Tikrit Journal of Engineering Sciences (2023) 30 (1): 119-129
DOI: http://doi.org/10.25130/tjes.30.1.12

IRAQI Academic Scientific Journals

College of Engineering

ISSN: 1813-162X (Print); 2312-7589 (Online)

Tikrit Journal of Engineering Sciences

available online at: http://www.tj-es.com

TJES Tikrit Journal of Engineering Sciences

Improving IoT Security using Lightweight Based Deep Learning Protection Model

Mahmood S. Mahmood *[a], *Najla B. Al-Dabagh* [b]

a College of Science, University of Mosul, Mosul, Iraq.

b College of Computer Science and Mathematics, University of Mosul, Mosul, Iraq.

Keywords:

Intrusion Detection System; Internet of Things; MQTT protocol; Deep learning; MQTTset dataset

ARTICLE INFO

Article history:

Received	28 Nov. 2022
Accepted	20 Mar. 2023
Available online	31 Mar. 2023

Citation: Mahmood MS, AL-Dabagh NB. Improving IoT Security Using Lightweight Based Deep Learning Protection Model. *Tikrit Journal of Engineering Sciences* 2023; 30(1): 119-129.
http://doi.org/10.25130/tjes.30.1.12

Corresponding author:

Mahmood S. Mahmood

College of Science, University of Mosul, Mosul, Iraq.

Abstract: The Internet of Things (IoT) has recently become an essential ingredient of human life. The main critical challenges that confront IoT are security and protection. Several methods have been developed to protect the IoT; among these methods is Intrusion Detection System (IDS) Deep Learning-based. On the other hand, these types of IDS have a complex operation that takes a long time when applied on IoT devices and is inconvenient for a massive system that includes many connected devices. Thus, this paper suggested a Lightweight Intrusion Detection System (LIDS) IoT model that depends on deep learning using a Multi-Layer Perceptron (MLP) network. LIDS has the following characteristics lightweight, high accuracy, high speed in detection, and deals with a few features in MQTT protocol. The MQTTset dataset was used in training, validating, and testing the proposed model to investigate the performance of the proposed LIDS. The achieved performance ratios for the proposed LIDS, as measured by accuracy and F1-score. The experiment results showed that for the balanced MQTTset dataset, the number of obtained features was 15 with accuracy (95.06) and F1_score (95.31). Also, for the imbalanced MQTTset, the number of obtained features was 12 with accuracy (96.97) and F1-score (98.24). The obtained results have shown the deep learning efficiency role in improving the accuracy of an intrusion detection model by approximately 3.5% compared to other methods in the literature. In addition, the proposed methods reduced the number of features by around 50% of the total number of features, leading to a LIDS operating in a constrained environment.

تحسين أمان إنترنت الأشياء باستخدام نموذج حماية التعلم العميق الخفيف الوزن

محمود صبحي محمود [1]، نجلاء بديع الدباغ [2]
[a] كلية العلوم / جامعة الموصل /العراق.
[b] كلية علوم الحاسبات والرياضيات / جامعة الموصل / العراق.

الخلاصة

في الوقت الحاضر، أصبح إنترنت الأشياء (IoT) مكونًا أساسيًا في حياة الإنسان. تتمثل التحديات الرئيسية الحاسمة التي تواجه إنترنت الأشياء هي الأمن والحماية. تم تطوير عدة طرق لحماية إنترنت الأشياء؛ من بين هذه الأساليب نظام كشف التسلل (IDS)القائم على التعلم العميق . من ناحية أخرى، فإن هذه الأنواع من IDS لها عملية معقدة تستغرق وقتًا طويلاً عند تطبيقها على أجهزة إنترنت الأشياء وهي غير ملائمة للانظمة الضخمة التي تتضمن العديد من الأجهزة المتصلة. وبالتالي، اقترحت هذه الورقة نموذج نظام كشف التسلل خفيف الوزن (LIDS) لإنترنت الأشياء استنادًا إلى التعلم العميق باستخدام شبكة متعددة الطبقات (MLP) . يتميز LIDS بالخصائص التالية خفيفة الوزن وعالية الدقة وسرعة عالية في الكشف وتتعامل مع بعض الميزات في بروتوكول MQTT . تم استخدام مجموعة بيانات MQTTset في التدريب والتحقق من صحة واختبار النموذج المقترح للتحقق من أداء LIDS المقترح. نسب الأداء المحققة لـ LIDS المقترح، كما تم قياسها بالدقة و F1-score. كانت نتائج التجارب كما يلي: بالنسبة لمجموعة بيانات MQTTset المتوازنة، كان عدد الميزات التي تم الحصول عليها 15 بدقة (95.06) و (95.31). F1-scoreبالنسبة لمجموعة بيانات MQTTset غير المتوازنة ، كان عدد الميزات التي تم الحصول عليها 12 بدقة (96.97) و (98.24) F1-score . أظهرت النتائج التي تم الحصول عليها كفاءة التعلم العميق في تحسين دقة نموذج كشف التسلل بحوالي 3.5٪ مقارنة بالطرق الأخرى في الأدبيات. بالإضافة إلى ذلك، خفضت الأساليب المقترحة عدد الميزات بحوالي 50٪ من إجمالي عدد الميزات، مما أدى إلى تشغيل LIDS في بيئة مقيدة.

الكلمات الدالة: انترنيت الاشياء، نظام كشف التطفل، التعلم العميق، بروتوكول MQTT، مجموعة بيانات MQTTset.

1.INTRODUCTION

The Internet of Things (IoT) consists of many constrained devices, such as mobile phones, smart watches, laptops, tablets, iPads, and other electronic devices and systems connected to the internet. The IoT network transfers the sensors' information and interacts with actuators [1]. Moreover, in 2025, the predicted number of IoT devices will be around 75 billion worldwide [2]. These devices have certain features, such as low computation capacity, and use special lightweight protocols such as Constrained Application Protocol (CoAP) and Message Queuing Telemetry Transport (MQTT) [3,4]. Of course, these features made them smaller, less consuming energy, and more efficient; however, these low settings decreased their security. Also, utilizing the current security mechanisms, such as encryption, authentication, access control, network protection, conventional IDS, and application control, would be time-consuming and inconvenient for a large system with many connected devices. In addition, some portions of the system have vulnerabilities [5]. Furthermore, utilizing a dataset throughout training, validating, and testing a machine learning model is essential. Even though there are very few datasets used in the IoT environment, in particular, Cyber-Security uses available datasets, such as UNSW-NB15 [6], KDDCUP99 [7], or NIMS [8]. However, they are inconvenient to the IoT environment because they partially support dedicated protocols utilized in IoT networks. A way to improve IoT network security is by using a Lightweight Intrusion Detection System (LIDS) [9]. For this reason, attention has been focused on developing a lightweight IDS using a machine learning (Deep Learning) model operating in the IoT environment [10,11]. Building an IDS that operates in an IoT environment has been a common research domain for the past few years due to the significant undesirable effects of attacks on the IoT environment. AP Haripriya, et al [12] proposed a Lightweight Intrusion Detection System based on fuzzy logic that detects malicious activity during IoT device connections called Secure-MQTT. The suggested system uses a fuzzy rule interpolation mechanism and a fuzzy logic-based system for detecting malicious activity on the node. The presented system offers strong protection against DoS attacks for low-configuration devices. The best results in terms of F1-score and FP were 90.9 and 2, respectively. Fenanir, S., et al [13] proposed a lightweight intrusion detection model. Both the machine learning approaches of feature selection and classification have been used, represented by Support Vector Machines (SVM), Decision Trees (DT), and k-Nearest Neighbor (KNN). The datasets NSL-KDD, KDD99, and UNSW-NB15 were used in experiments to train and evaluate the proposed models. The outcomes demonstrated that KNN and DT outperformed the other algorithms; however, KNN requires more time to categorize data than the DT algorithm. Alsamiri, J., et al [14] presented the literature by evaluating seven machine learning algorithms that effectively and quickly detect IoT attacks. First, the Bot-IoT dataset was used to evaluate the detection algorithms. They extracted 84 flow-based features from the raw traffic records of the dataset using CICFlowMeter [15]. Then, the Random Forest Regressor algorithm determines the features' importance. The achieved performance ratios of the utilized algorithms, as measured by F-measure, were as follows: Random Forest was 97; QDA was 86; Naive Bayes was 77; ID3 was 97; MLP was 83; AdaBoost was 97, and K Nearest Neighbors was 99. Vaccari et al. [16] provided a new brand MQTT dataset, a mix of legitimate and malicious traffic targeting the MQTT broker in the IoT network. The new dataset is called MQTTset, and it is freely accessible. Raw PCAP files were used to store the extracted features from traffic based on the packet level. The authors used machine learning techniques to validate this dataset, including gradient boosting, multi-layer perceptrons, decision trees, neural networks, naïve Bayes, and random forests. The results have shown that all algorithms achieved accuracy levels above 98%, and the F1 score was higher than 97% with an unbalanced dataset. After balancing, their accuracy and F1 score results differed, except that the naïve Bayes algorithm produced low performance of 64 % and 68 % for accuracy and F1_score, respectively. However, the remaining algorithms had accuracy and F1 scores between 87% and 91%, respectively. Susilo, B., et al [17] examined various machine-learning and deep-learning methods on the Bot-IoT dataset to improve IoT security. They proposed using RF, CNN, and MLP algorithms to develop a denial-of-service (DoS) attack detection system. Random forests and CNN achieved the best outcome regarding multi-class classification accuracy and AUC. However, the accuracy somewhat decreased with adding epochs in experiments with 32 batches (CNN = 88.30, MLP = 62) and 64 batches (CNN = 90.64, MLP = 53.89). However, the accuracy slightly increased in experiments with 128 batches (CNN = 91.27, MLP = 79.01). Dissanayake, M. B. et al [18] presented a double-sided analysis. As a first step, some of the conventional ML models (AdaBoost Classifier, MLP Classifier, light GBM Classifier, and Decision Tree Classifier) were all trained to identify the type of cyber-attack from incoming MQTT traffic data. The results demonstrated that these models could accurately detect cyber-attacks and that the MLP classifier offered the fastest average processing speed and 90% average accuracy. In the second step, it was concluded that focusing on ten features in scenarios with limited resources was adequate. Despite having a stunning 99290 traffic entries in the dataset, there was an obvious class imbalance. The average accuracy, F1 score, recall, and precision were the performance evaluation metrics. All models can, on average, be classified with 90% accuracy. Azizan, AH. et al. [19] proposed a Machain Learning -based Network IDS (ML-based NIDS) model that utilized three machine learning algorithms, which were decision jungle (DJ), random forest (RF), and support vector machine (SVM). KDD dataset and CIC-IDS2017 dataset were used to test the proposed IDS. The obtained average accuracy results were (DJ = 96.50%, RF = 96.76%, and SVM = 98.18%). Thus, The SVM method was the most effective approach for detecting system intrusions. Alfoudi, AS. et.al.[20] proposed a hyper clustering model for dynamic IDS by enhancing the cosine similarity and Density-Based Spatial-Clustering of Applications with Noise (DBSCAN) to resolve the dataset imbalance. In addition, a new classifier was proposed depending on the cosine similarity to detect the labeling of the abnormal behavior. According to the experiment results, the suggested model performed better than the original DBCAN and the related works.The proposed DBSCAN reduced the Mean Square Error (MSE) from (0.66) to (0.13) and reached 79.10%, 90.03%, and 86.82% in general accuracy on KDDTest-21 NSL-KDD, UNSW-NB15, and KDDTest+, respectively. The main weakness of the above works is the lack of using a specific IoT dataset (i.e., classical datasets like NSL-KDD, KDD Cup 99, IoT-23, and BoT-IoT). In addition, they used many communication protocols that work in different layers of the OSI suite and consume the high time for training and testing stages. So, this study's primary objective is to create a LIDS which can operate in a constrained environment. This paper suggests a new LIDS model that depends on DNN, which can classify the MQTT messages into either normal or attack behaviors. This study used the MQTTset dataset, which consists of many packets based on the MQTT protocol. The proposed LIDS differs from the abovementioned approaches by working on lightweight MQTT protocol, training, testing speed, and using a current dataset (MQTTset) in the training and testing stages. This dataset included thirty-four features extracted from the OSI model's fourth layer (transport layer) based on packet level. As a result, the following constitute this paper's contribution:

1- Introduce the MQTT protocol with the MQTTset dataset.
2- Propose a lightweight Intrusion Detection System Model that can classify the traffic in MQTTset into legitimate and attack traffic. The new model has the following characteristics: it is Lightweight and small enough to be implemented in edge devices

like the Raspberry Pi; it requires less power to process because it uses the lightweight MQTT protocol; and it has a small number of features to be tested, resulting in low traffic and high detection accuracy.

3- The proposed LIDS is evaluated in terms of detection accuracy, F1-score, and the amount of traffic. The dataset has been divided based on cross-validation for the training and validation phases.

The rest of the paper comprises the following: Section 2 presents the MQTT protocol and MQTTset Dataset. Section 3 demonstrates the proposed LIDS model proposed, which uses deep learning. Then, the implementation and the obtained results are in Section 4. Finally, the paper's conclusions are listed in Section 5, which also proposes recommendations for future works.

2.MQTT PROTOCOL AND MQTTSET DATASET

MQTT protocol is a client-server, lightweight publish/subscribe messaging transport protocol standardized by OASIS and ISO (ISO/IEC PRF 20922) submitted in 1999 [21-23]. This protocol's design minimizes device resource requirements and network bandwidth while ensuring reliable delivery [24]. The MQTT protocol is ideal for communication in constrained environments because of its many characteristics, including its openness, simplicity, lightweight, and ease of deployment. The protocol works with TCP/IP or other network protocols that offer lossless, ordered, and bidirectional connections [25]. Facebook Messenger is one of the most well-known MQTT applications. The MQTT protocol could solve the developers' most prevalent issues, such as bandwidth and battery life [26]. Clients, Servers (Brokers), Sessions, Subscriptions, and Topics comprise the MQTT model's main components [25,27], as shown in Fig. 1.

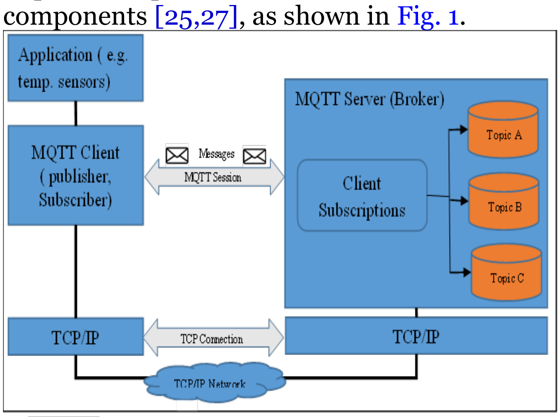

Fig. 1 The Main Components of the MQTT Model [27]

Meanwhile, numerous datasets are freely available to build an IDS for IoT; however, each type has a weak point, as summarized in Table 1.

Table 1 Available IoT Network Datasets [16]

Dataset	Lacks
KDDCUP99 [28,29]	don't focus on the IoT topic
UNSW-NB15 [30]	don't focus on the IoT topic
NIMS [30]	don't focus on the IoT topic
NLS-KDD [31]	don't focus on the IoT topic
N-BaIoT [32]	Focus well on Wi-Fi traffic
IoT-23 [33]	Focus well on DNS traffic for IoT topic
MedBIoT [34]	not found authentication phase and MQTT attacks
TON_IoT [35]	not found disconnection and authentication phase
BoT-IoT [36]	fresh PCAP traffic data about MQTT wasn't freely
Custom datasets	PCAP or fresh traffic missing file

On the other hand, The MQTTset dataset was constructed by utilizing IoT-FlocK [37], a tool for generating network traffic capable of emulating IoT environments based on CoAP and MQTT protocols. The traffic of the MQTT generated is then stored as a packet capture file (PCAP). The dataset files are available at "https://www.kaggle.com/cnrieiit/mqttset." It contains about 12000000 network packets; its total size is nearly 1,093,676,216 bytes. Each packet has 34 features which are explained in Table 2. Also, the MQTTset dataset is provided in CSV file format; however, KDD CUP 99 is provided in a CSV file format only. So, the MQTTset dataset is more flexible than KDD CUP 99 as it allows manually manipulating the new data and yielding different CSV files as required. In addition, there are many types of attacks in the MQTTset dataset, which are usually targeting the MQTT broker, as stated below [38-44]:

a- MQTT Publish-Flood attack.
b- Flooding DoS attack.
c- Slowite DoS attack.
d- Malformed Data attack.
e- Brute Force Authentication attack.

3.THE PROPOSED MODEL

This section discusses the data pre-processing and how the Deep Learning Algorithm was utilized to build a LIDS operating in an IoT environment.

3.1. Pre-Processing of Data

Before training and testing any LIDS model, the model's input data should be passed through the following pre-processing stages.

a. Feature Set

The proposed LIDS utilized a list of features, using them in both the training and testing stages, derived from the MQTTset dataset. The MQTTset dataset had 34 columns (33 features and 1 target), and after manipulation, using panda's library, found and removed 14 features with zero value and long message data. The result was a new dataset with 20 columns (19 features and 1 target).

b. Normalization

Some of the MQTTset dataset features had a different range of values. To normalize these values and make the values in all features have the same range (0,1), The three most commonly used pre-processing transformations, Min_Max or Unit Scaling, Standard Scaling, and Quantile Scaling, were chosen as the best pre-processing transformations to scale the value of a feature in the training and testing datasets. The Min_Max method was used in the proposed model, as defined in Eq. (1).

$$NewData = \left(\frac{OldData - \min}{max - \min}\right) \quad \textbf{(1)}$$

where NewData = the new feature's value after normalization; min= the minimum value in the feature; max= maximum value in the feature; and oldData = the feature's value before normalization. Also, the Quantile Scaling method was used in the proposed model.

3.2. Using Deep Learning for Anomaly-Based IDS

a- Training the Deep Learning Network

This subsection manifests the training stage of the suggested DNN model, as shown in Fig.2.

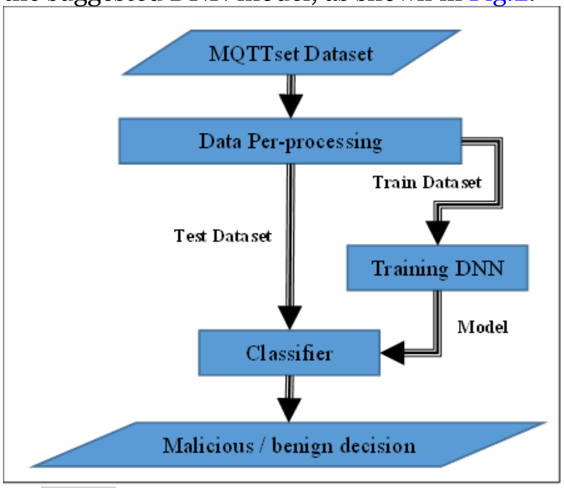

Fig. 2 The Suggested DNN Training and Testing Phases [45]

The features were entered into the DNN as a vector at the first level. Then, that went through every DNN layer, and each DNN layer's neural nodes used an activation function to compute the output, which results in a filtered output. The Rectified Linear Unit (ReLU), an activation function, was used to develop the proposed model. The ReLU function is demonstrated in Eq. (2) as follows:

$$F(x) = \max(0\,;\,x) \quad \textbf{(2)}$$

Each hidden_layer was connected to the following hidden_layer using the linear combinations of outcomes computed by ReLU and forwarding the output to the following layer.

b- Validation of Deep Learning Network

Developing a DNN model usually includes tuning its configuration. This stage involves, for instance, selecting the number of layers and the layer size (model hyperparameters) to discriminate hyperparameters from the parameters, which are regarded as network weights. This tuning is performed by utilizing, as a feedback signal, the model's performance on the validation data. This tuning is a learning form: searching for the best configuration in a certain parameter space [46]. There are three traditional evaluation techniques: K-fold validation, simple hold-out validation, and iterated K-fold validation with shuffling [46]. In this paper, iterated K-fold validation with shuffling was used.

c- Testing the Deep Learning Network

The MQTTset testing dataset was categorized as malicious or benign traffic in the test stage. DNN predicted and detected benign and malicious actions through the binary classifier. The proposed LIDS model computed the meta-feature vector and the minimum cost parameter for the packet of testing the dataset and contrasted them with the DNN classifier achieved through the training stage. From this operation, DNN produced a binary classification (benign or malicious) to the packet of the testing dataset, which was observed.

4.IMPLEMENTATION OF THE PROPOSED LIDS AND THE RESULTS OF THE EXPERIMENT

In this section, implementing the LIDS model proposed for the IoT environment was presented. The LIDS model depends on DNN, implemented through the Keras library, an open-source Python library. Implementing the model involved three stages, which were:

1- Input data and pre-processing.
2- Building and training the DNN classifier.
3- Testing the DNN classifier.

During input data and pre-processing, the MQTTset dataset was used as input to the LIDS model (Anomaly-Based detection). Moreover, every classifier node received weights during training to distinguish between different input data classes and create a binary classifier. Therefore, building and training the classifier of the DNN were essential to LIDS development. Moreover, the current subsection presents the results obtained after executing (testing) the proposed LIDS for the IoT environment.

4.1. Packages and Environment

The python language was used to implement the DNN (train and test stages) for the LIDS model proposed. Compared to other languages like C++ and Java, it was rather simple to use and understand. Furthermore, the NumPy library involved in Python was used for matrix manipulation (matrix and arithmetic manipulations). Using such a library simplified implementing an anomaly-based IDS model because the data in the anomaly-based IDS was typically represented as a matrix form. The Keras library was used in building a Sequential DNN model due to its lightweight, easy extensibility, and modularity, and it is created as a stack of DNN layers. Moreover, it is preferred to implement a sequential DNN model because it is easy and flexible for a resource-constrained IoT network. Furthermore, the Keras library can process enormous amounts of data without difficulty.

4.2. MQTTset Pre-Processing

The preceding section's MQTTset dataset comprised 12 million packets with 34 columns (33 input features and 1 target). After being downloaded, the MQTTset dataset could be in the format of a Comma Separated Values (CSV) file and can be edited by the Microsoft Excel application. However, panda's library was used for reasonable reasons as it is efficient, reliable, and fast in handling huge datasets with a few GBs in size. Columns 2 and 3 of Table 2 show the values of two selected messages (i.e., legitimate and DoS attack) in the MQTTset containing 34 features after eliminating the undesired columns (features with zero or long message value denoted by X) from the MQTTset dataset. The fresh dataset is shown in the last two columns in Table 2, which contains 20 features only instead of 34 features (19 features and 1 target). The 12 million packets were pruned out using the data pre-processing (Up-sampling and Down-Sampling) to create a fresh dataset that involves 565149 packets. The MQTTset dataset shows that the proposed LIDS model can classify testing samples as either malicious or benign activity.

Table 2 Raw Mqttset Dataset Before and After Pre-Processed [16,45].

Feature	Values_1	Values_2	Pre-processed Value_1	Pre-processed Value_2
Tcp-Flags	0x00000018	0x00000010	0.652174	0.956522
Tcp-time_delta	0.998867	6.70E-05	3.55E-06	3.44E-05
Tcp-len	10	1460	0	0.000122
Mqtt-conack.flags	0	0	X	X
Mqtt-conack.flags.reserved	0	0	X	X
Mqtt-conack.flags.sp	0	0	X	X
Mqtt-conack.val	0	0	0	1
Mqtt-conflag.cleansess	0	0	0	0
Mqtt-conflag.passwd	0	0	0	0
Mqtt-conflag.qos	0	0	X	X
Mqtt-conflag.reserved	0	0	X	X
Mqtt-conflag.retain	0	0	X	X
Mqtt-conflag.uname	0	0	0	0
Mqtt-conflag.willflag	0	0	X	X
Mqtt-conflags	0	0	0	0
Mqtt-dupflag	0	0	0	0
Mqtt-hdrflags	0x00000030	0x00000032	0	0.142857
Mqtt-kalive	0	0	0	0
Mqtt-len	8	169	0	0.00289
Mqtt-msg	32	6.4E+199	X	X
Mqtt-msgid	0	2714	0	0
Mqtt-msgtype	3	3	0	0.142857
Mqtt-proto_len	0	0	0	0
Mqtt-protoname	0	0	0	0
Mqtt-qos	0	1	0	0
Mqtt-retain	0	0	0	0
Mqtt-sub.qos	0	0	X	X
Mqtt-suback.qos	0	0	X	X
Mqtt-ver	0	0	0	0
Mqtt-willmsg	0	0	X	X
Mqtt-willmsg_len	0	0	X	X
Mqtt-willtopic	0	0	X	X
Mqtt-willtopic_len	0	0	X	X
Target	Legitimate	Dos Attack	0	1

4.3.The Implementation of Deep Learning Algorithm

In the proposed deep learning model, every hidden layer played the role of an encoding filter, which performed input filtering and provided the filtrate to the subsequent layer. Each layer must use an activation function to activate its neural nodes when receiving input. This model was created using a sequential deep learning architecture, each layer set up to be activated using the ReLU activation function. Using the Keras package to implement the sequential machine-learning model for the classification task, The Pandas library was used by the proposed LIDS to read the training dataset that was saved in a CSV format and saved it in a data frame as shown in the following pseudocode:

```
Dataset_Train=pd.read_csv("train_70_aug.csv")
```

The training dataset (balanced dataset) comprised 115824 benign network packets and 115821 malicious network packets, whereas the dataset for testing comprised 500000 benign network packets and 65149 malicious network packets. Then, three hidden layers were used to form a sequential DNN, and each layer's processing units were given a ReLU activation function, as shown by the pseudocode below:

```
Model1=sequential ()
Model1.add (dense (45, input_dim=x.shape[1]))
Model1.add (activation ("Relu"))
Model1.add (dropout (0.5))
Model1.add (dense (30))
Model1.add (activation ("Relu"))
Model1.add (dropout (0.5))
Model1.add (dense (15))
Model1.add(activation("Relu"))
Model1.add (dropout (0.5))
Model1.add (dcense(1, activation="sigmoid")
```

A straightforward solution to the overfitting problem in feed-forward neural networks was provided by the original dropout method introduced in 2012 [43]. During each training iteration, a neuron with the probability (p) was absent from the network. Once trained, the full network was utilized in the testing stage, multiplying the neuron outputs by the probability (p) representing the removed neurons. The probability can differ for each layer. Normally, it is recommended that p = 0:2 for the input layer and p = 0:5 for the hidden layers, and there are no dropped-out neurons in the output layer. Finally, in the output layer of the suggested model, the sigmoid function was utilized as an activation function. For the machine learning model, k-fold cross-validation was the golden standard. It provided a strong performance to the model concerning the new data. The training dataset was divided into k subsets via the k-fold cross-validation method. It took a (k-1) subset to train the models, and the remaining subset (one subset, the validating dataset) was held out to evaluate the model performance. This process was repeated until all subsets could be used as a validation dataset. The performance measurement was then averaged for all created models, as shown in the pseudocode below:

```
K_flod=Kflod(n_splits=4, shuffle=True, random_state=2)
```

After that, with 128 epochs, the suggested model was compiled and fitted. Finally, In the variable predictions, the classifiers were assigned and saved. Therefore, in each one of the tests, the classifier's output was normalized to a binary value. Once the predictions were generated, predictions were translated into concrete outcomes that determined the accuracy and F1 score metrics.

4.4. Experiments Results

The experimental results of the proposed LIDS are presented in this section, as the model was trained and tested using the MQTTset dataset. The following equations represent the main metrics that were used for the performance evaluation of the proposed LIDS:

$$\text{Accuracy} = (TP + TN)/(P + N) \quad \textbf{(3)}$$

$$\text{F1 score} = (2 * TP) / (2 * TP + FP + FN) \quad \textbf{(4)}$$

where TP (True Positive) reflects the attack instances which were classified correctly, TN (True Negative) reflects the benign instances which were classified correctly, P (Positive) reflects the total number of attack instances, N (Negative) reflects the total number of benign instances, FP (False Positive) stands for benign instances that are classified falsely as an attack, and FN (False Negative) stands for the attack instances falsely classified as benign. Table 3 shows the performance of the proposed model in terms of accuracy and F1-score based on Eqs. (3, 4), respectively. Seven experiments were applied based on two dataset categories (balanced and unbalanced). Each experiment depended on four training parameters: number of features, optimizer, number of epochs, and batch size. The first four experiments were implemented on the balanced dataset, and the rest were implemented on the unbalanced dataset. Regarding experiments (1 and 2), Adam was used as an optimizer and a min-max method for scaling. When removing the features containing the value 0 and the feature with long message data, the accuracy was almost equal. Half reduced the training time in experiment no. 2. The main reasons for this phenomenon are that the batch size in experiment no. 2 was more significant than in experiment no. 1. The number of features in experiment no. 2 was less than in experiment no. 1. In experiments (3 and 4), the accuracy was 95.06 and 84.81, respectively, because the Sgd was used as an optimizer instead of Adam in experiments (1 and 2). Furthermore, the quantile scaling method was used instead of the experiments' min-max (1 and 2). In addition, the accuracy of experiment no. 3 was more significant than the accuracy of experiment no. 4. Although the number of used features in experiment no. 3 was larger than the number of used features in experiment no. 4, indicating the importance of the selected features in experiment no. 3. On the other hand, when using an unbalanced dataset, where there were much more entries of the normal class than there were of the attack class, and in experiments, it gave high results in terms of accuracy and F1-score due to the drift of the training process to the class that had more records than others. However, as shown in experiments (5, 6, and 7), the results often are neither accurate nor convincing. Note: all the experiments used three layers (45,30,15). For the input and hidden layers, the ReLU function was utilized as an activation function, and the sigmoid function was employed as an activation function for the output layer. Also, the experiments used binary_crossentropy as a loss function. For more details, Figs. 3-8 show the relationships of the accuracy for both training and validation in the experiments (2-7), respectively. For more investigation into the proposed model's performance, comparisons with the performance of the other methods from the literature were made. Table 4 shows that the proposed system outperformed others compared to other methods in terms of no. of features, accuracy, and F1-score, which reached 12, 96.97, and 98,29, respectively, for unbalanced datasets and 15, 95.06, and 95,31, respectively for the balanced dataset.

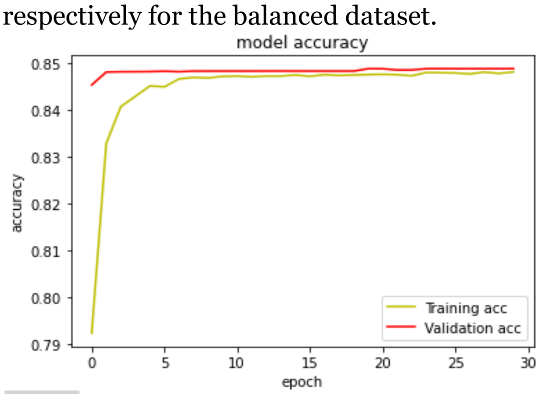

Fig. 3 Accuracy Diagram for Experiment no.2.

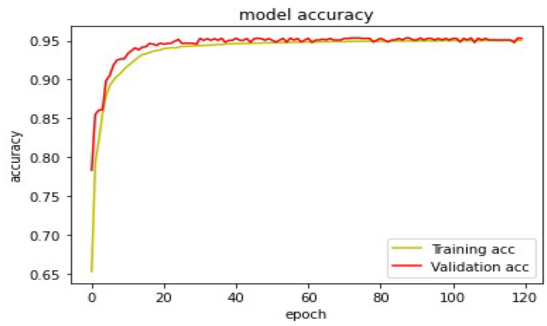

Fig. 4 Accuracy Diagram for Experiment No.3.

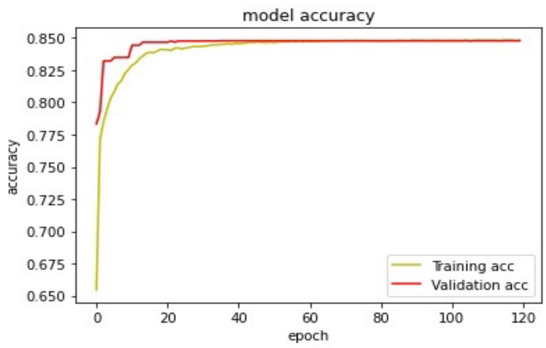

Fig. 5 Accuracy Diagram for Experiment No.4.

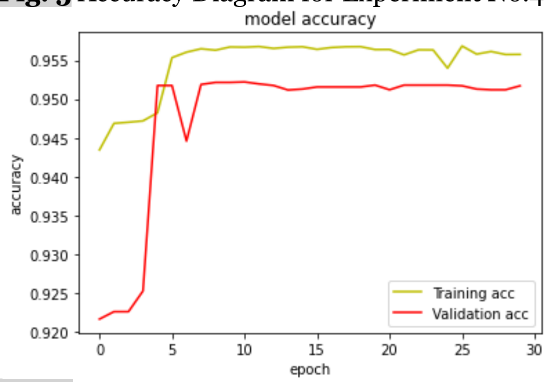

Fig. 6 Accuracy Diagram for Experiment No.5

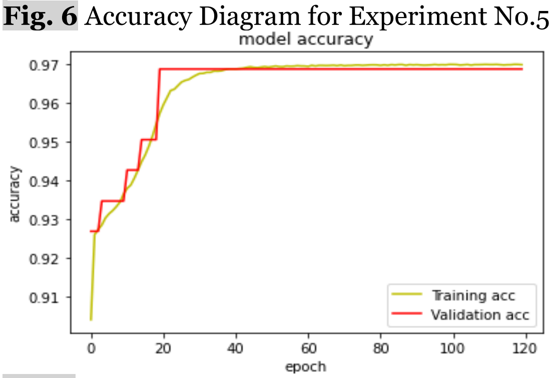

Fig. 7 Accuracy Diagram for Experiment No.6.

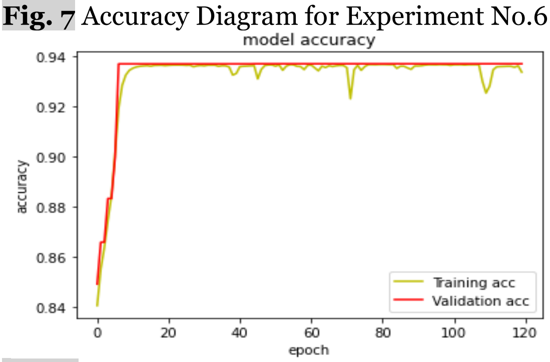

Fig. 8 Accuracy Diagram for Experiment No.7.

Table 3 Overall Accuracy and F1 Score Metrics for Different Experiments.

No.	No. of features	Optimizer	Epoch / batch size	No. of record in train / test	Accuracy training/ testing	F1-Score	Time(s) for train and test	Figure
1	32	Adam	20 / 32	231645 balanced/ 99289 with min-max scaling	83.70 / 84.01	-	68.0513	-
2	19	Adam	30 /128	231645 balanced / 565149 with min-max scaling	84.81 / 95.87	-	28.1358	Fig. 4
3	15	Sgd	120 / 256	231645 balanced / 99290 with quantile scaling	95.06 / 95.31	94.40	307.6173	Fig. 5
4	12	Sgd	120 / 256	231645 balanced / 99290 with quantile scaling	84.81 / 84.80	84.54	300.280	Fig. 6
5	19	Adam	30 / 128	565149 unbalanced /565149	95.58 / 95.17	-	75.7334	Fig. 7
6	12	Sgd	120 / 256	125954 unbalanced / 53990	96.97 / 96.80	98.29	51.8668	Fig. 8
7	12	Sgd	120 / 256	1048575 unbalanced / 1048575 with min-max scaling	93.36 / 93.76	93.48	412.9251	Fig. 9

Table 4 Comparison Between Proposed LIDS and Other Methods from the Literature.

Reference	Algorithm	Dataset	No. of feature	Accuracy	F1-score
[12]	Fuzzy Rule	Simulated dataset	*	*	90.9
[16]	MLP	MQTTset-Unbalance	34	97	98
	MLP	MQTTset-balance	34	90.38	90.18
	Neural Network	MQTTset-balance	34	90.44	90.23
	RF	MQTTset-balance	34	91.59	91.40
	NB	MQTTset-balance	34	64.38	68.72
	DT	MQTTset-balance	34	91.59	91.40
	Gradient Boost	MQTTset-balance	34	87.95	87.27
[18]	MLP	MQTTset-Unbalance	10	90	79
proposed model	MLP	MQTTset-Unbalance	12	96.97	98.29
		MQTTset-balance	15	95.06	95.31

5.CONCLUSIONS

This paper examined a deep learning algorithm to build a lightweight intrusion detection system (LIDS) that works in an IoT environment. MLP algorithm was used to construct a proposed model, and the MQTTset dataset was used to train, validate, and test the model. Many experiments were conducted until getting better results by changing the hyper-model parameters and the number of features input to the model. The experiment results have shown good performance for the proposed (LIDS) system in terms of accuracy and the F1 score when using an unbalanced dataset and accepted results for the balanced dataset. The proposed (LIDS) system also decreased the number of used features by around (50%) of the total features and increased the patch size leads to sped up the model's training. The proposed model has the following properties (Lightweight, high speed to detect attacks, deals with few features that exist in the MQTT message). The limitations of this study are represented by: the proposed LIDS depends on the MQTT protocol only and can detect the MQTT broker attacks only. In the future, it is aimed to develop the LIDS model that works with MQTT and other protocols' attacks. In addition, propose a LIDS which can classify the attacks after detecting them.

ACKNOWLEDGMENT

The authors would like to thank the University of Mosul / College of Computer Science and Mathematics for their facilities, which have helped to enhance the quality of this work.

REFERENCES

[1] Green J. (2014, March). The internet of things reference model. In Internet of Things World Forum (pp. 1-12). San Jose, CA, USA: CISCO.

[2] Ghannadrad A. (2021). **Machine Learning-Based DoS Attacks Detection for MQTT Sensor Networks**.

[3] Razzaq MA, Gill SH, Qureshi MA, Ullah S. **Security Issues in the Internet of Things (IoT): A Comprehensive Study**. *International Journal of Advanced Computer Science and Applications* 2017; **8**(6): 383-388.

[4] Galán CO, Lasheras FS, de Cos Juez, FJ, Sánchez AB. **Missing Data Imputation of Questionnaires by Means of Genetic Algorithms with Different Fitness Functions**. *Journal of Computational and Applied Mathematics* 2017; **311**: 704-717.

[5] Kolias C, Kambourakis G, Stavrou A, Voas J. **DDoS in the IoT: Mirai and other botnets**. *Computer* 2017; **50**(7): 80-84.

[6] Moustafa N, Slay J. (2015, November). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In 2015 military communications and information systems conference (MilCIS) (pp. 1-6). IEEE.

[7] Tavallaee M, Bagheri E, Lu W, Ghorbani AA. (2009, July). A detailed analysis of the KDD CUP 99 data set. In 2009 IEEE symposium on computational intelligence for security and defense applications (pp. 1-6). IEEE.

[8] Oluranti J, Omoregbe N, Misra S. Effect of feature selection on performance of internet traffic classification on NIMS multi-class dataset. *In Journal of Physics:*

Conference Series IOP Publishing 2019; **1299**(1): 012035.

[9] Ben-Asher N, Gonzalez C. **Effects of Cyber Security Knowledge on Attack Detection**. *Computers in Human Behavior* 2015; **48**: 51- 61.

[10] Prabha K, Sree SS. **A Survey on IPS Methods and Techniques**. *International Journal of Computer Science Issues (IJCSI)* 2016; **13**(2): 38.

[11] Hamed T, Ernst JB, Kremer SC. **A Survey and Taxonomy of Classifiers of Intrusion Detection Systems**. *Computer and Network Security Essentials* 2018; 21-39.

[12] AP H. **Secure-MQTT: An Efficient Fuzzy Logic-Based Approach to Detect Dos Attack in MQTT Protocol for Internet of Things**. EURASIP *Journal on Wireless Communications and Networking* 2019; (1): 1-15.

[13] Fenanir S, Semchedine F, Baadache A. **A Machine Learning-Based Lightweight Intrusion Detection System for the Internet of Things**. *Revue d'Intelligence Artificielle* 2019; **33**(3): 203-211.

[14] Alsamiri J, Alsubhi K. **Internet of Things Cyber Attacks Detection Using Machine Learning**. *International Journal of Advanced Computer Science and Applications* 2019; 10(12).

[15] Lashkari, A. H., Draper-Gil, G., Mamun, M. S. I., & Ghorbani, A. A. (2017, February). Characterization of tor traffic using time based features. In ICISSp (pp. 253-262).

[16] Vaccari I, Chiola G, Aiello M, Mongelli M, Cambiaso E. **MQTTset, A New Dataset for Machine Learning Techniques on MQTT**. *Sensors* 2020; **20**(22): 6578(1-17).

[17] Susilo B, Sari RF. **Intrusion Detection in IoT Networks using Deep Learning Algorithm**. *Information* 2020; **11**(5): 279(1-11).

[18] Dissanayake MB. Feature Engineering for Cyber-attack detection in Internet of Things. (2021).

[19] Azizan AH, Mostafa SA, Mustapha A, Foozy CFM, Wahab MHA, Mohammed M A, Khalaf BA. **A Machine Learning Approach for Improving the Performance of Network Intrusion Detection Systems**. *Annals of Emerging Technologies in Computing (AETiC)* 2021; **5**(5): 201-208.

[20] Alfoudi AS, Aziz MR, Alyasseri ZAA, Alsaeedi AH, Nuiaa RR, Mohammed MA, ... & Jaber MM. (2022). Hyper clustering model for dynamic network intrusion detection. *IET Communications*.

[21] Soni D, Makwana A. **A Survey on MQTT: A Protocol of Internet of Things (IoT)**. *In International conference on telecommunication, power analysis and computing techniques (ICTPACT-2017)* 2017; **20**: 173-177.

[22] R. K. Kodali and S. Soratkal, "MQTT Based Home Automation System using ESP8266," *2016 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*, Agra, India, 2016, pp. 1-5.

[23] Banks A. (2014). MQTT Version 3.1. 1. Edited by Andrew Banks and Rahul Gupta. *OASIS Standard*.

[24] Bandyopadhyay S, Bhattacharyya A. (2013, January). Lightweight Internet protocols for web enablement of sensors using constrained gateway devices. *In 2013 International Conference on Computing, Networking and Communications (ICNC)* (pp. 334-340). IEEE.

[25] Mishra B, Kertesz A. "The Use of MQTT in M2M and IoT Systems: A Survey," in IEEE Access, vol. 8, pp. 201071-201086, 2020.

[26] Fehrenbach, P. (2018). Messaging Queues in the IoT under pressure. *Computational Science and Its Applications, ICCSA*, 1-9.

[27] Egli, P. R. (2015). An introduction to MQTT, a protocol for M2M and IoT applications. Indigoo. com.

[28] Elkhadir, Z., Chougdali, K., & Benattou, M. (2017, November). An effective cyber attack detection system based on an improved OMPCA. In 2017 International Conference on Wireless Networks and Mobile Communications (WINCOM) (pp. 1-6). IEEE.

[29] Mohammadi S, Mirvaziri H, Ghazizadeh-Ahsaee M, Karimipour H. **Cyber Intrusion Detection by Combined Feature Selection Algorithm**. *Journal of Information Security and Applications* 2019; **44**: 80-88.

[30] Moustafa N, Turnbull B, Choo KKR. **An Ensemble Intrusion Detection Technique Based on Proposed Statistical Flow Features for Protecting Network Traffic of Internet of Things**. *IEEE Internet of Things Journal* 2018; **6**(3): 4815-4830.

[31] Shalaginov, A., Semeniuta, O., & Alazab, M. (2019, December). MEML: Resource-aware MQTT-based machine learning for network attacks detection on IoT edge devices. *In Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing Companion* (pp. 123-128).

[32] Meidan Y, Bohadana M, Mathov Y, Mirsky Y, Shabtai A, Breitenbacher D, Elovici Y. **N-Baiot—Network-Based Detection of Iot Botnet Attacks using Deep**

Autoencoders. *IEEE Pervasive Computing* 2018; **17**(3): 12-22.

[33] Li J, Zhao Z, Li R, Zhang H. **Ai-Based Two-Stage Intrusion Detection for Software Defined IoT Networks**. *IEEE internet of Things Journal* 2018; **6**(2): 2093-2102.

[34] Guerra-Manzanares, A., Medina-Galindo, J., Bahsi, H., & Nõmm, S. (2020, February). MedBIoT: Generation of an IoT Botnet Dataset in a Medium-sized IoT Network. In ICISSP (pp. 207-218).

[35] Moustafa, N. (2019, October). New generations of internet of things datasets for cybersecurity applications based machine learning: TON_IoT datasets. In Proceedings of the eResearch Australasia Conference, Brisbane, Australia (pp. 21-25).

[36] Ciklabakkal, E., Donmez, A., Erdemir, M., Suren, E., Yilmaz, M. K., & Angin, P. (2019, October). ARTEMIS: An intrusion detection system for MQTT attacks in Internet of Things. *In 2019 38th Symposium on Reliable Distributed Systems (SRDS)* (pp. 369-3692). IEEE.

[37] Ghazanfar, S., Hussain, F., Rehman, A. U., Fayyaz, U. U., Shahzad, F., & Shah, G. A. (2020, March). Iot-flock: An open-source framework for iot traffic generation. In 2020 International Conference on Emerging Trends in Smart Technologies (ICETST) (pp. 1-6). IEEE.

[38] Wood AD, Stankovic JA. **Denial of Service in Sensor Networks**. *Computer* 2002; **35**(10): 54-62.

[39] Vaccari I, Aiello M, Cambiaso E. **SlowITe, a Novel Denial of Service Attack Affecting MQTT**. *Sensors* 2020; **20**(10):2932.

[40] Cambiaso E, Papaleo G, Chiola G, Aiello M. **Slow DoS Attacks: Definition and Categorisation**. *International Journal of Trust Management in Computing and Communications* 2013; **1**(3-4): 300-319.

[41] Cambiaso, E., Papaleo, G., Chiola, G., & Aiello, M. (2015, June). Designing and modeling the slow next DoS attack. In Computational intelligence in security for information systems conference (pp. 249-259). Springer, Cham.

[42] Cambiaso, E., Papaleo, G., & Aiello, M. (2012, October). Taxonomy of slow DoS attacks to web applications. In International Conference on Security in Computer Networks and Distributed Systems (pp. 195-204). Springer, Berlin, Heidelberg.

[43] Bhagat Patil, A. R., & Thakur, N. V. (2019). Mitigation Against Denial-of-Service Flooding and Malformed Packet Attacks. In Third International Congress on Information and Communication Technology (pp. 335-342). Springer, Singapore.

[44] Stiawan, D., Idris, M., Malik, R. F., Nurmaini, S., Alsharif, N., & Budiarto, R. (2019). Investigating brute force attack patterns in IoT network. Journal of Electrical and Computer Engineering, 2019.

[45] Chawla, S. (2017). Deep learning based intrusion detection system for Internet of Things.thesis, University of Washington.

[46] Ketkar, N., & Santana, E. (2017). Deep learning with Python (Vol. 1). Berkeley: Apress.