brains ANN is a computer system component that attempts to simulate the human brain by personifying the nervous system. Many neurons link to one other. Every node or neuron in an artificial neural network follows the same paradigm for receiving information. An artificial neural network (ANN) consists of input neurons, weights, biases, hidden layers, activation functions, and an output layer that can be fed-forward or back-propagated.

11)Long short-term memory: Schmid Huber and Hoch Reiter proposed the LSTM deep learning algorithm in 1997 as a variant of the RNN network. It's made to store previously entered data. Classification and forecasting of time series are both possible applications most frequently used on Sequential data, such as speech recognition, translation, and picture captioning, which are well-suited to LSTM. LSTM is a good choice for applications related to the period. It types a back-propagation neural network. Forget, input, and output are the three gates of the LSTM [27].

The input gate picks the appropriate value from input samples to adjust the cell's memory and input. For three gates, the Sigmoid function is employed as a combining function that returns a value between 0 and 1. Calculated values are given additional weight by the Tanh or Relue functions. Sigmoid is a forget gate that decides whether new information should be stored in the cell state or discarded. Sigmoid input gate layers pick Tanh and Relue to update next. The next concealed state that should be remembered when the output gate is closed is selected. The concealed state decides what data should be preserved for the following phase and the hidden state is used for predictions.

We will use two layers of LSTM with 200-500 neurons and various optimizers, such as Adam, Rmsprob, Nadam, -SGD, loss sparse categorical cross-entropy, Activation Relu, and Activation Relu. Dropout = 0.2 or 0.5, 6 Dense layers with varying values, SoftMax for towing and four classes.

12)Convolutional neural network: ConvNet/CNN) is a deep learning system that can feed an image or numerical data and apply a filter, adding adjusted weights and minimizing biases to various objects in the picture and be able to discern one from the other. This approach requires less pre-processing than previous classification methods. Rather than relying on hand-engineered or traditional technique filters, CNN may be trained to learn these filter characteristics over time. The arrangement of the Visual Cortex served as inspiration for the design of the CNN architecture, which is similar to the neuronal connection pattern found in the human brain [28]. A visual field area known as the Receptive Field is where neurons respond to visual input. The whole visual field is covered by a group of such fields that overlap. Neurons in convolutional neural networks are geometric functions that calculate the weighted sums of numerous inputs, output and initiation values. The first layer of CNN commonly detects essential visual characteristics. Data is then passed from one layer to the next, allowing for increasingly complex features each time. The layers of CNN can recognize more complex characteristics, such as faces, objects etc.

IV. Experimental Results

In this experiment, four machine learning algorithms have been used KNN, Xgboost, SVM, and random forest. Each classifier is trained to classify the IoT traffic as either normal or abnormal. The binary classification shows excellent results in detecting the attack in passive mode and real-time mode, the result is shown in Table I.

TABLE I. Performance Metrics in Machine Learning

Classifier	Accuracy	Confusion Matrix
SVM	99.8	[[6860 1] [0 217712]]
RF	99.9	[[6861 0] [1 217711]]
XGBOOST	99.9	[[6861 0] [0 217712]]
KNN	99.7	[[6861 2] [0 217703]]

1) Artificial Neural Network (ANN): Artificial neural network model consists of three layers with 100 neurons for each layer, input shape 78, and RELU Activation function. And sigmoid for binary classification. The same architecture has been used on multi-classification but with differences in last layer SoftMax for five classes. The model was trained with 200 epochs by the Adam optimizer. the obtained accuracy reached 98.9 for binary classification and 83.2 % for multi-classification as shown in Fig. 7 and Fig. 8. Confusion matrix of ANN binary classification shown in Fig. 9.

In the LSTM model the architecture consists of two layers with 200 neurons for first one and 50 neurons for second. The next layers are dropout with 0.2, four dense layers with activation Relu, and the last layer is softmax for two classes. The model trained on binary classes and multiclass with five classes on different optimizers. The best result was obtained for the binary classes with an Adam optimizer, 200 epochs, and batch size 3000. The performance of the two models for LSTM shown in Fig. 10 and Fig. 11. The accuracy reached to 99% on binary classification and 84.7% for multiclassification. Confusion matrix of LSTM with multi classification shown in Fig. 12.

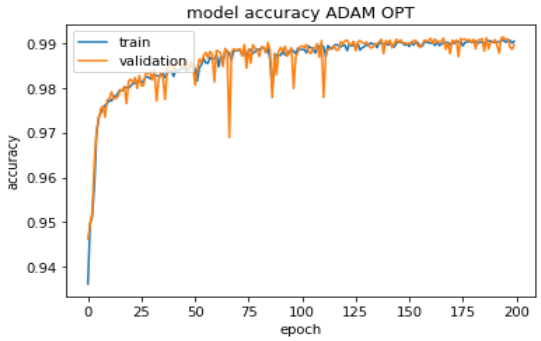

Fig. 7. Accuracy ANN Model with Binary Classification.

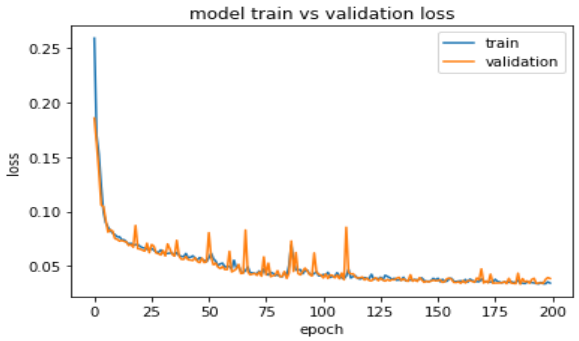

Fig. 8. Loss for Train vs Validation with Binary Classification.

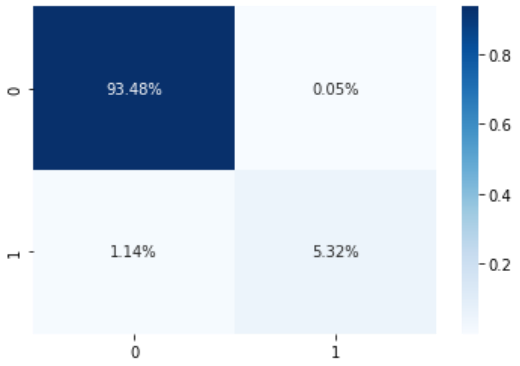

Fig. 9. Confusion Matric for ANN Model with Binary Classification.

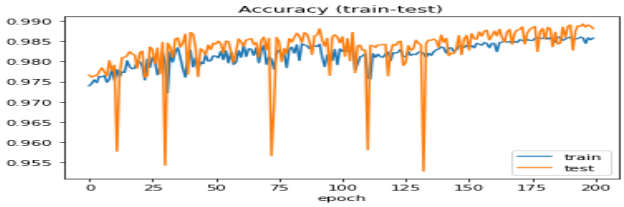

Fig. 10. Accuracy on Two Classes with Adam Optimizer.

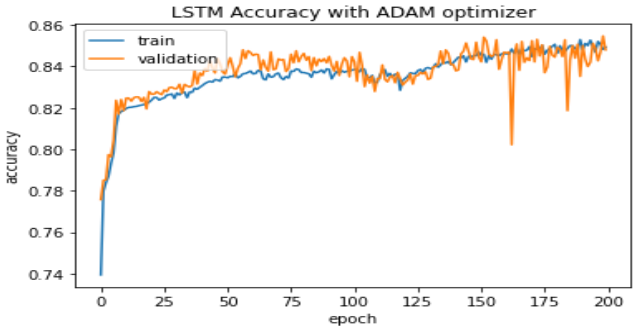

Fig. 11. Loss for Train vs Validation with Multi Classification.

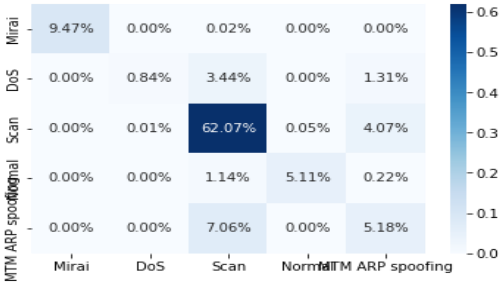

Fig. 12. Confusion Metric for LSTM Model with Multi Classification.

V. Discussion

After running an experiment, we triggered attacks for data collection purposes and trained three Machine Learning algorithms on the training set. In the Software-Defined Networking (SDN) environment we have made simple attacks to test the machine learning models in real-time. The second section shows the result deep learning experiment. The evaluation of the two experiments on two datasets is shown below. The result shows an excellent performance to detect the attack on IoT devices on smart homes that have been simulated on SDN and open virtual open flow switches and (Hosts) that represent IoT devices. Five machine learning classifiers have been used; the highest obtained accuracy achieved by Xgboost classifier. In the second experiment on IoTID20. Several experiments have been conducted using machine learning and deep learning:

- On binary classification two classes normal and attack.
- Multiclassification for five classes ("Mira, Dos, Scan cat, Normal, MTM ARP Spoofing").
- Binary classification with 5-k fold cross-validation and 10-k fold cross-validation.
- Multiclassification with 5-k fold cross validation and10-k fold cross-validation.

The result shows the superiority of deep learning in detecting electric attacks, Internet of things devices in smart homes as shown in Table II and Table III. The output has been evaluated using accuracy, precision, f1-score, Sensitivity, and Specificity. Two models have been implemented for each neural network. The evaluation on stander data splitting using traditional method 70% and 30 % is shown in Table IV. The second experiment conducted by splitting date to 10-k fold groups the result was better than 5-k fold groups is shown in Table IV. According to the results of our model compared to the results of previous studies shown in Table III, the proposed model achieved high accuracy results and it is recommended to apply it to actual discovery in IoT environments.

TABLE II. PERFORMANCE METRICS USING 70-30% SPLITTING

Model	Accuracy	Sensitivity	Specificity	F-Score Class0	F-Score Class1	F-Score Class2	F-Score Class3	F-Score Class4
ANN 2classes	98.9	84.9	99.9	0.99	0.91	-	-	-
ANN 5classes	83.2	85	99.2	0.99	0.25	.88	.76	.52
LSTM 2classes	98.9	83.3	99.8	0.99	0.90	-	-	-
LSTM 5classes	83.3	100	99.9	1.00	0.27	0.89	0.78	0.51
CNN2classes	98.6	86.8	99.5	0.99	0.90	-	-	-
CNN5classes	85.7	100	99.9	1.00	0.29	.90	.77	.53
SVM 2classes	96.5	83.4	99	.99	86	-	-	-
SVM5 classes	79.2	80	99	.99	0.26	.87	.75	.49
Xgboost 2classes	97.2	83	99.8	.99	.92	-	-	-
Xgboost 5classes	81.3	82	99	98	.27	.88	.76	.50
RF 2classes	96	83.4	99	.99	88	-	-	-
RF5classes	80.1	82	98	.98	.26	.85	77	.51
KNN2classes	97	84.4	99	.99	89	-	-	-
KNN5classes	77.5	92	80	.98	.24	.85	.73	.38

TABLE III. PERFORMANCE A COMPARISON OF MODEL WITH OTHER ML & DL MODELS FOR BINARY AND MULTI CLASSIFICATION

Ref	Algorithm	Accuracy in binary class %	Algorithm	Accuracy in multi class %	Dataset used
[29]	Xgboost	98.89	Xgboost	93.48	IoTID20
[30]	DT	94.22	ANN	96	Bot-IoT and the IoTID20
[31]	LR	91	ANN	96	IoTID20
[32]	autoencoder models	94.70	autoencoder	95.20	NSL-KDD, IoTID20
[33]	DCNN	99.84	DCNN	98.12	IoTID20
Our model	**Xgboost**	**99.9**	**LSTM**	**98.9**	**IoTID20 and SDN dataset collected**

TABLE IV. K-FOLD SPLITTING DATA TO 10 GROUPS

Model	Accuracy	Sensitivity	Specificity	F-Score Class0	F-Score Class1	F-Score Class2	F-Score Class3	F-Score Class4
ANN 2classes	97.3	84.9	99.8	0.97	0.92	-	-	-
ANN 5classes	83.2	85	99.2	0.99	0.25	.88	.76	.52
LSTM 2classes	98.9	83.3	99.8	0.99	0.90	-	-	-
LSTM 5classes	83.3	100	99.9	1.00	0.27	0.89	0.78	0.51
CNN2classes	98.6	86.8	99.5	0.99	0.90	-	-	-
CNN5classes	85.7	100	99.9	1.00	0.29	.90	.77	.53
SVM 2classes	96.5	83.4	99	.99	86	-	-	-
SVM5 classes	79.2	80	99	.99	0.26	.87	.75	.49
Xgboost 2classes	97.2	83	99.8	.99	.92	-	-	-
Xgboost 5classes	81.3	82	99	98	.27	.88	.76	.50
RF 2classes	96	83.4	99	.99	88	-	-	-
RF5classes	80.1	82	98	.98	.26	.85	77	.51
KNN2classes	97	84.4	99	.99	89	-	-	-

VI. Conclusion

There is a rapid expansion of the Internet of Things (IoT) into previously unimagined areas and domains. In addition to factories, agriculture, cities, and transportation are included in this. In recent years, the Internet of Things (IoT) has grown in popularity as a growing number of applications and devices, such as medical devices, home sensors, wireless sensors, and other closely linked IoT gadgets, have been implemented. IoT security is sometimes overlooked since it takes time to fully assess a device's vulnerabilities. Widespread use and quick adoption of such technologies raises security concerns. Cyberattacks affect resource availability, causing financial and material damage. As a result, fresh strategies are needed to increase not only IoT's security against cyberattacks, but also their identification and mitigation from IoT networks. Smart-home security IoT is a big concern due to lack of computing power and device heterogeneity. IoT devices can't handle computationally intensive and latency-sensitive security tasks due to their low processing power. Our research aimed to defend smart home-based Internet of Things devices from hacking. It is of the utmost need to develop low-cost solutions that can detect and prevent cyber-attacks that are carried out against or through IoT devices. We thus proposed in this work a smart system to guard against attacks on smart homes utilizing Software-Defined Networking, Machine learning, and Deep learning at a cheap cost to prevent threats on devices from both the inside and the outside.

We have implemented the classification process of the IoT traffic in a simulation environment using Ryu SDN controller and Mininet for detection attacks on traffic using Machine learning classifiers; SVM, KNN, RF, Xgboost, we collected data by trigged real attacks on IoT devices and generated normal traffic also, then we have trained the models and test them or testbeds environment (SDN). The result showed the ability to identify the normal and abnormal traffic with accuracy reached 99.9 %. Then we introduced deep learning as proof of concept to detect the attack in a smart home environment, IoTID20 dataset has been used for training the deep learning models, ANN, LSTM, and CNN. The turning process was conducted on the binary classification to classify their traffic as normal or abnormal, and multi-classification to identify the attack type on IoT. The highest achieved accuracy for binary classification was 98.9 % using long short-term memory (LSTM), 85.7 % on multiclass using Convolutional Neural Network (CNN). Other deep learning classifiers can be investigated for use in the improvement of future work. The suggested model's simulation results can be put to the test in a live setting with both heavy attack and typical traffic.

Acknowledgment

The authors would like to thank the Deanship of Graduate Studies at Jouf University for funding and supporting this research through the initiative of DGS, Graduate Students Research Support (GSR) at Jouf University, Saudi Arabia.

References

[1] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[2] G. Spanos, K. M. Giannoutakis, K. Votis and D. Tzovaras, “Combining statistical and machine learning techniques in IoT anomaly detection for smart homes,” in Proc. IEEE 24th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), pp. 1–6, 2019.

[3] S. Al-Sarawi, M. Anbar, R. Abdullah and A. B. Al Hawari, “Internet of things market analysis forecasts, 2020--2030,” in Proc. Fourth World Conference on smart trends in systems, security and sustainability (WorldS4), pp. 449–453, 2020.

[4] X. Huang, T. Yuan, G. Qiao and Y. Ren, "Deep Reinforcement Learning for Multimedia Traffic Control in Software Defined Networking," IEEE Network, pp. 35 - 41, 2018.

[5] S. Mahindrakar and R. K. Biradar, “Internet of things: Smart home automation system using raspberry Pi,” Int. J. Sci. Res, vol. 6, no. 1, pp. 901–905, 2017.

[6] X. Huang, T. Yuan, G. Qiao and Y. Ren, “Deep reinforcement learning for multimedia traffic control in software Defined networking,” IEEE Network, vol. 32, no. 6, pp. 35–41, 2018.

[7] G. Stampa, M. Arias, D. Sanchez-Charles, V. Muntes-Mulero and A. Cabellos, “A Deep-reinforcement learning approach for software-defined networking routing optimization,” arXiv preprint arXiv:1709.07080, 2017.

[8] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi and M. Ghogho, “Deep learning approach for network intrusion detection in software defined networking,” in Proc. International Conference on Wireless Networks and Mobile Communications (WINCOM), pp. 258–263, 2016.

[9] M. E. Ahmed, H. Kim and M. Park, “Mitigating DNS query-based DDoS attacks with machine learning on software-defined networking,” in Proc. MILCOM, IEEE Military Communications Conference (MILCOM), pp. 11–16, 2017.

[10] R. H. Jensen, Y. Strengers, J. Kjeldskov, L. Nicholls and M. B. Skov, “Designing the desirable smart home: a study of household experiences and energy consumption impacts,” in Proc. CHI Conference on Human Factors in Computing Systems, 2018.

[11] Q. Niyaz, W. Sun and A. Y. Javaid, “A Deep learning based DDoS detection system in software-defined networking (SDN),” ICST Transactions on Security and Safety, vol. 4, no. 12, pp. 153515, 2017.

[12] S. Sen, K. D. Gupta and Md. Manjurul Ahsan, “Leveraging machine learning approach to setup software-defined network (SDN) controller rules during DDOS attack,” in Proc. of International Joint Conference on Computational Intelligence, pp. 49–60, 2020.

[13] Y. Qin, J. Wei and W. Yang, “Deep learning based anomaly detection scheme in software-defined networking,” in Proc. 20th Asia-Pacific Network Operations and Management Symposium (APNOMS), pp. 1–4, 2019.

[14] S. S. Mohammed, R. Hussain, O. Senko, B. Bimaganbetov, J. Y. Lee et al., “A new machine learning-based collaborative DDOS mitigation mechanism in software-defined network,” in Proc. 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), pp. 1–8, 2018.

[15] S. K. Dey, Md. M. Rahman and Md. R. Uddin, “Detection of flow based anomaly in openflow controller: machine learning approach in software defined networking,” in Proc. 4th International Conference on Electrical Engineering and Information & Communication Technology (iCEEiCT), pp. 416–421, 2018.

[16] J. Wu, Y. Peng, M. Song, M. Cui and L. Zhang, “Link congestion prediction using machine learning for software-defined-network data plane,” in Proc. International Conference on Computer, Information and Telecommunication Systems (CITS), pp. 1–5, 2019.

[17] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, M. Ghogho and F. el Moussa, “DeepIDS: Deep learning approach for intrusion detection in software defined networking,” Electronics (Basel), vol. 9, no. 9, pp. 1533, 2020.

[18] B. Susilo and R. F. Sari, “intrusion detection in software defined network using deep learning approach,” in Proc. IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC), pp. 0807–0812, 2021.

[19] A. O. Alzahrani and M. J. F. Alenazi, "Designing a network intrusion detection system based on machine learning for software defined networks," Future Internet, vol. 13, no. 5, pp. 111, 2021.

[20] Q. Niyaz, W. Sun and A. Y. Javaid, "A deep learning based ddos detection system in software-defined networking (SDN)," arXiv preprint arXiv:1611.07400, 2016.

[21] I. Mansour, "Towards effective live cloud migration on public cloud IaaS," Ph.D. dissertation, Bournemouth University, 2018.

[22] S. Y. Wang, "Comparison of SDN openflow network simulator and emulators: EstiNet vs. Mininet," in Proc. IEEE Symposium on Computers and Communications, 2014.

[23] S. Asadollahi, B. Goswami and M. Sameer, "Ryu controller's scalability experiment on software defined networks," in Proc. IEEE International Conference on Current Trends in Advanced Computing, ICCTAC, pp. 1–5, 2018.

[24] I. Ullah and Q. H. Mahmoud, "A scheme for generating a dataset for anomalous activity detection in IOT networks," Lecture Notes in Computer Science, vol. 12109 LNAI, pp. 508–520, 2020.

[25] T. Chen and T. He, xgboost: eXtreme Gradient Boosting, 2022, 2004. [Online]. Available: https://cran.r-project.org/web/packages/xgboost/vignettes/xgboost.pdf.

[26] L. Auria and R. A. Moro, "Support vector machines (SVM) as a technique for solvency analysis," SSRN Electronic Journal, vol. 1, no. 1, pp. 1-18, 2008.

[27] J. Frausto-Solís, L. J. Hernández-González, J. J. González-Barbosa, J. Paulo Sánchez-Hernández, E. Román-Rangel et al., "Convolutional neural network–component transformation (CNN–CT) for confirmed covid-19 cases," Mathematical and Computational Applications, vol. 26, no. 2, pp. 29, 2021.

[28] Wani, A., & Khaliq, R. (2021). SDN-based intrusion detection system for IoT using deep learning classifier (IDSIoT-SDL). CAAI Transactions on Intelligence Technology, 6(3), 281-290.

[29] Bajpai, S., & Sharma, K. (2022). A Framework for Intrusion Detection Models for IoT Networks using Deep Learning.

[30] Albulayhi, K., Smadi, A. A., Sheldon, F. T., & Abercrombie, R. K. (2021). IoT Intrusion Detection Taxonomy, Reference Architecture, and Analyses. Sensors, 21(19), 6432.

[31] Song, Y., Hyun, S., & Cheong, Y. G. (2021). Analysis of autoencoders for network intrusion detection. *Sensors*, *21*(13), 4294.

[32] Ullah, S., Ahmad, J., Khan, M. A., Alkhammash, E. H., Hadjouni, M., G, Y., Hyun, S., & Cheong, Y. G. (2021). Analysis of autoencoders for network intrusion detection. Sensors, 21(13), 4294.

[33] hadi, Y. Y., ... & Pitropakis, N. (2022). A New Intrusion Detection System for the Internet of Things via Deep Convolutional Neural Network and Feature Engineering. Sensors, 22(10), 3607.