

Scalable AI-Based Framework for Anomaly Detection and MQTT Traffic Optimisation in IoT Systems

Emran Aljarrah¹ , Bilal Bataineh¹  Saleem Alzoubi¹  Md Nasiruzzaman²  and Mahdi H. Miraz^{2,3,4,*} 

¹Jadara University, Irbid, Jordan

o.aljarrah@jadara.edu.jo; b.bataineh@jadara.edu.jo; s.alzoubi@jadara.edu.jo

²School of Computing and Data Science, Xiamen University Malaysia, Malaysia

PCS2411001@xmu.edu.my; m.miraz@ieee.org

³Faculty of Arts, Science and Technology, Wrexham University, UK

m.miraz@ieee.org

⁴Faculty of Computing, Engineering and Science, University of South Wales, UK

m.miraz@ieee.org

*Correspondence: m.miraz@ieee.org

Received: 11 June 2026; Accepted: 29 September 2026; Published: 1 October 2026

Abstract: Internet of Things (IoT) networks are becoming increasingly prevalent, but facing serious challenges with regards to data integrity and communication security, due to resource constraint IoT devices being used in large distributed networks. This paper proposes a scalable AI-based anomaly detection framework and efficient traffic optimisation for Message Queuing Telemetry Transport (MQTT) messages sent through IoT systems. Both real-time data collected from the IOT nodes as well as the CICIDS2017 dataset have been utilised for this research. Due to biases and inaccuracies inherent in the data, a Conditional Generative Adversarial Network (CGAN) model was used for pre-processing the data. A novel selective feature process has also been presented, which first performs Principal Component Analysis (PCA), Genetic Algorithm and Univariate Statistical Test, and then perform dimensionality reduction using the Extreme Learning Machine (PCA-GA-UST-ELM) model. The traffic optimisation process for the overall IoT network uses the integration of a Deep Neural Network (DNN) model and an MQTT parsing engine, to facilitate processing of network traffic through MQTT. The anomalous threats presented in the IoT nodes were classified using the Bidirectional Gated Recurrent Unit (Bi-GRU) model to improve threats detection accuracy of the proposed DNN models. Finally, the routing process of messages has been optimised using a tail hybridisation method of GTLP SO (Game Theory, Linear Programming, Stochastic Problems). The performance of the framework has been evaluated in terms of accuracy, precision, latency, Prediction time and throughput, for varying numbers of epochs and IoT nodes.

Keywords: Anomaly detection; CICIDS2017 dataset; IoT systems; MQTT; Scalable AI

1. Introduction

The Internet of Things (IoT) consists of a collection of connected intelligent devices that can provide numerous services intended to make human life easier. IoT technology has fuelled numerous state-of-the-art applications, such as smart homes, smart cities, optimised traffic systems, automated parking solutions, smart industrial applications and smart wearables, which consistently enable us to live better lives. Despite all the new applications that IoT technology brings to our lives, securing these resource-constrained devices [1, 2] remains as a major concern. Recent cybersecurity incidents have highlighted serious vulnerabilities and weaknesses with respect to the security of the Internet of Things (IoT) devices [3]. The moment an IoT

device connects to the Internet, it becomes an attractive target for adversaries, seeking access to the device to obtain the data and take control of the device for malicious purposes. Many IoT devices have very limited storage and processing capacity [4]. IoT device manufacturers usually tend to incorporate popular new features and capabilities into their designs, simplify design for small footprints and add intelligence to minimise the cost of the devices, while ignoring the importance of implementing strong security measures [5]. Despite recognising the potential security threats, manufacturers often completely ignore them or treat them with much lower level of priority, when chasing the speed of the market. Consequently, the resulting products have many attractive features but assumes fundamental security flaws, which become desirable uptake for the cybercriminals [6]. Because of the resource limitations of IoT devices, it is important to consider IoT devices as vulnerable to various attacks [7]. The highly publicised Mirai attack illustrates how exploited access to default login information allowed potentially millions of Internet of Things (IoT) devices to be taken over, which subsequently became compromised systems that executed a huge distributed denial of service (DDoS) attack on large organisation servers [8].

As a result, there is an urgent requirement to protect IoT devices from exploitation. By design, however, the security requirements of many IoT devices are inadequate to combat these types of attacks, due to inherent vulnerabilities. As defined by the "Internet Engineering Task Force (IETF)", which defines Internet standards, a vulnerability "is a flaw or weakness in a system's design, implementation, or operation and management that could be exploited to violate the system's security policy" [9-11]. In the last few years, many vulnerabilities have surfaced in Internet of Things (IoT) devices. These widespread IoT vulnerabilities generally arise out of protocol implementation mistakes, device management issues, or errors when handling communication messages [12]. Addressing these vulnerabilities usually requires developers to patch IoT devices through a firmware update, but many of these devices lack the ability to be patched, as they were not designed for over-the-air (OTA) patches. Therefore, a large proportion of IoT devices are not able to receive the necessary updates, remaining vulnerable to security flaws and unprotected from being compromised [13, 14].

A more immediate option to protect IoT devices from current security threats would be configuring a network-oriented Intrusion Detection System (IDS), since IDSs typically offer better protection against protocol-based vulnerabilities found in IoT devices and can be specifically designed to understand the various communication protocols of IoT devices [15, 16]. In addition, the IDS can mitigate the likelihood of IoT devices being compromised during the parsing stage by specifying the rules to fix known vulnerabilities. Protocols, such as MQTT and Constrained Application Protocol (CoAP), are commonly found in IoT devices [17]. Unfortunately, ignoring the current specifications for managing these specialised protocols, most current network-based IDSs do not support them [18]. Therefore, we propose a method of providing support for IoT protocols in an open-source IDS, Suricata, with the intent to provide support and increase the security of IoT protocols, networks, and devices. Fig. 1 represents the MQTT protocol.

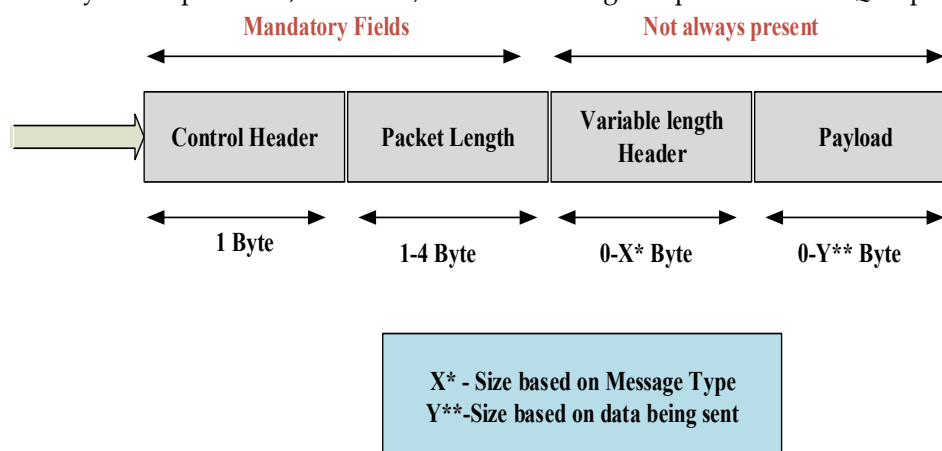


Figure 1. MQTT Protocol

The primary focus of the research is to protect against examples of vulnerabilities in the MQTT protocol, which is a very common IoT communication protocol [19, 20]. The proposed architecture utilises modern AI techniques for abnormal pattern discovery and data optimisation of MQTT stream data to and from clients. As the proposal focuses on using an intelligent data analysis technique through optimisation,

the proposed architecture is directed towards efficiency, scalability and improving security of IoT communication environments. The proposed framework utilises an AI model, providing real-time anomaly detection, to identify patterns exhibiting anomalous behaviour.

Existing IoT intrusion detection systems face critical challenges, including severe class imbalance, high-dimensional un-optimised feature sets, high real-time traffic throughput demands, prolonged neural network training times and dynamic routing protocol uncertainties. To overcome these limitations, this paper proposes an end-to-end scalable AI-based framework for robust anomaly detection and MQTT traffic optimisation in IoT systems. The principal contributions of this work are fivefold: First, we leverage the comprehensive CICIDS2017 dataset and employ a Siamese Conditional Generative Adversarial Network (SCGAN) to synthesise minority attack samples, effectively resolving severe class imbalance. Second, we introduce a novel hybrid feature selection mechanism (PCA-GA-UST-ELM) that eliminates non-informative features and reduces computational overhead. Third, we integrate a Deep Neural Network (DNN) with an inline MQTT parsing engine to pre-filter malformed packets and perform low-latency anomaly detection before broker overload occurs. Fourth, a Bidirectional Gated Recurrent Unit (Bi-GRU) model, optimised via Multiverse Optimisation (MVO), is deployed to capture complex temporal dependencies rapidly. Finally, we propose a Game Theory, Linear Programming and Stochastic Optimisation (GTLPSO) routing framework to eliminate decision-making uncertainty across multi-protocol IoT routing environments, achieving stable and high-throughput communication.

2. Literature Survey

Saba *et al.* [21] utilised a CNN-based approach for anomaly-based intrusion detection systems (IDS) in IoT networks. Their model demonstrated capabilities in real-time packet inspection, automated spatial feature learning and multi-class attack identification. However, their study faced degradation in threat detection rates when encountering rapidly evolving cyber-attack patterns. This highlights a critical need to formulate advanced security protocols combined with recurrent deep learning architectures; resource-constrained IoT nodes cannot support compute-heavy traditional security firewalls, making optimised, non-linear deep learning algorithms essential for identifying zero-day threats in dynamic network environments.

Laanaoui *et al.* [22] suggested a real-time data anomaly detection model utilising parallel data processing, with the aim to achieve faster and more accurate results. This can help to manage traffic in real-time and provide route guidance based on the vehicle density on the different sections. Moreover, a machine learning-based prediction model minimises congestion and accident risks. The simulation results demonstrate its low latency and high degree of precision. The modelling does not necessarily reflect real-life traffic complexities, where indeterminate factors like human behaviours, varying weather conditions and others (for example, car accidents) affect system reliability and performance. Lihua *et al.* [23] proposed an Energy-aware Intrusion Detection System (EIDS) with an intelligent two-phase contract management model for vehicle-to-vehicle (V2V) communication in Internet of Vehicles (IoV) scenarios. Their proposed EIDS is able to predict safe and energy-efficient communication locations between vehicles. The contract management process provides a safe transfer of information, and the system is tested with the NSLKDD dataset based on a regression algorithm. However, many baseline machine learning models applied to vehicular communications lack an optimised feature selection pipeline, passing redundant high-dimensional network attributes directly into classifiers. This leads to sub-optimal classification accuracy, increased latency and excessive energy consumption on battery-constrained vehicular nodes.

Wardana *et al.* [24] suggested a collaborative intrusion detection system (CIDS) to reinforce IoT security using a hierarchical architecture across edge, fog and cloud layers. Blockchain technology is employed to secure inter-node communication, while Federated Learning (FL) preserves data privacy by exchanging local model parameter updates rather than raw sensor payloads. While FL effectively decouples model training from local data exposure, extending CIDS frameworks to changing IoT ecosystems remains challenging: distributed edge nodes experience significant concept drift and heterogeneous traffic distribution, requiring dynamic local model updates rather than static global rule sets. Mutambik *et al.* [25] introduced Flow-based Intrusion Detection System for IoT (IoT-FIDS), a lightweight flow-based Intrusion Detection System for IoT. It detects anomalies by examining communication flows, services and packet

headers, instead of using conventional machine learning techniques. Experimental results indicate that IoT-FIDS can efficiently identify abnormal traffic with low false positives, providing a real-world security solution for IoT networks. IoT-FIDS is based on preconfigured flow-based profiles and thus experiences heightened false positives when faced with new or unknown devices. Its restricted flexibility in dynamic IoT environments lowers accuracy and necessitates enhancements to effectively manage constant network changes.

Balega *et al.* [26] assessed the performance of IoT anomaly detection with machine learning models, such as XGBoost, Support Vector Machines (SVMs) and deep Convolutional Neural Networks (DCNNs). Because of the varied uses and dynamic nature of IoT environments, choosing the best detection model is difficult. The study highlights the need to examine these models for various datasets in different environments. With the range of ML models and varying IoT environments, it is difficult to find a single model that can be effective in all IoT systems. More ML models need to be explored in order to increase detection accuracy and advance industrial and organisational IoT applications. Chohra *et al.* [27] proposed an optimisation strategy based on “swarm intelligence and ensemble techniques” to solve the “non-deterministic feature selection issue”. The strategy is tested on three datasets: “NSL-KDD, UNSW-NB15 and a custom IoT-Zeek dataset” generated from Zeek network logs of the IoT devices. Their optimisation strategy is constrained by the utilisation of a limited number of hyperparameters for Random Forests and XGBoost, limiting model performance. The feature selection algorithm also needs better scalability, and the absence of adaptive tuning for PSO hyperparameters impacts search efficiency. Mahadik *et al.* [28] introduced HetIoT-CNN IDS, an intelligent intrusion detection system (IDS) with a “convolutional neural network (CNN)” for the Heterogeneous IoT (HetIoT) setting. HetIoT-CNN IDS does not support sophisticated prediction functionalities for DDoS attacks. A Recurrent Neural Network (RNN) model needs to be developed for better detection and prediction, and reinforcement learning needs to be included for real-time attack detection in HetIoT environments. Awotunde *et al.* [29] proposed a multilevel random forest algorithm for intrusion detection, which integrates a fuzzy inference system. It incorporates filter and wrapper approaches for enhanced feature selection. The filter component employs correlation-based attribute elimination with genetic search to select significant features, whereas the wrapper component implements sequential forward selection to optimise them. The optimised features are then input into the random forest model, and fuzzy logic demarcates intrusions based on their severity to reduce misclassification. Their work on intrusion detection system does not possess sophisticated deep learning architectures with new optimisation methods. Moreover, their fuzzy logic model must be scaled up to cover greater severity classes in order to enable more accurate intrusion classification. Kethineni *et al.* [30] presented a privacy-encoding-based deep learning framework for data encryption and intrusion detection. The model is not well-tuned for optimal hyperparameters, which impacts the performance of the model. However, their future research plans to propose an effective optimisation algorithm, enhance encryption techniques for smart agriculture and improve early detection and classification of cyber-malware in sophisticated real-world systems.

Lian *et al.* [31] employ inverse reinforcement learning (RL) for anomaly detection and rectification. In training, inverse RL learns the normal system objective function from offline data. In the detection stage, it contrasts online observations with the learned normality to detect anomalies. Anomalies are subsequently rectified by bringing the system into alignment with normal objectives. Simulations on a quadrotor UAV verify the proposed methodology. That being said, their approach needs further improvement in terms of convergence rate, which is important for real-time anomaly detection. Their future research is set to focus on increasing computational efficiency and applying the method to nonlinear autonomous systems, such as autonomous vehicles, industrial production and power generation.

Sarwar *et al.* [32] presented an Improved Dynamic Sticky Binary Particle Swarm Optimisation (IDSBPSO) algorithm for feature selection with dynamic reduction of search space and adaptive parameters to improve the basic SBPSO. The suggested Intrusion Detection System (IDS) was evaluated using IoTID20 and UNSW-NB15 datasets, where comparable or better accuracy was achieved using fewer features. IDSBPSO also had lower computational costs and shorter prediction times compared to conventional PSO-based algorithms. Their future research directions will be on enhancing the subcategory classification accuracy and further reducing computation time. The IDSBPSO algorithm will also be investigated in other

areas other than IoT network security and experimented on using different classifiers to benchmark its performance.

Zakariah *et al.* [33] employed the UNSW-NB15 dataset containing 45 features to determine major attributes for IoT system anomaly detection. Active learning algorithms were utilised to enhance the accuracy of detection by minimising human labelling and adapting to changing environments. In addition, there is a potential of improving reliability and security of IoT systems by combining advanced anomaly detection methods with active learning techniques. Villegas-Ch *et al.* [34] contributed to smart city networks with a proactive approach for the early detection of anomalies to avoid sudden disruption, ensuring public safety and maintaining the integrity of urban infrastructures. However, consolidating various sources of data, including IoT sensors and surveillance, can further contribute to accuracy in detection and system reliability. Manokaran *et al.* [35] introduced an advanced model of anomaly detection via the boosted stacked ensemble learning approach in edge computing. A synthetic dataset was produced by the Cooja simulator, including cross-layer attacks, for training. An improved grey wolf optimisation (IGWO)-based model was implemented for optimising key parameters and improved the prediction power of the model using a stacking ensemble model.

Sharma *et al.* [36] presented the SLSTM-WCO algorithm, which integrates Stacked Long Short-Term Memory (LSTM) and Willow Catkin Optimisation for the detection of intrusion anomalies in IoT networks. The model improves on anomaly detection using deep learning and regularisation methods to efficiently identify complex patterns. In their research, they had challenges in class imbalance, and also it reduces the reliability while performing detection.

Alsulami *et al.* [37] proposed a prediction machine learning model for the detection and categorisation of IoT system network activities as normal or abnormal. Five supervised learning models—shallow neural networks (SNN), decision trees (DT), bagging trees (BT), k-nearest neighbor (kNN) and support vector machines (SVM)—were compared to their ability in detecting and classifying network anomalies. However, their research faced problems related to feature selection, which compromised the accuracy and efficiency of the model.

Gadal *et al.* [38] present a machine learning (ML)-based anomaly detection model that can enhance the accuracy of detection and minimise false positives. The proposed model integrates K-means clustering and Sequential Minimal Optimisation (SMO) to classify intrusions. The performance of the proposed model was assessed using the NSL-KDD dataset, illustrating that the hybrid model increases positive detection rates, reduces false alarms and delivers high classification accuracy. Yet, as the real-time traffic volume on the network increases, achieving acceptable performance is a challenge. This demands high-performance processing to efficiently and accurately conduct real-time analysis.

Chaganti *et al.* [39] introduced an LSTM-based mechanism for IoT network attack detection within an SDN-enabled environment. While their model improved multiclass classification accuracy across SDN-IoT datasets, the training time required was exceptionally high. This computational bottleneck stems from processing high-dimensional, raw sequence flows directly through multi-layered recurrent LSTM cells without prior feature dimensionality reduction, creating sequential backpropagation bottlenecks.

Khan *et al.* [40] designed a hybrid routing strategy for heterogeneous Rescue Ad Hoc Networks. In these proposed routing decisions, some nodes may implement the protocol, whereas others may choose not to. Intelligent nodes apply the proposed routing protocol for communication. They maintain reactive and proactive tables with respect to the neighbouring routing. So, these nodes can change their route. The game theory is also applied to manipulate its avoidance routing protocol. Smart nodes take several steps to select a routing protocol for themselves and the system. Smart node default protocols depend upon energy levels, energy consumption, and mobility ratios. They employed Ad-hoc On-demand Distance Vector (AODV) and Destination-Sequenced Distance-Vector (DSDV) as base protocols. These two can be substituted by any number and kind of protocols. The optimisation of normalisation for multimode routing solutions followed by all finite parameters falling within an ambit of network node would be solved by linear programming. Multiple protocol routing may introduce some level of ambiguity. Therefore, a method to resolve uncertain judgments would emanate through game theory and linear programming. That being said, the fusion of multiple routing protocols can lead to uncertainty issues, affecting the stability and performance of the network. This issue arises due to discrepancies in protocol behaviour and decision-making strategies. Table 1. provides the literature review summary which includes specified research gaps and objectives.

Table 1. Summary of Literature Survey

Ref. No.	Objective	Algorithm Used in This Research	Limitation
Saba <i>et al.</i> [21]	To detect IoT network anomalies.	CNN	- Evolving threat patterns.
Laanaoui <i>et al.</i> [22]	To achieve faster data anomaly detection.	Parallel data processing, Machine Learning	- Not reflecting real complexities.
Lihua [23]	To prediction safe V2V communication.	Regression Algorithm	- Unoptimised feature selection.
Wardana <i>et al.</i> [24]	To reinforce IoT security.	Blockchain, Federated Learning	- Evolving threat scenarios.
Mutambik [25]	To detect IoT communication anomalies.	Flow-based profiles	- Heightened false positives.
Balega <i>et al.</i> [26]	To assess IoT anomaly detection.	XGBoost, SVMs, DCNNs	- A single model is not effective.
Chohra <i>et al.</i> [27]	To optimise feature selection.	Swarm intelligence, Autoencoders	- Limited hyperparameters.
Mahadik <i>et al.</i> [28]	To detect and defend DDoS attacks.	CNN	- Lacks sophisticated prediction.
Awotunde <i>et al.</i> [29]	To classify intrusions by severity.	Multilevel Random Forest, Fuzzy Inference System	- Lacks deep learning architectures.
Kethineni & Gera [30]	To encrypt data and detect intrusions.	Sparse Capsule Autoencoder, Attention-based GRU	- Suboptimal hyperparameters.
Lian <i>et al.</i> [31]	To detect and rectify anomalies.	Inverse Reinforcement Learning	- Convergence rate challenges.
Sarwar <i>et al.</i> [32]	To improve feature selection efficiency.	Improved Dynamic Sticky Binary Particle Swarm Optimisation (IDSBPSO)	- Subcategory classification accuracy.
Zakariah & Almazyad [33]	To enhance anomaly detection accuracy.	Active learning, Random Forest	- Scalability and efficiency in large deployments.
Villegas-Ch <i>et al.</i> [34]	To detect smart city anomalies early.	Proactive approach	- Generalising methodologies to diverse cases.
Manokaran & Vairavel [35]	To detect anomalies via ensemble learning.	Boosted stacked ensemble learning, Improved IGWO	- Detecting zero-day attacks.
Sharma & Prasad [36]	Detection of intrusion anomalies on IoT networks.	SLSTM-WCO (Stacked Long Short-Term Memory and Willow Catkin Optimisation)	- Challenges in class imbalance, reducing detection reliability.
Alsulami <i>et al.</i> [37]	Detection and categorisation of IoT system network activities as normal or abnormal.	Five supervised learning models: Shallow Neural Networks (SNN), Decision Trees (DT), Bagging Trees (BT), k-Nearest Neighbour (kNN) and Support Vector Machines (SVM)	- Problems related to feature selection, compromising model accuracy and efficiency.
Gadal <i>et al.</i> [38]	Enhance the accuracy of anomaly detection and minimise false positives in an ML-based anomaly detection model.	K-means clustering and: Sequential Minimal Optimisation (SMO)	- Achieving acceptable performance is a challenge with increased real-time traffic volume, demanding high-performance processing for efficient and accurate real-time analysis.
Chaganti <i>et al.</i> [39]	IoT network attack detection using an SDN-enabled intrusion detection system and enhancement of multiclass classification accuracy for different types of network attacks.	LSTM-based mechanism	- The proposed method requires a lot of training time to detect anomalies in the IoT system.
Berraadi <i>et al.</i> [41]	Designing a hybrid routing strategy for heterogeneous Rescue Ad Hoc Networks.	Game theory and a linear programming-based approach, utilising AODV and DSDV as base protocols.	- Considering multiple routing protocols can lead to uncertainty issues, affecting network stability and performance due to discrepancies in protocol behaviour and decision-making strategies.

This article proposes a robust IoT security solution addressing class imbalance and improving detection accuracy and efficiency. It leverages a Conditional Generative Adversarial Network (CGAN) for data balance and employs a hybrid feature selection approach combining PCA, GA, UST, and ELM to

enhance data quality. A Deep Neural Network (DNN) integrated with an MQTT parsing engine ensures low-latency, high-accuracy real-time anomaly detection and mitigates malformed packet threats. Furthermore, a Bidirectional Gated Recurrent Unit (Bi-GRU) is utilised for rapid anomaly detection and reduced training times. Finally, a novel GTLPSO (Game Theory, Linear Programming, Stochastic Optimisation) framework resolves routing protocol uncertainty, enhancing network stability and routing efficiency.

3. Proposed Methodology

The goal of this study is to design and develop a large-scale AI framework for anomaly detection and optimisation of the MQTT traffic in the IoT environment. Whilst figure 2 depicts the overall architecture of this research, the main features of the proposed method are:

- Dataset collection;
- Preprocessing;
- Feature selection;
- MQTT-based traffic optimisation;
- Anomaly classification in IoT and
- Decision-making driven routing.

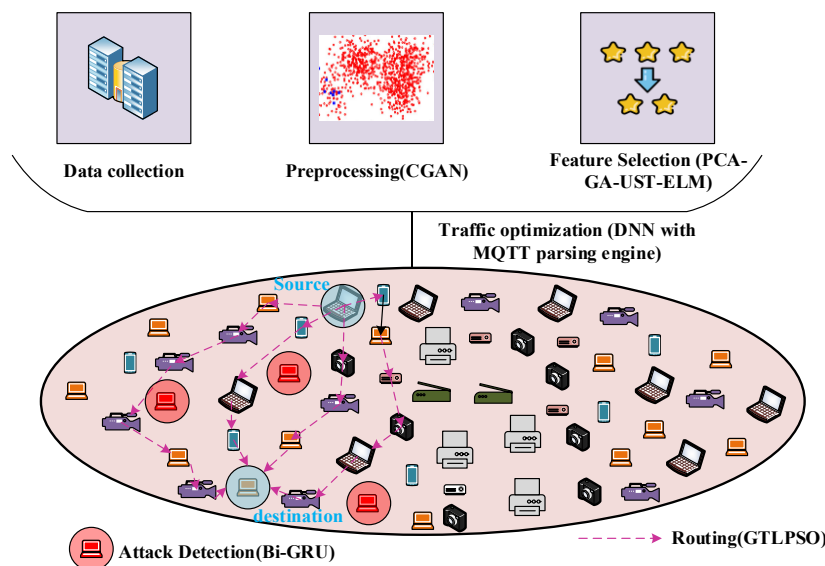


Figure 2. Overall Architecture of this Research

3.1. Dataset Collection

To establish a scalable anomaly detection and traffic optimisation framework, we acquire the CICIDS2017 dataset [1]. The dataset contains 2,830,743 network traffic flow vectors generated over 5 consecutive days, comprising normal background traffic alongside 14 contemporary cyber-attack classes (e.g., DoS, DDoS, Brute Force, Web Attacks, Infiltration, Botnet and Heartbleed) captured across 78 network flow features. The diversity and realistic volume of this dataset make it highly suitable for evaluating model generalisation in IoT environments¹.

3.2. Preprocessing

Data gathered during the previous step is hardly in a suitable form for modelling and hence proper preprocessing is needed to transform it into an organised version that helps facilitate model performance. Traditional approaches tend to be poorly suitable for omitted attack classes, which leads to biased and incorrect predictions. Therefore, we generate synthetic samples for minority attack classes with a uniform distribution over classes using CGANs. This approach balances the data and enables model generalisation

¹ <https://www.unb.ca/cic/datasets/ids-2017.html>

to detect rare outliers in the major category. A crucial metric for understanding data imbalance is the Imbalance Ratio (IR_i) for each class, calculated as:

$$IR_i = \frac{N_{\max}}{N_i} \quad (1)$$

Where $N_{\max}$ signifies the number of normal samples and N_i denotes the number of samples for the i -th attack type. Based on these IR_i values, intrusion detection traffic is categorised into three levels: ample, scarce and rare. Prior to applying augmentation techniques, Min-Max normalisation is performed on the dataset. This process transforms feature values to a range between 0 and 1, which is essential to prevent slow network convergence and neuron output saturation caused by wide disparities in feature values. The normalisation formula is as follows:

$$X_{\text{norm}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \quad (2)$$

Where X_{norm} is the normalised value, X is the original feature value, $X_{\min}$ is the minimum value of the feature and $X_{\max}$ is the maximum value of the feature. For ample-level categories, which exhibit complete distribution, no additional processing is typically performed to minimise computational overhead. For scarce-level attacks, a specialised technique entitled “Siamese Conditional Generative Adversarial Network (SCGAN)” is used. SCGAN combines a Siamese Autoencoder Network (SAN) with a “Generative Adversarial Network (GAN)” to learn the original data distribution and generate missing data by leveraging different information (s) from the SAN and random noise ($z \sim N(\mu, \sigma^2)$). The fundamental objective of SCGAN is expressed through a min-max game between the discriminator (D) and generator (G) as follows:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p} [\log D(x)] + \mathbb{E}_{x \sim p} [\log (1 - D(G(z | s)))] \quad (3)$$

Here, $V(D, G)$ represents the value function, p_x is the data distribution and p_z is the noise distribution. The loss function for the discriminator, L_D aims to correctly classify real and generated samples:

$$L_D = -\log (D(x)) - \log (1 - D(G(z | x))) \quad (4)$$

Conversely, the generator's loss function, L_G , focuses on producing samples that can deceive the discriminator, p :

$$L_G = -\log (D(G(z | s))) \quad (5)$$

Once the SCGAN model had achieved convergence, a filtering process was put in place to assist with removing the noise that was left over from the generated samples. This method addressed the low number of samples they had by limiting the identification of problems to the nearest neighbours of the existing rare-level instance and synthesised new samples based on the inputs from these neighbours. The generation of a new sample ($X_{\max}$) is typically performed using a transformation function:

$$X_{\max} = X_i + \lambda \times (X_j - X_i) \quad (6)$$

Where X_i is an existing rare-level sample, X_j is one of its k -nearest neighbours, and λ is a random number chosen from $[0, 1]$. This process effectively expands the original distribution from the feature space. Moreover, CGANs ensure the efficacy of the anomaly detection framework against both frequent and novel cyber-attacks, thereby improving the overall reliability of the IoT network. Ultimately, this pre-processing step forms the salient basis for the comprehensive and effective detection framework, competent enough to handle intricate scenarios of IoT traffic.

3.3. Feature selection

After preprocessing was complete, **Novel PCA, Genetic Algorithm and Univariate Statistical Test followed by Extreme Learning Machine (PCA-GA-UST-ELM)** was employed for feature selection. This hybrid method only chooses the most significant features, discarding noise in addition to improving data quality. PCA compresses dimensions by transforming the data into a collection of orthogonal components, in which variance is attenuated with the unexpected result of an increase in computational complexity. A genetic algorithm is a natural selection-inspired iterative optimisation procedure running across generations that enhances the performance of a feature subset and ensures that the most informative features are selected. UST provides with feature importance scores, calculated from statistical significance

across the features. Taking into consideration the feature importance scores anticipated by UST, the integration of ELM then takes forward the process of feature selection, and the overfitting is reduced.

Principal Component Analysis (PCA) is a dimensionality reduction technique. It aims to take data from the original feature space and put it into a new space with lower dimensionality that is comprised of orthogonal components, defined as the principal components. Ultimately, the goal of the principal components is to maximise the variance from the data while reducing the dimension space, meaning that many relevant information contained in the data can be preserved with much fewer variables. This data simplifying process effectively compresses the dimensions of the dataset, mitigating the impact of high dimensionality, which can degrade model performance. The total variance within the entire dataset, which contains M different data characteristics (features), can be represented as:

$$\text{Var}_{\text{total}} = \sum_{j=1}^M \sigma_j^2 \quad (7)$$

Where $\text{Var}_{\text{total}}$: Represents the total spread or variability across all data characteristics in dataset. M: The total number of original data characteristics (features). σ_j^2 : The variance (spread) of the j-th individual data characteristic.

PCA achieves dimensionality reduction “by projecting the data onto a lower-dimensional subspace defined by a subset of principal components” [27]. Each principal component, PC_k , is a linear combination of the original features, weighted by the eigenvectors of the covariance matrix. The covariance between any two data characteristics, X_a and X_b , helps us understand how they change together. It's calculated as:

$$\text{Cov}(X_a, X_b) = \frac{1}{N-1} \sum_{i=1}^N (x_{ia} - \bar{x}_a)(x_{ib} - \bar{x}_b) \quad (8)$$

Here N : The total number of data samples (e.g., network traffic records) in dataset. x_{ia} : The specific value of the i-th data sample for characteristic a. $\bar{x}_a$: The average value of data characteristic a across all samples. The eigenvalues (λ_k) derived from the covariance matrix:

$$PVE_k = \frac{\lambda_k}{\sum_{j=1}^M \lambda_j} \quad (9)$$

Where PVE_k is the amount of total data variability that can be explained by the k-th principal component. λ_k is the eigenvalue associated with the k-th principal component, showing the amount of variance explained by it. M: The total number of original data features. Choosing principal components with high eigenvalues allows for considerable amount of data.

Following or in conjunction with PCA, the Genetic Algorithm (GA) is employed as an iterative optimisation procedure inspired by natural selection, running across generations to enhance the performance of a feature subset. GA is particularly effective in optimising parameters, such as the weights of an Extreme Learning Machine. The core of GA involves maintaining a population of candidate solutions, known as chromosomes or individuals, which represent different subsets of features. The fitness of each individual is evaluated using a fitness function that quantifies how well a particular feature subset performs (e.g., in terms of classification accuracy). A common form of a fitness function $F(S)$ for a feature subset S is:

$$F(S) = \alpha \times \text{Accuracy} - \beta \times |S| \quad (10)$$

Here $F(S)$: The calculated score representing how good a specific group of features (subset S) is. A higher score means a better feature subset. Accuracy: The performance (e.g., detection accuracy) achieved by model when using only the features in subset S. $|S|$: The count of features included in the subset S. α, β : Adjustable weights that control how much importance is given to model accuracy versus reducing the number of features. A binary representation for a chromosome C with M total features visually shows which features are selected. It can be presented as:

$$C = [c_1, c_2, \dots, c_M] \quad (11)$$

C represents a potential solution or a specific group of selected features. c_j : Is a binary value (0 or 1) for the j-th feature. If $c_j = 1$, the feature is included; if $c_j = 0$, it is excluded. M: The total number of original data characteristics (features). The mutation operation randomly introduces changes to the selected features (genes) to maintain diversity and explore new combinations. This happens with a small probability P_m :

$$c'_j = \begin{cases} 1 - c_j & \text{if rand}() < P_m \\ c_j & \text{otherwise} \end{cases} \quad (12)$$

Where c'_j : The new (potentially changed) state of the j-th feature after mutation. P_m : The small chance (probability) that a feature's selection status (0 or 1) will randomly flip. $\text{rand}()$: A random number between

0 and 1, employed to decide whether mutation occurs. Crossover combines parts of two "parent" feature sets, P_1 and P_2 , at a randomly chosen point to create two new "offspring" feature sets, O_1 and O_2 . If the crossover point is k :

$$O_1 = [P_{1,1}, \dots, P_{1,k}, P_{2,k+1}, \dots, P_{2,M}] \quad (13)$$

$$O_2 = [P_{2,1}, \dots, P_{2,k}, P_{1,k+1}, \dots, P_{1,M}] \quad (14)$$

Here O_1, O_2 : The new groups of selected features created by combining parts of two parent groups. P_1, P_2 : The two existing groups of selected features that are being combined. k : A randomly chosen point in the list of features where the two parent groups are "split" and combined. M : The total number of original data characteristics (features). These operations iteratively refine the feature subset, ensuring the selection of the most informative features.

The Univariate Statistical Test (UST) further refines feature selection by providing importance scores calculated from the statistical significance of each feature. This helps to objectively identify which features are critical for distinguishing between different classes. Various statistical tests can be used, depending on the nature of the feature and the target variable (e.g., F-statistic for Analysis of Variance (ANOVA), Chi-squared test for categorical variables, t-tests for numerical differences, etc.). For instance, in the context of ANOVA, the F-statistic for a feature compares the variance between group means to the variance within the groups, indicating how well the feature separates the classes. The F-statistic can be represented as

$$F = \frac{MS_{\text{between}}}{MS_{\text{within}}} \quad (15)$$

Where F is the F-statistic value, which indicates how much the groups differ from each other compared to how much variability there is within each group. MS_{between} : The mean square between groups, representing the variance (spread) between the average values of different categories of target variable (e.g., normal vs. attack traffic). MS_{within} : The mean square within groups, representing the variance (spread) within each individual category of target variable. For a Chi-squared test, which assesses if there is a significant relationship between two categorical variables (e.g., feature X and target Y), it is calculated as:

$$\chi^2 = \sum_{i=1}^n \sum_{j=1}^c \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \quad (16)$$

Here χ^2 : The Chi-squared test statistic, which measures the change amongst observed and expected frequencies. A larger value indicates a stronger relationship. O_{ij} : The actual count (observed frequency) of data points in row i and column j of a contingency table. E_{ij} : The predict count (expected frequency) of data points in row i and column j , assuming no relationship between the variables. R : The number of unique categories for feature (rows in the table). C : The number of unique categories for target variable (columns in the table). These tests provide a statistical basis for ranking features by their individual relevance.

Finally, the Extreme Learning Machine (ELM) is integrated into the feature selection process, offering a faster and more efficient approach with reduced overfitting. The output weights are then analytically determined through a single step. For an ELM with L hidden processing units (nodes), the output generated for an input data sample x is given by:

$$f_L(x) = \sum_{j=1}^L \beta_j g(w_j \cdot x + b_j) \quad (17)$$

Here $f_L(x)$: The final prediction output of the ELM for a given input data sample x . L : The total number of hidden processing units (neurons) in the ELM's middle layer. β_j : The output weight connecting the j -th hidden unit to the final output. These are the values ELM learns. g : The activation function (e.g., sigmoid, ReLU) used by each hidden unit to transform its input. w_j : The randomly assigned input weights connecting the input data to the j -th hidden unit. I : The input data sample (vector of feature values). b_j : The randomly assigned bias for the j -th hidden unit. If H is the output matrix of the hidden layer (after applying the activation function to all training samples) and T is the target output matrix (the true labels), then the output weights $\hat{\beta}$ are calculated using the Moore-Penrose generalised inverse (H^+) of the hidden layer output matrix (H), as presented below:

$$\hat{\beta} = H^+ T \quad (18)$$

Here $\hat{\beta}$: The calculated matrix of output weights, which are the values ELM learns to connect the hidden layer to the output. H^+ : The pseudoinverse of the hidden layer output matrix H . This mathematical operation allows us to directly solve $\hat{\beta}$. T : The matrix of actual target values or true labels for training data.

This analytical solution makes ELM significantly faster than traditional neural networks, contributing to a more efficient feature selection process. The integration of GA to optimise ELM's input parameters further enhances its performance for feature selection. This integrated PCA-GA-UST-ELM approach guarantees that the model works with only the most relevant and high-quality features, which directly enhances accuracy and performance, particularly for anomaly detection in IoT networks.

3.4. MQTT-based traffic optimisation

Once feature selection has taken place, real-time network traffic management of the IoT network employs the proposed architecture, which is a hybrid approach utilising a Deep Neural Network (DNN) with an MQTT parsing engine so that the traffic can be well managed as well as real-time anomaly detection can be efficiently performed. The DNN will have low detection latency and offer high detection rate while performing real-time monitoring, even when the traffic load fluctuates. The MQTT parsing engine will inspect the incoming packets for malformation. Any malformed packets identified will be discarded, thus protecting the system being overloaded with them. Thus, by using this two-layer solution, the MQTT broker can benefit from low processing delay, reduced latency and improved throughput. DNNs, as a class of artificial neural networks with multiple hidden layers, are exceptionally adept at learning complex, non-linear relationships within vast datasets. In this context, the DNN is trained on historical IoT network traffic data, encompassing both normal and anomalous patterns. During real-time operation, incoming network traffic data, potentially pre-processed through feature selection, is fed as input to the DNN. Each input vector, x , represents a snapshot of network characteristics, e.g., packet size, inter-arrival time, protocol headers, payload features. The DNN processes these inputs through its layers, applying a series of weighted sums and activation functions to ultimately produce an output representing the likelihood or classification of an anomaly. For a DNN with L layers, the output $h^{(l)}$ of layer l is calculated as:

$$h^{(l)} = f^{(l)}(W^{(l)}h^{(l-1)} + b^{(l)}) \quad (19)$$

Where $h^{(0)}=x$ is the input vector, $W^{(l)}$ is the weight matrix for layer l , $b^{(l)}$ is the bias vector for layer l , $f^{(l)}$ is the activation function (e.g., ReLU, sigmoid, tanh) for layer l . The final output layer, $y=h^{(L)}$, might produce a probability score for anomaly presence (e.g., using a sigmoid activation for binary classification) or a categorical classification for different anomaly types (e.g., using a softmax activation for multi-class classification). Fig. 3 shows the architecture of the proposed DNN.

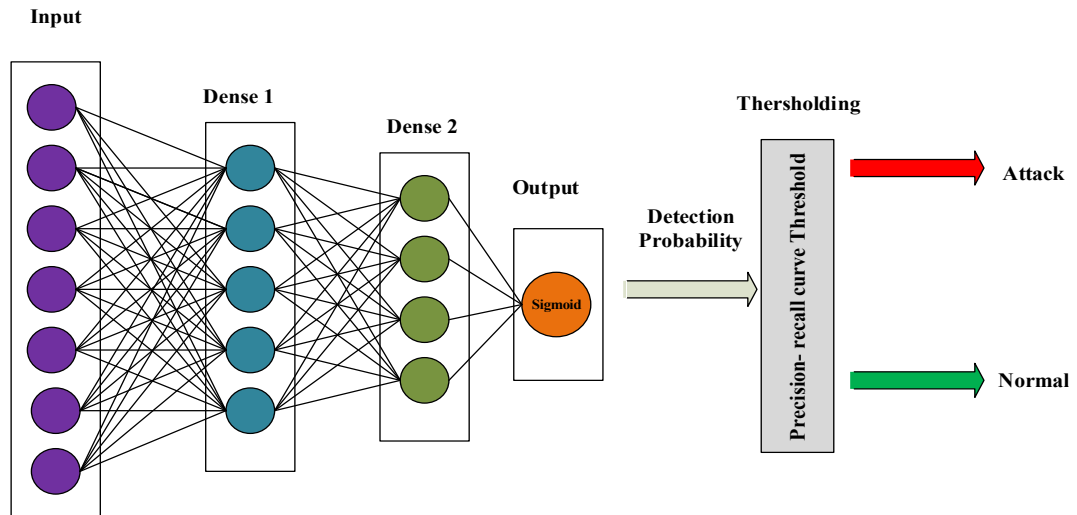


Figure 3. DNN Architecture

3.4.1. MQTT-based traffic optimisation

The **MQTT parsing engine** operates in parallel and in conjunction with the DNN, serving as a crucial first line of defence against malformed or malicious packets. Its primary role is to inspect incoming MQTT packets at a lower level of the protocol stack, identifying and dropping packets that do not conform to the MQTT specification before they can consume valuable processing resources or potentially overload the MQTT broker.

3.4.2. Elaboration on MQTT (Message Queuing Telemetry Transport)

MQTT is an extremely lightweight publish–subscribe messaging protocol designed for constrained devices and low-bandwidth, high-latency, or unreliable networks. MQTT is built on top of TCP/IP, and it is favoured for use in the Internet of Things (IoT) applications because of its efficiency and simplicity.

3.4.3. Key Concepts of MQTT

Publish/Subscribe Model: Unlike traditional request/response models, MQTT decouples the senders (publishers) from the receivers (subscribers) via a central intermediary known as the **MQTT Broker**. Publishers send messages on a specific ‘topic’ and subscribers receive messages from topics they have subscribed to. This allows for many-to-many communication without direct sender-receiver coupling.

- **Publisher:** An entity that sends application messages to a broker on a specific topic.
- **Subscriber:** An entity that receives application messages by registering interest with a broker for specific topics.
- **Broker:** The central hub that receives messages from publishers, filters them by topic and sends them to interested subscribers.

Topics: Messages are organised by topics, which are UTF-8 strings. They are hierarchical, allowing for flexible message routing. Wildcards can be used by subscribers.

Quality of Service (QoS): MQTT defines three levels of QoS to ensure message delivery reliability:

- **QoS 0 (At Most Once):** Messages are delivered with no guarantee of arrival. The sender sends a message once and doesn't wait for any acknowledgment. This offers the lowest latency and bandwidth usage, as shown in Fig. 4.
- **QoS 1 (At Least Once):** Messages are guaranteed to arrive at least once. The sender sends a message and expects an acknowledgment (PUBACK). If no PUBACK is received within the timeout period, the message is re-sent. Duplicate messages are possible.
- **QoS 2 (Exactly Once):** Messages are guaranteed to arrive exactly once. This is the slowest and most resource-intensive Quality of service (QoS) level, involving a four-way handshake (PUBLISH, PUBREC, PUBREL and PUBCOMP) to eliminate the chances of packet duplication and ensure guaranteed delivery service, as shown in fig. 5.

The choice of QoS level, $QoS_L \in \{0,1,2\}$, significantly impacts network overhead, latency and reliability. Higher QoS levels, while offering greater reliability, introduce more control packets, thereby increasing the effective data size and processing overhead.

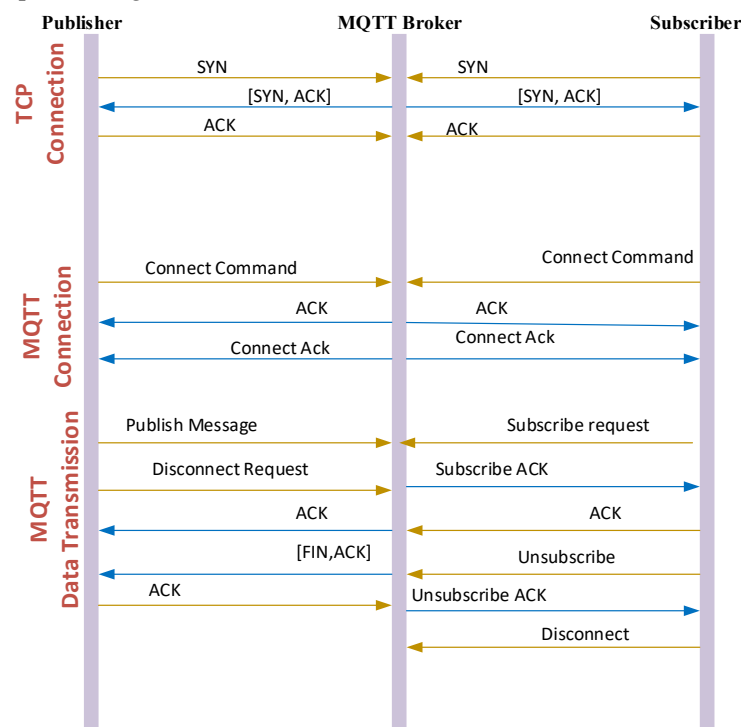


Figure 4. MQTT (QoS 0)

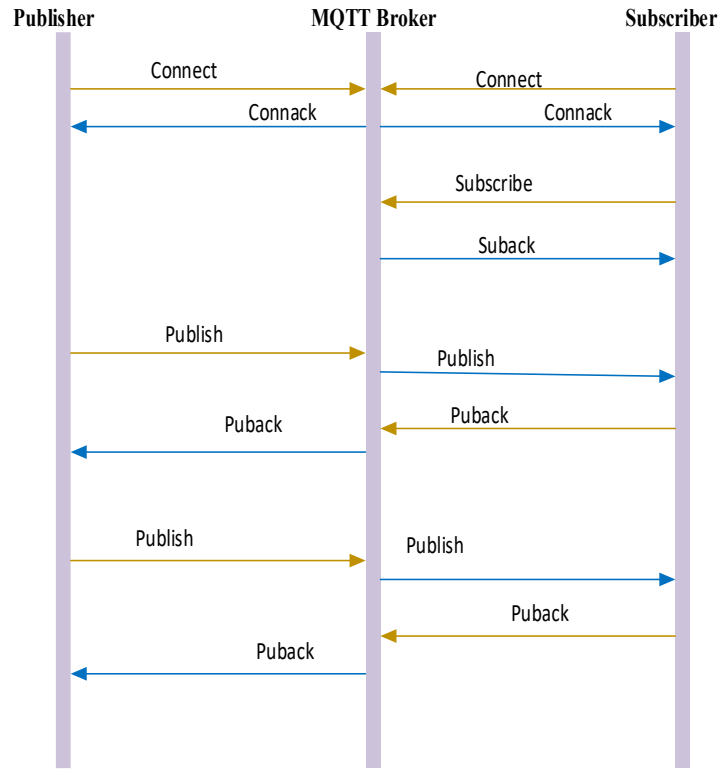


Figure 5. MQTT (QoS 2)

3.4.4. Conceptual Relationship for QoS Impact on Latency and Throughput

Let T_P be the processing time for a single MQTT packet at the broker and O_{QoS} be the overhead associated with the chosen QoS level (number of control packets per application message). The effective latency, L_{eff} , for a message could be conceptually expressed as:

$$L_{eff} = L_{trans} + P_{Queue} + T_P \cdot (1 + O_{QoS}) \quad (20)$$

where L_{trans} is transmission latency and P_{Queue} is queuing delay. O_{QoS} would be 0 for QoS 0, 1 for GoS1 and 3 for GoS2 (approximately for the handshake messages).

Packet Structure: MQTT packets consist of a fixed header, an optional variable header and an optional payload. The fixed header contains crucial information like the packet type (e.g., CONNECT, PUBLISH, SUBSCRIBE) and flags. The parsing engine meticulously examines these fields.

$$N_{valid} = N_{total} - N_{malformed} \quad (21)$$

Let N_{total} be the total number of incoming MQTT packets. Let N_{mal} represent be the number of malformed packets identified by the parsing engine. Let N_{valid} be the number of valid packets passed to the DNN. The Drop Rate (DRate) of malformed packets is:

$$DRate = \frac{N_{malformed}}{N_{total}} \quad (22)$$

By dropping malformed packets early, the parsing engine DNN prevents the overload of the MQTT broker. An overloaded broker, B_{ov} , experiences:

- Increased message queuing delays, Q_D .
- Reduced throughput, Th_B .
- Potential service disruptions or crashes.

The parsing engine's immediate action on malformed packets implies that the processing delay D_{parse} , for such packets is minimal. This direct reduction in junk traffic leads to:

Reduced Latency: Less time spent by the broker and DNN on invalid data.

$$L_{system} \downarrow \text{ as } N_{malformed} \uparrow \text{ (and are dropped)} \quad (23)$$

Improved Throughput: The system can process a higher volume of legitimate traffic.

$$Th_{system} \uparrow \text{ as } N_{malformed} \uparrow \text{ (and are dropped)} \quad (24)$$

Conceptual Throughput Improvement: Let T_{DNN} be the average processing time per packet for the DNN and T_{broker} be the average processing time per valid packet for the MQTT broker. The total processing capacity, C_{system} , of the system is inversely proportional to the combined average processing time per packet. By reducing the number of packets that reach the DNN and the MQTT broker, the effective processing load is reduced, leading to an increase in sustainable throughput.

$$\text{Effective Throughput } (Th_{eff}) = \frac{N_{valid}}{T_{DNN} + T_{broker}} \quad (25)$$

Compared to a scenario without the parsing engine, where N_{total} packets (including malformed ones) might consume resources. Thus, the detailed analysis achieved using the integrated DNN and MQTT parsing engine not only accelerates the detection of various types of anomalies but also protects the network infrastructure without compromising network resource capabilities. As the term suggests, an end-to-end approach in IoT systems allows large streams of data to be pulled in, processed and analysed to derive useful insights, and thus enables uninterrupted traffic optimisation. Algorithm 1 represents the MQTT based traffic optimisation process.

Algorithm 1. MQTT-based traffic optimisation

```

1.  Input: MQTT packet stream  $P[1 \dots N]$ 
2.  Initialise: Valid  $\leftarrow []$ , Malformed  $\leftarrow []$ 

3.  for  $i = 1$  to  $N$ 
4.     $pkt \leftarrow P[i]$ 
5.    if it is Malformed( $pkt$ )
6.      Malformed.append( $pkt$ )
7.      continue
8.    else
9.      Valid.append( $pkt$ )
10.   end
11. end
12. for  $pkt$  in Valid
13.    $x \leftarrow \text{extractFeatures}(pkt)$ 
14.    $y \leftarrow \text{DNN.predict}(x)$ 
15.   if  $y = \text{"anomaly"}$ 
16.     alert( $pkt$ )
17.   else
18.     broker.forward( $pkt$ )
19.   end
20. end

```

3.5. Anomaly Classification in IoT

After optimising the traffic, the proposed model acts towards anomaly classification in the IoT system. This is achieved through a **Bidirectional Gated Recurrent Unit (Bi-GRU) model**, which is well known for its success in discovering temporal dependencies in sequence data. Unlike basic recurrent models, Bi-GRU allows data to be processed in both the directions, which brings a much clearer perspective towards data patterns. This bidirectional ability is critical, for instance, in the case of time-spanning, fine-grained anomaly detection.

Unlike conventional recurrent neural networks, the Bi-GRU offers a distinct advantage by processing input data in two directions: forward and backward. This bidirectionality offers a much more transparent and conclusive inspection of data patterns, which is necessary in order to focus on time-relative subtle anomalies. Additionally, working in two directions is necessary, from all the points in a prior sequence and also to look at patterns over the whole series of data points that gradually build. The Bi-GRU will not only continue to improve the structural fidelity of the anomaly detection model, but it also improves the efficacy of the training process by better handling long-term dependencies. For instance, methods such as GRUs and RNNs can eliminate a lot of the overhead of having a large computational burden whilst improving classification accuracy. Thus, it aims to support a truly effective and efficient anomaly detection technique for IoT networks.

The Bi-GRU architecture builds upon the standard Gated Recurrent Unit (GRU), which manages sequential context memory within its hidden state through two gated components: the update gate and the reset gate. These gates regulate the flow of temporal information across long time series steps.

3.5.1. Update Gate (z_t)

The decision gate weighs a portion of the prior hidden state to be held over to the present hidden state and determine the amount of new candidate hidden state to be directly incorporated. It uses a sigmoid “activation” function to output values between 0 and 1, effectively acting as a filter.

$$z_t = \sigma(\mathbf{W}_z \mathbf{h}_{t-1} + \mathbf{U}_z [\mathbf{w}_t, \mathbf{v}_a] + \mathbf{b}_z) \quad (26)$$

3.5.2. Reset Gate (r_t)

The “reset gate decides how much of the previous hidden state to **forget**” [39]. A value close to 0 suggests forgetting most of the past information, while a value close to 1 suggests retaining it.

$$r_t = \sigma(\mathbf{W}_r \mathbf{h}_{t-1} + \mathbf{U}_r [\mathbf{w}_t, \mathbf{v}_a] + \mathbf{b}_r) \quad (27)$$

3.5.3. Candidate Hidden State ($\tilde{\mathbf{h}}_t$)

This state represents a potential new hidden state, calculated using a tanh activation function, incorporating the current input and the (reset-gated) previous hidden state.

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_h (\mathbf{h}_{t-1} \odot r_t) + \mathbf{U}_h [\mathbf{w}_t, \mathbf{v}_a] + \mathbf{b}) \quad (28)$$

3.5.4. Current Hidden State ($\mathbf{h}_t$)

The final hidden state at time t is a linear combination of the earlier hidden state and the candidate hidden state, measured by the update gate. This allows the GRU to smoothly transition between retaining old information and incorporating new information

$$\mathbf{h}_t = \mathbf{h}_{t-1} \odot (1 - z_t) + z_t \odot \tilde{\mathbf{h}}_t \quad (29)$$

Here, the variables are defined as $\mathbf{h}_{t-1}$: The hidden state vector from the previous time step. $\mathbf{w}_t$: The embedded vector of the current input word/feature at time t . $\mathbf{v}_a$: The aspect embedded vector, potentially representing contextual information. $\mathbf{W}_z, \mathbf{W}_r, \mathbf{W}_h$: Weight matrices for the update, reset and hidden state calculations, respectively, where $\mathbf{W}_z, \mathbf{W}_r, \mathbf{W}_h \in \mathbb{R}^4 \times \frac{4}{4}$ (hidden dimension by hidden dimension). $\mathbf{U}_z, \mathbf{U}_r, \mathbf{U}_h$: Weight matrices for the input-related components, where $\mathbf{U}_z, \mathbf{U}_r, \mathbf{U}_h \in \mathbb{R}^{d^1 \times (d_a + d_a)}$ (hidden dimension by concatenated input/aspect dimension). $\mathbf{b}_z, \mathbf{b}_r, \mathbf{b}$: Bias vectors for the update, reset and hidden state calculations, respectively, where $\mathbf{b}_z, \mathbf{b}_r, \mathbf{b} \in \mathbb{R}^d$ (hidden dimension). $\sigma(\cdot)$: The sigmoid activation function, used to control the preservation of effective data and removal of redundant data. $\tanh(\cdot)$: The hyperbolic tangent activation function. $\odot$ Element-wise multiplication. $[\mathbf{w}_t, \mathbf{v}_a]$: Denotes the splicing (concatenation) vector of the embedding word $\mathbf{w}_t$ and the embedding aspect $\mathbf{v}_a$. For Bidirectional GRU (Bi-GRU), the core idea is to process the sequence in two directions: a forward pass and a backward pass. This allows the model to capture dependencies from both past and future contexts simultaneously, which is critical for complex anomaly patterns that might be recognised only when considering the full temporal scope. The Bi-GRU involves separate hidden states for each direction at each time step.

3.5.5. Forward Hidden State ($\mathbf{h}_i$)

Calculated by processing the sequence from the beginning to the end using the GRU equations:

$$z_i = \sigma(\mathbf{W}_z^f \mathbf{h}_{i-1} + \mathbf{U}_z^f [\mathbf{w}_i, \mathbf{v}_a] + \mathbf{b}_z^f) \quad (30)$$

$$r_i = \sigma(\mathbf{W}_r^f \mathbf{h}_{i-1} + \mathbf{U}_r^f [\mathbf{w}_i, \mathbf{v}_a] + \mathbf{b}_r^f) \quad (31)$$

$$\tilde{\mathbf{h}}_i = \tanh(\mathbf{W}_h^f (\mathbf{h}_{i-1} \odot r_i) + \mathbf{U}_h^f [\mathbf{w}_i, \mathbf{v}_a] + \mathbf{b}^f) \quad (32)$$

$$\mathbf{h}_i = \mathbf{h}_{i-1} \odot (1 - z_i) + z_i \odot \tilde{\mathbf{h}}_i \quad (33)$$

3.5.6. Backward Hidden State ($\tilde{\mathbf{h}}_i$)

Calculated by processing the sequence from the end to the beginning, also using the GRU equations, but with separate weights:

$$\tilde{z}_i = \sigma(\mathbf{W}_z^b \tilde{\mathbf{h}}_{i-1} + \mathbf{U}_z^b [\mathbf{w}_i, \mathbf{v}_a] + \mathbf{b}_z^b) \quad (34)$$

$$\tilde{r}_i = \sigma(\mathbf{W}_r^b \tilde{\mathbf{h}}_{i-1} + \mathbf{U}_r^b [\mathbf{w}_i, \mathbf{v}_a] + \mathbf{b}_r^b) \quad (35)$$

$$\tilde{\tilde{\mathbf{h}}}_i = \tanh(\mathbf{W}_h^b (\tilde{\mathbf{h}}_{i-1} \odot \tilde{r}_i) + \mathbf{U}_h^b [\mathbf{w}_i, \mathbf{v}_a] + \mathbf{b}^b) \quad (36)$$

$$\tilde{\mathbf{h}}_i = \tilde{\mathbf{h}}_{i-1} \odot (1 - \tilde{z}_i) + \tilde{z}_i \odot \tilde{\tilde{\mathbf{h}}}_i \quad (37)$$

Where f and b indicate parameters (weights and biases) specific to the forward and backward passes, respectively. The term d_k characterises the number of hidden units in each layer.

3.5.7. Combined Contextual Hidden Representation (h_i^{final})

$$h_i^{\text{final}} = [h_i; \tilde{h}_i] \quad (38)$$

This final vector, $h_i^{\text{final}} \in \mathbb{R}^{2\%}$, encapsulates information from both preceding and succeeding elements in the sequence, offering a richer context for anomaly detection. This advanced anomaly classification method is crucial for effective threat mitigation and significantly bolsters the resilience of IoT systems against various malicious behaviours. The BiGRU trains our system for rapid classification of diverse attack types while maintaining the flexibility required to adapt to evolving threat patterns. The overall process of the Intelligent Multiverse Optimisation with Deep Learning Enabled Intrusion Detection System (IMVO-DLIDS), which incorporates this Bi-GRU based classification as depicted in Fig. 6, is designed to identify and classify intrusions efficiently within the IoT environment. The novelty of this approach, as mentioned, lies in the application of the Multiverse Optimisation (MVO) algorithm for fine-tuning the hyperparameters of the Bi-GRU model, further enhancing its performance. Algorithm 2 depicts the Anomaly classification in IoT.

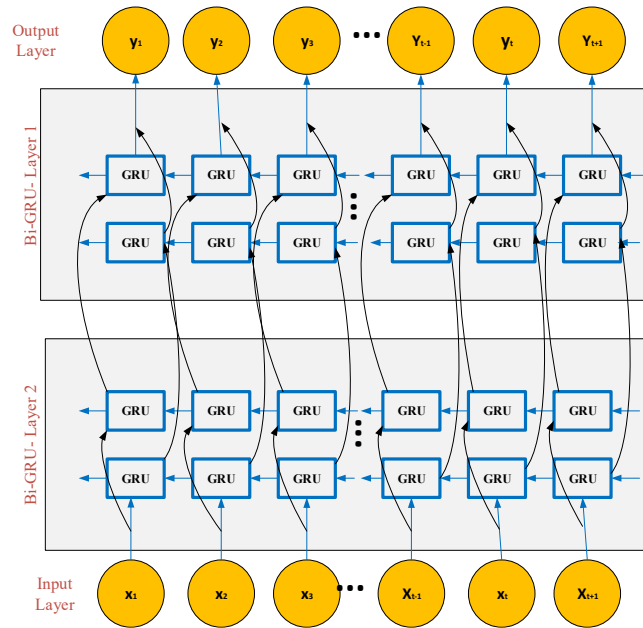


Figure 6. Structure of Bi-GRU

3.6. Anomaly classification in IoT

Routing is a critical part of the network that can greatly impact the performance as well as the stability of any dynamic Internet of Things scenario. Our model is able to address implementation issues due to conflicting objectives of multiple routing protocols by applying a decision-making method in which **game theory, linear programming and stochastic optimisation methods** are combined in an integrated framework named GTLPSO. However, Game Theory is the interplay of competing protocols, allowing optimal choice through the principles of strategic analysis. Linear programming is the optimal allocation of resources based on mathematical solutions of descriptions of network goals. Dynamic Routing System enhances the existing neural networks to address stochastic optimisation problems. By virtue of this holistic strategy, the framework is making intelligent routing decisions to achieve optimal performance while mitigating the disturbances introduced by protocol differences.

Algorithm 2. Anomaly classification in IoT Networks

Input: IoT data D, Labels Y, Epochs E, Population Size P, MVO Parameters

Output: Prediction Anomaly Labels Y

1. Initialise population of universes U_1 to U_p with random Bi-GRU parameters Θ
2. For epoch = 1 to E do
3. For each universe U_p in population do
4. Set $\Theta \leftarrow$ parameters from U_p
5. For each sample x_t in D do
6. Compute forward pass, h_t^f using GRU equations (30–33)
7. Compute backward pass h_t^b using GRU equations (34–37)
8. Concatenate $h_t^{\text{final}} = [h_t^f; h_t^b]$

```

9.      End for
10.     Prediction  $\hat{Y}$  using softmax on  $h_i^{\text{final}}$ 
11.     Compute fitness =  $\text{CrossEntropy}(Y, \hat{Y}) + \lambda \cdot \text{Reg}(\theta)$ 
12.   End for
13.   Update universes using MVO rules
14.   If fitness ( $U_{\text{best}}$ ) < threshold then
15.     Break training
16.   Else
17.     Continue optimisation
18.   End if
19. End for
20. Return final prediction  $\hat{Y}$  from best universe

```

3.6.1. Game Theory for Strategic Protocol Interplay

Game theory analyses the interplay between competing routing protocols or network entities, where each participant (e.g., a routing node or a protocol) aims to maximise its own utility or achieve its specific objective. It allows for the identification of optimal choices by applying principles of strategic analysis. In the context of routing, this helps in understanding how different protocols or nodes might behave under various network conditions, particularly when their objectives conflict (e.g., one protocol aims for minimal latency and another for energy efficiency). By modelling these interactions as a game, the framework can prediction stable states, or "equilibria", where participants do not have any incentive to unilaterally change its strategy. The utility or payoff function for a player (e.g., a routing protocol instance or node) i choosing a strategy s_i when other players choose s_{-i} (all strategies except s_i) can be expressed as:

$$U_i(s_i, s_{-i}) \quad (39)$$

Here U_i : Represents the utility or benefit that Player i receives. s_i is the strategy or action chosen by Player i . s_{-i} : Denotes the strategies chosen by all other players in the network. A key concept in game theory for finding stable solutions is the Nash Equilibrium, where no player can improve their outcome by changing their strategy while others keep theirs constant. This state for each player i satisfies:

$$U_i(s_i^*, s_{-i}^*) \geq U_i(s_i, s_{-i}^*) \forall s_i \neq s_i^* \quad (40)$$

Where s_i^* : The optimal strategy for Player i at equilibrium. s_{-i}^* : The optimal strategies for all other players at equilibrium.

3.6.2. Linear Programming for Optimal Resource Allocation

Linear Programming (LP) is applied to optimally allocate network resources based on mathematical descriptions of network goals, typically focusing on minimising costs (e.g., energy consumption) or maximising benefits (e.g., throughput, network lifetime) under a set of linear constraints. This component translates the network's objectives and limitations (such as bandwidth availability, energy budget of devices, or node capacity) into a solvable mathematical problem. It determines the best routing paths or resource assignments to achieve the desired outcome. A general linear programming objective function often aims to either minimise or maximise a linear combination of decision variables. For a minimisation problem (e.g., energy consumption):

$$\text{Minimise } C_1x_1 + C_2x_2 + \dots + C_nx_n \quad (41)$$

And for a maximisation problem (e.g., network utility or throughput):

$$\text{Maximise } D_1y_1 + D_2y_2 + \dots + D_my_m \quad (42)$$

Where C_j, D_j : Are coefficients representing costs or benefits, x_j, y_j : Are decision variables (e.g., amount of data routed through a path, activation state of a node). These objectives are subject to linear constraints, which define the boundaries within which the solution must operate:

- *Equality Constraints*: Represent exact resource requirements or balances (e.g., total incoming traffic must equal total outgoing traffic at a node).

$$A_{eq}\mathbf{x} = \mathbf{B}_{eq} \quad (43)$$

$$A_{ineq}\mathbf{x} \leq \mathbf{B}_{ineq} \quad (44)$$

- *Variable Bounds*: Define the permissible range for each decision variable (e.g., a flow rate cannot be negative).

$$\mathbf{L} \leq \mathbf{x} \leq \mathbf{U} \quad (45)$$

Here, $\mathbf{x}$ is a vector of decision variables, A_{eq} and A_{ineq} are matrices of coefficients, $\mathbf{B}_{eq}$ and $\mathbf{B}_{ineq}$ are vectors of constant terms and $\mathbf{L}$ and $\mathbf{U}$ are lower and upper bounds for $\mathbf{X}$.

3.6.3. Stochastic Optimisation for Dynamic and Random Traffic Patterns

The Dynamic Routing System within GTLPSO improves the performance of traditional neural networks for stochastic optimisation problems. In IoT deployments, network conditions (traffic patterns, link quality and availability of nodes) are largely uncertain in behaviours and they exhibit random variations. Stochastic optimisation algorithms are designed to optimise decision-making problems under uncertainty. The stochastic model incorporated into the subsystem is essentially a probabilistic model to foresee future network status or traffic loads, to help predict network conditions so the routing framework can react accordingly. This helps to make strong decisions that on average will perform well, even when inputs are uncertain. In stochastic optimisation problems, one must minimise the expected value of a certain cost function on random variables:

$$\text{Minimise } E[C(\mathbf{x}, \xi)] \quad (46)$$

Where $E[\cdot]$: is the expected value (an average outcome over many random happenings), $C(\mathbf{x}, \xi)$: is the cost function which depends on decision variables $\mathbf{x}$ and a random variable ξ (i.e., unpredictable traffic volume). A concrete instance from related work, such as GPR, used for traffic prediction is considered to be probabilistic forecast. The mean prediction for a new input x^* (e.g., future traffic conditions) in GPR is:

$$\mu(x^*) = k(x^*, X)(K(X, X) + \sigma_n^2 I)^{-1} y \quad (47)$$

And the variance prediction at x^* is:

$$\text{Var}(x^*) = k(x^*, x^*) - k(x^*, X)(K(X, X) + \sigma_n^2 I)^{-1} k(X, x^*)^T \quad (48)$$

Here $\mu(x^*)$: The predicted average (mean) value for a new, unseen input x^* . $\text{Var}(x^*)$: The **predicted uncertainty (variance)** around that mean prediction for x^* . X : The **matrix of historical input data**. y : The **vector of corresponding historical output values**. $k(x^*)$: A **kernel function** that measures similarity between data points. $K(X, X)$: The **kernel matrix** computed for all pairs of historical data points. σ_n^2 : The **noise variance** in the observations. I : The **identity matrix**. The issues of instability, packet delivery performance and random traffic patterns are addressed in one top level with GTLPSO. Algorithm 3 represents the routing driven by Decision-making.

As a result of reconciling GTLPSO, the framework stabilises the network, increases packet delivery effectiveness and brings down random traffic pattern effects. In diverse operational conditions, keeping systems durable and scalable with IoT is primarily a routing process.

4. Experimental Analysis

The experimental examination of the proposed MQTT traffic-based secure communication in IoT environments is presented in this section. The discussion covers the comparative analysis, study report and simulation setup, etc. The results show that the suggested method effectively reduces the security-based limitation.

4.1. Simulation setup

The MQTT traffic-based secure communication in the IoT environment is described in this section along with the simulation setup. The system setup is shown in Table 2.

Fig. 7 is the output from an ns-3 platform that simulates an IoT network. The nodes appear to be communicating with one another while the base stations are connecting to a server. The illustration shows connectivity (and related data flow) coverage within the IoT network. From an analytical perspective, this demonstration can provide some insights into the network performance or device interactions.

Algorithm 3. Routing driven by Decision-making

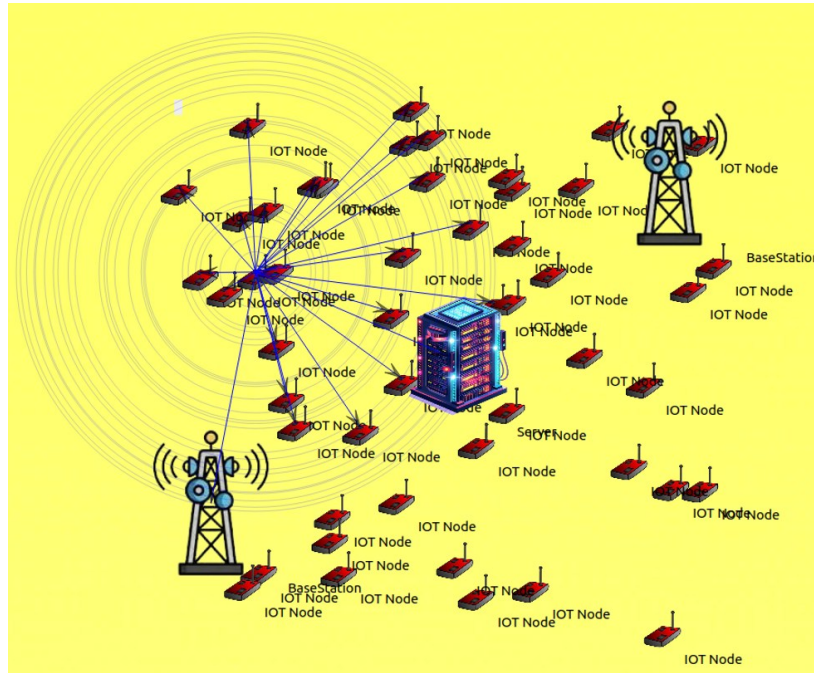
Input: T (Topology), ξ (Traffic Load), S (Protocol Set), C (Constraints)
Output: R_opt(Optimised Routing Path)

1. Initialise strategies s_i for each protocol $i \in S$
2. Compute utility $U_i(s_i, s_{-i}) \leftarrow [\text{Eq. (40)}]$
3. If $\forall s_i \neq s_i^*, U_i(s_i^*, s_{-i}^*) \geq U_i(s_i, s_{-i})$ $\leftarrow$ [Nash Eq. (40)]
4. Solve Linear Programming using Eq. (41–45)

5. Predict traffic using GPR: $\mu(x^*), \text{Var}(x^*) \leftarrow [\text{Eq. (47-48)}]$
6. Compute expected cost: $E[C(x, \xi)] \leftarrow [\text{Eq. (46)}]$
7. If $\mu(x^*) < \text{Threshold AND } E[C(x, \xi)] \text{ minimal:}$
8. Return $R_{\text{opt}} \leftarrow \text{Optimal routing path with stable strategy}$
9. Else: Update $s_i \leftarrow \text{Best Response and repeat from Step 2}$
10. End If

Table 2. System Specifications

Hardware specifications	Hard disk	100 GB
	RAM	4-8 GB
Software specifications	Simulation tools	NS 3.30 or above with Python
	OS	Ubuntu 18.04 LTS (64-bit) or above

**Figure 7.** IoT network simulation outcomes using NS-3.30

4.2. Comparative Analysis

In this section, the proposed method is compared with other existing approaches, including the Game Theory Approach (GTA) [32], Convolutional Neural Network-Autoencoder (CNN-A) (Berraadi *et al.* [41], Fully Homomorphic Encryption (FHE) (Tariq *et al.*, 2023), and Trusted Execution Environment (TEE) [38] methods, in order to assess its performance. Finally, we plot performance for the following metrics: Number of Epochs vs. Accuracy (%), Number of Epochs vs. Precision (%), Number of IoT Nodes vs. Latency (ms), Number of IoT Nodes vs. Prediction Time (s), and Number of IoT Nodes vs. Throughput (bytes).

4.2.1. Accuracy

Accuracy quantifies the ratio of correctly identified data instances to the total evaluated data. It reflects the overall reliability of the classification process in detecting both normal and anomalous patterns

$$\mathcal{A} = \frac{\mathcal{X}_c}{\mathcal{X}_T} \quad (49)$$

- $\mathcal{A}$: Accuracy (%)
- $\mathcal{X}_c$ is the number of correctly classified instances, and
- $\mathcal{X}_T$ is the total number of evaluated data instances

The proposed model shows a steady increase in accuracy from 86% to 98% as training epochs increase from 10 to 50, indicating strong anomaly-detection capability with minimal degradation at scale. By comparison, GTA improves from 85% to 90%, and CNN-A reaches only 88%, showing that the proposed model outperforms both alternatives in learning generalised attack patterns. This high accuracy reflects the effective integration of PCA-GA-UST feature selection and the Bi-GRU classifier. The model maintains stability in decision boundaries even under varying network loads, proving its robustness, as shown in Fig. 8 and Table 3.

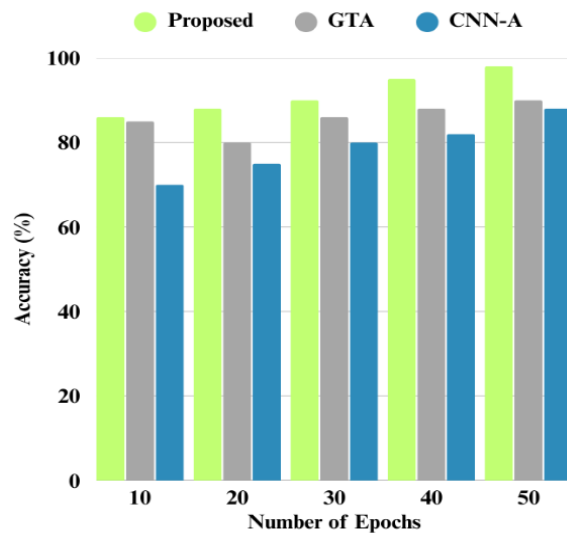


Figure 8. Accuracy

Table 3. Numerical outcomes of Accuracy

No. of epochs	Proposed	GTA	CNN-A
10	86	85	70
20	88	80	75
30	90	86	80
40	95	88	82
50	98	90	88

The proposed model shows a consistent increase in accuracy from 86% to 98% as the number of training epochs increases from 10 to 50, indicating strong anomaly-detection capability with minimal degradation at scale. Comparatively, GTA improves from 85% to 90%, and CNN-A only reaches 88%, showing that the proposed model outperforms both alternatives in learning generalised attack patterns. This high accuracy reflects the effective integration of PCA-GA-UST feature selection and the Bi-GRU classifier. The model maintains stability in decision boundaries even under varying network loads, proving its robustness, as shown in Fig. 8 and Table 3.

4.2.2. Precision

Precision measures how many of the positively identified anomalies were actually true anomalies. It helps minimise unnecessary alerts in resource-constrained IoT systems.

$$\mathcal{P} = \left(\frac{\eta_e}{\eta_\theta} \right) \times 100 \quad (50)$$

- $\mathcal{P}$ is the precision (%),
- η_e is the actual anomaly instances correctly flagged, and
- η_θ is the total predicted anomaly instances.

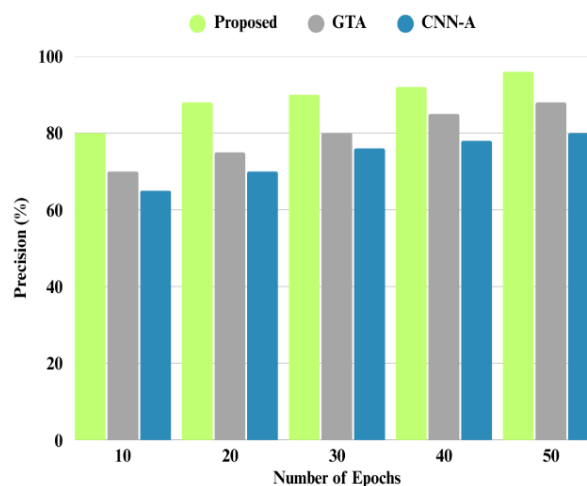


Figure 9. Precision

Table 4. Numerical outcomes of Precision

Number of Epochs	Proposed	GTA	CNN-A
10	80	70	65
20	88	75	70
30	90	80	76
40	92	85	78
50	96	88	80

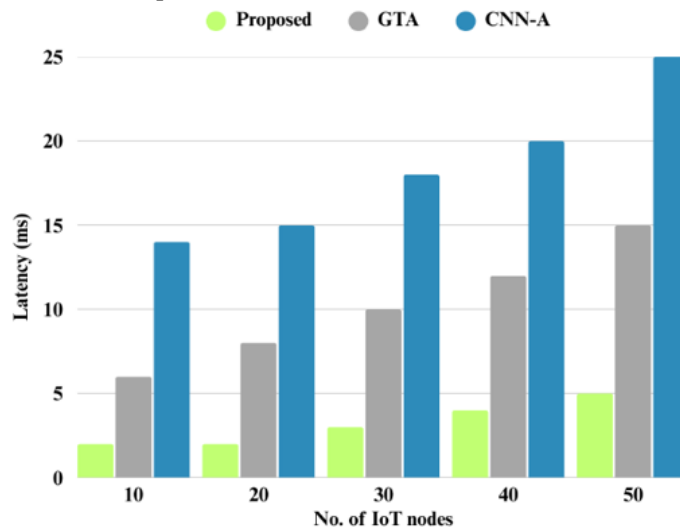
Precision, which measures the true relevancy of positive predictions, improves from **80% to 96%** for the proposed model, confirming its ability to reduce false alarms and increase detection exactness. GTA and CNN-A show relatively modest improvements, reaching only **88% and 80%**, respectively. The optimisation of MQTT parsing and noise filtration through CGAN enhances the true positive rate significantly. The proposed architecture thus ensures low false-positive rates and accurate target classification even under dense traffic conditions in table 4 and fig. 9.

4.2.3. Latency

Latency represents the time delay experienced between a request generated by an IoT device and the response received. Low latency supports real-time security decisions.

$$\mathcal{L} = \mathcal{T}_r - \mathcal{T}_s \quad (51)$$

- $\mathcal{L}$ is the latency in milliseconds,
- $\mathcal{T}_r$ response timestamp and
- $\mathcal{T}_s$ signal (or request) timestamp.

**Figure 10.** Latency**Table 5:** Numerical outcomes of latency

IoT Nodes	Proposed	GTA	CNN-A
10	2	6	14
20	2	8	15
30	3	10	18
40	4	12	20
50	5	15	25

The latency for the proposed system ranges from **2 ms to 5 ms**, showing ultra-low delay performance across all node densities, which is crucial for real-time IoT threat detection. In contrast, GTA and CNN-A experience latency increases up to **15 ms and 25 ms**, respectively, highlighting their inefficiencies under network stress. The incorporation of edge computing and lightweight CGAN preprocessing contributes to reduced processing time. The result supports real-time processing demands for smart IoT environments without compromising responsiveness, as presented in fig. 10 and table 5.

4.2.4. Prediction Time

Prediction time refers to the computation duration required by the detection model to classify a given instance. Lower values improve system responsiveness in dense IoT networks.

$$T_p = \frac{\sum_{i=1}^n \Delta_i}{n} \quad (52)$$

- T_p : Average prediction time (seconds),
- Δ_i : Time taken for classification of the i data sample and
- n : Total number of data samples.

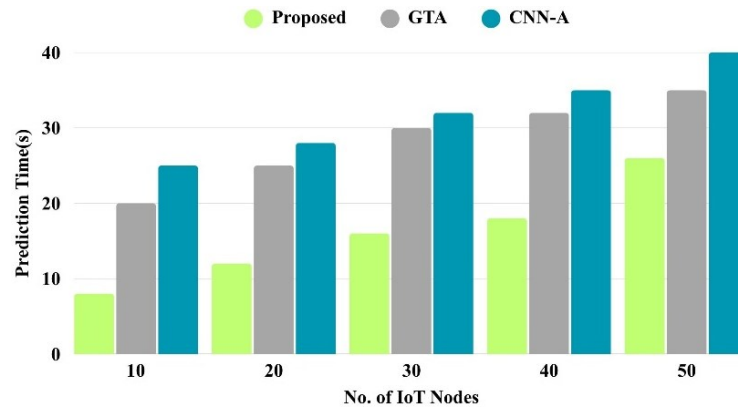


Figure 11. Prediction Time

Table 6. Numerical outcomes of the prediction time

IoT Nodes	Proposed	GTA	CNN-A
10	8	20	25
20	12	25	28
30	16	30	32
40	18	32	35
50	26	35	40

Prediction time in the proposed model gradually increases from **8s to 26s** as the network scales, significantly outperforming GTA (up to 35s) and CNN-A (up to 40s). This improvement stems from the optimised inference path facilitated by the Bi-GRU classifier and GTLPPO routing. The shallow architecture of ELM and the deep optimisation pipeline contribute to faster convergence during classification. This low prediction time makes the framework suitable for latency-sensitive IoT systems, such as smart healthcare and industrial automation, as shown in fig. 11 and table 6.

4.2.5. Throughput

Throughput denotes the amount of useful data successfully processed and transferred per unit time. It reflects the communication capacity of the IoT network.

$$\theta = \frac{\Omega_d}{\Omega_t} \quad (53)$$

- θ : Throughput (bytes/second),
- Ω_d : Total volume of data transmitted (bytes) and
- Ω_t : Time taken for transmission (seconds).

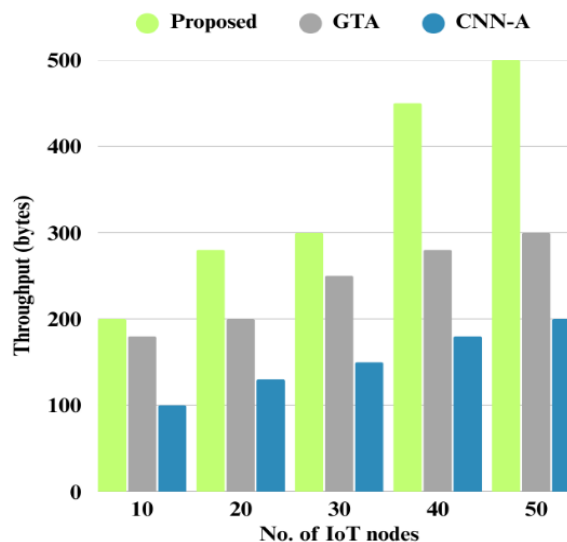


Figure 12. Throughput

Table 7. Numerical Outcomes of Throughput

IoT Nodes	Proposed	GTA	CNN-A
10	200	180	100
20	280	200	130
30	300	250	150
40	450	280	180
50	500	300	200

The throughput of the proposed model scales from **200 bytes to 500 bytes**, demonstrating efficient data processing and minimal packet drops in high-load environments. GTA and CNN-A lag behind, reaching only **300 and 200 bytes**, respectively, indicating lower transmission capacity. The high throughput is attributed to the MQTT optimisation and effective resource allocation by GTLPSO. This result verifies the framework's ability to manage dense data streams while preserving communication quality and system throughput, as shown in Fig. 12 and table 7.

4.3. Research summary

This research work implements a scalable AI-based approach to solving anomalies and optimising MQTT network traffic in an IoT environment. The CICIDS2017 dataset incorporates normal traffic and both benign and malicious (network security attackers) traffic to train the method. A Conditional Generative Adversarial Network (CGAN) addresses class imbalance and data augmentation issues. The hybrid approach-based method (PCA-GA-UST-ELM) provides an efficient feature selection and dimensionality reduction method. The MQTT parsing engine and supporting Deep Neural Network (DNN) offers real-time anomaly detection with low latency. A Bi-GRU model is used for learning temporal dependencies for classification and detecting anomalies. Our routing approach uses the Game Theory, Linear Programming and Stochastic Optimisation (GTLPSO) paradigm to optimise the IoT MQTT route. The experimental results demonstrate better accuracy and precision, low latency, more complete prediction time and better throughput.

5. Conclusion

This research has proposed a new scalable AI-based framework to secure IoT networks through optimisation of MQTT traffic and detection of anomaly. By utilising the CICIDS2017 dataset, CGAN was established to balance/maintain class distribution and enhance generalisation performance. The hybrid feature selection (PCA-GA-UST-ELM) allocation successfully dropped dimensions and computation time. DNN + MQTT parsing reduced overall latency and broker overload. GTLPSO improved routing stability by resolving uncertainty in multiple protocols. Performance analytics confirmed that accuracy (up to 98%), precision (96%) and latency (2-5 ms) experienced significant gains. The proposed framework had lower prediction times, larger throughput and efficiency with growing node density. This security architecture is computationally light and can support secure, real-time use cases within constrained IoT environments. This system effectively addresses important major weaknesses in previous approaches, i.e., class imbalance, feature selection and protocol uncertainty. The findings provide validation of the capacity of the proposed end-to-end detection and optimisation system. Therefore, it can be concluded that the research established the groundwork to build a scalable, intelligent IoT security framework.

Future improvements may include extending the work to analyse other IoT protocols such as CoAP and integrating blockchain-based trust management protocols. Additionally, we will evaluate the improvement of model generalisation through adaptive hyperparameter tuning and reinforcement learning.

CRediT Author Contribution Statement

Emran Aljarrah: Conceptualisation, Methodology, Writing - Original Draft; Bilal Bataineh: Investigation, Data Curation; Saleem Alzoubi: Validation, Resources, Project administration; Md Nasiruzzaman: Formal Analysis, Visualisation; Mahdi H. Miraz: Supervision, Writing - Review & Editing.

Acknowledgements

The authors are truly grateful to Jadara University for the support and encouragement during my research. Their financial assistance, resources, guidance, and encouragement provided by the university played a significant role in the successful completion of this work.

References

- [1] Md. Milon Islam, Sheikh Nooruddin, Fakhri Karray and Ghulam Muhammad, "Internet of Things: Device Capabilities, Architectures, Protocols, and Smart Applications in Healthcare Domain", *IEEE Internet of Things Journal*, 15 February 2023, Vol. 10, No. 4, pp. 3611–3641, Published by IEEE, DOI: 10.1109/JIOT.2022.3228795, Available: <https://ieeexplore.ieee.org/document/9983826/>.
- [2] Pavanalaxmi, Kripa, Roopashree Nayak, A. B. Abhishek and M. Praveen Kumar, "Integrating Machine Learning and IoT: Pioneering Solutions for Sustainable Smart Cities", In *Applications of Computational Learning and IoT in Smart Road Transportation System*, Springer Cham, 2025, Print ISBN: 978-3-031-87626-4, Online ISBN: 978-3-031-87627-1, Vol. 22, ch. 3, pp. 37–54, DOI: 10.1007/978-3-031-87627-1_3, Available: https://link.springer.com/chapter/10.1007/978-3-031-87627-1_3.
- [3] Usman Tariq, Irfan Ahmed, Ali Kashif Bashir and Kamran Shaukat, "A Critical Cybersecurity Analysis and Future Research Directions for the Internet of Things: A Comprehensive Review", *Sensors*, 19 April 2023, Vol. 23, No. 8, Article No. 4117, Published by MDPI, DOI: 10.3390/S23084117, Available: <https://www.mdpi.com/1424-8220/23/8/4117>.
- [4] Princy Victor, Arash Habibi Lashkari, Rongxing Lu, Tinshu Sasi, Pulei Xiong *et al.*, "IoT malware: An attribute-based taxonomy, detection mechanisms and challenges", *Peer-to-Peer Networking and Applications*, 10 May 2023, Vol. 16, No. 3, pp. 1380–1431, Published by Springer, DOI: 10.1007/S12083-023-01478-W, Available: <https://link.springer.com/article/10.1007/s12083-023-01478-w>.
- [5] Raihan Uddin and Insoo Koo, "Real-Time Remote Patient Monitoring: A Review of Biosensors Integrated with Multi-Hop IoT Systems via Cloud Connectivity", *Applied Sciences*, 25 February 2024, Vol. 14, No. 5, Article No. 1876, Published by MDPI, DOI: 10.3390/APP14051876, Available: <https://www.mdpi.com/2076-3417/14/5/1876>.
- [6] Hanane Alloui and Youssef Mourdi, "Exploring the Full Potentials of IoT for Better Financial Growth and Stability: A Comprehensive Survey", *Sensors*, 22 September 2023, Vol. 23, No. 19, Article No. 8015, Published by MDPI, DOI: 10.3390/S23198015, Available: <https://www.mdpi.com/1424-8220/23/19/8015>.
- [7] George Vardakis, George Hatzivasilis, Eleftheria Koutsaki and Nikos Papadakis, "Review of Smart-Home Security Using the Internet of Things", *Electronics*, 22 August 2024, Vol. 13, No. 16, Article No. 3343, Published by MDPI, DOI: 10.3390/ELECTRONICS13163343, Available: <https://www.mdpi.com/2079-9292/13/16/3343>.
- [8] Haitham Ameen Noman and Osama M. F. Abu-Sharkh, "Code Injection Attacks in Wireless-Based Internet of Things (IoT): A Comprehensive Review and Practical Implementations", *Sensors*, 30 June 2023, Vol. 23, No. 13, Article No. 6067, Published by MDPI, DOI: 10.3390/S23136067, Available: <https://www.mdpi.com/1424-8220/23/13/6067>.
- [9] Ömer Aslan, Semih Serkant Aktuğ, Merve Ozkan-Okay, Abdullah Asim Yilmaz and Erdal Akin, "A Comprehensive Review of Cyber Security Vulnerabilities, Threats, Attacks, and Solutions", *Electronics*, 11 March 2023, Vol. 12, No. 6, Article No. 1333, Published by MDPI, DOI: 10.3390/ELECTRONICS12061333, Available: <https://www.mdpi.com/2079-9292/12/6/1333>.
- [10] Muhammad Husnain, Khizar Hayat, Enrico Cambiaso, Ubaid U. Fayyaz, Maurizio Mongelli *et al.*, "Preventing MQTT Vulnerabilities Using IoT-Enabled Intrusion Detection System", *Sensors*, 12 January 2022, Vol. 22, No. 2, Article No. 567, Published by MDPI, DOI: 10.3390/S22020567, Available: <https://www.mdpi.com/1424-8220/22/2/567>.
- [11] Swastik Kumar Sahu and Kaushik Mazumdar, "Exploring security threats and solutions techniques for Internet of Things (IoT): from vulnerabilities to vigilance", *Frontiers in Artificial Intelligence*, 15 May 2024, Vol. 7, Article No. 1397480, Published by Frontiers Media, DOI: 10.3389/frai.2024.1397480, Available: <https://www.frontiersin.org/journals/artificial-intelligence/articles/10.3389/frai.2024.1397480/full>.
- [12] Xiaotao Feng, Xiaogang Zhu, Qing-Long Han, Wei Zhou, Sheng Wen *et al.*, "Detecting Vulnerability on IoT Device Firmware: A Survey", *IEEE/CAA Journal of Automatica Sinica*, January 2023, Vol. 10, No. 1, pp. 25–41, Published by IEEE, DOI: 10.1109/JAS.2022.105860, Available: <https://ieeexplore.ieee.org/document/9878283/>.
- [13] Taimur Bakhshi, Bogdan Ghita and Ievgeniia Kuzminykh, "A Review of IoT Firmware Vulnerabilities and Auditing Techniques", *Sensors*, 22 January 2024, Vol. 24, No. 2, Article No. 708, Published by MDPI, DOI: 10.3390/S24020708, Available: <https://www.mdpi.com/1424-8220/24/2/708>.
- [14] Sarah Bin Hulayyil, Shancang Li and Lida Xu, "Machine-Learning-Based Vulnerability Detection and Classification in Internet of Things Device Security", *Electronics*, 18 September 2023, Vol. 12, No. 18, Article No. 3927, Published by MDPI, DOI: 10.3390/ELECTRONICS12183927, Available: <https://www.mdpi.com/2079-9292/12/18/3927>.

- [15] Rana Alasmari and Areej Abdullah Alhogail, "Protecting Smart-Home IoT Devices From MQTT Attacks: An Empirical Study of ML-Based IDS", *IEEE Access*, 2024, Vol. 12, pp. 25993–26004, Published by IEEE, DOI: 10.1109/ACCESS.2024.3367113, Available: <https://ieeexplore.ieee.org/document/10439191/>.
- [16] Albara Awajan, "A Novel Deep Learning-Based Intrusion Detection System for IoT Networks", *Computers*, 5 February 2023, Vol. 12, No. 2, Article No. 34, Published by MDPI, DOI: 10.3390/COMPUTERS12020034, Available: <https://www.mdpi.com/2073-431X/12/2/34>.
- [17] Ahmed J. Hintaw, Selvakumar Manickam, Mohammed Faiz Aboalmaaly and Shankar Karuppayah, "MQTT Vulnerabilities, Attack Vectors and Solutions in the Internet of Things (IoT)", *IETE Journal of Research*, 4 May 2021, Vol. 69, No. 6, pp. 3368–3397, Published by Taylor & Francis, DOI: 10.1080/03772063.2021.1912651, Available: <https://www.tandfonline.com/doi/full/10.1080/03772063.2021.1912651>.
- [18] Inês Martins, João S. Resende, Patrícia R. Sousa, Simão Silva, Luís Antunes *et al.*, "Host-based IDS: A review and open issues of an anomaly detection system in IoT", *Future Generation Computer Systems*, August 2022, Vol. 133, pp. 95–113, Published by Elsevier, DOI: 10.1016/J.FUTURE.2022.03.001, Available: <https://www.sciencedirect.com/science/article/abs/pii/S0167739X22000760?via%3Dihub>.
- [19] Gianluca Iadicicco, "Development and Optimization of a Firmware for Real-Time Monitoring of Network Traffic and Threat Detection", *WebThesis*, 2025, pp. 1-124, published by Politecnico di Torino, Available: <https://webthesis.biblio.polito.it/35239/>
- [20] José Roldán-Gómez, Javier Carrillo-Mondéjar, Juan Manuel Castelo Gómez and Sergio Ruiz-Villafranca, "Security Analysis of the MQTT-SN Protocol for the Internet of Things", *Applied Sciences*, 30 October 2022, Vol. 12, No. 21, Article No. 10991, Published by MDPI, DOI: 10.3390/APP122110991, Available: <https://www.mdpi.com/2076-3417/12/21/10991>.
- [21] Tanzila Saba, Amjad Rehman, Tariq Sadad, Hoshang Kolivand and Saeed Ali Bahaj, "Anomaly-based intrusion detection system for IoT networks through deep learning model", *Computers and Electrical Engineering*, April 2022, Vol. 99, Article No. 107810, Published by Elsevier, DOI: 10.1016/J.COMPELECENG.2022.107810, Available: <https://www.sciencedirect.com/science/article/abs/pii/S0045790622001100?via%3Dihub>.
- [22] My Driss Laanaoui, Mohamed Lachgar, Hanine Mohamed, Hrimech Hamid, Santos Gracia Villar *et al.*, "Enhancing Urban Traffic Management Through Real-Time Anomaly Detection and Load Balancing", *IEEE Access*, 2024, Vol. 12, pp. 63683–63700, Published by IEEE, DOI: 10.1109/ACCESS.2024.3393981, Available: <https://ieeexplore.ieee.org/document/10508783>.
- [23] Lu Lihua, "Energy-Aware Intrusion Detection Model for Internet of Vehicles Using Machine Learning Methods", *Wireless Communications and Mobile Computing*, 26 May 2022, Vol. 2022, Article No. 9865549, Published by Hindawi, DOI: 10.1155/2022/9865549, Available: <https://onlinelibrary.wiley.com/doi/10.1155/2022/9865549>.
- [24] Aulia Arif Wardana, Grzegorz Kołaczek and Parman Sukarno, "Lightweight, Trust-Managing, and Privacy-Preserving Collaborative Intrusion Detection for Internet of Things", *Applied Sciences*, 12 May 2024, Vol. 14, No. 10, Article No. 4109, Published by MDPI, DOI: 10.3390/APP14104109, Available: <https://www.mdpi.com/2076-3417/14/10/4109>.
- [25] Ibrahim Mutambik, "An Efficient Flow-Based Anomaly Detection System for Enhanced Security in IoT Networks", *Sensors*, 20 November 2024, Vol. 24, No. 22, Article No. 7408, Published by MDPI, DOI: 10.3390/S24227408, Available: <https://www.mdpi.com/1424-8220/24/22/7408>.
- [26] Maria Balega, Waleed Farag, Xin-Wen Wu, Soundararajan Ezekiel and Zaryn Good, "Enhancing IoT Security: Optimizing Anomaly Detection through Machine Learning", *Electronics*, 31 May 2024, Vol. 13, No. 11, Article No. 2148, Published by MDPI, DOI: 10.3390/ELECTRONICS13112148, Available: <https://www.mdpi.com/2079-9292/13/11/2148>.
- [27] A. Chohra, Paria Shirani, ElMouatez Billah Karbab and Mourad Debbabi, "CHAMELEON: Optimized feature selection using particle swarm optimization and ensemble methods for network anomaly detection", *Computers and Security*, June 2022, Vol. 117, Article No. 102684, Published by Elsevier, DOI: 10.1016/J.COSE.2022.102684, Available: <https://www.sciencedirect.com/science/article/abs/pii/S0167404822000827?via%3Dihub>.
- [28] Shalaka Mahadik, Pranav M. Pawar and Raja Muthalagu, "Efficient Intelligent Intrusion Detection System for Heterogeneous Internet of Things (HetIoT)", *Journal of Network and Systems Management*, 6 October 2022, Vol. 31, No. 1, Article No. 2, Published by Springer, DOI: 10.1007/S10922-022-09697-X, Available: <https://link.springer.com/article/10.1007/s10922-022-09697-x>.
- [29] Joseph Bamidele Awotunde, Femi Emmanuel Ayo, Ranjit Panigrahi, Amik Garg, Akash Kumar Bhoi *et al.*, "A Multi-level Random Forest Model-Based Intrusion Detection Using Fuzzy Inference System for Internet of Things Networks", *International Journal of Computational Intelligence Systems*, 12 March 2023, Vol. 16, No. 1, Article No. 31, Published by Springer, DOI: 10.1007/s44196-023-00205-w, Available: <https://link.springer.com/article/10.1007/s44196-023-00205-w>.
- [30] Keerthi Kethineni and Pradeepini Gera, "IoT-Based Privacy-Preserving Anomaly Detection Model for Smart Agriculture", *Systems*, 13 June 2023, Vol. 11, No. 6, Article No. 304, Published by MDPI, DOI: 10.3390/SYSTEMS11060304, Available: <https://www.mdpi.com/2079-8954/11/6/304>.

- [31] Bosen Lian, Yusuf Kartal, Frank L. Lewis, Dariusz G. Mikulski, Gregory R. Hudas *et al.*, "Anomaly Detection and Correction of Optimizing Autonomous Systems With Inverse Reinforcement Learning", *IEEE Transactions on Cybernetics*, July 2023, Vol. 53, No. 7, pp. 4555–4566, Published by IEEE, DOI: 10.1109/TCYB.2022.3213526, Available: <https://ieeexplore.ieee.org/document/9925982>.
- [32] Asima Sarwar, Abdullah M. Alnajim, Safdar Nawaz Khan Marwat, Salman Ahmed, Saleh Alyahya *et al.*, "Enhanced Anomaly Detection System for IoT Based on Improved Dynamic SBPSO", *Sensors*, 29 June 2022, Vol. 22, No. 13, Article No. 4926, Published by MDPI, DOI: 10.3390/S22134926, Available: <https://www.mdpi.com/1424-8220/22/13/4926>.
- [33] Mohammed Zakariah and Abdulaziz S. Almazyad, "Anomaly Detection for IOT Systems Using Active Learning", *Applied Sciences*, 4 November 2023, Vol. 13, No. 21, Article No. 12029, Published by MDPI, DOI: 10.3390/APP132112029, Available: <https://www.mdpi.com/2076-3417/13/21/12029>.
- [34] William Villegas-Ch, Jamie Govea and Angel Jaramillo-Alcazar, "IoT Anomaly Detection to Strengthen Cybersecurity in the Critical Infrastructure of Smart Cities", *Applied Sciences*, 5 October 2023, Vol. 13, No. 19, Article No. 10977, Published by MDPI, DOI: 10.3390/APP131910977, Available: <https://www.mdpi.com/2076-3417/13/19/10977>.
- [35] J. Manokaran and G. Vairavel, "IGWO-SoE: Improved Grey Wolf Optimization Based Stack of Ensemble Learning Algorithm for Anomaly Detection in Internet of Things Edge Computing", *IEEE Access*, 2023, Vol. 11, pp. 106934–106953, Published by IEEE, DOI: 10.1109/ACCESS.2023.3319814, Available: <https://ieeexplore.ieee.org/document/10265035>.
- [36] Tripti Sharma and Sanjeev Kumar Prasad, "Enhancing cybersecurity in IoT networks: SLSTM-WCO algorithm for anomaly detection", *Peer-to-Peer Networking and Applications*, 3 May 2024, Vol. 17, No. 4, pp. 2237–2258, Published by Springer, DOI: 10.1007/S12083-024-01712-Z, Available: <https://link.springer.com/article/10.1007/s12083-024-01712-z>.
- [37] Abdulaziz A. Alsulami, Qasem Abu Al-Haija, Ahmad Tayeb and Ali Alqahtani, "An Intrusion Detection and Classification System for IoT Traffic with Improved Data Engineering", *Applied Sciences*, 2 December 2022, Vol. 12, No. 23, Article No. 12336, Published by MDPI, DOI: 10.3390/APP122312336, Available: <https://www.mdpi.com/2076-3417/12/23/12336>.
- [38] Saad Gadai, Rania Mokhtar, Maha Abdelhaq, Raed Alsaqour, Elmustafa Sayed Ali *et al.*, "Machine Learning-Based Anomaly Detection Using K-Mean Array and Sequential Minimal Optimization", *Electronics*, 10 July 2022, Vol. 11, No. 14, Article No. 2158, Published by MDPI, DOI: 10.3390/ELECTRONICS11142158, Available: <https://www.mdpi.com/2079-9292/11/14/2158>.
- [39] Rajasekhar Chaganti, Wael Suliman, Vinayakumar Ravi and Amit Dua, "Deep Learning Approach for SDN-Enabled Intrusion Detection System in IoT Networks", *Information*, 9 January 2023, Vol. 14, No. 1, Article No. 41, Published by MDPI, DOI: 10.3390/INFO14010041, Available: <https://www.mdpi.com/2078-2489/14/1/41>.
- [40] Shafiullah Khan, Muhammad Muneer Umar, Chunhua Jin, Shaozhang Xiao, Zeeshan Iqbal *et al.*, "Game-Theory-Based Multimode Routing Protocol for Internet of Things", *Electronics*, 12 December 2022, Vol. 11, No. 24, Article No. 4134, Published by MDPI, DOI: 10.3390/ELECTRONICS11244134, Available: <https://www.mdpi.com/2079-9292/11/24/4134>.
- [41] Oumayma Berraadi, Hicham Gibet Tani and Mohamed Ben Ahmed, "Enhancing Security and Privacy in IoT Data Streams: Real-Time Anomaly Detection for Threat Mitigation in Traffic Management", *Computer Sciences & Mathematics Forum*, 16 June 2025, Vol. 10, No. 1, Article No. 8, Published by MDPI, DOI: 10.3390/CMSF2025010008, Available: <https://www.mdpi.com/2813-0324/10/1/8>.



© 2026 by the author(s). Published by Annals of Emerging Technologies in Computing (AETiC), under the terms and conditions of the Creative Commons Attribution (CC BY) license which can be accessed at <http://creativecommons.org/licenses/by/4.0>.