Journal of Computer Science

Research Article

Smart and Secure Anomaly Detection in IoT Using Quantum Inspired Kernel Ensemble Learning

Abrar M. Alajlan

Self-Development Skills Department, King Saud University, Riyadh, KSA, Saudi Arabia

Article history
Received: 21-07-2025
Revised: 26-11-2025
Accepted: 20-01-2026

Email: aalajlan1@ksu.edu.sa

Abstract: The exponential growth of the Internet of Things networks has greatly increased the attack surface for cyber threats, necessitating the use of strong and perceptive anomaly detection systems. Traditional intrusion detection systems often struggle with high false alarm rates, limited generalization, and inefficiency in processing high-dimensional heterogeneous IoT data. To address this issue, a novel Hybrid Quantum-Enhanced Kernel Ensemble Learning model is proposed for efficient and scalable anomaly detection in the Internet of Things environment. Initially, different datasets were gathered and preprocessed using data encoding, data cleaning, missing data handling, and null value removal. A novel Crossover Strategy Enhanced Wombat Optimization Algorithm is developed to select optimal features that are suitable for classifying anomalies. To improve detection performance and lower computational overhead, the most pertinent features are chosen using a Gradient-based Feature Evaluator combined with a Quantum Kernel Estimation Layer. In order to attain strong performance, the refined features are subsequently fed into an ensemble classification framework that integrates predictions. Thus, the proposed framework guarantees safe, scalable, and intelligent anomaly detection designed for real-time Internet of Things network protection, while also reducing false positives.

Keywords: Anomaly Detection, Wombat Optimization Algorithm, Crossover Strategy, Quantum Kernel Support Vector Machine, Explainable Boosted Decision Forests, Linear Regression

Introduction

In Internet-based services, there has been an immense surge in the growth of the Internet, which leads to the fast expansion of the Internet of Things (IoT). IoT involves the connection and computational capability of smart vehicles, applications, sensors, and industrial systems. It connects various devices, including health monitors, sensors, home appliances, and autonomous vehicles, into an intelligent IoT ecosystem (Bałdyga et al., 2024). They gather and share data to help people and systems make better decisions. IoT is applied in various areas. As more devices are connected, the network becomes difficult and larger. The growth of interconnected devices, which are beneficial in many ways, also presents challenges in monitoring and ensuring proper functioning (Ahmed et al., 2023). Abnormal behavior of the system leads to various problems, including software bugs, cyberattacks, and hardware failure (Rahman et al., 2024). Detecting these anomalies at an early stage is essential to prevent system breakdowns, data corruption, and security breaches. Basic statistical and rule-based techniques are traditional anomaly detection techniques that are easy to implement but struggle in dynamic environments. To detect unknown anomalies, these methods assume a predefined pattern in the data and make it less adaptive (Duraj et al., 2025). They resulted in high false positive rates and normal behavior misclassified as anomalies. IoT devices are resource-constrained and face certain challenges such as limited energy, low processing power, and limited memory. Resource-constrained devices are computationally difficult because of their limited processing power, battery life, and memory (Rahim et al., 2023). Additionally, even an unidentified anomaly triggers failure, endangers public safety, and disrupts operations. These issues produce a need for lightweight, intelligent, and adaptive anomaly detection systems that perform well in decentralized, real-time settings.

Machine Learning (ML) methodologies learn from large amounts of data and recognize difficult patterns that are hard to model manually (Gong et al., 2025). By analyzing data deviations, correlations, and behavioral patterns, they identify unknown and known anomalies. This model maintains performance while being lightweight, making it ideal for use on edge devices with limited resources. Because of detecting known and unknown attacks, they are extremely valuable in the dynamic and distinct IoT environment. In addition, resource constraints in IoT devices make it difficult to use inappropriate models. These problems are tackled by the Hybrid Quantum-Enhanced Kernel Ensemble Learning (HQEKEL) model, which integrates ensemble learning, explainable models, and quantum-enhanced kernels.

Novelty

Quantum-Enhanced Feature Evaluation: The model integrates a Quantum Kernel Estimation Layer (QKEL) and gradient-based feature scoring for identifying high-value features effectively in high-dimensional data.

Hybrid Kernel-Based Ensemble Framework: To improve robustness and generalization over varied IoT datasets, the proposed model utilizes a meta-learner and various base learners with multiple kernels.

Adaptive Learning with Handling Class Imbalance: To enable better learning in imbalanced class scenarios, the model integrates adaptive gradient-based optimization often found in real-world IoT traffic datasets.

Cross-Dataset Scalability and Robustness: The HQEKEL framework is evaluated based on four distinct datasets, including CIC IoT 2023, Ton-IoT, NF-ToN-IoT, and UNSW-NB15. These datasets are selected to exhibit well-performing and cross-domain generalization, a complex problem ignored by current anomaly detection techniques.

Contribution

Design of the HQEKEL Architecture: By applying several base learners and quantum-inspired kernel estimation, the proposed model enables robust anomaly detection in IoT environments.

Gradient-Based Feature Optimization with Quantum Kernels: The model applies a gradient-enhanced quantum kernel selection mechanism to prioritize relevant features and remove unnecessary ones. This enhances detection accuracy over complex IoT data distributions and reduces overfitting issues.

Integrated Ensemble with Adaptive Kernel Learning: A novel multi-kernel ensemble classifier is developed, in which various learners apply certain kernel functions. A meta-learner that dynamically adjusts to the properties of the data is used to fuse these outputs, improving the robustness and accuracy of classification.

Performance Evaluation Across Multiple Real-World Datasets

Four real-world IoT datasets, CIC IoT 2023, Ton-IoT, NF-ToN-IoT, and UNSW-NB15, are used to thoroughly assess the HQEKEL model. With 98.54% accuracy, 97.84% precision, and 97.92% MCC, it outperforms six existing models across a variety of evaluation metrics.

Related Work

This section discusses certain existing works in terms of ML and Deep Learning (DL)-based Smart Anomaly Detection in IoT devices.

Smart Anomaly Detection in IoT Devices Using ML

To enhance security in IoT networks, Punia et al. (2025) developed an anomaly detection (AD) approach, namely, that integrates Modified Whale Transfer and Sine-Cosine algorithms (MWTS-CA). The MWTS-CA has identified the malicious behaviors by integrating statistical techniques. This approach was suitable for the dynamic and evolving nature of IoT networks, as it incorporates the strong search capabilities of whale optimization. The experimental findings demonstrated that the performance of the MWTS-CA model was higher and required lower training time.

Sarwar et al. (2022) presented an effective anomaly detection model for IoT. By enabling an Improved dynamic Sticky Binary Particle Swarm Optimization (IDSBPSO), this work has offered an effective feature selection process. The SBPSO algorithm's searchability has been improved using a reduction strategy. In the detection phase, malicious data traffic was detected by using an IDS. The performance evaluation of the algorithm confirmed that the IDSBPSO mechanism was better with greater performance ratios.

Gad (2025) developed an IoT-based anomaly detection framework, namely Threshold optimization and causal analysis for IoT (TOCA-ToT). This TOCA-ToT model attained sufficient performance in anomaly detection with the robust detection of the causal relationships in network data. Here, anomaly classification was conducted based on the optimal thresholds that were selected by the refined TOC mechanism. Using the IoT network dataset, the model's efficiency was estimated, where the model ensured outstanding performance rates.

Mutambik (2024a) presented a lightweight detection model, namely a Flow-based Intrusion Detection System for IoT. This work has classified the anomalies based on the Packet header details and services used that were captured by the flow-based representations. In the validation phase, different IoT-based datasets were deployed. The overall outcome has shown its enhanced performance rates on each dataset.

To ensure IoT security, Ahmad et al. (2023) developed a Multistage Spectrogram image-based network Anomaly Detection System (MS-ADS). Here, spectrogram image datasets were utilized to train the MS-ADS approach.

Smart Anomaly Detection in IoT Devices Using DL

Deivakani (2025) developed a Bidirectional 3D Quasi-Recurrent Neural Network with a Coati Optimization Algorithm (B3DQRNN-COA) for anomaly detection. This work, the DS2OS dataset, gathered data and was preprocessed by applying an Implicit UnScented Particle Filter (IUPF). Furthermore, the Coati Optimization Algorithm has accurately classified anomaly detection in IoT networks. The developed model attained better results in terms of superior accuracy and minimized error rates.

To improve energy efficiency and security in IoT communications, Lydia et al. (2023) presented Green Energy-efficient Routing with DL-based Anomaly Detection (GEER-DLAD). The Moth Flame Swarm Optimisation (MSO) algorithm was applied in order to choose the best routes and guarantee efficient data transmission paths. The GEER-DLAD model's application showed notable gains in anomaly detection performance and energy efficiency. To improve IoT security, Mutambik (2024b) created a hybrid DL-based AD genetic algorithm (GA-HDLAD). Increased detection efficiency and accuracy were made possible by the integration of GA and DL approaches. The evaluations indicated that the anomaly detection mechanism attained a higher accuracy rate.

To identify intrusions in IoT networks, an intrusion detection system using chaotic poor and rich optimization with a DL model (IDCPRO-DLM) was developed by Alrayes et al. (2023). In order to determine which features were most significant for intrusion detection, a chaotic poor, and rich optimization algorithm was applied for the feature selection phase.

For privacy-preserving anomaly detection in the IoT, Kethineni and Gera (2023) designed the Sparse Capsule-Auto Encoder (SCAE). The SCAE model captured hierarchical relationships in IoT data using sparse representation and capsule networks while protecting user privacy. Attention GRU (AGRU) was employed, which extends traditional GRU by integrating an attention mechanism to detect anomalies in the privacy-preserved data. Results from experiments showed that the developed model achieved higher efficiency with respect to robustness against privacy attacks and high detection accuracy.

Billiard-Based Optimization with a DL-driven AD and Classification (BBODL-ADC) model for IoT-assisted sustainable smart cities was introduced by Manickam et al. (2023). To choose effective features, a Binary Pigeon Optimization algorithm (BPEO) was applied. For categorizing and recognizing anomalies, the developed model used the Elman Recurrent Neural Network (ERNN). The evaluations indicated that the BBODL-ADC exhibited notable improvement in detection accuracy and created safe smart city infrastructures.

Zulfiqar et al. (2024) introduced the Deep Detect DL framework, in which a multilayer CNN was applied to extract and learn spatial features from IoT data. Capturing the correlation between the features and the gradient vanishing problem was solved by the Gated Recurrent Unit (GRU). The performance validation achieved superior performance in terms of computational resource usage and accuracy.

Limitations and Research Gaps

Despite the existing techniques containing certain benefits in IoT anomaly detection, several challenges still exist, such as high false positive rates, cross-domain generalization, inability to handle imbalanced data, and limited computational resources. In various IoT environments, the majority of traditional techniques perform effectively only on particular datasets, and they fail to adapt to changing traffic patterns and attack types. Additionally, many models lack interpretability, treating all detected anomalies equally without considering their severity, and often operate as black boxes with no transparency. For real-time or edge-level deployment, their computational problem makes them inappropriate. To tackle these issues, the HQEKEL combines Quantum Kernel Estimation with gradient-based feature selection and integrates predictions from Quantum Kernel Support Vector Machine (QKSVM), Explainable Boosted Decision Forests (EBDF), and Linear Regression (LR) to provide resource-efficient solutions for accurate anomaly detection. Table 1 summarizes various smart anomaly detection methods.

System Design

This section discusses the overview of smart city architecture and core challenges that necessitate the HQEKEL model to detect anomalies in IoT environments. Figure 1 illustrates the system model.

The smart city architecture comprises an IoT sensing layer, edge-level computing, and centralized analytics. Moreover, the smart city model contains three layers, which include the IoT sensing layer, cloud computing, and fog computing layer. The detailed description of these layers is discussed below.

The variable notation and its description are provided in Table 2.

System Design

This section discusses the overview of smart city architecture and core challenges that necessitate the HQEKEL model to detect anomalies in IoT environments. Figure 1 illustrates the system model.

The smart city architecture comprises an IoT sensing layer, edge-level computing, and centralized analytics.

Moreover, the smart city model contains three layers, which include the IoT sensing layer, cloud computing, and fog computing layer. The detailed description of these layers is discussed below.

IoT sensing layer: The IoT sensing layer is placed at the base level, which is responsible for gathering real-time data from various sources or regions. This layer is otherwise known as the perception/device layer. This layer contains traffic cameras, environmental sensors, GPS modules, utility meters, and health monitors. Each sensor node SN_j where $j = 1, 2, ...m$ produces a stream of raw data D_j. The total data gathered from all sensor nodes can be expressed as:

$$D_{Total} = \sum_{j=1}^{m} D_j(s) \quad (1)$$

Table 1: Summary of Existing ML and DL-Based Smart Anomaly Detection in IoT

Author and Year	Methods	Technique Used	Dataset Applied	Key Outcomes	Limitations
ML-Based Approaches					
Punia et al. (2025)	MWTS-CA	Modified Whale Transfer + Sine-Cosine algorithms	IoT traffic data	High detection accuracy, reduced training time	Limited cross-domain adaptability
Sarwar et al. (2022)	IDSBPSO	Improved Sticky Binary Particle Swarm Optimization for feature selection + IDS	IoT dataset	Effective feature selection, improved performance	Performance drop on unseen traffic
Gad (2025)	TOCA-ToT	Threshold optimization + causal analysis with Random Forest	IoT traffic dataset	Robust causal relationship detection	Threshold tuning complex
Mutambik (2024a)	Flow-based IDS	Packet header & service flow representations	Multiple IoT datasets	Lightweight, fast, dataset-validated	Limited deep feature learning capability
Ahmad et al. (2023)	MS-ADS	Spectrogram image conversion + CNN + STFT	IoT spectrogram dataset	High detection accuracy, robust classification	Computationally intensive image processing
DL-Based Approaches					
Deivakani (2025)	B3DQRNN-COA	Bidirectional 3D Quasi-RNN + Coati Optimization	DS2OS dataset	Superior accuracy, reduced error rates	Model complexity, high training cost
Lydia et al. (2023)	GEER-DLAD	DL anomaly detection + Moth Flame Swarm Optimisation for routing	IoT routing datasets	High accuracy + energy-efficient routing	Dataset-specific tuning required
Mutambik (2024b)	GA-HDLAD	Hybrid DL with Genetic Algorithm for feature selection	IoT datasets	Higher detection accuracy and efficiency	Training overhead due to GA
Alrayes et al. (2023)	IDCPRO-DLM	Chaotic Poor & Rich Optimization + DL-based IDS	Smart city IoT datasets	Strong feature selection, improved security	Complexity of chaotic optimization
Kethineni and Gera (2023)	SCAE + AGRU	Sparse Capsule Autoencoder + Attention-GRU	Smart agriculture IoT	Privacy-preserving, robust anomaly detection	Model interpretability limited
Manickam et al. (2023)	BBODL-ADC	Binary Pigeon Optimization + Elman RNN	Smart city IoT datasets	Safe infrastructure, improved accuracy	Higher complexity, lower precision
Zulfiqar et al. (2024)	DeepDetect	CNN for spatial features + GRU for sequential learning	IoT datasets	Superior performance, efficient resource usage	Limited validation on diverse IoT traffic

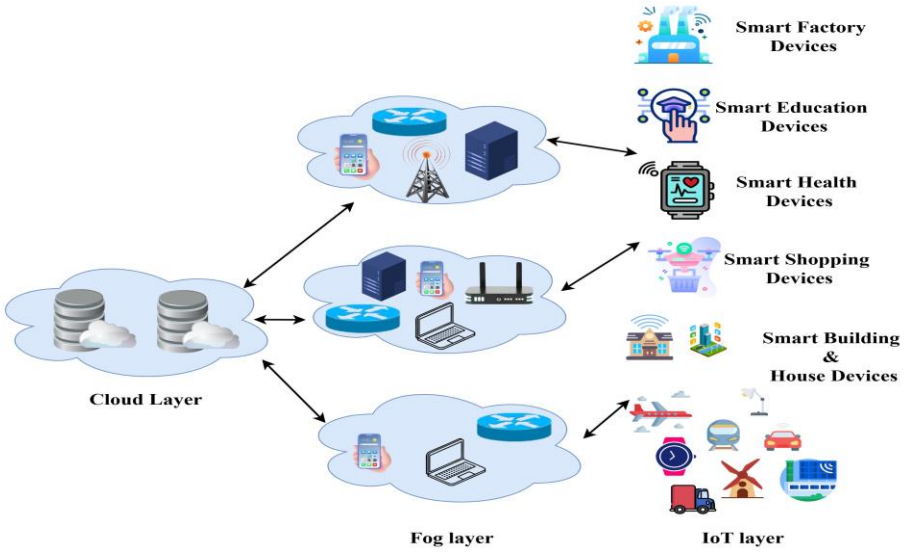

Fig. 1: Smart IoT model

Table 2: Variable Description

Variables	Description
K_{fog}	Fog layer
C_j	Data volume
D_e	Computational capacity of fog node/cloud
S_e	Local processing time
K_{cloud}	Latency incurred in processing data in the cloud.
C_{cloud}	Computational power of the cloud
S_d	Processing time
Ut	Utility function
K_{avg}	Average latency for all layers
E_{avg}	Average energy used for sensing and computation
A_{cc}	Decision accuracy
$\beta_1, \beta_2, \beta_3 \in \Re^+$	User-defined weights
C	Original dataset
C_{clean}	Cleaned dataset
C_{ij}	Value at j^{th} row and i^{th} column
W_j	Observed non-missing values
$W_{missing}$	Imputed value
m	Number of available values
$G(m)$	Gini Index at a node m
P_j	Probability of a data instance being classified into classes j
D	Total number of classes
M	Total number of samples at the parent node
M_l, M_R	Number of samples in the left and right child nodes
$G(M_L), G(M_R)$	Gini indices of the respective child nodes
F(W)	Fitness function
E(W)	Classification error of a given model using subset W
$\lvert W \rvert$	Number of selected features
n	Features
$\beta \in [0,1]$	Balancing coefficient
W_q	Randomly chosen solution
$W_j^{(s)}$	J^{th} wombat's current position
q	Random scalar controlling the exploration degree
$\oplus$	Bitwise XOR
W_{best}	Global best solution

W_b and W_a	Best-performing solutions
w_i^b and w_i^a	i^{th} bit of parent vectors W_b and W_a
T	Maximum iterations
γ	Crossover probability
β	Balancing coefficient
$\varphi(\mathrm{w})$	Nonlinear quantum feature embedding
Hs	Quantum Hilbert space
$V(\mathrm{w})$	Quantum circuit
$\langle\phi(\mathrm{w})\rangle$	Quantum state
$z_j \in \{-1,+1\}$	Binary labels
β_j	Lagrangian multipliers
D	Soft margin constant
Hy	Hyperplane
ϕ_j	SHAP values
F	Set of input features
T	Subset excluding J
f_T	Output of the model using subset T
α_0	Bias term
α_j	Feature weight
$P_{QKSVM} \in [0,1]$	Quantum SVM-based confidence score for an anomalous sample
$P_{EBDF} \in [0,1]$	Probability estimated by the ensemble voting mechanism
$\hat{z}_{reg} \in \Re$	Anomaly severity score

From expression (1), $D_j(\mathrm{s})$ denotes the time-varying data produced by j^{th} sensor. Typically, these data streams are large in size, sensitive to latency, and heterogeneously diverse.

Fog computing layer: Directly sending massive data to cloud servers causes high latency and ineffective bandwidth utilization. To overcome this, the fog layer acts as the mediator between the sensing layer and the cloud. This layer is made up of edge devices, including routers, gateways, and micro-data centers, graphically placed close to the sensor nodes. This layer performs the task of pre-processing, filtering, and aggregation of raw data. By employing rule-based inference or lightweight ML mechanisms, it executes localized decision-making. Mathematically, the latency incurred at the fog layer expressed as:

$$K_{fog} = \frac{C_j}{D_e} + S_e \tag{2}$$

From expression (2), C_j, D_e, and S_e indicate data volume, computational capacity of fog node, and local processing time. The fog layer selects only relevant data sent to the cloud. The ratio of filtered data at the fog level is $\eta \in [0,1]$. Then the efficient data sent from each sensor node to the cloud is provided in Equation (3):

$$C_e = (1-\eta).\mathrm{C}_{total} \tag{3}$$

The empirical tuning showed that approximately 60% of the raw traffic is removed at the fog level. Therefore, η = 0.40, indicating that 40% of the original data is transmitted to the cloud.

Cloud computing layer: For learning-based decisions, deep analytics, trend analysis, and large-scale data storage, the cloud computing layer provides centralized services. It supports long-term data mining, model training, and inter-domain correlation analysis by utilizing high-performance computing infrastructure. The mathematical expression of the latency incurred in processing data in the cloud K_{cloud} is expressed as:

$$K_{cloud} = \frac{D_e}{C_{cloud}} + S_d \tag{4}$$

From Equation (4), D_e, C_{cloud}, and S_d, and depict the volume of data received, computational power of the cloud, and processing time. Generally, $K_{cloud} > K_{fog}$ supports fog computing's use in latency-sensitive applications like emergency services or driverless cars. The inequality is used because cloud servers possess significantly higher computing power than fog nodes, enabling them to handle large-scale deep analytics that cannot be efficiently executed at the fog layer. To model the system's overall performance, a utility function. is applied that considers latency, energy consumption, and

decision accuracy. Let K_{avg}, E_{avg}, and A_{cc} ,represent the average latency for all layers, the average energy used for sensing and computation, and the decision accuracy based on the data. The overall utility is then determined by:

$$Ut = \beta_1 . \frac{1}{K_{avg}} + \beta_2 . \frac{1}{E_{avg}} + \beta_3 . A_{cc} \quad (5)$$

From Equation (5), $\beta_1, \beta_2, \beta_3 \in \Re^+$ are user-defined weights represent the relative significance of latency, energy, and accuracy for the particular smart city service.

Materials and Methods

The proposed architecture is illustrated in Figure 2. A novel HQEKEL methodology is developed to detect anomalies in IoT environments. An initial stage of the HQEKEL framework is data collection, where datasets such as ToN-IoT, UNSW-NB15, NF-ToN-IoT, and CIC IoT 2023 were applied for training and evaluation. In order to identify subtle patterns that differentiate malicious activity from legitimate operations in smart city infrastructures, each dataset comprises multiple classes. The second phase is data preprocessing, which ensures data quality and model effectiveness. The data preprocessing step contains normalization, null value elimination, mean imputation, data cleaning, and one-hot encoding to enhance the quality of the data. After preprocessing, the Feature selection phase is placed, where the Crossover Strategy Enhanced Wombat Optimization Algorithm is utilized for feature selection. Then the selected features are sent to the ensemble classification model to integrate and combine predictions from several base learners. This module contains QSVM for optimal generalization, Linear Regression (LR) to create a severity score, and Explainable Boosted Decision Forests (EBDF) to maintain transparency and provide interpretability. Finally, the threshold mechanism is applied to categorize the score as anomalous or normal.

Data Collection

Four publicly available benchmark datasets, CIC IoT Dataset 2023, ToN-IoT, NF-ToN-IoT, and UNSW-NB15, are used to improve cybersecurity in IoT networks. Together, these datasets cover a wide variety of network traffic patterns, attack methods, and typical behavior in different IoT environments. A synopsis of each dataset is described.

CIC IoT Dataset 2023: The CIC IoT Dataset 2023 dataset is specially designed for anomaly detection in the IoT environment. Seven traffic classes are included in this dataset, such as Mirai, Reconnaissance, MITM (Man-in-the-Middle), DDoS, Benign, DoS, and Theft. Since it includes malicious and benign traffic for cyber threat detection and it is applied to train and evaluate supervised learning.

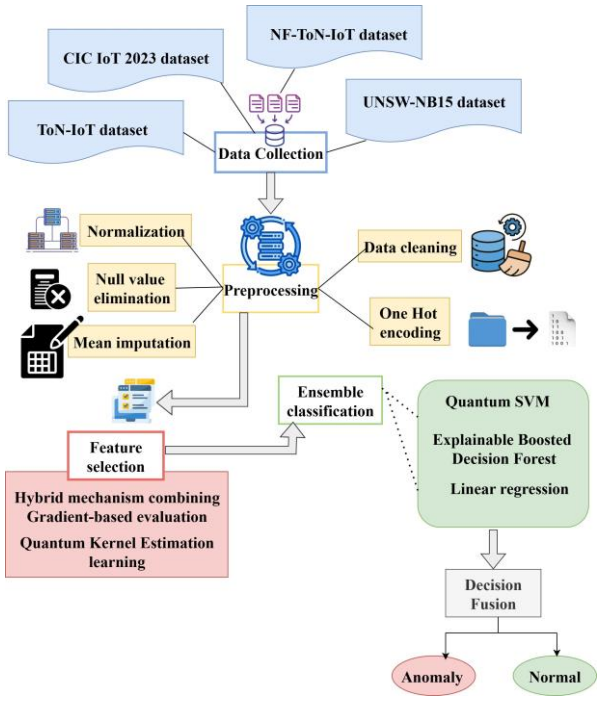

Fig. 2: Overall architecture of the proposed model

ToN-IoT Dataset: The ToN-IoT dataset contains a variety of IoT devices to aid AI-based cybersecurity research. This dataset contains ten distinct classes, including reconnaissance, DDoS, injection attacks, password cracking, ransomware, backdoor, XSS, normal traffic, MITM, and DoS. The labeling of each instance makes supervised classification tasks easier.

NF-ToN-IoT Dataset: NF-ToN-IoT dataset records real-time traffic from a variety of IoT devices and is intended for network intrusion and anomaly detection. It covers classes like XSS, DoS, ransomware, backdoors, injection, reconnaissance, password cracking, DDoS, MITM, and normal traffic. Particularly, it is great for Industrial IoT and smart city research.

UNSW-NB15 Dataset: The UNSW-NB15 dataset is a commonly used benchmark in cybersecurity research, with 2,218,761 benign and 321,283 malicious network flow records included in the UNSW-NB15 dataset. Reconnaissance, worms, shellcode, backdoor, DoS, Normal, analysis, fuzzers, ransomware, and Generic are ten types of attack classes. For training, 80% and for testing, 20 % of the data is employed.

Data Pre-Processing

Preprocessing is an essential step to ensure the consistency and quality of data for subsequent analysis and model training. Data encoding, data cleaning, missing

data handling, and null value removal are certain preprocessing techniques used.

Null Value Removal: IoT traffic data null values are frequently caused by communication errors, packet loss, or sensor malfunctions.

$$C_{clean} = C - \sum_{j=1}^{n}\sum_{i=1}^{m}\delta(\mathrm{c}_{ij}) \tag{6}$$

From Equation (6), C, C_{clean}, and c_{ij} depict the original dataset, cleaned dataset, and the value at j^{th} row and i^{th} column. If c_{ij} is null, $\delta(\mathrm{c}_{ij}) = 1$; other wise 0.

Missing Data Handling: Mean imputation is used in place of deleting rows with missing values:

$$w_{missing} = \frac{1}{m}\sum_{j=1}^{m} w_j \tag{7}$$

From Equation (7), w_j, $w_{missing}$, and m represent observed non-missing values, imputed value, and number of available values.

Data Encoding: Various IoT features fall into one of several categories. The one hot encoding equation is mentioned in Equation (8):

$$O_{HE}(\mathrm{w}) = \begin{cases} 1, & if\ w = category_j \\ 0, & otherwise \end{cases} \tag{8}$$

In Equation (8), category j denotes the specific class within a categorical feature, and its role is to indicate which position in the one-hot encoded vector is set to 1 while the remaining positions are set to 0. TCP's encoded vector is [1, 0, 0] for the protocol type feature that has the categories TCP, UDP, and ICMP.

Data cleaning: To enhance the dataset's integrity, data cleaning entails eliminating duplicate, irrelevant, or corrupted entries. It involves removing noisy records, random traffic records, and error packets from IoT networks that could affect model performance.

Determining Feature Importance and Ranking

After the data is preprocessed, the features are ranked according to their importance scores to determine which features are most important for model training. Selecting the most appropriate features increases model accuracy and computational efficiency in cybersecurity applications, particularly for anomaly detection. Feature importance quantifies each feature's contribution to a model's predictive performance. This is commonly measured in tree-based models by the drop-in node impurity, which is weighted by the probability that a sample will reach that node. The probability of a node is determined by dividing the number of samples that make it by the total number of samples. Features are considered more significant if there is a higher reduction in impurity. The feature importance score falls between 0 and 1. A feature that receives a 0 score means that it does not influence the output model. If a feature receives a score of 1, it means that the model's decision is totally based on it.

In statistical analysis and data mining, Gini impurity is a common metric used to evaluate node purity. For binary splits, the Gini Index $G(m)$ at a node m is calculated as:

$$G(\mathrm{m}) = 1 - \sum_{j=1}^{D} p_j^2 \tag{9}$$

From Equation (9), p_j and D depicts the probability of a data instance being classified into classes j and the total number of classes. After a node splits into two child nodes M_L and M_R, the impurity reduction is evaluated as:

$$\Delta G = G(\mathrm{m}) - \left(\frac{M_L}{M} G(\mathrm{M}_L) + \mathrm{G}\left(M_R\right)\right) \tag{10}$$

From expression (10), M M_L, M_R and $G(\mathrm{M}_L), \mathrm{G}\left(M_R\right)$ denote the total number of samples at the parent node, the number of samples in the left and right nodes, and the Gini indices of the respective child nodes. Because ΔG it makes a larger contribution to the division of the data into pure subsets, a feature that yields greater significance is regarded as more significant. On the basis of their ranking, the best features l are then chosen. This choice makes it easier to create a strong and effective tree-based security model by lowering the feature space dimensionality.

Crossover Strategy Enhanced Wombat Optimization Algorithm (CS-WOA) for Feature Selection

A binary optimization technique designed for efficient feature selection in high-dimensional datasets is the Crossover Strategy Enhanced Wombat Optimization Algorithm (CS-WOA) (Benmamoun et al., 2024). optimization integrates a crossover strategy. In the search space, this is inspired by a wombat's behavior, like tunnel seeking and habitat optimization to mimic their natural search. Each wombat in the population represents a binary vector in the context of cybersecurity feature selection, where each bit indicates whether a specific feature is included (1) or excluded (0). The fitness function for a feature subset $F(W)$ is expressed as:

$$F(\mathrm{W}) = \beta.\mathrm{E}(\mathrm{W}) + (1-\beta).\frac{|\mathrm{W}|}{m} \tag{11}$$

From expression (11), $E(W)$, $|W|n$ and $\beta \in [0,1]$ denotes sthe classification error of a given model using subset W , number of selected features, features, and balancing coefficient.

Initializing a population of binary vectors that represent random feature subsets is the initial step in the CS-WOA. By interacting with a randomly chosen solution W_q, a wombat explores new areas of the feature space during the exploration phase. A mathematical model for this interaction is as follows:

$$W_j^{(s+1)} = W_j^{(s)} \oplus q.\left(W_q^s \oplus W_j^{(s)}\right) \quad (12)$$

From expression (12), $W_j^{(s)}$, q, $\oplus$ denotes the j^{th} wombat's current position, a random scalar controlling the exploration degree, and bitwise XOR. In the exploitation phase, the solution is refined by moving closer to the global best solution W_{best} as follows:

$$W_j^{(s+1)} = W_j^{(s)} \oplus t.\left(W_{best} \oplus W_j^{(s)}\right) \quad (13)$$

From expression (13), another random scalar t that regulates the exploitation step size is taken. Following each round of position updates, a crossover mechanism is integrated to enhance the population's diversity and convergence. Two of the best-performing solutions *Wb* and *Wa* are chosen by this mechanism, which executes single-point crossover at a randomly chosen position *d*. The offspring produced by integrating traits from both parents is provided by:

$$W_{offspring} = \left[w_1^b, w_2^b, ...w_d^b, w_{d+1}^a, ..., w_m^a \right] \quad (14)$$

From the expression (14), w_i^b and w_i^a represent the i^{th} bit of parent vectors W_b and W_a If this offspring's fitness is higher than the population's worst solution, it is used to replace it. It is assessed using the same fitness function. Algorithm 1 outlines the process of selecting the most relevant IoT traffic features using the hybrid quantum kernel-enhanced evolutionary learning strategy for CS-WOA.

Algorithm 1: Pseudocode for CS-WOA

Input:
- Dataset with selected features
- Population size
- Maximum number of iterations
- Crossover probability
- Balancing coefficient

Step 1: Initialize the population
- Create a population of binary solution vectors
- Each vector represents a candidate feature subset

Step 2: Evaluate initial population
- Compute the fitness of every solution in the population
- Identify the best solution found so far

Step 3: Begin main optimization loop
For each iteration:

Step 3.1: Update each solution
For each solution in the population:
- Decide whether to update the solution using exploration behavior
- If exploration is selected:
* Choose another solution at random
* Modify the current solution based on the random solution
- Otherwise:
* Modify the solution using exploitation behavior
- Evaluate the updated solution

Step 3.2: Apply crossover (if triggered)
- If crossover is allowed in this iteration:
* Select the two best solutions in the population
* Choose a random crossover point
* Create a new child solution by combining the two parents
* Evaluate the child solution
* If the child is better than the worst solution:
Replace the worst solution with the child

Step 3.3: Update global best
- If any solution in the population is better than the current best:
Update the best solution

Step 4: End loop when maximum iterations are reached

Output:
- The best feature subset discovered during optimization

Output: Optimal feature subset

The following phases contain the CS-WOA structure: The first step in the process is population initialization, which creates the initial candidate feature subsets using randomly generated binary vectors. A specified fitness function that takes into account both the subset size and the classification error is then used to assess each solution. The major loop starts with exploration, in which randomly chosen peers use XOR-based updates to perturb solutions.

An exploitation phase follows, during which solutions use another XOR-based transformation to get closer to the most well-known solution. Occasionally, top-performing solutions are subjected to a crossover operation in order to produce a potentially superior offspring. The population is assessed and updated in accordance with each update. Then, the algorithm returns the ideal feature subset that maximizes classification accuracy while reducing the objective function. Figure 3 depicts the flowchart representation of CS-WOA.

Hybrid Quantum Enhanced Kernel Explainable Linear (HQEKEL) Framework for Smart Anomaly Detection in IoT Devices

IoT devices are vulnerable to complex anomalies and attacks in the dynamic landscape of cyber-physical systems. Conventional ML models frequently experience nonlinearity issues, poor interpretability, or inflexibility in the imbalance and data drift. To tackle these, HQEKEL is developed, a synergistic ensemble that uses statistical modeling, explainable ensemble trees, and quantum-enhanced learning with a well-planned fusion strategy for optimal generalization.

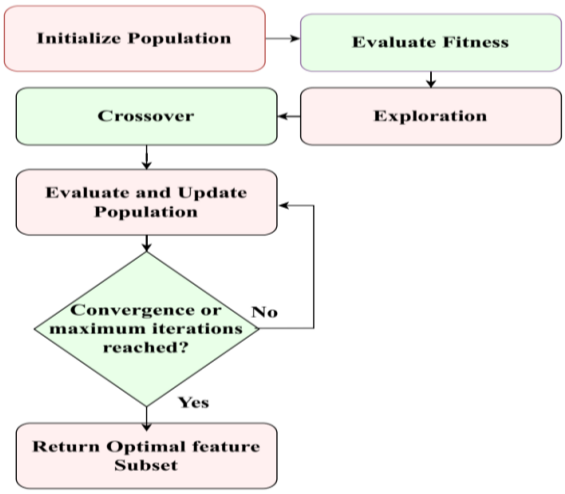

Fig. 3: Flowchart representation of CS-WOA Model

Quantum Kernel Support Vector Machine (QKSVM) Submodule

The HQEKEL framework initiates QSVM (Miroszewski et al., 2023), which uses a nonlinear quantum feature embedding $\varphi(\mathrm{w})$ to map high-dimensional sensor data $\mathrm{W} \in \mathrm{R}^m$ into a quantum Hilbert space Hs in order to classify it into normal and anomalous classes. A quantum circuit $V(\mathrm{w})$ transforms classical input into a quantum state $\langle \phi(\mathrm{w}) \rangle$ in the quantum kernel space. The following is the definition of the kernel similarity between two instances w_j and w_i is given in Equation (15):

$$L_R(w_j, w_i) = |\langle \phi(w_j) | \phi(w_i) \rangle|^2 \tag{15}$$

This is evaluated through a swap-test quantum circuit. The QKSVM objective is expressed as:

$$\min_{\beta} \frac{1}{2} \sum_{j,i=1}^{n} \beta_j \beta_i z_j z_i L_R(w_j, w_i) - \sum_{j=1}^{n} \beta_j \tag{16}$$

Subject to:

From the above expressions (16) and (17), $z_j \in \{-1, +1\}$, β_j, and D represents binary labels, Lagrangian multipliers, and soft margin constant. Due to quantum-enhanced kernels, the QKSVM returns a margin-maximizing hyperplane *zz*, separating anomalous traffic from normal traffic with better generalization:

$$\sum_{j=1}^{n} \beta_j z_j = 0, \quad 0 \le \beta_j \le D \tag{17}$$

Explainable Boosted Decision Forests (EBDF)

Although QKSVM improves handling accuracy and nonlinearity, it lacks interpretability, which is crucial in IoT environments that are regulated or mission-critical. EBDF is an ensemble of Gradient Boosted Trees (GBT), which is used in the second stage to improve classification while maintaining explainability through SHAP (Shapley Additive Explanations). As mentioned in equation (18), $K(\mathrm{z}, \hat{\mathrm{z}})$ including weak learners, EBDF reduces the empirical loss iteratively:

$$\hat{z}^{(\mathrm{s})} = \hat{z}^{(\mathrm{s}-1)} + \eta . f_s(\mathrm{w}) \tag{18}$$

Each tree $f_s \in F$ learns the residual errors from the previous iteration. SHAP values ϕ_j are evaluated for each feature *j* is as follows:

$$\phi_j = \sum_{T \subseteq F \setminus \{\mathrm{j}\}} \frac{|\mathrm{T}|!(|\mathrm{F}| - |\mathrm{T}| - 1)!}{|\mathrm{F}|!} \left[f_{T \cup \{\mathrm{j}\}}(\mathrm{w}) - \mathrm{f}_T(\mathrm{w}) \right] \tag{19}$$

From expression (19), *F*, *T*, and f_T denotes the set of input features, subset excluding *j*, and output of the model using subset *T*.

Linear Regression-Based Anomaly Severity Scoring

In order to prioritize responses to various threats, HQEKEL uses an LR (Piekutowska et al., 2021) model to generate a continuous anomaly severity score after SHAP ranks the most informative features. Let $w' = [w_1, w_2, ...w_l]$ be the features that are top-*l* ranked features. The mathematical expression in (20) of the regression model is:

$$\hat{z}_{reg} = \alpha_0 + \sum_{j=1}^{l} \alpha_j w_j \quad (20)$$

According to the above expression, s and α_j stands for the bias term and feature weight. This output measures the degree of anomaly in a given input and is particularly helpful in automated security operations centers where severity ranking is required.

Decision Fusion Strategy in HQEKEL Framework

Performance, interpretability, and precision in IoT anomaly detection are all critically dependent on the integration of heterogeneous models such as QKSVM, EBDF, and LR into a single decision system. The Decision Fusion Strategy attempts to combine the strengths of each component: Non-linear separability, feature importance and explainability, and anomaly severity scoring into a single, all-inclusive anomaly index known as the Anomaly Score. In real-time, this score acts as a single, cohesive metric for recognizing and prioritizing possible threats. A weighted sum of the three models' outputs is calculated as part of the core fusion mechanism. Extracted from the decision function and subjected to a sigmoid activation, let $P_{QKSVM} \in [0,1]$ represent the quantum SVM-based confidence score for an anomalous sample. $P_{EBDF} \in [0,1]$ indicate the probability estimated by the ensemble voting mechanism, which is reflected in the boosted forest confidence for an anomaly classification. $\hat{z}_{reg} \in \Re$ indicate the anomaly severity score derived from regression, which can be normalized to the [0, 1] range. The fused anomaly score is expressed in Equation (20):

$$Anomaly\_score = \lambda_1 * P_{QKSVM} + \lambda_2 * P_{EBDF} + \lambda_3 * \hat{z}_{reg} \quad (21)$$

From the expression (21), the fusion coefficients that establish the relative influence of each submodule are , , .These weights can be determined empirically through validation experiments, learned through meta-optimization or adaptively assigned through reinforcement learning or attention mechanisms. The Bayesian probability combination principles and ensemble learning theory serve as the foundation for this fusion approach, which aggregates various hypotheses (model outputs) linearly to reduce expected generalization error. In contrast to a binary classifier, the regression-based severity score guarantees that the framework prioritizes threats according to their level of severity in addition to identifying them.

Thresholding Mechanism

Following the computation of the fused Anomaly Score, a binary classification decision must be made. This is accomplished by applying a threshold that is established by cost-sensitive tuning:

$$Class = \begin{cases} Anomalous, & if \ \ Anomaly\_score \geq \tau \\ Normal, & otherwise \end{cases} \quad (22)$$

The threshold in Equation (22) is determined using a cost-sensitive optimization over the validation set, where τ is chosen to maximize the Youden-J statistic while penalizing high false positive and false negative rates according to predefined cost weights. Depending on the application's risk tolerance, it is tuned to minimize False Positives (FP) or False Negatives (FN) by striking a balance between sensitivity (recall) and specificity.

Integration With Confidence and Uncertainty

Frameworks for modern loT anomaly detection must function in an uncertain environment. Model confidence intervals or epistemic uncertainty can be added to the fusion strategy in HQEKEL:

$$\Delta G \quad (23)$$

From expression (23), σ_j signifies the standard deviation of prediction from the model *j*. The value "3" refers to the three prediction sources used in the HQEKEL decision-level fusion:

(i) QKSVM probability (PQK)
(ii) EBDF probability (PEBDF)
(iii) Regression-based anomaly severity score

Algorithm 2 describes the training procedure with optimizer enhancement and decision-level fusion to improve anomaly detection performance in HQEKEL. The fusion weights are not manually set; they are learned automatically through a validation-based constrained optimization process. A grid/Bayesian search is performed under the constraint and the weight triplet that maximizes the validation F1-score (or minimizes misclassification cost) is selected

Figure 4 illustrates the decision fusion process in the HQEKEL framework. The decision fusion module combines the outputs of the QKSVM module, EBDF classifier, and Linear Regression scorer, using a weighted fusion mechanism to compute the final anomaly score.

Algorithm 2: Decision Fusion Strategy in HQEKEL

Input:
- IoT feature vector of the incoming sample
- Fusion weight for the Quantum Kernel SVM output
- Fusion weight for the Boosted Decision Forest output
- Fusion weight for the Regression-based severity output
- Classification threshold
- Trained Quantum Kernel SVM model
- Trained Boosted Decision Forest model

- Trained Linear Regression severity model

Step 1: Generate individual model outputs
- Obtain the probability score from the Quantum Kernel SVM
- Obtain the confidence score from the Boosted Decision Forest
- Obtain the severity value from the Regression model

Step 2: Normalize the severity value
- Convert the regression output into a normalized score between zero and one

Step 3: Compute the fused anomaly score
- Multiply each model's score by its assigned fusion weight
- Add the weighted scores together to obtain a single fused score

Step 4: Make the final decision
- If the fused score is greater than or equal to the threshold:
  Return "Anomalous"
- Otherwise:
  Return "Normal"

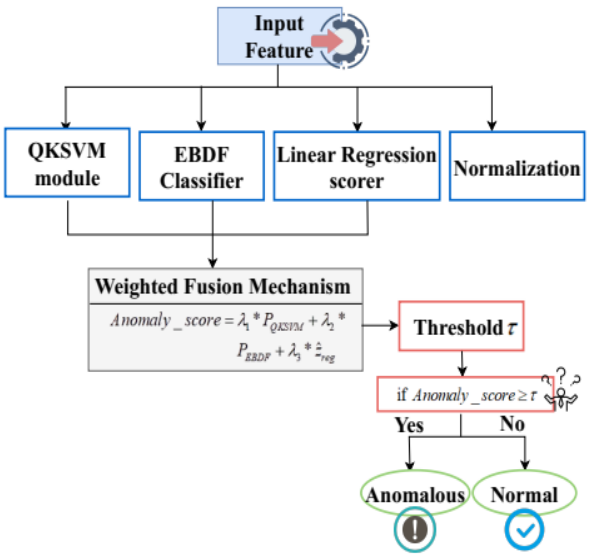

Fig. 4: Architecture of the HQEKEL Decision Fusion Module

Experimental Results

The experimental outcomes of the proposed model, hyperparameters and rates, evaluation metrics, performance assessment, comparative analysis, and statistical testing of the proposed approach are all demonstrated in this section. Conclusively, the results of the ablation study and SHAP analysis are presented. These analyses show how well the suggested approach works to identify IoT anomalies.

Experimental Configuration

An Intel® Core™ i7-5960X CPU, 32 GB of RAM, and an NVIDIA GeForce RTX 2060 GPU were installed on a high-performance workstation for the experimental evaluation. The operating system used was Linux Mint 20.3 Cinnamon, which offered a reliable and effective setting for tasks involving data processing and deep learning. Python 3.8.10 was used to conduct all experiments.

Hyper Parameter Configuration

Table 3 indicates the Hyper Parameter configuration for anomaly detection in IoT, highlighting key parameters like learning rate, loss function, dropout rate, and more. The hyperparameters of the HQEKEL framework were optimized through a combination of grid search and empirical validation. A grid search was first performed on a reduced training subset to explore candidate values for the learning rate, dropout rate, and batch size. The final parameters were selected based on the configuration that maximized validation accuracy while minimizing overfitting.

Table 3: Hyper Parameter configuration

Parameter	Values
Optimizer	Adam
Learning rate	0.001
Loss function	Binary cross-entropy
Batch size	64
Number of Epochs	100
Dropout rate	0.3
Activation function	ReLU
Output activation	Sigmoid

- Learning rate (0.001): Selected after testing values in the range {0.1, 0.01, 0.001, 0.0001}. A higher rate caused unstable convergence, while a lower rate slowed training significantly
- Dropout rate (0.3): Tuned over {0.1, 0.2, 0.3, 0.5}. A value of 0.3 provided the best trade-off between generalization and model stability by reducing overfitting
- Batch size (64): Tested across {32, 64, 128}. A batch size of 64 achieved a balance between computational efficiency and stable gradient updates
- Epochs (100): Determined through early stopping criteria, as training beyond 100 epochs did not yield further improvement in validation metrics

This structured tuning process ensures that the chosen hyperparameters are not arbitrary but empirically validated for optimal performance across the datasets.

Evaluation Measures

A number of evaluation metrics are used in order to thoroughly assess how well the suggested model performs for improving anomaly detection in IoT networks. In addition to standard classification metrics like accuracy, precision, recall, specificity, F1-score, Matthews Correlation Coefficient (MCC), and Cohen's Kappa coefficient, these include regression-based metrics like

Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). Security breach rate and detection rate are among the other domain-specific metrics that are considered.

Accuracy: By calculating the ratio of correctly predicted instances (both normal and anomalous) to total instances, it evaluates the model's overall correctness:

$$A_{cc} = \frac{Tr_{Ps} + Tr_{Ng}}{Tr_{Ps} + Tr_{Ng} + Fs_{Ps} + Fs_{Ng}} \tag{24}$$

Precision: The number of successfully recognized positive predictions (attacks), as shown in Equation (25), is the measure of the model's ability to generate accurate predictions:

$$P_{re} = \frac{Tr_{Ps}}{Tr_{Ps} + Fs_{Ps}} \tag{25}$$

Recall: Recall calculates the ratio of accurately predicted positive cases to all positive cases:

$$\mathrm{Re}_{call} = \frac{Tr_{Ps}}{Tr_{Ps} + Fs_{Ng}} \tag{26}$$

Specificity: Specificity evaluates the model's ability to correctly identify normal traffic and minimize false positives:

$$Sp_{ec} = \frac{Tr_{Ng}}{Tr_{Ng} + Fs_{Ps}} \tag{27}$$

F1-score: It measures the model's performance by considering both false positives and false negatives by computing the harmonic mean of precision and recall:

$$F1_s = \frac{2 \times \mathrm{P}_{re} \times \mathrm{Re}_{call}}{\mathrm{P}_{re} + \mathrm{Re}_{call}} \tag{28}$$

MCC: MCC takes into account all four confusion matrix classes and is considered a balanced metric for binary classification tasks:

$$MCC = \frac{Tr_{Ps} \times Tr_{Ng} - Fs_{Ps} \times Fs_{Ng}}{\sqrt{\left(Tr_{Ps} + Fs_{Ps}\right)\left(Tr_{Ps} + Fs_{Ng}\right)\left(Tr_{Ng} + Fs_{Ps}\right)\left(Tr_{Ng} + Fs_{Ng}\right)}} \tag{29}$$

From Equation (24) to (29), the true positive, true negative, false positive and false negative is Tr_{ps}, Tr_{Ng}, Fs_{ps} and $Fs_{Ng.}$

Cohen's kappa coefficient: This metric calculates the degree of agreement between predicted and actual classifications:

$$\kappa = \frac{P_o - P_e}{1 - P_e} \tag{30}$$

From Equation (30), P_o and P_e denotes observed agreement and expected agreement.

MAE: To determine the average prediction error, As mentioned in Equation (31), MAE calculates the average of the absolute differences between the predicted z_j and actual values $\hat{\mathrm{z}}_j$:

$$MAE = \frac{1}{n}\sum_{j=1}^{m} |\,\mathrm{z}_j - \hat{\mathrm{z}}_j\,| \tag{31}$$

MSE: The average of squared deviations between expected and actual values is assessed by MSE which is depicted in Equation (32):

$$MSE = \frac{1}{n}\sum_{j=1}^{m} (\mathrm{z}_j - \hat{\mathrm{z}}_j)^2 \tag{32}$$

RMSE: The standard deviation of prediction errors is shown by the RMSE, which is the square root of the MSE. The RMSE is mentioned in Equation (33):

$$RMSE = \sqrt{MSE} \tag{33}$$

Detection Rate (DR): DR as mentioned in equation (34) calculates the ratio of real anomalies that the system correctly detects:

$$DR = \frac{Correctly\,\mathrm{det}\,ected\ attacks}{Total\ actual\ attacks} \tag{34}$$

Security Breach Rate (SBR): SBR quantifies the ratio of successful attacks to the number of attempted attacks:

$$SBR = \frac{M_{sb}}{M_{ta}} \tag{35}$$

From expression (35), M_{sb} and M_{ta} is the number of successful breaches and total number of attack attempts:

Performance Analysis

The performance of the HQEKEL method through comprehensive performance analysis is evaluated in this section. In this section, accuracy analysis, loss analysis, Precision-Recall analysis, and AUC-ROC analysis are carried out to determine the performance of the proposed method. In addition, Confusion matrices provide a detailed breakdown of classification results. Overall, this section provides a comprehensive understanding of the model's performance and its suitability for enhancing security in IoT networks. Table 4 presents the Performance of the proposed model on different datasets.

Figure 5 (a) and (b) show training and testing accuracy across all datasets. Both training and testing accuracy were evaluated based on the CIC-IoT dataset 2023, Ton-IoT, NF-ToN-IoT, and INSW-NB15 dataset, which achieved the training accuracy of 0.982, 0.975, 0.951, and 0.923, respectively, as mentioned in Figure 5 (a).

Table 4: Performance of the proposed model on different datasets

Measures	Datasets				Average
	CIC IoT Dataset 2023	Ton-IoT	NF-ToN-IoT	UNSW-NB15	
Accuracy (%)	98.75	98.70	97.97	98.81	98.54
Precision (%)	97.82	98.20	97.60	97.77	97.84
Recall (%)	97.80	96.31	96.03	95.63	96.44
Specificity (%)	98.12	97.33	97.51	96.70	97.41
F1-score (%)	97.80	97.24	96.80	96.68	97.13
MCC (%)	98.22	98.04	97.89	97.55	97.92
Cohen's Kappa coefficient (%)	97.21	97.42	96.11	98.18	97.23

The absence of major fluctuations demonstrates that the selected hyperparameters (learning rate 0.001, dropout 0.3) provide a balanced trade-off between fast convergence and generalization. As mentioned in Figure 5 (b), the testing accuracy of the CIC-IoT dataset 2023, Ton-IoT, NF-ToN-IoT, and INSW-NB15 dataset is 0.978, 0.971, 0.964, and 0.931. The lack of divergence between curves suggests that underfitting and overfitting are effectively mitigated. This confirms that the early stopping criterion was applied at the optimal point, avoiding unnecessary computation without sacrificing performance.

Figure 6 (a) and (b) illustrate the loss analysis of the proposed HQEKEL models for training and testing data. The loss analysis of training and testing was conducted in CIC-IoT Dataset 2023, Ton-IoT, NF-ToN-Iot, and INSW-NB15 dataset, which attains the training loss of 0.110, 0.124, 0.16, and 0.19, respectively, which is shown in Figure 6 (a).

The uniform convergence across datasets indicates that the HQEKEL framework generalizes well to diverse traffic patterns. This enhancement confirms that the model maintains stable learning dynamics.

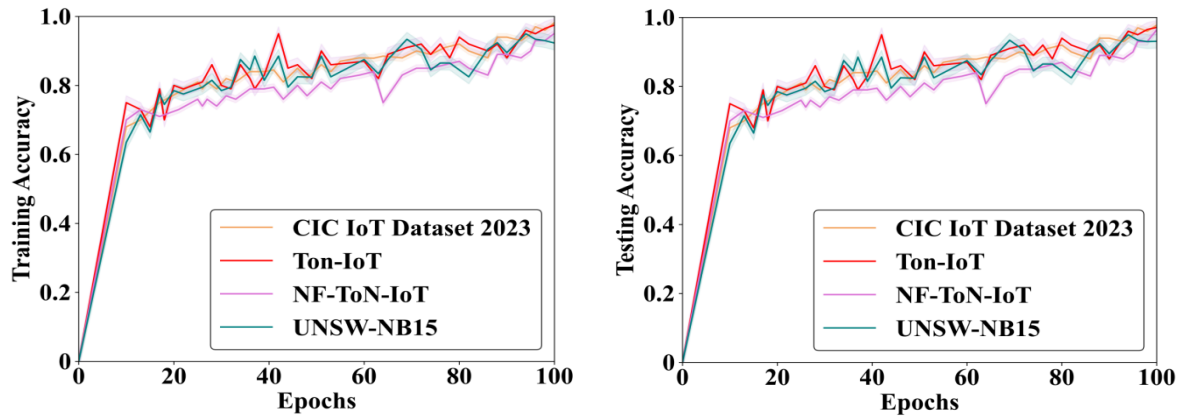

Fig. 5: Training and Testing accuracy across all datasets (a) training accuracy (b) testing accuracy

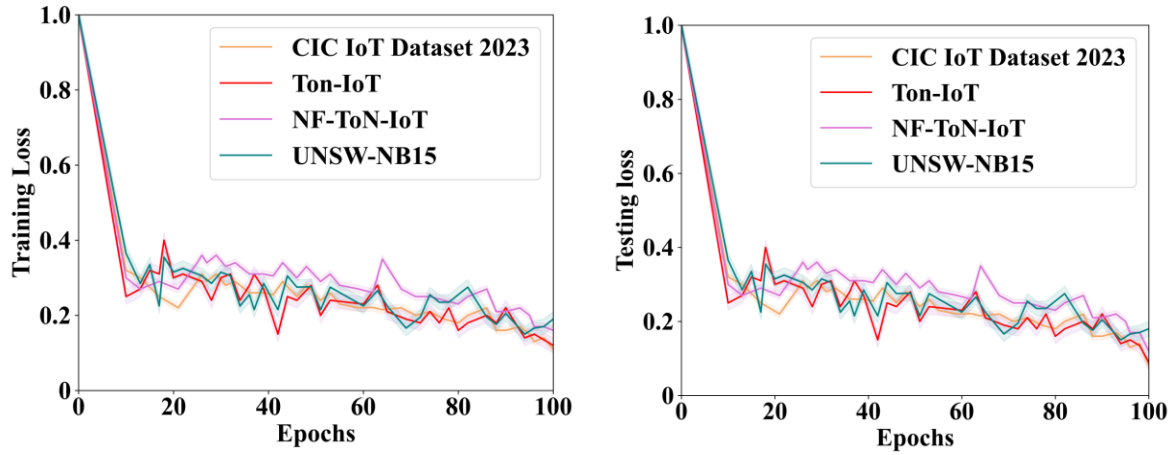

Fig. 6: Loss analysis on all datasets (a) training loss (b) testing loss

As mentioned in Figure 6 (b), the testing loss of the CIC-IoT dataset 2023, Ton-IoT, NF-ToN-IoT, and INSW-NB15 dataset is 0.08, 0.09, 0.12, and 0.18. This highlights the robustness of HQEKEL against dataset variability.

Figure 7 provides the graphical analysis to determine the precision-recall curve and AUC-ROC curve. Figure 7 (a) showcases the precision-recall curve of the HQEKEL technique on the CIC IoT Dataset 2023, Ton-IoT, NF-ToN-IoT, and UNSW-NB15 datasets. The HQEKEL technique obtains higher PR values on four distinct datasets. It demonstrates that the HQEKEL approach attains a higher precision rate and a high recall rate. From this graph, all four datasets show high precision for moderate to high recall, highlighting robust classification performance. Figure 7 (b) illustrates the AUC-ROC curve of the HQEKEL model on the CIC IoT Dataset 2023, Ton-IoT, NF-ToN-IoT, and UNSW-NB15 datasets. The HQEKEL framework attains 0.976, 0.964, 0.958, and 0.961 on four datasets, respectively. This curve shows that the proposed technique provides excellent performance.

Figure 8 (a), (b), (c), and (d) depict the confusion matrix for four datasets, including the CIC IoT Dataset 2023, Ton-IoT, NF-ToN-IoT, and UNSW-NB15 dataset.

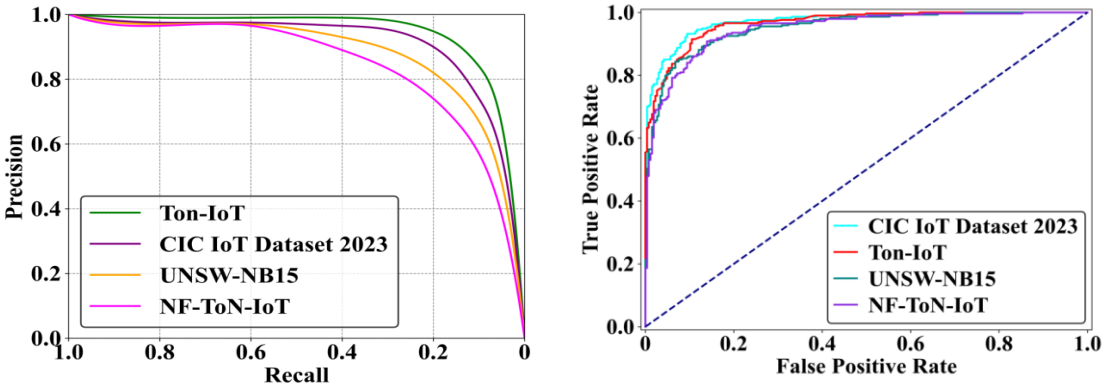

Fig. 7: (a) Precision-Recall Curve analysis (b) AUC-ROC curve analysis of a proposed model for all datasets

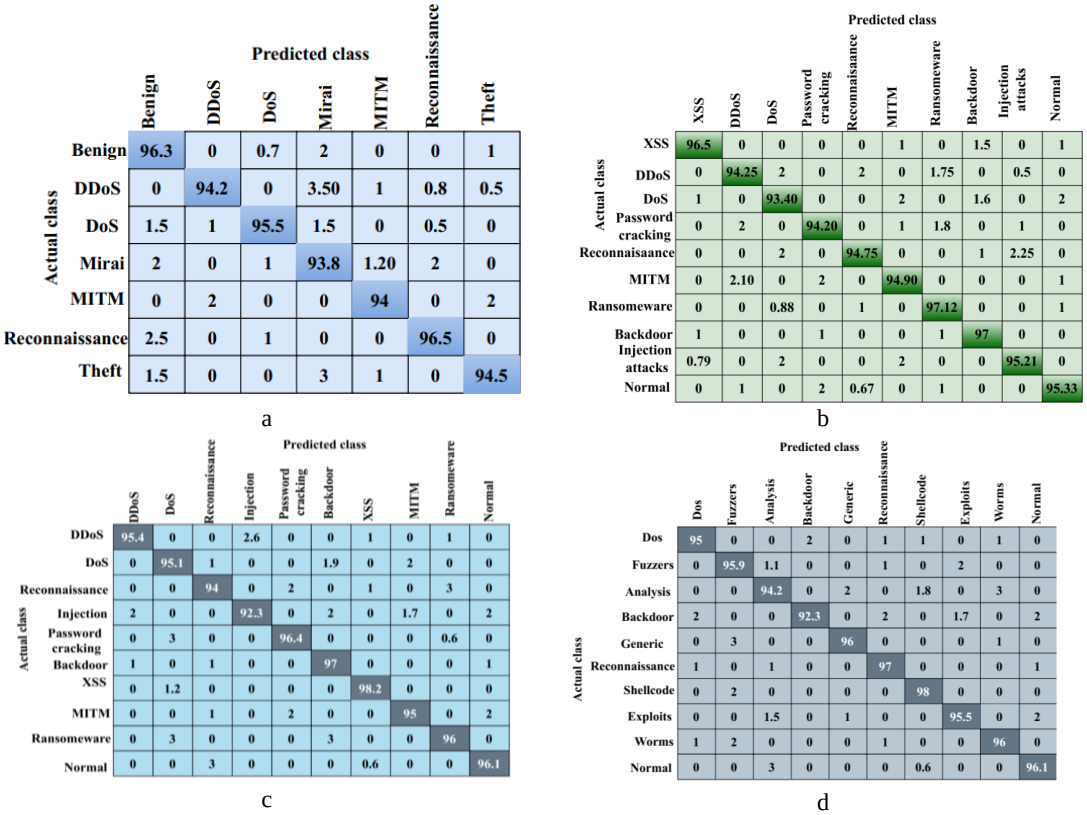

Fig. 8: Confusion matrix for (a) CIC IoT Dataset 2023, (b) Ton-IoT, (c) NF-ToN-IoT, and (d) UNSW-NB15

In these confusion matrices, the predictions are plotted against the actual labels. In Figure 8(a), corresponding to the CIC IoT 2023 dataset, the majority of predictions are concentrated along the diagonal, confirming high accuracy with very few false positives. Figure 8(b), which reports the results on the ToN-IoT dataset, shows balanced detection across multiple attack categories, with only a small number of false negatives, highlighting the robustness of the model in handling diverse traffic patterns. For the NF-ToN-IoT dataset in Figure 8(c), the confusion matrix reflects near-perfect classification, with minimal misclassifications, although some overlap in minority attack categories indicates the challenge of imbalanced data. Finally, Figure 8(d), showing the UNSW-NB15 dataset results, demonstrates reliable classification with low false negatives and only slightly higher false positives compared to the other datasets.

Figure 9 depicts the SHAP analysis of various features in different networks. The horizontal bars represent the average SHAP values, indicating the contribution of each feature to the model's output. Features consistently exhibit the highest SHAP values across all datasets, confirming their strong influence in distinguishing between benign and malicious IoT traffic.

Network-level statistical attributes play a significant role, highlighting the importance of connection frequency and error-related features in intrusion characterization.

Figure 10 depicts the fitness score, which compares the performance of two algorithms such as CS-WOA and WOA, over 100 iterations. CS-WOA performs better than WOA throughout the optimization process. Both algorithms show significant improvement in the initial iterations at the end of the iteration; the performance slows down. Conclusively, the CS-WOA algorithm is more robust in finding an optimal solution than WOA.

Comparative Analysis

The performance of the proposed model is compared with six existing models, including MWTS-CA (Punia et al., 2025), TOCA (Gad, 2025), MS-ADS (Ahmad et al., 2023), SCAE (Kethineni and Gera, 2023), BBODL-ADC (Manickam et al., 2023), and GEER-DLAD (Lydia et al., 2023) is discussed here.

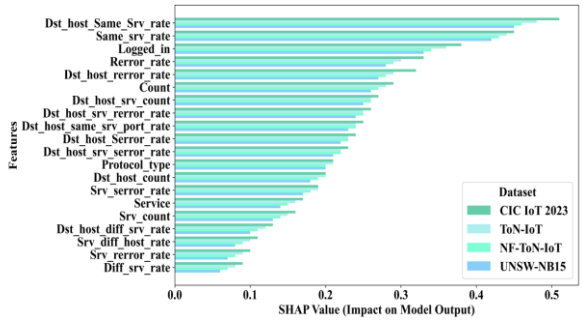

Fig. 9: Feature Importance analysis

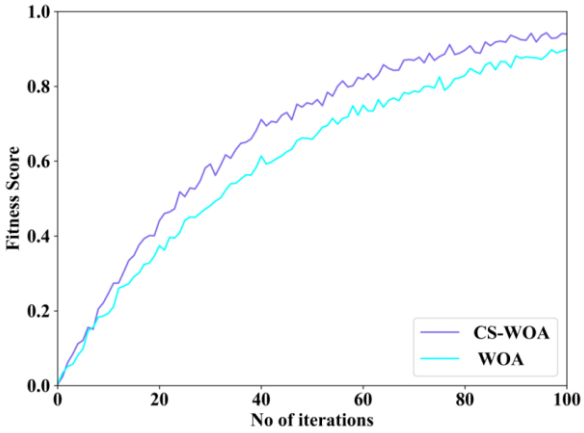

Fig. 10: Fitness Score Analysis

Table 5: Hyper Parameter configuration

Parameter	Values
Optimizer	Adam
Learning rate	0.001
Loss function	Binary cross-entropy
Batch size	64
Number of Epochs	100
Dropout rate	0.3
Activation function	ReLU
Output activation	Sigmoid

Table 5 shows the comparison between the HQEKEL technique and existing methods. According to Table 5, the HQEKEL method achieved the highest accuracy of 98.54%. The accuracy of existing strategies, such as MWTS-CA, TOCA, MS-ADS, SCAE, BBODL-ADC, and GEER-DLAD, attained 97.62, 96.35, 97.11, 96.23, 95.82, and 94.96%, respectively. Furthermore, the achieved precision values are 97.84% (Proposed), 96.54% (MWTS-CA), 96.32% (TOCA), 95.28% (MS-ADS), 94.66% (SCAE), 93.36% (BBODL-ADC), and 94%(GEER-DLAD). In addition, the recall values of the MWTS-CA and TOCA frameworks were 95.90% and 95.11%. Similarly, the recall rates of the MS-ADS, SCAE, BBODL-ADC, and GEER-DLAD strategies were 95.28, 94.66, 93.36, and 94%. The HQEKEL approach outperformed with a 96.44% recall. The F1-score values acquired by the existing frameworks are MWTS-CA (96.38%), TOCA (95.89%), MS-ADS (94.67%), SCAE (93.58%), BBODL-ADC (92.41%), and GEER-DLAD (92%). The HQEKEL method obtained a high F1 score of 97.13%. The high specificity value of HQEKEL is 97.41%. Moreover, the existing techniques obtained specificity values of MWTS-CA is 97%, TOCA is 96.56%, MS-ADS is 95.53%, SCAE at 94.48%, BBODL-ADC is 96.12%, and GEER-DLADis95.25%. The MCC rate of the proposed method is 97.92% and the existing methods such as MWTS-CA, TOCA, MS-ADS, SCAE, BBODL-ADC, and GEER-

DLAD attained the lowest MCC of 96.33, 95.22, 93.71, 92.85, 94.44, and 90.36%. The Cohen's kappa coefficient rate of the proposed method is 97.23% and the existing methods such as MWTS-CA, TOCA, MS-ADS, SCAE, BBODL-ADC, and GEER-DLAD attained the lowest Cohen's kappa coefficient of 95.68, 96.57, 94.55, 94, 93.35, and 92.88%. This represents the performance of the HQEKEL strategy in Anomaly detection.

The MAE, MSE, RMSE, DR, and SBR analysis of the HQEKEL and existing techniques are illustrated in Figure 11 (a) to (d). In Figure 11 (a), the proposed method attained 0.64 of MAE and the existing techniques, such as MWTS-CA, TOCA, MS-ADS, SCAE, BBODL-ADC, and GEER-DLAD, achieved MAE of 0.77, 0.79, 0.82, 0.85, 0.9, and 0.94. In Figure 11 (b), the proposed method attained 0.76 of MSE and the existing techniques, such as MWTS-CA, TOCA, MS-ADS, SCAE, BBODL-ADC, and GEER-DLAD, attained MSE of 0.78, 0.86, 0.88, 0.91, 0.96, and 0.93. In Figure 11 (c), the HQEKEL method attained 0.87 of RMSE and the existing techniques such as MWTS-CA, TOCA, MS-ADS, SCAE, BBODL-ADC, and GEER-DLAD attained RMSE of 0.89, 0.91, 0.96, 0.93, 0.92, and 0.95. In Figure 11 (d), the HQEKEL method attained 38.2% of SBR and 97.95% of DR the existing techniques such as MWTS-CA, TOCA, MS-ADS, SCAE, BBODL-ADC, and GEER-DLAD, attained SBR and DR of 46.6, 59.8, 62.2, 79.5, 88.1, and 90% as well as 95.18, 96.3, 92.28, 90.19, 93.33, and 94.47.

Ablation Study

Table 6 presents the ablation study conducted to analyze the impact of different components of the HQEKEL model. Overall, the ablation results confirm that each module contributes to the performance of the HQEKEL framework. While each variant retains reasonable accuracy, the integration of feature selection, optimizer enhancement, and fusion modules collectively enables the model to achieve state-of-the-art performance across all metrics.

Statistical Analysis

In anomaly detection for IoT applications, the performance of the developed model is analyzed through the McNemar p-value test, as shown in Table 7. Statistical analysis plays an important role in anomaly detection within IoT systems, which identify unusual patterns from expected behavior.

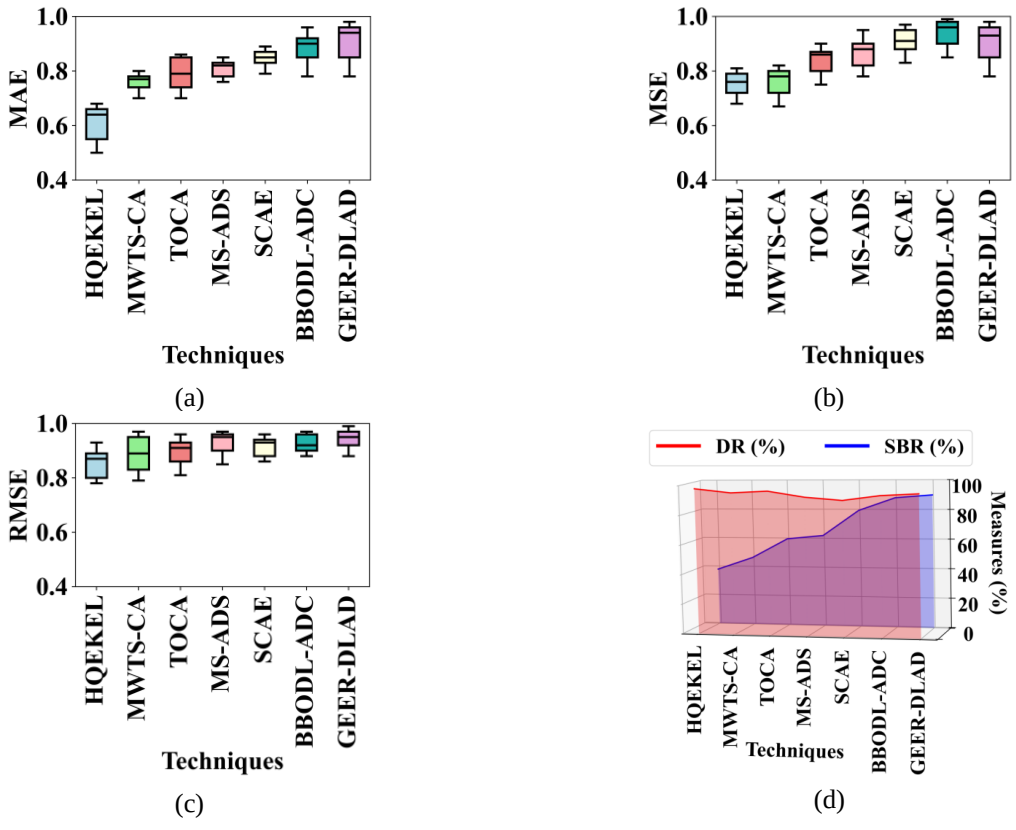

Fig. 11: Comparative analysis (a) MAE (b) MSE (c) RMSE (d) DR and SBR

Table 6: Ablation study analysis

Category	Variant	Description	Accuracy (%)	F1-Score (%)	AUC (%)
Component Testing	HQEKEL (Full Proposed)	QKSVM + EBDF + Regression + Optimized Fusion	98.54	97.13	99.12
	QKSVM Only	Tests Quantum contribution	97.10	95.90	97.84
	EBDF Only	Tests tree-ensemble contribution	95.86	95.10	96.45
	Regression Only	Severity scorer alone	84.32	82.20	86.52
	HQEKEL w/o QKSVM	Removes quantum kernel	97.82	96.55	98.21
	HQEKEL w/o EBDF	Removes EBDF	97.45	96.12	98.00
	HQEKEL w/o Regression	Removes regression scorer	97.01	95.88	97.64
	HQEKEL with Uncalibrated Probabilities	No calibration before fusion	96.88	95.43	96.91
Fusion Weight Sensitivity	Equal-Weight Fusion (⅓,⅓,⅓)	Baseline for weight sensitivity	97.92	96.48	98.02
	Random Weight Fusion (avg of 10 runs)	10 random λ combinations	97.30	96.00	97.55
	Fusion Weight Sweep (Grid Search)	Best on validation set	98.61	97.25	99.20
	Sensitivity: +0.05 perturbation	Slight deviation	98.12	96.88	98.70
	Sensitivity: +0.10 perturbation	Larger deviation	97.63	96.40	98.22

Table 7: Statistical Performance

Model / Component	Training (s) mean ± SD	Accuracy (%) mean ± SD	95% CI (Accuracy)	McNemar p-value (vs HQEKEL)
HQEKEL (full)	148.6±3.9	98.54±0.42	[98.00, 99.08]	-
QKSVM only	92.4±2.7	97.10±0.67	[96.43, 97.77]	0.002
EBDF only	38.7±1.8	95.86±0.81	[94.98, 96.74]	<0.001
Regression only	2.1±0.2	84.32±1.50	[81.39, 87.25]	<0.001
HQEKEL w/o QKSVM	56.2±2.5	97.82±0.51	[97.31, 98.33]	0.015
HQEKEL w/o EBDF	109.3±3.3	97.45±0.59	[96.77, 98.13]	0.008

Table 8: Computational Analysis of HQEKEL and Classical Kernel Baselines

Component / Model	Training Time (s)	Inference Latency per Sample (ms)	Peak Memory (MB)
HQEKEL (Full Model)	61.8	1.35	273 MB
Quantum Kernel	92.4	1.12	428 MB
Classical Polynomial Support Vector Machine	148.6	1.87	612 MB

Table 8 depicts the evaluation based on computational analysis evaluating the training time, inference latency per sample as well as peak memory.

Discussion

The results obtained from the HQEKEL framework demonstrate that the integration of quantum kernels with ensemble learning significantly enhances anomaly detection performance in IoT environments. In particular, the consistent performance on heterogeneous datasets such as CIC IoT 2023, ToN-IoT, NF-ToN-IoT, and UNSW-NB15 highlights the scalability of the proposed approach in diverse IoT traffic scenarios.

A critical observation from the experiments is that while HQEKEL clearly outperforms existing ML and DL-based models, this improvement comes at the cost of increased computational complexity. The inclusion of a quantum kernel layer enhances feature separability but also increases processing overhead compared to lightweight ML baselines. Similarly, the ensemble structure provides robustness and interpretability through EBDF and LR scoring, yet contributes to higher training and inference time. This indicates a trade-off between performance and efficiency that must be balanced when considering deployment on constrained IoT devices.

Another important aspect is the interpretability offered by the EBDF component and SHAP-based feature ranking. Unlike most deep learning models that function as black boxes, HQEKEL provides insights into which features contribute most to anomaly detection. This improves trustworthiness and makes the model suitable for regulated or mission-critical IoT environments where explainability is essential. Nevertheless, certain gaps remain. The current evaluation does not include a real-time deployment analysis on edge hardware, where

inference latency and memory footprint are critical. Additionally, the framework was validated only under centralized settings, leaving distributed learning scenarios unexplored. Finally, while the model performs well across balanced datasets, anomaly detection in highly imbalanced data streams may require additional mechanisms such as cost-sensitive learning or adaptive reweighting.

An important consideration in IoT anomaly detection is the management of false positives, since excessive alarms can overload security operators and lead to alert fatigue. In practical deployments, a high false positive rate may reduce system trustworthiness and delay responses to genuine attacks. The results of HQEKEL show a consistent reduction in false positives compared to baseline models, as evidenced by the confusion matrices (Figure 8) and precision–recall analysis (Figure 9). This improvement is primarily attributed to the ensemble boosting decision fusion, which stabilizes classification outcomes, and the feature selection mechanism, which filters redundant attributes that often lead to misclassifications. By maintaining a lower false positive rate while preserving high recall, HQEKEL ensures that IoT networks can operate with fewer unnecessary alerts, making the system more reliable for real-world security monitoring.

Limitations

Despite achieving superior detection performance across multiple datasets, several limitations remain:

- The current evaluation focuses on detection accuracy, precision, and robustness but does not include a real-time deployment analysis. Metrics such as inference latency, model size, and memory footprint on edge devices are not reported, which are critical for IoT-based environments with resource-constrained hardware
- The model was validated using centralized training, but distributed deployment scenarios such as federated learning have not yet been explored
- While the ensemble improves accuracy and interpretability, the computational overhead may challenge ultra-low-power IoT devices

Conclusion

For effective and scalable anomaly detection within the IoT, this research proposed a novel HQEKEL technique. To analyze data and identify anomalies, the HQEKEL model uses the integration of DL and ML. The data were gathered from four distinct datasets including CIC IoT 2023, ToN-IoT, NF-ToN-IoT, and UNSW-NB15 datasets. After the data collection phase preprocessing phase takes place where several preprocessing techniques are applied to ensure high-quality data. To achieve high detection accuracy, the HQEKEL integrates quantum kernel estimation, gradient-base validation, and ensemble classification techniques. After the data is preprocessed, the features are ranked according to their importance scores to determine which features are most important for model training. Particularly, for feature selection, this approach is efficient for detecting and choosing relevant features thereby enhancing the effectiveness of the model. Then the selected features are employed by an ensemble classification module that applies a kernel-based ensemble approach for enhancing efficacy. The decision fusion module combines the outputs of the QKSVM module, EBDF classifier, and Linear Regression scorer, using a weighted fusion mechanism to compute the final anomaly score. The performance of the proposed model is validated using four distinct datasets. The proposed model outperformed existing techniques across all key metrics, attaining an average accuracy of 98.54%, precision of 97.84%, recall of 96.44%, specificity of 97.41%, F1-score of 97.13%, and DR of 97.95%. These experimental results demonstrate the proposed model's superior accuracy and robustness in anomaly detection. The current work evaluates the model on publicly available benchmark datasets but does not include real-world attack simulations. Practical deployment should incorporate live traffic generation tools (Mirai botnet emulators and injection attack frameworks) to validate model robustness under operational conditions. Future research can explore combining privacy-preserving models, such as federated learning or homomorphic encryption, to ensure secure collaborative learning over distributed IoT devices while maintaining data confidentiality.

Acknowledgment

Thank you to the publisher for their support in the publication of this research article. We are grateful for the resources and platform provided by the publisher, which have enabled us to share our findings with a wider audience. We appreciate the efforts of the editorial team in reviewing and editing our work, and we are thankful for the opportunity to contribute to the field of research through this publication.

Funding Information

The authors have not received any financial support or funding to report.

Ethics

The use of real-time IoT surveillance data for anomaly detection raises important ethical implications. Continuous monitoring of devices, networks, and user activities can introduce privacy concerns if sensitive information is exposed or misused. While the datasets

used in this study are publicly available and anonymized, real-world deployment would require careful compliance with data protection regulations such as GDPR and HIPAA, depending on the application domain.

An additional concern is the potential for misuse of anomaly detection systems in contexts where surveillance could infringe upon civil liberties. Therefore, safeguards such as anonymization, encryption, and strict access control policies must be implemented to ensure that the system enhances security without compromising individual rights.

References

Ahmad, Z., Khan, A. S., Zen, K., & Ahmad, F. (2023). MS-ADS: Multistage Spectrogram image-based Anomaly Detection System for IoT security. *Transactions on Emerging Telecommunications Technologies*, *34*(8), e4810. https://doi.org/10.1002/ett.4810

Ahmed, I., Ahmad, M., Chehri, A., & Jeon, G. (2023). A Smart-Anomaly-Detection System for Industrial Machines Based on Feature Autoencoder and Deep Learning. *Micromachines*, *14*(1), 154. https://doi.org/10.3390/mi14010154

Alrayes, F. S., Asiri, M. M., Maashi, M., Salama, A. S., Hamza, M. A., Ibrahim, S. S., Zamani, A. S., & Alsaid, M. I. (2023). Intrusion Detection Using Chaotic Poor and Rich Optimization with Deep Learning Model for Smart City Environment. *Sustainability*, *15*(8), 6902. https://doi.org/10.3390/su15086902

Bałdyga, M., Barański, K., Belter, J., Kalinowski, M., & Weichbroth, P. (2024). Anomaly Detection in Railway Sensor Data Environments: State-of-the-Art Methods and Empirical Performance Evaluation. *Sensors*, *24*(8), 2633. https://doi.org/10.3390/s24082633

Benmamoun, Z., Khlie, K., Dehghani, M., & Gherabi, Y. (2024). WOA: Wombat Optimization Algorithm for Solving Supply Chain Optimization Problems. *Mathematics*, *12*(7), 1059. https://doi.org/10.3390/math12071059

Deivakani, M. (2025). Anomaly Detection in IoT Network Traffic Using Bidirectional 3D Quasi-Recurrent Neural Network Optimize With Coati Optimization Algorithm. *Transactions on Emerging Telecommunications Technologies*, *36*(1), e70026. https://doi.org/10.1002/ett.70026

Duraj, A., Szczepaniak, P. S., & Sadok, A. (2025). Detection of Anomalies in Data Streams Using the LSTM-CNN Model. *Sensors*, *25*(5), 1610. https://doi.org/10.3390/s25051610

Gad, I. (2025). TOCA-IoT: Threshold Optimization and Causal Analysis for IoT Network Anomaly Detection Based on Explainable Random Forest. *Algorithms*, *18*(2), 117. https://doi.org/10.3390/a18020117

Gong, S., Kim, T., & Jeong, J. (2025). SPT-AD: Self-Supervised Pyramidal Transformer Network-Based Anomaly Detection of Time Series Vibration Data. *Applied Sciences*, *15*(9), 5185. https://doi.org/10.3390/app15095185

Kethineni, K., & Gera, P. (2023). Iot-Based Privacy-Preserving Anomaly Detection Model for Smart Agriculture. *Systems*, *11*(6), 304. https://doi.org/10.3390/systems11060304

Lydia, E. L., Jovith, A. A., Devaraj, A. F. S., Seo, C., & Joshi, G. P. (2021). Green Energy Efficient Routing with Deep Learning Based Anomaly Detection for Internet of Things (IoT) Communications. *Mathematics*, *9*(5), 500. https://doi.org/10.3390/math9050500

Manickam, P., Girija, M., Sathish, S., Dudekula, K. V., Dutta, A. K., Eltahir, Y. A. M., Zakari, N. M. A., & Gilkaramenthi, R. (2023). Billiard based optimization with deep learning driven anomaly detection in internet of things assisted sustainable smart cities. *Alexandria Engineering Journal*, *83*, 102–112. https://doi.org/10.1016/j.aej.2023.10.039

Miroszewski, A., Mielczarek, J., Czelusta, G., Szczepanek, F., Grabowski, B., Le Saux, B., & Nalepa, J. (2023). Detecting Clouds in Multispectral Satellite Images Using Quantum-Kernel Support Vector Machines. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *16*, 7601–7613. https://doi.org/10.1109/jstars.2023.3304122

Mutambik, I. (2024a). An Efficient Flow-Based Anomaly Detection System for Enhanced Security in IoT Networks. *Sensors*, *24*(22), 7408. https://doi.org/10.3390/s24227408

Mutambik, I. (2024b). Enhancing IoT Security Using GA-HDLAD: A Hybrid Deep Learning Approach for Anomaly Detection. *Applied Sciences*, *14*(21), 9848. https://doi.org/10.3390/app14219848

Piekutowska, M., Niedbała, G., Piskier, T., Lenartowicz, T., Pilarski, K., Wojciechowski, T., Pilarska, A. A., & Czechowska-Kosacka, A. (2021). The Application of Multiple Linear Regression and Artificial Neural Network Models for Yield Prediction of Very Early Potato Cultivars before Harvest. *Agronomy*, *11*(5), 885. https://doi.org/10.3390/agronomy11050885

Punia, A., Tiwari, M., & Verma, S. S. (2025). A machine learning-based efficient anomaly detection system for enhanced security in compromised and maligned IoT Networks. *Results in Engineering*, *26*, 105562. https://doi.org/10.1016/j.rineng.2025.105562

Rahim, A., Zhong, Y., Ahmad, T., Ahmad, S., Pławiak, P., & Hammad, M. (2023). Enhancing Smart Home Security: Anomaly Detection and Face Recognition in Smart Home IoT Devices Using Logit-Boosted CNN Models. *Sensors*, *23*(15), 6979. https://doi.org/10.3390/s23156979

Rahman, M. M., Gupta, D., Bhatt, S., Shokouhmand, S., & Faezipour, M. (2024). A Comprehensive Review of Machine Learning Approaches for Anomaly Detection in Smart Homes: Experimental Analysis and Future Directions. *Future Internet*, *16*(4), 139. https://doi.org/10.3390/fi16040139

Sarwar, A., Alnajim, A. M., Marwat, S. N. K., Ahmed, S., Alyahya, S., & Khan, W. U. (2022). Enhanced Anomaly Detection System for IoT Based on Improved Dynamic SBPSO. *Sensors*, *22*(13), 4926. https://doi.org/10.3390/s22134926

Zulfiqar, Z., Malik, S. U. R., Moqurrab, S. A., Zulfiqar, Z., Yaseen, U., & Srivastava, G. (2024). DeepDetect: An innovative hybrid deep learning framework for anomaly detection in IoT networks. *Journal of Computational Science*, *83*, 102426. https://doi.org/10.1016/j.jocs.2024.102426