electronics

MDPI

Article

A Robust Framework for MADS Based on DL Techniques on the IoT

Hussah Talal [1,*] and Rachid Zagrouba [2]

[1] Department of Computer Science, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 7059, Dammam 32252, Saudi Arabia
[2] Department of Computer Information Systems, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam 31441, Saudi Arabia; rmzagrouba@iau.edu.sa
* Correspondence: Hussah_Talal@outlook.sa

Abstract: Day after day, new types of malware are appearing, renewing, and continuously developing, which makes it difficult to identify and stop them. Some attackers exploit artificial intelligence (AI) to create renewable malware with different signatures that are difficult to detect. Therefore, the performance of the traditional malware detection systems (MDS) and protection mechanisms were weakened so the malware can easily penetrate them. This poses a great risk to security in the internet of things (IoT) environment, which is interconnected and has big and continuous data. Penetrating any of the things in the IoT environment leads to a penetration of the entire IoT network and control different devices on it. Also, the penetration of the IoT environment leads to a violation of users' privacy, and this may result in many risks, such as obtaining and stealing the user's credit card information or theft of identity. Therefore, it is necessary to propose a robust framework for a MDS based on DL that has a high ability to detect renewable malware and propose malware Anomaly detection systems (MADS) work as a human mind to solve the problem of security in IoT environments. RoMADS model achieves high results: 99.038% for *Accuracy*, 99.997% for Detection rate. The experiment results overcome eighteen models of the previous research works related to this field, which proved the effectiveness of RoMADS framework for detecting malware in IoT.

Keywords: anomaly detection system; deep learning techniques; IoT; malware detection; LSTM autoencoder

Citation: Talal, H.; Zagrouba, R. A Robust Framework for MADS Based on DL Techniques on the IoT. *Electronics* **2021**, *10*, 2723. https://doi.org/10.3390/electronics10212723

Academic Editors: Amir Mosavi and Hamed Taherdoost

Received: 16 September 2021
Accepted: 1 November 2021
Published: 8 November 2021

1. Introduction

Many countries and governments around the world are currently seeking to switch from the traditional environments to IoT environments in various fields, whether in industry, healthcare, oil and gas, or smart cities, in order to take advantage of the services and facilities provided by the IoT. But until now there are still concerns about the security and privacy [1–3]. The essential components of internet of things (IoT) systems are IoT network and IoT devices. If one of them is compromised, this may lead to the compromise of the whole IoT system. Therefore, we proposed, in this paper, a robust framework to detect the malware that targets the IoT system. We named our proposed framework RoMADS, which refers to Robust Malware Anomaly Detection Systems (MADS).

In IoT systems, the objects contact and interact with each other in a permanent and continuous way. The percentage of their exposure to any penetration is very high in comparison with the traditional networks. So, a continuous monitoring of the data traffics and devices is required. Therefore, using an MDS has an effective role in detecting strange malware behavior before causing any damage to the system. To improve the effectiveness of the IoT system, there is need to think of saving energy and time [1]. In traditional networks, the connection is between a server and a client, so high-security levels can be provided by adding a firewall and protection programs to the server. But in IoT network,

every object is connected to each other; therefore, the penetration of one device may cause the compromise of the entire IoT system, hence the necessity of using MADS in IoT system. The IoT security field needs more research and studies to ensure a high degree of security and to gain the confidence of people to live in a smart society with a high degree of security and privacy [1]. According to Kaspersky statistics [4], most of the violations in the IoT environment are caused by malware. It is a malicious software or program that is designed to penetrate different devices, systems, and networks, for specific purposes, either to sabotage, or damage networks or devices, or make money in an illegal way. It performs various functions such as encrypting, deleting sensitive data, stealing data and information, spying, or disrupting systems [5,6].

Therefore, this is the motive to conduct this paper in the field of MADS based on DL in IoT environments, as a contribution to improve the security in IoT environments and achieve the main goal of the IoT, which ensures a high life quality and security to facilitate human lives and get their trust.

RoMADS framework includes IoT architecture, IoT network, IoT devices, and the MDS based on LSTM Autoencoder. The main goal of RoMADS framework is to contribute in improving security in IoT environments and overcoming malware attacks. Our Contributions in this paper are as follows:

- The proposed solution is a robust MADS based on LSTM Autoencoder that has the capabilities to detect malware and gets a high accuracy, called RoMADS.
- RoMADS model performance was evaluated and we compared its results with the previous research results in this field.
- RoMADS model is distinguished by its ability to extract the features of data and reduce dimensions.
- It is also distinguished by its ability to take sequential data which has a different or fixed length. This is appropriate for the nature of the data in the IoT environments.
- RoMADS framework helps to take advantage of the devices currently on the market and adapt them to benefit from them in the IoT environments, which contributes to the rapid spread of the IoT and the increased dependence on it.
- RoMADS model contributes to saving time and energy consumption, as it checks every device that wants to enter the IoT network and converts its traditional protocol to the IoT protocol if need. Also, RoMADS framework depends on the fog computing architecture that is based on the principle of distributing the processing, which helps to reduce the time and energy consumption.

So, this paper explains RoMADS framework for MADS in IoT, which is proposed to protect the IoT environments from malicious attacks. In addition, it explains the related theories, RoMADS proposed framework, performance criteria, scenarios and algorithms. After that, the paper presents Simulation Assumptions, Simulation Scenario, Simulation Environment, Result & Discussion, and finally the Conclusions.

2. Background

2.1. Autoencoders

An autoencoder is considered a type of neural network and a data compression algorithm. The main principal function of the autoencoder algorithm depends on encoding and decoding. It takes a large amount of data, compresses it, and extracts the main features that have a major impact on the data. After that, it uses the compressed data in training neural networks. The applying of autoencoder algorithm helps in improving the learning process and reducing the time needed for training the model. After completing the training process, the autoencoder decodes and reconstructs the data. Then, it compares the original data with the reconstructed data to get the prediction error. This algorithm improves the accuracy by a gradual self-training. In the past, it was used only for feature learning or dimensionality reduction, but nowadays it is also regarded as a generative model [7]. The Autoencoder algorithm merges the most important data features to get the least possible dimension reduction. These new features are called the latent variables, or the encoded

features. The benefit of using latent feature representations (encoding) is to improve the performance of the model and reduce dimensions, which will result in the improvement of the training time [8].

Autoencoder algorithm is very suitable to be used in IoT environments, due to its nature. Hence, the use of this algorithm helps reducing the size of data, removing the noises, extracting the main features of the data as well as reducing the data dimensions. This contributes to the improvement of data quality before using it in training the MDS models, reduction of the time spent in training, and the enhancement of the speed and accuracy of the models.

In the following equations, *Xi* refers to the input data, $\widetilde{X}$ refers to the reconstructed data, *E* refers to the encoder function, *D* refers to the decoder function, *z* refers to the compressed data, *W* refers to the weight, *B* refers to the bias, σ refers to the activation function, and *L* refers to the loss function. The autoencoder architecture consists of three main layers [7,8], as shown in Figure 1.

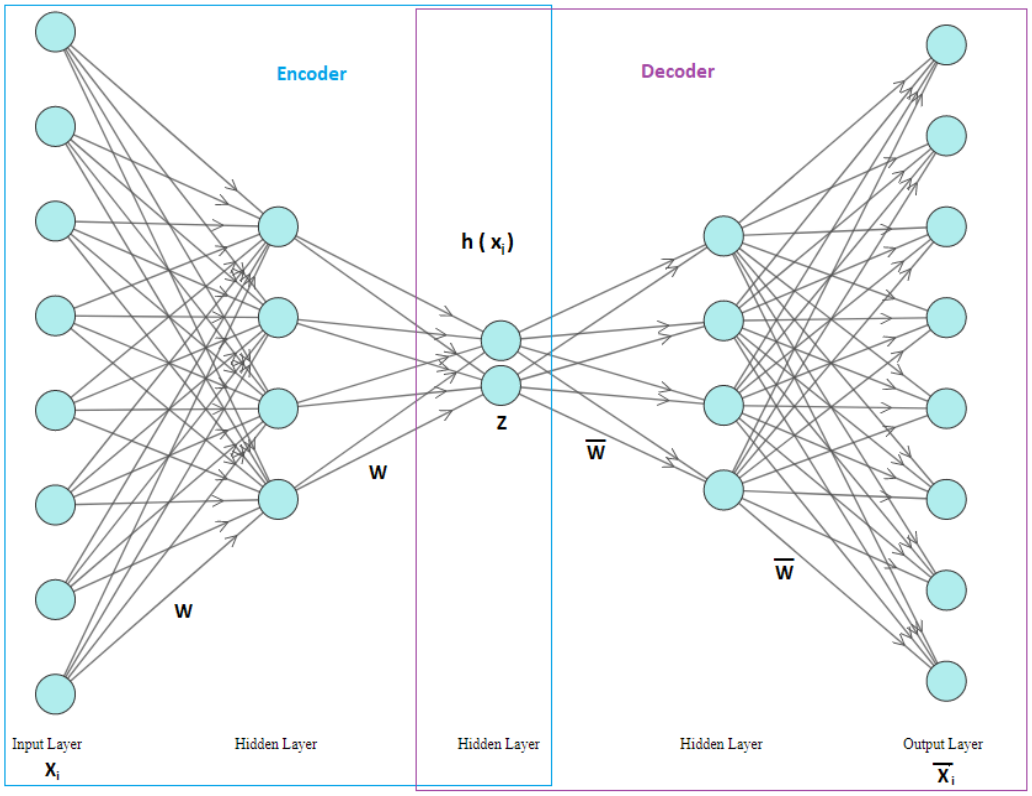


Figure 1. Autoencoder Architecture.

- Inputs layer, the original data is entered from input layer.

$$X_i : \{X_1, X_2, X_3, \ldots, X_i,\} \quad (1)$$

- Outputs layer, the reconstructed data comes out through the output layer.

$$\widetilde{X}_i : \left\{\widetilde{X_1}, \widetilde{X_2}, \widetilde{X_3}, \ldots, \widetilde{X_i}\right\} \quad (2)$$

- Hidden layers, it contains several layers.

The Autoencoder consists of four main components [9,10]:

a. Encoder:

In this phase the original data is entered into the model. The model starts learning compresses data and reduces its dimensions to obtain the encoded data [11].

$$E{:}X \rightarrow Z \tag{3}$$

$$\sigma(n) = 11 - e - n \tag{4}$$

$$Z = \sigma(Wx + B) \tag{5}$$

b. Bottleneck

In this layer, there is compressed data that has the lowest possible dimensions. It is called the latent-space representation [9].

$$Z = h(Xi) \tag{6}$$

c. Decoder

At this stage, the model is taught and trained to reconstruct the composed data and create data that it is very similar to the input data [9].

$$D : Z \rightarrow \widetilde{X} \tag{7}$$

$$\widetilde{X} = \widetilde{\sigma}\left(\widetilde{W_z} + \widetilde{B}\right) \tag{8}$$

d. Reconstruction Loss

Through the reconstruction loss method, the difference between the original data and the reconstructed data is calculated, to assess the performance of the decoding and how close the new reconstructed data is to the original data [12].

$$L\left(X, \widetilde{X}\right) = \|X - \widetilde{X}\|^2 = \|X - \widetilde{\sigma}\left(\widetilde{W}(\sigma(W_X + B)) + \widetilde{B}\right)\|^2 \tag{9}$$

For malware detection models, the autoencoder algorithm helps to detect malware by observing the error rate in reconstruction of the data, which is referred to by the loss rate. If the loss rate is large and exceeds the threshold, this means that something abnormal or anomalous has entered into the system [9].

2.2. LSTM

Long Short-Term Memory (LSTM) is a distinctive type of Recurrent Neural Network (RNN), it was developed to overcome the problem of vanishing gradient, which leads to the loss of information when backpropagation is applied. In 1997s, Hochreiter & Schmidhuber proposed the LSTM to solve one of the most complex tasks in DL, which is the sequence prediction. LSTM has the ability to learn long- and short-term dependencies [13].

The main part of LSTMs is the cell state, which allows the information to pass through it and apply numbers of processes. The way it functions is similar to the conveyor belt. LSTM organizes data traffic by gates, where it can add or remove data from a cell state via these gates. Every gate consists of a pointwise multiplication operation and a sigmoid neural network layer. The Figure 2 presents LSTM cell architecture; it has four neural network layers: input gate, output gate, forget gate, and cell state, and they have special dedicated ways to interact with each other. Each one of them has its specific task to do [14,15], as following:

a. Forget gate (*ft*) (a sigmoid layer)

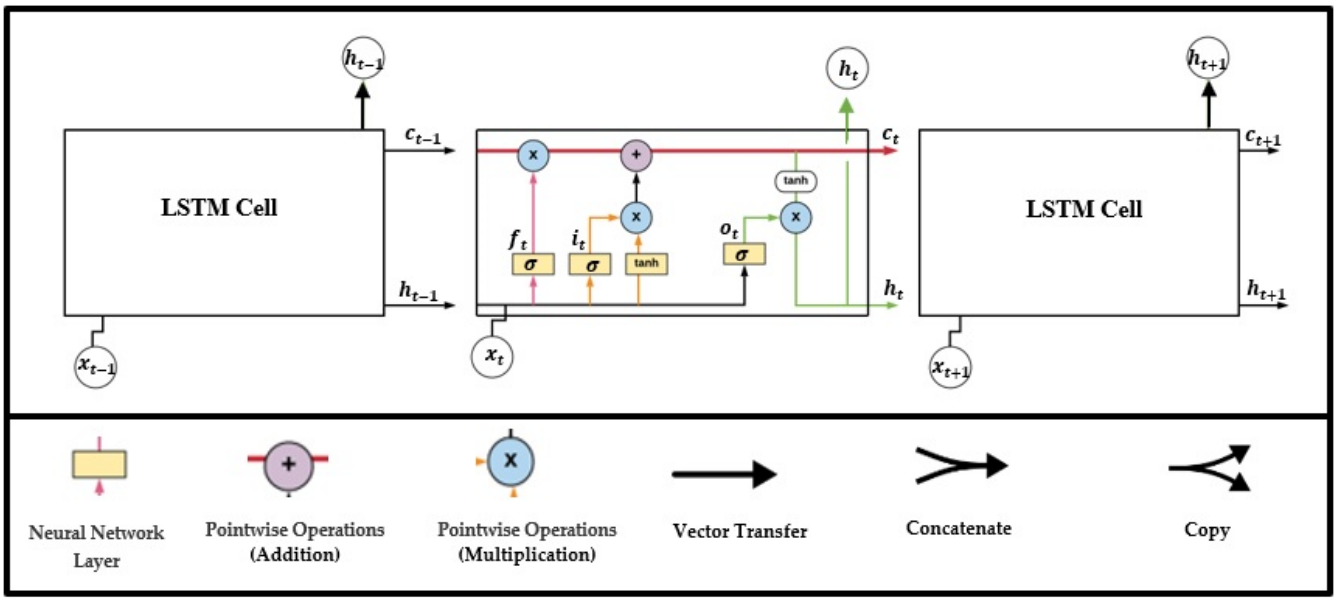


Figure 2. LSTM architecture.

It is the first phase for the LSTM process. This gate has an important role in improving the performance of the network by removing unnecessary information to complete the task. The forget gate takes two entries; Xt refers to the new information and $ht-1$ refers to the hidden cells of the previous cells. The outputs are between zero and one for each element in the cell state $Ct-1$. If the output is equal to zero, it means do not take anything from the previous cell. If the output is equal to one, the cell will keep everything from the previous cell. Wf refers to weights; bf refers to bias.

$$ft = \sigma(Wf \cdot [ht-1, Xt] + bf) \tag{10}$$

b. Input gate

The input gate is responsible for the decisions making with respect to the new information that will be stored in the cell state and responsible for entering new information into the cell. This phase is divided into two parts:

- Input gate layer (it) (a sigmoid layer):

This layer is responsible for deciding which value to update.

$$it = \sigma(Wi \cdot [ht-1, Xt] + bi) \tag{11}$$

- A tanh layer

This layer is responsible for taking the candidate value Ct and creating a vector for it, to add it into the current cell state.

$$\widetilde{Ct} = tanh(WC \cdot [ht-1, Xt] + bC) \tag{12}$$

The previous steps are combined to create a new cell by performing a multiplication operation between the value of the previous cell and the forgetting gate; then, the product of the operation ($it \times \widetilde{Ct}$) will be added to them.

$$Ct = ft * Ct-1 + it * \widetilde{Ct} \tag{13}$$

c. Output gate

This gate is responsible for selecting and extracting the necessary information only. This phase is divided into two parts:

- Output gate layer (ot) (a sigmoid layer):

This gate is responsible for deciding the output value.

$$ot = \sigma(Wo \cdot [ht - 1, Xt] + bo) \quad (14)$$

- A tanh layer:

After that, tanh function is applied on the value of the cell state Ct, to get a value between 1 and -1, then it multiplicities the result with the output value ot.

$$ht = ot \times tanh(Ct) \quad (15)$$

d. Cell state

The Cell state organizes the input sequence, allows data to pass through specific mechanisms, and organizes the passage of information through gates. Moreover, it stores the additional values of the cell over time [14,16].

The problem of sequence prediction was one of the most difficult problems facing scientists before the LSTM is proposed. The LSTM often overcomes RNN and conventional feed-forward neural networks in many applications, due to its distinctive characteristics, and its ability to select and remember for a long period of time [17]. One of the most important advantages of using LSTM network in anomaly detection models is their ability to analyze one or more features at one single time, which allows the early detection of any strange processes by malware. It is also distinguished by its ability to train models to very deep layers and the ability to memorize data for a long time. These make it suitable to use it for anomaly detection in IoT environments and big data [18,19].

3. Literature Review

Today, the world lives in a new revolution; it is the IoT revolution, where everything has become connected to everything, and everything is permanently connected to the internet. But some simple IoT devices are characterized by low energy consumption and low complexity, which makes them an easy target for malware. Most manufacturers of these devices are concerned with ease of use and have not given much attention to privacy and security. Among the reasons that made the IoT devices an easy target for attackers is their lack of encryption techniques, as well as the use of the weakness of passwords and the backdoors that manufacturers put in order to support and develop the product, but hackers exploit it and it becomes a front door for hackers.

As a result of the nature of IoT networks and devices and their permanent connection to each other and to the Internet, and the presence of a large number of connected devices, a penetration of one of the devices easily leads to the penetration of all devices on the IoT network. This made the IoT environment an easy target for attackers to build Botnets malware that are subsequently independent such as a DoS malware, ransomware, or others. Malware in IoT networks differs from the traditional networks in that it does not penetrate just thousands or tens of thousands of devices, as in traditional networks. It actually penetrates hundreds of thousands of devices in IoT network [20].

IoT Malware Attacks

IBM developers claim in [21] that the main goals of attacks in the IoT environment are dominance, control, and exploiting devices. Therefore, the attacks are divided into two phases:

a. The first phase: scan and takeover

In this phase, the attacker works to control the largest number of IoT devices using their IP addresses then converts them into bot units; then, they can be used to perpetrate attacks.

b. The second phase: attack launch

Using one unit of the botnet does not cause significant damage, but if the attacker controls an army of botnets, he/she can cause significant and frightening damage. The hacker often uses his/her army for DoS attacks or spambots. Figure 3 presents IoT Malware attacks.

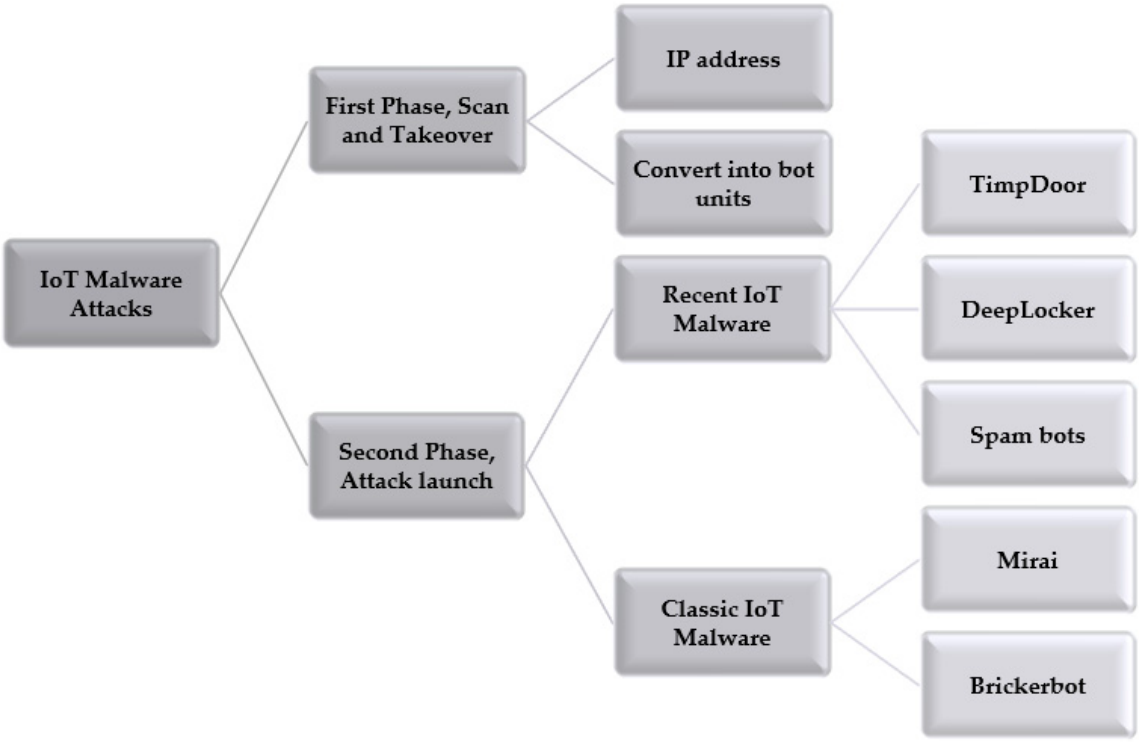


Figure 3. IoT Malware attacks.

Often these two phases are performed using the Command and Control (CNC) program. Some types of IoT malware families are introduced below [21]:

i. DoS Attack

Hackers use hundreds of thousands of IoT devices to send requests to the target host, which results in the host's inability to respond to this huge number of requests; consequently, the service of this host will be interrupted.

ii. Spam Bots Attack

The hacker sends a lot of spam emails to a specific email by an army of IoT devices that the hacker has controlled.

iii. Brickers Attack

Malware is designed to destroy IoT devices and make them unusable.

iv. Cryptomining Bots Attack

It is a type of botnet, which always uses infected IoT devices to mine cryptocurrencies such as Moner. It often works in the background.

v. TimpDoor Attack

It is a type of malware that targets Android devices by asking the user to install a fake voicemail application. Then, it uses the infected device as a proxy server for encrypted traffic. Hackers can detect the entire user's network, whether it is a home network or a private company one.

vi. DeepLocker Attack

IBM Researchers developed a new type of malware based on AI, and they used a prototype based on a Deep Neural Network (DNN) technology to show how dangerous this type is and the possibility of its spread in the future. They urge the researchers to get ahead to resist this new type of malware.

vii. Mirai Attack

It controls hosts with an open port 23 (telnet) for using them to create an army that disrupts traffic and overpower and disable the target host. Figure 4 presents IoT malware.

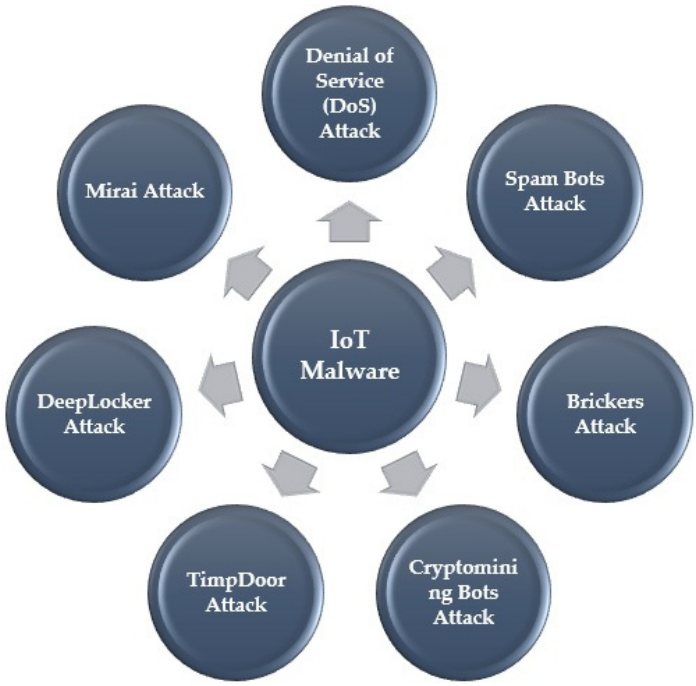


Figure 4. IoT Malware.

Scientists and researchers have been interested—over time—in researching and finding appropriate solutions to obtain an IoT environment that is secure and free of security-privacy threats, and until now, research is still conducted regarding these points.

V. HASSIJA et al. l discussed in [22] a security solutions in IoT environments. They divided them into four main categories: IoT security solution based on blockchain, IoT security solution based on fog computing, IoT security solution based on edge computing, and IoT security solution based on ML.

S. Malge and P. Singh discussed in [23] the main layers in the IoT architecture from a security perspective and the threats each layer is exposed to. Then, they discussed a number of techniques used in securing IoT environments and maintaining user privacy, such as Mechanism of Encryption, Secure Communication, Secure sensor, and the Algorithms of Cryptography.

L. Xiao et al. [24] discussed the role of AI and ML in securing IoT. Then, they discussed four types of solutions that depend on ML to secure the IoT environment, which are ML-based authentication, ML-based access control, ML-based secure offloading, and ML-based malware detection. The authors also discussed the challenges related to security in IoT environments that need to be addressed, in order to obtain ideal IoT environment appropriate for human life safely.

Scientists and researchers have tended to use AI in MDS to increase its efficiency, effectiveness, and suitability for the rapid development that humanity is experiencing at this time. Table 1 presents comparison between studies [25–30] that related to IoT MDS based on DL techniques.

Table 1. Related works comparison.

Ref.	Proposed Solutions	Techniques	ML/DL	Detection Approaches	Dataset	Scalability	Result
[25]	MDS based on DL called BDLF, which combines behavior graphs of API calls with the SAEs model.	SAEs, DT, KNN, NB, and SVM	DL, ML	Anomaly	Collected the dataset sample by VX Heaven	The system is scalable due to its dependence cloud	The results of the experiment showed an improvement in the detection precision of the model.
[26]	Using Fuzzy pattern and the fast Fuzzy pattern for MDS	Fuzzy pattern	ML	Anomaly	They used four datasets, which are IoT, Vx-Heaven, Kaggle, and Ransomware	-	they have proven the effectiveness of their proposed.
[27]	Converting the binary codes of the malware into images	CNN	DL	Anomaly	Malware image data from Vision Research Lab	-	High accuracy and low loss ratio
[28]	Building a MDS for Android devices based on SVM	SVM	ML	Anomaly	Create dataset using application in the actual mobile environment	-	The results proved outperforms other ML techniques.
[29]	Building a MDS for Android devices	Random Forest algorithm	ML	Anomaly	Antimalware dataset	-	Their model has yielded promising results
[30]	A framework that is a botnet detection system based on two-level DL framework. The first level is responsible for similarity measures in the DNS services based on a predefined threshold to deter the most common DNS information over Ethernet connections. In the second level, they proposed to add an algorithm for a domain generation depending on DL architectures to use it for classifying the domain names into normal and abnormal ones.	Number of DL techniques	DL	Anomaly	The similarity checker dataset is called DS1, and the DGA detection dataset is called DS2.	Highly scalable due to use the DNS big data.	The result of their experiment ensured the improvements in detection rate, F1-score, and false alarm rate.

In paper [1], researchers conducted a comprehensive study on security solutions in IoT applications, IDS, MDS, and the role of AI in improving security in IoT. After they analyzed more than one hundred state of the art research studies, they recommended continuous research in the field of MDS based on DL in IoT environments. It is still a new field that needs more research and studies.

In fact, the field of research in the MDS based on DL in IoT environments is still a new field that needs more research and studies. Also, it needs development and improvement in several aspects such as detection rate and accuracy, to obtain high-security IoT environments, and to maintain the privacy of users to gain their confidence and increase their reliance on IoT environments. This will contribute to achieving the main goal of IoT, which is to obtain comfort and security for users. Therefore, it is necessary to propose a robust MDS based on DL that has a high ability to detect renewable malware by getting MDS work as a human mind to solve the problem of security in IoT environments.

4. Methodology

This research paper is an exploratory study to investigate how DL techniques are utilized, to improve MADS in IoT systems, and maintain security and privacy. After the state-of-the-art of studies and research related to improving the security in the IoT system were examined, it was highly recommended to take advantage of the capabilities of DL and harness them to increase the efficiency of MADS and improve the level of security in IoT environments. As far as the researcher knows, and based on our literature review, the research topic has been chosen for the present study is considered a new and unexplored topic that needs further study and dedicated research.

So, the proposed solution was a robust framework for malware anomaly detection system based on LSTM Autoencoder DL techniques. The primary goal is to improve the level of security in IoT environments and protect them from malware, then increasing the confidence of humans by relying on them completely. To conduct this research, a secondary dataset (which is IoT malware dataset) is selected to evaluate the effectiveness of the proposed framework. This dataset is DS2OS traffic traces [31]. The dataset is cleaned, processed, and divided into train dataset, validation dataset, and test dataset, to use them in a train and test the proposed model, as shown Figure 5.

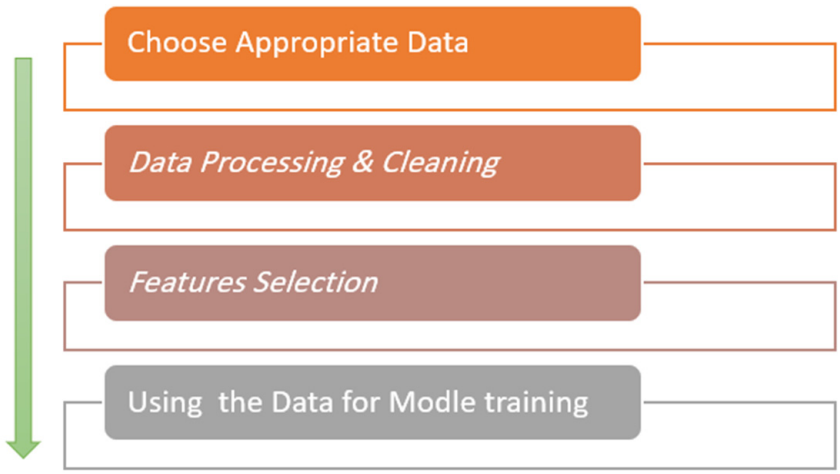


Figure 5. Data processing steps.

Finally, the proposed model is evaluated based on simulations. After that, the experiment results are discussed and compared with other models in previous works, in order to prove the efficiency and effectiveness of the proposed solution in improving the security and privacy within the IoT environments.

4.1. Data Set

To conduct the experiment, the samples of malware and benign were captured from IoT environments in line with the proposed framework are selected. The selected dataset is:

- DS2OS Data Set

The DS2OS traffic traces -IoT traffic traces gathered in the DS2OS IoT environment. They prepared this dataset using simulated IoT positions with several kind of services: smart doors, light controller, movement sensors, thermometer, batteries, washing machines, thermostats, and smart phones. Each position had a several kinds of services and different organizations [31]. The samples were collected from the communications between different IoT nodes. Table 2 presents the features of DS2OS dataset and the type of each feature. The dataset contains 357,941 samples 347,924 normal samples and 10,017 malware samples. This dataset was chosen because it contains a large number of normal samples and a smaller number of malware samples which is closer to the actual reality in IoT environments. Figure 6 presents the distribution of the normal and malware samples.

Table 2. DS2OS dataset features.

Features	Type
sourceID	String []
sourceAddress	String []
sourceType	String []
sourceLocation	String []
destinationServiceAddress	String []
destinationServiceType	String []
destinationLocation	String []
accessedNodeAddress	String []
accessedNodeType	String []
operation	String []
value	String []
timestamp	Integer []
normality	String []

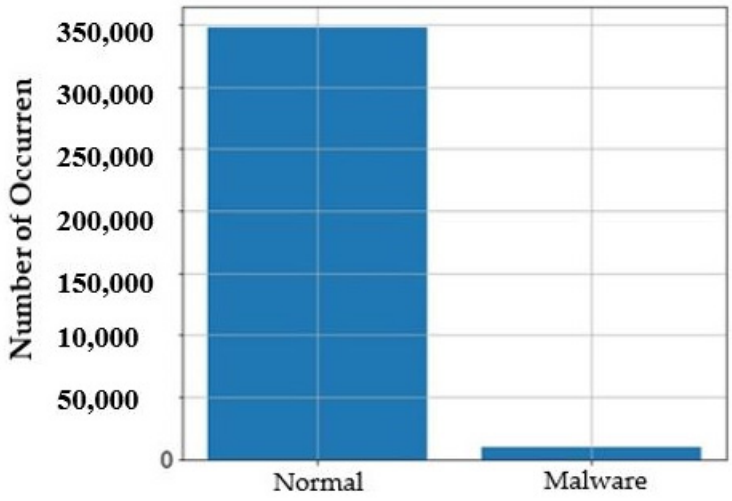


Figure 6. Illustration the number of normal and malware samples.

4.2. Data Processing & Cleaning

In the beginning, the data are reviewed and visualized in order to understand them and ensure their integrity. Figure 7 presents the "normality" column distribution and Figure 8 presents the "Accessed Node" column distribution. This dataset is clean and just some rows from the dataset are removed because there are some cells (in value attribute column) were blank. So, the entire row was removed. Dealing with the scikit-learn library [32] and ML algorithms needs accurate and error-free data to avoid the problems and errors during the training of the model. Then, the target column is determined based on the research topic, which is a robust framework for MADS in IoT environments. The "normality" column is defined as a target column. It is created as a variable independent vector. This column contains normal and malware samples. It contains six different families of malware, as shown in Figure 7.

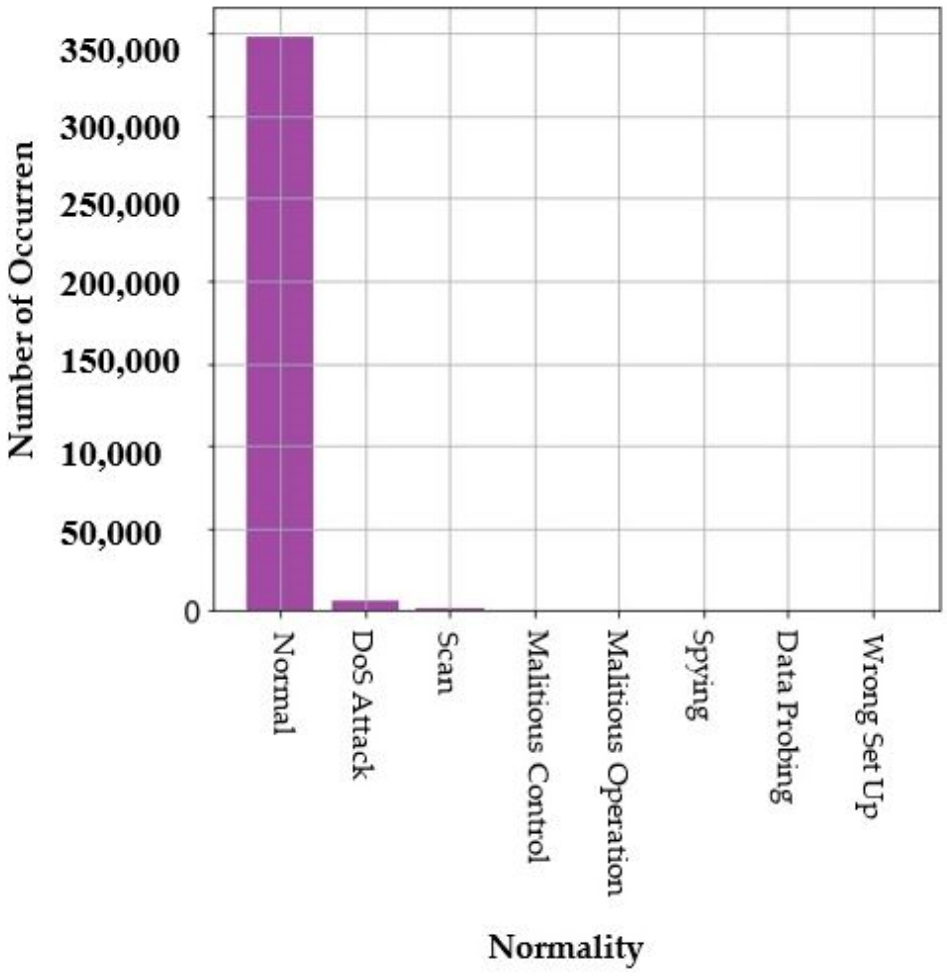


Figure 7. "Normality" column distribution.

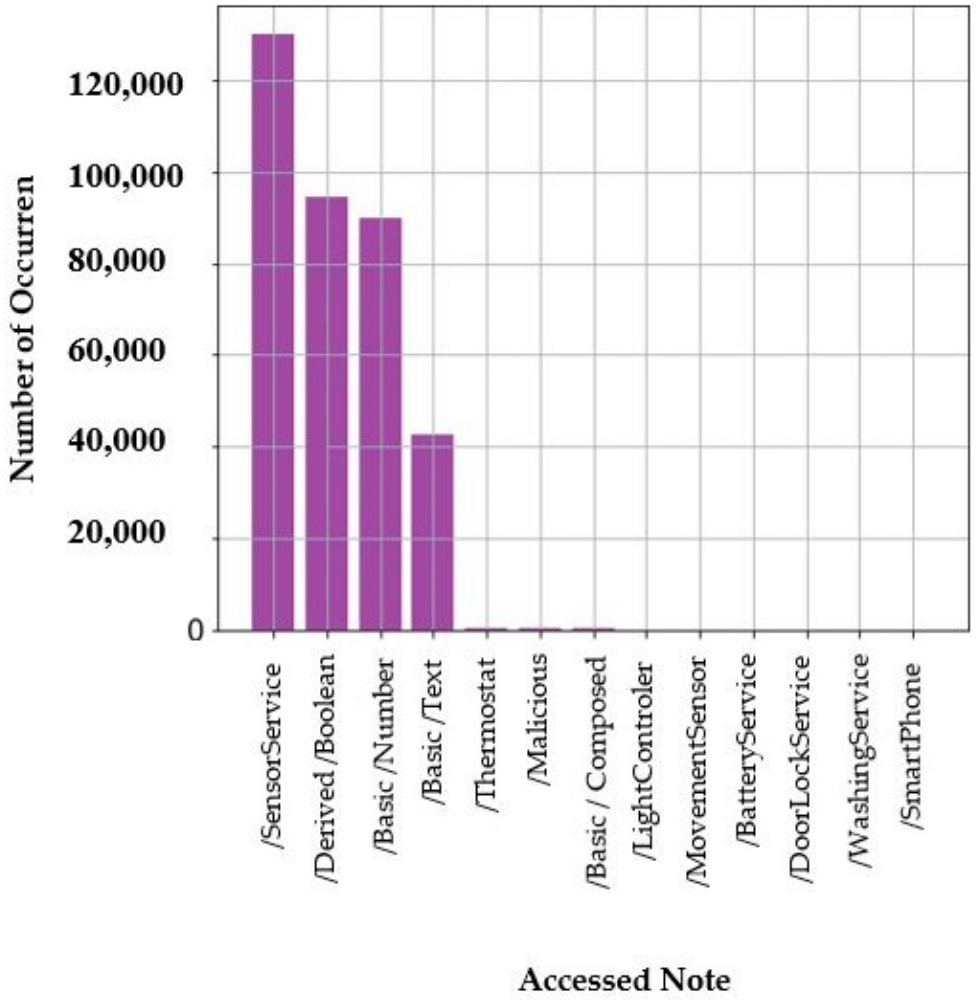


Figure 8. "Accessed Node" column distribution.

The ML algorithms that were used to implement the model do not deal with the categorical data, so the categorical data in the dataset are converted into numerical data by applying the Label Encoder technique. This enables the algorithms to understand the dataset and analyze it. Also, the features selection techniques are applied to identify the features that have a strong relationship with the target column, as well as to reduce the number of features. This will contribute to improving the prediction accuracy.

4.3. Feature Selection

Features Selection techniques have an essential role in improving the model's performance. In this experiment, two of the features' selection techniques are applied. They depend on the ML to help representing and understanding the data. Then, the features that had the highest impact on our target column (Normality) are selected to obtain accurate data, which will make it easy for the model to understand and obtain the model with a higher predictability.

The Correlation Matrix with Heatmap technique [33] is used by the Seaborn library to present all the features of the dataset and their relationship with each other. This helped to determine their relationship with the target column and selected the features that have a strong relationship with it. Figure 9 illustrates the features in the dataset and their relationship with each other.

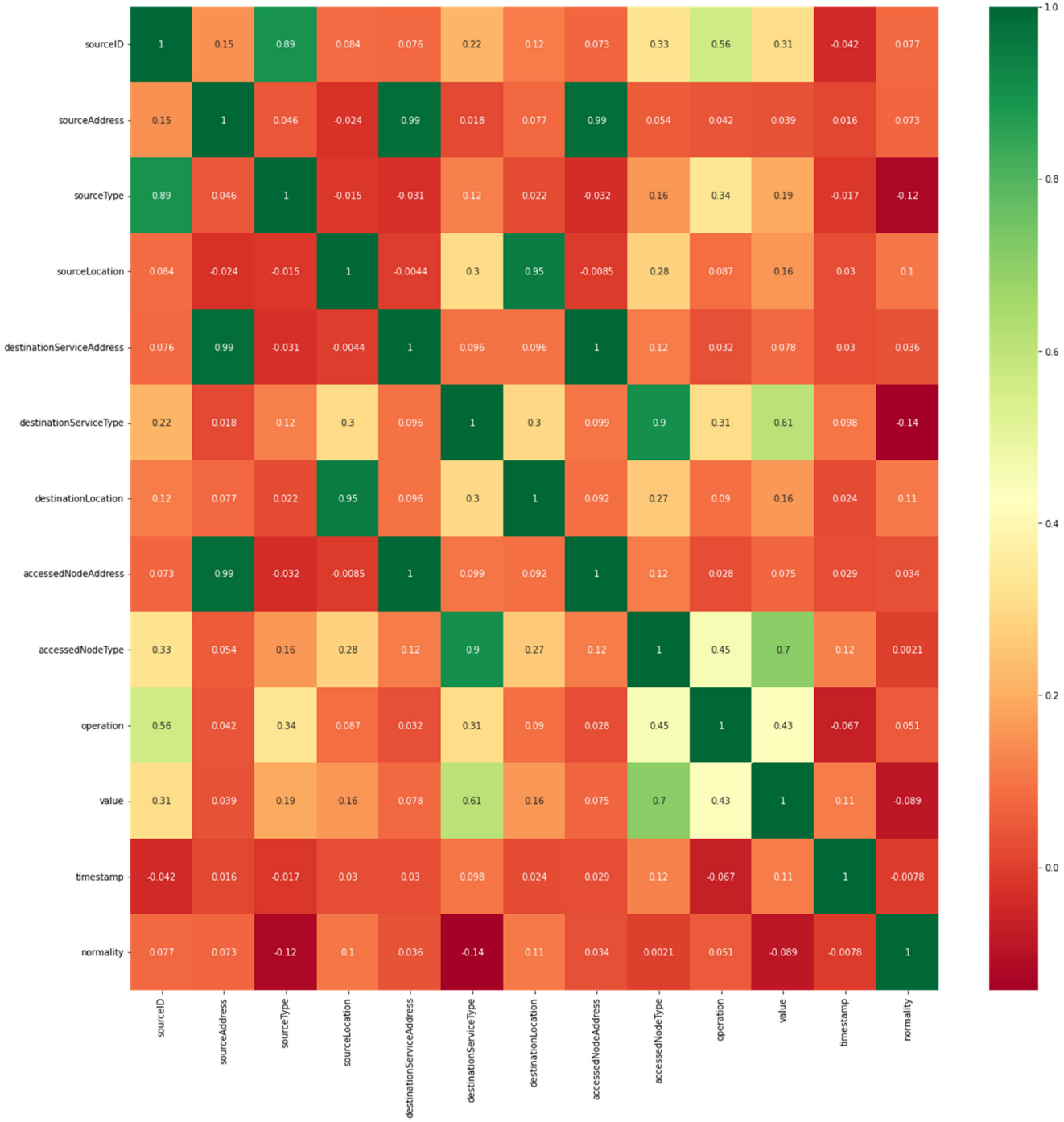


Figure 9. Illustration of the relationship between the dataset features.

On the other hand, the Univariate Selection technique based on statistical tests [32] is applied to find the features that have a strong relationship with a target column. The chi-squared (chi^2) statistical test from the scikit-learn library—which provides the SelectKBest class—is implemented, and the top five features that have a strong relationship with the target column are determined as shown in Figure 10. After applying these two feature selection tests, the best 5 features that have a strong relationship with the target column are selected. So, the number of features of the dataset was reduced from sixteen to five features, as is evident in Table 3. The features that have strong relationships with each other were chosen.

	Specs	Score
10	value	2.770316e+07
0	sourceID	1.367500e+05
7	accessedNodeAddress	5.005638e+04
1	sourceAddress	4.536765e+04
4	destinationServiceAddress	2.647609e+04

Figure 10. Top 5 features that has a strong relationship with the target variable.

Table 3. DS2OS dataset features after Feature Selection.

Features	Type	Description
sourceID	String []	The ID of the source.
sourceAddress	String []	The Address of the source.
destinationServiceAddress	String []	The Address of the destination Service.
accessedNodeAddress	String []	The Address of the accessed Node.
value	String []	The value of the package.
normality	String []	This column contains normal and six different types of malware. We used it as a target column.

4.4. Experiment Environment

The simulation on Google Colaboratory [34], which is an environment based on the cloud, supports the Python programming language, and provides free access to GPUs.

5. Proposed Solution

This section explains the proposed model RoMADS in detail under four subsections, which are RoMADS architecture, RoMADS network, RoMADS devices, and RoMADS MDS based on LSTM Autoencoder. As shown in Figure 11.

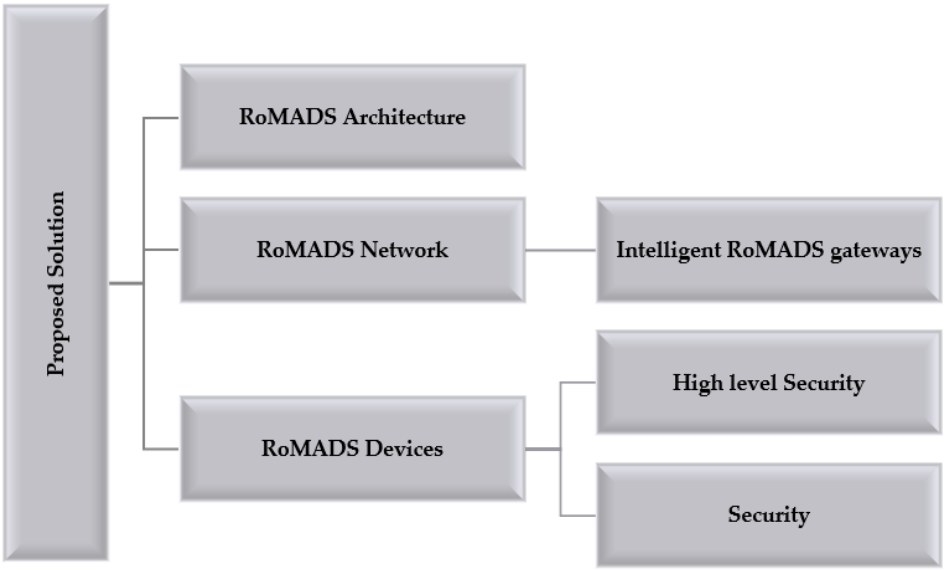


Figure 11. Proposed Solution Structure.

5.1. RoMADS Architecture

Until now, there is no an entirely reliable architecture for the IoT to trust and consider totally safe and secure. Thus, there is a need for more research and developments to obtain a higher secure architecture. Here, the proposed solution is a new architecture that combines fog computing architecture and IoT social architecture. The fog computing architecture is concerned with the IoT data in terms of analysis, processing, storage, and protection. It was proposed by Cisco [35] based on the principle of distributing the processing and load on a number of smart gateways and smart sensors to perform analysis and processing before storing them on the cloud, which helps to reduce the time spent to complete tasks as well as reduce problems resulting from the use of central processing.

On the other hand, IoT social architecture's [36] main function ideas are ideas taken from the social communication between humans. The devices recognize each other automatically without human intervention, identify the device that was previously dealt with, build trust, and determine the level of security based on their previous relation; therefore, when any new device is connected, security protocols are tightened until the confirmation that this new device is secure and trustworthy is provided.

RoMADS framework architecture combines these two architectures, which contributes to creating a more robust architecture concerned about data and relationships between devices. It also works on improving the performance of real-time systems.

5.2. RoMADS Network

This section explains the intelligent RoMADS gateways and how the RoMADS model deals with the protocols.

Intelligent RoMADS Gateways

RoMADS framework contains a number of distributed intelligent IoT gateways, as shown in Figure 12, which act as an intermediary between the IoT network and external networks such as the Internet. Any communication between the IoT network and external networks must pass through one of the RoMADS gateways. The RoMADS framework distributes a number of intelligent gateways to overcome the overload problems of central processing, which causes services interruption or delay.

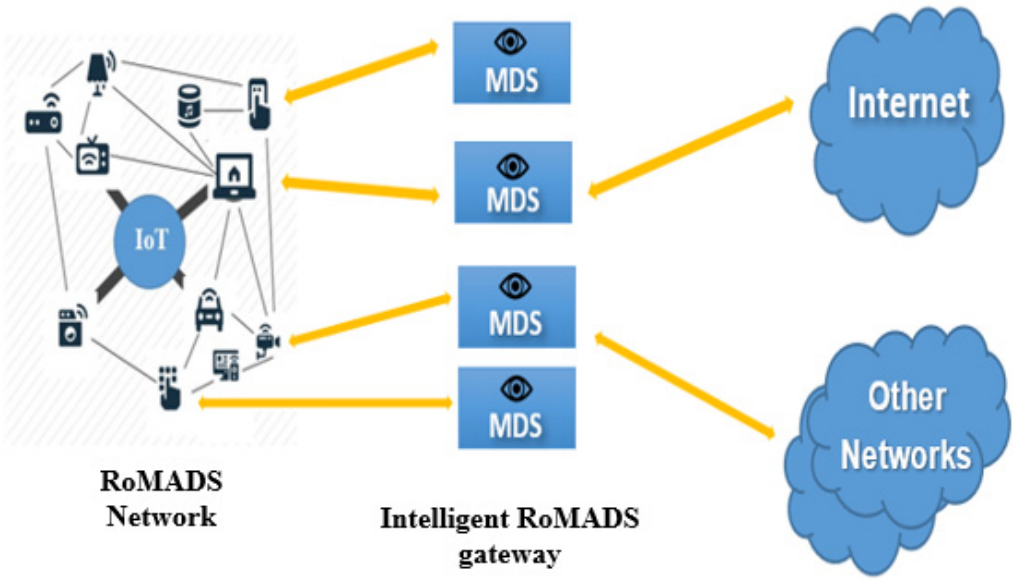


Figure 12. Intelligent RoMADS gateway.

The RoMADS framework adds some of the security procedures on the intelligent RoMADS gateway by giving them the ability to detect the malware using MADS based on

LSTM Autoencoder, which continuously monitors the data traffic. When any malware is detected, it sets an alarm to stop this malware and prevent it from entering the IoT system or causing any damage to it.

When the packages arrive from the Internet, the intelligent RoMADS gateway checks the packages; if those packages are secure and do not contain any malicious software, the intelligent RoMADS gateway allows them to pass and complete their path to enter the IoT system (as shown Figures 13 and 14). Otherwise, if they contain any malicious software, the intelligent RoMADS gateway sets an alarm to stop this malware and prevent it from entering the IoT system.

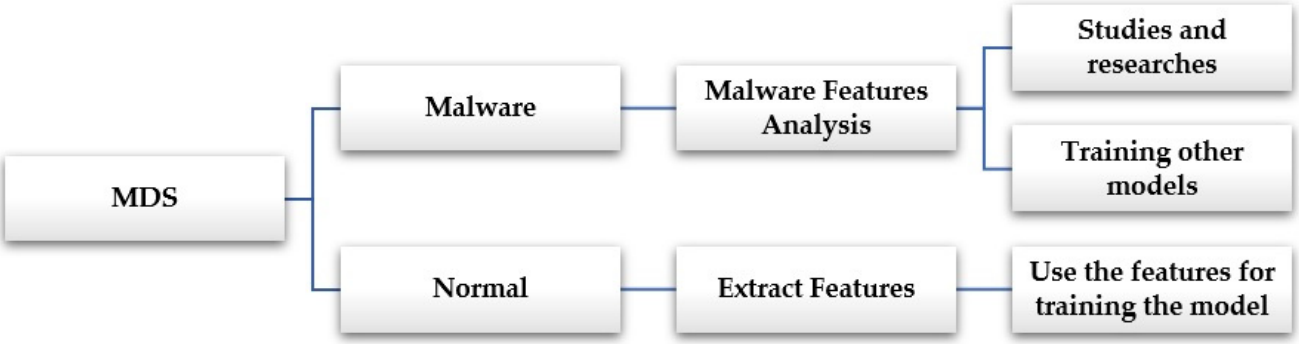


Figure 13. Normal and Malware features analysis on MDS.

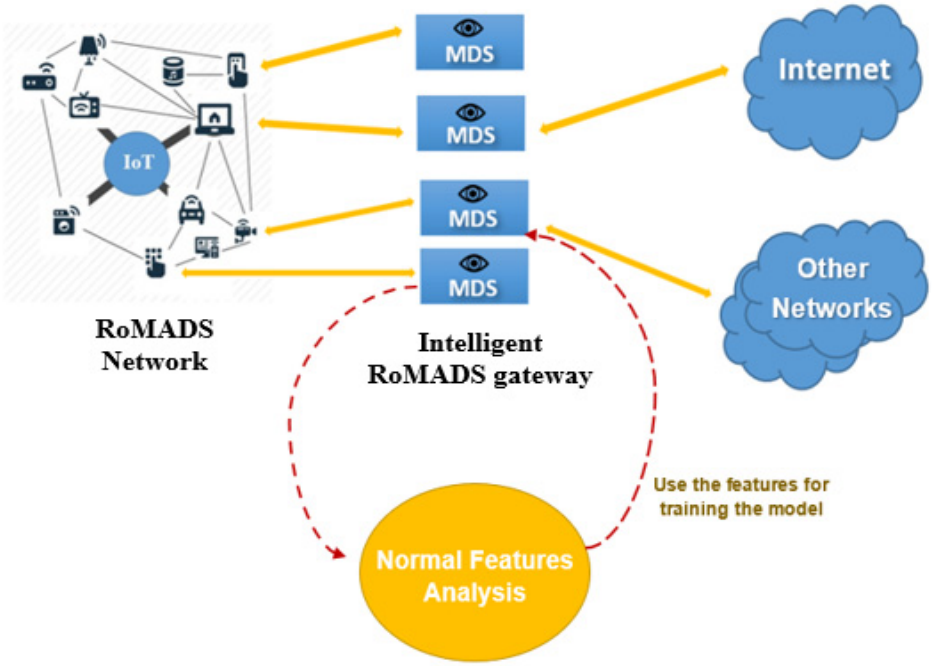


Figure 14. Normal features analysis.

Figures 13 and 15 explain the process of the Malware Features Analysis, which is located on the intelligent RoMADS gateway. It helps to understand the nature of the malware that target IoT systems analyzes them, extracts their features, studies their behavior, and uses them in the future for studies, research, and the training of other models. Moreover, it allows us to take appropriate precautions to prevent it and reduce their risks and effects.

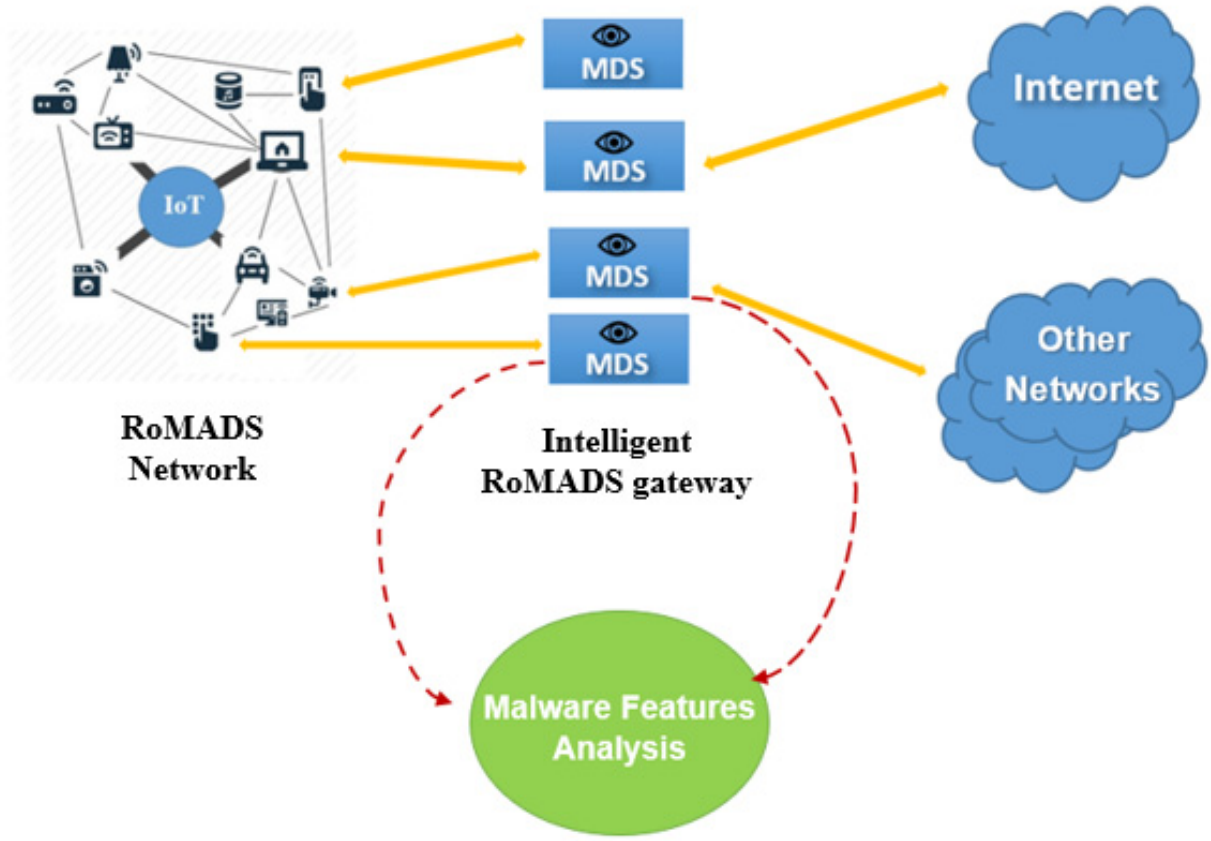


Figure 15. Malware features analysis.

IoT objects communicate with each other in IoT networks with simple and light protocols such as Constrained Application Protocol (CoAP), Message Queuing Telemetry Transport (MQTT), User Datagram Protocol (UDP), and 6LoWPAN [37]. They are often used in the IoT systems simple and uncomplicated protocols, to reduce the energy and time consumed. Therefore, one of the advantages of the intelligent gateways in RoMADS framework is their ability to convert and replace the traditional protocols for the objects that come from external networks to the IoT protocols, which are characterized by their simplicity and lightness. As an example, it converts from Hypertext Transfer Protocol (HTTP) to CoAP, switches IPv6 address to 6LoWPAN addresses, and Transmission Control Protocol (TCP) to UDP for the objects. This helps to rapidly spread the IoT systems, because it becomes possible for the markets now to benefit from these resources and adapt them using intelligent gateways, to suit the nature of the IoT system. The lightweight protocols, which are appropriate for the use in IoT systems, often lack attention to the security layer. So, the RoMADS framework overcomes this problem by distributing MADS that are based on LSTM Autoencoder in the IoT system in order to secure data and keep it free from malware when moving from one place to another.

5.3. RoMADS Devices

RoMADS framework classified IoT devices into two categories according to the importance of the device and the sensitivity of its data and information, as shown in Figure 16:

- Devices that need a high level of security because they contain sensitive data and critical user information, such as the following: passwords, user personal information, or financial information.
- Devices that need security but less than the previous one, because they do not contain any sensitive data or information.

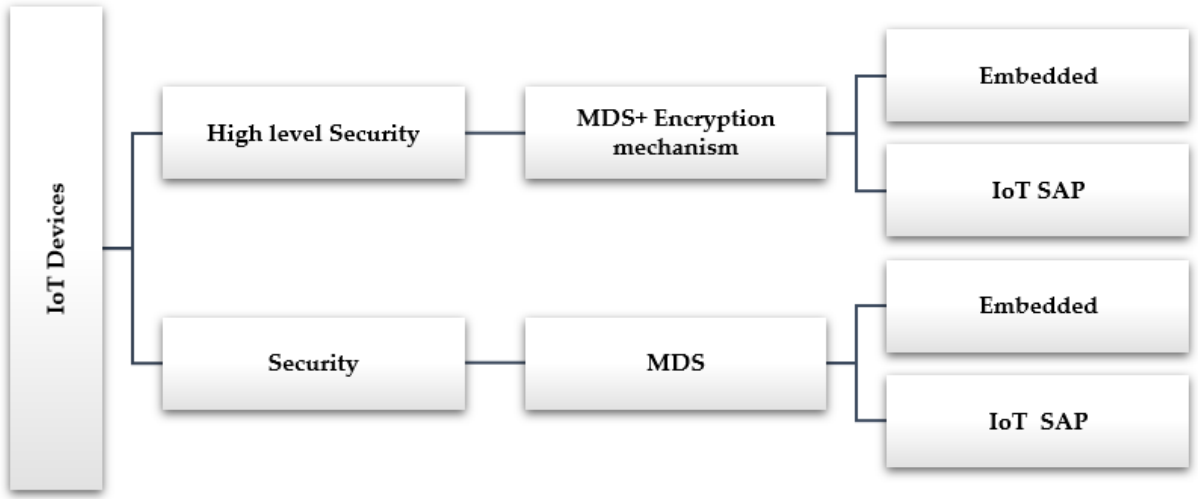


Figure 16. RoMADS devices.

(a) High-level security (MDS + Encryption mechanism)

The first category is for the devices that need a high level of security because they contain sensitive data, such as bank account information, personal data, and so on. The sensitive data are encrypted before sending them to any other IoT devices or to the external networks using encryption mechanisms. Where the sensitive data are transferred in encrypted form.

RoMADS framework classified devices need a high level of a security in two categories, an Embedded and IoT SAP. As shown in Figure 16.

- Embedded:

  MDSs and encryption mechanisms built into the devices from the factory.

- IoT Security Anchor Point (SAP):

  In this category, the light devices or the devices that lack the security mechanisms, such as encryption mechanisms and MDSs, are connected to IoT SAP that acts as an intermediary, and contains various protection systems. The IoT device must pass through the IoT SAP before contacting other IoT devices or the external networks, as shown in Figure 17.

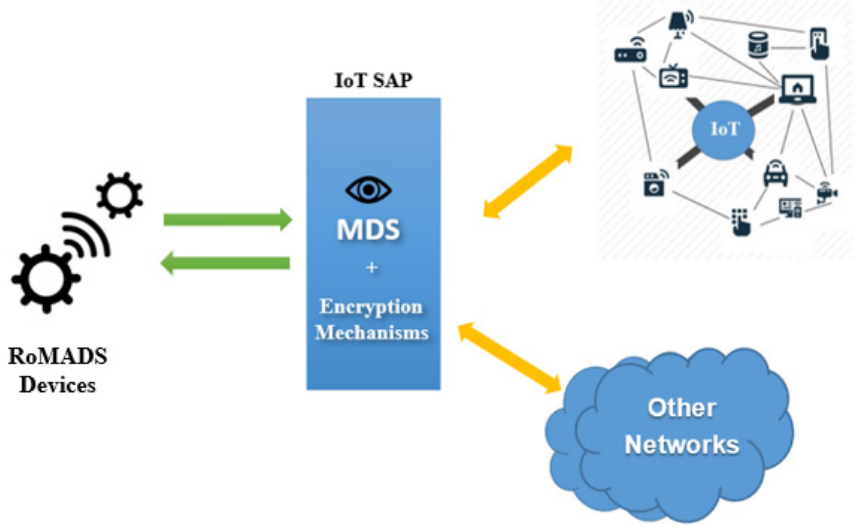


Figure 17. IoT SAP.

(b) Security (MDS)

The second category is for devices that do not contain any sensitive data. They do not need to add encryption mechanisms on them. They only need to add MDS to detect malware instead. Using a MADS based on LSTM Autoencoder is suitable for the nature of IoT devices, where MADS does not need the databases like the detection systems that are based on signatures. This makes it more suitable for the nature of IoT devices. IoT devices in the second category are classified into two categories, an Embedded and IoT SAP as in high-level security category.

As it can be noticed from the previous classification of IoT devices in the RoMADS model, it is very concerned with securing every device connected to the IoT network as it works continuously on managing and monitoring these devices in addition to being keen on regular security updates. Unsecured devices are the easiest gates for attackers to exploit, as every single device that is added to a network is considered a new gateway that attackers can exploit to penetrate the network. Because the penetration of any device into the network leads to a penetration into the entire IoT network, and thus ensuring the safety and security of each device on the network leads to securing the entire network and providing it with a strong protection against harmful attacks.

Most of the companies producing IoT devices are keen to produce devices that are appropriate to the nature of the IoT environment and provide the required services well without any concerns about the security aspect of these devices [2]. Hence, the RoMADS model helps in resolving and developing the security aspects of these devices, as it provides security for this type of device by connecting it to the IoT SAP.

The RoMADS model is distinguished by its ability to dynamically change permissions according to changes in the network and the type of connected devices. The RoMADS identifies each device connected to the network, then it determines its permissions and powers based on the concept of social relations and whether it was previously dealt with or not. The RoMADS also continuously monitors the devices on the IoT network to ensure that they are not exposed to attacks and take immediate measurements to reduce the resulting risks.

RoMADS also uses encryption techniques for encrypting important data, which provides more security for users' data. If hackers managed to steal them, either during transmitting them on the network or while being stored in the cloud, the hacker will not be able to understand it or benefit from it.

The RoMADS analyses malware to understand the targets of the attackers and the purpose of their attack, and then it takes the necessary precautions to prevent them from achieving their goals. Knowing the attackers' goals helps in confronting and misleading them.

RoMADS framework ensures that the IoT system is fully secured from malware. Whereas any data that comes from outside the IoT network must pass through one of the intelligent RoMADS gateways, which check the data and ensure their integrity and that they are free of any malware, then change their protocol to IoT protocols if it is needed before allowing them to enter the IoT network.

In addition, RoMADS framework secures the IoT system from inside the network, as all devices are secured either from the device itself (i.e., through the factory), or through IoT SAP, with light devices that do not contain protection programs and security mechanisms (such as a water boiler) that must be connected to IoT SAP and pass through it before conducting any communication inside or outside the IoT network.

5.4. RoMADS MDS Based on LSTM Autoencoder

The nature of data in IoT environments is often big, sequential and has different lengths, so the use of LSTM Autoencoder from DL techniques was suggested, which goes hand in hand with the nature of the data in IoT environments. It takes different lengths of sequential data as input and it also outputs the sequential data that have different lengths.

The structure of the LSTM Autoencoder is used to solve many complex sequencing prediction problems such as text translation and speech recognition. Chitta Ranjan [38,39] used it in rare event classification for multivariate time-series data. In this paper, it is used to detect the malware that tries to penetrate the IoT environments, as shown in Figure 18.

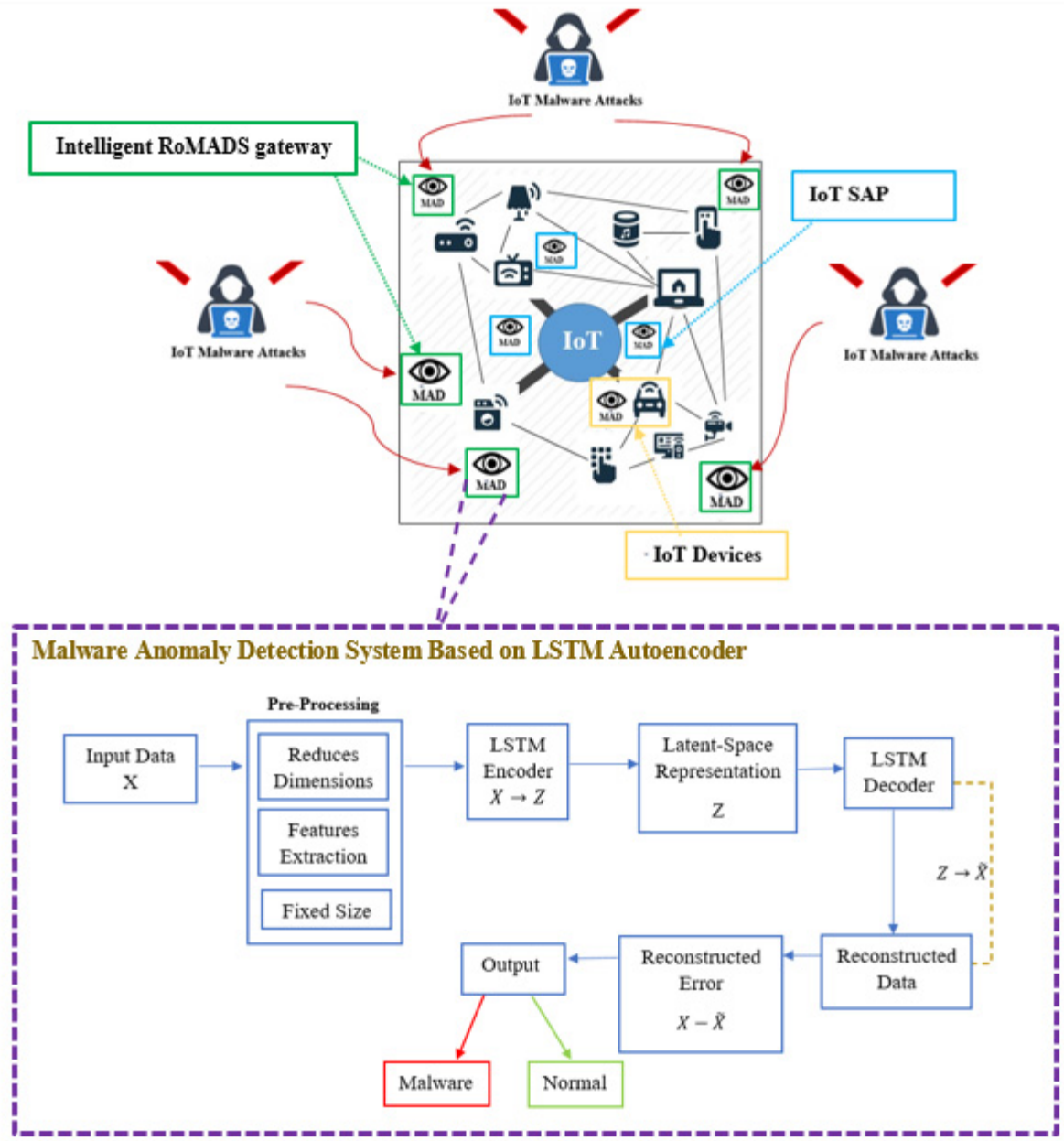


Figure 18. MAD based on LSTM Autoencoder on IoT system.

RoMADS model for MDS based on DL combines between LSTM networks and Autoencoder neural networks. Each of them has its own characteristics and advantages, so combining them will result in a robust model. Moreover, each of them makes up and offsets the deficiency in the other. Autoencoder [7,8] has the ability to deal with the sequential of data. It has a major role in reducing the dimensions, noising, and extracting features of the input data. In addition to that, it converts all the input data with different length to have the same fixed length, which are the encoders data. Then, it reconstructs the composed data to get new data similar to the original ones. Finally, calculate the reconstructed loss to get the accuracy of the model.

On other hand, The LSTM network [13] helps in deep and continuous training in case there is a huge amount of data that needs to be trained to very deep layers for long time to get high detection accuracy, as in IoT systems [40]. So, when combining an LSTM network with an Autoencoder neural network in a RoMADS model, we will have a robust MDS model used to protect IoT systems. Also, the LSTM networks is used in RoMADS model, because it has long and short-term memory features that allow for storing the important features of data for a long time, which contributes to obtaining a smart and intelligent model that has a high detection capacity.

Figure 19 presents the relationship between LSTM network and Autoencoder neural network for RoMADS model to detect the malware. The data are entered into Autoencoder $X_i : \{X_1, X_2, X_3, \ldots, X_i,\}$ to reduce the dimensions and extract the basic features of the data. LSTM cells are distinguished by their ability to remember data for a long period of time, as well as their ability to train data to very deep layers, which allows us to obtain a robust model that has the ability to detect malware and protect IoT environments.

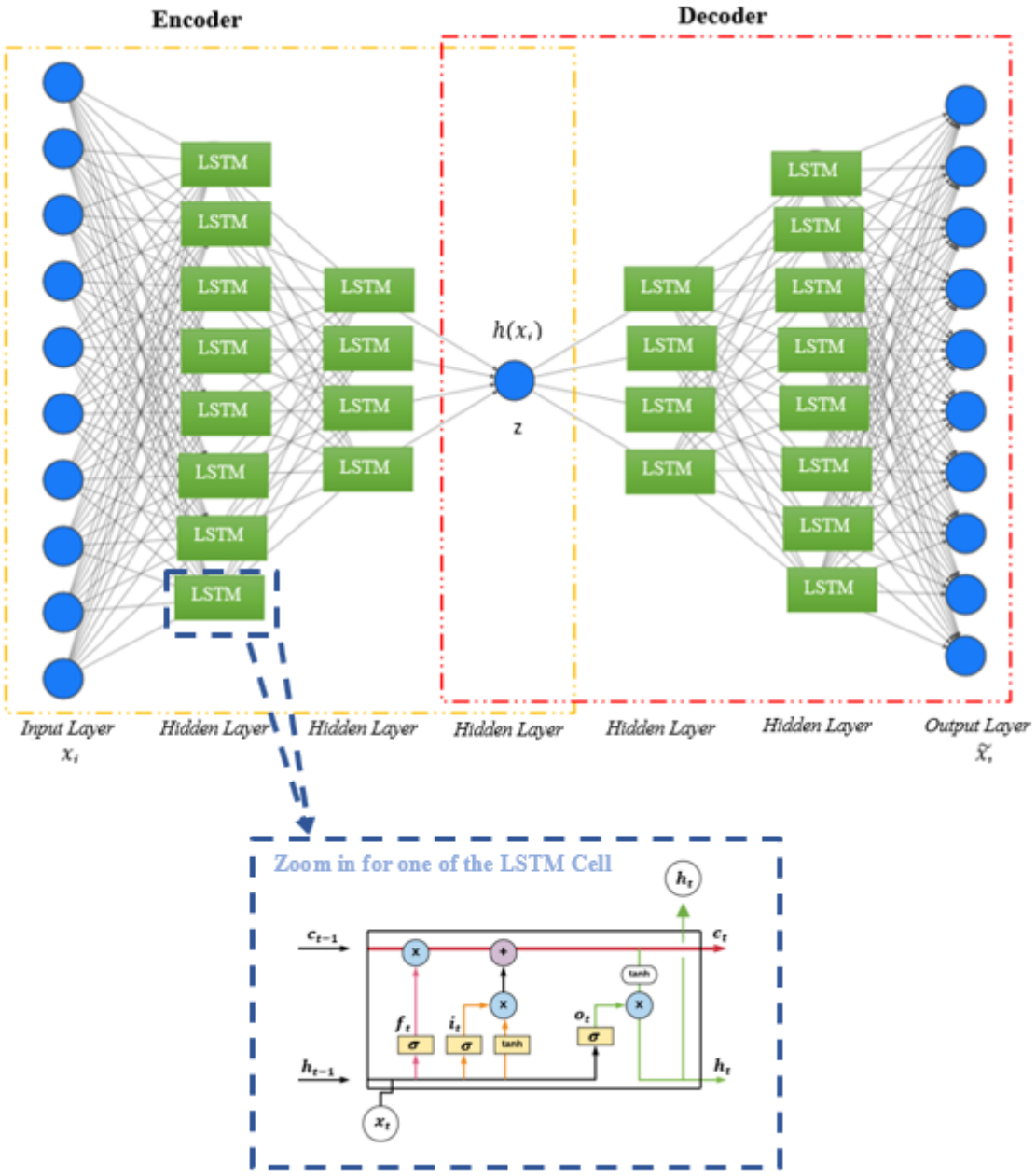


Figure 19. LSTM Autoencoder architecture.

LSTM Autoencoders [40] are divided into two phases. The first phase is LSTM Encoders, and the second phase is LSTM Decoders:

i. In LSTM encoders phase

After data is entered into the model, the model starts analyzing the data, extracting its features, reducing dimensions, and converting it into encoded data with a fixed size.

$$E : X \rightarrow Z \tag{16}$$

$$Z = \sigma(W_x + B) \tag{17}$$

ii. In LSTM Decoders phase

It takes the composed data and reconstructs it to produce data similar to the original ones. Consequently, the model becomes able to understand the nature of IoT data and can easily distinguish malware.

$$D : Z \rightarrow \widetilde{X} \tag{18}$$

$$\widetilde{X} = \widetilde{\sigma}\left(\widetilde{W_z} + \widetilde{B}\right) \tag{19}$$

With the continuous training of the model on the normal IoT data, the effectiveness and detection accuracy of the LSTM Autoencoders model are improving by taking the differences between original data and reconstructed data, until the differences between them decrease, thereby improving the model's understanding of the distinctive characteristics of the IoT data. Then, when any strange or abnormal data passes across the RoMADS MDS, the difference between the original and reconstructed data will become large and exceed the selected threshold. Thus, the model will detect malware and distinguish them from normal data.

$$Error = D(E(X)) - X \tag{20}$$

$$Anomaly\ Score = Reconstruction\ Error \tag{21}$$

In the RoMADS proposed model, the extract features of data and dimensions' reduction will not affect the accuracy, as the RoMADS model uses the Autoencoder technique, where it encodes the data, reconstructs it, then calculates the accuracy through the differences between the original data and the data that the model reconstructed. If the error rate is low, this means that the data is normal and the RoMADS model has become accustomed to it. If the error rate is greater than the threshold, this means that the entered data is strange and that the model is not familiar with it, so it will be classified as malware.

The accuracy is calculated through the error rate, i.e., the difference between the original data and the reconstructed data, and thus extracting the features of data and reducing dimensions does not affect or violate the accuracy of the model.

The main principal function of the autoencoder algorithm depends on encoding and decoding. This algorithm analyzes, understands the data, and extracts the features, then uses these features to reconstruct the data. If this algorithm is more trained on the data, the difference between the original data and the data that was reconstructed using the extracted features becomes small, which means when the difference is large, there are strange data that the model is not familiar with it. Also, we used LSTM because the LSTM network has the ability to memorize data for a long time. So, complaints between the LSTM networks and the AutoEncoder algorithm give us a strong model that has the ability to predict, identify malware, and memorize the features of IoT data for a long time.

6. Performance Criteria

RoMADS framework is evaluated based on the following:

a. Model *Accuracy*

It is used to measure to what extent the ML model can correctly predict, after training and testing the model. It is also used to evaluate the classification model. It divides the

correct number of predictions on all prediction numbers [41]. The main goal of the thesis is to get a higher accuracy for the proposed model.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{22}$$

which are:

- True Positive (*TP*),
- True Negative (*TN*),
- False Positive (*FP*),
- False Negative (*FN*).

b. *Detection Rate* of Malware

It is a true positive rate. The presence of malware is discovered by measuring reconstruction error. If the reconstruction error is higher than the threshold, this means that these entries are abnormal and that the model has not seen them before; it is malware then [42].

$$Anomaly\ Score = Reconstruction\ Error \tag{23}$$

$$Detection\ rate = \frac{TP}{TP + FN} \tag{24}$$

c. *Recall*

It is a true positive rate or sensitivity, which is used to calculate the correct positive prediction divided by the number of all the positive predictions that the model could predict. It is used to measure the model's ability to recognize the positive class [43].

$$Recall = \frac{TP}{TP + FN} \tag{25}$$

d. *Precision*

It is a Positive Predictive Value (PPV). It is used to calculate the correct positive prediction divided by all positive predictions that the model predicted. It is used to measure the validity of the prediction made by the model [43].

$$PPR = \frac{TP}{TP + FP} \tag{26}$$

e. *F1-Score*

It is used to get one score to evaluate the complete performance of the model by combining all the features of *Recall* and *Precision*. It measures the weighted average of *Precision* and *Recall* [43].

$$F1 - Score = \frac{2 * (Recall * Precision)}{(Recall + Precision)} \tag{27}$$

f. Error Rate

It is used to judge and evaluate the quality of the proposed model. Also, it judges the ability of the model to predict outputs that are very similar to actual values, lower error rate, and improve model performance. It has several types. Some of them are listed below as follows [44,45]:

- *MAE*

Mean Absolute Error (*MAE*), measures the absolute value of the residue by subtraction, by subtracting the predicted value from the measured value for each data point. Then, it calculates the average of all absolute value of residuals. The lower the *MAE* is, the

more accurate the model becomes. Its effectiveness is usually based on the proportional distribution of residual on the total error [44,45].

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}| \tag{28}$$

- *MSE*

Mean Squared Error (*MSE*) measures the square value of the residual by subtraction, by subtracting the predicted value from the measured value for every data point. Then, it calculates the average of all square values of residuals. The lower the *MSE* is, the more accurate the model becomes. Its effectiveness is usually based on detecting outliers in the data [44].

$$MSE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|^2 \tag{29}$$

- *RMSE*

Root Mean Squared Error (*RMSE*) used to measure the square root of the average of the residuals. Residuals are calculated by taking the square of the differences between the predicted value and the measured value for every data point. Then, it calculates the average of all the values of residuals and takes its square root. The lower the *RMSE* is, the more accurate the model becomes. Its effectiveness is usually when the objective is to eliminate large-scale errors [45].

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|^2} \tag{30}$$

7. Smart City Scenarios

Smart cities are regarded as a promising future for humankind. These cities are environmentally friendly, and aims to improve the quality of life by relying on information and communications technology. RoMADS framework can be adapted to be applied in smart cities, which will contribute to increased security in IoT environments and smart cities.

In the following, two scenarios about using RoMADS framework in smart cities are prepared.

A. First Scenario: Secured RoMADS Network

A company has several branches distributed in many cities, one of its branches is in the smart city A, and a second one is in the traditional city B. The company's branch B has sent an email to the company's branch A. When this email reaches the IoT network of the company's branch in city A, the package of data cannot enter the IoT network, it must pass through the intelligent RoMADS gateway.

The intelligent RoMADS gateway contains RoMADS MDS that checks the package and ensures that it is safe and free of malware. Then, it checks the protocols of the package and replaces them with protocols that suit the IoT systems, like changing TCP to UDP, as well as changing HTTP to CoAP. It allows the package, after that, to cross into the IoT network of branch A of the company.

Algorithm 1 explains the steps in detail that are taken in RoMADS framework when a package arrives from outside the IoT network, to get a secure IoT network and ensure that no malware can enter the IoT network.

Algorithm 1: Data come from external networks to IoT network.

Input: $X_i = \{X_1,\ X_2,\ X_3,\ X_4,\ \ldots, X_i\}$
Output: Malware or Normal
-Data X enter to Intelligent IoT gateway
{
-X enter to MDS
{
-LSTM Encoder
{
-Features extraction,
-Reduce the dimensions,
-Fixed length
}

-Get latent space representation Z

-LSTM Encoder
{
-Reconstruction of the data $\widetilde{X}$
}
-Calculation of the reconstruction error $S = \widetilde{X} - X$

-If S > our set threshold

-This is Malware
{
-Issuing the alarm until it is prevented from entering.
-Take benefit from the features of malware that are targeted IoT in research and studies.
}
-Else
-Normal
{
-Are the protocols compatible with IoT systems?
{
-Yes
{
-Cross the intelligent IoT gateway and enter the IoT network.
}
-No
{
-Convert its protocols into IoT protocols.
{
-Cross the intelligent IoT gateway and enter the IoT network.
}
} }
}
}
}

B. Second Scenario: Secured RoMADS Device

A smartphone device sends a signal to the door lock to open the lock; if the door lock is integrated with the security layer and there is RoMADS MDS inside it, it analyzes the signal to ensure that there is no malware within that signal. Otherwise, if the device lacks the protection layer, it connects to IoT SAP, which contains RoMADS MDS. The IoT SAP analyzes the signal and makes sure that it is a real signal and not a counterfeit by malware. If it is a real signal, the IoT SAP sends the signal to the door lock to open.

The Algorithm 2 represents the relationship between devices on the same IoT network. No device in the network receives anything before it is tested by RoMADS MDS; either RoMADS MDS has to be integrated into the same device, or if the device lacks a protection layer it communicates with an IoT SAP that contains RoMADS MDS. After RoMADS MDS confirms that the sent object is not malicious software, it sends a signal to the device that the sent object is safe and can be received.

Algorithm 2: Data come from another IoT device in same IoT network.

Input: $X_i = \{X_1, X_2, X_3, X_4, \ldots, X_i\}$
Output: Malware or Normal
-Is the device built with a security layer that contains the malware detection system?
{
-Yes
{
- Continue,
}

-No
{
- Connect device to IoT SAP,
- Continue,
}

-X enter to MDS
{
-LSTM Encoder
{
-Features extraction,
-Reduce the dimensions,
-Fixed length
}

-Get latent space representation Z
-LSTM Encoder
{
-Reconstruction of the data $\widetilde{X}$
}
-Calculation of the reconstruction error $S = \widetilde{X} - X$

-If S > our set threshold
-This is Malware
{
-Issuing the alarm until it is prevented from entering.
-Take benefit from the features of malware that are targeted IoT in research and studies.
}
-Else
-Normal
{
-Connect to other IoT device.
}
}
}

This section presents the proposed solution to improve the security in IoT environments; it is a robust framework for MDS based on LSTM Autoencoder. It uses the DL techniques to build RoMADS model for malware detecting, because it has the ability to

train models and improve the detection rate, and it is also appropriate for the nature of big data in IoT environments.

RoMADS framework distributes the RoMADS MDS in IoT system by embedding it in intelligent RoMADS gateways, IoT SAP, and IoT devices, to detect and monitor the malware that targets IoT system, and prevent it from causing any problem to the IoT system.

8. Experiment Results and Discussion

This section presents the simulation assumptions and scenario, and explains the procedures to evaluate the effectiveness of RoMADS model, and evaluates the ability of the RoMADS model to predict the malware based on the *accuracy*, *recall* & *detection rate* (%), *F*1-*score*, *precision*, *MAE*, *MSE*, and *RMSE*. Finally, it presents the comparison between the results of the proposed framework with the results of eighteen published scientific papers based on MDS.

RoMADS framework is considered a solution to improve the security in IoT environments. This framework analyzes the data traffic to detect malware at an early stage and prevent it from damaging the system and violating user privacy. The RoMADS framework is concerned with securing networks in the IoT system, by placing intelligent RoMADS MADS gateways that examine data before entering the system, as well as converting the traditional protocols to suit IoT environments. RoMADS framework is also concerned with securing the devices in the networks, whether preventive security measures are integrated into the device itself from the factory or by connecting it to IoT SAP, in order to ensure the security of lightweight devices that do not contain preventive security measures.

8.1. Simulation Assumptions

The IoT environment consists of heterogeneous devices that communicate with each other to build the IoT networks. When one of these devices is compromised, all devices on the same network will be at risk. Also, it is difficult to know and guess the different signatures of malware because it constantly renews itself. This means that the risk of not detecting malware at an early stage has increased, which may lead to blowing the system.

For that reason, a robust framework for MDS is proposed. It is trained to understand the nature of the normal IoT data traffic in its area. When anything strange is detected, the system issues an early alert to take the necessary action. The proposed solution is building a system to detect malware by relying on some DL techniques, which are LSTM and Autoencoder. This system has a great potential to provide a high degree of security in IoT environments, thus increasing users' confidence and dependence on the IoT.

8.2. Simulation Scenario

When a person deals with a supermarket for the first time, he/she needs to install the application of the supermarket on his mobile phone and create his/her personal profile. Then, the application of the supermarket connects to the smart refrigerator to see which items are out of stock. The smart refrigerator displays a list of all the items that need to be purchased and asks the user to approve them to complete the order. The user chooses the products he wants and send the request. Then the application on the user's mobile phone creates the order and sends it to the supermarket, which in turn sends the invoice and asks the user for payment. The user checks the issued invoice, and once he/she approves it, the order's cost will be deducted from his credit card, and the user receives a confirmation notification that his/her order has been successfully placed.

- ***Communication Objects:*** It is a term referring to the communication between the smartphone, the smart refrigerator, the smart supermarket system, as well as the bank to deduct the amount.
- ***Smartphone:*** RoMADS MDS is integrated on the same device because it is considered a smart device with high capabilities. The Smartphone may become connected to the same network with the smart refrigerator or any other network.

- ***Smart Refrigerator:*** RoMADS MDS is not integrated in the same device because of the smart refrigerator's lack of security mechanisms. So, it connects to the IoT SAP for security mechanisms. It may become connected to the same network with the smartphone or any other network.
- ***Smart Supermarket System:*** The smart supermarket has its own IoT network, which contains smart RoMADS gateways with RoMADS MDS built into it. So, anything that has already been examined and checked is allowed to enter the IoT network or communicate with it.
- ***The Bank and Credit Card Information:*** The bank has its own IoT network, which contains smart RoMADS gateways with RoMADS MDS built into it. So, anything that has been already examined and checked is allowed to enter the IoT network or communicate with it. All users' information must be fully encrypted and connected with a strong authentication software to ensure the user's identity, and to make sure that they will not be breached or violated. This kind of network is classified among the highest level of security in our framework.

8.3. Case Study

When the communication between a smartphone and a smart refrigerator occurs for the first time, based on the concept of the IoT social architecture that RoMADS framework depends on, it will need additional precautionary security measures to ensure the integrity of the device and that it is real and not a malware. But in case there is a previous interaction or a knowledge relationship between the two objects, depending on the principle of the IoT social architecture that RoMADS framework depends on, the security measures will be fewer. After that, RoMADS MDS checks the integrity of the IoT devices, then the communication between the two IoT devices is completed and the user profile is created. The smart refrigerator informs the user of everything that is missing in the refrigerator so that the user has a list of missing items, then the user can accept all the items in the list or just select some of them. After that, the smart refrigerator takes the approval of the user and communicates with the smart supermarket system. If they communicate for the first time, they need additional precautionary security measures to ensure the integrity of the connection. After that, the smart refrigerator sends the order to the smart supermarket to prepare it. The supermarket IoT network contains RoMADS MDS, which is located in the smart RoMADS gateways to ensure the security of things coming from outside the network. An invoice is sent to the user until he/she confirms it by completing the payment process. Bank information must be encrypted with a high level of security to ensure its integrity and that they are free of malware that targets the sensitive information, particularly the payment information. Finally, after the payment process is done, the supermarket sends the items to the user at the time the user had previously chosen.

Definition 1. *Data X passes through the RoMADS MDS, analyzes it, and extracts its distinctive features Z (Encoder process); after that, it reconstructs it $\widetilde{X}$ based on its distinctive properties (the Decoder process). Then, it measures the difference between the original data X and the reconstructed data $\widetilde{X}$. If the difference is small, that means it is normal data; since the model has been trained on it before, it has the ability to reconstruct it again with a high accuracy.*

$$X - \widetilde{X} = small = Normal\ Behavior \tag{31}$$

Definition 2. *Data X passes through the RoMADS MDS, analyzes it, and extracts its distinctive features Z (Encoder process); after that, it reconstructs it $\widetilde{X}$ based on its distinctive properties (Decoder process). Then, it measures the difference between the original data X and the reconstructed data $\widetilde{X}$. If the difference is large, that means it is an abnormal data; it is malware. Since the model has not been trained on it before, it does not have the ability to reconstruct it again with high accuracy.*

$$X - \widetilde{X} = large = Malware\ Behavior \tag{32}$$

Figures 20 and 21 present more clarification of the previous scenario. The numbers in Figure 21 present the sequence of steps during the communication process between objects according to the scenario that was explained.

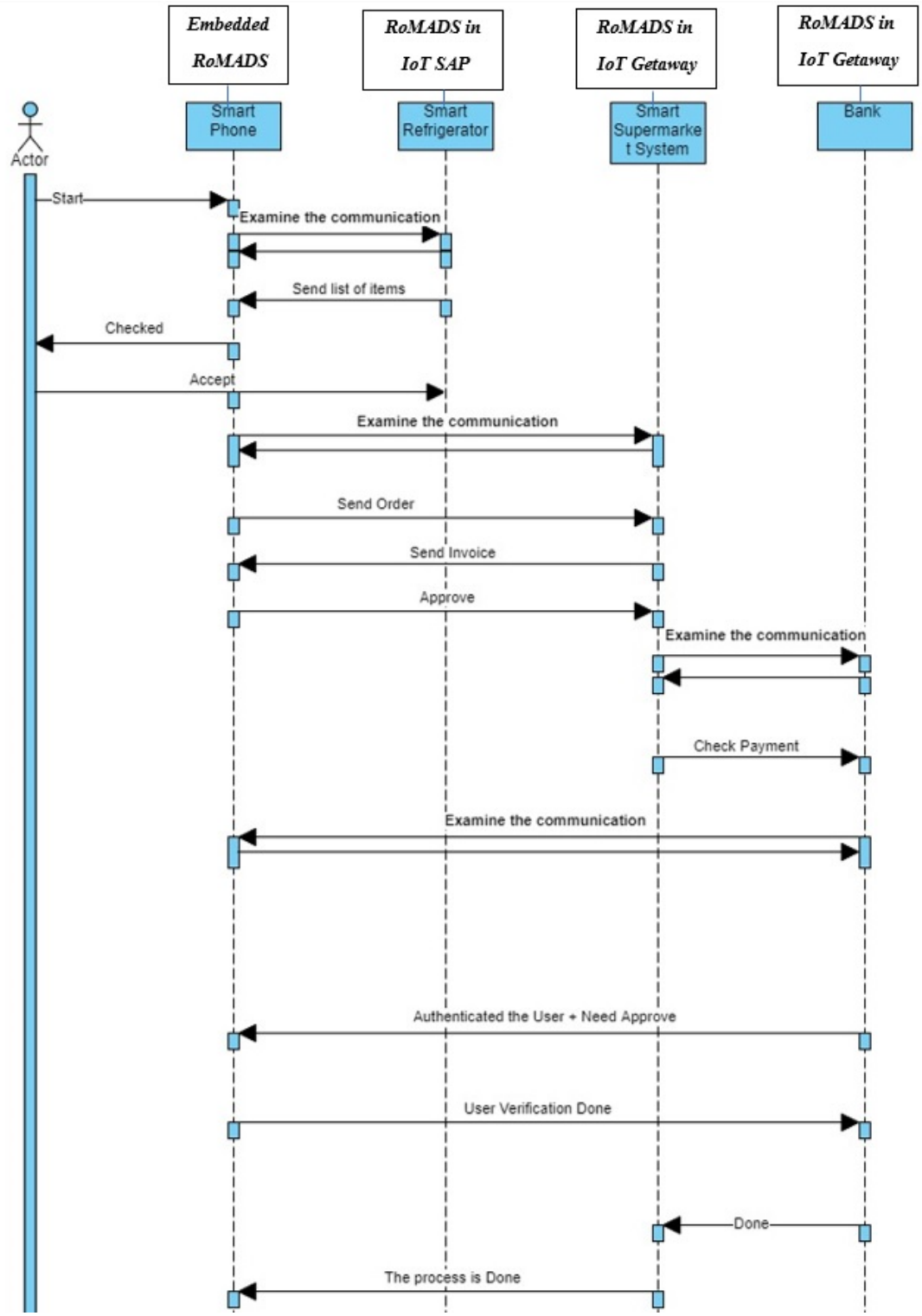


Figure 20. RoMADS sequence diagram.

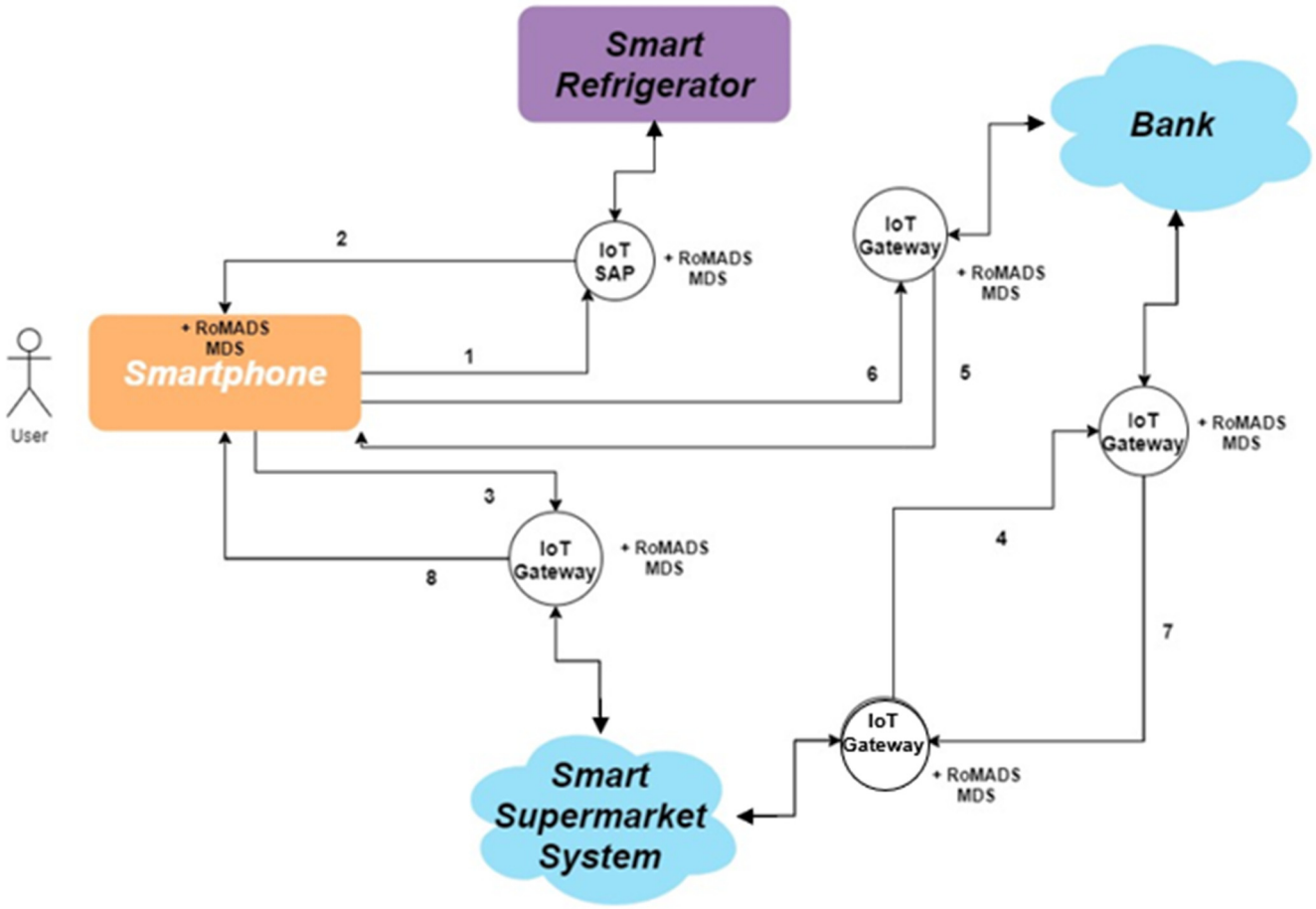


Figure 21. Objects connection diagram.

8.4. Simulation Environment

The experiment was done on Google Colaboratory [34] on the cloud using Python language and GPU. After preparing the data to be trained by LSTM Autoencoder model, the dataset is split into a training dataset, validation dataset, and test dataset, in order to have the training dataset completely isolated from the test data to obtain real results. Also, both the training dataset and the validation dataset are divided into a normal dataset without malware samples, and a test dataset contains normal samples and malware samples. The proposed model is trained using the dataset that contains normal samples without malware samples. This step helped the LSTM Autoencoder model understand the natural behaviors of the IoT system and acquire the ability to distinguish malware behaviors.

Then, a shift technique was applied to improve the model's predictability. After that, the data are reshaped and standardized by applying data scaling techniques, which made all data approximately in the same range to improve the model's predictability. Some ML algorithms focus on larger range data while training the model, which leads to an imbalance in prediction accuracy. So, applying the scaling technique contributes to solving this problem. Figure 22 presents the dataset features distribution after scaling.

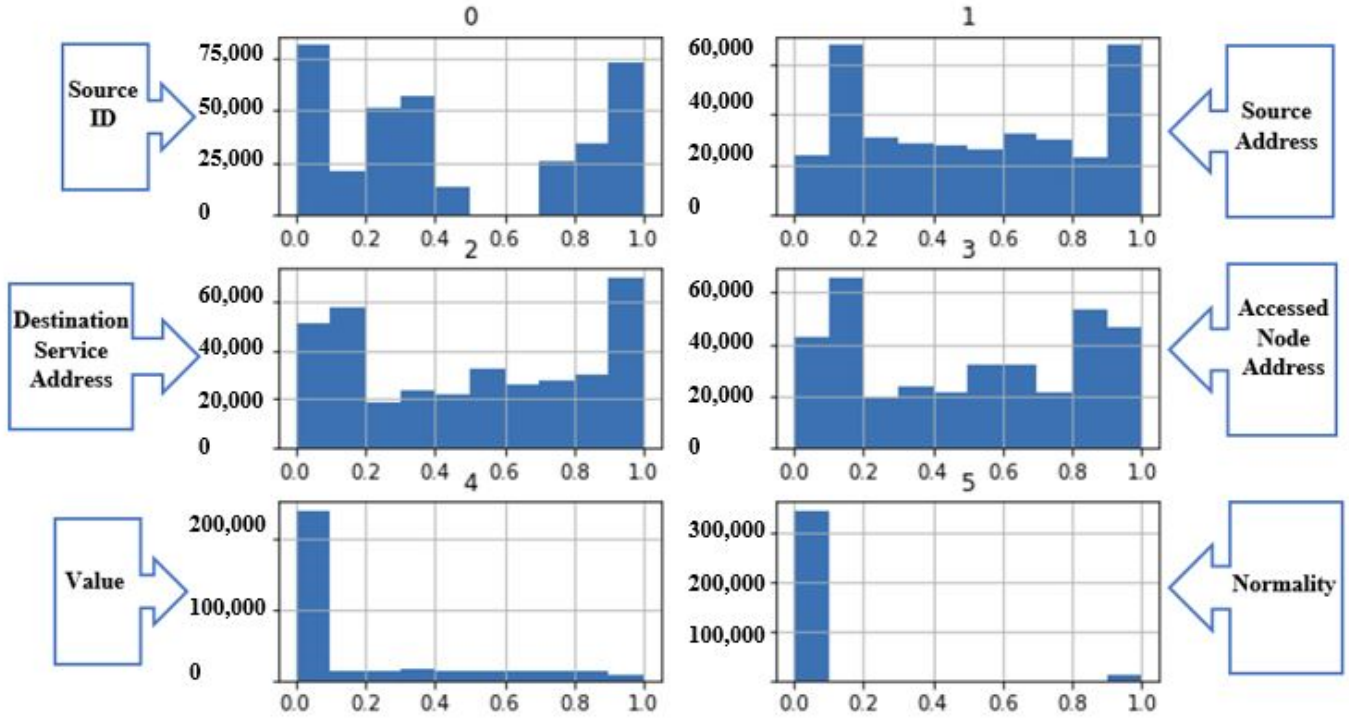


Figure 22. The dataset features distribution after scaling.

The RoMADS model for a MADS on the IoT system based on the LSTM Autoencoder is a multivariate model. Figure 23 present more details about our model. RoMADS model contains two input layers, two output layers, repeat vector layer and time distributed layer. The first input layer (LSTM (32)) took input data and outputs 32 features with 5 timesteps for each. The second layer (LSTM (16)) took input from the first layer, which is 32 × 5, then it reduced it to 16 features.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_4 (LSTM)                (None, 5, 32)             4864
_________________________________________________________________
lstm_5 (LSTM)                (None, 16)                3136
_________________________________________________________________
repeat_vector_1 (RepeatVecto (None, 5, 16)             0
_________________________________________________________________
lstm_6 (LSTM)                (None, 5, 16)             2112
_________________________________________________________________
lstm_7 (LSTM)                (None, 5, 32)             6272
_________________________________________________________________
time_distributed_1 (TimeDist (None, 5, 5)              165
=================================================================
Total params: 16,549
Trainable params: 16,549
Non-trainable params: 0
_________________________________________________________________
```

Figure 23. The details about RoMADS model during the training.

The repeat vector layer served as a bridge between the input layers (encoder process) and output layers (decoder process). Output layers are the same as the input layers, but in

a reverse way. Finally, the last layer is the time distributed layer. The input of the LSTM Autoencoder is a 3D array, a 2D array for LSTM, and 1D array for time steps. The 3D array contains (samples, time steps, features). The batch size set as 64, and the model was trained for 400 epochs. The model is classified based on the LSTM Autoencoder reconstruction error. If the number of errors is more than the threshold, this means there is an abnormal behavior. To determine the threshold, the precision-recall curve was used, which calculated the precision and recall for each threshold, then it shows them on the curve, and it determines the best value of the threshold, which is 1.2 in this experiment. Figure 24 presents the model loss during the training and validation time. The threshold value can be chosen according to the percentage of security that we need to secure the assets, so that whenever the assets are more important, the threshold value is reduced. Ideally, a value should be chosen to strike a balance between security and usability.

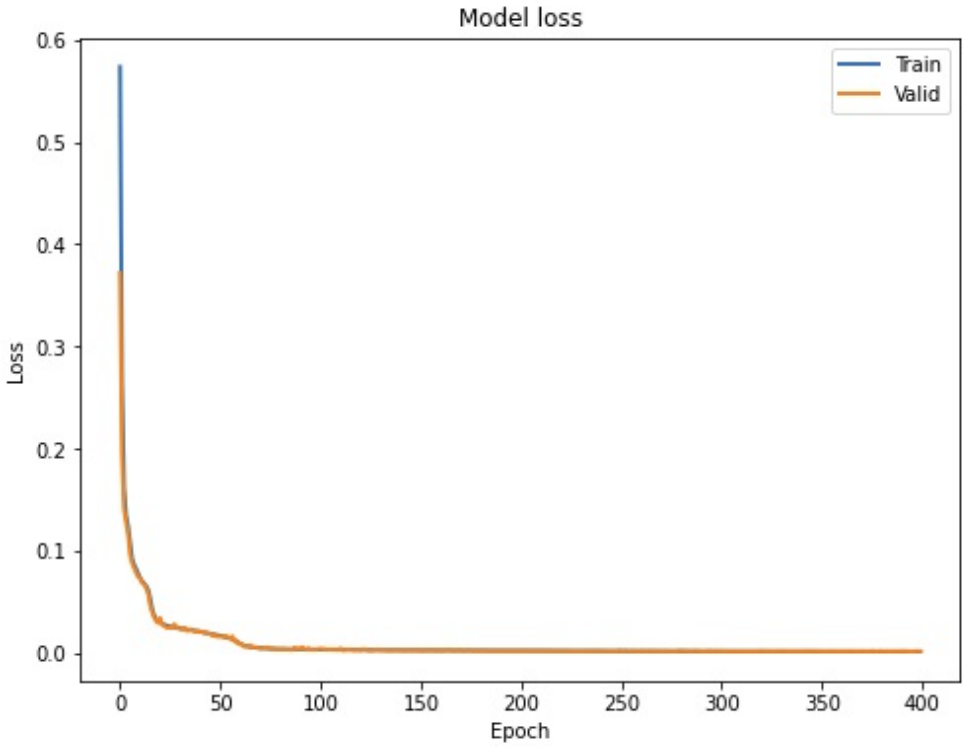


Figure 24. Model loss during the training and validation time.

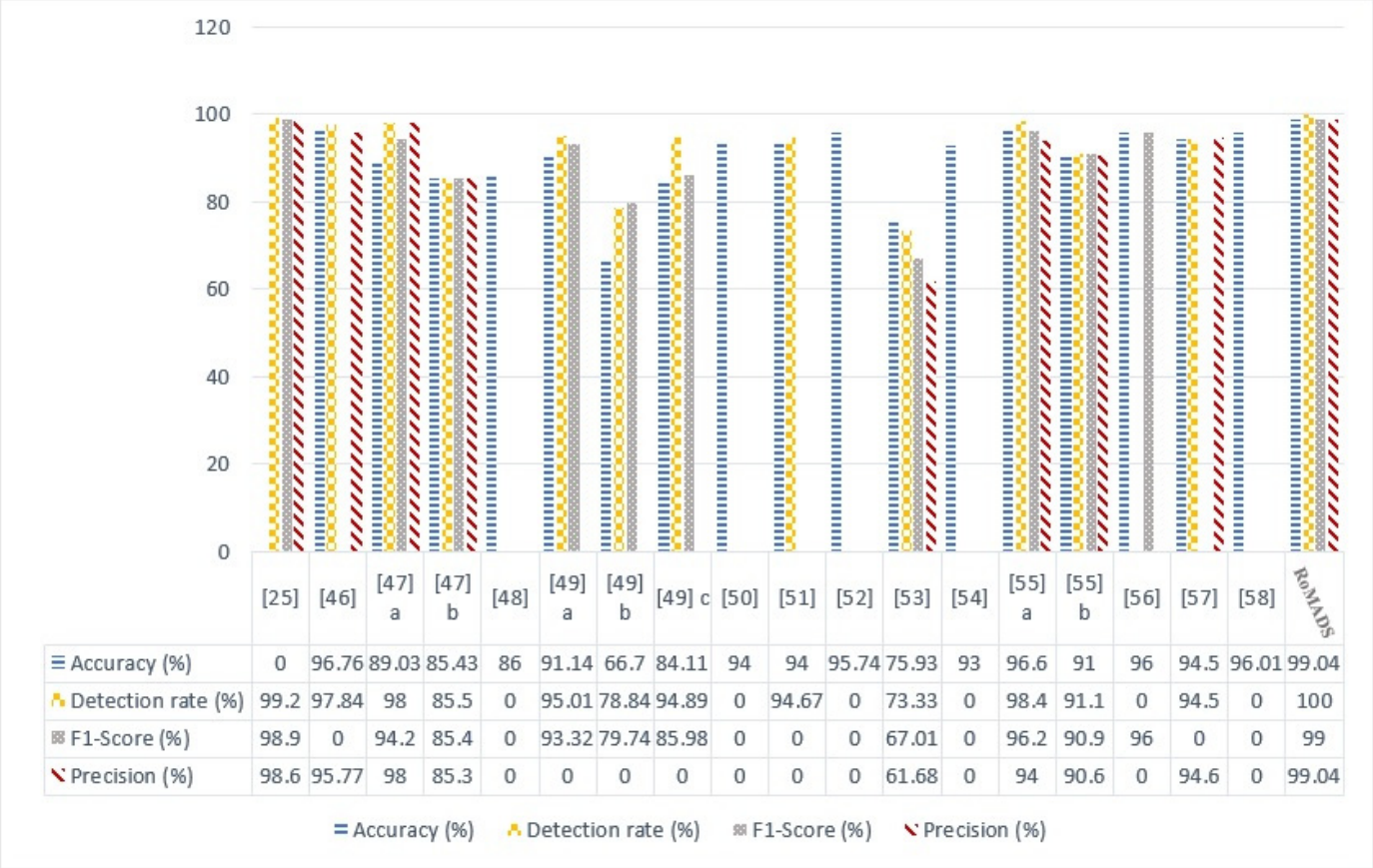
8.5. Result and Discussion

This section presents the experiment results and the effectiveness of the RoMADS model. Also, it presents a comparison between its results and the previous works in this area. RoMADS framework is a robust framework for MADS based on LSTM Autoencoder DL techniques. To prove the robustness of RoMADS model and how effective it is in detecting malware, the RoMADS model is evaluated based on many parameters: the *accuracy*, *recall* & *Detection rate*, *F1-Score*, *precision*, *MAE*, *MSE*, and *RMSE*.

Then, the RoMADS model results are compared with the previous eighteen models results from published scientific papers. RoMADS model for malware detection surpassed these previous models, and its results were higher than all of them.

As we notice from Figure 25, the blue column represents the accuracy. The model in the paper [46], which is based on DBN, got 96.76 as a result, which is the highest accuracy so far with respect to the previously discussed models. Yet, the RoMADS model was able to exceed them by obtaining 99.4%. As the yellow column represents the detection rate, the best rate for detecting malware is 99.02 for the model in the paper [25], which is based on SAE-DT. The RoMADS model could exceed it by getting a detection rate of 100%, which means that RoMADS model ensures a very high security and protection ratio for IoT

environments and guarantees its non-vulnerability to malware. The RoMADS detection rate is higher than all other models in this comparison.



	[25]	[46]	[47] a	[47] b	[48]	[49] a	[49] b	[49] c	[50]	[51]	[52]	[53]	[54]	[55] a	[55] b	[56]	[57]	[58]	RoMADS
Accuracy (%)	0	96.76	89.03	85.43	86	91.14	66.7	84.11	94	94	95.74	75.93	93	96.6	91	96	94.5	96.01	99.04
Detection rate (%)	99.2	97.84	98	85.5	0	95.01	78.84	94.89	0	94.67	0	73.33	0	98.4	91.1	0	94.5	0	100
F1-Score (%)	98.9	0	94.2	85.4	0	93.32	79.74	85.98	0	0	0	67.01	0	96.2	90.9	96	0	0	99
Precision (%)	98.6	95.77	98	85.3	0	0	0	0	0	0	0	61.68	0	94	90.6	0	94.6	0	99.04

Figure 25. Comparison between RoMADS model results and the previous paper's results based on *Accuracy*, *Recall* & *Detection rate*, *F1-Score*, *Precision*.

The gray column represents F1-score and the red column indicates the precision. The model in the paper [25] shows a very high percentage for both of them in comparison to the other models, which is 98.9% for *F1-score* and 98.4 for precision. Yet, the RoMADS model got the highest percentage in both of them: 99% for *F1-Score* and 99.4 for precision. Thus, it becomes clear to us that the RoMADS model is able to improve all ratios in this comparison and gets the highest rate among all of them, and consequently, IoT environments would be safer and more secure for users and their important data.

Figure 26 shows us the comparison between the ROMAD model and three other models depending on the *MAE*. If the error rate is decreased, the model accuracy is increased. According to what we can notice in Figure 26, model [47] a got the lowest error rate, which is 0.1097, in comparison to the other two models. Again, The RoMADS model was able to obtain the lowest error rate, which is 0.009614, and thus it is obvious to us that RoMADS model overcomes all other models in this study.

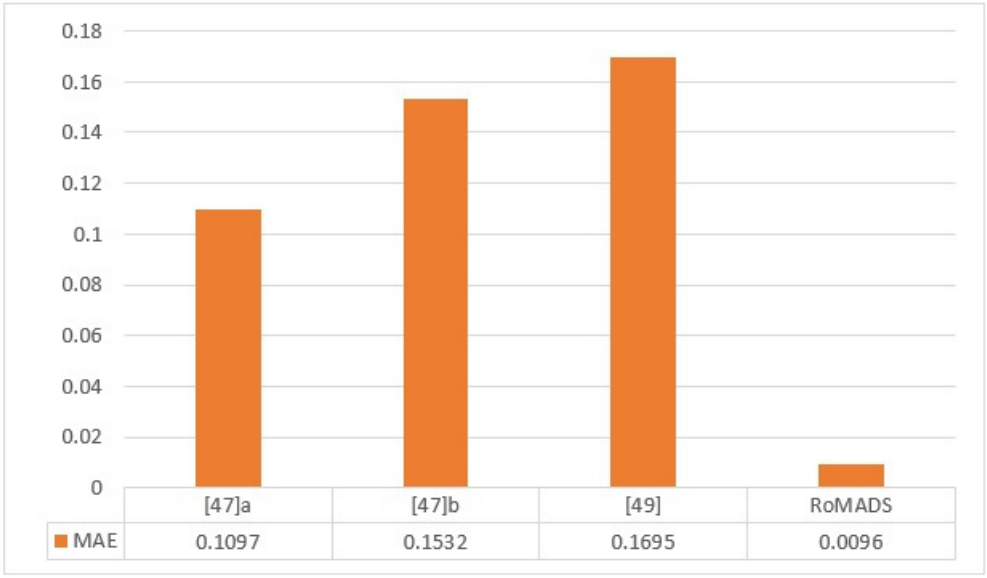


Figure 26. Comparison between RoMADS model result and the previous paper's results based on *MAE*.

Figure 27 shows the comparison study of the *RMSE* result of the RoMADS model to the four previous models that were proposed in papers [48,49]. Whereas, in the paper [49] the researcher proposed three models, which are model A that is based on Neural Network, model B that is based on Multiple Association Rule, and the third is model C, which is based on Bayesian technology. In paper [48] the researcher proposed one model based on Artificial Neural Network (ANN) and it got a higher result than the other previous models in this comparison. It reduced the *RMSE* ratio of 0.202. The RoMADS model was able to overcome this ratio by getting 0.09805, which is considered the lowest error rate, and this means that the RoMADS model provides the highest degree of security among this comparison study.

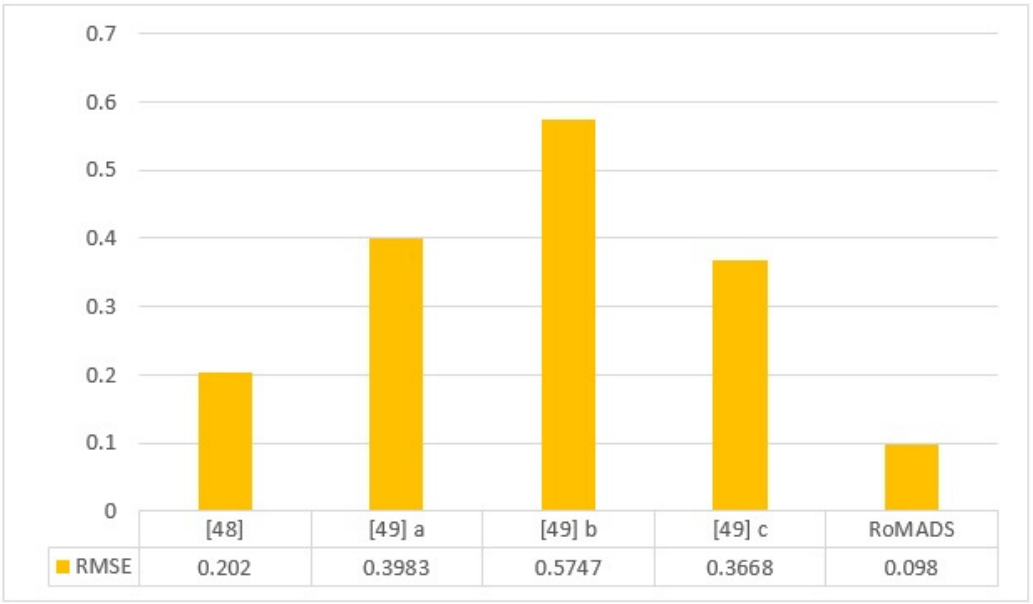


Figure 27. Comparison between RoMADS model result and the previous paper's results based on *RMSE*.

As shown in Figure 28, there is no large number of studies concerned with reducing the *MSE*. The researchers in paper [48] proposed a model that is based on ANN and had a low *MSE* error rate, which is 0.041, but the RoMADS model overcame it by obtaining a better result, which is 0.0096. If the model gets a lower error rate, that means it gets more accuracy and increases its ability to detect intelligent malware.

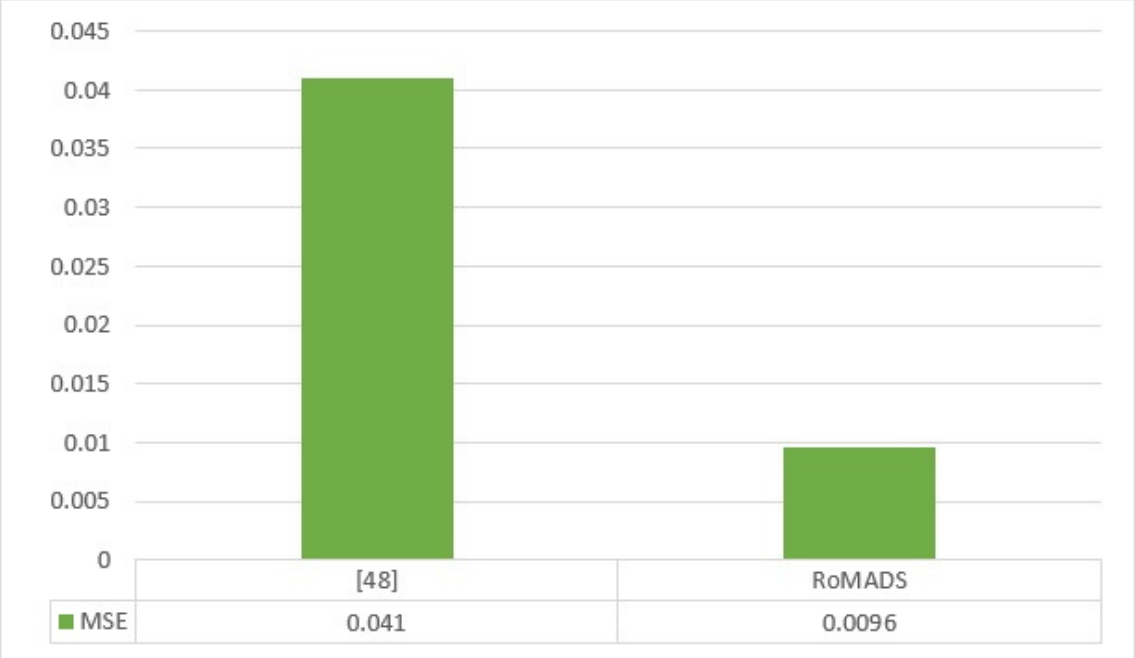


Figure 28. Comparison between RoMADS model result and the previous paper's results based on *MSE*.

Table 4 presents how the RoMADS model improved all the comparison factors, which are *accuracy*, *recall* & *detection rate*, *F1-score*, *precision*, *MAE*, *MSE*, and *RMSE*. RoMADS model obtained distinctive results and overcame all the other models, as it obtained 99.0386% for the accuracy, 99.9970 for the *detection rate*, 99.00 for *F1-score*, and 99.0414 for *precision*. These are very high ratios that exceed all the results of the previous eighteen.

The RoMADS model was also able to obtain the lowest error rate, which means that the RoMADS model has the highest protection ability, as it obtained 0.009614 for *MAE*, 0.09805 for the *RMSE*, and reduces the *MSE* to 0.0096. Thus, the RoMADS model is distinguished by its ability to improve all the seven different factors in this comparison. It exceeds all the other eighteen models in this comparison by obtaining the best results and lowest error rates, which undoubtedly makes it stronger and more reliable for use and more adaptable to the real IoT environments.

IoT environments are considered non-stationary environments because the malware is constantly getting updated by the adversary to overcome the existing MDSs. So, after a period of time the MDS performance begins deteriorating, even though it has been well trained and has a high accuracy in predicting and detecting malware. Therefore, the models need continuous maintenance from time to time to ensure high performance, but this is very expensive and costly.

In order to solve this problem and get a powerful system for detecting and predicting the malware, the shift technique is applied on the dataset. This technology can make the correct distribution of training and testing dataset. It has played an effective role in improving the classification and prediction effectiveness, especially in systems that continuously detect the updating malware—the malware that tries to overcome the system—such as in the IoT environments. Table 5 presents the results of two models that were trained; the first is with the shift and the second is without the shift. The results with the shift were better and more accurate.

Table 4. Comparison between RoMADS model result and the previous paper's results.

Ref.	Classifier	*Accuracy* (%)	*Recall* & Detection Rate (%)	*F1-Score* (%)	*Precision* (%)	Error Value		
						MAE	*MSE*	*RMSE*
[25]	SAE-DT	-	99.2	98.9	98.6	-	-	-
[46]	DBN	96.76	97.84	-	95.77	-	-	-
[47]	Random forst	89.03	98.0	94.2	98.0	0.1097	-	-
[47]	SVM	85.43	85.5	85.4	85.3	0.1532	-	-
[48]	ANN	86.0	-	-	-	-	0.041	0.202
[49]	Neural Network	91.14	95.01	93.32	-	-	-	0.3983
[49]	Multiple Association Rule	66.70	78.84	79.74	-	-	-	0.5747
[49]	Bayesian	84.11	94.89	85.98		0.1695		0.3668
[50]	RNN (LSTM)	94.0	-	-	-	-	-	-
[51]	CNN	94.0	94.67	-	-	-		
[52]	Deep autoencoder (DAE + GAN)	95.74	-	-	-	-	-	-
[53]	Neural Networks	75.93	73.33	67.01	61.68	-	-	-
[54]	ARI-LSTM	93.0	-	-	-	-	-	-
[55]	CNN	96.6	98.4	96.2	94.0	-	-	-
[55]	DNN	91.0	91.1	90.9	90.6	-	-	-
[56]	DBN—decision tree	96.0	-	96.0	-	-	-	-
[57]	CNN	94.5	94.5	-	94.6	-	-	-
[58]	RNN	96.01	-	-	-	-	-	-
The proposed model RoMADS	AE-LSTM	99.0385720855369	99.99709795841374	99.00	99.0414187577247	0.00961427914463095	0.00961427914463095	0.09805243059012331

Table 5. RoMADS model results with the shift and without the shift.

	With Shifting	**Without Shifting**
Epoch	200	200
Timesteps	5	5
Batch size	64	64
LSTM Layer 1	(None, 5, 32)	(None, 5, 32)
LSTM Layer 2	(None, 16)	(None, 16)
LSTM Layer 3	(None, 5, 16)	(None, 5, 16)
LSTM Layer 4	(None, 5, 32)	(None, 5, 32)
ACC (%)	99.0385720855369	97.22854708257086
TPR (%)	99.99709795841374	1.0
PPV (%)	99.0414187577247	97.2285083675579
Recall (%)	99.0	97.0
F1-score (%)	99.0	96.0
MAE	0.00961427914463095	0.027714529174291423
MSE	0.00961427914463095	0.027714529174291423
RMSE	0.09805243059012331	0.16647681272264742

The experiment is repeated several times with some changes in the number of epochs, batch-size, or time steps. Table 6 presents the results for different training modes.

Table 6. RoMADS results for different training modes.

	1	**2**	**3**	**4**
Epoch	200	400	90	1
Timesteps	5	5	5	1
Batch Size	64	64	64	32
LSTM Layer 1	(None, 5, 32)	(None, 5, 32)	(None, 5, 64)	(None, 5, 32)
LSTM Layer 2	(None, 16)	(None, 16)	(None, 32)	(None, 16)
LSTM Layer 3	(None, 5, 16)	(None, 5, 16)	(None, 5, 32)	(None, 5, 16)
LSTM Layer 4	(None, 5, 32)	(None, 5, 32)	(None, 5, 64)	(None, 5, 32)
ACC	99.0385720855369	99.0385720855369	99.0385720855369	99.0385720855369
TPR	99.99709795841374	99.99709795841374	99.99709795841374	99.99709795841374
Precision	99.0414187577247	99.0414187577247	99.0414187577247	99.0414187577247
Recall	99.00	99.00	99.00	99.00
F1-Score	99.00	99.00	99.00	99.00
MAE	0.00961427914463095	0.00958553690503564	0.009599908024833294	0.009700366458288425
MSE	0.00961427914463095	0.00958553690503564	0.009599908024833294	0.009700366458288425
RMSE	0.09805243059012331	0.09790575521916799	0.09797912035139576	0.09849043841047934

In this experiment, As the number of epochs was increasing, the RoMADS model has reduced the validation and training losses, that means no overfitting in RoMADS model during our simulation. This means that the RoMADS model has the ability to understand the data and conclude, and thus it can be more reliable.

Previous results present how effective is the RoMADS model in detecting the malware in IoT environments and overcomes the previous research papers' results. This helps in obtaining safe, secure IoT environments, free from malware threats.

The RoMADS model is adaptable, scalable, and evolvable to obtain better results when applied in a real IoT environment. It contributes to improving security, safety, and protecting IoT environments from malicious software that may cause major disasters, threaten human lives, and violate their privacy. Consequently, overcoming this malicious software has a great and effective role in spreading safety, security, and increasing human confidence to rely on this modern technology that provides a new, more comfortable, and happy life for humanity. The RoMADS model has a promising future that contributes to solving many security and privacy issues in IoT environments.

9. Conclusions

This research paper studies how we can utilize techniques of DL to improve MADS in IoT systems and maintain security and privacy, in order to overcome malware and the harm it causes to the IoT environments. It presents the background and related theories. Also, it discusses and presents the state-of-the-art papers that related to the study. Until now, there are many vulnerabilities in IoT based MDS systems security, where we need to improve the *accuracy*, *recall* and *detection rate*, *F*1-*Score*, *precision*, and reduce the error rate (*MAE*, *MSE*, *RMSE*).

After that, it presents the methodology that we followed in conducting this research, the proposed solution, performance criteria, scenarios, and algorithms. The proposed solution is a robust framework for the MADS based on LSTM Autoencoder DL technique. It's called RoMADS. Then, it explains how is the RoMADS model is evaluated, based on a number of parameters: the *accuracy*, *recall* & *detection rate*, *F*1-*Score*, *precision*, *MAE*, *MSE*, and *RMSE*. It presents the comparison between RoMADS model results with the previous eighteen models' results from published scientific papers, which the RoMADS model overcame. The accuracy of the RoMADS model is 99.038%. RoMADS model results are considered strong results. It can contribute to more secure and reliable IoT systems without any fears or concerns about security and privacy. The RoMADS model is also scalable and adaptable to real IoT environments, improving security, preserving human privacy, and gaining more humans' confidence. With respect to the results obtained after conducting the experiment, the proposed RoMADS model results successfully achieved the research objectives.

This research paper has added many benefits and values to the IoT environment security improving field. Among these gained values we may list the following:

(1) It presents the state-of-the-art research and studies related to our study.
(2) Proposed a solution based on DL techniques which is RoMADS framework, in order to improve the security and protection from malware.
(3) Presented RoMADS framework architecture that combines fog computing architecture with IoT social architecture. The fog computing architecture gives RoMADS framework the concept of distrusting the load on a number of smart gateways and smart sensors, which contributes to improving the time consuming aspect. The IoT social architecture works in a similar manner to the social relations between humans and, consequently, puts preventive security measures accordingly. This helps in improving the performance of real-time systems.

(4) To reduce the energy and time consumed, RoMADS framework checks and converts the traditional protocols to IoT protocols for all the packages coming to the IoT system.
(5) RoMADS framework has the ability to deal with the sequential of data with different lengths. It works to reduce the dimensions, remove the noise, and extract the main features.
(6) RoMADS framework has the ability to remember and store the important features of data for a long time, which contributes to obtaining an intelligent model that has a high capacity of detection, understanding, and analysis.
(7) RoMADS framework helps to exploit the devices currently available on markets by adapting and exploiting them in IoT environments. This helps to speed up the transition to smart IoT environments, which provides a comfortable and healthy life for humanity.
(8) RoMADS model results were as follows: 99.03% for *Accuracy*, 99.99% for *Recall* & *Detection rate*, 99.00% for *F*1-*Score*, 99.04% for *Precision*, 0.0096% for *MAE*, 0.0096 for *MSE*, and 0.0980 for *RMSE*.
(9) The RoMADS model experiment results surpassed eighteen models of previous research works related to this field. RoMADS framework is scalable, evolvable, and adaptable when it is applied in different real IoT environments.

Depending on the previous findings, results, and features, we expect that the RoMADS framework model will have an effective role in improving security of the IoT environments, increasing people's confidence, as well as preserving their privacy.

This paper focuses on obtaining a high level of security and safety. In the coming research, we will have to study more in depth the false alarms to achieve a balance between usability and security, as well as implementing the RoMADS, which is a robust framework for MADS based on LSTM Autoencoder DL technique in a real IoT environment, and test how effective it is.

The major obstacle that this research faced was in finding a real IoT environment available for conducting experiments and testing the RoMADS framework, along with the high cost of the hardware and tools needed to build an IoT environment.

Author Contributions: Conceptualization, H.T.; Data curation, H.T.; Formal analysis, H.T.; Funding acquisition, H.T.; Investigation, H.T.; Methodology, H.T.; Project administration, H.T.; Resources, H.T.; Software, H.T.; Supervision, H.T. and R.Z.; Validation, H.T.; Visualization, H.T.; Writing—original draft, H.T.; Writing—review & editing, H.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received funding from SAUDI ARAMCO Cybersecurity chair.

Data Availability Statement: To perform simulations in this research, we used the previously mentioned [DS2OS] data set [27] which are available on: https://www.kaggle.com/francoisxa/ds2ostraffictraces/ (accessed on 20 January 2020).

Acknowledgments: We would like to thank SAUDI ARAMCO Cybersecurity Chair for funding this project.

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

Abbreviations

The following abbreviations are used in this manuscript:

IoT	Internet of Things
Ref.	References
MDS	Malware Detection System
ADS	Anomaly Detection System
ML	Machine Learning
ANN	Artificial Neural Network
CNN	Convolutional Neural Network

Ars	Association Rules
DL	Deep Learning
DBN	Deep Belief Network
DNN	Deep Neural Network
LSTM	Long Short-Term Memory
GAN	Generative Adversarial Network
KNN	K-Nearest Neighbor
NB	Naive Bayes
PCA	Principal Component Analysis
SAEs	Neural Network-Stacked AutoEncoders
RNN	Recurrent Neural Network
SVMs	Support Vector Machines
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MSE	Mean Square Error
RMSE	Root Mean Square Error
DT	Decision Tree

References

1. Talal, H.; Zagrouba, R. MADS Based on DL Techniques on the Internet of Things (IoT): Survey. *Electronics* **2021**, *10*, 2598. [CrossRef]
2. Thalesgroup. IoT Security Issues in 2021: A Business Perspective. 2021. Available online: https://www.thalesgroup.com/en/markets/digital-identity-and-security/iot/magazine/internet-threats (accessed on 16 August 2021).
3. Balogh, S.; Gallo, O.; Ploszek, R.; Špaček, P.; Zajac, P. IoT Security Challenges: Cloud and Blockchain, Postquantum Cryptography, and Evolutionary Techniques. *Electronics* **2021**, *10*, 2647. [CrossRef]
4. Kaspersky IoT Under Fire: Kaspersky Detects More Than 100 Million Attacks on Smart Devices in H1 2019. Available online: https://www.kaspersky.com/about/press-releases/2019_iot-under-fire-kaspersky-detects-more-than-100-million-attacks-on-smart-devices-in-h1-2019 (accessed on 8 June 2021).
5. Cisco What Is Malware? 2020. Available online: https://www.cisco.com/c/en/us/products/security/advanced-malware-protection/what-is-malware.html (accessed on 20 June 2021).
6. Rouse, M. What Is Malware? 2019. Available online: https://searchsecurity.techtarget.com/definition/malware (accessed on 18 June 2021).
7. Goodfelow, I.; Bengio, Y.; Courville, A. Autoencoders. In *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016; pp. 499–523.
8. Ciha, T. PCA Autoencoders: Algorithms Everyone Can Understand. Towards Data Science. 2018. Available online: https://towardsdatascience.com/understanding-pca-autoencoders-algorithms-everyone-can-understand-28ee89b570e2 (accessed on 20 August 2021).
9. Badr, W. Auto-Encoder: What Is It? And What Is It Used for? (Part 1). Towards Data Science. 2019. Available online: https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726 (accessed on 20 June 2021).
10. Dertat, A. Applied Deep Learning—Part 3: Autoencoders. Towards Data Science. 2017. Available online: https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798 (accessed on 21 June 2021).
11. Luo, T.; Nagarajan, S.G. Distributed Anomaly Detection Using Autoencoder Neural Networks in WSN for IoT. In Proceedings of the IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 2018; pp. 1–6. [CrossRef]
12. Stewart, M. Comprehensive Introduction to Autoencoders. Towards Data Science. 2019. Available online: https://towardsdatascience.com/generating-images-with-autoencoders-77fd3a8dd368 (accessed on 25 August 2021).
13. Missinglink. Deep Learning Long Short-Term Memory (LSTM) Networks: What You Should Remember. Available online: https://missinglink.ai/guides/neural-network-concepts/deep-learning-long-short-term-memory-lstm-networks-remember/ (accessed on 18 August 2021).
14. Olah, C. Understanding LSTM Networks. 2015. Available online: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (accessed on 16 August 2021).
15. Chawla, A.; Lee, B.; Jacob, P.; Fallon, S. Bidirectional LSTM Autoencoder for Sequence Based Anomaly Detection in Cyber Security. *Int. J. Simul. Syst. Sci. Technol.* **2019**, 1–6. [CrossRef]
16. Kapur, R.; Rohan, L. Recurrent Neural Networks & LSTMs. Ayearofai. 2017. Available online: https://ayearofai.com/rohan-lenny-3-recurrent-neural-networks-10300100899b (accessed on 11 August 2021).
17. Srivastava, P. Essentials of Deep Learning: Introduction to Long Short Term Memory. 2017. Available online: https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/ (accessed on 10 August 2021).
18. Korneev, E. LSTM Neural Networks for Anomaly Detection Medium. 2018. Available online: https://medium.com/datadriveninvestor/lstm-neural-networks-for-anomaly-detection-4328cb9b6e27 (accessed on 25 June 2021).

19. Larzalere, B. LSTM Autoencoder for Anomaly Detection. Towards Data Science. 2019. Available online: https://towardsdatascience.com/lstm-autoencoder-for-anomaly-detection-e1f4f2ee7ccf (accessed on 23 June 2021).
20. Vacca, J.R. *Network and System Security*; Elsevier: Amsterdam, The Netherlands, 2014.
21. Perry, J.S. Anatomy of an IoT Malware Attack IBM. 2019. Available online: https://developer.ibm.com/articles/iot-anatomy-iot-malware-attack/ (accessed on 6 July 2021).
22. Hassija, V.; Chamola, V.; Saxena, V.; Jain, D.; Goyal, P.; Sikdar, B. A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures. *IEEE Access* **2019**, *7*, 82721–82743. [CrossRef]
23. Malge, S.; Singh, P. Internet of Things IoT: Security Perspective. *Int. J. Trend Sci. Res. Dev.* **2019**, 1041–1043. [CrossRef]
24. Xiao, L.; Wan, X.; Lu, X.; Zhang, Y.; Wu, D. IoT Security Techniques Based on Machine Learning: How Do IoT Devices Use AI to Enhance Security? *IEEE Signal Process. Mag.* **2018**, *35*, 41–49. [CrossRef]
25. Xiao, F.; Lin, Z.; Sun, Y.; Ma, Y. Malware Detection Based on Deep Learning of Behavior Graphs. *Math. Probl. Eng.* **2019**, *2019*, 8195395. [CrossRef]
26. Dovom, E.M.; Azmoodeh, A.; Dehghantanha, A.; Newton, D.E.; Parizi, R.M.; Karimipour, H. Fuzzy pattern tree for edge malware detection and categorization in IoT. *J. Syst. Arch.* **2019**, *97*, 1–7. [CrossRef]
27. Cui, Z.; Du, L.; Wang, P.; Cai, X.; Zhang, W. Malicious code detection based on CNNs and multi-objective algorithm. *J. Parallel Distrib. Comput.* **2019**, *129*, 50–58. [CrossRef]
28. Ham, H.-S.; Kim, H.-H.; Kim, M.-S.; Choi, M.-J. Linear SVM-Based Android Malware Detection. *Lect. Notes Electr. Eng.* **2014**, *301*, 575–585. [CrossRef]
29. Alam, M.S.; Vuong, S.T. Random Forest Classification for Detecting Android Malware. In Proceedings of the 2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, Beijing, China, 20–23 August 2013; pp. 663–669.
30. Vinayakumar, R.; Alazab, M.; Srinivasan, S.; Pham, Q.-V.; Padannayil, S.K.; Simran, K. A Visualized Botnet Detection System Based Deep Learning for the Internet of Things Networks of Smart Cities. *IEEE Trans. Ind. Appl.* **2020**, *56*, 4436–4456. [CrossRef]
31. Aubet, F.-X. DS2OS Traffic Traces Kaggle. 2018. Available online: https://www.kaggle.com/francoisxa/ds2ostraffictraces/ (accessed on 20 July 2021).
32. Pedregosa, F. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
33. Zipporah Luna Feature Selection in Machine Learning: Correlation Matrix, Univariate Testing, RFECV Medium 2021. Available online: https://medium.com/geekculture/feature-selection-in-machine-learning-correlation-matrix-univariate-testing-rfecv-1186168fac12 (accessed on 15 September 2021).
34. Google. Google Colaboratory. 2020. Available online: https://colab.research.google.com/notebooks/welcome.ipynb (accessed on 25 July 2021).
35. Derhamy, H.; Eliasson, J.; Delsing, J.; Priller, P. A survey of commercial frameworks for the Internet of Things. In Proceedings of the 2015 IEEE 20th International Conference on Emerging Technologies & Factory Automation (ETFA 2015), Luxembourg, 8–11 September 2015; pp. 1–8. [CrossRef]
36. Atzori, L.; Iera, A.; Morabito, G.; Nitti, M. The Social Internet of Things (SIoT)—When social networks meet the Internet of Things: Concept, architecture and network characterization. *Comput. Netw.* **2012**, *56*, 3594–3608. [CrossRef]
37. Gregersen, C. A Complete Guide to IoT Protocols & Standards in 2021. 2020. Available online: https://www.nabto.com/guide-iot-protocols-standards/ (accessed on 25 August 2021).
38. Ranjan, C. LSTM Autoencoder for Extreme Rare Event Classification in Keras. Towards Data Science. 2019. Available online: https://towardsdatascience.com/lstm-autoencoder-for-extreme-rare-event-classification-in-keras-ce209a224cfb (accessed on 5 July 2021).
39. Ranjan, C.; Reddy, M.; Mustonen, M.; Paynabar, K.; Pourak, K. Dataset: Rare Event Classification in Multivariate Time Series. *arXiv* **2018**, arXiv:1809.10717.
40. Brownlee, J. A Gentle Introduction to LSTM Autoencoders. Machine Learning Mastery. 2018. Available online: https://machinelearningmastery.com/lstm-autoencoders/ (accessed on 15 July 2021).
41. Baranwal, V.M.A.; Bagwe, B.R. Machine Learning in Python: Diabetes Prediction Using Machine Learning. In *Handbook of Research on Applications and Implementations of Machine Learning Techniques*; IGI Global: Hershey, PA, USA, 2019.
42. Hartson, R.; Pyla, P. Chapter 28—Background: UX Evaluation. In *The UX Book*, 2nd ed.; Hartson, R., Pyla, P., Eds.; Morgan Kaufmann: Boston, MA, USA, 2019; pp. 601–621.
43. Yunqian, M.; Haibo, H. *Imbalanced Learning: Foundations, Algorithms, and Applications*, 1st ed.; Wiley-IEEE Press: Hoboken, NJ, USA, 2013.
44. Pascual, C. Understanding Regression Error Metrics in Python Data Quest 2018. Available online: https://www.dataquest.io/blog/understanding-regression-error-metrics/ (accessed on 20 September 2021).
45. Kampakis, S. Performance Measures: RMSE and MAE. The Data Scientist. 2020. Available online: https://thedatascientist.com/performance-measures-rmse-mae/ (accessed on 20 September 2021).
46. Yuan, Z.; Lu, Y.; Xue, Y. Droiddetector: Android malware characterization and detection using deep learning. *Tsinghua Sci. Technol.* **2016**, *21*, 114–123. [CrossRef]
47. Villanueva, J.A.; Juanatas, R.; Lacatan, L.L. Malware predictor using machine learning techniques. *Test Eng. Manag.* **2020**, *82*, 5665–5674.

48. Adamu, U.; Awan, I. Ransomware Prediction Using Supervised Learning Algorithms. In Proceedings of the 2019 7th International Conference on Future Internet of Things and Cloud (FiCloud), Istanbul, Turkey, 26–28 August 2019; pp. 57–63.
49. Adebayo, O.S.; Aziz, N.A. Improved Malware Detection Model with Apriori Association Rule and Particle Swarm Optimization. *Secur. Commun. Netw.* **2019**, *2019*, 2850932. [CrossRef]
50. HaddadPajouh, H.; Dehghantanha, A.; Khayami, R.; Choo, K.-K.R. A deep Recurrent Neural Network based approach for Internet of Things malware threat hunting. *Future Gener. Comput. Syst.* **2018**, *85*, 88–96. [CrossRef]
51. Su, J.; Vasconcellos, V.D.; Prasad, S.; Daniele, S.; Feng, Y.; Sakurai, K. Lightweight Classification of IoT Malware Based on Image Recognition. In Proceedings of the 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), Tokyo, Japan, 23–27 July 2018; Volume 2, pp. 664–669. [CrossRef]
52. Kim, J.-Y.; Bu, S.-J.; Cho, S.-B. Zero-day malware detection using transferred generative adversarial networks based on deep autoencoders. *Inf. Sci.* **2018**, *460–461*, 83–102. [CrossRef]
53. Azmoodeh, A.; Dehghantanha, A.; Conti, M.; Choo, K.-K.R. Detecting crypto-ransomware in IoT networks based on energy consumption footprint. *J. Ambient. Intell. Humaniz. Comput.* **2018**, *9*, 1141–1152. [CrossRef]
54. Agrawal, R.; Stokes, J.W.; Selvaraj, K.; Marinescu, M. Attention in Recurrent Neural Networks for Ransomware Detection. In Proceedings of the ICASSP 2019—2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 3222–3226.
55. Vinayakumar, R.; Alazab, M.; Soman, K.P.; Poornachandran, P.; Venkatraman, S. Robust Intelligent Malware Detection Using Deep Learning. *IEEE Access* **2019**, *7*, 46717–46738. [CrossRef]
56. Yuxin, D.; Siyi, Z. Malware detection based on deep learning algorithm. *Neural Comput. Appl.* **2017**, *31*, 461–472. [CrossRef]
57. Cui, Z.; Xue, F.; Cai, X.; Cao, Y.; Wang, G.-G.; Chen, J. Detection of Malicious Code Variants Based on Deep Learning. *IEEE Trans. Ind. Inform.* **2018**, *14*, 3187–3196. [CrossRef]
58. Rhode, M.; Burnap, P.; Jones, K. Early-stage malware prediction using recurrent neural networks. *Comput. Secur.* **2018**, *77*, 578–594. [CrossRef]