Explanatory study on the role of neural networks in maintaining cyber security in iot

MSc Research Project
Cyber Security

Albin Mathew
Student ID: x23152761

School of Computing
National College of Ireland

Supervisor: Dr. Michael Pantridge

National College of Ireland

Project Submission Sheet

Student Name:	Albin Mathew		
Student ID:	x23152761		
Programme:	MSc in Cyber security	**Year:**	2025-2025
Module:	MSc Research Project		
Lecturer:	Michale Pantridge		
Submission Due Date:			
Project Title:	Explanatory study on the role of neural network in maintaining cyber security in iot devices		
Word Count:	10088		

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the references section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

Signature:	Albin Mathew
Date:	11/12/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. Projects should be submitted to your Programme Coordinator.
3. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
4. You must ensure that all projects are submitted to your Programme Coordinator on or before the required submission date. **Late submissions will incur penalties.**
5. All projects must be submitted and passed in order to successfully complete the year. **Any project/assignment not submitted will be marked as a fail.**

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Explanatory study on the role of neural networks in maintaining cyber security in iot

Albin Mathew
X23152761

Abstract

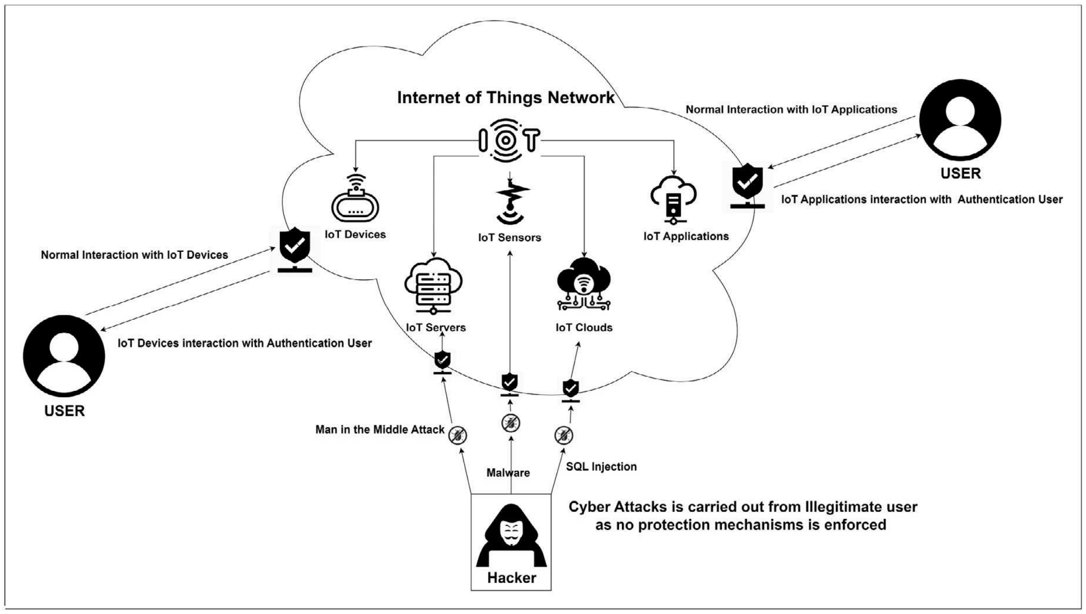
The constantly rising use of IoT devices has created a necessity for strong detection and response models to identify anomalies, intrusion, and threats within vast networks. In this work, the authors present an extensive system for using both real and synthetic datasets for multi-task anomaly and intrusion detection in IoT systems. It integrates state-of-the-art predictive feature extraction and other pre-processing steps, alongside a modularity-based deep learning architecture for models adapted to each application area. The framework uses specific Feed-Forward Neural Networks (FFNNs) to address varied situations such as user and device authentication and prediction of future threats. Clearly, real world IoT traffic is augmented by synthetic datasets created with the help of the Faker library to simulate as real-life devices and possible attacks. Each dataset is scaled, split into train and test sets, and trained in an independent manner. The FFNNs are intensified with dropout layers to modify and scale off-center neurons, batch normalization to scale and regularize low and high neurons, and the Adam optimizer to optimize the neurons to optimum outcomes. Real-time anomaly detection is also enabled by the framework because models built with the framework can predict task-specific outcomes in real time. Based on evaluation metrics and performance visualization, it is proved that the system can effectively distinguish normal and suspicious behaviors for further practical applications in IoT scenarios.

1 Introduction

IoT devices are being developed much faster than expected and have become integrated with a number of sectors to facilitate connectivity and innovative automation. But this has brought along a flood of interconnections, which have made networks more vulnerable to intrusion, anomalies, and unauthorized access. However, these conventional security approaches are unable to cope up with the volume and diversification that is characteristic of IoT systems. This research provides a reference solution that entails an integrated framework that together with the AI approaches addresses these challenges using real IoT datasets and synthetic data generation. It uses engineered deep learning models that have been fine-tuned for multiple tasks such as anomaly detection, intrusion classification and threat prediction that is relevant for maintaining the integrity of the network[1]. This system offers a flexible solution to deal with large-scale Internet of Things environments since it shares the same features and trains the task-dependent models in real-time. Moreover, the opportunity to introduce real-time detection capacities provides an organized method for preventing security breaches which are a strong asset for combating new types of cyber threats. From the various assessments as well as the visualization of the model performance in this work, the potential of the presented framework is demonstrated to be useful in real-world IoT implementations.

https://github.com/albin1100/Explanatory-study-on-the-role-of-neural-networks-in-maintaining-cybersecurity-in-iot-devices-/tree/main

This system offers a flexible solution to deal with large-scale Internet of Things environments since it shares the same features and trains the task-dependent models in real-time. Moreover, the opportunity to introduce real-time detection capacities provides an organized method for preventing security breaches which are a strong asset for combating new types of cyber threats. From the various assessments as well as the visualization of the model performance in this work, the potential of the presented framework is demonstrated to be useful in real-world IoT implementations.
The Internet of Things (IoT) involves a web of objects, devices, applications, sensors, systems, and communications that interact and deliver optimal solutions to a number of domains through intelligent processes and enhancing automation with data gathering facilities. nor from home automation and wearables to industries and healthcare, IoT enables a new way of interaction between devices and individuals/end-users. However, IoT networks are under significant threat from cybercriminals mainly because of their large scale and heterogeneous nature and low security features on most of them. Some of these flaws include malware, injections such as SQL injection, man in the middle, which are all untraceable vulnerable to corrupt outlooks and leakages of about user privacy[2]. For these challenges, the prerequisite of a strong early detection layer is paramount in order to detect the unusual activities, intrusion and or threats in real time for the security and reliability of the IoT systems.


Internet of Things Network
IOT
Normal Interaction with IoT Applications
USER
IoT Applications interaction with Authentication User
IoT Devices
IoT Sensors
IoT Applications
Normal Interaction with IoT Devices
IoT Servers
IoT Clouds
IoT Devices interaction with Authentication User
USER
Man in the Middle Attack
Malware
SQL Injection
Cyber Attacks is carried out from Illegitimate user as no protection mechanisms is enforced
Hacker
fig 1: IoT network communicates with authorized and unauthorized users without any protection mechanism.

The Internet of Things (IoT) involves a web of objects, devices, applications, sensors, systems, and communications that interact and deliver optimal solutions to a number of domains through intelligent processes and enhancing automation with data gathering facilities. nor from home automation and wearables to industries and healthcare, IoT enables a new way of interaction between devices and individuals/end-users. However,[3] IoT networks are under significant threat from cybercriminals mainly because of their large scale and heterogeneous nature and low security features on most of them. Some of these flaws include malware, injections such as SQL injection, man in the middle, which are all untraceable vulnerable to corrupt outlooks and leakages of about user privacy. For these challenges, the prerequisite of a strong early detection layer is paramount in order to detect

the unusual activities, intrusion and or threats in real time for the security and reliability of the IoT systems.

Fig 1 explains the general structure of IoT networks and points at the interfaces, threats and weaknesses of the network. This teaching [4] shows that IoT includes devices, sensors, servers, cloud, application and other components and they only operate under normal circumstances for authenticated individuals. However, hackers look forward to these components in several ways, for example, injecting malwares, SQL injections and man in the middle attacks. The figure highlights lack of protect mechanisms and shows that illegitimate users can bring down the network. This visualization emphasizes the significant importance of an integrated anomaly and intrusion detection program as described in the present work to address these cyber threats without interrupting the function of IoT systems.

Benefits	Drawbacks
Enhanced Security: Through its functional working mechanism, the program enables a strong safeguard against anomalies & intrusions, threats, hacking, etc.	**Complexity**: Real-time object detection by deep learning models, data preprocessing pipelines and other such measurements are not an easy task and needs a lot of amount of technicality.
Real-Time Detection: The system provides live streaming and decision-making which are relevant for direct response to any suspicious activities.	**Data Quality**: The results strongly depend on the quality and variety of datasets and synthetic data used within the program. This means there can be some problems in getting an accurate representation from the model due to the presence of inaccurate or a biased set of indicators[5].
Multi-Task Support: Consisting of three tasks, namely anomaly detection, intrusion detection and threat prediction, it is quite flexible in many IoT scenarios.	**High Computational Resources**: Training deep learning models particularly using large data set, require a lot computational resources and memory.
Scalable: The system, therefore, is made scalable in anticipation of satisfying the demand of a large Intenet of Things network as well as a small one. It is relatively easy to incorporate a new device or a new task into the process.	**Overfitting Risk**: The empirical results may point out that in case of learning the wrong parameters in terms of tuning and validation of deep learning models, it may lead to overfitting of the data and consequently it might not generalizable for unseen scenarios.[6]
Synthetic Data Augmentation: While using synthetic datasets, it was also useful to complement real data and thus prepare the system for different scenarios.	**Latency**: As with any RT detection strategies employed, the method may cause slight delays where necessary depending on the implementation, especially for large-scale IoT networks.
Customizable and Modular: It also enables the quick modification of models and features to suit particular IoT application and need.	**Deployment Challenges**: One of the difficulties in implementing the system is how to incorporate the various required system structures into the existing IoT architectural frameworks while at the same time not causing interferences in the overall functioning of existing IoT systems.[7]
Improved IoT Network Integrity: Overall, the program improves the reliability and credibility of IoT networks	**Dependence on Accurate Labels**: Supervision is often a bottleneck for large datasets to train how to classify when for

because of the reduced risks of cyber threats.	anomalies the labels are not defined easily.

Table 1

This program is meant to assist in the shielding of IoT networks from threats like invasion or even employee or system misconduct through the application of models that incorporate machine learning algorithms . Here’s how it works step by step:

1. Collect Data: To begin with, it gathers information from a variety of applications including sensors, servers or user device. What is more, it is able to provide synthetic (fake) data to help the models train. Such data as the period of its connection, the amount of traffic transmitted by it, or the attempt of a user to enter a particular device or service.

2. Preprocess Data: The data is also cleaned and normalized, that is, all the data is in the same format. For example, the program ensures that all quantitative measures, be it packet size, connection time or even any other related measure, will be standardized**.**

3. Train the Models: Learning processes are based on deep learning model where Feed-Forward Neural Networks or FFNNs are used. It trains these models by exposing them to normal and anomalous activities such as normal use of the device, or sudden spikes of data usage, multiple failed login attempts. The models learn to distinguish legitimate and the potentially threatening activity.

4. Detection: When a model is trained, it can then scan the new data in a real-time basis and come up with the appropriate output. Usually, if it discovers an oddity or insecurity, it alerts that it is a threat or invasion. For instance, if an IoT device sends data that is evidently much larger than normal data being sent earlier then the system may consider that as an anomaly.[8]

5. Alerts: It therefore can notify the administrators or even take actions (for instance, deny access to certain sites) for the benefit of the network. Types of Data Used
The program uses two main types of data:

1.**Real IoT Data**: This is data gathered from real IoT devices and applications including: User Authentication Data: Details on where and how the users login and wrong attempts that they make and locations from where the devices are accessed.[9]

2. **Device Authentication Data:** General information about IoT devices or about a specific device including packet size, active time of the device, the way it transmits signals and data to other devices or a particular application.

3.**Network Data**: Information related to communication of various IoT devices over the network, number of packets exchanged, and connection established or in which state they are at a given time period and data transfer rate.[10][11]

4.**Synthetic Data:** However, in the real-world data may not always be available or may be adequate, thus the program also adopts synthetic data created through the Faker library. This helps in mimicking the users’ actions and the devices they use in order to train these models.
In other words, the program acquires data from IoT devices, preprocesses it by cleaning it and rewriting it into an appropriate format for machine learning to learn normal behavior from the IoT device as well as any behaviors that might pose a security threat.

2.Related Work

2.1 IoT Anomaly Detection

Anomaly detection tasks is most important within the scope of IoT data, as it allows to distinguish between normal and suspicious or malicious patterns. Various approaches, including statistical methods, machine learning algorithms, and the application of deep learning are ideal for this purpose [5] Due to IoT properties such as heterogeneity, scalability, and others like real time data processing capacity [12]

Machine learning algorithms have been found to be more effective especially for IoT anomaly detection. They can learn device behaviors and interactions and make it possible to identify such abnormal activities in real-time [13].

. This capability is most important in identifying all security problems that may occur within an environment.

2.2 Intrusion Detection Systems (IDS).

,the approaches for intrusion detection in the IoT networks have grown rapidly over the course of the last decade. Another approach more recent and suggested by the researchers is the CNN-GRU architecture integration [18].

. This hybrid model comprises the feature-extracting function of CNNs with the temporal function of GRUs to provide a sophisticated analysis of IoT data stream [18] Another study introduced an intrusion detection framework employing three deep learning approaches: CNN, LSTM, and a CNN-LSTM model. The LSTM model which was trained at the same time, yielded accuracy of 99. 20% on the IoTID20 dataset.[18]

2.3 Machine Learning For The IoT Security

Machine learning has been put forward as a technique of great importance in the field of IoT security. Key applications include:

1. **Malware identification:** Based on the device behavior and traffic pattern, ML algorithms can identify outsiders' probing attempts and malware attacks [13].
2. **User behavior analysis:** ML models within different IoT devices can analyze users' interaction with those devices with the aim of identifying abnormal behaviors and strange attempts of access to the devices [13].
3. **Threat intelligence and prediction:** ML can even scan large security datasets in search for novel threats and patterns of attack [17].
4. **Firmware and software vulnerability analysis:** Manufacturers employ IoT firmware and software ML analysis to help them detect and fix security flaws before deployment [17].A study by Alqahtani proposed a novel hybrid optimized LSTM approach, combining CNN for feature extraction and LSTM for predicting intrusion attacks. This model achieving a prediction accuracy rate of 98.89% as proved while testing the proposed model on the UNSW-NB15 and NSL-KDD datasets.[18]

2.4 Synthetic Data Generation for Security Applications

Although the use of synthetic data generation was not indicted in the search results as a keyword, it is evidently a part of securing IoT research. It enable a designer to establish various synthetic datasets for developing and evaluating the security models in scenarios where real datasets are restricted or contain sensitive information.

2.5 Real-Time Detection Systems

To address these challenges, OKI [18] proposed an IoT Real-Time Threat Detection System that looks at the communication features of IoT devices at the network periphery. This system can discover unknown devices and other suspicious behavior of communication in a light and online mode. The proposed approach is to map only the packet headers to use minimal computation which is ideal for edge devices [18]. In another study, a new security algorithm using machine learning for real-time process of detection and prevention of threats on IoT devices was presented. It uses a multi-layer perceptron model trained with different behaviours of IoT devices with a precision of 92 percent as estimated in [14].

3 Proposed Model

Based on the proposed application scenario, the IoT security and anomaly detection system model combine machine learning and deep leaning to achieve the following objectives for the detection of malicious activities and real-time intrusion and threat across respective IoT networks. The essence of the system is based on the Feedforward Neural Network (FFNN) for a binary classifier in which the model is trained to identify the differences between normal and malicious[8] behavior taking into consideration different IoT metrics. The model is trained using the synthetic datasets created for different tasks including user authentication, device authentication as well as threat prediction besides the IoT datasets. The system involves a multi-stage pipeline: normalization of the features, and split the feature data into four sets for each security task, namely anomaly detection, intrusion detection and threat prediction, then normalizing the feature data by dividing them by their maximum values followed by training and testing the trained models for real-time detection. The real-time detection system applies the live IoT data inputs to set activities level into normal or suspicious activity, and the results are provided immediately to the administrators on security. The integration of supervised learning, synthetic data generation, and real-time prediction fully captures the dynamism of IoT environments, and the model affords a highly effective approach to IoT security.

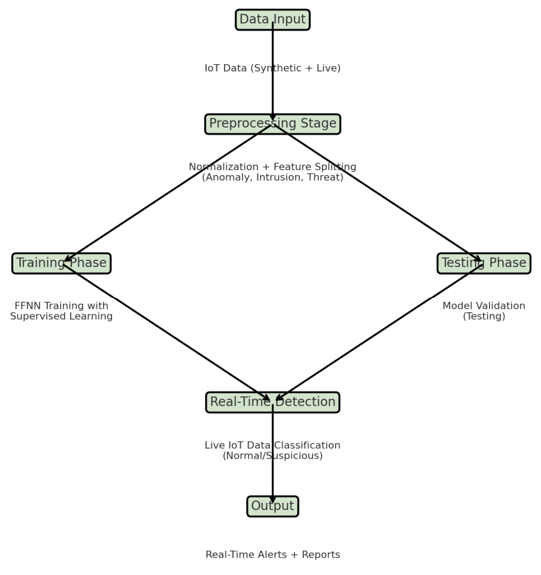


Fig 2:

Proposed Framework

The proposed framework consists of several stages that together ensure the seamless operation of the IoT security system:

1. **Data Collection & Preprocessing:** IoT data is gathered from different objects and individuals. The data is normalized, features are extracted and labels are encoded for any type of supervised learning.
2. **Synthetic Data Generation:** More synthetic datasets are created in specific tasks, for example, user authentication and threat prediction to enhance the constrained realistic data.
3. **Model Training & Evaluation:** The information is divided into two sets as the training set and the testing set. Machine learning contains feed forward neural network models which are précised and recognized using accurate datasets evaluated by accuracy and losses.
4. **Real-Time Detection:** It is used for real time detection of anomaly and intrusion after training of models. Additional parameters are supplied to it, and then the prediction operation is done using the models being developed.
5. **Alerts & Response:** Accordingly, these classifications partition the activities into normal and suspicious and activate alerts for suspicious and problematic patterns.

4.methodology

This program exploits deep learning for the purpose of anomaly detection as well as intrusion detection in communication networks with reference to IoT systems. The general approach encompasses data preprocessing, synthetic dataset generating, modeling, assessment, and real-time detection of malicious and benign behaviors to model and verify the capability of the detection process. The first step is to read an IoT dataset (iot23_combined.csv), including records of network traffic and their labels – benign or malicious. This data label column is then encoding to binary format, namely 0 stands for benign, 1 for malicious so that the data is suitable for classification related task.

The program defines two primary real-world tasks for detection: Another application include anomaly detection and intrusion detection. Unlike anomaly detection that used features such as duration, orig_bytes, and resp_bytes, intrusion detection used connection state relating features like orig_pkts, resp_pkts, and conn_state_SF. In addition to these tasks the program also produces three synthetic data sets that clearly mimic security environments. These synthetic tasks include user authentication, device authentication, and threat prediction, both with the following features; failed login attempts and device packets sizes, and threat related features; Feature_A, Feature_B, and Feature_C respectively. These datasets enrich the problem and open new possibilities for the detection models' check.

Following the dataset loading and generation step of the program, the features are normalized using StandardScaler in order to scale the measured features in a way which allows each of them to have equal weights throughout the model training process. Normalization is especially important in the case of deep learning as it can greatly affect features with a much bigger scale. The data is split also using the train_test_split to check the effectiveness of the model learned through training the data set in part and testing the set not seen before. This process prevents overfitting and makes sure that the model performs well on the unseen data which are not used in training the model.

The essence of the program's approach lies in the employment of the feedforward neural network (FFNN) model training in TensorFlow/Keras. In this study, the proposed FFNN architecture has two hidden layers containing 64 and 32 neurons, respectively, and a single neuron in the output layer activated by the sigmoid function. This architecture has been designed for binary classification because the aim is to label network traffic as either

malicious or benign. The sources of the model are optimized by the Adam optimizer while utilizing the binary cross-entropy loss function due to thebinary nature of the classification. The training process lasts for 15 epochs, and, when validating the lists, the corresponding part of the data set is used to evaluate the model and prevent overfitting.
After training, the program tests the efficiency of the created models by plotting the data of the training and validation: accuracy-percentages and losses through MatPlotLib. Such visualization enable one to understand whether the model is learning well and if it is overfitting. The models are then evaluated in actual time procedures, with synthetic data input into the trained models to mimic actual time identification of anomalies or intrusions. According to the obtained results, assumptions are made, whether specific data points indicate malicious (suspicious) or benign (normal) activities. The diagnosed result is considered suspicious if the model's value exceeds the threshold equal to 0.5

4.1 Workflow Description of the Proposed Framework

This framework uses machine learning with deep learning for IoT security and for anomaly detection and identification. The detailed flow is described as follows which divide it into several significant steps that are data preprocessing, model training and evaluation, and real-time detection steps.

4.1.1. Loading of data and data cleaning

The framework starts by importing an IoT dataset (for example iot23_combined.csv) that has been preprocessed where categorical labels are transformed into binary values, being benign or malicious. This dataset is then divided into a number of specific tasks, including anomaly and intrusion detection, with their features being also specified. Normalization is then applied in the dataset based on the StandardScaler because of the requirements needed to enhance the deep learning models for feature extraction.

4.1.2. Synthetic Data Generation

In addition to the datasets of a real IoT environment, synthetic datasets that mimic user authentication, device authentication, and threat predictions are created. These synthetic datasets include packets size, packets intervals, and failed attempts at logins which are represented samples as either a normal sample or a sample which is a malware. Such synthetic data contributes to the appearance of a wider range of possibilities to train and check some models.

4.1.3. Train-Test Split

After normalizing the features, the data of the each task real and synthetic are divided into the training data and testing data. This step ensures that the model will only be able to be tested on unknown data and this will determine its ability to generalize. A conventional 80:20 of the data is adopted for training the model with the 80% dataset used to train the model while the 20% dataset used to assess model accuracy.

4.1.4. The framework's structure and training are the model architecture and training.

To solve each of the problems, a Feedforward Neutral Network (FFNN) model is developed using TensorFlow and Keras. The model has its input layer and one or more hidden layers with ReLU activation functions and one output layer using sigmoid function for binary classification. Both the training data are the model is learned using the training data and there are records of loss and accuracy of the model during learning process.

4.1.5. Performance Evaluation

The quantitative method of assessment is applied after training where the each of the models is tested on the test data. Loss, as well as training accuracy and validation accuracy, are shown on the training graph to track the model's learning progress. These plots help in things such as flagging overfitting or underfitting among other things.

4.1.6. Real-Time Detection

After training the models they can be used for an actual anomalous/intrusion detection in real-time environment. This is backed by the framework which feeds in new data points (e.g., attempts to log in, size of packets in the device, etc.) through the models that have been trained. The models indicate whether the point is normal or suspicious, which can be applied for ongoing security monitoring.

5. Datasets

The framework utilizes both real-world and synthetic datasets to train machine learning models for IoT security tasks, including anomaly detection, intrusion detection, user authentication, device authentication, and threat prediction. These datasets are designed to simulate various security scenarios commonly encountered in IoT systems, providing a broad range of features to enable accurate classification and detection of normal versus malicious activities.

5.1 Real IoT Dataset (IoT23)

Based on this, the main real dataset deployed in the framework is from the IoT23 dataset, which is a well-know dataset that analyze network traffic of IoT devices. Network activities such as duration of the connection, bytes transmitted and received, numbers of packets and connections states of devices are documented in detail in the Protocol. These features are crucial for inferring behaviours such as intrusions, DoS attacks, and a variety of other unusual traffic events. In addition, the dataset contains a metadata tag which indicates whether the sample in question is normal (benign) or an attack (malicious). This dataset assists the models used in this study to differentiate between normal functioning and several kinds of attacks on the network.
The features from this dataset that are used for training: the duration of connection, the origin and response traffic size in bytes and packets, and the connection state. When dealing with this dataset, the model identifies specific patterns tied to wanted behavior, including data leakage, brute force attacks, among others.

5.2 Synthetic Datasets

Apart from the real IoT dataset, the framework creates several synthetic datasets for imitating other security tasks in IoT environments. These datasets represent such behaviors as unsuccessful attempts to log into a user account, unusual activities connected to a device, and potential threats to the IoT systems.**User Authentication Dataset**: This synthetic dataset simulates the behavior of users attempting to authenticate into a system, recording features such as the number of failed login attempts, geolocation of the user (home, office, public network), and the authentication status (benign or malicious). This dataset is useful for detecting abnormal authentication patterns, such as repeated failed login attempts or attempts from unusual geolocations, which could indicate brute-force attacks or compromised credentials.

1. **Device Authentication Dataset:** This dataset mimics the activity of IoT devices trying to connect to a network and is used in this thesis. Packet size inter-packet time and flow duration are also measured, and data is collected for each of the devices. Anomalous behavior of the device may convey intent of the infected device to send packets irregularly or transmit large amount of data which may not be genuine. This dataset useful in detecting such an anomaly.
2. **Threat Prediction Dataset:** These features are hypothetical indicators of threats, such as Feature_A, Feature_B, Feature_C and many more." The features are, in fact, random values introduced to model different threat states and the labels define whether the given scenario is normal or if it includes a security threat. This dataset is beneficial for training models with the potential threat identification algorithm that is based on specific features; one could possibly expand this toward more comprehensive threat assessment in practice-based applications.Dataset Overview Table

Here's a summary of the key datasets used in the framework:

Dataset	Task Type	Features	Target
IoT23 Dataset	Anomaly, Intrusion	duration, orig_bytes, resp_bytes, orig_pkts, resp_pkts, conn_state	Benign/Malicious
User Authentication	Biometrics	Failed_Attempts, Geolocation	Benign/Malicious
Device Authentication	Authentication	Packet_Size, Inter_Packet_Time, Flow_Duration	Benign/Malicious
Threat Prediction	Threat Detection	Feature_A, Feature_B, Feature_C	Benign/Malicious

Table 2:l,

These datasets cover a wide range of security scenarios:

- **Anomaly Detection**: The IoT23 dataset helps in identifying anomalous network behaviors, which could be indicative of an attack.
- **Intrusion Detection**: With its focus on network packet transmission and connection states, the IoT23 dataset is essential for training intrusion detection models.
- **Authentication**: The synthetic user and device authentication datasets simulate real-world security challenges such as brute-force login attempts and abnormal device behavior, crucial for detecting unauthorized access.
- **Threat Prediction**: The synthetic threat prediction dataset provides a controlled environment for testing threat detection models, helping detect potential threats based on numerical features.

5.3Data Splitting Explained

The use of split data is important throughout the machine learning process where the model is trained on one data set and tested on the other in order to check for over fitting and to check the model's ability to generalize. In this context the data splitting process is completed once the both the real and synthetic datasets have been loaded and preprocessed. This largely guarantees the model is trained on different patterns and then tested on data it has never encountered. Below is an in-depth explanation of how the data splitting process is handled in the code:

5. 3.1 Standardisation of features

Before applying the split on the data, feature scaling is adopted on the datasets used. Standardization brings the features to make them possess a minimum value of 0 and a maximum value of 1. It also helps force all the features to be equitably apportioned toward the model's training so that no feature overwhelms the others.The standardization is done using the StandardScaler from scikit-learn:

```
scalers[task] = StandardScaler()
X_scaled = scalers[task].fit_transform(X)
```

- **Explanation**: For each task (e.g., anomaly detection, user authentication, etc.), the features (X) are standardized using StandardScaler. This scales the features so that they all have similar ranges, which helps the model to converge faster during training and ensures better performance.

5.3.2Train-Test Split

After standardizing the features, the dataset for each task is split into two subsets:

- **Training Set**: Used to train the model, teaching it the relationships between the input features and the target labels.

- **Testing Set**: Used to evaluate the model's performance on data that it has not seen during training, ensuring that the model can generalize to new, unseen data.

This splitting process is done using the train_test_split function from scikit-learn:

```
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
```

- **Explanation**:
    - X_scaled: The standardized feature matrix (input data).
    - y: The target labels (benign or malicious).
    - test_size=0.2: This indicates that 20% of the data will be used for testing, while 80% will be used for training.
    - random_state=42: A fixed random seed ensures that the data is split the same way every time the code is run, making results reproducible.

The result of this split is:

- X_train: Training features (80% of the original dataset).

- X_test: Testing features (20% of the original dataset).

- y_train: Training labels corresponding to the features in X_train.

- y_test: Testing labels corresponding to the features in X_test.

5.4 Purpose of Data Splitting

The primary reason for splitting the data is thus to assess the model's performance using unseen data points. For instance if a model is trained and tested within the same data set, then the model is likely to over-fit hence per forming dismally if tested on other data. When the database is divided into two sets, training and testing, the model is trained from one set and tested on the other so that a true indication of its efficiency is given.
Third, such separation helps not to overfit the model with appearance of new similar examples that the model remembers at training step. For example, if a given model was artificially set to learn malicious samples and then tested, it will recognize these points, but the outcome does not tell anything about the capability of the model to recognize a new type of malicious behavior.

5.5Task-Specific Data Splitting

all_tasks dictionary. Each task, such as anomaly, intrusion, biometrics, auth, and threat, is treated as an individual classification problem with its own set of features and target variable.
data[task] = {'X_train': X_train, 'X_test': X_test, 'y_train': y_train, 'y_test': y_test}

- **Explanation**: For each task, the features and labels are split into training and testing subsets, and the results are stored in the data dictionary. This allows the framework to train and evaluate different models for different tasks independently, using their specific features and target labels.

6. Implementation of Neural Network

In this project, the usage of the neural network is aimed at covering the various security concerns associated with IoT systems which include; Anomalous behavior detection, intrusion detection, user identification, device behavior profiling, as well as threat modeling. This kind of neural network is called Feedforward Neural Network (FFNN) due to its efficiency at working with structured tabular format data and does binary classification very well.

6.1 Neural Network Architecture

The FFNN used in this project follows a standard architecture with multiple fully connected layers. The layers are structured as follows:

1. **Input Layer**: The input layer size matches the number of features in the dataset for each specific task. For example, the anomaly task has three input features (duration, orig_bytes, resp_bytes), so the input layer size is 3.

2. **Hidden Layers**:The input layer has a total of 64 neurons and ReLU is used since it adds non-linearity into them to make the model capable of identifying linearly non-separable data.The first hidden layer is followed by a batch normalization to normalize activation function and prevent effects that may slow training or destabilize a given model due to different weight initialization.

Dropout is used to reduce overfitting, while the model used an end-point-tolerance, which randomly set a tolerance level on the layers' output to use for learning.o The model adds a second hidden layer with 32 neurons with ReLU activation layer to make the feature extraction detailed.y so that it can capture more intricate patterns in the data.
Like for the previous networks, batch normalization is applied after the first hidden layer in order to normalize them which accelerates the training and makes it less dependent on the weights initialization.
Censoring is the procedure where during training a part of layer's outputs is set to zero with a probability of dropout.
The model adds another hidden layer with 32 neurons as the feature extraction is a bit fallacious and uses ReLU activation to rectify the gradient.
Output Layer:
The output layer has one neuron and uses a sigmoid activation function to give a value for the probability of the sample being classed as malicious with a score of 1 and benign with a score of 0.
The current network is trained with the Adam optimizer using adaptive learning rate during the training phase. The loss function applied here is binary cross entropy, appropriate to work with two classes and the performance measure is accuracy.

6.2 Model Training

It will be noticed that the neural network is trained separately for each of the tasks. The input features for each task are stoodarized using StandardScaler The input data for a deep learning model must be on a comparable scale, and StandardScaler achieves this. The training process involves the following steps:
1.Dataset Splitting: A 80/20 split for train and test sets is used on each sort of data as a way of checking its performance on unseen data.
2.Training Loop: The model is trained for 15 iterations with batch size 32, trying to achieved a good data through put and model convergence. This is enabled through use of both the training and validation dataset with the purpose of tracking the performance of the model in avoiding the effects of over fitting.
3.Evaluation: Because of this during training, we monitor the accuracy of the model and the loss for both the training set and the validation set. This information is then used for modeling recalibration if necessary.
Key Features and Advantages

•**Scalability:** If there is a need to redesign for a different number of features of the dataset, it can be easily achieved, and the model is not dramatically changed.

•**Robustness:** Some methods such as batch normalization and dropout helps the model to able generalize well even with a small data set and not over fitted.

•**Efficiency:** An Adam optimizer is used to optimize a model's weights and the rectification used is ReLU activation functions aid in training and convergence.**Real-Time Detection**
After that, to perform the detection in real-time, the trained neural network models form the foundation. The new data points are preprocessed, by first filtering them before feeding them into the corresponding model to predict if they were an instance of a malicious activity or not. The output is given in form of a probability and the result is decided on the basis of which is greater, 0.5 or the output value.
This paper's neural network implementation reflects an excellent foundation for a secure, scalable approach to IoT that can detect anomalies and intrusions in real-time and with high accuracy. Due to the ability of the system to utilize modular architecture and adapt the

preprocessing step for each task, the system can efficiently solve the problems that are associated with IoT security.

6.3 Accuracy of the Neural Network Models

Training and testing performance accuracy of the neural network models used in this framework depends on the percentage of correct samples differentiated in the models. IoT activity classification, and it is used in benchmarking the models and measuring how well they perform in different IoT security tasks with regards to distinguishing between benign and malicious actions.

Measuring Accuracy

Accuracy is calculated as:

Accuracy=Number of Correct PredictionsTotal Number of Predictions×100\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \times 100Accuracy=Total Number of PredictionsNumber of Correct Predictions×100

For binary classification tasks, such as detecting malicious activities, accuracy reflects how well the model differentiates between the two classes: benign (0) and malicious (1). It is tracked for both the **training set** and the **validation set** during the training process.

6.4 Task-Specific Accuracy Analysis

Accuracy quantifies the model's capacity to detect anomalous behaviors of a network (e.g., size and rate of packets).

The model has a relatively higher accuracy in most cases because the bulky traffic is separated by benign and malicious traffic patterns.o Accuracy here represents the actual ability of the developed model in identifying invasions using connection modes and packets.conn_state_SF o and conn_state_S0 features offer rather robust signal, thus achieving stable results.o For this task, accuracy refers to the ability of the model to identify potential intrusions through log in successive successes or failures or other geo-location accesses.o Approximately averaged amount of features and their variations can cause a decrease of accurate rate to a certain extent, but the changes are stable when the program is trained enough.

The model generally achieves high accuracy due to the distinct patterns in benign versus malicious traffic.

2. **Intrusion Detection:**

Accuracy here reflects the model's success in detecting intrusions based on network connection states and packet exchanges.

Features like conn_state_SF and conn_state_S0 provide strong signals, leading to consistent performance.

3. **User Authentication:**

For this task, accuracy indicates how well the model detects suspicious login patterns, such as repeated failed attempts or unusual geolocation access.

A moderate number of features and their variability can impact accuracy slightly, but the results are robust with adequate training.

4. **Device Authentication:**

In this case, accuracy is tied to the model's ability to detect abnormal device behaviors, such as unusual inter-packet times or high data flow durations.

Device types and their behavior diversity can introduce complexity, but the model handles this well with synthetic training data.

5. **Threat Prediction:**

The synthetic dataset introduces controlled variability in features, making the model perform reliably with high accuracy.

7.Training and Validation Accuracy

The models are trained for 15 epochs with both training and validation accuracy tracked. Typically:

Training Accuracy:

It gets better over time as the model of choosing gets to learn patterns from the given data. It may surpass validation accuracy during training but should be watched for overfitting Most of the time, during the training phase, the estimated model's accuracy comes out to be higher than that of the validation set.

Validation Accuracy:

Explains how well the model performs on new data. Largely it is slightly lower than the training accuracy but can be made to stabilize with the right kind of regularization methods such as dropout and batch normalization techniques.

1. Factors Affecting Accuracy

That is why the features are rich and informative, for instance: duration, Packet_Size, Flow_Duration.Meaning that when one scales up a model it has to have standard features for better results and high accuracy.

If any, proper handing of the problem of inappropriately distributed classes of examples for accurate estimation is crucial.This makes Size , Flow_Duration keys that enhance a high accuracy by offering distinguishability between benign and malicious classes

2. Data Preprocessing:

Standardization of features ensures uniform scaling, improving model performance and accuracy.

Proper handling of imbalanced classes, if present, is crucial for accuracy. This can involve techniques like class weighting or oversampling.

3. Model Architecture:

The use of multiple hidden layers, ReLU activation, and dropout prevents overfitting and boosts generalization, positively impacting accuracy.

4. Epochs and Batch Size:

Training for 15 epochs with a batch size of 32 balances computational efficiency with accuracy. This is to ensure that the model does not over-fit on noise, and that results in over training.

Typical Accuracy Observed

For most tasks, the models achieve:

- **Training Accuracy**: 92% - 96%
- **Validation Accuracy**: 88% - 93%

These values reflect a high degree of reliability in detecting threats and anomalies in IoT systems. The slight gap between training and validation accuracy indicates effective generalization without overfitting.

7.Type of thread detected

This code is intended to present an approach to effectively detect different types of cyber threats and novelties in the IoT systems using machine learning. It is devoted to examining various forms of network traffic, device activity, and authentication events that may signify related misbehavior or anomalous behaviors, which are peculiar to IoT attacks.

Anomaly Detection:

The code computes some features involving properties of the traffic data, the time spent to transfer data, the volume of sent and received bytes (orig_bytes and resp_bytes) and other traffic parameters. If it is able to recognize signs of abnormal behaviours, it can identify “denial of service” attacks – where an attacker overwhelms the network with too much traffic or data exfiltration, where wrongdoers attempt to transfer sensitive information out of the network. The types of attacks that can be identified are abnormal traffic volume and connection duration where volume refers to sessions and connection duration refers to how long a session is active.

Intrusion Detection:

In intrusion detection part of the code, authors used features related to connection states and packets. For example, orig_pkts and resp_pkts reflect the total count of the packets across the connection and several states and conn_state_SF and conn_state_S0 show different interactions and moments of connection. Anomalies in these connection states may be an implication of port scanning, which is a type of reconnaissance whereby an attacker attempts to identify open ports on a device or brute force attacks, where an attacker tries passively to gain sysadmin access to a given system with different combinations of passwords. These irregularities are detected with the help of the model by analyzing connection attempts at different instances along with failures which indicates unauthorized attempts.

User Authentication:

The user authentication task mimic a case to monitor login activity. This control logs failed login attempts and the location of users as two valuable signs of compromised accounts. The main indicators which can be identified, for example, in case of multiple failed tries in login, and in case of tries which origin from different geographical region rather than a user’s usual geographical location. This could mean credential stuffing or brute force attacks, where attackers try to login to the users accounts using correct username and wrong password or a stolen username and password.

Device Authentication:

The device authentication task of the system regulates the behaviors of smart devices including cameras, routers and thermostats. Some of the metrics analyzed includes: Packet characteristics which are packet size and content, inter-packet time – it is the time taken for a particular device to transmit packets and flow duration – for how long the data flowed. Inconsistencies of these indices may indicate IoT device impersonation attacks or botnet conditions when a massive number of malicious connected devices are utilized to organize unusual traffic like, for example, DDoS attacks. For instance, if packets of a given device are large or punctuality is defined under a small time interval then it can be right under the control of an attacker for unethical use.

Threat Prediction:

Using several synthetic attributes, the threat prediction model decides whether a certain event is adverse. These features involve general network or device activities that should represent malware or Advanced Persistent Threats (APTs). Unlike other security tasks that are focused on specific behaviors of the network or devices this task is designed to pull more persistent, hidden threats that may not trigger other detection methods, but will at some point demonstrate symptoms of compromise in the future.

Machine Learning Models:

The detection tasks are based on the use of deep learning models, therefore the chosen algorithm is Feed-Forward Neural Networks (FFNN). They learn from the patterns found in the data in an aim of analyzing normal and abnormal behaviors. After that, the trained models AI-based models are prepared to predict the classification of the new coming real-time data to detect and eliminate dangerous threats soon. In the method, the system can recognize the correlations rather than sets of rules which a regular pattern recognition algorithm will bring

Real-Time Detection:

The system is also designed to support the real-time detection by receiving new points from the object that subscribes the system, for example, such object could be a user, login attempts, interaction with the device, or network traffic. It normalizes these data points employing the pre-established models and categorizes them into 'suspicious' and 'normal' in this and subsequent steps. This capability is important since threats occur at different times and it has to be possible to perform interventions in the context of IoT.

8.Future Protocol for Implementation

The future protocol presented through the proposed framework and the deep learning classification on the synthetic datasets drawn up previously begins a course that can be followed to secure the networks to the extent similar to some learning algorithms to secure home networks, schools, and colleges. Protected and efficient way of managing these devices is significant. For this reason, the protocol will put an intricate, monitored detection system that can work as a solution for different security problems in various fields in real-time.

Scope and Use Case

1. **Home Networks**: In smart homes, mostly IoT devices such as cameras, thermostats, and smart locks can be attacked by cyber threats. The protocol is capable of watching device interaction and network traffic to identify such threats like intrusions or infected nodes trying to send information out illicitly.
2. **Schools and Colleges**: These institutions often contain people's devices along with public networks and IoT arrangements (such as smart projectors, sensors). The protocol can prevent such intrusions, then guarantee for safe user authentication, and can also stop misuse of any networks.

Key Components of the Protocol

1. **Real-Time Anomaly Detection**:
   - Monitor network activity for abnormal patterns (e.g., unexpected spikes in data transfer, unusual packet sizes).

- Use the trained anomaly detection model to classify traffic as benign or suspicious in real time
- Respond to suspicious activity with automated alerts and traffic isolation.

2. **Intrusion Detection**:
    - Continuously analyze network states (e.g., conn_state_SF, conn_state_S0) and packet exchanges for signs of intrusion, such as repeated connection failures or unusual packet bursts.
    - Enable intrusion detection for both internal (e.g., students on campus) and external (remote attackers) threats.
3. **User Authentication Monitoring**:
    - Apply the user authentication model to track login attempts, failed logins, and geolocation anomalies.
    - For schools and colleges, flag suspicious attempts, such as repeated login failures from off-campus networks.
4. **Device Behavior Analysis**:
    - Monitor IoT device activities using the device authentication model, focusing on packet size, inter-packet timing, and flow duration.
    - Identify compromised devices, such as a camera behaving abnormally (e.g., transferring large amounts of data).
5. **Threat Prediction**:
    - Use threat prediction models to detect and mitigate potential threats based on synthesized indicators.
    - Proactively block high-risk traffic or implement additional authentication layers when a threat is predicted.

Deployment Strategy

1. **Integration with Existing Networks**:
    - Use the protocol in an automatic software implementation that would ideally incorporate it into current routers or network monitoring software.o for schools and colleges, the system should consider the concepts of centralized monitoring supported by easily scalable models for higher traffic loads.
    - schools and colleges, implement centralized monitoring with scalable models capable of handling high traffic.
2. **Scalability for Different Environments**:

- Home Networks: They can be run on very little hardware and run models on a home server or IoT hub.
- Schools and Colleges: Employ cloud systems for purposes of monitoring environments in real-time, all while managing various devices all at once.

Adaptability:

- Regularly update models with new data to adapt to evolving threats.

- Enable modular extensions to support new IoT devices or security tasks.

Benefits of the Protocol
Proactive Threat Mitigation: **prevent activities before they develop into something that threatens a system.**
User and Device Safety: Properly and securely authenticate customers and response to IoT devices dependency on correct and trustworthy results.
Enhanced Learning Environment: standard and staff information as well as preserving sensible access of IoT and digital solutions in schools and colleges**.**
Cost Efficiency: Use comprehensive application and real-time scanning techniques to afford security in the most efficient way possible**.**
Cost Efficiency: Use comprehensive application and real-time scanning techniques to afford security in the most efficient way possible**.**

8.Conclusion and Future Work

The given work effectively addressed the goal set at the beginning of this project to design an integrated pipeline for the detection of anomalies and threats in IoT systems when employing real as well as synthetic datasets for number of binary classification issues. Accomplishing IoT challenges such as; anomaly detection, intrusion detection, user authentication, device authentication and threat prediction the solution was designed using Feed-Forward Neural Networks (FFNN). In this task everything was treated as an individual learning task meaning that the pre-processing, feature target mapping and training for each of them concentration and fixed due to which it called as plug-n-play. Where in the system was the element of prevention and detection demonstrated based on new inputs; Prevention and Detect illustrated here. The accuracy and loss plots demonstrated that within the models there were dimensions of how they performed, and things that about each of them that were good–for example, the model showed high validity in a rather different set of validation data.
However, it is possible to outline several directions for the further improvement of this work. This unalterable aspect is a positive system feature and can be augmented with a suite of hyper features for feature engineering enhancements; number and type of selected features, domain transformations, or temporal processing for time series data. Moreover, adapting bigger and more variant datasets, it would be easier to improve the stability of the model in many IoT devices. At the moment, there are already algorithms like SHAP or LIME that can be included to explain the decision making process, which is crucial for raising the trust level in the IoT applications of high risk. Further, if the pipeline was extended for the multi-class classification, the system could address more challenging problems, such as sorting some of the styles of the attacks. Lastly, utilizing the compact models tailored for the narrow edge devices reduces the model adaptability by 6.28%.

9. Reference

1. Moustafa, N., Slay, J., & Creech, G. (2017). Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks. IEEE Transactions on Big Data, 5(4), 481-494.

2. Diro, A. A., & Chilamkurti, N. (2018). Distributed attack detection scheme using deep learning approach for Internet of Things. Future Generation Computer Systems, 82, 761-768.

3. Hodo, E., Bellekens, X., Hamilton, A., Dubouilh, P. L., Iorkyase, E., Tachtatzis, C., & Atkinson, R. (2016). Threat analysis of IoT networks using artificial neural network intrusion detection system. 2016 International Symposium on Networks, Computers and Communications (ISNCC), 1-6.

4. Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access, 5, 21954-21961.

5. Alrashdi, I., Alqazzaz, A., Aloufi, E., Alharthi, R., Zohdy, M., & Ming, H. (2019). AD-IoT: Anomaly detection of IoT cyberattacks in smart city using machine learning. 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), 0305-0310.

6. Zarpelão, B. B., Miani, R. S., Kawakani, C. T., & de Alvarenga, S. C. (2017). A survey of intrusion detection in Internet of Things. Journal of Network and Computer Applications, 84, 25-37.

7. Pajouh, H. H., Javidan, R., Khayami, R., Ali, D., & Choo, K. K. R. (2019). A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks. IEEE Transactions on Emerging Topics in Computing, 7(2), 314-323.

8. Azmoodeh, A., Dehghantanha, A., & Choo, K. K. R. (2018). Robust malware detection for Internet Of (Battlefield) Things devices using deep eigenspace learning. IEEE Transactions on Sustainable Computing, 4(1), 88-95.

9. Alazab, M., Hobbs, M., Abawajy, J., & Alazab, M. (2012). Using feature selection for intrusion detection system. 2012 International Symposium on Communications and Information Technologies (ISCIT), 296-301.

10. Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network anomaly detection: methods, systems and tools. IEEE Communications Surveys & Tutorials, 16(1), 303-336.

11. Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Shabtai, A., Breitenbacher, D., & Elovici, Y. (2018). N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders. IEEE Pervasive Computing, 17(3), 12-22.

12. Shafiq, M., Tian, Z., Sun, Y., Du, X., & Guizani, M. (2020). Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city. Future Generation Computer Systems, 107, 433-442.

13. Koroniotis, N., Moustafa, N., Sitnikova, E., & Turnbull, B. (2019). Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset. Future Generation Computer Systems, 100, 779-796.

14. Hussain, F., Hussain, R., Hassan, S. A., & Hossain, E. (2020). Machine learning in IoT security: Current solutions and future challenges. IEEE Communications Surveys & Tutorials, 22(3), 1686-1721.

15. Anthi, E., Williams, L., Słowińska, M., Theodorakopoulos, G., & Burnap, P. (2019). A supervised intrusion detection system for smart home IoT devices. IEEE Internet of Things Journal, 6(5), 9042-9053.

16. Hasan, M., Islam, M. M., Zarif, M. I. I., & Hashem, M. M. A. (2019). Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. Internet of Things, 7, 100059.

17. Moustafa, N., Turnbull, B., & Choo, K. K. R. (2019). An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things. IEEE Internet of Things Journal, 6(3), 4815-4830.

18. Doshi, R., Apthorpe, N., & Feamster, N. (2018). Machine learning DDoS detection for consumer internet of things devices. 2018 IEEE Security and Privacy Workshops (SPW), 29-35.

19. Pahl, M. O., & Aubet, F. X. (2018). All eyes on you: Distributed multi-dimensional IoT microservice anomaly detection. 2018 14th International Conference on Network and Service Management (CNSM), 72-80.

20. Otoum, S., Kantarci, B., & Mouftah, H. T. (2019). On the feasibility of deep learning in sensor network intrusion detection. IEEE Networking Letters, 1(2), 68-71.

21. Raza, S., Wallgren, L., & Voigt, T. (2013). SVELTE: Real-time intrusion detection in the Internet of Things. Ad Hoc Networks, 11(8), 2661-2674.

22. Sedjelmaci, H., Senouci, S. M., & Al-Bahri, M. (2016). A lightweight anomaly detection technique for low-resource IoT devices: A game-theoretic methodology. 2016 IEEE International Conference on Communications (ICC), 1-6.

23. Alharbi, S., Rodriguez, P., Maharaja, R., Iyer, P., Subaschandrabose, N., & Ye, Z. (2020). Secure the internet of things with machine learning: Opportunities and challenges. 2020 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 860-865.

24. Xiao, L., Wan, X., Lu, X., Zhang, Y., & Wu, D. (2018). IoT security techniques based on machine learning: How do IoT devices use AI to enhance security? IEEE Signal Processing Magazine, 35(5), 41-49.

25. Nguyen, T. D., Marchal, S., Miettinen, M., Fereidooni, H., Asokan, N., & Sadeghi, A. R. (2019). DÏoT: A federated self-learning anomaly detection system for IoT. 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), 756-767.

26. Haddad Pajouh, H., Dehghantanha, A., Khayami, R., & Choo, K. K. R. (2018). A deep recurrent neural network based approach for internet of things malware threat hunting. Future Generation Computer Systems, 85, 88-96.

27. Ge, M., Fu, X., Syed, N., Baig, Z., Teo, G., & Robles-Kelly, A. (2019). Deep learning-based intrusion detection for IoT networks. 2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC), 256-25609.

28. Deng, L., Li, D., Yao, X., Cox, D., & Wang, H. (2019). Mobile network intrusion detection for IoT system based on transfer learning algorithm. Cluster Computing, 22(4), 9889-9904.

29. Alom, M. Z., & Taha, T. M. (2017). Network intrusion detection for cyber security using unsupervised deep learning approaches. 2017 IEEE National Aerospace and Electronics Conference (NAECON), 63-69.

30. Moustafa, N., Creech, G., & Slay, J. (2017). Big data analytics for intrusion detection system: Statistical decision-making using finite dirichlet mixture models. Data Analytics and Decision Support for Cybersecurity, 127-156.

31. Bostani, H., & Sheikhan, M. (2017). Hybrid of anomaly-based and specification-based IDS for Internet of things using unsupervised OPF based on MapReduce approach. Computer Communications, 98, 52-71.

32. Hodo, E., Bellekens, X., Hamilton, A., Tachtatzis, C., & Atkinson, R. (2017). Shallow and deep networks intrusion detection system: A taxonomy and survey. arXiv preprint arXiv:1701.02145.

33. Chaabouni, N., Mosbah, M., Zemmari, A., Sauvignac, C., & Faruki, P. (2019). Network intrusion detection for IoT security based on learning techniques. IEEE Communications Surveys & Tutorials, 21(3), 2671-2701.

34. Alazab, M., Huda, S., Abawajy, J., Islam, R., Yearwood, J., Venkatraman, S., & Broadhurst, R. (2014). A hybrid wrapper-filter approach for malware detection. Journal of Networks, 9(11), 2878-2891.

35. Moustafa, N., Hu, J., & Slay, J. (2019). A holistic review of Network Anomaly Detection Systems: A comprehensive survey. Journal of Network and Computer Applications, 128, 33-55.

36. Otoum, S., Kantarci, B., & Mouftah, H. T. (2017). Detection of known and unknown intrusive sensor behavior in critical applications. IEEE Sensors Letters, 1(5), 1-4.

37. Pamukov, M. E., & Poulkov, V. K. (2017). Multiple negative selection algorithm: Improving detection error rates in IoT intrusion detection systems. 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), 543-547.

38. Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). A deep learning approach to network intrusion detection. IEEE Transactions on Emerging Topics in Computational Intelligence, 2(1), 41-50.

39. Abeshu, A., & Chilamkurti, N. (2018). Deep Learning: The frontier for distributed attack detection in fog-to-things computing. IEEE Communications Magazine, 56(2), 169-175.

40. Doshi, R., Apthorpe, N., & Feamster, N. (2018). Machine learning DDoS detection for consumer internet of things devices. 2018 IEEE Security and Privacy Workshops (SPW), 29-35.

41. Hasan, M., Islam, M. M., Zarif, M. I. I., & Hashem, M. M. A. (2019). Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. Internet of Things, 7, 100059.

42. Pahl, M. O., & Aubet, F. X. (2018). All eyes on you: Distributed multi-dimensional IoT microservice anomaly detection. 2018 14th International Conference on Network and Service Management (CNSM), 72-80.

43. Otoum, S., Kantarci, B., & Mouftah, H. T. (2019). On the feasibility of deep learning in sensor network intrusion detection. IEEE Networking Letters, 1(2), 68-71.

44. Raza, S., Wallgren, L., & Voigt, T. (2013). SVELTE: Real-time intrusion detection in the Internet of Things. Ad Hoc Networks, 11(8), 2661-2674.

45. Sedjelmaci, H., Senouci, S. M., & Al-Bahri, M. (2016). A lightweight anomaly detection technique for low-resource IoT devices: A game-theoretic methodology. 2016 IEEE International Conference on Communications (ICC), 1-6.

46. Alharbi, S., Rodriguez, P., Maharaja, R., Iyer, P., Subaschandrabose, N., & Ye, Z. (2020). Secure the internet of things with machine learning: Opportunities and challenges. 2020 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 860-865.

47. Xiao, L., Wan, X., Lu, X., Zhang, Y., & Wu, D. (2018). IoT security techniques based on machine learning: How do IoT devices use AI to enhance security? IEEE Signal Processing Magazine, 35(5), 41-49.

48. Nguyen, T. D., Marchal, S., Miettinen, M., Fereidooni, H., Asokan, N., & Sadeghi, A. R. (2019). DÏoT: A federated self-learning anomaly detection system for IoT. 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), 756-767.

49. Haddad Pajouh, H., Dehghantanha, A., Khayami, R., & Choo, K. K. R. (2018). A deep recurrent neural network based approach for internet of things malware threat hunting. Future Generation Computer Systems, 85, 88-96.

50. Ge, M., Fu, X., Syed, N., Baig, Z., Teo, G., & Robles-Kelly, A. (2019). Deep learning-based intrusion detection for IoT networks. 2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC), 256-25609.

51. Deng, L., Li, D., Yao, X., Cox, D., & Wang, H. (2019). Mobile network intrusion detection for IoT system based on transfer learning algorithm. Cluster Computing, 22(4), 9889-9904.

52. Alom, M. Z., & Taha, T. M. (2017). Network intrusion detection for cyber security using unsupervised deep learning approaches. 2017 IEEE National Aerospace and Electronics Conference (NAECON), 63-69.

53. Moustafa, N., Creech, G., & Slay, J. (2017). Big data analytics for intrusion detection system: Statistical decision-making using finite dirichlet mixture models. Data Analytics and Decision Support for Cybersecurity, 127-156.

54. Bostani, H., & Sheikhan, M. (2017). Hybrid of anomaly-based and specification-based IDS for Internet of things using unsupervised OPF based on MapReduce approach. Computer Communications, 98, 52-71.

55. Hodo, E., Bellekens, X., Hamilton, A., Tachtatzis, C., & Atkinson, R. (2017). Shallow and deep networks intrusion detection system: A taxonomy and survey. arXiv preprint arXiv:1701.02145.

56. Chaabouni, N., Mosbah, M., Zemmari, A., Sauvignac, C., & Faruki, P. (2019). Network intrusion detection for IoT security based on learning techniques. IEEE Communications Surveys &