A Novel Network Intrusion Detection System Based on Semi-Supervised Approach for IoT

Durga Bhavani A[1], Dr. Neha Mangla[2]

Assistant Professor, Department of Computer Science and Engineering, BMS Institute of Technology and Management, Bangalore, India[1]

Associate Professor, Department of Information Science and Engineering, Atria Institute of Technology, Bangalore, India[2]

Abstract—**An intrusion detection system (IDS) is one of the most effective ways to secure a network and prevent unauthorized access and security attacks. But due to the lack of adequately labeled network traffic data, researchers have proposed several feature representations models over the past three years. However, these models do not account for feature generalization errors when learning semantic similarity from the data distribution and may degrade the performance of the predictive IDS model. In order to improve the capabilities of IDS in the era of Big Data, there is a constant need to extract the most important features from large-scale and balanced network traffic data. This paper proposes a semi-supervised IDS model that leverages the power of untrained autoencoders to learn latent feature representations from a distribution of input data samples. Further, distance function-based clustering is used to find more compact code vectors to capture the semantic similarity between learned feature sets to minimize reconstruction loss. The proposed scheme provides an optimal feature vector and reduces the dimensionality of features, reducing memory requirements significantly. Multiple test cases on the IoT dataset MQTTIOT2020 are conducted to demonstrate the potential of the proposed model. Supervised machine learning classifiers are implemented using a proposed feature representation mechanism and are compared with shallow classifiers. Finally, the comparative evaluation confirms the efficacy of the proposed model with low false positive rates, indicating that the proposed feature representation scheme positively impacts IDS performance.**

Keywords—IoT; security; intrusion detection system; semi-supervised; autoencoder; clustering; machine learning

I. Introduction

The rise of IoT has increased the number of connected devices, resulting in a growing need for effective security measures to prevent data breaches and cyber-attacks [1]. The MQTT (Message Queuing Telemetry Transport) protocol is widely adopted in IoT-enabled real-world applications such as smart homes, smart cities, smart industries, and smart healthcare due to its low bandwidth cost, low memory requirements, and minimal packet loss [2]. The MQTT protocol is based on a publish/subscribe mechanism and a central communication module, often called a broker, which is vulnerable to various security threats and attacks. In [3], researchers reported security risks related to the MQTT protocol and found over 53000 publicly accessible MQTT-based IoT devices.

In the existing literature, various security and prevention mechanisms based on cryptography and digital signature have been presented to deal with various attacks [4–5]. Cryptography is an effective solution for providing better security services by enabling source authentication mechanisms, but it is not robust against usability or availability attacks [6]. With the advent of machine learning (ML) techniques, more attention has been paid to studying intrusion and anomaly detection systems for IoT. Intrusion detection is an important aspect of IoT security which involves identifying abnormal behavior patterns and potential security threats in real-time. IDS and cryptographic-based approaches complement each other to ensure maximum security features such as confidentiality, integrity, authenticity, and availability [7–8]. Network IDS can be classified into signature-based and anomaly-based methods [9]. Signature-based IDS considers attack traffic patterns, while the anomaly-based approach usually considers normal traffic patterns. However, signature-based IDS only relies on known traffic patterns and faces limitations in detecting new sophisticated attacks whereas anomaly-based IDSs usually have a high false positive rate, which may require additional resources and time to eliminate the large number of alerts generated. Furthermore, existing IDSs are not well suited for MQTT-enabled IoT ecosystems, as they were designed with little attention to resource efficiency and scalability features [10].

Recently, many researchers have attempted to explore the ability of feature representations to handle large-scale traffic sampling and class imbalance in attack detection based on predictive modeling. In [11], an autoencoder-based evaluation framework is given to assess the effectiveness of various feature representation techniques for handling large-scale information streaming in text classification tasks. Similarly, researchers in [12] presented feature selection techniques based on unsupervised learning schemes to enhance data analytics tasks and computational intelligence performance on large datasets. Researchers in [13, 14, 15] reported enhancing the performance of real-time analytics operations on big data by using appropriate feature representation techniques. Furthermore, the adoption of autoencoders for feature representation is seen in many applications, such as the classification of anti-drug response [16] and diagnosis of autism spectrum disorder [17]. In [18], the authors applied the reported use of deep learning to feature learning to deal with big biological data for disease investigation.

Taking inspiration from the literature in which many researchers employ autoencoders for feature modeling, we are motivated to propose a new feature representation model for learning-driven IDS that can keep up with high network traffic and exhibit high accuracy in IoT intrusion detection.

This manuscript proposes a novel anomaly and intrusion detection solution in MQTT-driven IoT systems that combine unsupervised and supervised learning techniques. The proposed solution uses an autoencoder to extract robust features from normal traffic data, which are then clustered and assigned different cluster IDs based on their unique features. This information creates a supervised classifier to detect real-time anomalies and intrusions. The proposed solution offers several advantages over traditional security measures, including detecting unknown threats and adapting to changing threat landscapes. The scope of this research article is to provide an in-depth analysis of the proposed solution, including mathematical models and algorithms, as well as a discussion of the potential benefits and scope of real-world applications.

The remaining sections of this paper are organized as follows: Section II presents related work; Section III highlights the research problem explored and addressed in the proposed work; Section IV presents the system design and dataset details and preprocessing; the proposed feature representation mechanism is discussed in Section V, along with experimental analysis and result, discussion are carried to justify the scope of the proposed system, and finally, the entire work is concluded in Section VI.

II. Related Work

This section presents a brief review of existing literature on IoT security that uses machine learning concepts in IDS design and approaches adopted for feature enhancement towards better classification outcomes.

The work carried out by Kaur et al. [19] employed a Convolution Neural Network (CNN) to recognize and describe various security attacks. The authors have validated their attack detection model through CICIDS2017 and CICIDS2018 datasets. However, this study offers a multiclass classification feature that does not meet higher accuracy for detecting many attacks. Another study by Odetola et al. [20] also adopted the CNN model to design a multiclass classification system for the IoT network. The authors have utilized a single CNN with multiple layers and a customizable loss function, which achieves less delay in the attack detection process. The authors in the study of Haripriya and Kulothungan [21] designed an IDS based on the approach of a fuzzy logic system for IoT devices against DoS attacks. The fuzzy logic has been widely used in many applications to deal with ambiguity in the dataset. However, using the fuzzy logic approach in this study for IDS design does not show its effectiveness due to its high complexity with an increase in the input dimension. In the IoT ecosystem, massive data is forwarded continuously. Also, this study is quite specific for the particular type of attacks, and more advanced or unknown attacks have been left untouched. The work of Ciklabakkal et al. [22] utilizes multiple machine learning techniques along with the autoencoder to detect intrusions in the IoT network. However, the authors have not explicitly discussed the type of attacks they are considering. Faker and Dodge [23] employed a deep learning approach in IDS design and evaluated its performance on the two datasets, namely CIC-IDS2017 and UNSW-NB15. Alkadi et al. [24] used blockchain technology to ensure security and reliability in distributed IoT-IDS. In this study, Ethereum blockchain-based smart contracts are implemented to ensure privacy features, and Bi-Long Short-Term Memory (LSTM) is used to develop IDS that deal with sequential network flow. The effectiveness of the presented model is evaluated based on the UNSW-NB15 and BoT-IoT datasets. AI-enabled IDS is designed in the study of Fatani et al. [25]. The authors have addressed feature engineering issues using CNN and extracted relevant attributes in this study. Further, a transient search optimization mechanism is used to select the feature.

A game theory-based strategy is also adopted in the design of IDS. In the study of Wang et al. [26], the authors suggested an analytical framework to identify the circumstances under which malevolent devices have no motivations to perform the attack in the IDS attack-prevention game. The work of Basset et al. [27] presented a semi-supervised approach for IDS that considers sequential features of the traffic flow in the training phase. The authors have also focused on enhancing the learning performance of the system by tuning the system configuration during the generalization of spatiotemporal representations from the input dataset. A modified version of the attention mechanism helps the system focus on the significant attributes during the training process. Dutta et al. [28] suggested an anomaly detection scheme based on the sparse autoencoder, and an LSTM with logistic regression is further employed to detect malicious traffic. In Aygun et al. [29] study, a comparative analysis is carried out between two autoencoders for recognizing new or unknown attacks. Shone et al. [30] introduce an un-symmetric autoencoder system that offers better data generalization and dimensionality reduction. The work carried out by Tabassum et al. [31] suggested privacy-aware IDS. The preprocessing operations using autoencoder eliminate redundancies and present precise dataset modeling. The experimental analysis has shown the benefit of using an autoencoder in reducing the feature dimension and the computational complexity in the training phase. There are other recent works on IDS for detecting MQTT attacks in IoT using ML approaches highlighted in Table I.

TABLE I. Highlights on Exisitng IDS for MQTT Attacks

Authors	Approach	Dataset	Metric
[32]	RF, NB NDT, MLP	MQTTset	Accuracy and F1 Score
[33]	Tree Based Approach	Simulation	Detection rate
[34]	NB, LR, KNN,	MQTT-IoT-IDS2020	Accuracy, Recall, F1-score, Precision
[35]	XGBoost, LSTM, GRU	MQTT-IoT-IDS2020	F-beta score, Accuracy, Log loss
[36]	DNN	MQTT-IoT-IDS2020	Accuracy, Precision, Recall, F1-score
[37]	Transfer learning	MQTT-IoT-IDS2020 and IoT-23	Accuracy, Precision Recall, F1-score

The existing literature has widely adopted machine learning techniques in IoT intrusion detection. However, previous solutions have some shortcomings, which should be handled efficiently. The findings reveal that most of the existing IDS schemes are specific to a particular adversarial scenario and may not be adaptive to detect unknown attacks and multi-class intrusions. It has also been noticed that majority of the recent works on IDS lack focus on handling class imbalance factor as most network dataset is imbalanced. Such schemes may be subjected to a higher false-positive rate in the intrusion classification task. There is also not much emphasis given by researchers on feature enhancement, and optimization in computational resources especially when the dataset is quite massive, like MQTT-IoT-IDS2020 (3.6 GB) and IoT-23 dataset (i.e., 21 GB). Therefore, the scope of such existing systems is limited and cannot be applied to large and dynamic networking scenarios.

The problem statement for the proposed work can be stated as "*It is challenging to design a feature representation mechanism that can offer a higher degree of optimization in the training samples, allowing the learning model to discover the precise pattern required for the detection of attacks with minimal false-positive and high accuracy rate*. "

III. Proposed Solution

The proposed approach is based on a combination of autoencoder-based anomaly detection and supervised classifier-based multiclass intrusion detection. The proposed approach consists of the following steps:

- Training the autoencoder with normal traffic data to learn a compressed representation of the data.
- Using the Gini measure to compute a threshold for assigning different cluster IDs to different classes of intrusions.
- Using the autoencoder to map the network traffic into the learned compressed representation.
- Introducing the Gini-based threshold with an ML classifier to perform multiclass intrusion detection.

The autoencoder is trained with normal traffic data to learn a compressed representation of the data, which can be used to detect deviations from normal behavior. The Gini measure is then used to compute a threshold for assigning different cluster IDs to different classes of intrusions. The autoencoder is used to map the network traffic into the learned compressed representation, and the Gini-based threshold is introduced with an ML classifier to perform multiclass intrusion detection. The first step is to train an autoencoder using normal MQTT message data. This autoencoder will learn the patterns of normal behavior and create a compressed representation of these patterns that can be used as a feature set for the intrusion detection system. Next, a supervised classifier is trained using a multiclass classification approach. The classifier is trained using labeled MQTT messages that correspond to different types of intrusions. The classification is performed on the compressed feature set produced by the autoencoder. To assign different cluster IDs to different classes of intrusions, a threshold is computed using a Gini measure. The threshold is used to assign different cluster IDs to different classes of intrusions. This will enable the classifier to accurately identify and classify different types of intrusions. Once the classifier is trained, it can be used to monitor MQTT traffic in real-time. Any anomalous activity or intrusion that does not match the patterns of normal behavior will be detected by the autoencoder and flagged as a potential threat. The feature set produced by the autoencoder is then used as input to the classifier, which can classify the intrusion and take appropriate action. By combining an autoencoder-based anomaly detection system with a supervised classifier for multiclass intrusion detection, this solution can effectively detect and classify security threats in MQTT-driven IoT networks. This approach provides early detection of security threats, reduces false positives, and improves accuracy, making it a powerful tool for protecting IoT networks against attacks. In this work, the study deals with security on all three layers: the network layer, the transport layer (TCP and UDP analysis), and the application layer. The proposed model is based on the semi-supervised approach, where a combination of an autoencoder and supervised learning techniques is used to perform feature modeling and intrusion detection.

IV. Mathematical Model and Algorithm

This section discusses the implementation procedure adopted in designing feature learning-based IDS.

The mathematical model includes training the autoencoder to learn the compressed representation of normal behavior, training a supervised classifier using the compressed feature set as input, computing a threshold value using a Gini measure, and using the classifier and threshold to detect and classify intrusions in real-time. Let $X = \{x1, x2, \ldots, xn\}$ be the input dataset consisting of n-dimensional feature vectors.

- Autoencoder: $Let\ AE(X) = \{y1, y2, \ldots, yn\}$ be the compressed feature set obtained by passing X through the autoencoder.
- Computing Distance: Let $d(x, y)$ be the distance function that measures the distance between the input vector x and its corresponding output vector y obtained from the autoencoder. $Let\ D = \{d(x1, y1), d(x2, y2), \ldots, d(xn, yn)\}$ be the set of distances between the input and output vectors.
- Computing Thresholds: $Let\ T = \{T1, T2, T3, T4\}$ be the set of threshold values that divide the distance values into five clusters, where the first four clusters represent the different categories of attacks and the last cluster represents normal behavior. The threshold values are computed using the Gini index, which finds the optimal split of the distances into the five clusters.
- Cluster ID Assignment: $Let\ C_{id} = \{c1, c2, \ldots, cn\}$ be the set of cluster ID assignments for the input vectors, where ci belongs to $\{0,1,2,3,4\}$. The cluster ID assignments are determined by comparing the distance values with the threshold values and assigning the corresponding cluster ID.

- Training Supervised Classifier: Let us consider $F = \{f1, f2, \ldots, fm\}$ be the set of training features obtained by taking the mean of the compressed feature set $AE(X)$ for each cluster ID. Let $Y = \{y1, y2, \ldots, yn\}$ be the set of class labels for the input vectors. Let clf be the supervised classifier, which takes F as input and learns to classify the input vectors based on their features. The classifier is trained using a labeled dataset that includes the input vectors X and their corresponding class labels Y. Once the classifier is trained, it can be used to classify new input vectors based on their features.

A. *Training Autoencoder*

Let X be a matrix of MQTT message data, such that $X \in \{X_1, X_2, X_3 \cdots X_n\}$ where n denotes number of data samples. Here, each row represents a single message, and each column represents a feature. The autoencoder is trained to learn the compressed representation of normal behavior, such that $X \approx X'$, where X' is the reconstructed matrix of X obtained by encoder module expressed as follows:

$$Z = f(X, \theta) \tag{1}$$

Where, Z is encoding function produces compressed feature set X' of input X and θ represents the learnable parameters of the autoencoder given as follows:

$$\theta = W_i + b \tag{2}$$

Algorithm 1: Training Autoencoder

Input: X (Training Dataset), E (Training Epoch)
Output: X' & Re (reconstruction error)
Start

```
1. Init W (weights)
2. $for\ each\ E$ do
3.   $for\ each\ mini\ batch$ do
4.     Encode $X_i \rightarrow Z_i : f(X_i, \theta)$ // feature representation
5.     Decode $Z_i \rightarrow X_i' : f(Z_i, \theta)$ // input reconstruction
6.     Tune hyperparameter using GST (grid search technique)
7.   end for
8. end for
9. Compute Re $\leftarrow f_1(X_i, X_i')$
     // Re is the mean reconstruction error on training data
```

End

The autoencoder is trained with normal traffic logs (X_i) in an unsupervised manner, i.e., without labels, and creates a compact representation of the input samples, which is a reconstructed version (X_i') of the normal traffic log pattern as closely as possible. However, if the given input data actually belongs to the attack category, the reconstructed data can be different from the original input data. This means that the category of the network traffic data is determined according to the difference between the input and its output. It is considered an attack if the difference is greater than the set threshold or cut-off value. In this regard, the study computes a mean reconstruction error (Re) using the function $f_1(\,)$ with an input argument X_i and X_i'. Basically, this function computes the square of the minimum error between the input data and the reconstructed data given as follows:

$$\text{Re} = \frac{1}{n}\sum_{i=1}^{n}(x_i - x_i')^2 \tag{3}$$

Where, n denotes the total number of data samples in the training samples, and the computed Re is obtained is 0.001.

B. *Threshold Computation*

To assign different cluster IDs to different classes of intrusions, a threshold is computed using a Gini measure. The Gini measure is used to identify the feature with the highest discriminatory power between classes. This feature is used to compute the threshold value, which is used to assign different cluster IDs to different classes of intrusions (refer Table II). For example, Let us consider d be the number of features in the compressed feature set Z. The threshold is computed using the following formula:

$$T = (max(Z[:, i]) - min(Z[:, i]))/2 + min(Z[:, i]) \tag{4}$$

$$i = argmax(Gini(Z[:, j])) \tag{5}$$

$$j = 1, \ldots, d \tag{6}$$

where $Z[:, i]$ represents the i-th feature of the compressed feature set Z, and $Gini(Z[:, j])$ represents the Gini measure for the j-th feature of Z.

In the proposed modeling the Gini index is a measure of the impurity or diversity of a set of values. In the context of intrusion detection, the Gini index can be used to determine the threshold value for the anomaly score function g. Let D be the set of all instances in the dataset X. Let C be the set of possible cluster labels, and let D_c be the subset of instances in D that belong to cluster c. The Gini index for a given threshold value T is defined as:

$$G(T) = \sum\{c \in C\} p_c(T)\,(1 - p_c(T)) \tag{7}$$

where $p_c(T)$ is the fraction of instances in D_c that have an anomaly score greater than T, such that:

$$p_c(T) = |\{x_i \in D_c \mid g(x_i) > T\}| \,/\, |D_c| \tag{8}$$

Algorithm 3: GINI Measure Based Threshold setting (T_i)

Input: D (Distance values), X' (Sample outputs)
Output: T_i {T1, T2, T3, T4}
Start

```
1. Load sample outputs 10% of $X'$
2. Get unique values: U = $f_2$(X')
3. for each i in U do
4.   Initialize ginis as G
5.     G → [ ] // empty vector
6.     for d in range min(D) to max(D) do
7.       K = Y, where D<d
8.       $\lambda = 1 - \sum(\text{P}(\text{K} == \text{I}))^2$
9.       Append λ to G
10.    end for
11. Ti = d where G is maximum in ginis
12. end for
```

End

The algorithm 3 consists an input variable D (Distance values), and X (Sample outputs). After execution it returns an output $T_i \in \{T1, T2, T3, T4\}$. The first step of the algorithm loads a random subset of the sample outputs, X'. Afterwards, it gets the unique output values from X': $U = \{u_1, u_2, \ldots, u_n\}$. For each u_i in U, the algorithm performs initialization of a vector subjected to Gini indices, $G = [\,]$. Then, for each distance value d in the range $[min(D), max(D)]$, the algorithm computes set of K outputs from X' that have a distance less than d from u_i. Next it computes the probability $P(K == j)$ that a sample output in K belongs to each class j in the set of possible classes. Afterwards the computation of the Gini index λ is carried out as $1 - \sum\{j = 1\}^m \left(P(K == j)\right)^2$, where m is the number of possible classes. Finally, the computed λ is appended to vector G. Based on the previous outcome the algorithm set T_i to d, where d is the value of D that maximizes the Gini index in G. After successful execution of the all steps, the algorithm returns an output of the threshold values T1, T2, T3, and T4 indicating distinct values for assigning clusters for each specific intrusion classes. The Gini index is used as a measure of how well a particular distance value separates the normal class from the different attack classes. The threshold values T1, T2, T3, and T4 are used to divide the range of possible distances into five intervals, which are then mapped to the five output classes (Normal, DoS, Probe, R2L, and U2R) for use in the supervised classifier.

C. Cluster ID Assignment

The assignment of the cluster ID is done based on the threshold value Ti obtained from the algorithm 2 and distance vector D computed using Eq. (9).

Algorithm 3: Cluster ID Assignment

Input: AuE (Trained Autoencoder), X_{Ts} (Testing Dataset),
Output: C_Id (cluster id)
Start

```
1.  Call Algorithm-1
2.  Phase: Model Testing
3.    Test model with entire text data → $X_{Ts}$
4.     Compute D using Eq. (5)
5.         //D← distance between original input and output
6.  Phase: Clustering
7.  for each data class do
8.    Compute Ti // where i = {1 , 2 , 3, 4}
9.      Call Algorithm-2
10. end for
11. Phase: C_id assignment
12. Check:
13.    $if\ D < T1$ do
14.     C_id →0 // Nor
15.       $elif\ D < T2$ do
16.        C_id →1 // MBA
17.       $elif\ D < T3$ do
18.         C_id →2 // SCA
19.        $elif\ D < T3$
20.          C_id →3 // SPA
21.    else
22.     C_id →4 // SUA
```

End

Algorithm 3 comprises three phases for implementing the proposed feature representation scheme. i) distance computation, ii) threshold computation, and iii) cluster id assignment. The primary step algorithm employs the computing operation and functions discussed in the Algorithm 1, which is about training autoencoder model. Further, the algorithm computes the distance vector using a distance function that uses the Euclidian distance formula expressed as follows:

$$\vec{D} = \sqrt{\sum_{i=1}^{n} |x_i - x_i'|^2} \qquad (9)$$

Where vector D consists value of the distance computed between input data (x_i) and reconstructed data (x_i'), where $x_i \in X_{Ts}$ and $x_i' \in X_{Ts}'$.

The subsequent steps of the algorithm are subjected to clustering operations. In this phase, the algorithm uses threshold vector (T_i) obtained from the Algorithm 2. This vector consists of different threshold values specific to each data class. Next, the threshold values contained in T_i are compared with distance value based on which cluster-id (C_id) is assigned to groups containing unique features specific to normal class attack classes (algorithmic steps: 14 to 22).

TABLE II. GINI VALUES AND THRESHOLD OF EVERY CLASS

Label	Attack Labels/Class	Gini	Threshold
0	Norm	0.9214	0.0116
1	MBA	0.9142	0.0592
2	SCA	0.9321	1.0753
3	SUA	0.9154	1.2643
4	SPA	0.9431	1.6823

D. Intrusion Detection

The proposed research work's primary aim is to evolve a novel intrusion detection system (IDS) to detect and forecast cyber-attacks with high detection accuracy and to recover the IoT application from the attack at the earliest by activating a suitable response. The intrusion detection problem in the proposed work is considered to be a multiclass classification task using a supervised ML classifier. The proposed study implements three different shallow ML classifiers i) Naïve Bayes (NB), ii) K-Nearest Neighbor (KNN), and iii) Random Forest (RF). These classifiers take cluster id as input and return observed data as normal and attack classes. In order to train the classifier, 50% of the dataset is considered, and 50% dataset is considered for model evaluation. The reason behind this is that the dataset set is quite huge and big consists of approximately more than 30 million of traffic samples; taking more than 70% of dataset in processing often encounter computing resource exhaustion error. To address this kind of problem only the proposed research work has introduced an efficient feature representation technique which does not need to be introduced with 100% of the dataset, taking few samples it can provided sufficient feature code for training machine learning classifiers. The uniqueness of the proposed work lies in the blend of 3-

layer autoencoders that enhance feature generalization capability of any learning model independent of large traffic samples and without imposing memory overhead problem.

V. Implementation and Result

The design and development of the proposed system is carried out using python programming language scripted in Jupyter notebook on an anaconda environment.

A. System Design and Implementation

The proposed model is based on the semi-supervised approach, where a combination of an autoencoder and supervised learning techniques is used to perform feature modeling and intrusion detection. The first module of the proposed system focuses on data preprocessing, which is meant to handle null and missing values and data encoding. The second module of the proposed system emphasizes feature modeling and representation using autoencoder and distance function. The third module of the proposed system is about implementing machine learning classifiers for multiclass intrusion detection and classification. Fig. 1 illustrates the architectural design of the proposed system followed with methodology discussed in the above section. In this work, the study deals with security on all three layers: the network layer, the transport layer (TCP and UDP analysis), and the application layer.

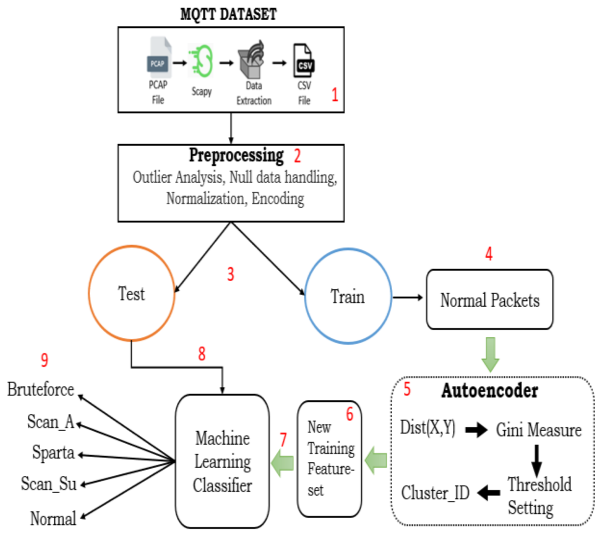

Fig. 1. Illustartion of methodology adopted in system design.

B. Dataset Used

The MQTTIoT-IDS-2020 dataset is adopted in this research work downloaded from the IEEE data port using the link [38]. The attack simulation setup mimics realistic 12 MQTT sensors in a network, consisting of a central communication module (broker), a camera feed server, and an attacker. This network setup consists of both generic networking scanning and MQTT brute-force attack. The structure of an MQTT packet looks like as shown in Fig. 2.

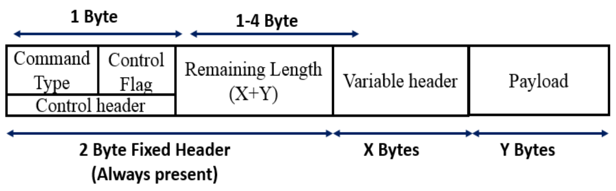

Fig. 2. Visualization of MQTT packet structure.

The dataset is downloaded in PCAP data format created using a software called Wireshark. There is an excellent utility in python called Scapy, which can be used for packet manipulation and reading PCAP files in python. The same utility is used to convert dataset into .csv file format. It has been analyzed there are total five classes such as Normal, MQTT_Bruteforce, Scan_A, Scan_sU, and Sparta. In a 'Normal' there is no attack; it is just a normal operation in the network. The 'MQTT_Bruteforce' is an attack file where the attacker tries to get the MQTT broker's password by trying all combinations of letters. 'Scan_A' is an aggressive scan attack where an attacker looks for all IOT nodes on the network. 'Scan_sU' is a UDP scan looking for all CCTV cameras on the network, and 'Sparta' means SSH brute force trying to gain access to the main server. The distribution of each packet classes is illustrated in Fig. 3.

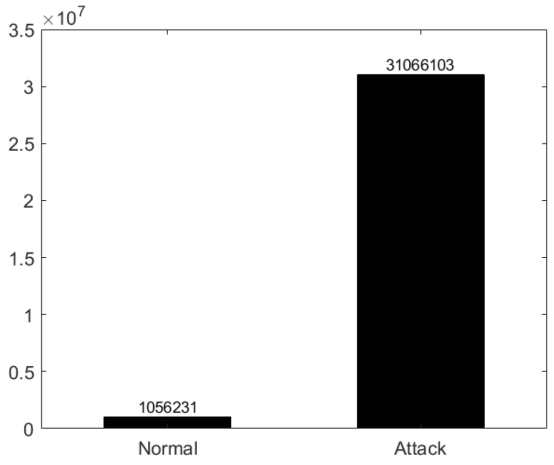

Fig. 3. Statistics of data distribution.

The graph trend exhibits that there is a total of 1026231 normal samples (Nor) and a total of 31066103 attack samples in the dataset. This analysis also reveals that the dataset is extensive and quite imbalanced, as more than 90% of the samples are subjected to the attack class.

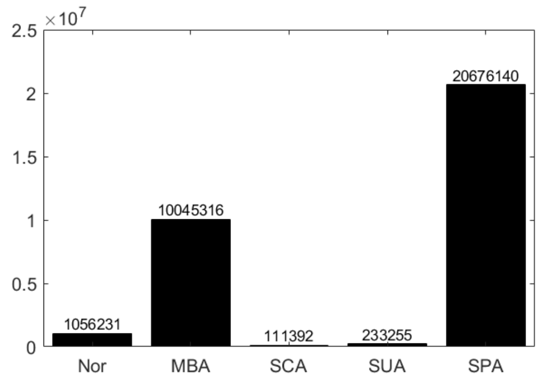

Fig. 4. Statistics of the attack class distribution.

Fig. 4 shows the distribution of attack classes. Sparta (SPA) and MQTT Bruteforce (MBA) attacks have a significantly higher number of samples compared to other attack classes such as scan_UDP (SUA) and scan_aggressive (SCA). From the exploratory analysis, it is clear that this dataset is associated with class imbalance problems and cannot be directly introduced to the learning model; otherwise, the identification or classification results tend toward majority classes, i.e., SPA and MBA.

Further a basic preprocessing operation is executed where the index of each data sample is reset, and an extra column is added for the label representing 0 for normal class and 1 for attack classes. The data frame consists of 33 columns representing feature descriptors and 32092334 rows representing the total number of samples belonging to each feature descriptor. Hence, the dataset is big and consumes 8 GB of memory on the local hard disk. Therefore, the study puts effort into optimizing the memory requirement to hold such a large amount of network traffic data for ease of computing tasks. In this regard, irrelevant feature descriptor such as index, timestamp, and src_port is dropped. The index column has no importance in the classification problem, and the timestamp is not considered because the study does not process data as time-series samples.

On the other hand, src_port is a random number that can also be ignored. As a result, data retrieval can be speedup and memory can be optimized to counter the curse of dimensionality issues. Apart from this, the data preprocessing includes outlier analysis, removal of missing and NAN values, filtering out irrelevant feature descriptors, and performing normalization and data encoding operations for the categorical data types. The study also splits predictor and response data which is then stored in the separate variables x and y. Also, it has been noticed that that in x data, there are three categorical values such as src_ip, dst_ip, and protocol. The src_ip giveaway the attack as normal since there is only one malicious computer in the network. Therefore, this feature is dropped. Similarly, the dst_ip is irrelevant for attack prediction since it does not contain information about whether the packet is an attack or normal. Therefore, this will be dropped for further analysis. The protocol feature is relevant and needs encoding. Since the study adopts an autoencoder for feature modeling, nominal encoding is preferred where protocol names are replaced with a random number. The remaining feature descriptors need no encoding, and all Nan values are filled with zero to avoid computational error during the autoencoder training.

C. Performance Indicators

The performance analysis is carried out based on classification metrics and comparative assessment is done to analyze the effectiveness of proposed feature representation framework from the multiple dimensions. The performance indicators used are briefly discussed as follows:

$$Precision = \frac{1}{n}\sum\frac{TP_i}{TP_i+FP_i} \tag{10}$$

$$Recall\ rate = \frac{1}{n}\sum\frac{TP_i}{TP_i+FN_i} \tag{11}$$

$$F1\_Score = \frac{2(Precision*Recall)}{Precision+Recall} \tag{12}$$

Where i denotes the number of classes, TP denotes the true positive metric that shows attack packets are classified correctly, TN denotes the true negative metric that shows the normal traffic samples are correctly classified as normal, FP denotes the false-positive metric that shows normal traffic samples are misclassified as attack samples, and FN denotes false-negative metric that shows attack samples are misclassified as normal samples.

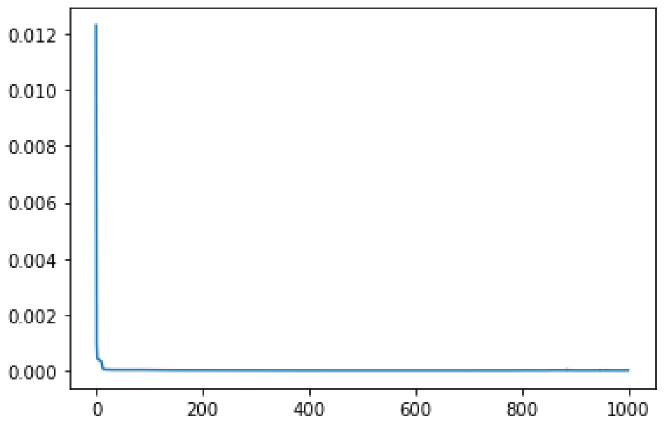

Fig. 5. Loss curve of trained autoencoder.

Fig. 5 presents the autoencoder's loss curve for analyzing its training performance. A closer analysis shows that a smooth curve has been obtained at each epoch, indicating a lower loss in the training process; hence a better performing autoencoder.

D. Comparative Analysis

The section presents outcome and performance analysis to demonstrate how the proposed feature representation technique enhances the IDS performance. In this regard, the performance evaluation is done by implementing ML classifiers with proposed feature representation model (i.e., WFR) and without feature representation model (i.e., WoFR). Since the proposed study focuses on multiclass intrusion detection, the precision, recall, and F1 score are computed for each data class. In addition, considering three ML classifiers demonstrates classification performance from the perspective of extensive analysis and against class imbalance problems.

1) Analysis of NB classifier: In this section the performance of proposed method will be evaluated with NB classifier (WFR) considering its comparison where the dataset is directly introduced with NB classifier (WoFR). The outcome obtained is shown in Table III.

TABLE III. NUMERICAL OUTCOME FOR NB CLASSIFIER (%)

	NB With Feature Representation (WFR)			NB Without Feature Representation (WoFR)		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
Norm	96.55	98.88	97.70	91.53	91.45	91.49
MBA	98.47	97.28	97.87	93.36	90.23	91.77
SCA	15.29	97.03	26.42	9.45	90.32	17.12
SUA	9.18	97.48	16.78	0	90.23	0
SPA	99.55	97.27	98.40	94.45	90.34	92.35

Table III shows the numerical outcome for the NB classifier for both case scenarios, i.e., WFR and WoFR. In the case of WFR, the NB exhibits a precision score of 96.55 %, recall rate of 98.88%, and F1 score of 97.70% for normal (Norm) class prediction. In the case of WoFR, the NB classifier shows a precision score of 91.53 %, recall rate of 91.45%, and F1 score 91.49%. Similar analysis can also be seen for the MBA class, where the NB classifier has achieved higher performance in the case of WFR. However, in predicting SCA and SUA, the NB classifiers do not show good performance. A closer analysis of outcome reveals the proposed system can efficiently enhance the performance of learning driven IDS without posing higher false positive rate unlike case of WoFR. This is because there are a smaller number of labels, and the dataset is associated with an imbalance factor. In the same way, the NB classifier also suffers in the classification of SUA. Although in the case of WFR, it has shown a 9.18% of precision score, whereas in case of WoFR, the NB could not predict SUA attack, i.e., it has achieved 0% precision rate.

The reason behind this that, SUA exhibits similar pattern to the normal class, and NB has no better generalization in this context. For SPA classification, NB classifier has shown better performance in case of WFR than a generic case, i.e., WoFR. A closer analysis reveals that the performance of IDS builds using NB classifier can be enhanced with the proposed method (WFR).

2) Analysis of KNN classifier: A closer analysis of the outcome from Table IV reveals that the KNN classifier shows better performance in the case of WFR than WoFR. However, at the same time, it can also be seen that KNN suffers from class imbalance factors for detecting both SCA and SUA classes.

TABLE IV. NUMERICAL OUTCOME FOR KNN CLASSIFIER (%)

	KNN With Feature Representation (WFR)			KNN Without Feature Representation (WoFR)		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
Norm	97.73	99.29	98.50	90.33	99.32	94.61
MBA	99.03	98.25	98.64	92.23	91.23	91.73
SCA	22.11	98.15	36.10	16.34	91.34	27.72
SUA	13.91	98.32	24.38	11.43	91.34	20.32
SPA	99.72	98.26	98.98	92.43	91.45	91.94

In the case of WFR, the KNN classifier performs better than the NB classifier. Also, for SPA attack, KNN performs very well, exhibiting 99.72%, 98.26, and 92.43% precision score, recall rate, and F1 score, respectively. On the other hand, KNN in case of WoFR, shows 92.43%, 91.45%, and 91.94% of precision, recall, and F1 scores, respectively. Moreover, the worst thing is that the SCA and SUA exhibits very similar pattern to the normal (Norm) class due to which the model did not reveal distinct pattern in the training process. Therefore, in the case of SUA and SCA detection, the KNN is not able to generalize and distinguish distribution SUA pattern as it faces difficulty in predicting the correct value of K because of closer similarity among the feature of normal and SUA. Although, simulation outcome shows that the proposed method enhances the performance of KNN for the multiclass intrusion detection even the dataset is imbalanced. But still there is a scope for the improvement for handling class imbalance problem, which will be done in our future work.

3) Analysis of RF classifier: The quantified outcome from Table V demonstrates that in the case of WFR, RF has achieved a precision score of 99.92 %, recall rate of 99.97%, and F1 score of 99.95%. Whereas, in the case of WoFR, RF classifier shows a precision score of 89.21 %, recall rate 89.63%, and F1 score 89.42%. For classification of a normal class, it can be analyzed that RF with proposed method (WFR) has achieved very good performance than other classifiers such as NB and KNN.

TABLE V. NUMERICAL OUTCOME FOR RF CLASSIFIER (%)

	RF With Feature Representation (WFR)			RF Without Feature Representation (WoFR)		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
Norm	99.92	99.97	99.95	89.21	89.63	89.42
MBA	99.97	99.94	99.95	89.22	90.32	89.77
SCA	89.57	100	94.5	87.34	90.23	88.76
SUA	82.63	99.87	90.44	82.43	90.23	86.15
SPA	99.99	99.94	99.96	92.24	90.34	91.28

In the case of WFR, the RF classifier for the SCA classification shows an 89% precision score, which is better than NB (15.29%) and KNN (22.11%). RF classifier also achieved a 100% recall rate where NB and KNN showed 97.03% and 98.15 %, respectively. With the proposed method, RF classifiers correctly identify normal packets and classify different attack classes. Hence, based on the above inferencing and discussion, it is concluded that the proposed work of feature representation can significantly enhance the outcome of any prediction system for designing enhanced IDS.

E. Discussion and Implication

The proposed work aims to enhance the performance of data driven or learning based IDS for the accurate detection and classification of attacks in dynamic networks such as IoT. The proposed study has considered the case study of MQTT enabled IoT networks because MQTT-based attacks are more dynamic and complex as they can easily mimic benign or normal behavior. In such cases, precise identification of attacks or intrusion becomes very difficult and challenging for any kind of IDS. Therefore, the proposed study has presented a feature representation framework that helps machine learning models better generalize the latent features and learn the distribution of data features more precisely. Three supervised classifiers have been implemented and trained with features enhanced by the proposed scheme (WFR). Their outcome is then compared with the classifiers trained with normal data i.e., without applying the proposed method (WoFR). The comparative assessment shows that the outcome achieved using the proposed feature representation (FR) is promising. The classifier trained with data that are enhanced by the proposed scheme WFR is superior. However, among three classifiers RF has achieved superior performance in both the cases of WFR and WoFR. This can be also evident in Fig. 6, where comparative analysis of each technique is given in terms of F1_score.

The RF is based on the design principle of the decision tree (DT) algorithm, which adopts similar principle of using din DT to determine how the features of a dataset should split nodes to form the tree. The KNN is based on Euclidian distancing; better performance is expected for the labels with lower support values. The NB classifier is a probabilistic model based on the Bayes' Theorem that has a specific assumption that the data is a particular feature of data not related to any other form of features. Therefore, this assumption of the NB classifier makes it incapable in the case of SUA attack classification as SCA attack that mimics a similar pattern of normal class. Since the dataset is associated with an imbalance factor and, KNN and NB suffers in classification and do not provide a good result.

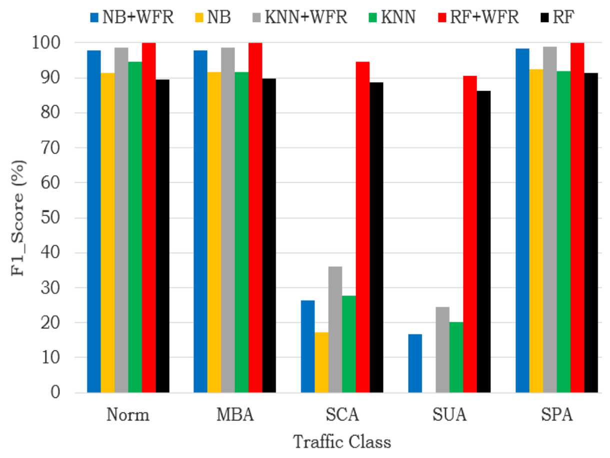

Fig. 6. Comparative analysis in terms of F1_score.

It is to be noted that the extensive outcome is not conducted due to the lack of sufficient and relevant research works on the similar dataset adopted in the proposed study. To date, the dataset has only 49 citations evident from the google scholar. However, in future work, the study considers handling data imbalance factors to strengthen the classification performance, making it more suitable to fit the real-time scenario. In future work, the proposed technique will be extended with the deep learning approach by calibrating it to generalize what to learn and what not to learn.

VI. CONCLUSION

The proposed research work deals with the intrusion detection system in the IoT ecosystem against MQTT attacks. The main contribution of the proposed study is a novel autoencoder-based clustering method that ultimately assigns a cluster-id to every packet. The proposed work offers an advanced data treatment method for critical feature modeling using an artificial neural network. The study also explores the effectiveness of different machine learning algorithms evaluated with standard and new datasets preprocessed via the proposed autoencoder scheme. The outcome statistics reveal the efficacy of the proposed feature representation technique, and comparative assessment justifies its scope in the real-time application. In future work, the proposed preprocessing scheme will be introduced with more advanced methods such as deep learning with a novel regularizer scheme to deal with dataset imbalance factors and achieve reliability in the classification process.

REFERENCES

[1] E. Al-Masri et al., "Investigating Messaging Protocols for the Internet of Things (IoT)," in IEEE Access, vol. 8, pp. 94880-94911, 2020, doi: 10.1109/ACCESS.2020.2993363.

[2] R. F. Al-Mutawa and F. Albouraey, "A smart home system based on internet of things," Int. J. Adv. Comput. Sci. Appl., vol. 11, no. 2, 2020.

[3] M. S. Harsha, B. M. Bhavani, and K. R. Kundhavai, “Analysis of vulnerabilities in MQTT security using Shodan API and implementation of its countermeasures via authentication and ACLs,” in 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2018.

[4] S. K. Mousavi, A. Ghaffari, S. Besharat, and H. Afshari, “Security of internet of things based on cryptographic algorithms: a survey,” Wirel. netw., vol. 27, no. 2, pp. 1515–1555, 2021.

[5] J. M. Carracedo et al., "Cryptography for Security in IoT," 2018 Fifth International Conference on Internet of Things: Systems, Management and Security, 2018, pp. 23-30, doi: 10.1109/IoTSMS.2018.8554634.

[6] J. Jang-Jaccard and S. Nepal, “A survey of emerging threats in cybersecurity,” J. Comput. Syst. Sci., vol. 80, no. 5, pp. 973–993, 2014.

[7] F. Alfaleh and S. Elkhediri, “Efficient Security Solutions for IoT Devices” International Journal of Advanced Computer Science and Applications(IJACSA),2021.

[8] M. Gurunathan and Moamin, “A review and development methodology of a LightWeight security model for IoT-based smart devices,” Int. J. Adv. Comput. Sci. Appl., vol. 11, no. 2, 2020.

[9] A. B. Mohamed, N. B. Idris, and B. Shanmugum, “A brief introduction to intrusion detection system,” in Communications in Computer and Information Science, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 263–271.

[10] Thakkar A, Lohiya R. A review on machine learning and deep learning perspectives of IDS for IoT: recent updates, security issues, and challenges. Archives of Computational Methods in Engineering. 2021 Jun;28(4):3211-43.

[11] S. Wang, J. Cai, Q. Lin and W. Guo, “An overview of unsupervised deep feature representation for text categorization,” IEEE Transactions on Computational Social Systems, vol. 6, no. 3, pp. 504–517, 2019.

[12] C. Tang, M. Bian, X. Liu, M. Li, H. Zhou et al., “Unsupervised feature selection via latent representation learning and manifold regularization,” Neural Networks, vol. 117, no. 9, pp. 163–178, 2019.

[13] Z. Huang, X. Xu, J. Ni, H. Zhu and C. Wang, “Multimodal representation learning for recommendation in internet of things,” IEEE Internet of Things Journal, vol. 6, no. 6, pp. 10675–10685, 2019.

[14] O. Aydogdu and M. Ekinci, “A new approach for data stream classification: Unsupervised feature representational online sequential extreme learning machine,” Multimedia Tools and Applications, vol. 79, no. 37–38, pp. 1–23, 2020.

[15] N. Wang, W. Zhou, Y. Song, C. Ma, W. Liu et al., “Unsupervised deep representation learning for real-time tracking,” arXiv preprint arXiv: 2007.11984, 2020.

[16] X. Xu, H. Gu, Y. Wang, J. Wang, and P. Qin, “Autoencoder based feature selection method for classification of anticancer drug response,” Front. Genet., vol. 10, p. 233, 2019.

[17] H. Sewani and R. Kashef, “An autoencoder-based deep learning classifier for efficient diagnosis of autism,” Children (Basel), vol. 7, no. 10, 2020.

[18] X. Qiang, C. Zhou, X. Ye, P. Du, R. Su et al., “CPPred-FL: A sequence-based predictor for large-scale identification of cell-penetrating peptides by feature representation learning,” Briefings in Bioinformatics, vol. 21, pp. 11–23, 2020.

[19] G. Kaur, A. H. Lashkari, and A. Rahali, "Intrusion traffic detection and characterization using deep image learning," in Proc. IEEE Intl Conf Dependable, Autonomic Secure Comput., Int. Conf Pervas. Intell. Comput., Intl Conf Cloud Big Data Comput., Int. Conf Cyber Sci. Technol. Congr. (DASC/PiCom/CBDCom/CyberSciTech), Aug. 2020, pp. 55–62.

[20] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," J. Inf. Secur. Appl., vol. 50, Feb. 2020, Art. no. 102419.

[21] Haripriya, A.; Kulothungan, K. Secure-MQTT: An efficient fuzzy logic-based approach to detect DoS attack in MQTT protocol for internet of things. EURASIP J. Wirel. Commun. Netw. 2019, 2019, 90.

[22] Ciklabakkal, E.; Donmez, A.; Erdemir, M.; Suren, E.; Yilmaz, M.K.; Angin, P. ARTEMIS: An intrusion detection system for MQTT attacks in Internet of Things. In Proceedings of the 2019 38th Symposium on Reliable Distributed Systems (SRDS), Lyon, France, 1–4 October 2019; pp. 369–3692.

[23] Faker, O.; Dogdu, E. Intrusion detection using big data and deep learning techniques. In Proceedings of the 2019 ACM Southeast Conference, Kennesaw, GA, USA, 18–20 April 2019; pp. 86–93.

[24] O. Alkadi, N. Moustafa, B. Turnbull and K. -K. R. Choo, "A Deep Blockchain Framework-Enabled Collaborative Intrusion Detection for Protecting IoT and Cloud Networks," in IEEE Internet of Things Journal, vol. 8, no. 12, pp. 9463-9472, 15 June15, 2021

[25] A. Fatani, M. Abd Elaziz, A. Dahou, M. A. A. Al-Qaness and S. Lu, "IoT Intrusion Detection System Using Deep Learning and Enhanced Transient Search Optimization," in IEEE Access, vol. 9, pp. 123448-123464, 2021

[26] D. -C. Wang, I. -R. Chen and H. Al-Hamadi, "Reliability of Autonomous Internet of Things Systems With Intrusion Detection Attack-Defense Game Design," in IEEE Transactions on Reliability, vol. 70, no. 1, pp. 188-199, March 2021

[27] M. Abdel-Basset, H. Hawash, R. K. Chakrabortty and M. J. Ryan, "Semi-Supervised Spatiotemporal Deep Learning for Intrusions Detection in IoT Networks," in IEEE Internet of Things Journal, vol. 8, no. 15, pp. 12251-12265, 1 Aug.1, 2021

[28] Dutta, V.; Chora's, M.; Pawlicki, M.; Kozik, R. A Deep Learning Ensemble for Network Anomaly and Cyber-Attack Detection. Sensors 2020, 20, 4583.

[29] Aygun, R.C.; Yavuz, AG Network anomaly detection with stochastically improved autoencoder based models. In Proceedings of the 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud), New York, NY, USA, 26–28 June 2017; pp. 193–198.

[30] Shone, N.; Ngoc, TN; Phai, VD; Shi, Q. A deep learning approach to network intrusion detection. IEEE Trans. Emerg. Top. Comput. Intell. 2018, 2, 41–50.

[31] A. Tabassum, A. Erbad, A. Mohamed and M. Guizani, "Privacy-Preserving Distributed IDS Using Incremental Learning for IoT Health Systems," in IEEE Access, vol. 9, pp. 14271-14283, 2021.

[32] Vaccari, I.; Chiola, G.; Aiello, M.; Mongelli, M.; Cambiaso, E. MQTTset, a New Dataset for Machine Learning Techniques on MQTT. Sensors 2020, 20, 6578.

[33] Ahmadon, M.A.B.; Yamaguchi, N.; Yamaguchi, S. Process-Based Intrusion Detection Method for IoT System with MQTT Protocol. In Proceedings of the 2019 IEEE 8th Global Conference on Consumer Electronics (GCCE), Osaka, Japan, 15–18 October 2019; pp. 953–956

[34] Hindy, H.; Bayne, E.; Bures, M.; Atkinson, R.; Tachtatzis, C.; Bellekens, X. Machine learning based IoT Intrusion Detection System: An MQTT case study (MQTT-IoT-IDS2020 dataset). In Proceedings of the International Networking Conference, Online, 19–21 September 2020; Springer: Cham, Switzerland, 2020; pp. 73–84

[35] Alaiz-Moreton, H.; Aveleira-Mata, J.; Ondicol-Garcia, J.; Muñoz-Castañeda, A.L.; García, I.; Benavides, C. Multiclass classification procedure for detecting attacks on MQTT-IoT protocol. Complexity 2019, 2019, 6516253

[36] Khan, M.A., Khan, M.A., Jan, S.U., Ahmad, J., Jamal, S.S., Shah, A.A., Pitropakis, N. and Buchanan, W.J., 2021. A Deep Learning-Based Intrusion Detection System for MQTT Enabled IoT. Sensors, 21(21), p.7016.

[37] I. Ullah and Q. H. Mahmoud, "Design and Development of a Deep Learning-Based Model for Anomaly Detection in IoT Networks," in IEEE Access, vol. 9, pp. 103906-103926, 2021, doi: 10.1109/ACCESS.2021.3094024.

[38] H. Hindy, “Hanan Hindy,” IEEE DataPort, 23-Jun-2020. [Online]. Available: https://ieee-dataport.org/authors/hanan-hindy. [Accessed: 31-Oct-2022].