Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.084624

ARTICLE

# DMSALA: A Dynamic Multi-Subpopulation Artificial Lemming Algorithm for Feature Selection in IoT Intrusion Detection

**Hui Xu, Ruiqi Qu* and Xinlu Zong**

School of Computer Science and Artificial Intelligence, Hubei University of Technology, Wuhan, China

*Corresponding Author: Ruiqi Qu. Email: 102411222@hbut.edu.cn

Received: 26 April 2026; Accepted: 29 July 2026; Published: 15 September 2026

**ABSTRACT:** With the rapid growth of the Internet of Things, intrusion detection systems face severe challenges in processing massive, high-dimensional, and redundant network traffic while satisfying strict low-latency and high-efficiency requirements. To address these challenges,this paper improves the original artificial lemming algorithm (ALA) and proposes a dynamic multi-subpopulation artificial lemming algorithm (DMSALA) for feature selection, and then constructs an intrusion detection framework for IoT based on DMSALA. The proposed DMSALA introduces an adaptive clustering-based dynamic multi-subpopulation structure to alleviate premature convergence during the search process. In addition, a cosine-based nonlinear weighting strategy is designed to achieve a smooth transition toward the global optimum during optimization, while an adaptive Gaussian perturbation mechanism is incorporated to enhance local exploitation capability. To verify the effectiveness of the proposed DMSALA, its optimization performance is first evaluated on the CEC2017 benchmark functions and compared with several other classical algorithms. The experimental results show that DMSALA exhibits better convergence accuracy and stability on the benchmark problems. Furthermore, the NF-ToN-IoT-v2 and NF-BoT-IoT-v2 dataset are employed to conduct feature selection experiments, where classification performance, feature reduction capability and computational cost are comprehensively analyzed. In both multiclass and binary classification tasks, the proposed DMSALA strikes a better trade-off between feature reduction and classification accuracy. The results indicate that DMSALA can provide a reasonable balance between detection effectiveness and computational overhead, making it more suitable for low-latency and efficient IoT intrusion detection scenarios.

**KEYWORDS:** Internet of Things; intrusion detection; artificial lemming algorithm; feature selection

## 1 Introduction

The Internet of Things (IoT) drives digital transformation through widespread sensor deployment, enabling comprehensive perception and intelligent processing. However, due to open interconnection and resource-constrained heterogeneous devices, IoT systems face a broader attack surface than traditional networks, making them difficult to secure through single preventive mechanisms [1]. Consequently, Intrusion Detection Systems (IDS) are essential for continuously monitoring network traffic to identify potential attacks and ensure overall IoT security [2].

Extensive research has explored machine learning-based IDS for IoT environments. Adewole et al. evaluated various ensemble learning models to construct a supervised framework for IoT network attack detection [3]. Eid et al. optimized several classical models with data preprocessing and class balancing strategies to counter typical DDoS threats in the Industrial IoT (IIoT) [4]. From an interpretability perspective,

Jamshidi et al. comparatively analyzed the performance and resource overhead of different models in IoT edge gateways [5].

Deep learning methods have also been widely adopted to capture complex traffic patterns. Nandanwar and katarya proposed AttackNet, a hybrid Convolutional Neural Network (CNN)-Gated Recurrent Unit (GRU) model designed to extract spatial-temporal dependencies in IIoT scenarios [6]. To improve cross-scenario applicability, Ahmad et al. utilized deep transfer learning based on a pretrained CNN, enabling efficient generalization for unknown attacks [7]. Additionally, Abiha et al. proposed a cGAN-enhanced ensemble deep learning framework with adversarial training for resilient anomaly detection in heterogeneous IoT environments [8].

Despite these advancements, IoT traffic features are often high-dimensional and redundant. Modeling directly on such data can cause the curse of dimensionality, introduce noise, and increase deployment costs. Feature selection mitigates this by retaining the most informative subset, making the IDS more suitable for resource-constrained edge environments [9]. In this domain, Ayad et al. proposed a hybrid strategy combining correlation analysis and recursive feature elimination (RFE) to reduce inference overhead [10]. Wakili and Bakkali integrated statistical filtering, RFE, and tree-based embedding through an ensemble weighted ranking strategy [11]. For large-scale data, Wang et al. employed a multi-criteria Las Vegas Wrapper to iteratively screen optimal feature subsets [12]. In IIoT, Awotunde et al. utilized the $\chi^2$ statistic to remove redundant features before feeding the data into tree-based classifiers [13].

Metaheuristic algorithms provide another effective approach by formulating feature selection as an optimization problem to systematically explore high-dimensional spaces [14]. Gong et al. proposed HMDOA, which integrates multiple swarm algorithms (ABC, SSA, GTO) to optimize feature subsets and improve classification efficiency [15]. Abid et al. applied the grey wolf optimizer (GWO) alongside a Light Gradient Boosting Machine (LightGBM)-CNN hybrid detector to enhance IoT intrusion detection accuracy [16].

Our prior work [17–21] involves the application of metaheuristic optimization algorithms to feature selection, intrusion detection, and traffic identification in traditional network environments.

ALA is a swarm intelligence optimizer inspired by lemming behaviors, dynamically balancing exploration and exploitation via an energy factor [22]. Recent improvements to ALA include EALA, which incorporates Kent chaotic initialization and hybrid mutation for UAV path planning [23], and MsIALA, which combines cubic chaotic initialization and adaptive perturbation to accelerate convergence in complex routing environments [24].

To address feature redundancy and improve IDS efficiency in resource-constrained IoT environments, this paper proposes a Dynamic Multi-Subpopulation Artificial Lemming Algorithm (DMSALA) for feature selection. DMSALA employs an adaptive clustering-based dynamic multi-subpopulation structure to maintain population diversity, a cosine-based nonlinear weighting strategy for staged subpopulation guidance, and adaptive Gaussian perturbation to enhance local exploitation. Additionally, a Sigmoid function maps the continuous solution space into a binary space, yielding a compact and discriminative feature subset. Unlike operator-focused ALA variants, DMSALA innovates through a dynamic multi-subpopulation architecture that synergizes nonlinear weighting with Gaussian perturbation for superior exploration–exploitation balance.

The main contributions of this paper are summarized as follows:

(a) An adaptive clustering-based dynamic multi-subpopulation structure is proposed to reduce over-reliance on a single global optimum, thereby enhancing population diversity and mitigating premature convergence in multimodal scenarios.

(b) A DMSALA-based feature selection framework is constructed for IoT intrusion detection, effectively reducing computational costs while jointly considering classification performance and the number of selected features.

The remainder of this paper is organized as follows. Section 2 introduces the basic ALA. Section 3 details the proposed DMSALA method. Section 4 describes the IoT intrusion detection framework. Section 5 presents the experimental results and analysis. Finally, Section 6 concludes the paper and discusses future work.

## 2 Artificial Lemming Algorithm

ALA is a swarm intelligence method inspired by four lemming behaviors: migration, burrowing, foraging, and predator avoidance. The transition between global exploration and local exploitation is governed by a decaying energy factor $E(t)$:

$$E(t) = 4 \times \arctan\left(1 - \frac{t}{T_{max}}\right) \times \ln\left(\frac{1}{\text{rand}}\right). \tag{1}$$

When $E(t) > 1$, ALA performs the exploration phase. Depending on a random probability (e.g., rand < 0.5), it expands the search space either through long-distance migration using Brownian motion ($\overrightarrow{BM}$) or through burrowing:

$$\vec{Z}_i(t+1) = \begin{cases} \vec{Z}_{best}(t) + F \times \overrightarrow{BM}, & \text{rand} < 0.5 \\ \vec{Z}_i(t) + F \times L \times \left(\vec{Z}_{best}(t) - \vec{Z}_b(t)\right), & \text{rand} > 0.5 \end{cases} \tag{2}$$

where $F \in \{1, -1\}$ is a directional sign, $\vec{Z}_{best}(t)$ is the current best individual, $\vec{Z}_b(t)$ is a randomly selected individual, and $L$ is a cosine-based adjustment coefficient.

Conversely, when $E(t) \leq 1$, ALA enters the exploitation phase. It performs local search around the optimum via foraging (spiral wrapping) or predator avoidance (Lévy flight), dictated by another random probability (e.g., rand < 0.3):

$$\vec{Z}_i(t+1) = \begin{cases} \vec{Z}_{best}(t) + F \times \text{spiral} \times \text{rand} \times \vec{Z}_i(t), & \text{rand} < 0.3 \\ \vec{Z}_{best}(t) + F \times G \times \text{Levy}(Dim) \times \left(\vec{Z}_{best}(t) - \vec{Z}_i(t)\right), & \text{rand} > 0.3 \end{cases} \tag{3}$$

where spiral is a perturbation term for local search, $G$ is a linearly decreasing escape coefficient, and $\text{Levy}(Dim)$ provides random step sizes to prevent stagnation at local optima.

## 3 Proposed Improvement Strategies

### *3.1 Dynamic Multi-Subpopulation Structure*

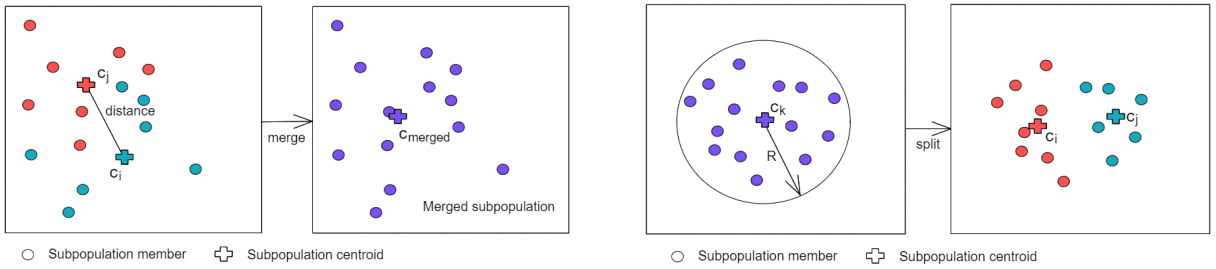
To overcome the original ALA's premature convergence caused by over-reliance on a single global optimum, we introduce an adaptive clustering-based dynamic multi-subpopulation structure [25,26]. At iteration $t$, the population is partitioned into $K(t)$ independent subpopulations $P_k(t)$. For each $P_k(t)$, its centroid $c_k(t)$, local optimal individual $L_k(t)$, and the global optimum $G^*(t)$ are defined as:

$$c_k(t) = \frac{1}{|P_k(t)|} \sum_{x \in P_k(t)} x, \quad L_k(t) = \arg\min_{x \in P_k(t)} f(x), \quad G^*(t) = \arg\min_{x \in P(t)} f(x). \tag{4}$$

To avoid stagnation, a dynamic reconstruction strategy is employed (Fig. 1). Subpopulations $i$ and $j$ are merged to reduce search overlap if their centroids are too close, where the merge threshold is set to 0.05 times

the search-space scale. Conversely, subpopulation $k$ is split to improve local search precision if its individuals are overly dispersed, with the split threshold set to 0.15 times the search-space scale. The merge and split criteria are respectively defined as:

$$\|c_i(t) - c_j(t)\| < d_{\text{merge}}, \qquad \max_{x \in P_k(t)} \|x - c_k(t)\| > r_{\text{split}}. \tag{5}$$




(**a**) Dynamic subpopulation merging (**b**) Dynamic subpopulation splitting

**Figure 1:** Schematic of dynamic merging and splitting mechanisms.

### *3.2 Hybrid Guidance with Cosine-Based Nonlinear Weighting Strategy*

Based on the dynamic multi-subpopulation structure, to achieve a phased balance that emphasizes subpopulation-level exploration in the early stage and global convergence in the later stage, a cosine-based nonlinear weighting strategy is designed.

$$w(t) = \frac{1}{2}\left(1 + \cos\left(\frac{\pi t}{T}\right)\right), \qquad 0 \le t \le T, \tag{6}$$

where $t$ is the current iteration number, and $T$ is the maximum iteration number. This weight satisfies $w(0) = 1$ and $w(T) = 0$, indicating that the search prioritizes subpopulation-level guidance in early iterations and gradually shifts to global guidance as iterations proceed.

Based on this weight, a hybrid guidance vector $B_i(t)$ is constructed for an individual $x_i(t)$ (belonging to subpopulation $k(i)$):

$$B_i(t) = w(t)\, L_{k(i)}(t) + \big(1 - w(t)\big)\, G^*(t). \tag{7}$$

### *3.3 Adaptive Gaussian Perturbation*

To mitigate the decline in exploitation capability during the later stages, an adaptive Gaussian random perturbation is introduced in the exploitation phase. It is designed to decay progressively with iterations, enhancing fine-tuning capabilities and solution stability without disrupting the convergence trend [27,28]. While converging toward the subpopulation optimum, a time-decaying Gaussian micro-perturbation is superimposed to improve final convergence accuracy. When an individual triggers exploitation behavior, the following update rule is applied:

$$X_i(t+1) = X_i(t) + \alpha(t)\Big(L_{k(i)}(t) - X_i(t)\Big) + \varepsilon_i(t), \tag{8}$$

where $\alpha(t)$ is a contraction coefficient that increases with iterations, driving individuals to gradually approach the subpopulation optimum:

$$\alpha(t) = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})\frac{t}{T}, \qquad 0 < \alpha_{\min} < \alpha_{\max} \le 1. \tag{9}$$

The perturbation term $\varepsilon_i(t)$ employs zero-mean Gaussian noise, with its scale adaptively decaying over time:

$$\varepsilon_i(t) \sim \mathcal{N}\big(0,\ \mathrm{diag}(\sigma^2(t))\big), \qquad \sigma(t) = \eta(t)\left(1 - \frac{t}{T}\right)(\mathrm{ub} - \mathrm{lb}), \tag{10}$$

where $\eta(t)$ is a perturbation scaling factor that can be adjusted in conjunction with stagnation detection.

### *3.4 Time Complexity Analysis*

Time complexity is a crucial metric for evaluating algorithmic efficiency. Let $N$ be the population size, $T$ be the maximum number of iterations, $D$ be the problem dimension, and $K$ be the number of subpopulations. The maximum number of function evaluations is $t_{\max} = NT$. In each generation, DMSALA primarily consists of three phases: dynamic multi-subpopulation reconstruction, position updating, and fitness evaluation.

Specifically, the reconstruction phase involves updating centroids and local optima, as well as splitting and merging subpopulations, which take $O(ND + K^2D)$ time per generation. The position updating requires a $D$-dimensional vector operation for $N$ individuals, taking $O(ND)$ time. Assuming the cost of a single function evaluation is constant, the fitness evaluation takes $O(N)$ time per generation.

By aggregating the computational cost of these three phases over $T$ iterations, the overall time complexity of DMSALA can be expressed as:

$$O(\mathrm{DMSALA}) = \underbrace{O\big(T(ND + K^2D)\big)}_{\text{Reconstruction}} + \underbrace{O(TND)}_{\text{Update}} + \underbrace{O(TN)}_{\text{Evaluation}} \approx O(TND) = O(t_{\max}D). \tag{11}$$

Since the number of subpopulations is typically much smaller than the population size ($K \ll N$), the additional overhead introduced by the dynamic multi-subpopulation reconstruction is negligible. Consequently, DMSALA maintains the same dominant asymptotic time complexity as the original ALA.

### *3.5 Algorithm Pseudocode*

Algorithm 1 outlines the main framework of DMSALA. By introducing dynamic multi-subpopulation reconstruction, hybrid guidance, and adaptive perturbation, the proposed method improves the balance between exploration and exploitation. These strategies enhance population diversity and help the algorithm avoid premature convergence.

## 4 Construction of the IoT Intrusion Detection Framework

Fig. 2 illustrates the proposed IoT intrusion detection framework. To tackle the high dimensionality and redundancy of IoT traffic, we design a three-phase framework: data preprocessing, feature selection, and classification. The dataset is split into training and test sets at a 4:1 ratio. Data preprocessing is performed on the training dataset, while feature selection and model training are conducted exclusively on the training set; classification and evaluation are carried out on the held-out test set.

Raw IoT traffic is heterogeneous and noisy, causing modeling biases if used directly [29,30]. This phase standardizes the data by cleaning duplicates and missing values, applying Label Encoding to categorical data, and normalizing numerical features to eliminate dimensional discrepancies and provide a unified baseline. To prevent data leakage, all preprocessing is fitted only on the training dataset.

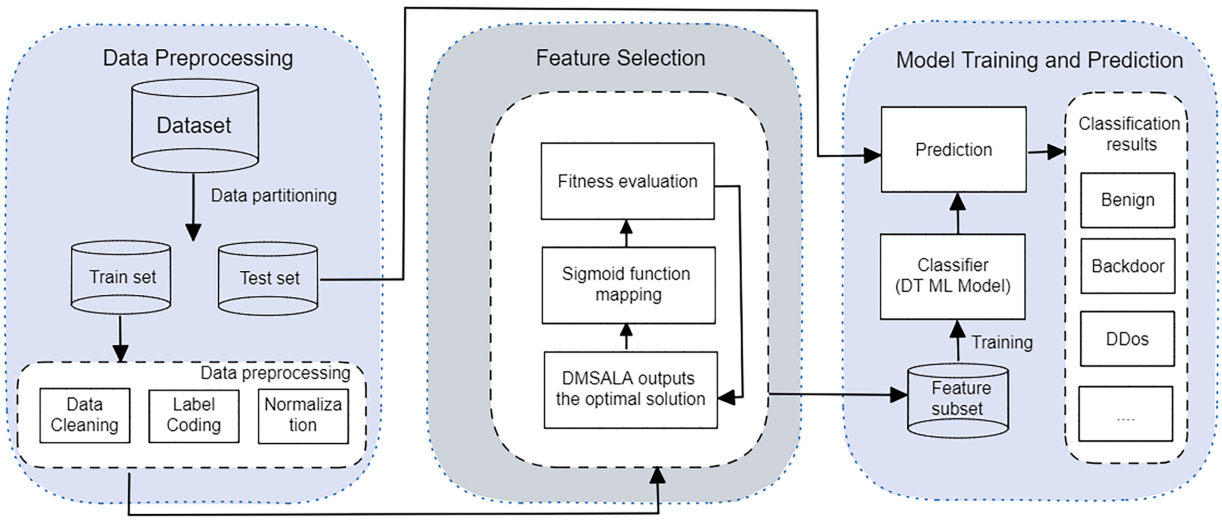



**Figure 2:** Overall workflow of the proposed IoT intrusion detection framework based on DMSALA.

**Algorithm 1:** Dynamic multi-subpopulation artificial lemming algorithm (DMSALA)

**Input:** Maximum iteration number $T_{\max}$, population size $N$, and dimension size $Dim$
**Output:** Global optimal solution $G^*(t)$ and its fitness value $f(G^*(t))$

Randomly initialize population positions and evaluate fitness;
Initialize subpopulations and compute centroids $c_k(t)$ and local optima $L_k(t)$ using Eq. (4);
$t \leftarrow 1$;
**while** $t \leq T_{\max}$ **do**
  Execute subpopulation merge/split reconstruction based on Eq. (5);
  Update $c_k(t)$, $L_k(t)$, and $G^*(t)$ using Eq. (4);
  **for each** search individual $X_i(t)$ **do**
    Construct hybrid guidance vector $B_i(t)$ using Eqs. (6) and (7), replacing $Z_{\text{best}}(t)$ with $B_i(t)$;
    Calculate energy factor $E(t)$ using Eq. (1);
    **if** $E(t) > 1$ **then**
      Update position using exploration equations (Eq. (2));
    **else**
      **if** $rand < 0.6$ **then**
        Update position using adaptive Gaussian perturbation (Eqs. (8)–(10));
      **else**
        Update position using exploitation equations (Eq. (3));
      **end if**
    **end if**
  **end for**
  Re-evaluate the fitness of all individuals;
  Update $c_k(t)$, $L_k(t)$, and $G^*(t)$ using Eq. (4);
  $t \leftarrow t + 1$;
**end while**

Redundant IoT features increase computational overhead and blur classification boundaries [31]. To address this, we utilize DMSALA as a wrapper-based feature selector. It employs a Sigmoid function to binarize continuous positions and optimizes a fitness function balancing classification error and feature count. DMSALA's dynamic multi-subpopulation structure effectively compresses the feature space, outputting a low-redundancy, highly discriminative subset.

The original dataset is projected onto the selected feature subspace to train the classifier. Eliminating redundant noise enhances the signal-to-noise ratio, enabling the classifier to focus on the essential distribution differences between normal traffic and various attacks (e.g., DDoS, Backdoor). The finalized classifier can efficiently categorize unknown traffic and detect complex IoT attacks. Both phases adopt the same base classifier, a Decision Tree (DT) with Gini impurity, unlimited depth, and the default minimum split of 2 samples per node and 1 sample per leaf.

## 5 Experiments

### *5.1 Experimental Setup*

For a fair comparison, all algorithms were implemented in Python and evaluated on a unified platform: Intel Core i5-12600KF (3.70 GHz), 32 GB RAM, Windows 11 (64-bit). In addition, all algorithms were run 30 times using the same randomly selected seeds.

Feature subset quality depends on the classification error rate and subset size. Thus, the fitness function combines them via a weighted sum:

$$\text{fitness} = \alpha \cdot \text{error} + \beta \cdot \frac{|N_f|}{D}, \tag{12}$$

where error denotes the classification error rate, $|N_f|$ is the selected feature count, $D$ is the original dimension, and the weights are set to $\alpha = 0.99$ and $\beta = 0.01$. This weighting scheme prioritizes classification performance while retaining a small penalty on subset size, thereby encouraging the algorithm to obtain an accurate yet compact feature subset [18–20].

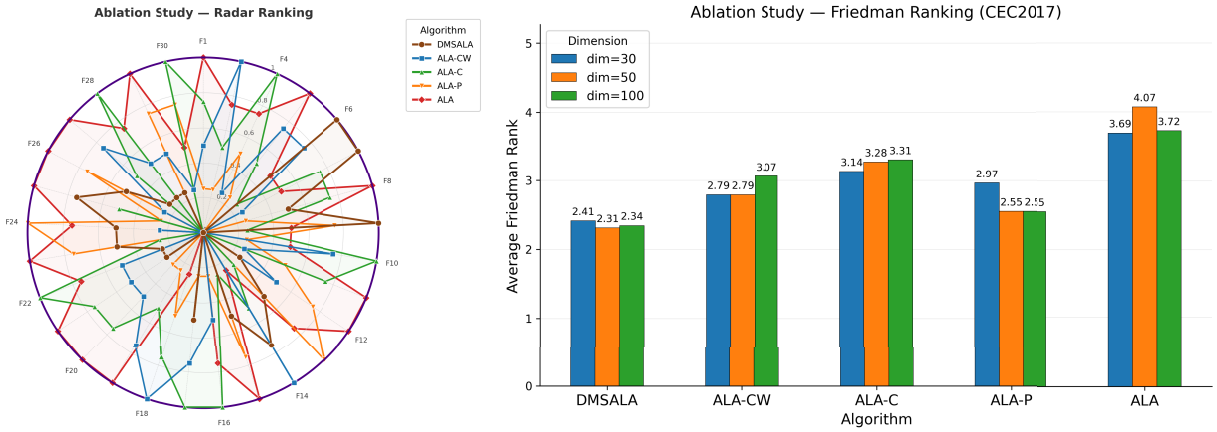
To evaluate the classification performance comprehensively, four standard evaluation metrics are employed in our experiments: Accuracy, Precision, Recall, and $F_1$-score.

### *5.2 Ablation Study on Improvement Strategies*

Ablation experiments on CEC2017 compare three variants: ALA-C (linear weighting), ALA-CW (cosine weighting, Eq. (6)), and ALA-P (Gaussian perturbation). As shown in Fig. 3, DMSALA outperforms all variants, and ALA-CW consistently surpasses ALA-C, confirming that all strategies contribute positively and their combination is complementary.

### *5.3 Experimental Results and Analysis on CEC2017 Test Functions*

DMSALA is evaluated on the CEC2017 benchmark suite against fungal growth optimizer (FGO) [32], multi-subswarm cooperative particle swarm optimization (MCS-PSO) [33], projection-iterative-methods-based optimizer (PIMO) [34], grey wolf optimizer (GWO) [35], and ALA. All algorithms use 500 iterations, a population size of 30, and 30 independent runs [20,21]. As Table 1 shows, at $D = 50$, it still obtains better Ave with low Std on several difficult functions.




(**a**) The radar chart. (**b**) The average rank chart.

**Figure 3:** Friedman rank and radar chart comparison of different DMSALA strategies on CEC2017 functions.

**Table 1:** Comparison of results on CEC2017 benchmark functions (D = 50).

| ID | | ALA | MCS-PSO | FGO | GWO | PIMO | DMSALA |
|---|---|---|---|---|---|---|---|
| F1 | Ave | 2.85E+10 | 6.08E+09 | 3.17E+10 | 9.91E+10 | 5.43E+10 | **4.14E+08** |
| | Std | 2.27E+10 | 1.34E+10 | 8.50E+09 | 4.12E+10 | 1.63E+10 | **9.69E+07** |
| F3 | Ave | 1.63E+05 | 1.60E+05 | 1.92E+05 | 1.57E+05 | 1.72E+05 | **8.37E+04** |
| | Std | 4.76E+04 | 2.83E+04 | 3.12E+04 | 2.79E+04 | 1.93E+04 | **1.68E+04** |
| F4 | Ave | 1.10E+03 | 7.46E+02 | 9.95E+02 | 1.34E+03 | 1.03E+03 | **6.39E+02** |
| | Std | 3.44E+02 | 4.33E+02 | 3.64E+02 | 5.91E+02 | 1.95E+02 | **5.23E+01** |
| F5 | Ave | 9.95E+02 | 7.57E+02 | 8.73E+02 | 7.77E+02 | 9.42E+02 | **7.17E+02** |
| | Std | 9.58E+01 | 1.03E+02 | 2.56E+01 | 8.11E+01 | 3.93E+01 | **2.37E+01** |
| F6 | Ave | 7.61E+02 | **6.34E+02** | 7.09E+02 | 7.15E+02 | 6.80E+02 | 7.01E+02 |
| | Std | 2.17E+01 | **7.40E+00** | 1.26E+01 | 1.47E+01 | 9.81E+00 | 1.38E+01 |
| F7 | Ave | 1.37E+03 | 1.08E+03 | 1.02E+03 | 9.97E+02 | 1.64E+03 | **9.60E+02** |
| | Std | 1.11E+02 | 9.94E+01 | **2.57E+01** | 5.60E+01 | 1.02E+02 | 3.37E+01 |
| F8 | Ave | 1.10E+03 | **1.04E+03** | 1.07E+03 | **1.04E+03** | 1.27E+03 | 1.05E+03 |
| | Std | 5.11E+01 | 9.29E+01 | 2.04E+01 | 4.43E+01 | 4.32E+01 | **1.84E+01** |
| F9 | Ave | 1.31E+04 | **3.10E+03** | 5.57E+03 | 5.69E+03 | 2.00E+04 | 3.96E+03 |
| | Std | 1.20E+03 | **9.48E+02** | 2.73E+03 | 2.00E+03 | 4.24E+03 | 1.18E+03 |
| F10 | Ave | 1.02E+04 | 1.25E+04 | 1.30E+04 | 1.12E+04 | 1.27E+04 | **7.44E+03** |
| | Std | 1.04E+03 | 2.55E+03 | 4.84E+02 | 3.49E+03 | 7.36E+02 | **4.05E+02** |
| F11 | Ave | 2.26E+03 | 1.65E+03 | 1.91E+03 | 3.97E+03 | 2.67E+03 | **1.59E+03** |
| | Std | 4.29E+02 | 2.40E+02 | **7.61E+01** | 1.49E+03 | 3.55E+02 | 9.59E+01 |

(Continued)

**Table 1 (continued)**

| ID | | ALA | MCS-PSO | FGO | GWO | PIMO | DMSALA |
|---|---|---|---|---|---|---|---|
| F12 | Ave | 3.69E+07 | 1.36E+09 | 2.24E+08 | 5.35E+09 | 8.23E+08 | **3.16E+07** |
| | Std | **1.89E+07** | 3.02E+09 | 8.26E+07 | 7.49E+09 | 5.18E+08 | 2.32E+07 |
| F13 | Ave | 3.19E+06 | 5.43E+08 | 4.68E+07 | 4.78E+09 | 1.85E+06 | **3.12E+05** |
| | Std | 5.30E+06 | 2.33E+09 | 3.75E+07 | 4.39E+09 | 1.16E+06 | **1.29E+05** |
| F14 | Ave | 1.72E+05 | 5.95E+05 | 5.53E+04 | 9.15E+06 | 2.11E+05 | **1.66E+04** |
| | Std | 1.37E+05 | 5.18E+05 | 2.98E+04 | 5.70E+06 | 1.04E+05 | **1.07E+04** |
| F15 | Ave | 1.45E+05 | **2.74E+04** | 3.87E+07 | 9.04E+10 | 2.68E+05 | 2.91E+04 |
| | Std | 2.21E+05 | 8.24E+04 | 3.57E+07 | 1.19E+11 | 2.63E+05 | **1.39E+04** |
| F16 | Ave | 4.16E+03 | 3.76E+03 | 4.51E+03 | 3.73E+03 | 4.38E+03 | **3.44E+03** |
| | Std | 5.48E+02 | 6.98E+02 | **1.91E+02** | 6.26E+02 | 5.19E+02 | 2.41E+02 |
| F17 | Ave | 3.52E+03 | 3.20E+03 | 3.01E+03 | **2.73E+03** | 3.65E+03 | 2.94E+03 |
| | Std | 2.68E+02 | 4.40E+02 | **2.23E+02** | 3.39E+02 | 2.71E+02 | 2.77E+02 |
| F18 | Ave | 4.67E+06 | 3.51E+06 | 1.43E+06 | 7.19E+07 | 1.97E+06 | **1.22E+06** |
| | Std | 3.93E+06 | 2.44E+06 | 1.03E+06 | 5.70E+07 | 1.22E+06 | **6.67E+05** |
| F19 | Ave | 8.91E+04 | **2.08E+04** | 2.46E+08 | 4.23E+10 | 5.12E+06 | 4.47E+04 |
| | Std | 6.60E+04 | **1.15E+04** | 3.08E+08 | 5.40E+10 | 4.68E+06 | 3.46E+04 |
| F20 | Ave | 3.99E+03 | 3.58E+03 | 4.06E+03 | 3.35E+03 | **3.28E+03** | 3.43E+03 |
| | Std | 2.43E+02 | 4.46E+02 | **1.82E+02** | 3.21E+02 | 2.31E+02 | 2.16E+02 |
| F21 | Ave | 3.32E+03 | **2.54E+03** | 3.20E+03 | 3.18E+03 | 2.77E+03 | 3.15E+03 |
| | Std | 5.71E+01 | 8.53E+01 | 1.95E+01 | 3.42E+01 | 5.29E+01 | **1.79E+01** |
| F22 | Ave | 1.75E+04 | 7.72E+03 | 1.87E+04 | **1.44E+04** | **1.44E+04** | 1.58E+04 |
| | Std | 1.83E+03 | 5.22E+03 | 1.20E+03 | 2.59E+03 | **7.41E+02** | 8.22E+02 |
| F23 | Ave | 3.98E+03 | **3.12E+03** | 3.55E+03 | 3.45E+03 | 3.30E+03 | 3.34E+03 |
| | Std | 1.40E+02 | 1.25E+02 | 4.38E+01 | 1.03E+02 | 7.72E+01 | **2.92E+01** |
| F24 | Ave | 4.15E+03 | **3.33E+03** | 3.88E+03 | 3.76E+03 | 3.40E+03 | 3.62E+03 |
| | Std | 1.59E+02 | 1.49E+02 | 5.43E+01 | 1.61E+02 | 8.50E+01 | **4.82E+01** |
| F25 | Ave | 3.64E+03 | **3.13E+03** | 3.62E+03 | 3.66E+03 | 3.53E+03 | 3.52E+03 |
| | Std | 2.87E+01 | 4.10E+01 | 3.01E+01 | 3.48E+01 | 3.03E+02 | **2.73E+01** |
| F26 | Ave | 2.00E+04 | **5.68E+03** | 1.68E+04 | 1.53E+04 | 9.69E+03 | 1.60E+04 |
| | Std | 2.32E+03 | 1.55E+03 | **5.27E+02** | 1.55E+03 | 8.11E+02 | 6.04E+02 |
| F27 | Ave | 3.67E+03 | 3.63E+03 | 3.63E+03 | 3.64E+03 | 3.64E+03 | **3.60E+03** |
| | Std | 3.51E+01 | 1.77E+02 | 1.24E+01 | 2.62E+01 | 2.47E+02 | **5.87E+00** |
| F28 | Ave | 1.31E+04 | **3.58E+03** | 1.19E+04 | 1.18E+04 | 4.90E+03 | 1.15E+04 |
| | Std | 1.76E+03 | **2.46E+02** | 7.21E+02 | 1.56E+03 | 1.98E+03 | 3.57E+02 |

(Continued)

**Table 1 (continued)**

| ID | | ALA | MCS-PSO | FGO | GWO | PIMO | DMSALA |
|---|---|---|---|---|---|---|---|
| F29 | Ave | 8.25E+03 | **5.03E+03** | 7.92E+03 | 7.28E+03 | 5.79E+03 | 7.65E+03 |
| | Std | 4.72E+02 | 4.73E+02 | 2.69E+02 | 4.41E+02 | 5.08E+02 | **1.32E+02** |
| F30 | Ave | 1.38E+09 | **8.21E+06** | 1.69E+08 | 3.68E+11 | 8.02E+07 | 2.25E+07 |
| | Std | 1.72E+09 | 1.26E+07 | 5.52E+07 | 3.89E+11 | 5.13E+07 | **1.02E+07** |

Note: Bold values indicate the best results.

To illustrate the optimization performance of DMSALA on CEC2017, Figs. 4 and 5 present the convergence curves and boxplots of six representative functions ($F_1$, $F_3$, $F_{12}$, $F_{19}$, $F_{24}$, and $F_{28}$) at $D = 50$ over 30 independent runs. DMSALA converges faster and more steadily than the compared algorithms, with rapid early descent indicating efficient exploration and sustained late-stage improvement reflecting strong exploitation. The boxplots further confirm that DMSALA achieves the lowest median fitness with compact interquartile ranges, demonstrating superior convergence accuracy and stability.

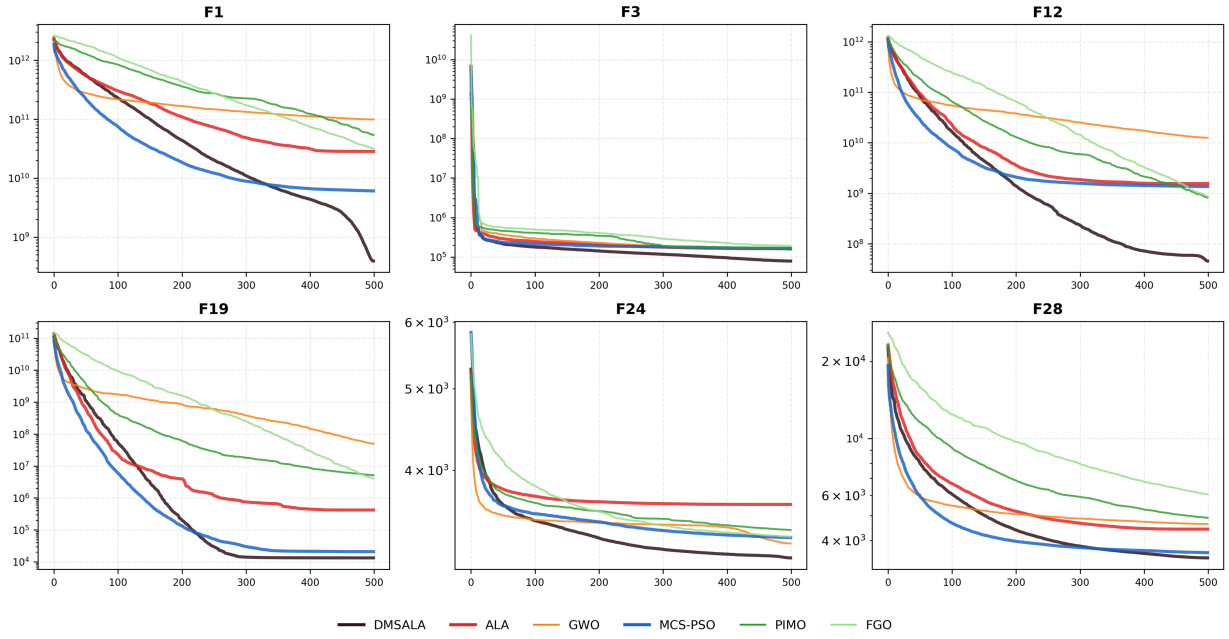



**Figure 4:** Convergence curves of DMSALA and other algorithms on six representative CEC2017 functions ($D = 50$).

Fig. 6 presents the radar chart and average-rank bar chart based on Friedman tests for CEC2017. A smaller radius in Fig. 6a indicates a better rank, and DMSALA shows smaller and more stable radii on most functions. Fig. 6b further confirms that DMSALA achieves the best average rank across dimensions, especially in higher dimensions. Overall, these results demonstrate the strong optimization capability and robustness of DMSALA on CEC2017.

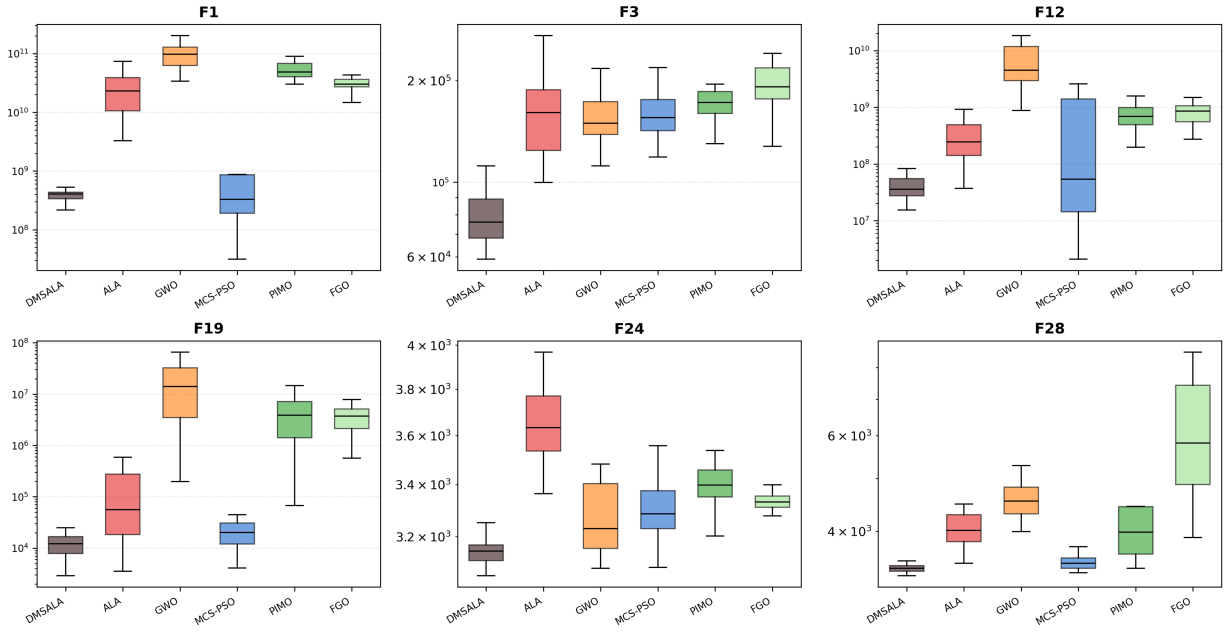
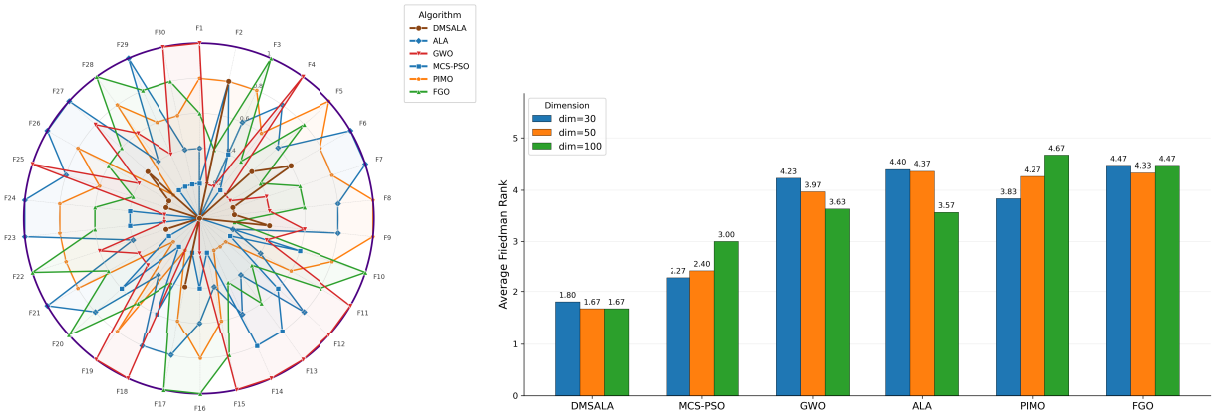



**Figure 5:** Boxplots of DMSALA and other algorithms on six representative CEC2017 functions ($D$ = 50).




(**a**) The radar chart. (**b**) The average rank chart.

**Figure 6:** Friedman rank and radar chart comparison of DMSALA and other algorithms on CEC2017 functions.

### *5.4 Parameter Sensitivity Analysis*

As shown in Table 2, the default setting $d_{\text{merge}} = 0.05$ and $r_{\text{split}} = 0.15$ achieves the lowest Mean Rank of 3.1, indicating the best performance among all tested combinations.

### *5.5 Comparison with Other Metaheuristic Optimization Algorithms*

The NF-ToN-IoT-v2 dataset is derived from the ToN-IoT testbed and transforms raw Packet Capture (PCAP) traffic into NetFlow records more suitable for high-throughput edge networks. It provides a unified extended feature set with 43 NetFlow features and corresponding intrusion labels,and the NF-BoT-IoT-v2 dataset is derived from the BoT-IoT dataset. Both transform raw PCAP traffic into NetFlow records, improving reproducibility [36].

**Table 2:** Sensitivity analysis of the merge and split thresholds.

| Metric | $d_{merge} = 0.02$, $r_{split} = 0.05$ | $d_{merge} = 0.02$, $r_{split} = 0.15$ | $d_{merge} = 0.02$, $r_{split} = 0.30$ | $d_{merge} = 0.05$, $r_{split} = 0.05$ | $d_{merge} = 0.05$, $r_{split} = 0.15$ | $d_{merge} = 0.05$, $r_{split} = 0.30$ | $d_{merge} = 0.10$, $r_{split} = 0.05$ | $r_{split} = 0.15$ $r_{split} = 0.15$ | $d_{merge} = 0.10$, $r_{split} = 0.30$ |
|---|---|---|---|---|---|---|---|---|---|
| Mean Rank ↓ | 4.4 | 5.9 | 6.1 | 4.8 | **3.1** | 6.1 | 4.5 | 4.6 | 5.5 |

Note: ↓ indicates that lower values are better. Bold values indicate the best results.

Tables 3 and 4 present the binary classification results on the NF-ToN-IoT-v2 and NF-BoT-IoT-v2 datasets, respectively. DMSALA achieves the highest accuracy, precision, recall, and F1-score on both datasets, with the fewest selected features and lowest evaluation time. The Receiver Operating Characteristic (ROC) curves in Fig. 7 further confirm its superior discriminative ability.

**Table 3:** Metrics summary for binary classification on the NF-ToN-IoT-v2 dataset.

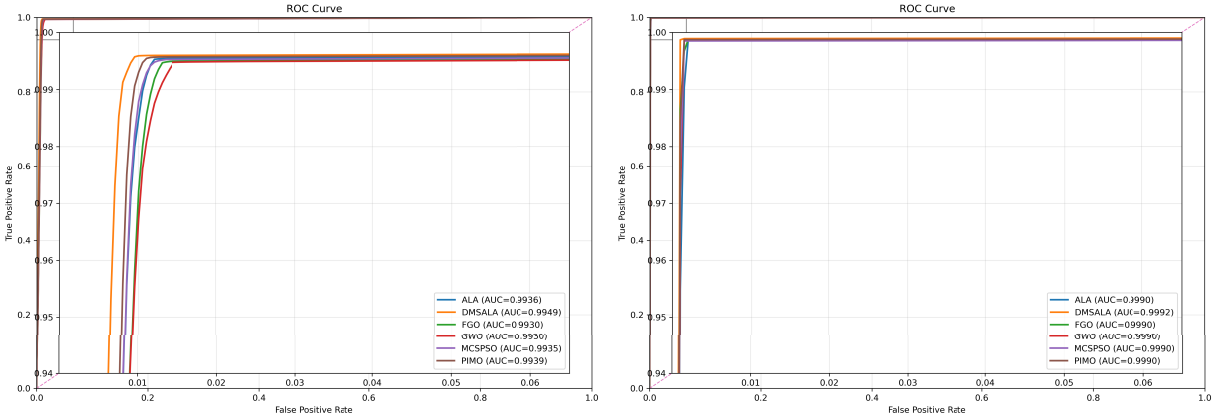
| Algorithm | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) | Features | Train Time (s) | Eval Time (s) |
|---|---|---|---|---|---|---|---|
| ALA | 99.303 | 99.395 | 99.517 | 99.456 | 9.17 | 0.084 | 0.006 |
| MCS-PSO | 99.288 | 99.384 | 99.504 | 99.444 | 10.38 | 0.119 | 0.009 |
| DMSALA | **99.410** | **99.505** | **99.573** | **99.539** | **8.30** | **0.076** | **0.005** |
| FGO | 99.254 | 99.378 | 99.457 | 99.417 | 10.67 | 0.086 | 0.007 |
| GWO | 99.228 | 99.353 | 99.441 | 99.397 | 8.97 | 0.078 | 0.006 |
| PIMO | 99.329 | 99.405 | 99.547 | 99.476 | 11.96 | 0.122 | 0.008 |

Note: Bold values indicate the best results.

**Table 4:** Metrics summary for binary classification on the NF-BoT-IoT-v2 dataset.

| Algorithm | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) | Features | Train Time (s) | Eval Time (s) |
|---|---|---|---|---|---|---|---|
| ALA | 99.896 | 99.921 | 99.871 | 99.896 | 5.90 | 0.077 | 0.005 |
| MCS-PSO | 99.894 | 99.926 | 99.861 | 99.894 | 5.94 | 0.068 | 0.005 |
| DMSALA | **99.912** | **99.934** | **99.889** | **99.912** | **5.50** | **0.066** | **0.004** |
| FGO | 99.898 | 99.925 | 99.870 | 99.898 | 7.57 | 0.068 | 0.005 |
| GWO | 99.895 | 99.920 | 99.870 | 99.895 | 6.34 | 0.076 | 0.005 |
| PIMO | 99.898 | 99.924 | 99.872 | 99.898 | 7.22 | 0.077 | 0.005 |

Note: Bold values indicate the best results.




(**a**) ROC curve on NF-ToN-IoT-v2 (**b**) ROC curve on NF-BoT-IoT-v2

**Figure 7:** ROC curves comparison of DMSALA and other algorithms.

As shown in Table 5, DMSALA achieves strong multiclass detection performance on NF-ToN-IoT-v2, with high Recall, Precision, and F1-score on most attack classes. It performs especially well on DDoS, DoS, MITM, and XSS, and attains the highest overall Accuracy of 97.235% among all methods. Moreover, DMSALA selects only 7.40 features on average, the fewest among all methods, and achieves the lowest Eval time of 0.006 s. Overall, it maintains a good balance among detection accuracy, feature reduction, and inference efficiency.

**Table 5:** Metrics summary for multiclass classification on the NF-ToN-IoT-v2 dataset.

| Metric | Algo | Backdoor | Benign | DDoS | DoS | Injection | MITM | Password | Ransomware | Scanning | XSS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Recall (%) | ALA | 99.825 | 98.891 | 97.721 | 88.139 | 83.849 | 85.566 | 94.447 | 99.727 | **98.722** | 97.532 |
| | MCS-PSO | **99.882** | 98.901 | 97.761 | 88.215 | 84.296 | 85.728 | 94.600 | 99.613 | 98.692 | 97.713 |
| | DMSALA | 99.869 | **98.982** | **97.936** | **88.546** | **85.253** | **87.632** | **95.460** | **99.752** | 98.638 | **97.772** |
| | FGO | 99.861 | 98.925 | 97.763 | 88.347 | 85.056 | 86.000 | 94.733 | 99.620 | 98.623 | 97.534 |
| | GWO | 99.837 | 98.817 | 97.701 | 87.911 | 84.145 | 85.621 | 94.623 | 99.630 | 98.649 | 97.663 |
| | PIMO | 99.859 | 98.926 | 97.774 | 88.398 | 84.578 | 85.746 | 94.489 | 99.668 | 98.642 | 97.707 |
| Precision (%) | ALA | 99.893 | **99.003** | 97.638 | 81.822 | 87.966 | 89.640 | 93.493 | 99.015 | 99.138 | 96.347 |
| | MCS-PSO | 99.809 | 98.997 | 97.692 | 81.911 | 88.551 | 89.812 | 93.775 | **99.371** | 99.097 | 96.433 |
| | DMSALA | **99.917** | **99.003** | **98.359** | **82.421** | **89.666** | **90.179** | **94.183** | 99.188 | 99.064 | **97.054** |
| | FGO | 99.877 | 98.931 | 97.757 | 81.992 | 88.429 | 89.867 | 94.049 | 99.119 | **99.215** | 96.459 |
| | GWO | 99.770 | 98.983 | 97.566 | 81.994 | 88.677 | 89.525 | 93.674 | 99.273 | 98.925 | 96.403 |
| | PIMO | 99.855 | 98.967 | 97.603 | 81.871 | 88.428 | 89.905 | 94.122 | 99.206 | 99.158 | 96.432 |
| F1-score (%) | ALA | 99.859 | 98.947 | 97.679 | 84.862 | 85.856 | 87.554 | 93.966 | 99.370 | **98.929** | 96.935 |
| | MCS-PSO | 99.845 | 98.949 | 97.726 | 84.944 | 86.368 | 87.721 | 94.184 | **99.492** | 98.894 | 97.069 |
| | DMSALA | **99.893** | **98.993** | **98.147** | **85.373** | **87.402** | **88.887** | **94.816** | 99.469 | 98.850 | **97.412** |
| | FGO | 99.869 | 98.928 | 97.760 | 85.049 | 86.705 | 87.889 | 94.388 | 99.369 | 98.918 | 96.993 |
| | GWO | 99.804 | 98.900 | 97.633 | 84.848 | 86.349 | 87.529 | 94.145 | 99.451 | 98.787 | 97.029 |
| | PIMO | 99.857 | 98.946 | 97.688 | 85.007 | 86.453 | 87.775 | 94.303 | 99.436 | 98.899 | 97.065 |
| Accuracy (%) | ALA | 96.95 | | Selected features | 7.70 | Train time (s) | 0.084 | Eval time (s) | 0.008 | | |
| | MCS-PSO | 97.011 | | | 8.80 | | 0.090 | | 0.009 | | |
| | DMSALA | **97.235** | | | **7.40** | | **0.076** | | **0.006** | | |
| | FGO | 97.033 | | | 10.20 | | 0.086 | | 0.008 | | |
| | GWO | 96.943 | | | 8.80 | | 0.078 | | 0.009 | | |
| | PIMO | 97.020 | | | 9.23 | | 0.089 | | 0.009 | | |

Note: Bold values indicate the best results.

On NF-BoT-IoT-v2 (Table 6), DMSALA also achieves the highest overall Accuracy of 99.155% and the best F1-score on Reconn, DDoS, and DoS classes, while selecting only 8.35 features with competitive training and evaluation times, further confirming its robustness across IoT environments.

**Table 6:** Metrics summary for multiclass classification on the NF-BoT-IoT-v2 dataset.

| Metric | Algo | Benign | Reconn | DDoS | DoS | Theft |
|---|---|---|---|---|---|---|
| Recall (%) | ALA | 99.788 | 98.881 | 99.350 | 98.046 | 99.771 |
| | DMSALA | 99.816 | **99.075** | 99.412 | **98.260** | 99.701 |
| | FGO | 99.774 | 98.908 | 99.343 | 98.104 | 99.665 |
| | GWO | 99.787 | 98.809 | 99.444 | 97.902 | **99.832** |
| | MCS-PSO | 99.781 | 98.923 | 99.395 | 98.066 | 99.706 |
| | PIMO | **99.819** | 98.822 | **99.447** | 97.882 | 99.665 |

(Continued)

**Table 6 (continued)**

| Metric | Algo | Benign | Reconn | DDoS | DoS | Theft |
|---|---|---|---|---|---|---|
| Precision (%) | ALA | 99.759 | 98.278 | 99.565 | 98.531 | **99.147** |
| | DMSALA | **99.830** | **98.395** | **99.692** | **98.714** | 99.079 |
| | FGO | 99.797 | 98.297 | 99.594 | 98.516 | 98.971 |
| | GWO | 99.818 | 98.314 | 99.387 | 98.543 | 98.660 |
| | MCS-PSO | 99.798 | 98.349 | 99.522 | 98.576 | 98.964 |
| | PIMO | 99.803 | 98.294 | 99.416 | 98.519 | 99.062 |
| F1-score (%) | ALA | 99.773 | 98.578 | 99.457 | 98.287 | **99.458** |
| | DMSALA | **99.823** | **98.734** | **99.552** | **98.486** | 99.389 |
| | FGO | 99.785 | 98.601 | 99.468 | 98.309 | 99.317 |
| | GWO | 99.803 | 98.561 | 99.416 | 98.221 | 99.243 |
| | MCS-PSO | 99.789 | 98.635 | 99.458 | 98.320 | 99.333 |
| | PIMO | 99.811 | 98.557 | 99.432 | 98.199 | 99.362 |

| Accuracy (%) | | | Selected features | | Train time (s) | | Eval time (s) | |
|---|---|---|---|---|---|---|---|---|
| | ALA | 99.034 | | 8.87 | | 0.117 | | 0.011 |
| | DMSALA | **99.155** | | **8.35** | | 0.107 | | **0.007** |
| | FGO | 99.047 | | 9.44 | | 0.074 | | 0.008 |
| | GWO | 99.006 | | 9.00 | | **0.067** | | 0.008 |
| | MCS-PSO | 99.057 | | 9.24 | | 0.133 | | 0.011 |
| | PIMO | 99.009 | | 10.44 | | 0.087 | | 0.010 |

Note: Bold values indicate the best results.

The Friedman average ranking results, calculated using Accuracy, Precision, Recall, and F1-score, are shown in Table 7, where DMSALA achieves the best overall rank of 1.63 among all compared algorithms. Excluding selected features, training time, and evaluation time, the Friedman test confirms that the performance differences are statistically significant, with $p = 4.16 \times 10^{-20}$.

**Table 7:** Friedman average ranking of all algorithms.

| | DMSALA | ALA | FGO | GWO | MCS-PSO | PIMO |
|---|---|---|---|---|---|---|
| Overall rank | **1.63** | 3.67 | 3.91 | 4.47 | 3.70 | 3.63 |

Note: Bold values indicate the best results.

### *5.6 Comparison with Recent IoT Intrusion Detection Methods*

To further validate the effectiveness of DMSALA, it is compared with four recent IoT intrusion detection methods. EGWO-RF enhances standard GWO with an omega wolf for binary feature selection and employs Random Forest as the classifier. SDFC is a two-layer feature selection framework combining statistical filters with Support Vector Machine (SVM)-RFE and Particle Swarm Optimization (PSO), followed by a two-stage LightGBM–eXtreme Gradient Boosting (XGBoost) classifier. DTXGRF uses a soft-voting ensemble of Decision Tree, XGBoost, and Random Forest with Mutual Information (MI)-based feature ranking. CorChi selects features via the union of Chi-square and Pearson correlation and trains a lightweight Decision Tree.

Tables 8–11 present the experimental results. On the binary NF-ToN-IoT-v2 task, DMSALA leads across all metrics with the fewest features. On the multiclass NF-ToN-IoT-v2 task, DMSALA outperforms all

methods in accuracy, precision, recall, and F1-score; SDFC selects marginally fewer features but suffers from notably low classification performance. On NF-BoT-IoT-v2, DTXGRF achieves slightly higher accuracy than DMSALA in both binary and multiclass tasks, yet requires approximately three times as many features with substantially longer training and evaluation times. Overall, DMSALA offers the most favorable trade-off among detection accuracy, feature reduction, and computational efficiency.

**Table 8:** Comparison with other IoT IDS methods on the NF-ToN-IoT-v2 dataset (binary).

| Method | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) | Features | Train Time (s) | Eval Time (s) |
|---|---|---|---|---|---|---|---|
| EGWO-RF | 98.32 | 99.11 | 98.27 | 98.68 | 11.8 | 0.417 | 0.034 |
| SDFC | 97.43 | 98.82 | 97.14 | 97.97 | **5.2** | 0.148 | 0.012 |
| DTXGRF | 98.79 | 99.27 | 98.84 | 99.05 | 21.8 | 1.921 | 0.064 |
| CorChi | 95.96 | 99.21 | 94.44 | 96.76 | 17.0 | 0.134 | 0.009 |
| DMSALA | **99.41** | **99.51** | **99.57** | **99.54** | 8.30 | **0.076** | **0.005** |

Note: Bold values indicate the best results.

**Table 9:** Comparison with other IoT IDS methods on the NF-BoT-IoT-v2 dataset (binary).

| Method | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) | Features | Train Time (s) | Eval Time (s) |
|---|---|---|---|---|---|---|---|
| EGWO-RF | 99.80 | 99.91 | 99.70 | 99.80 | 9.6 | 0.315 | 0.032 |
| SDFC | 99.67 | 99.87 | 99.46 | 99.67 | **5.4** | 0.214 | 0.012 |
| DTXGRF | **99.96** | **99.98** | **99.95** | **99.96** | 26.0 | 4.360 | 0.131 |
| CorChi | 99.92 | 99.94 | 99.89 | 99.92 | 19.0 | 0.123 | 0.014 |
| DMSALA | 99.91 | 99.93 | 99.89 | 99.91 | 5.50 | **0.066** | **0.004** |

Note: Bold values indicate the best results.

**Table 10:** Per-class comparison of other IoT IDS methods on the NF-ToN-IoT-v2 dataset (multiclass).

| Metric | Method | Backdoor | Benign | DDoS | DoS | Injection | MITM | Password | Ransomware | Scanning | XSS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Recall (%) | EGWO-RF | 99.57 | 97.80 | 94.96 | **97.74** | 82.25 | 75.63 | 90.47 | 99.91 | 94.91 | 96.78 |
| | SDFC | 99.25 | 91.07 | 74.64 | 13.75 | 2.82 | 22.12 | 12.21 | 80.12 | 85.28 | 87.75 |
| | DTXGRF | **99.90** | 98.88 | **98.02** | 86.46 | **86.00** | 85.93 | 95.33 | **99.88** | 96.37 | **98.42** |
| | CorChi | 99.82 | 98.82 | 97.48 | 80.49 | 80.26 | 86.36 | 91.83 | 99.74 | **98.75** | 95.92 |
| | DMSALA | 99.87 | **98.98** | 97.94 | 88.55 | 85.25 | **87.63** | **95.46** | 99.75 | 98.64 | 97.77 |
| Precision (%) | EGWO-RF | **99.98** | 97.17 | 97.14 | 73.75 | 74.97 | **94.05** | 90.86 | **99.45** | 98.02 | 95.34 |
| | SDFC | 99.76 | 91.23 | 77.07 | **99.16** | 64.65 | 81.69 | 66.39 | 86.57 | 98.06 | 38.77 |
| | DTXGRF | 99.95 | 97.77 | **98.53** | 80.56 | **91.56** | 88.97 | **95.32** | 99.74 | 98.90 | 95.93 |
| | CorChi | 99.83 | 98.81 | 97.41 | 81.43 | 81.03 | 85.69 | 91.72 | **99.45** | 98.93 | 95.62 |
| | DMSALA | 99.92 | **99.00** | 98.36 | 82.42 | 89.67 | 90.18 | 94.18 | 99.19 | **99.06** | **97.05** |
| F1-score (%) | EGWO-RF | 99.77 | 97.48 | 96.02 | 84.04 | 78.40 | 83.81 | 90.66 | 99.68 | 96.43 | 96.05 |
| | SDFC | 99.50 | 91.12 | 71.55 | 24.12 | 5.22 | 31.03 | 19.74 | 79.35 | 90.95 | 52.78 |
| | DTXGRF | **99.93** | 98.32 | **98.27** | 83.40 | **88.68** | 87.41 | **95.32** | **99.81** | 97.62 | 97.16 |
| | CorChi | 99.83 | 98.82 | 97.44 | 80.96 | 80.63 | 86.03 | 91.78 | 99.59 | 98.84 | 95.77 |
| | DMSALA | 99.89 | **98.99** | 98.15 | **85.37** | 87.40 | **88.89** | 94.82 | 99.47 | **98.85** | **97.41** |
| Accuracy (%) | EGWO-RF | 95.09 | | | 20.0 | | 0.449 | | 0.058 | | |
| | SDFC | 75.17 | | | **3.0** | | 1.162 | | 0.027 | | |
| | DTXGRF | 96.72 | | Selected | 25.0 | Traintime | 9.331 | Evaltime | 0.251 | | |
| | CorChi | 96.25 | | features | | (s) | 0.232 | (s) | 0.015 | | |
| | DMSALA | **97.24** | | | 7.40 | | **0.076** | | **0.006** | | |

Note: Bold values indicate the best results.

**Table 11:** Per-class comparison of other IoT IDS methods on the NF-BoT-IoT-v2 dataset (multiclass).

| Metric | Method | Benign | Reconn | DDoS | DoS | Theft |
|---|---|---|---|---|---|---|
| Recall (%) | EGWO-RF | 99.85 | 94.74 | 99.16 | 98.39 | **99.88** |
| | SDFC | 99.80 | 93.80 | 99.43 | 98.29 | 98.85 |
| | DTXGRF | **99.92** | **99.28** | **99.50** | **98.88** | **99.88** |
| | CorChi | 99.82 | 98.42 | 99.12 | 97.83 | 99.30 |
| | DMSALA | 99.82 | 99.08 | 99.41 | 98.26 | 99.70 |
| Precision (%) | EGWO-RF | 99.43 | 98.80 | 99.44 | 94.77 | 98.47 |
| | SDFC | 99.53 | 98.56 | 99.43 | 94.88 | 90.57 |
| | DTXGRF | **99.88** | **99.04** | **99.75** | **98.93** | **99.71** |
| | CorChi | 99.82 | 98.35 | 99.25 | 97.82 | 98.78 |
| | DMSALA | 99.83 | 98.40 | 99.69 | 98.71 | 99.08 |
| F1-score (%) | EGWO-RF | 99.64 | 96.72 | 99.30 | 96.55 | 99.17 |
| | SDFC | 99.66 | 96.11 | 99.43 | 96.55 | 94.52 |
| | DTXGRF | **99.90** | **99.16** | **99.62** | **98.90** | **99.79** |
| | CorChi | 99.82 | 98.39 | 99.18 | 97.83 | 99.04 |
| | DMSALA | 99.82 | 98.73 | 99.55 | 98.49 | 99.39 |
| Accuracy (%) | EGWO-RF | 98.08 | Selected features | 10.4 | Train time (s) 0.329 | Eval time (s) 0.044 |
| | SDFC | 97.86 | | **6.2** | 0.545 | 0.011 |
| | DTXGRF | **99.41** | | 26.0 | 9.976 | 0.250 |
| | CorChi | 98.81 | | 20.0 | 0.202 | 0.011 |
| | DMSALA | 99.16 | | 8.35 | **0.107** | **0.007** |

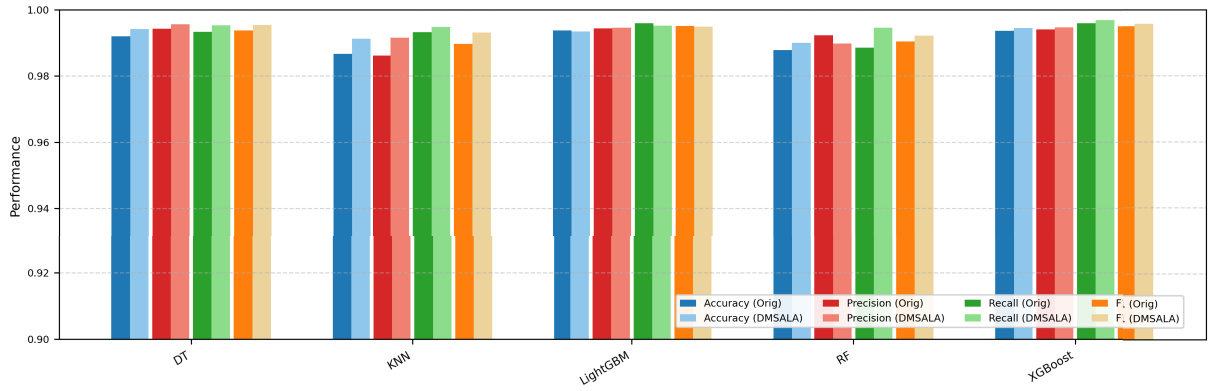
Note: Bold values indicate the best results.

### *5.7 Performance Evaluation of Feature Selection*

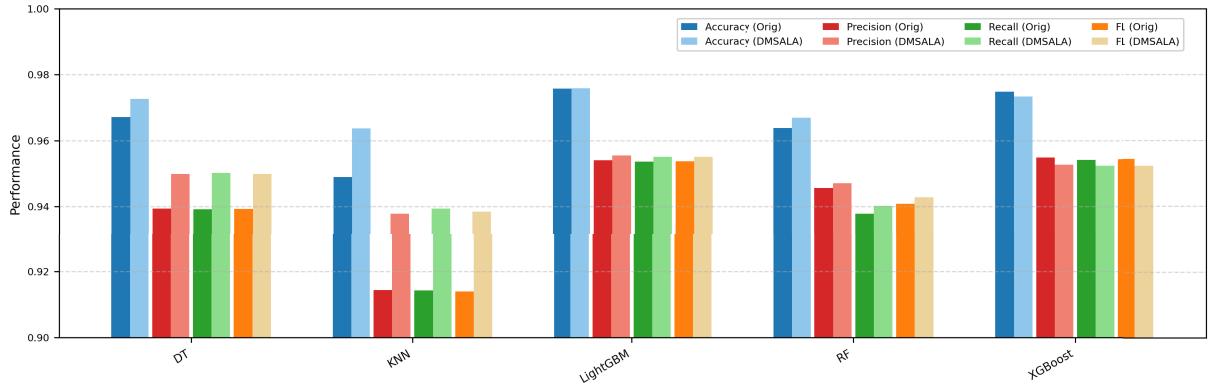
To evaluate DMSALA's feature selection capability, we compare the full feature set (Orig) and the subset selected by DMSALA across models regarding classification performance, time, and resource overhead. We first assess whether the subset maintains or improves detection capability after dimensionality reduction.

Fig. 8 shows that DMSALA achieves comparable or better performance on most models. In binary classification, DT, K-Nearest Neighbors (KNN), Random Forest (RF), and XGBoost yield slightly higher Accuracy, Precision, Recall, and F1-score, while LightGBM remains on par with the baseline. In multiclass classification, the DMSALA subset improves all four metrics for DT, KNN, RF, and LightGBM, with only XGBoost showing a marginal decline. Overall, DMSALA enhances detection performance on most models while maintaining high accuracy.

Table 12 shows that DMSALA generally reduces evaluation time across most models in both tasks. In binary classification, DT, KNN, RF, and XGBoost run faster with DMSALA, while LightGBM shows a slight increase. This stems from LightGBM's leaf-wise growth, which produces deeper trees with fewer features, while its exclusive feature bundling already alleviates high-dimensional overhead, reducing the marginal gain of feature reduction. In multiclass classification, all models achieve lower evaluation time. Overall, DMSALA reduces inference cost for most models without sacrificing performance.




(**a**) Binary classification results




(**b**) Multiclass classification results

**Figure 8:** Performance comparison of DMSALA against the baseline across different machine learning models.

**Table 12:** Evaluation-time comparison between Orig and DMSALA on binary and multiclass classification tasks.

| Task | DT | | KNN | | LightGBM | | RF | | XGBoost | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Orig** | **DMSALA** | **Orig** | **DMSALA** | **Orig** | **DMSALA** | **Orig** | **DMSALA** | **Orig** | **DMSALA** |
| Binary classification | 0.001492 | **0.001359** | 1.201111 | **0.979709** | **0.219735** | 0.223537 | 0.269170 | **0.266481** | 0.098969 | **0.089040** |
| Multiclass classification | 0.002065 | **0.001887** | 1.202051 | **0.986782** | 2.729501 | **2.696263** | 0.438563 | **0.425849** | 0.724098 | **0.660242** |

Note: Bold values indicate the best results.

Table 13 shows DMSALA generally reduces model storage costs. In binary tasks, all models shrink except RF, notably KNN. In multiclass tasks, only LightGBM slightly increases. Overall, DMSALA effectively compresses models while maintaining performance.

**Table 13:** Model size comparison of Orig vs. DMSALA in binary and multiclass tasks (MB).

| Method | DT | | KNN | | LightGBM | | RF | | XGBoost | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Orig | DMSALA | Orig | DMSALA | Orig | DMSALA | Orig | DMSALA | Orig | DMSALA |
| Binary model size (MB) | 0.105 | **0.102** | 13.735 | **2.748** | 0.994 | **0.991** | **10.337** | 10.987 | 0.989 | **0.976** |
| Multiclass model size (MB) | 0.653 | **0.454** | 13.732 | **3.968** | **9.804** | 9.861 | 40.135 | **34.529** | 6.821 | **6.136** |

Note: Bold values indicate the best results.

## 6 Conclusions

To reduce computational overhead arising from high-dimensional IoT intrusion detection data, we propose a lightweight feature selection and detection framework based on DMSALA. Experiments show that DMSALA achieves superior convergence and stability on the CEC2017 benchmark. On NF-ToN-IoT-v2 and NF-BoT-IoT-v2, it significantly reduces dimensionality and overhead while maintaining high classification performance, demonstrating its viability for resource-constrained edge deployment.

Future work includes: (1) online feature selection and concept drift adaptation for data streams; (2) cost-sensitive or multi-objective constraints for class imbalance; (3) although real IoT deployment validation is still lacking, deploying the trained feature subset and classifier on resource-constrained edge devices such as IoT gateways and Software-Defined Networking (SDN) controllers, where feature selection is performed offline on historical traffic while online detection runs directly on lightweight edge nodes to evaluate end-to-end latency and energy efficiency in real-world environments.

**Acknowledgement:** The authors would like to take this opportunity to express their sincere gratitude to all those who provided valuable suggestions and support for the completion of this manuscript. The authors also acknowledge the use of GPT-5.4 (OpenAI) for language polishing during the preparation of this manuscript.

**Funding Statement:** The authors received no specific funding for this study.

**Author Contributions:** Study conception and design: Hui Xu, Ruiqi Qu; data collection: Hui Xu, Ruiqi Qu, and Xinlu Zong; analysis and interpretation of results: Hui Xu, Ruiqi Qu; draft manuscript preparation: Hui Xu, Ruiqi Qu, and Xinlu Zong. All authors reviewed and approved the final version of the manuscript.

**Availability of Data and Materials:** The dataset used in this study is openly accessible and reliable. The NF-ToN-IoT-v2 dataset can be obtained from the following website: https://doi.org/10.48610/38a2d07. The NF-BoT-IoT-v2 dataset can be obtained from the following website: https://doi.org/10.48610/ec73920. The CEC2017 benchmark functions can be obtained from the following website: https://github.com/P-N-Suganthan/CEC2017-BoundContrained.

**Ethics Approval:** Not applicable.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Furstenau LB, Rodrigues YPR, Sott MK, Leivas P, Dohan MS, López-Robles JR, et al. Internet of Things: conceptual network structure, main challenges and future directions. Digit Commun Netw. 2023;9(3):677–87. doi:10.1016/j.dcan.2022.04.027.
2. Heidari A, Ali Jabraeil Jamali M. Internet of Things intrusion detection systems: a comprehensive review and future directions. Cluster Comput. 2023;26(6):3753–80. doi:10.1007/s10586-022-03776-z.

3. Adewole KS, Jacobsson A, Davidsson P. Intrusion detection framework for internet of things with rule induction for model explanation. Sensors. 2025;25(6):1845. doi:10.3390/s25061845.
4. Eid AM, Soudan B, Bou Nassif A, Injadat M. Comparative study of ML models for IIoT intrusion detection: impact of data preprocessing and balancing. Neural Comput Appl. 2024;36(13):6955–72. doi:10.1007/s00521-024-09439-x.
5. Jamshidi S, Nafi KW, Nikanjam A, Khomh F. Evaluating machine learning-driven intrusion detection systems in IoT: performance and energy consumption. Comput Ind Eng. 2025;204(1):111103. doi:10.1016/j.cie.2025.111103.
6. Nandanwar H, Katarya R. Deep learning enabled intrusion detection system for Industrial IOT environment. Expert Syst Appl. 2024;249(11):123808. doi:10.1016/j.eswa.2024.123808.
7. Ahmad B, Wu Z, Huang Y, Rehman SU. Enhancing the security in IoT and IIoT networks: an intrusion detection scheme leveraging deep transfer learning. Knowl Based Syst. 2024;305(1):112614. doi:10.1016/j.knosys.2024.112614.
8. Abiha UE, Rehman A, Abbas A, Haider MA, Al-Yarimi FAM, Gul MU, et al. Improving adversarial resilience for anomaly detection in the heterogeneous Internet of Things through ensemble models. Future Gener Comput Syst. 2026;178(4):108299. doi:10.1016/j.future.2025.108299.
9. Li J, Othman MS, Chen H, Yusuf LM. Optimizing IoT intrusion detection system: feature selection versus feature extraction in machine learning. J Big Data. 2024;11(1):36. doi:10.1186/s40537-024-00892-y.
10. Ayad AG, Sakr NA, Hikal NA. A hybrid approach for efficient feature selection in anomaly intrusion detection for IoT networks. J Supercomput. 2024;80(19):26942–84. doi:10.1007/s11227-024-06409-x.
11. Wakili A, Bakkali S. A resilient IoT intrusion detection system using hybrid feature selection and explainable ensemble learning. Results Eng. 2025;28(13):107392. doi:10.1016/j.rineng.2025.107392.
12. Wang J, Xiong X, Chen G, Ouyang R, Gao Y, Alfarraj O. Multi-criteria feature selection based Intrusion detection for internet of things big data. Sensors. 2023;23(17):7434. doi:10.3390/s23177434.
13. Awotunde JB, Folorunso SO, Imoize AL, Odunuga JO, Lee CC, Li CT, et al. An ensemble tree-based model for Intrusion detection in industrial internet of things networks. Appl Sci. 2023;13(4):2479. doi:10.3390/app13042479.
14. Ahsan MS, Islam S, Shatabda S. A systematic review of metaheuristics-based and machine learning-driven intrusion detection systems in IoT. Swarm Evol Comput. 2025;96(22):101984. doi:10.1016/j.swevo.2025.101984.
15. Gong X, Yang Y, Zhang Y, Li N, Guan Y, Jiang RK. Feature selection method for network intrusion based on hybrid meta-heuristic dynamic optimization algorithm. Comput Secur. 2025;156(10):104512. doi:10.1016/j.cose.2025.104512.
16. Abid T, Ahmim A, Maazouzi F, Chefrour D, Ullah I, Ahmim M, et al. A novel IoT threat detection using GWO feature selection and CNN-enhanced LightGBM. J Cloud Comput. 2025;14(1):72. doi:10.1186/s13677-025-00785-2.
17. Xu H, Przystupa K, Fang C, Marciniak A, Kochan O, Beshley M. A combination strategy of feature selection based on an integrated optimization algorithm and weighted K-nearest neighbor to improve the performance of network Intrusion detection. Electronics. 2020;9(8):1206. doi:10.3390/electronics9081206.
18. Xu H, Hu Y, Cao W, Han L. An improved jump spider optimization for network traffic identification feature selection. Comput Mater Contin. 2023;76(3):3239–55. doi:10.32604/cmc.2023.039227.
19. Xu H, Lu Y, Guo Q. Application of improved butterfly optimization algorithm combined with black Widow optimization in feature selection of network Intrusion detection. Electronics. 2022;11(21):3531. doi:10.3390/electronics11213531.
20. Xu H, Huang W, Bai L. AI-integrated feature selection of intrusion detection for both SDN and traditional network architectures using an improved crayfish optimization algorithm. Comput Mater Contin. 2025;84(2):3053–73. doi:10.32604/cmc.2025.064930.
21. Xu H, Chen J, Hu Z. Metaheuristic-driven abnormal traffic detection model for SDN based on improved tyrannosaurus optimization algorithm. Comput Mater Contin. 2025;83(3):4495–513. doi:10.32604/cmc.2025.062189.
22. Xiao Y, Cui H, Abu Khurma R, Castillo PA. Artificial lemming algorithm: a novel bionic meta-heuristic technique for solving real-world engineering optimization problems. Artif Intell Rev. 2025;58(3):84. doi:10.1007/s10462-024-11023-7.
23. Zhu X, Jia C, Zhao J, Xia C, Peng W, Huang J, et al. An enhanced artificial lemming algorithm and its application in UAV path planning. Biomimetics. 2025;10(6):377. doi:10.3390/biomimetics10060377.

24. Xie Y, Sun Z, Sun Z, Yuan K, Sun Z, Sun Z. 3D UAV route optimization in complex environments using an enhanced artificial Lemming algorithm. Symmetry. 2025;17(6):946. doi:10.3390/sym17060946.
25. Ding Z, Cao L, Chen L, Sun D, Zhang X, Tao Z. Large-scale multimodal multiobjective evolutionary optimization based on hybrid hierarchical clustering. Knowl Based Syst. 2023;266(1):110398. doi:10.1016/j.knosys.2023.110398.
26. Long S, Zheng J, Deng Q, Liu Y, Zou J, Yang S. A similarity-detection-based evolutionary algorithm for large-scale multimodal multi-objective optimization. Swarm Evol Comput. 2024;87(3):101548. doi:10.1016/j.swevo.2024.101548.
27. Zheng J, Yuan T, Xie W, Yang Z, Yu D. An enhanced flower Pollination algorithm with Gaussian perturbation for node location of a WSN. Sensors. 2023;23(14):6463. doi:10.3390/s23146463.
28. Choi KP, Kam EHH, Tong XT, Wong WK. Appropriate noise addition to metaheuristic algorithms can enhance their performance. Sci Rep. 2023;13(1):5291. doi:10.1038/s41598-023-29618-5.
29. Cao Z, Zhao Z, Shang W, Ai S, Shen S. Using the ToN-IoT dataset to develop a new intrusion detection system for industrial IoT devices. Multimed Tools Appl. 2025;84(16):16425–53. doi:10.1007/s11042-024-19695-7.
30. Luo Y, Chen X, Ge N, Feng W, Lu J. Transformer-based device-type identification in heterogeneous IoT traffic. IEEE Internet Things J. 2023;10(6):5050–62. doi:10.1109/jiot.2022.3221967.
31. Walling S, Lodh S. A review on feature selection techniques and its significance on IoT IDS. SN Comput Sci. 2025;6(8):935. doi:10.1007/s42979-025-04480-6.
32. Abdel-Basset M, Mohamed R, Abouhawwash M. Fungal growth optimizer: a novel nature-inspired metaheuristic algorithm for stochastic optimization. Comput Meth Appl Mech Eng. 2025;437(10):117825. doi:10.1016/j.cma.2025.117825.
33. Tang Y, Huang K, Tan Z, Fang M, Huang H. Multi-subswarm cooperative particle swarm optimization algorithm and its application. Inf Sci. 2024;677(1):120887. doi:10.1016/j.ins.2024.120887.
34. Yu D, Ji Y, Xia Y. Projection-iterative-methods-based optimizer: a novel metaheuristic algorithm for continuous optimization problems and feature selection. Knowl Based Syst. 2025;326(2):113978. doi:10.1016/j.knosys.2025.113978.
35. Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Adv Eng Softw. 2014;69:46–61. doi:10.1016/j.advengsoft.2013.12.007.
36. Sarhan M, Layeghy S, Portmann M. Towards a standard feature set for network intrusion detection system datasets. Mob Netw Appl. 2022;27(1):357–70. doi:10.1007/s11036-021-01843-0.