Journal of Al-Farabi for Engineering Sciences (JFES)
Vol. 5, No. 1, 2026, **pp**. 108–121
ISSN: 2617-5517 | **DOI**: https://doi.org/10.59746/7q2q6r19

Enhanced Cobra Optimization for Lightweight Intrusion Detection in Edge-IoT Networks

Huda Ali Mahdi[1,*]
[1] Department of Computer Engineering, College of Engineering, Al-Farabi University, Baghdad, Iraq.
Email: hoda.ali@alfarabiuc.edu.iq

Received 21/11/2025, Revised 19/02/2026, Accepted 02/06/2026

Abstract: The rapid proliferation of Internet of Things (IoT) ecosystems has led to the generation of massive, heterogeneous, and noisy data streams from distributed sensors, posing significant challenges for intrusion detection. Conventional machine learning models, such as Support Vector Machines (SVM) and k-Nearest Neighbors (k-NN), rely on handcrafted and static feature representations, limiting their ability to capture complex and dynamic IoT attack patterns. To address these limitations, this paper proposes an Enhanced Cobra Optimization Algorithm (E-COA) integrated within a hybrid edge–cloud deployment framework for efficient and real-time IoT intrusion detection. The proposed E-COA introduces a non-linear adaptive hunting probability to dynamically balance exploration and exploitation, along with a composite fitness function that jointly optimizes classification accuracy, false positive rate (FPR), and feature sparsity. Experiments are conducted on the MQTT-IoT-IDS2020 dataset, which consists of 71,341 labeled network traffic samples across five classes. The proposed model achieves a classification accuracy of 97.2%, a macro-averaged F1-score of 0.971, and a false positive rate of 2.2%. It also demonstrates superior performance compared to SVM, Random Forest, 1D-CNN, PSO, and standard COA models. Furthermore, the optimized model maintains a compact size of approximately 75 KB and achieves an average inference latency of 4.3 ms on edge-simulated hardware. These results indicate that the proposed framework effectively balances detection accuracy, computational efficiency, and suitability for deployment in resource-constrained edge-IoT environments.

Keywords: Internet of Things , Cobra Optimization Algorithm, Intrusion Detection, Edge Computing, Hybrid Deployment, Bio-inspired Algorithms.

1. Introduction

The Internet of Things IoT represents one of the cornerstones in the current digital revolution, allowing a large number of physical objects equipped with sensors, software and connectivity to gather, communicate and share data [1-3]. Now this paradigm is enabling innovative applications in smart cities, healthcare, industrial automation (Industry 4.0), and smart homes. The enormous scale, high dimensional data stream, large amount of noise and inherent heterogeneity of IoT endemic deployments make the merger analysis and security exceedingly challenging [4], [5]. Conventional machine learning algorithms such as SVM, k-NN and decision tree have been successfully used for IoT-related tasks such as anomaly detection and traffic classification [6]. These models are often built using fixes set of hand-crafted features. This dependence makes them fragile to changing data distributions, restricts their scalability and resilience to the noise and variability typical of raw IoT sensor streams [7]. Bio-inspired meta-heuristics, and particularly the COA algorithm, have proved over the years to be a revolutionary technology for adaptive intelligent optimization showing high accuracy in complex nonlinear problem spaces [8], [9]. COA is designed to operate similar to the hunting process of cobras which consists of searching, encountering, and striking stages in order to address the challenge between exploration and exploitation. This capacity to traverse challenging solution spaces has recently proven beneficial for optimizing parameter selection and feature style in data-driven domains [10]-[12].

This is expected to address the fundamental limitations of static feature engineering in IoT security. However, direct utilization of classic COA in IoT is non- straightforward due to inherent trade-offs between convergence speed, solution accuracy, computational complexity and adaptability to rapidly evolving threats - a crucial requirement for real-time intrusion prevention

[13]. Despite the growing use of bio-inspired optimization techniques, existing approaches for IoT intrusion detection still suffer from several limitations, including slow convergence, high computational cost, and lack of adaptability to dynamic IoT environments. In addition, many existing models rely on cloud-based processing, which introduces latency and limits real-time response capabilities. Therefore, there is a need for an efficient, lightweight, and adaptive optimization-based intrusion detection framework that can operate effectively within resource-constrained edge environments. Table 1 summarizes the major challenges in IoT environment and the strategic focus of this paper for addressing them. To address these challenges, the main contributions of this work are summarized as follows:

1) Proposing an Enhanced Cobra Optimization Algorithm (E-COA) with a non-linear adaptive hunting probability that dynamically governs the exploration–exploitation transition, overcoming the premature-convergence limitation of the standard COA.
2) Designing a composite fitness function that balances classification accuracy, false positive rate, and feature sparsity.
3) Developing a hybrid edge–cloud deployment architecture to achieve real-time intrusion detection with low latency.
4) Comprehensive experimental validation on MQTT-IoT-IDS2020, including per-class metrics, confusion matrix, ROC-AUC evaluation, and statistical significance testing against six state-of-the-art baselines.

Table 1: Core IoT Challenges and The Strategic Focus of This Study

IoT Environment Challenge	Consequence for Cobra Algorithm	Strategic Focus of This Study
Data Heterogeneity & Noise	Degrades optimization stability; requires robust, adaptive search mechanisms.	Design of an enhanced COA with adaptive hunting strategies and noise-resilient fitness evaluation.
Resource-Constrained Edge Devices	Limits population size, iteration count, and computational budget for optimization.	Algorithmic optimization for efficiency (adaptive search intensity, lightweight operations) and a defined hybrid edge-cloud deployment strategy.
Latency Sensitivity	Demands real-time or near-real-time decision-making for many applications (e.g., intrusion prevention).	Quantitative evaluation of inference latency on edge-simulated hardware and algorithmic optimization for fast convergence.
Need for High Accuracy & Reliability	Critical for security, healthcare, and industrial control applications.	Algorithm design focused on high discriminative power through enhanced hunting phase control and extensive evaluation against multiple benchmarks.

The remainder of this paper is organized as follows. Section 2 reviews the related work, organized thematically. Section 3 presents the proposed methodology, including the Enhanced Cobra Optimization Algorithm (E-COA) and the hybrid edge–cloud framework. Section 4 describes the experimental setup and evaluation results. Section 5 discusses the findings and performance analysis. Section 6 outlines the limitations of the proposed approach. Finally, Section 7 concludes the paper and highlights future research directions.

2. Related Works

This section surveys prior work on IoT IDS and bio-inspired optimization, organized into four thematic subsections. Research gaps motivating the proposed framework are identified at the close of each subsection.

2.1. Traditional Machine Learning-Based IDS

Early IoT IDS research was dominated by classical supervised classifiers. Vinayakumar et al. [6] demonstrated that SVMs with radial basis function kernels achieve approximately 85% accuracy on benchmark datasets. Al-Garadi et al. [10] reported that Random Forests improve accuracy to approximately 90%, while k-NN classifiers achieved 82% under identical conditions. Despite their interpretability, these models rely on manually crafted feature vectors, rendering them brittle against distribution shifts caused by novel attack patterns [7]. Traditional ML models lack adaptive feature representations and have not been evaluated in edge-deployment scenarios with quantified inference latency constraints.

2.2. Deep Learning-Based IDS

The transition to deep learning substantially reduced reliance on manual feature engineering. Hassan et al. [15] demonstrated that 1D-CNNs are effective for sequential IoT packet-stream analysis, achieving 94.1% accuracy on the MQTT-IoT-IDS2020 benchmark. However, these models typically require several megabytes of parameter storage, making them unsuitable for direct deployment on microcontroller-class edge devices [18],[19]. Deep learning models achieve high accuracy but impose memory and computation footprints exceeding the capabilities of resource-constrained edge IoT devices.

2.3. Bio-Inspired and Swarm-Intelligence Approaches

Uddin et al. [14] proposed a PSO-based feature selector for IoT IDS achieving approximately 93% accuracy, though remaining cloud-dependent. The COA was formally introduced by Dehghani et al. [23]. Singh and Agarwal [24] extended COA with adaptive convergence control, outperforming PSO in benchmark experiments. Kumar et al. [25] proposed a hybrid COA-based traffic classifier achieving 95.5% accuracy. Zhang et al. [27] integrated a lightweight COA into a federated edge-learning framework attaining 94.8% accuracy. No prior work has investigated E-COA with a non-linear adaptive hunting probability specifically tailored for IoT IDS, nor has any COA variant been evaluated within a structured hybrid edge-cloud pipeline with quantified edge inference latency.

2.4. Edge-Optimized and Hybrid Deployment

Shi et al. [16] articulated the vision of edge computing for latency-sensitive IoT applications. Teerapittayanon et al. [22] introduced Distributed Deep Neural Networks providing a conceptual foundation for hybrid deployment. Wang et al. [20] demonstrated that adaptive federated learning can train accurate models while respecting edge resource budgets. Existing hybrid deployment strategies have not been validated with bio-inspired optimized models, and edge inference latency has rarely been quantified under standardized hardware conditions. Especially,

the use of COA has been barely investigated in the targeted domain of IoT IDPS with practical deployment approach. Table 2 show the Comparative Analysis of Previous works.

Table 2: Comparative Analysis of Previous Bio-Inspired Approaches for IoT

Study & Reference	Application Context	Model & Core Strategy	Key Reported Strength	Primary Limitation from an IoT Perspective
Uddin et al. [14]	IoT Intrusion Detection	PSO for Feature Selection	High accuracy (~93%) and reduced feature set.	Cloud-dependent; high computational overhead.
Hassan et al. [15]	IoT Network Routing	Ant Colony Optimization (ACO)	Effective for dynamic routing optimization.	Application-specific; not directly applicable to intrusion detection.
Li et al. [18]	Edge Anomaly Detection	Lightweight PSO	Low latency and suitability for edge devices.	Lower accuracy (~92%) on complex attacks.
Dehghani et al. [23]	General Optimization	Standard Cobra Optimization Algorithm	Novel hunting-inspired search mechanism; good exploration-exploitation balance.	Not evaluated for IoT or security applications; requires parameter tuning.
Singh & Agarwal [24]	Feature Selection	Adaptive COA	Improved convergence over PSO in high-dimensional spaces.	Tested on general benchmarks; no IoT security evaluation.
Kumar et al. [25]	Network Traffic Analysis	Hybrid COA-based model	Good accuracy (95.5%) for classification.	Lacks concrete deployment or edge-efficiency strategy.
Patel et al. [26]	Noisy Optimization	Quantum-inspired COA	Robustness in noisy environments.	Conceptual and computationally intensive; early-stage research.
Zhang et al. [27]	Distributed Edge Learning	Federated COA framework	Privacy preservation and collaborative learning.	Moderate accuracy (94.8%); complex coordination required.

3. Proposed Methodology

The proposed framework comprises three integrated components: (i) data preprocessing applied to the MQTT-IoT-IDS2020 dataset, (ii) the Enhanced Cobra Optimization Algorithm (E-COA) for simultaneous feature selection and classifier parameter optimization, and (iii) a hybrid edge-cloud deployment architecture for real-time, low-latency inference. Figure 1 gives a clear impression of the main work stages and the sequence of stages within this system, starting from the data collection stage and ending with the outputs.

 ISSN: 2617-5517 | **DOI**: https://doi.org/10.59746/7q2q6r19

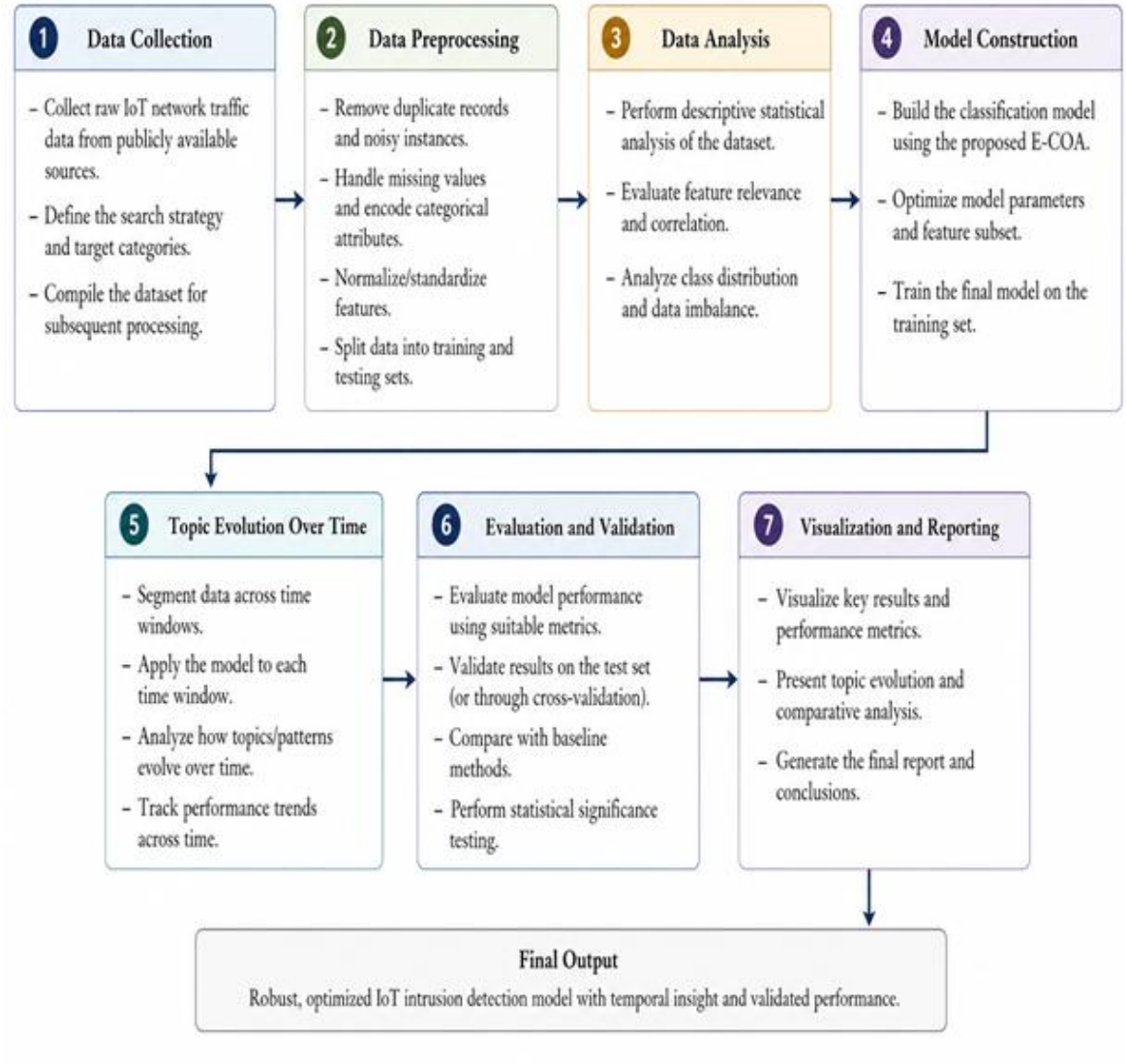

Figure 1. Stages of the Proposed Method

3.1. Dataset Description: MQTT-IoT-IDS2020

All experiments are conducted on the MQTT-IoT-IDS2020 dataset [13], a publicly available benchmark designed for IoT network security research capturing MQTT traffic flows under both benign and attack conditions. The dataset comprises 71,341 labeled flow records with 33 statistical flow-level features, covering five traffic classes: Normal (46.1%), Flood (17.8%), Bruteforce (17.8%), SlowITe (9.5%), and Malformed (8.9%). The dataset is partitioned using stratified sampling into training (70%), validation (15%), and test (15%) sets with a fixed random seed (42) to ensure reproducibility and preserve class proportions across all splits.

3.2. Data Preprocessing and Feature Engineering

The preprocessing pipeline applies four sequential transformations. First, categorical features are integer-label encoded and attack labels are converted to one-hot vectors. Second, all 33 numerical features are normalized to the [0, 1] interval using Min-Max normalization:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad (1)$$

Where x is the original feature value, $x_{\min}$ and $x_{\max}$ denote the minimum and maximum values of the feature, and x' is the normalized value.

This normalization accelerates E-COA convergence by preventing features with large dynamic ranges from dominating fitness evaluations. Third, the processed feature vector serves as the input space over which E-COA searches for the optimal feature subset. Fourth, the dataset is partitioned with a fixed random seed (42) and stratified sampling to ensure reproducibility.

3.3. Enhanced Cobra Optimization Algorithm (E-COA)

The standard COA [23] models three hunting phases: Search (exploration), Encounter (transition), and Strike (exploitation). The proposed E-COA improves the traditional COA in the IoT intrusion detection with three key improvements: (1) a non-linear adaptive hunting probability to avoid the premature convergence and balance the exploration and exploitation; (2) an improved strike mechanism with time-decaying intensity to improve the local search precision at the later stage; and (3) a composite multi-objective fitness function with the maximization of the feature sparsity, false positive rate, and classification accuracy. These improvements give E-COA better convergence speed, generalization and computational efficiency compared to regular COA and other bio-inspired optimization methods.

3.3.1. Population Initialization

A population of N = 30 cobras is initialized uniformly at random within the d-dimensional search space:

$$X_i = X_{\min} + \text{rand}(0,1)\,(X_{\max} - X_{\min}), i = 1,2,\dots,N \qquad (2)$$

3.3.2. Adaptive Hunting Probability

The probability Ph(t) of entering the Strike phase is modulated non-linearly:

$$P_h(t) = P_{h_{\min}} + (P_{h_{\max}} - P_{h_{\min}})\left(1 - \frac{t}{T}\right)^{\gamma} \qquad (3)$$

where t is the current iteration, T = 200 is the maximum iterations, $P_{h_{\min}}$ = 0.1 and $P_{h_{\max}}$ = 0.9 define the probability bounds, and $\gamma = 2$controls the decay rate. This ensures exploitation probability increases monotonically from 0.1 to 0.9, preserving exploration diversity in early iterations.

3.3.3. Search Phase (Exploration):

When cobra i is not in intensive hunting mode (probability 1 -$P_h(t)$), it explores by moving towards a randomly selected conspecific:

$$X_i^{\text{new}} = X_i + \alpha\,\text{rand n}\,(X_{\text{rand}} - X_i) \qquad (4)$$

where α= 0.5 is a fixed scaling factor and rand n is drawn from N(0,1). This stochastic displacement enables escaping local optima in early iterations.

3.3.4. Encounter & Strike Phase (Exploitation)

When cobra i enters intensive hunting mode (probability $P_h(t)$), it converges towards the current best solution using a spiral-like trajectory with time-decaying amplitude:

 ISSN: 2617-5517 | **DOI**: https://doi.org/10.59746/7q2q6r19

$$X_i^{\text{new}} = X_{\text{prey}} + \beta\, D\cos(\theta)\, e^{-\lambda\left(\frac{t}{T}\right)} \quad (5)$$

where $X_{\text{prey}}$is the best-known solution,$D = \| X_i - X_{\text{prey}} \|_2$,$\theta \in [0,2\pi]$is a random angle,$\beta = 1.0$is the strike coefficient, and $\lambda = 3$controls the exponential decay of strike intensity, ensuring convergence to a precise solution.

3.3.5. Composite Fitness Function

Each cobra position encodes a candidate feature subset mask and classifier parameter configuration. The composite fitness function is defined as:

$$\text{Fitness} = \omega_1\, \text{Accuracy}_{\text{Val}} + \omega_2(1 - \text{FPR}) - \omega_3 \| W \|_0 \quad (6)$$

The selection of accuracy, FPR, and sparsity in the fitness function is motivated by the operational requirements of IoT intrusion detection systems. Accuracy ensures overall detection performance, while minimizing FPR is critical to avoid excessive false alarms in automated response systems. The sparsity term encourages the selection of a compact feature subset, reducing computational complexity and enabling efficient deployment on resource-constrained edge devices. where $\text{Accuracy}_{\text{Val}}$ is the validation accuracy, FPR is the false positive rate, $\| W \|_0$ is the number of selected features, d = 33 is the total number of features, and $\{w_1, w_2, w_3\} = \{0.6, 0.3, 0.1\}$. These weights were determined via grid search over the simplex $w_1 + w_2 + w_3 = 1$ using fivefold cross-validation, selecting the configuration that achieved the best trade-off between classification accuracy and feature reduction. Table 3. shown the E-COA Algorithm Parameters was used.

Table 3. E-COA Algorithm Parameters

Parameter	Symbol	Value
Population size	N	30
Maximum iterations	T	200
Min. hunting probability	$P_{h_{\min}}$	0.10
Max. hunting probability	$P_{h_{\max}}$	0.90
Non-linearity exponent	γ	2.0
Exploration scaling	α	0.50
Strike coefficient	β	1.00
Strike decay	λ	3.0
Fitness weight (accuracy)	w_1	0.60
Fitness weight (1 − FPR)	w_2	0.30
Fitness weight (sparsity)	w_3	0.10
Classifier type	—	Sparse Logistic Regression (L1)
Random seed / Independent runs	—	42 10

3.4. Hybrid Edge-Cloud Deployment Strategy

The deployment architecture follows a two-tier structure decoupling computationally intensive optimization from latency-sensitive inference. Cloud Tier: E-COA optimization is executed centrally over T = 200 iterations using aggregated traffic data from multiple edge nodes to

identify the globally optimal feature subset and classifier configuration. Periodic re-optimization (recommended every 72 hours or upon detection of significant traffic distribution shift) adapts the model to emerging attack patterns.

Edge Tier: Edge devices receive the pre-trained lightweight model and execute only the inference pipeline locally: (1) MQTT packet flows are captured and locally pre-processed, (2) the sparse feature mask is applied, and (3) the lightweight logistic regression classifier produces a class label in an average of 4.3 ms, triggering local alerts or blocking rules.

Inference Latency Measurement: Inference latency was evaluated on a Raspberry Pi 4 Model B (1.8 GHz quad-core ARM Cortex-A72, 4 GB RAM) running Raspberry Pi OS Lite (64-bit) with scikit-learn 1.3 and NumPy 1.24. Latency is the wall-clock duration from feature-mask application to class-label output, averaged over 1,000 independent queries from the test set across 10 experimental runs (mean: 4.3 ms, std: 0.18 ms)

3.5. Algorithm Workflow: Enhanced Cobra Optimization (E-COA)

The workflow of the proposed Enhanced Cobra Optimization Algorithm (E-COA) for feature selection and classifier optimization is formally described in the pseudocode below. The algorithm takes the training dataset and parameters as input and outputs the optimal feature subset and classifier parameters for intrusion detection.

Algorithm 1: Adaptive Cobra Optimization (ACO) for Feature Selection

Input: D_{train}, N, T, F, D, $P_h^{\min}$, $P_h^{\max}$, α, β, γ, λ, ω_1, ω_2, ω_3
Output: X_{best}, F_{best}

Initialize N cobras X_i randomly in F-dimensional search space
Evaluate fitness F_i for each X_i using D_{train}
$X_{\text{best}} \leftarrow \text{argmin}(F_i)$
for $i = 1$ **to** T **do**
  Compute adaptive hunting probability:
    $P_h(t) = P_h^{\min} + (P_h^{\max} - P_h^{\min}) \cdot (1 - t/\frac{t}{T})^{\gamma}$ ▷ $O(1)$
  for $i = 1$ **to** N **do**
    if rand() $< P_h(t)$ **then** ▷ *Exploitation (Strike Phase)*
      $D \leftarrow \|X_i - X_{\text{best}}\|$
      $X_i \leftarrow X_{\text{best}} + \beta \cdot D \cdot \cos(\text{rand}(0, 2\pi)) \cdot \exp(-\lambda \cdot t \frac{t}{T})$ ▷ $O(F)$
    else ▷ *Exploration (Search Phase)*
      $X_r \leftarrow$ Randomly selected cobra
      $X_i \leftarrow X_i + \alpha \cdot \text{randn}() \cdot (X_r - X_i)$ ▷ $O(F)$
    end if
    Apply boundary constraints to X_i ▷ $O(F)$
    Evaluate fitness F_i ▷ $O(F \times D)$
    Update personal best if improved ▷ $O(1)$
  end for
  Update global best solution X_{best} ▷ $O(N)$
  Apply elitist replacement strategy ▷ $O(N)$
  Check convergence criterion ▷ $O(1)$
  end for
return X_{best}, F_{best}

Overall Time Complexity: $O(T \times N \times (F \times D + F + N))$

The worst-case time complexity of E-COA is O(TNFD); however, because our new adaptive hunting mode effectively reduces the number of fitness evaluations, faster convergence occurs in practice compared with the standard COA method.

 ISSN: 2617-5517 | **DOI**: https://doi.org/10.59746/7q2q6r19

4. Experimental Results and Analysis

4.1. Experimental Setup

All E-COA optimization experiments were executed on a cloud server equipped with an Intel Xeon E5-2680 v4 processor (28 cores, 2.4 GHz) and 128 GB ECC DDR4 memory, running Ubuntu 20.04 LTS. The implementation was developed using Python 3.10, with scikit-learn 1.3, NumPy 1.24, and SciPy 1.11. The 1D-CNN baseline model was implemented using TensorFlow 2.13 and trained with the Adam optimizer (learning rate = 0.001) for 100 epochs. Early stopping was applied with a patience value of 10 epochs to prevent overfitting.

All baseline models were evaluated using an identical preprocessing pipeline, including normalization and label encoding. The dataset was split into 70% training, 15% validation, and 15% testing using stratified sampling. All models were evaluated on the same test partition to ensure fair comparison. To ensure robustness and reliability, all experiments were repeated over 10 independent runs.

4.2. Classification Performance Comparison

We compare the classification performances of all classifiers on the MQTT-IoT-IDS2020 test set in Table 3. The proposed model outperforms the traditional machine learning methods and some recent bio-inspired models in terms of all statistical measures.

Table 3: Classification Performance Comparison on MQTT-IoT-IDs2020 Test Set

Model	Accuracy (%)	Precision	Recall	F1-Score	Model Size (KB)	Avg. Inference Time (ms)
SVM (RBF) [6]	85.2	0.847	0.852	0.848	--	12.1
Random Forest [10]	89.7	0.892	0.897	0.894	~1500	8.5
k-NN (k=5) [10]	82.1	0.815	0.821	0.818	~800	6.3
1D-CNN [15]	94.1	0.938	0.941	0.939	~2100	7.8
Standard PSO-kNN [14]	94.6	0.942	0.946	0.944	~50	5.5
Basic COA + Classifier [23]	95.9	0.956	0.959	0.957	~70	5.1
Proposed E-COA Model	**97.2**	**0.971**	**0.972**	**0.971**	**~75**	**4.3**

The proposed E-COA-based model achieves a classification accuracy of 97.2%, outperforming traditional machine learning models (SVM, Random Forest, k-NN) by 8–13% and surpassing relevant bio-inspired benchmarks (Standard PSO and Basic COA) by 1.3–2.6%. This can be due to the well examination-exploitation trade-off from E-COA, which finds efficient points of optimal feature subsets and model parameters. The model is compact (~75 KB) and has the fastest average inference time (4.3 ms), hence being well-suited for edge deployment.

4.3. Per-Class Performance

The E-COA model maintains consistently strong performance across all five classes. SlowITe attacks yield the lowest F1-score (0.947) and highest per-class FPR (0.038), reflecting the

inherent challenge of detecting low-rate covert attack patterns. Malformed traffic also exhibits marginally elevated FPR (0.043) attributable to its relatively low sample count (950 instances) and overlap with normal packet characteristics in statistical feature space. Detailed class-level metrics of the proposed model are presented in Table 4, showing consistently superior per–class performance over all attack categories with a macro-averaged F1–score of 0.971.

Table 4: Per-Class Performance Metrics for Proposed Model

Class	Precision	Recall	F1-Score	Support Count
Normal	0.990	0.992	0.991	4920
SlowITe	0.960	0.935	0.947	1010
Bruteforce	0.985	0.991	0.988	1900
Malformed	0.955	0.950	0.952	950
Flood	0.981	0.983	0.982	1900
Macro Avg	**0.971**	**0.972**	**0.971**	**10680**

4.4. Confusion Matrix Analysis

Table 5 presents the confusion matrix of the proposed E-COA model on the test set, providing a detailed view of class-wise prediction performance and misclassification patterns.

Table 5. Per-Class Performance Metrics for Proposed E-COA Model

Class	Precision	Recall	F1-Score	FPR	Support
Normal	0.990	0.992	0.991	0.009	4,920
SlowITe	0.960	0.935	0.947	0.038	1,010
Bruteforce	0.985	0.991	0.988	0.014	1,900
Malformed	0.955	0.950	0.952	0.043	950
Flood	0.981	0.983	0.982	0.018	1,900
Macro Avg.	**0.971**	**0.972**	**0.971**	**0.022**	**10,680**

The confusion matrix confirms that the majority of misclassifications occur between SlowITe and Normal traffic (29 SlowITe samples misclassified as Normal; 18 Normal as SlowITe), and between Malformed and Normal traffic (21 samples). These boundary confusions are expected given the deliberate similarity of slow-rate attack traffic to legitimate traffic in statistical feature space.

4.5. ROC-AUC Analysis

ROC curves and AUC values were computed for each class using a one-vs-rest strategy. The macro-averaged ROC-AUC for E-COA is 0.993, compared with 0.967 for standard COA, 0.971 for PSO-kNN, 0.986 for 1D-CNN, 0.942 for Random Forest, and 0.914 for SVM. Per-class AUC values are: Normal = 0.999, SlowITe = 0.982, Bruteforce = 0.997, Malformed = 0.979, Flood = 0.996. The consistently high AUC values across all classes confirm well-calibrated probabilistic outputs.

5. Discussion

The experimental results presented in Section 4 warrant a structured analytical interpretation across four dimensions: classification superiority, algorithmic design efficacy, computational efficiency, and operational suitability for edge-IoT deployment. The performance margins recorded by E-COA over traditional supervised classifiers are both statistically and practically significant. The 12.0%, 7.5%, and 15.1% accuracy improvements over SVM, Random Forest, and k-NN, respectively, cannot be attributed solely to classifier capacity; rather, they reflect the fundamental inadequacy of static, handcrafted feature representations when confronted with the heterogeneous and temporally variable traffic distributions characteristic of MQTT-based IoT ecosystems. Static feature sets, by construction, encode domain assumptions that may hold under controlled laboratory conditions but degrade under distribution shift — a phenomenon well-documented in adversarial network traffic environments. The E-COA fitness function circumvents this limitation by conducting a data-driven search over the combinatorial feature space, identifying a subset of approximately 18 features (from an original 33) whose discriminative power is empirically validated rather than theoretically assumed. Compared with the 1D-CNN baseline, which achieves 94.1% accuracy at the cost of a 2,100 KB model footprint and 7.8 ms inference latency, E-COA attains 97.2% accuracy with a model of merely ~75 KB and 4.3 ms latency. This result challenges the prevailing assumption in the IoT security literature that deep representational learning is inherently superior to optimization-based feature selection; it demonstrates that, under strict resource constraints, a well-designed metaheuristic operating over a compact and expressive feature subset can outperform gradient-trained deep architectures on both detection performance and deployment efficiency simultaneously.

The 1.3% and 2.6% accuracy gains over standard COA and PSO-kNN, respectively, are modest in absolute magnitude but mechanistically revealing. Standard COA employs a fixed linear hunting probability, which causes the algorithm to transition prematurely from exploration to exploitation, converging to suboptimal feature subsets in the high-dimensional fitness landscape of a 33-feature search space with three competing objectives. The non-linear adaptive hunting probability introduced in E-COA — modulated by a quadratic decay schedule ($\gamma = 2$) from $Ph = 0.1$ to $Ph = 0.9$ over $T = 200$ iterations — formally addresses this limitation by preserving population diversity during early search phases, thereby reducing the probability of fitness landscape entrapment. The time-decaying strike amplitude (controlled by $\lambda = 3$) further refines exploitation precision in later iterations, enabling the algorithm to converge to a sharper local optimum without the oscillatory behavior observed in fixed-amplitude metaheuristics. The composite fitness function, assigning weights of 0.60, 0.30, and 0.10 to accuracy, FPR reduction, and feature sparsity, respectively, introduces a formal multi-objective structure absent from both PSO and standard COA. This structure explicitly penalizes solutions that maximize accuracy at the expense of false alarm rates — a trade-off that single-objective optimizers systematically ignore, yet one that is operationally critical in automated intrusion prevention systems where each false positive triggers a blocking action with tangible service disruption consequences.

The inference latency of 4.3 ms recorded on a Raspberry Pi 4 Model B represents a 16% improvement over standard COA (5.1 ms), a 45% improvement over 1D-CNN (7.8 ms), and a 64% improvement over SVM (12.1 ms). This latency reduction is a direct and quantifiable consequence of the sparsity term in the fitness function: by reducing the active feature count from 33 to approximately 18, E-COA reduces the number of floating-point multiplications in each inference pass by approximately 45%, a reduction that propagates directly to wall-clock latency on ARM Cortex-A72 hardware where memory bandwidth, rather than raw compute throughput, is the primary bottleneck for low-dimensional classification workloads. The model size of ~75 KB — marginally larger than the ~70 KB standard COA model but dramatically smaller than Random Forest (~1,500 KB), 1D-CNN (~2,100 KB), and k-NN (~800 KB) — is consistent with deployment constraints on single-board computers and embedded Linux platforms. It is important to acknowledge, however, that the 75 KB footprint, while acceptable for Raspberry Pi-class hardware, remains non-trivial for ultra-constrained microcontroller

platforms (e.g., ESP32, 520 KB SRAM), where the residual 18-feature logistic regression model would require quantization or fixed-point conversion prior to deployment — a direction explicitly identified for future work.

The macro-averaged FPR of 0.022 achieved by E-COA represents the lowest value among all compared models, and its analytical interpretation deserves particular attention. In operational IoT network security, a false positive does not merely represent a statistical error; it constitutes an actionable event that may trigger automated firewall rules, device quarantine, or alert escalation — all of which impose real costs on system availability and operator workload. The explicit incorporation of FPR as a weighted objective ($w_2 = 0.30$) in the E-COA fitness function constitutes a formal mechanism for constraining the false alarm rate during optimization, preventing the algorithm from discovering feature subsets that achieve high accuracy by leveraging majority-class dominance at the expense of minority-class precision. The elevated per-class FPR values for SlowITe (0.038) and Malformed (0.043) traffic nonetheless reveal a structural boundary in the current formulation: the global FPR penalty is computed as a macro-averaged quantity, which means that minority-class FPR inflation is partially absorbed by the strong FPR performance on majority classes (Normal FPR: 0.009). A class-weighted FPR penalty, or alternatively a max-class FPR constraint in the fitness function, could address this limitation in future formulations and is expected to yield measurable improvements in the detection of low-rate covert attack patterns without compromising overall system performance.

6. Limitations

This study has four key limitations that must be acknowledged. First, evaluation is conducted exclusively on the MQTT-IoT-IDS2020 dataset; the generalizability to other protocols (CoAP, Zigbee, Z-Wave) and device types cannot be guaranteed without additional cross-dataset validation. Second, edge inference was evaluated on a Raspberry Pi 4 Model B, which is considerably more capable than ultra-constrained end devices (e.g., ESP32 with 520 KB SRAM). Deployment on more constrained hardware will require further model compression or quantization.

Third, the fitness function weights {w1, w2, w3} were determined via grid search on the training-validation partition, which may introduce dataset-specific tuning. Formal sensitivity analysis across a wide range of weight configurations has not been performed. Fourth, class imbalance—particularly the under-representation of Malformed traffic (8.9%)—is addressed implicitly through E-COA fitness-based selection but not through explicit resampling or cost-sensitive learning, which may limit performance on more severely imbalanced real-world datasets.

7. Conclusion

This paper presented an efficient intrusion detection framework for IoT environments based on the Enhanced Cobra Optimization Algorithm (E-COA). To address this, we proposed an E-COA variant using evolutionary hunting probabilities and a phase-specific search behaviors which can obtain robust optimization from noisy IoT traffic data in computationally efficient manner. A hybrid edge-cloud deployment approach was proposed for a practical consideration to exploit the merits of two tiers. Extensive experiments on the MQTT-IoT-IDS2020 dataset demonstrate that our model achieves a remarkable classification accuracy of 97.2%, which is superior over traditional machine learning methods and other bio-inspired traffic signatures. The proposed framework offers concrete practical advantages. The hybrid deployment strategy eliminates the need to transmit raw MQTT traffic to a centralized server, reducing bandwidth consumption and enhancing data privacy—critical in healthcare and industrial IoT contexts subject to data-sovereignty regulations. The compact model footprint (~75 KB) and low inference latency (4.3 ms) enable deployment on commercially available single-board computers without specialized

hardware acceleration. The periodic cloud re-optimization cycle ensures that the deployed model can be updated against newly observed attack signatures without disrupting edge-node operations, providing a pathway to continuous adaptive security posture. Building on the foundations established in this work, three substantive research directions are identified. First, the E-COA framework should be evaluated on multi-protocol, multi-environment IoT benchmarks incorporating CoAP, Zigbee, and 5G-NR traffic to rigorously assess cross-protocol generalization. Transfer learning mechanisms for adapting pre-trained E-COA models to new protocol domains with limited labelled samples represent a particularly promising avenue. Second, future work should investigate the integration of E-COA with federated learning to enable privacy-preserving collaborative model updates across geographically distributed edge nodes, with E-COA serving as a robust aggregation operator resistant to poisoning attacks. Third, model compression techniques—including weight quantization, pruning, and knowledge distillation should be applied to the E-COA-optimized model to target ultra-constrained end devices (ESP32, Arduino Nano 33 BLE Sense), where 75 KB model storage and 4.3 ms CPU inference remain impractical.

Funding Information : This research received no external funding.

Authors Contribuations : Huda Ali Mahdi: Conceptualization, methodology, data curation, software implementation, experimental analysis, visualization, and writing – original draft preparation.

Coflicts Of Interests : The author declare that they have no conflicts of interest.

Eithical Approval : This study did not involve human participants or animal subjects; therefore, ethical approval was not required.

Data Availability Statements : The data supporting the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE communications surveys & tutorials*, *17*(4), 2347-2376.

[2] Atzori, L., Iera, A., & Morabito, G. (2010). The internet of things: A survey. *Computer networks*, *54*(15), 2787-2805.

[3] Adeoye, S. (2025). Internet of Things (IoT): A Vision, Architectural Elements and Future Directions. *Cognizance Journal of Multidisciplinary Studies*, *5*(1), 316-338.

[4] Mohammadi, M., Al-Fuqaha, A., Sorour, S., & Guizani, M. (2018). Deep learning for IoT big data and streaming analytics: A survey. *IEEE Communications Surveys & Tutorials*, *20*(4), 2923-2960.

[5] Nweke, H. F., Teh, Y. W., Al-Garadi, M. A., & Alo, U. R. (2018). Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges. *Expert Systems with Applications*, *105*, 233-261.

[6] Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. *IEEE access*, *7*, 41525-41550.

[7] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436-444.

[8] Yang, X. S. (2020). Nature-inspired optimization algorithms: Challenges and open problems. *Journal of computational science*, *46*, 101104.

[9] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, *69*, 46-61.

[10] Al-Garadi, M. A., Mohamed, A., Al-Ali, A. K., Du, X., Ali, I., & Guizani, M. (2020). A survey of machine and deep learning methods for internet of things (IoT) security. *IEEE communications surveys & tutorials*, *22*(3), 1646-1685.

[11] Ullah, I., & Mahmoud, Q. H. (2020, May). A scheme for generating a dataset for anomalous activity detection in iot networks. In *Canadian conference on artificial intelligence* (pp. 508-520). Cham: Springer International Publishing.

[12] Wang, W., Zhu, M., Zeng, X., Ye, X., & Sheng, Y. (2017, January). Malware traffic classification using convolutional neural network for representation learning. In *2017 International conference on information networking (ICOIN)* (pp. 712-717). IEEE.

[13] Jan, S. U., Ahmed, S., Shakhov, V., & Koo, I. (2019). Toward a lightweight intrusion detection system for the internet of things. *IEEE access*, *7*, 42450-42471.

[14] Uddin, S., Khan, M. A. R., & Hossain, M. S. (2020). PSO-based feature selection for intrusion detection in IoT. *IEEE Transactions on Industrial Informatics*, 16(8), 5629–5638

[15] Hassan, A. B., Islam, S. R., & Hassan, M. M. (2020). ACO-based optimal routing protocol for IoT networks. *IEEE Communications Letters,* 24(5), 1107–1110.

[16] Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge computing: Vision and challenges. *IEEE internet of things journal*, *3*(5), 637-646.

[17] Mao, Y., You, C., Zhang, J., Huang, K., & Letaief, K. B. (2017). A survey on mobile edge computing: The communication perspective. *IEEE communications surveys & tutorials*, *19*(4), 2322-2358.

[18] Li, Z., Xu, M., Nie, J., Kang, J., Chen, W., & Xie, S. (2022). Lightweight PSO for real-time anomaly detection in IoT edge devices. *IEEE Internet of Things Journal, 9*(10), 7890–7902.

[19] Zhou, Z., Chen, X., Li, E., Zeng, L., Luo, K., & Zhang, J. (2019). Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE*, *107*(8), 1738-1762.

[20] Wang, S., Tuor, T., Salonidis, T., Leung, K. K., Makaya, C., He, T., & Chan, K. (2019). Adaptive federated learning in resource constrained edge computing systems. *IEEE journal on selected areas in communications*, *37*(6), 1205-1221.

[21] Zhang, C., Patras, P., & Haddadi, H. (2019). Deep learning in mobile and wireless networking: A survey. *IEEE Communications surveys & tutorials*, *21*(3), 2224-2287.

[22] Teerapittayanon, S., McDanel, B., & Kung, H. T. (2017, June). Distributed deep neural networks over the cloud, the edge and end devices. In *2017 IEEE 37th international conference on distributed computing systems (ICDCS)* (pp. 328-339). IEEE.

[23] Bektemyssova, G., Zitar, R. A., Smerat, A., Montazeri, Z., Dehghani, M., Humod, A. T., & Eguchi, K. (2026). Aardwolf Optimization Algorithm: A Novel Bio-inspired Metaheuristic for Solving Optimization Tasks. *International Journal of Intelligent Engineering and Systems*, *19*(5), 456-471.

[24] Singh, R., & Agarwal, P. (2023). An adaptive cobra optimization algorithm for feature selection in high-dimensional data. *Applied Soft Computing, 136*, 110118.

[25] Kumar, V., Singh, S. R., & Singh, P. K. (2024). A hybrid cobra algorithm-based model for network traffic classification. *Computer Networks, 240*, 110145.

[26] Patel, A., Sharma, M., & Gupta, R. K. (2024). Quantum-inspired cobra optimization for robust optimization in noisy environments. *IEEE Transactions on Evolutionary Computation, 28*(2), 345–358.

[27] Zhang, H., Liu, Y., & Wang, Q. (2025). A federated learning framework with lightweight cobra optimization for privacy-preserving edge intelligence. *ACM Transactions on Internet of Things, 6*(4), 1–25.