Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.072806

ARTICLE

Enhancing Intrusion Detection Systems Using Hybrid AI-Based Approaches

Mohammad Alshinwan[1], Radwan M. Batyha[1,2], Walaa Alayed[3,*], Saad Said Alqahtany[4], Suhaila Abuowaida[5], Hamza A. Mashagba[6], Azlan B. Abd Aziz[6,*] and Samir Salem Al-Bawri[7]

[1]Faculty of Information Technology, Applied Science Private University, Amman, Jordan
[2]Department of Computer Science, Faculty of Science and Information Technology, Irbid National University, Irbid, Jordan
[3]Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia
[4]College of Computer and Information Systems, Islamic University of Madinah, Madinah Munawarah, Medina, Saudi Arabia
[5]Department of Data Science and Artificial Intelligence, Faculty of Information Technology, Al al-Bayt University, Mafraq, Jordan
[6]Centre for Wireless Technology (CWT), Faculty of Engineering and Technology, Multimedia University, Melaka, Malaysia
[7]Space Science Centre, Institute of Climate Change, Universiti Kebangsaan Malaysia (UKM), Bangi, Malaysia

*Corresponding Authors: Walaa Alayed. Email: wmalayed@pnu.edu.sa; Azlan B. Abd Aziz. Email: azlan.abdaziz@mmu.edu.my

Received: 04 September 2025; Accepted: 08 January 2026; Published: 12 March 2026

ABSTRACT: Safeguarding modern networks from cyber intrusions has become increasingly challenging as attackers continually refine their evasion tactics. Although numerous machine-learning-based intrusion detection systems (IDS) have been developed, their effectiveness is often constrained by high dimensionality and redundant features that degrade both accuracy and efficiency. This study introduces a hybrid feature-selection framework that integrates the exploration capability of Prairie Dog Optimization (PDO) with the exploitation behavior of Ant Colony Optimization (ACO). The proposed PDO–ACO algorithm identifies a concise yet discriminative subset of features from the NSL-KDD dataset and evaluates them using a Support Vector Machine (SVM) classifier. Experimental analyses reveal that the PDO–ACO model achieves superior detection accuracy of 98% while significantly lowering false alarms and computational overhead. Further validation on the CEC2017 benchmark suite confirms the robustness and adaptability of the hybrid model across diverse optimization landscapes, positioning PDO–ACO as an efficient and scalable approach for intelligent intrusion detection.

KEYWORDS: Intrusion detection system; prairie dog optimization; artificial bee colony; support vector machine

1 Introduction

The rapid development of the Internet and IoT has promoted the widespread deployment of various innovative city applications. Connectivity between various devices and the Internet has already become an inevitable trend in modern times. This includes IoV, smart home gadgets, and industrial gear. Cyber breaches can occur on these devices due to hardware resource limitations and a weak security infrastructure. Configuring security measures (encryption algorithms) for IoT devices is challenging since their cost is low, their power consumption is low, and their computational resources are restricted. On the other hand, the transparency of the IoT makes devices, networks, and platforms easy targets. All kinds of networks, including the Internet and local area networks, are directly accessible to devices. By analyzing private and sensitive data on the same devices using simple technologies, criminals can launch large-scale attacks [1].

Complementary to intrusion detection, recent advances in chaotic and quantum encryption have fortified data confidentiality within the IoT and multimedia environments. Several works have been done to introduce new chaos-driven frameworks for image and quantum data protection that offer lightweight yet highly unpredictable cryptographic mechanisms. For example, a cosine-modulated-polynomial chaotic map is considered a compact PRNG featuring superior ergodicity and sensitivity for image encryption. Similarly, a quantum image encryption algorithm based on a three-dimensional bounded nonlinear map (3D-BNM) utilizes the Novel Enhanced Quantum Representation (NEQR), Secure Hash Algorithm 256-bit (SHA-256) hashing, and qubit-level operations to achieve high-entropy quantum ciphers. Further research on four-dimensional (4D) chaotic quantum encryption demonstrates secure implementations for Noisy Intermediate-Scale Quantum (NISQ) devices, and a two-phase chaotic quantum confusion–diffusion architecture enhances encryption strength in post-quantum consumer technologies. Collectively, these approaches highlight the role of chaotic and quantum cryptography as complementary defences to intrusion detection systems (IDS), reinforcing end-to-end protection across emerging IoT and multimedia ecosystems.

The IDS is deployed inside the Network, and, like a network sniffer, it unsystematically safeguards and gathers network packets to detect any harmful event that may affect the Network. After the intrusion detection system (IDS) evaluates the captured network traffic, network administrators are notified to limit the attack's connections and prevent further network harm from malicious attacks. A firewall—a vital equipment for preserving network security—is also connected to the system. The IDS detects intrusions using signature-based detection (SBD) and anomaly-based detection (ABD). SBD can only detect known attack scenarios and is only as effective as its signatures; it cannot identify novel attacks. On the other hand, ABD distinguishes between the attacker's actions and those of a regular user. In circumstances of unknown attacks, the anomaly detection rates are highly inaccurate [2].

Among the most common kinds of intrusion detection systems are [3]:

- Network IDS (NIDS): Hardware (sensors) and software (console) make up a network intrusion detection system (NIDS), which controls and monitors packets of network traffic from various places for possible intrusions or anomalies.
- Host IDS: An IDS is specific to a host machine (or server). With its ability to decrypt data in transit across the Network, including registries, file attributes, and system configuration databases, this solution outperforms NIDS while limited to a single system.
- Cloud IDS: The cloud layer provides secure authentication into a shared group or API for demand-based access. Similarly, it will integrate with pre-existing intrusion detection systems and hypervisors.

Many research domains frequently employ feature selection. Intrusion detection systems, for example, utilize it. The feature selection process can be categorized by three primary methods: embedding, filter, and wrapper. The feature selection for a particular learning algorithm is incorporated into the training process using the embedded approach. The features predicted by various learning algorithms with high accuracy will be selected using the wrapper technique. The filter technique attempts to classify the remaining characteristics by randomly selecting a portion of the original features. The evaluation criteria are used to select those attributes [4]. Nevertheless, single-population metaheuristics like PSO and GWO often suffer from premature convergence, getting trapped in local optima when handling high-dimensional feature spaces.

One of the most crucial steps in optimizing IDS systems is the Feature Selection procedure, also known by a number of different names including variable selection, attribute selection, among others. FS decreases the dataset dimensionality by identifying and retaining only those most valuable and relevant attributes from the huge volume of network data. The detection procedure is faster and the computational

overhead reduced. It also reduces the impacts of superfluous or redundant components that can affect the accuracy and reliability of the IDS [5]. IDS deals with huge data volumes; therefore, the FS has to be complex. False correlations between the IDS attributes may slow down the learning or make the process run incorrectly. Apart from reducing the feature set and total volume of data, FS is used for performance and understanding enhancement by providing relevant and interesting information in a focused manner. Swarm-based algorithms take inspiration from social and behavioral patterns of a number of species. The optimal way for solving this problem is by utilizing one of several computer methods scientists have proposed to simulate the feeding habits of various species. Dimensionality reduction is a common machine learning phase required when working with extremely complicated feature sets. To choose an ideal subset of features to characterize the entire dataset, dimensionality reduction requires FS. FS will be utilized in numerous domains, including object identification, data mining, and classification, to eliminate excessive and redundant features from the original dataset. Given the volume of data IDSs manage, FS is an essential duty [6].

Metaheuristic algorithms have gained prominence due to their superior efficiency and speed compared to traditional deterministic methods, which consume fewer computational resources. These algorithms are versatile, easily applicable across various fields, and proficient in evading the trap of local optima, a common shortfall in deterministic methods lacking stochastic elements in their final stages [7]. Metaheuristics can effectively search the solution space more thoroughly than classical optimization techniques by introducing some randomness into the problem, and often even allow for multiple optimal solutions to be found. Their independence from gradient data and preference for starting searches with random solutions make them excellent at handling nonlinear problems, being particularly useful in situations where the derivative data is too poor or unknown [8].

Although gradient descent methods may converge more quickly than metaheuristics because resource slacks exist across many instances, metaheuristics prove to be more effective in linear settings than traditional optimization algorithms, which rely on deterministic mechanisms driven by theoretical or adjacent approaches, by using a random sample of the solution space to locate near-optimal solutions. Summary: Metaheuristics are a specialized type of global optimization technique based on swarm and evolutionary strategies, encompassing a wide range of solvers that include various types of evolutionary algorithms (EAs) inspired by the natural selection process. Successful metaheuristics combine exploration, the essentially random search of a vast solution space, with exploitation, which reduces the footprint around previously found solutions established in the encountered areas. It is important to note that it is challenging to optimize one of these phases without hurting the other, depending on what exact optimization task we are talking about [9].

The Ant colony optimization algorithm is an increasingly popular way to solve many engineering complex problems, which involves the optimal structural and mechanical design of large-scale systems with multiple parameters, subject to engineering constraints and performance requirements. Indeed, the ACO algorithm has already been used to design such engineering systems more efficiently and effectively (Dorigo et al., 2006). What is presented in this work is the hybrid of an immune system-based grasshopper optimization algorithm and an ant colony for a novel feature selection approach. The reason behind doing this is to find out which features are helpful in detecting spam and how few of the attributes we need to produce either an optimal or near-optimal result [10].

PDO-ACO combines both algorithms into one, using the balance property of ACO by including the best candidates from each cluster obtained in PDO and global solutions so as to enhance the ability of the algorithm to escape from local optima, hence effective in a vast search space. We will evaluate the

performance of this hybrid method on the NSL-KDD dataset for FS and comprehensively compare it with the existing metaheuristic FS algorithms [11].

We were motivated to select ACO and PDO due to their complementary characteristics for feature selection problems in high-dimensional space. ACO is effective with regard to discrete combinatorial search, where pheromone guides exploitation towards better solutions, while PDO provides adaptive exploration, with dynamic adjustment and rapid convergence facilitated by social and environmental modeling. The integration of both algorithms enables a balanced trade-off between exploration and exploitation, which is critical for identifying discriminative intrusion features while avoiding premature convergence. Compared with other metaheuristics such as PSO, GA, GWO, or DE, the PDO–ACO hybrid demonstrates superior adaptability and accuracy in nonlinear, multi-modal optimization scenarios typical of intrusion detection systems.

The remainder of this paper is organized as follows: Section 2 describes the related work. In Section 3, we introduce the PDO-ACO approach. Section 4 presents the experiment results. Section 5 concludes the proposed work.

The main scientific contributions of this work are summarized below:

- This paper presents a new way of combining Prairie Dog Optimization and Ant Colony Optimization for intrusion detection feature selection, where PDO is used to explore the search space first and ACO is then applied to refine the most promising solutions.
- A distinct solution-passing mechanism is adopted in which feature subsets obtained during the PDO stage are reused to initialize the ACO process, helping the algorithm avoid early stagnation during later iterations.
- The feature-selection task is formulated as a joint optimization problem that considers both classification accuracy and the number of selected features, rather than optimizing accuracy alone.
- The proposed approach is evaluated on both CEC2017 benchmark functions and the NSL-KDD dataset, where it achieves competitive detection performance while relying on a relatively small subset of features.

The remainder of this paper is organized as follows. Section 2 reviews related studies on intrusion detection systems and recent metaheuristic-based feature-selection approaches. Section 3 describes the proposed hybrid PDO–ACO framework in detail, including the underlying optimization principles, mathematical formulation, and classification model. Section 4 presents the experimental setup and results, covering both benchmark optimization functions and intrusion detection performance evaluation. Finally, Section 5 concludes the paper and outlines potential directions for future research.

2 Related Work

With the growing number of computer threats, intrusion detection systems (IDS) have become very vital to network security. This stems from a noticeable increase in access to information technology and the Internet throughout the preceding several decades. A hybrid-based solution that is reliant on the Particle Swarm Optimization (PSO) algorithm, plus Artificial Bee Colony (ABC) [12]. The techniques are combined to discover superior optimization results, and the 10-fold cross-validation is used to validate the classification accuracies. Elngar et al., presents an efficient information distribution system (IDS) using a hidden naive Bayes classifier in combination with Particle Swarm Optimization (PSO) and the Information Entropy Minimization (IEM) discretize method [13]. Many experiments are carried out to test the efficacy of the proposed Network IDS using the NSL-KDD network intrusion detection dataset. The HNB classifier is compared with Information Gain, a popular feature selection technique. The gray wolf optimizer is implemented to explore the feature space and identify the optimal feature subset that improves classification

accuracy. Initially, the Gray Wolf Optimizer employs filter-based techniques to discover solutions with minimal repetition, which are mutually informative [14]. An optimization wrapper method elevates the classifier's performance in the later stages of the process. By using the NSL KDD Dataset, the performance of the grey wolf optimizer is gauged against that of other metaheuristic algorithms. Due to the vast amount of data, intrusion detection systems (IDSs) must maintain high-quality features that accurately represent the data while excluding duplicate and irrelevant characteristics. The cuttlefish algorithm (CFA) is used to get a feature subset for another filter, and then the decision tree classifier was employed to evaluate those features provided by CFA [6]. In a different study, Mohammadi et al. (2019) proposed a feature clustering approach that integrates both filtering and wrapping strategies. They utilized the linear correlation coefficient algorithm as a filter and the CFA as a wrapper to enhance the intrusion detection process [15].

To carry out feature selection for SVM and parameter optimization, Gauthama Raman et al. apply the Hypergraph-based Genetic Algorithm (HG-GA) [16]. The hyper-clique property of a hypergraph can be utilized effectively to rapidly generate an initial population and to escape from local minima during the search process for a better and optimal solution. On the one hand, the HG GA combines the requirements of increasing Detection Rate (DR) and reducing False Alarm Rate (FAR), without exaggerating their importance. On the other hand, it incorporates a weighted objective function to assess the number of features precisely.

The AdaBoost algorithm is mearge with ABC algorithm to achieve a high rate of detection while keeping the FPR low [17]. The features are passed through AdaBoost for evaluation and classification, while ABC is used to select them. It is validated in simulation results using the NSL-KDD and ISCX IDS 2012 datasets; this hybrid method signifies a significant leap from existing IDS methods performed on the same dataset and works fairly well in various attack-related levels. However, the use of ensemble classifiers like AdaBoost often increases the training time and computational complexity, making it less ideal for real-time environments with limited resources.

Artificial Fish Swarm (AFS) and ABC algorithms come into play in unveiling a new hybrid classification strategy [18]. The Fuzzy C-Means Clustering technique is employed to partition the training dataset, and Correlation-based Feature Selection (CFS) is implemented to eliminate redundant features. By using the CART technique, if-then rules are formulated based on selected criteria to differentiate between normal and abnormal data. In a similar vein, the devised rules are used to train the hybrid technique under consideration. The suggested method outperforms the others with a 99% detection rate and a 0.01% false positive rate on the UNSW-NB15 and NSL-KDD datasets.

Recently, several studies have reported high detection accuracy using advanced hybrid or deep metaheuristic techniques. Khare et al. introduced the SMO-DNN model, combining Support-Matrix Optimization with a deep neural network and achieving 99.1% accuracy on the NSL-KDD dataset [19]. Despite this high accuracy, Deep Neural Networks (DNNs) operate as 'black boxes' with low interpretability and require substantial computational power (GPUs), which restricts their deployment in edge-based or resource-constrained IDS sensors. Kunhare et al. employed a PSO-based feature-selection strategy with an SVM classifier, obtaining 98.7% accuracy while using roughly one-quarter of the features [20]. Kumari et al. utilized the Spider Monkey Optimization algorithm for feature selection on the UNSW-NB15 dataset, reaching 99.3% accuracy [21]. Although recent studies report impressive detection rates, a closer look at the literature highlights a significant trade-off. Models based on deep neural networks often achieve high accuracy, but they require substantial computational power. This heavy overhead makes them difficult to deploy in real-time or resource-limited sensors. On the other hand, standard metaheuristic algorithms are more efficient but frequently get trapped in local optima, failing to fully explore complex feature spaces. This leaves a clear gap in the research: there is a need for a lightweight optimization framework that can find

highly relevant features without the high resource demands of deep learning. This study aims to bridge that gap with a hybrid PDO–ACO approach. We designed this method to balance global exploration with local exploitation, delivering high detection performance while minimizing computational costs. The prior IDS techniques proposed are illustrated in Table 1.

Table 1: Overview of existing intrusion detection systems and their underlying optimization methodologies.

Study	Optimization Method	Feature Selection Method	Classifier	Dataset	Performance Metrics
[12]	Artificial Bee Colony, Particle Swarm Optimization (PSO)	10-fold cross-validation, Combined algorithms	Not specified	Not specified	Classification accuracies
[13]	Particle Swarm Optimization (PSO), Hidden Naïve Bayes (HNB)	Information Entropy Minimization (IEM) discretized approach	HNB classifier	NSL-KDD dataset	Performance assessment (not specified), Comparison with IG
[14]	Gray Wolf Optimizer	Filter-based methods, Optimization wrapper strategy	Not specified	NSL KDD Dataset	Performance evaluation, Comparison with other metaheuristic algorithms
[6]	Cuttlefish Algorithm (CFA)	Decision Tree classifier	Decision tree classifier	KDD'99 dataset	Performance evaluation
[15]	Feature clustering (Filter and wrapper methods)	Linear correlation coefficient algorithm (Filter), CFA (Wrapper)	Decision Tree classifier	Not specified	Not specified
[16]	Hypergraph-based Genetic Algorithm (HG-GA)	Not specified	SVM	Not specified	Detection rate, False alarm rate, Feature optimization
[17]	AdaBoost, Artificial Bee Colony (ABC)	ABC for feature selection, AdaBoost for classification	Not specified	ISCXIDS2012, NSL-KDD datasets	Detection rate, False positive rate

(Continued)

Table 1 (continued)

Study	Optimization Method	Feature Selection Method	Classifier	Dataset	Performance Metrics
[18]	Artificial Fish Swarm (AFS), ABC	Fuzzy C-Means Clustering, Correlation-based Feature Selection (CFS)	CART technique	UNSW-NB15, NSL-KDD datasets	Detection rate, False positive rate
[19]	Spider Monkey Optimization (SMO) used for dimensionality reduction	SMO used to select/reduce features (i.e., feature subset)	Deep Neural Network (DNN)	NSL-KDD and KDD Cup 99	Accuracy, Detection Rate, Precision, F1-score, False Alarm Rate.
[21]	Spider Monkey Optimization (SMO) used to optimize neural network parameters/structure	Implicit feature/parameter optimization (not purely feature selection)	ANN	NSL-KDD, CIC-IDS 2017, Luflow	Accuracy.
[20]	Particle Swarm Optimization (PSO)	Random Forest (RF) used for feature-importance-based selection (10 features selected from 41)	k-NN, SVM, Logistic Regression (LR), Decision Tree (DT), Naïve Bayes (NB)—compared with PSO	NSL-KDD dataset	Accuracy, Detection Rate, Precision, F1-score, False Alarm Rate.

Table 1 highlights a recurring trade-off in existing intrusion detection research. On one hand, deep learning and ensemble classifiers often deliver excellent accuracy, but their heavy computational requirements make them difficult to deploy for real-time monitoring. On the other hand, standard metaheuristics like PSO or GWO are much faster but often struggle with high-dimensional data, leading to premature convergence. The hybrid framework proposed here addresses this specific limitation. We combine the global search capability of Prairie Dog Optimization with the local refinement of Ant Colony Optimization. This design aims to achieve the high detection rates typically associated with heavier models, but without sacrificing the efficiency needed for practical network security.

3 Proposed Works

The methodology is structured as follows: Section 3.1 outlines the exploration mechanics of the PDO algorithm, Section 3.2 describes the exploitation strategy of ACO, and Section 3.3 details the integration of these two methods into the proposed hybrid framework.

Optimization of the algorithms has been trending to enhance the performance of machine learning models. This part of the algorithm utilizes two state-of-the-art optimization algorithms: Ant Colony Optimization (ACO) and Prairie Dog Optimization (PDO). Inspired by natural processes, these algorithms are designed to explore large solution spaces in search of global optima efficiently. The Support Vector Machine (SVM) is a fast and reliable machine learning algorithm that can function as both a regression and classification model, as well as for outlier detection (where an arbitrary value is significantly distant from normal values). The accuracy of the model is improved using PDO and ACO optimization for SVM parameters, and also reduces overfitting. At the same time, the performance of ML models and their results can also be improved when these optimization techniques are used in conjunction with SVM. The study utilizes the NSL-KDD dataset, which is widely recognized and commonly used for benchmarking various intrusion detection systems. The subsequent sections provide a more detailed account of the multiple phases within the proposed approach.

3.1 Prairie Dog Optimization Algorithm (PDO)

PDO is a new optimization technique designed to mimic the foraging and burrow construction actions of prairie dogs. The process begins with a random search pattern to identify the area of interest with potential. Three mathematical stages (initialization and assessment, exploration, and exploitation) are available for carrying out the iterative process in PDO. During the initialization and evaluation phase, PDO examines a group of *(COT)* coteries, each containing *M* prairie dogs (P.D.s). The colony of *COT* coteries is represented by a matrix showing the possible positions of these coteries and the colony (*C.O.*) using Eq. (1), where $CO_{i,j}$ represents the *i*th coterie of the *j*th element within the colony. Each coterie is represented by the matrix (*P.D.*) containing the possible positions of *M* prairie dogs [11].

$$C.O. = \begin{bmatrix} CO_{11} & CO_{12} & \dots & CO1_s \\ \vdots & \vdots & CO_{ij} & \dots \\ CO_{COT1} & CO_{COT2} & \dots & CO_{COtd} \end{bmatrix} \tag{1}$$

$$P.D. = \begin{bmatrix} S_{11} & S_{12} & \dots & S1_K \\ \vdots & \vdots & S_{ij} & \dots \\ S_{M1} & S_{M2} & \dots & S_{MK} \end{bmatrix} \tag{2}$$

The symbol S_{ij} signifies the *j*th element of the *i*th set. Find the partial derivative of the function with respect to the given variable and ensure it is less than or equal to *COT*. Following initialization, the prairie dog's position fitness is assessed using the goal or target function, which indicates the quality of the discovered food supply. Regarding the minimization aim, the lowest fitness value indicates the optimal solution of the colony. During the exploration phase, P.D.s search for new burrows near a rich food source as part of their exploratory behavior. The PDO conducts the exploration search based on two criteria. The initial criterion utilizes Levy flying to enable prairie dogs to make extended jumping movements to search for new food sources. The second criterion evaluates the efficiency of the excavation process and the quality of the food sources. The updated location for constructing the burrow is represented as follows (Eq. (3)), where the symbol Sbest denotes the globally optimal solution.

$$Si + 1, j + 1 = S_{best,j} \times Sq, j \times DS \times Levy\,(M)\ \forall Lmax4 \le l < Lmax2 \tag{3}$$

The variables CO_{besti}, as shown in Eq. (4), examine the impacts of the most efficient solution to date. $B_{r,j}$ represents a solution generated randomly. X_i, as defined in Eq. (5), indicates the combined impact of each prairie dog in the colony, whereas PD*s*, as defined in Eq. (6), denotes the digging ability of the group.

The *Levy* distribution function, denoted as *Levy*(*Z*), is utilized to provide increased exploration by incorporating a variety of jumping steps.

$$CO_{best,j} = B_{best,j} \times \lambda + \frac{B_{i,j} \times mean\left(P_{M,COT}\right)}{B_{best,j} \times \left(Y_j^{LB} - Y_j^{UB}\right) + \lambda} \tag{4}$$

$$X_{i,j} = \frac{B_{best,j} - B_r}{Bbest,\, j + \lambda} \tag{5}$$

$$PDs = 1.5 \times \mu \left(1 - \frac{w}{W_{\max}}\right)^{\left(2 \times \frac{w}{W_{\max}}\right)} \tag{6}$$

here, μ represents a stochastic *number* that alternates between 1 and −1 based on whether the iteration is even or odd, λ represents a small integer, and w and $W_{\max}$ represent the current iteration and the maximum number of iterations, respectively. Algorithm 1 presents the pseudo-code for the PDO algorithm.

Algorithm 1: PDO pseudo-code

Step	**Description**
Input:	M (Population size), W (Maximum iterations), LB, UB (Lower/Upper bounds)
Output:	CO_{Best}, S_{Best} (Best solution found)
1	Initialize all necessary variables
2	Evaluate fitness for each solution and record the best one (S_{Best})
3	**While** iteration count < W **do**
4	Calculate PDs and X according to the current phase
5	**For** each solution i = 1 to M **do**
6	Calculate CO and B parameters
7	Update solution depending on the current phase of iteration
8	Apply boundary conditions to the new solution
9	Calculate new fitness
10	**If** new fitness is better **then**
11	Update solution and its fitness
12	**End If**
13	**If** current fitness is globally best **then**
14	Update global best fitness (SBest) and solution (CO_{Best})
15	**End If**
16	**End For**
17	Update convergence curve
18	Display the best solution fitness
19	Increment iteration count
20	**End While**
21	**Return** S_{Best}

3.2 Ant Colony Optimization (ACO)

One of the primary concerns in intrusion detection systems (IDSs) is improving detectability while minimizing computational requirements; both can be achieved by selecting relevant features. It provides a robust method for hyperparameter tuning in the large, nonlinear search space of feature combinations.

Within the ACO framework, a swarm of artificial agents, called ants, iteratively construct candidate solutions by probabilistically selecting features based on two main signals: (1) a pheromone trail that encodes learned desirability from past iterations, and (2) a heuristic measure that quantifies the inherent usefulness of a feature.

Let F = $\{f_1, f_2, \ldots, f_n\}$ denote the complete set of features, and let $\tau_i(t)$ represent the intensity of the pheromone associated with the feature f_i at iteration t. The heuristic desirability of each feature, denoted η_i, is computed based on a domain-specific measure such as Information Gain or Mutual Information. At each construction step, the probability $p_i^k(t)$ that ant k selects feature f_i is given by:

$$p_i^k(t) = \frac{[\tau_i(t)]^\alpha \cdot [\eta_i]^\beta}{\sum_{j \in N_k} [\tau_i(t)]^\alpha \cdot [\eta_i]^\beta} \tag{7}$$

here, $N_k \subseteq F$ denotes the set of features not yet selected by ant k, while α and β are control parameters that regulate the influence of pheromone and heuristic information, respectively.

Once the solution construction phase is over, the selected feature subset S_k of each ant is evaluated using a fitness function. In this respect, a SVM classifier is trained with the selected features, and the classification performance is measured. The fitness function L_k used in this work balances the predictive performance with the dimensionality reduction by taking into account both accuracy and the size of the feature subset. It is defined as follows:

$$\boldsymbol{L_k} = \boldsymbol{w_1} \cdot (\mathbf{1} - \boldsymbol{Accuracy}(\boldsymbol{S_k})) + \boldsymbol{w_2} \cdot \frac{|\boldsymbol{S_k}|}{|\boldsymbol{F}|} \tag{8}$$

where $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ are weighting coefficients satisfying $\boldsymbol{w}_1 + \boldsymbol{w}_2 = 1$, and $|\mathbf{S_k}|$ is the cardinality of the selected feature subset.

The pheromone update rule comprises two stages: evaporation and reinforcement. First, existing pheromone levels decay over time to prevent early convergence:

$$\tau_i(t) \leftarrow (1 - \rho) \cdot \tau_i(t) \tag{9}$$

where $\rho \in (0, 1)$ is the pheromone evaporation rate, subsequently, pheromones are reinforced on features that contribute to high-quality solutions. The pheromone increment $\Delta\tau_i^k(t)$ for feature f_i selected by ant k is given by:

$$\Delta\tau_i^k(t) = \begin{cases} \dfrac{Q}{L_k}, & \text{if } f_i \in S_k \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

The total pheromone update is expressed as:

$$\tau_i(t+1) = \tau_i(t) + \sum_{k=1}^{m} \Delta\tau_i^k(t) \tag{11}$$

where m is the number of ants and Q is a positive constant.

The ACO process continues over multiple iterations until a predefined termination criterion is met, such as a maximum number of iterations or stagnation in the best-obtained fitness. The subset with the lowest cost function L_k is retained as the optimal outcome of feature selection. Algorithm 2 presents the pseudo-code of the ACO algorithm.

Algorithm 2: ACO pseudo-code

Input: Dataset D with feature set $F = \{f_1, f_2, \ldots, f_n\}$ Number of ants m, maximum iterations T Pheromone factor α, heuristic factor β, evaporation rate ρ Max subset size max_features Classifier for evaluation (e.g., SVM)

Output: Optimal feature subset *F_best*

Initialize pheromone levels $\tau_i(0) = \tau_0$ for all features $f_i \in F$
Compute heuristic desirability η_i for each feature (e.g., Information Gain)
Set *F_best* $\leftarrow \varnothing$ and *L_best* $\leftarrow \infty$
For t = 1 to T do:
    For each ant k = 1 to m **do**:
        Initialize subset $S_k \leftarrow \varnothing$
        While
            Select feature $f_j \notin S_k$ with probability: $P_j(t) \propto [\tau_j(t)]^{\alpha} \times [\eta_j]^{\beta}$
            Add f_j to S_k
        Evaluate fitness L_k of S_k using the classifier
        If $L_k <$ L_best then:
            Update F_best $\leftarrow S_k$ and L_best $\leftarrow L_k$
        End If
    End For
    For each feature $f_i \in F$ **do**:
        Apply evaporation: $\tau_i(t) \leftarrow (1 - \rho) \times \tau_i(t)$
    End For
End For
    For each ant k = 1 to m **do**:
        If $f_i \in S_k$ **then**:
            $\tau_i(t) \leftarrow \tau_i(t) + Q/L_k$
        End If
    End For
Return F_best

3.3 Proposed Hybrid PDO-ACO Algorithm

This section presents the core contribution of this study: a unified optimization architecture that merges the search capabilities of PDO and ACO.

To overcome the limitations of individual metaheuristics and capitalize on their strengths, this study introduces a novel hybrid optimization framework that combines Prairie Dog Optimization (PDO) and Ant Colony Optimization (ACO). While PDO performs well in global exploration with its stochastic burrowing and foraging behavior, it suffers from slow convergence and premature stagnation in local optima. On the other hand, ACO is a strong local optimizer that effectively exploits the search space by pheromone-guided learning; however, its global diversification potential is not very effective, especially during the early stages.

The proposed hybrid approach combines the strengths of PDO and ACO: the PDO algorithm is used to generate a high-quality pool of candidate feature subsets by using its exploration capability. These candidate solutions, called coteries, are then fed into the ACO phase that refines the most promising subsets by means of pheromone-driven selection probabilities and heuristic evaluation. Fig. 1 depicts the proposed algorithm.

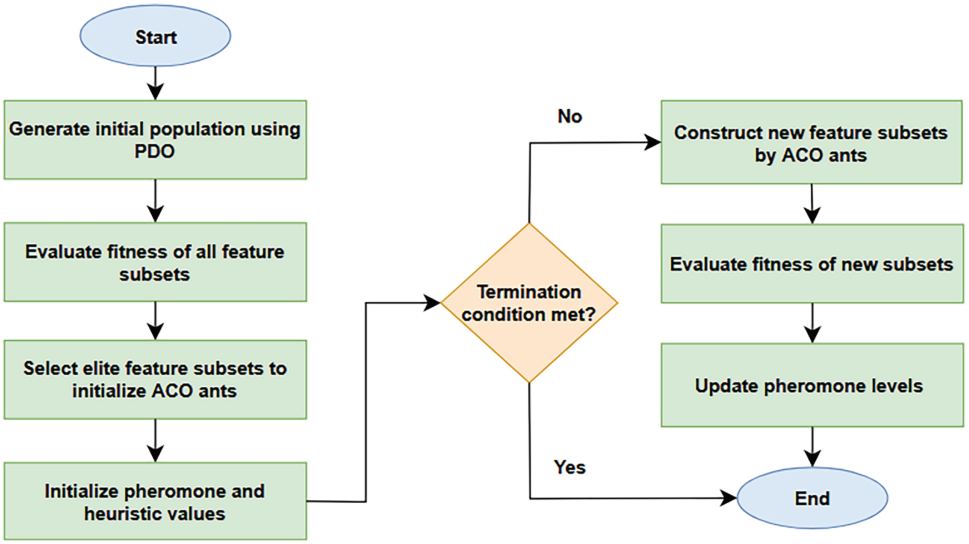


Figure 1: Schematic diagram illustrating the sequential PDO–ACO feature selection mechanism.

It is important to note that some comparative algorithms, such as the Whale Optimization Algorithm (WOA) [22], the Salp Swarm Algorithm (SSA), and the Slime Mold Algorithm (SMA), were originally formulated for continuous search spaces. In this study, these methods were adapted for feature selection using a standard sigmoid transfer function to map continuous positions into binary decision values—a strategy that has been widely adopted in recent feature-selection research [23,24]. This adaptation ensures a consistent experimental framework across multiple optimizers. However, unlike these single-phase binary adaptations, the proposed PDO–ACO framework employs a two-stage cooperative mechanism, where PDO performs global exploration to generate diverse candidate subsets and ACO refines these subsets through pheromone-guided local exploitation. Therefore, while the sigmoid mapping provides baseline comparability for continuous algorithms, PDO–ACO fundamentally differs by combining two complementary search strategies that interact sequentially rather than relying on a single binary transformation.

Mathematical Representation of the Hybrid Mechanism

Let:

- $C = \{C_1, C_2, ..., C_N\}$ be the set of coteries (feature subsets) generated by PDO.
- Each $C_i \in C$ consists of M prairie dogs, i.e., feature subsets $S_{i,j}$,
- $F_{elite} \subset \bigcup C$ set of top-ranked feature subsets selected based on a fitness function L,
- $\tau_f(t)$: pheromone level for feature f at iteration t,
- ηf: heuristic value of feature f, e.g., Information Gain,
- $p_i^k(t)$: probability that ant k selects feature f during ACO-based refinement.

The transition probability for each ant to choose a feature f is defined as:

$$P_f^k(t) = \frac{\left[\tau_f(t)\right]^{\alpha} \cdot \left[\eta_f\right]^{\beta}}{\sum_{g \in U_k} \left[\tau_g(t)\right]^{\alpha} \cdot \left[\eta_g\right]^{\beta}} \tag{12}$$

where U_k is the set of features not yet selected by ant k, and α, β are tuning parameters.

The fitness function L_k of each subset S_k is evaluated using an SVM classifier and incorporates both classification accuracy and feature set size (Eq. (8)).

Algorithm 3 shows the pseudo-code of the proposed PDO-ACO algorithm.

Algorithm 3: Pseudo-code of the proposed PDO-ACO algorithm

Step	Description
	Dataset D with feature set $F = \{f_1, f_2, ..., f_n\}$.
	Number of coteries N, prairie dogs per coterie M.
Input:	Number of ants m, iterations T.
	Parameters: α, β, ρ, max_features, fitness weights w_1,w_2.
	Classifier: SVM.
Output:	Optimal feature subset F_{best}.
1.	Generate initial population using PDO: create N coteries with M feature subsets each
2.	Evaluate fitness of all $S_{i,j} \in C$ using SVM-based cost function L
3.	Select top m elite feature subsets from C to initialize ACO ants
4.	Initialize pheromone values τ_f and heuristic values η_f
5.	**Set** $F_{best} \leftarrow \varnothing$, $L_{best} \leftarrow \infty$
6.	**For** $t = 1$ **to** T:
7.	**For** each ant $k = 1$ to m:
8.	**Initialize** $S_k \leftarrow \varnothing$
9.	**While**
10.	Select feature $f \notin S_k$ using $P_f^k(t)$
11.	Add f to S_k
12.	Evaluate fitness L_k using the classifier
13.	**End While**
14.	**If** $L_k < L_{best}$: **update** $F_{best} \leftarrow S_k$, $L_{best} \leftarrow L_k$
15.	**End If**
16.	**End For**
17.	**For** each feature $f \in F$:
18.	Evaporate pheromone: $\tau_f(t) \leftarrow (1-\rho) \cdot \tau_f(t)$
19.	Reinforce: for each ant k, if $f \in S_k$, then $\tau_f(t) \leftarrow \tau_f(t) + Q/L_k$
20.	**End For**
21.	**Return** F_{best}
	End For

3.4 Classification Using Support Vector Machines (SVMs)

SVMs are related to supervised machine learning (ML) models that examine data to classify target objects, a type of linear classification and regression. SVMs are notable for their ability to perform and generalize effectively, even on relatively small sample sizes for training or learning data. Because of their resilience, adaptability, and effectiveness on datasets of various sizes, they are a commonly used approach, particularly in intrusion detection. SVMs make no assumptions about the data because they aim to find a hyperplane in an N-dimensional space that can categorize the data points.

The core concept is to utilize intelligent calculation to transform a low-dimensional space into a high-dimensional space, and then apply this transformation to identify the optimal classification method.

In IDS, distributions of different types of hits are usually imbalanced, in which low-frequency attacks are significantly smaller compared to high-frequency attacks. Due to their outstanding generalization ability even with small datasets, low classification time, and efficient categorization of various classes, SVMs are among the most successful and popular intrusion detection algorithms. The SVM classifier is of primary importance in intrusion detection. The system generates a standard model of the target system, and links attribute state data to a categorization outcome. Classification aims to create a decision function using existing sample data. Fig. 2 illustrates the proposed SVM model for categorizing the outcomes obtained from the PDO-ACO system.

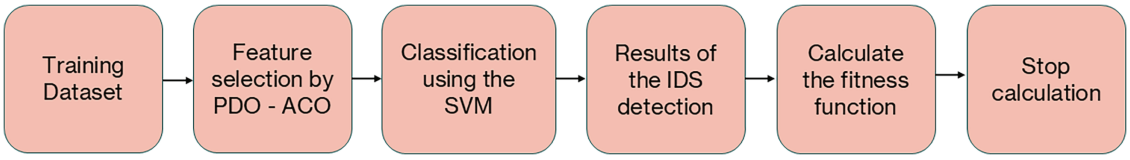


Figure 2: Structure of proposed IDS based on PDO-ACO and SVM.

3.5 Dataset

The KDD CUP 1999 dataset is a widely used benchmark for evaluating various intrusion detection algorithms. The dataset was obtained from the DARPA 98 IDS evaluation program led by Lincoln Labs. It comprises five million connection records emulated from the U. S. Air Force military environment. The NSL-KDD dataset is introduced as an improvement over the KDD Cup 1999 dataset, which had issues such as duplications, unequal sample distribution, and redundant records. The NSL-KDD dataset consists of 41 conditional attributes and a class label for each sample. The conditional attributes can be classified into four features: time-based traffic, host-based traffic, basic, and content, which can be discrete or continuous (see Fig. 3). Any network activity that strays from the "Normal" is classified as an attack (class label). The NSL-KDD dataset comprises 24 categories of assaults, classified into Denial of Service (DoS), Probe, User-to-Root (U2R), and Remote-to-Local (R2L) attacks, as illustrated in Fig. 4.

Time- based traffic	Host	Basic	Content
Dst Host Count	Count	Duration	Hot
Dst Host Srv Count	Srv Count	Protocol Type	Number Faild Logins
Dst Host Same Srv Rate	Serror Rate	Service	Logged In
Dst Host Diff Srv Count	Srvserror Rate	Flag	Num Compromised
Dst Host Same Src Port Ratet	Same Srv Rate	Source Bytes	Root Shell
Dst Host Srv Diff Host Rate	Diff Srv Rate	Destination Bytes	Su Attempted
Dst Host Serror Rate	Srv Diff Host Rate	Land	Num Root
		Wrong Fragment	Num File Creations
		Urgent	Num Shells
			Num Access Files
			Num Outbound Cmds
			Is Host Login
			Is Guest Login

Figure 3: Categorization of the conditional attributes within the NSL-KDD dataset.

Attacks			
DoS	Probe	U2R	R2L
Smurf	Ipsweep	Buffer overflow	Ftp write
Back	Nmap	Load module	Guess password
Land	Satan	Per1	Imap
Neptune	Portsweep	Per1	Mutlihop
Pod			Spy
Tear drop			Phf
			Warezclient
			Warezmaster

Figure 4: Taxonomy of attack types in the NSL-KDD dataset categorized by class (DoS, Probe, U2R, and R2L).

3.6 Time Complexity Analysis

The analysis of the time complexity of the proposed PDO-ACO algorithm provides an insight into its computational feasibility, which, in particular, is needed for real-time intrusion detection systems where efficiency matters. The total time complexity of PDO-ACO is due to two sequential optimization phases: global exploration performed by Prairie Dog Optimization (PDO) and local exploitation and refinement handled by Ant Colony Optimization (ACO).

3.6.1 Time Complexity of PDO Phase

Let:

- N: number of coteries (groups),
- M: number of prairie dogs per coterie,
- T_p: number of PDO iterations,
- d: dimensionality of the feature space (i.e., number of features).

For each iteration, every prairie dog evaluates a solution based on the current feature subset, involving fitness computation (e.g., SVM classification). The fitness evaluation dominates the cost.

- PDO Phase Complexity:

  $O(T_p \cdot N \cdot M \cdot C_{fitness})$

  where $C_{\text{fitness}} \approx O(d)$ assumes a linear classifier and a proportional relation to feature count.

3.6.2 Time Complexity of ACO Phase

Let:

- m: number of ants,
- T_a: number of ACO iterations.

In each ACO iteration, each ant builds a solution by probabilistically selecting features and evaluates its fitness. Pheromone update occurs after each iteration.

- ACO Phase Complexity:

  $O\,(T_a \cdot m \cdot d + T_a \cdot d)$

  where:

- $T_a \cdot m \cdot d$: solution construction and fitness evaluation,
- $T_a \cdot d$: pheromone update per feature.

3.6.3 Combined Time Complexity of PDO-ACO

Combining both phases, the overall time complexity of PDO-ACO becomes:

$O(T_p \cdot N \cdot M \cdot d + T_a \cdot m \cdot d)$

Assuming $T_p \approx T_a$ and $N \cdot M \approx m$, this can be simplified as:

$O(T \cdot m \cdot d)$

where T represents the total number of iterations for both phases.

This linear time complexity regarding feature count d and population size mmm shows that PDO-ACO exhibits good scalability for the moderate-sized feature spaces found in typical intrusion detection datasets. The extra computational load added by ACO's pheromone update mechanism is negligible when compared to fitness evaluations. Thus, PDO-ACO remains computationally efficient and suitable for real-time IDS applications, especially when combined with the feature reduction strategies shown here.

It is important to note that feature reduction and classification accuracy are treated as distinct yet complementary objectives within the optimization process. The fitness formulation does not assume that fewer features inherently yield higher accuracy; instead, the hybrid PDO–ACO algorithm searches for a Pareto-optimal balance that maximizes detection performance while minimizing feature redundancy and computational cost. This ensures that the resulting subset reflects both predictive relevance and model simplicity, rather than prioritizing one objective in isolation.

4 Experimental Results

This section validates the performance of the proposed PDO–ACO algorithm through two distinct evaluation phases. First, the global optimization capability of the hybrid engine is tested against the CEC2017 benchmark suite to assess its convergence behavior and stability. Second, the framework is applied to the intrusion detection domain, where its feature selection efficiency and classification accuracy are benchmarked on the NSL-KDD dataset against leading metaheuristic approaches

4.1 Benchmark Evaluation on CEC2017 Test Functions

The performance of the proposed PDO-ACO algorithm is evaluated by applying the standard CEC benchmark functions of CEC2017, which are widely utilized to evaluate the capability of metaheuristic algorithms with regard to global optimization. These functions represent the categories of unimodal, multimodal, hybrid, and composite problems, hence providing a good ground for validation. A summary of the CEC2017 benchmark functions is given in Table 2. The results reported in Table 3 represent the best, average, and standard deviation values obtained over these runs using five agents, fifty iterations, and a dimensionality of one hundred.

Table 2: Summary of CEC2017 benchmark functions.

NO.	Name	Equation
CEC01	Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$
CEC02	Rosenbrock	$f(x) = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2\right]$
CEC03	Rastrigin	$f(x) = An + \sum_{i=1}^{n} \left[x_i^2 - A\cos(2\pi x_i)\right], A = 10$

(Continued)

Table 2 (continued)

NO.	Name	Equation
CEC04	Griewank	$f(x) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right)$
CEC05	Ackley	$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$
CEC06	Schwefel	$f(x) = 418.9829n - \sum_{i=1}^{n} x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)$
CEC07	Zakharov	$f(x) = \sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} 0.5 i x_i\right)^2 + \left(\sum_{i=1}^{n} 0.5 i x_i\right)^4$
CEC08	Dixon–Price	$f(x) = (x_1 - 1)^2 + \sum_{i=2}^{n} i\left(2x_i^2 - x_{i-1}\right)^2$
CEC09	Michalewicz	$f(x) = -\sum_{i=1}^{n} \sin(x_i)\left[\sin\left(\frac{i x_i^2}{\pi}\right)\right]^{2m}, m = 10$
CEC10	Levy	$w_i = 1 + \frac{x_i - 1}{4}, f(x) = \sin^2(\pi w_1) + \sum_{i=1}^{n-1}(w_i - 1)^2\left[1 + 10\sin^2(\pi w_i + 1)\right] + (w_n - 1)^2\left[1 + \sin^2(2\pi w_n)\right]$

Table 3: Ranking results of the compared algorithms on the CEC2017 benchmark functions using 5 search agents, 50 iterations, and 100 dimensions.

Function	Measure	Algorithms					
		SSA	WOA	SMA	PDO	ACO	PDO-ACO
CEC01	Best	1,004,533.654	3.04245E+12	8.61677E+11	1.68273E+12	5.5077E+11	8.77509E+11
	Average	304,236.1806	1.76684E+12	3.9043E+11	7.8847E+11	3.34022E+11	4.85498E+11
	Worst	90,833.13108	6.63827E+11	1.3166E+11	1.41011E+11	30,599,880,299	10,350,671,261
	STD	392,622.2063	1.07682E+12	2.82839E+11	6.69567E+11	2.43137E+11	3.16037E+11
	p-value	0.00888828	0.034011549	0.629711957	0.386943816	0.420321804	1
	h	1	1	0	0	0	0
CEC02	Best	18.28566316	13,404.88227	8109.546863	123.1932438	572.403714	6485.278406
	Average	17.72180159	8734.384633	3079.583743	68.33935691	216.957496	3071.609822
	Worst	17.39040434	3812.190054	1250.476632	37.09659324	53.42493003	363.6039582
	STD	0.371872805	4006.090764	2854.710069	34.73653694	232.4481203	2558.93079
	p-value	0.028424938	0.028639396	0.996403032	0.030455776	0.037860035	1
	h	1	1	0	1	1	0
CEC03	Best	12.70449129	12.70806037	12.70620448	12.70669696	12.70499713	12.70815903
	Average	12.70333219	12.70747553	12.70497757	12.70411635	12.70434794	12.70446981
	Worst	12.70246417	12.70663869	12.70249274	12.70248467	12.70335591	12.70240847
	STD	0.000908042	0.000578721	0.001512009	0.001762211	0.000790107	0.00233884
	p-value	0.340311554	0.02357593	0.694202041	0.794072544	0.914826231	1
	h	0	1	0	0	0	0
CEC04	Best	30,475.41958	42,586.39167	15,409.09065	22,353.84191	17,186.01154	20,178.4362
	Average	14,909.78224	35,020.12264	9553.172867	16,552.82953	8532.643273	14,628.3242
	Worst	6915.4464	20,820.07013	4607.604757	12,192.44793	3529.459844	8671.836313
	STD	9964.643332	9200.729697	4805.983282	4369.762554	5289.687521	4725.107516

(Continued)

Table 3 (continued)

Function	Measure	Algorithms					
		SSA	WOA	SMA	PDO	ACO	PDO-ACO
	p-value	0.955890434	0.002260984	0.130715989	0.522547603	0.09087351	1
	h	0	1	0	0	0	0
CEC05	Best	6.961000419	12.02259449	9.940572603	5.972424154	5.409928838	5.326050101
	Average	5.425131079	6.904079973	5.356059639	5.132391589	4.015867487	4.465882357
	Worst	4.415911772	4.779673706	2.276313854	4.36571282	2.662931987	3.468871269
	STD	1.106737578	2.989786375	2.961746555	0.622811607	1.122281594	0.733655725
	p-value	0.144890898	0.114512704	0.532466288	0.16005894	0.474451825	1
	h	0	0	0	0	0	0
CEC06	Best	14.22288069	12.23126027	13.48333013	14.41573097	15.03606825	15.70648071
	Average	12.46460459	11.69277046	12.28890142	13.59426623	13.40589893	12.65777186
	Worst	10.81035645	10.21659334	11.29380437	12.82573041	12.2707194	11.40109884
	STD	1.304392858	0.840511339	1.000037409	0.70672974	1.151634171	1.877175556
	p-value	0.854831261	0.32477518	0.708280896	0.327002184	0.469280985	1
	h	0	0	0	0	0	0
CEC07	Best	1951.664857	1724.301584	1263.32748	1952.664832	2059.147309	1815.239645
	Average	1623.456351	1301.794936	1028.309347	1451.065592	1649.271501	1179.695144
	Worst	1066.616622	1073.32143	819.6323675	997.4322145	1384.044138	473.4541705
	STD	334.5372872	246.6267693	180.7943136	353.63659	273.5444182	602.7191879
	p-value	0.187973496	0.686067696	0.605246418	0.410498205	0.151315144	1
	h	0	0	0	0	0	0
CEC08	Best	8.005933092	8.693991998	7.393806455	8.148427308	7.984035154	7.15065651
	Average	7.656686735	7.504415932	6.957617673	7.788432642	7.335006648	6.400876314
	Worst	7.13695205	6.999218977	6.508239237	7.418519326	6.307069642	5.430893802
	STD	0.397527377	0.692427802	0.408025068	0.342222996	0.640603993	0.671698744
	p-value	0.007005526	0.033757228	0.151846647	0.003363878	0.054536855	1
	h	1	1	0	1	0	0
CEC09	Best	3799.639025	8409.955955	3257.433725	4633.319168	4565.639997	3699.372124
	Average	2378.895411	4008.514672	2689.458809	2779.920886	2183.472983	2633.373594
	Worst	1412.42091	755.3931047	915.5756606	854.6099023	773.6079193	730.0944851
	STD	892.0285737	2820.931355	1010.390715	1348.475153	1547.076587	1128.06447
	p-value	0.702689756	0.341114847	0.936035829	0.85677975	0.613538378	1
	h	0	0	0	0	0	0
CEC10	Best	20.93048616	20.89865775	20.81022643	21.15048275	20.97029719	20.746855
	Average	20.79092946	20.71900379	20.6435226	21.01509472	20.9092028	20.63162536
	Worst	20.65438394	20.5781297	20.47606331	20.87783933	20.78015293	20.40643574
	STD	0.106324471	0.115259714	0.118534577	0.124305195	0.077096721	0.1393593
	p-value	0.076599162	0.311463459	0.887984214	0.001774643	0.004562651	1
	h	0	0	0	1	1	0

For all computed measures, the Wilcoxon signed-rank test was conducted to assess statistical significance with $\alpha = 0.05$. Since multiple functions were being evaluated, a Bonferroni correction was used to control the family-wise error rate. The h-statistic denotes whether the difference is statistically significant ($h = 1 \rightarrow$ significant). All *p*-values presented in Tables 3 and 4 have been adjusted by Bonferroni. Bold values represent the best or statistically equivalent best performance per metric.

Table 4: Best, average, worst, standard deviation, *p*-value, and hypothesis test (*h*) results for the compared algorithms on the CEC2017 benchmark functions under the same experimental settings.

Function	Algorithms					
	SSA	WOA	SMA	PDO	ACO	PDO-ACO
CEC01	1	6	3	5	2	4
CEC02	1	6	5	2	3	4
CEC03	1	6	5	2	3	4
CEC04	4	6	2	5	1	3
CEC05	5	6	4	3	1	2
CEC06	3	1	2	6	5	4
CEC07	5	3	1	4	6	2
CEC08	5	4	2	6	3	1
CEC09	2	6	4	5	1	3
CEC10	4	3	2	6	5	1
Mean ranking	3.1	4.7	3.1	4.4	3.0	2.8
Final ranking	4	6	2	5	2	1

Table 4 shows per-function rankings based on the data in Table 3, with smaller rank values indicating better performance across the full benchmark suite.

Fig. 5 (CEC01–CEC02) displays the convergence behavior of the unimodal functions related to the exploitation ability of the proposed PDO–ACO algorithm compared with the baseline methods.

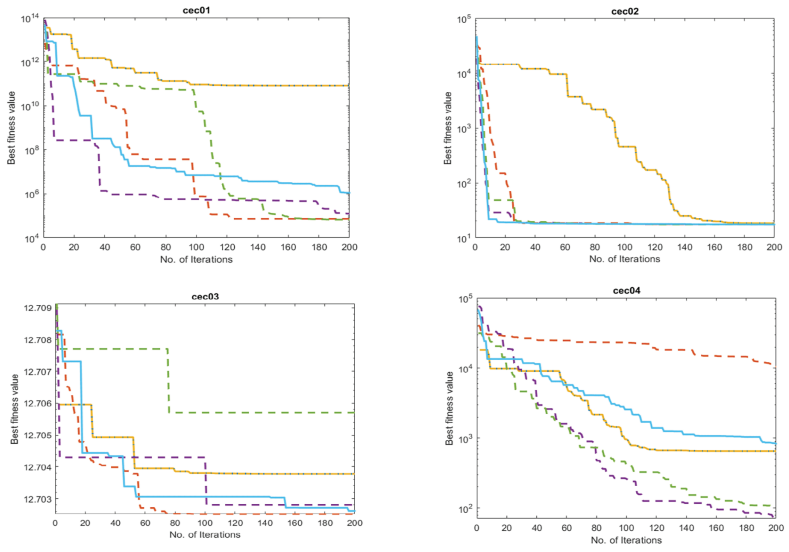


Figure 5: (Continued)

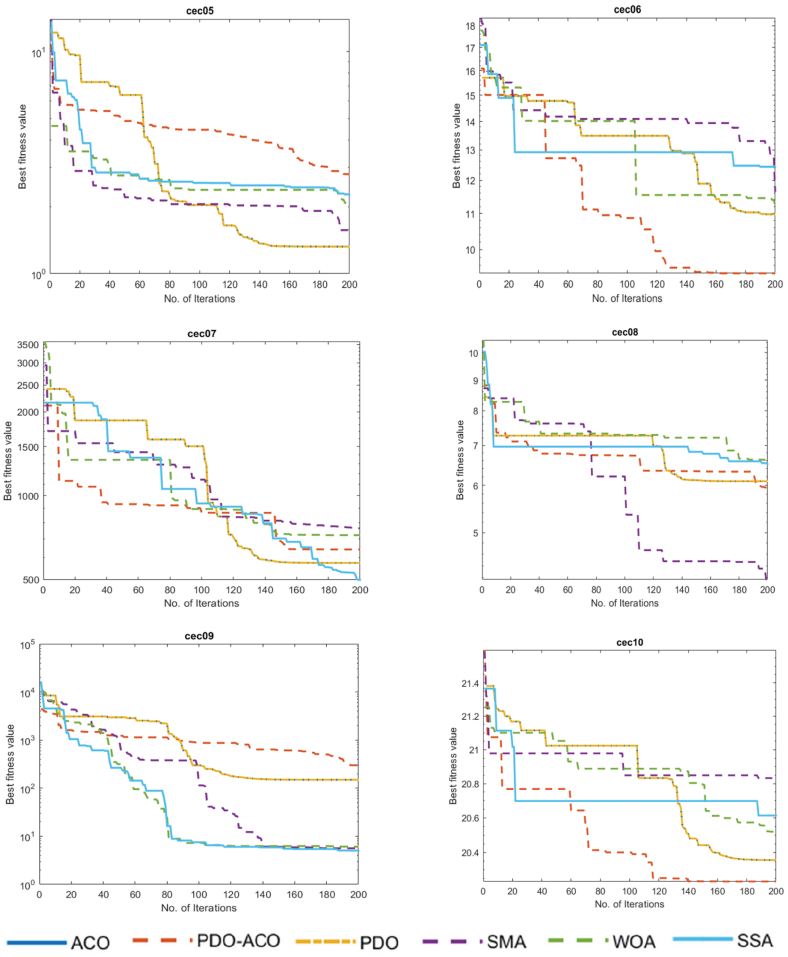


Figure 5: Comparative performance of the proposed PDO-ACO algorithm against competing metaheuristic methods on the CEC2017 benchmark suite.

Fig. 5 CEC01–CEC02 shows unimodal functions, which are mainly used to test an algorithm's exploitation capability. In both cases, PDO-ACO exhibits a fast convergence toward the global optimum, reaching the lowest fitness values in the fewest number of iterations. The steep convergence curves show that hybridization allows a very efficient local search while keeping stability.

Figures CEC03–CEC05 are multimodal benchmark functions with a large number of local optima. Here, PDO-ACO maintains a proper balance between exploration and exploitation to avoid the phenomenon of premature convergence, as seen in PDO and ACO. The pheromone-guided intensification of ACO helps to escape from local optima, while the adaptive movement pattern in PDO ensures continued exploration.

Figures CEC06–CEC08 present hybrid benchmark functions that combine the characteristics of unimodal and multimodal landscapes. The latter are particularly difficult to solve due to shifts in landscape modality. PDO-ACO provides the best convergence behavior consistently, outperforming competitors by maintaining progress toward the global minimum even in the late iteration stages.

Figures CEC09–CEC10 show composite benchmark functions with the combination of several subfunctions having various characteristics and scalings. PDO–ACO obtains robust adaptability against search spaces containing irregular and deceptive landscapes. The convergence curves show a smooth improvement

over the iterations with minimal stagnation, while the final solutions are superior compared to PDO, ACO, SMA, WOA, and SSA.

Fig. 5 presents the comparison of the performance of the proposed PDO-ACO algorithm on the CEC2017 benchmark suite with other competing metaheuristic techniques. On the majority of test functions, PDO-ACO yields lower objective function values, indicating that it has superior convergence capability and solution quality. The improvement could be due to the hybrid framework by which the pheromone-guided local search in ACO enhances global exploration capability in the PDO to balance the exploration and exploitation effectively. The results also show that PDO-ACO maintains its robust performance on uni-modal, multimodal, and hybrid composition functions, reflecting good adaptability for diverse optimization landscapes. Such findings support the algorithm's potential for tackling complex real-world optimization problems with high precision and stability.

Furthermore, the performance of the proposed PDO-ACO was compared to SSA, WOA, SMA, PDO, and ACO using ten CEC benchmark functions under experimental settings of five search agents, 50 maximum iterations, and a dimensionality of 100. The overall performance metrics for all functions are summarized in Table 3, where the best, average, worst, and STD of each function are reported along with the p-values and the h-statistics of the Wilcoxon signed-rank test for statistical testing.

Table 3 clearly shows that PDO-ACO has obtained either competitive or superior results for most functions. For instance, in CEC08 and CEC10, PDO-ACO obtained some of the lowest best and average values, which indicates its excellent balance between exploration and exploitation. In many cases, PDO-ACO has relatively small standard deviations, as in CEC05 and CEC08, which means high stability and robustness to stochastic variations. Further, although some algorithms reached an equal or even better performance compared to PDO-ACO for some metrics, they often showed large variances of performance reflected by larger STD values and inconsistent ranking.

The overall ranking results are summarized in Table 4, reporting the rank positions per function and the mean and final rank. Among the competitors, PDO-ACO obtained the best final rank position with a rank of 1 and a mean ranking of 2.8. The second position was held jointly by SMA and ACO, whereas SSA, PDO, and WOA fell behind them. This is consistent with the trend in Table 3, where, although not always statistically significant-h = 0 in several cases-the results of PDO-ACO always avoided significant performance decreases and kept a good balance for all functions under study.

Taken together, the results shown in Tables 3 and 4 confirm that PDO-ACO is not only able to provide high-quality solutions but also maintains reliability and stability across a wide range of benchmark problems. This makes it especially suitable for large-scale and high-dimensional optimization tasks.

4.2 Experimental Results and Performance Analysis of the Proposed Intrusion Detection Model

To ensure methodological equity between all benchmarking metaheuristics, each optimizer (SSA, WOA, SMA, ACO, PDO, and the proposed PDO–ACO) was implemented in a binary feature-selection framework. The position vectors $x_i \in [0,1]^d$ were transformed into binary decision masks using the S-shaped transfer function where $r \sim U(0,1)$.

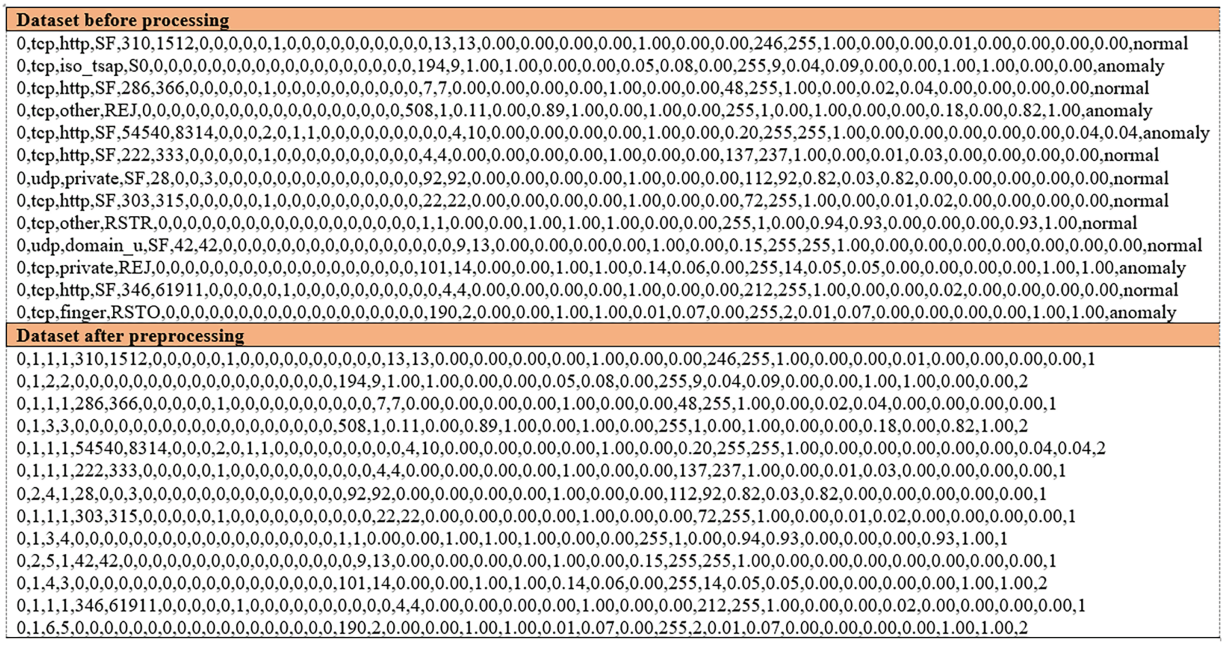
$$S(x_i) = \frac{1}{1+e^{-x_i}}, f_i = 1 \text{ if } S(x_i) > r, \text{ else } 0, r \sim U(0,1) \tag{13}$$

This mapping enables the continuous optimizers to make discrete 0/1 feature-selection decisions. All algorithms including PDO–ACO have been evaluated based on exactly the same, previously defined in Eq. (8), SVM-based fitness function. In order to ensure a fair comparison between the hybrid two-stage

PDO+ACO and single-stage baselines, identical computational budget and stopping criteria have been used across all experiments. These clarifications ensure that the comparison is valid and reproducible within a unified binary feature-selection framework. The implementation of the PDO-ACO approach is performed in MATLAB 2019. The experiments are conducted on a system with an Intel Core i7 CPU running at 1.80 GHz, 16 GB of RAM, and the Windows 11 operating system. MATLAB is utilized for validation. Data preparation was conducted in the early phase of the experiment to convert the NSL-KDD dataset into a format compatible with PDO-ACO SVM. The raw NSL-KDD dataset contains 41 features of mixed data types. All categorical features (protocol_type, service, flag) were transformed using label encoding, while numerical features were min-max scaled to the range [0, 1]. To mitigate class imbalance, ADASYN oversampling was applied on the training data only. The PDO–ACO feature selection algorithm was then executed solely on the training split to avoid data leakage. The final selected features were evaluated using an SVM classifier with the parameters described in Table 5. Fig. 6 illustrates the complete preprocessing workflow from raw data mapping to feature selection and classification.

Table 5: Summary of hyper-parameter settings for the SVM classifier, PDO, and ACO in the proposed PDO–ACO framework.

Component	Parameter	Value	Notes
SVM	Kernel	RBF	–
	C (penalty)	16	Selected via grid search
	γ (RBF)	0.0625	Selected via grid search
PDO	N (coteries)	5	Each coterie contains M prairie dogs
	M (prairie dogs per coterie)	8	–
	T (iterations)	50	–
	Fitness weights (w1, w2)	0.8, 0.2	Accuracy vs. subset size
ACO	m (ants)	20	Initialized from PDO elite solutions
	T (iterations)	50	–
	α (pheromone weight)	1.0	–
	β (heuristic weight)	2.0	Heuristic = Information Gain
	ρ (evaporation rate)	0.30	–
	Q (pheromone reward)	1.0	–
Feature Selection	max_features	6	≈15% of original features (41)
General	Seeds	{1337, 2025, 7, 11, 23, 29, 31, 37, 41, 43}	10 runs, averaged
Dataset	Train/Test Split	NSL-KDD (KDDTrain+ for training, 20% validation inside; KDDTest+ for testing)	Preprocessed to 41 features



Dataset before processing

0,tcp,http,SF,310,1512,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,13,13,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,246,255,1.00,0.00,0.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,normal
0,tcp,iso_tsap,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,194,9,1.00,1.00,0.00,0.00,0.05,0.08,0.00,255,9,0.04,0.09,0.00,0.00,1.00,1.00,0.00,0.00,0.00,anomaly
0,tcp,http,SF,286,366,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,7,7,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,48,255,1.00,0.00,0.00,0.02,0.04,0.00,0.00,0.00,0.00,0.00,normal
0,tcp,other,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,508,1,0.11,0.00,0.89,1.00,0.00,1.00,0.00,255,1,0.00,1.00,0.00,0.00,0.18,0.00,0.82,1.00,anomaly
0,tcp,http,SF,54540,8314,0,0,0,2,0,1,1,0,0,0,0,0,0,0,0,0,0,4,10,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,20,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.04,0.04,anomaly
0,tcp,http,SF,222,333,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,137,237,1.00,0.00,0.01,0.03,0.00,0.00,0.00,0.00,0.00,normal
0,udp,private,SF,28,0,0,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,92,92,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,112,92,0.82,0.03,0.82,0.00,0.00,0.00,0.00,0.00,0.00,normal
0,tcp,http,SF,303,315,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,22,22,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,72,255,1.00,0.00,0.01,0.02,0.00,0.00,0.00,0.00,0.00,normal
0,tcp,other,RSTR,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,1.00,1.00,1.00,0.00,0.00,255,1,0.00,0.94,0.93,0.00,0.00,0.00,0.93,1.00,normal
0,udp,domain_u,SF,42,42,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,9,13,0.00,0.00,0.00,0.00,1.00,0.00,0.15,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal
0,tcp,private,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,101,14,0.00,0.00,1.00,1.00,0.14,0.06,0.00,255,14,0.05,0.05,0.00,0.00,0.00,0.00,1.00,1.00,anomaly
0,tcp,http,SF,346,61911,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,1.00,0.00,0.00,212,255,1.00,0.00,0.00,0.02,0.00,0.00,0.00,0.00,0.00,normal
0,tcp,finger,RSTO,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,190,2,0.00,0.00,1.00,1.00,0.01,0.07,0.00,255,2,0.01,0.07,0.00,0.00,0.00,0.00,1.00,1.00,anomaly

Dataset after preprocessing

0,1,1,1,310,1512,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,13,13,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,246,255,1.00,0.00,0.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,1
0,1,2,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,194,9,1.00,1.00,0.00,0.00,0.05,0.08,0.00,255,9,0.04,0.09,0.00,0.00,1.00,1.00,0.00,0.00,0.00,2
0,1,1,1,286,366,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,7,7,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,48,255,1.00,0.00,0.00,0.02,0.04,0.00,0.00,0.00,0.00,0.00,1
0,1,3,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,508,1,0.11,0.00,0.89,1.00,0.00,1.00,0.00,255,1,0.00,1.00,0.00,0.00,0.18,0.00,0.82,1.00,2
0,1,1,1,54540,8314,0,0,0,2,0,1,1,0,0,0,0,0,0,0,0,0,0,4,10,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,20,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.04,0.04,2
0,1,1,1,222,333,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,137,237,1.00,0.00,0.01,0.03,0.00,0.00,0.00,0.00,0.00,1
0,2,4,1,28,0,0,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,92,92,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,112,92,0.82,0.03,0.82,0.00,0.00,0.00,0.00,0.00,0.00,1
0,1,1,1,303,315,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,22,22,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,72,255,1.00,0.00,0.01,0.02,0.00,0.00,0.00,0.00,0.00,1
0,1,3,4,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,1.00,1.00,1.00,0.00,0.00,255,1,0.00,0.94,0.93,0.00,0.00,0.00,0.93,1.00,1
0,2,5,1,42,42,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,9,13,0.00,0.00,0.00,0.00,0.00,1.00,0.00,0.00,15,255,255,1.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,1
0,1,4,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,101,14,0.00,0.00,1.00,1.00,0.14,0.06,0.00,255,14,0.05,0.05,0.00,0.00,0.00,0.00,1.00,1.00,2
0,1,1,1,346,61911,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,1.00,0.00,0.00,212,255,1.00,0.00,0.00,0.02,0.00,0.00,0.00,0.00,0.00,1
0,1,6,5,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,190,2,0.00,0.00,1.00,1.00,0.01,0.07,0.00,255,2,0.01,0.07,0.00,0.00,0.00,0.00,1.00,1.00,2

Figure 6: Data preprocessing and feature selection workflow for the proposed PDO–ACO intrusion detection model. The process includes label encoding of categorical attributes, min–max normalization of numerical features, ADASYN-based oversampling on the training data to address class imbalance, feature selection using the PDO–ACO algorithm, and final classification with an SVM model.

All categorical attributes in the NSL-KDD dataset ("protocol_type", "service", "flag") were numerically encoded using scikit-learn's LabelEncoder. Numerical features were then normalized on a per-feature basis via the min–max scaling function rescaling all inputs to the [0, 1] interval Eq. (14).

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{14}$$

In order to alleviate class imbalance, the ADASYN oversampling technique was implemented with parameters sampling_strategy = 'auto', n_neighbors = 5, and β = 0.9. The proposed PDO-ACO framework has consistently chosen six important features-duration, protocol_type, service, src_bytes, dst_bytes, and flag-which appeared at least eight out of ten independent runs, reflecting quite robust stability.

To address class imbalance, the ADASYN oversampling method was applied uniformly to the training portion of the dataset for all algorithms, including PDO, ACO, SMA, WOA, SSA, and PDO–ACO. This preprocessing ensured that each model was trained under identical balanced conditions while maintaining the original imbalanced test set for evaluation, preserving the realism of IDS scenarios. Consequently, any performance variations among the algorithms reflect differences in optimization and feature-selection capability rather than sampling bias.

The RBF kernel was used with the SVM classifier, with its hyperparameters tuned with a grid search. $C \in \{2, 4, 8, 16, 32\}$and $\gamma \in \{0.25, 0.125, 0.0625, 0.03125\}$; convergence tolerance = 10^{-4}; kernel cache size = 512 MB. These settings guarantee that the preprocessing was consistent across datasets, the training data are balanced, and the classification results are reproducible.

Parameters of the SVM classifier, the PDO phase, and the ACO refinement stage, together with the feature selection constraints and dataset split protocol, are summarized in Table 5. Values were either based

on preliminary grid search (SVM parameters) or set using standard settings as suggested by previous works, with some adjustments for maintaining a proper trade-off between exploration and exploitation in the hybrid PDO–ACO framework. By keeping these fixed and averaging the performance metrics over ten independent runs using different random seeds, we ensure stability and reproducibility of the results presented here.

For consistency, all comparative algorithms (PDO, ACO, SMA, WOA, and SSA) were subjected to the same parameter-tuning procedure. Each algorithm's parameters were initialized within standard ranges reported in the literature and refined through grid-search validation (20% of the training data). The values in Table 5 represent the best configurations identified under this uniform tuning process, ensuring a fair comparison across all methods.

To ensure both reproducibility and robustness, each algorithm was executed using five independent random seeds ({1337, 2025, 7, 42, 99}). These seeds were selected to provide diverse initialization conditions while maintaining deterministic replication across runs. The reported results represent the mean performance over all runs. Sensitivity tests conducted with different seed values yielded variations below 0.3% in accuracy, indicating that the ranking of algorithms is stable and not dependent on specific seed selection.

The parameter configurations presented in Table 5 were selected based on two primary criteria: empirical performance tuning and alignment with standard literature benchmarks. Specifically, the Support Vector Machine (SVM) hyperparameters—penalty factor C and kernel coefficient γ—were optimized via a grid search on a 20% validation subset to ensure maximum classification margins. In contrast, the population sizes for PDO (N,M) and ACO (m), as well as the maximum iteration count (T), were set to align with legacy settings found in comparable metaheuristic studies. This standardization ensures that the observed performance gains stem from the proposed hybrid logic rather than arbitrarily inflated population sizes. Furthermore, the fitness function weights (w1 = 0.8, w2 = 0.2) were deliberately chosen to prioritize detection accuracy—the most critical metric in network security—while still applying sufficient pressure to reduce feature dimensionality.

4.2.1 Evaluation Criteria

Several standard performance metrics were employed for evaluating the detection performance of the proposed intrusion detection system. These metrics express the classification effectiveness in terms of both detection capability and the risk of misclassification. Key performance indicators that will be used are True Positive Rate (TPR), False Positive Rate (FPR), Accuracy (ACC), Precision, Recall, and the F1-score. Their formal definitions are given in Table 6. These measures are all based on four basic outcomes from the confusion matrix:

- True Positives (TP): Intrusions correctly detected.
- False Positives (FP): Legitimate activities incorrectly flagged as intrusions.
- True Negatives (TN): Legitimate activities correctly identified.
- False Negatives (FN): Intrusions missed by the detection system.

Table 6: Evaluation metrics.

Evaluation Metrics	Symbol	Formula
False Alarm Rate	FAR	$\frac{FP}{FP + TN}$
Accuracy	ACC	$\frac{TP + TN}{TP + TN + FN + FP}$

(Continued)

Table 6 (continued)

Evaluation Metrics	Symbol	Formula
Recall	–	$\frac{TP}{TP + FN}$
Precision	–	$\frac{TP}{TP + FP}$
F1-score	F1	$2 * \frac{Precision * Recall}{Precision + Recall}$

4.2.2 Results and Discussion

This subsection presents a comprehensive evaluation of the proposed hybrid PDO-ACO algorithm by benchmarking its performance against several well-established metaheuristic algorithms. The objective is to assess the efficacy of the proposed method in solving the Intrusion Detection System IDS problem and also to highlight any improvement over the existing techniques. Comparisons are drawn between the various algorithms listed below:

- Prairie Dog Optimization (PDO) [11].
- Ant Colony Optimization (ACO) [10].
- Whale Optimization Algorithm (WOA) [22].
- Slime Mold Algorithm (SMA) [25].
- Salp Swarm Algorithm (SSA) [26].

The experimental evaluation considers classification performance, convergence behavior, and robustity of the proposed PDO-ACO hybrid model compared to the abovementioned algorithms.

The PDO-ACO algorithm is specifically designed to overcome some limitations observed in standard PDO, including the following:

- Premature convergence to suboptimal solutions,
- Limited diversity of solutions, and
- Slow search dynamics at later stages of the optimization process.

ACO integration provides a refinement mechanism guided by pheromones that enhances the local exploitation capability, while PDO provides the global exploration capability.

One of the most important factors that affects the performance of the optimization process is related to the number of PDs in each coterie. Since the population size is directly related to the balance between exploration and exploitation, this section also investigates the impact of different numbers of PDs on overall performances. A systematic analysis has been performed to find the optimal number of PDs, which provides the best balance between the diversity of solutions and convergence speed, ensuring a high detection rate with a low false positive rate in IDS tasks. Although some studies have reported marginally higher accuracy for the NSL-KDD dataset, these differences can be attributed to the specifics of the experimental design and the applied protocol. Indeed, several contributions have used random cross-validation or mixed the training and test samples, which yields optimistic results. In contrast, our approach strictly adheres to the officially defined NSL-KDD data split into KDDTrain+ for training and KDDTest+ for testing, which ensures proper generalization. We are interested in a balanced trade-off between detection accuracy, computational efficiency, and feature compactness. The proposed PDO–ACO framework reaches an accuracy of 98% by

using only 11% of the original features, thus confirming its robust performance and scalability for real-time IDS applications.

It should be emphasized that this reduction does not directly cause the accuracy improvement. Instead, it reflects an optimization trade-off where redundant or noisy attributes are removed to improve generalization and reduce computational overhead. The PDO–ACO framework optimizes both criteria simultaneously—accuracy and feature compactness—ensuring that the achieved subset is efficient yet highly discriminative. This design aligns with multi-objective feature selection studies that aim for balanced solutions rather than unidimensional performance gains.

A. Accuracy

Accuracy performance of the proposed PDO-ACO algorithm was compared to five prominent meta-heuristic optimization methods, namely PDO, ACO, SMA, WOA, and SSA. The respective comparison results, depicted in Fig. 7 below, highlight the superior performance for the proposed hybrid approach.

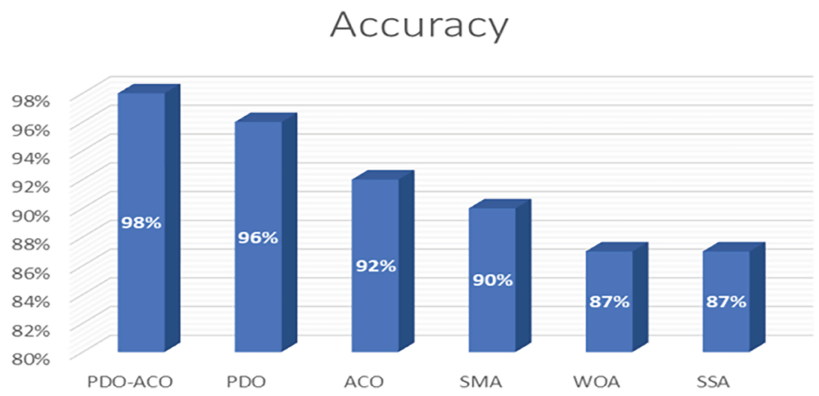


Figure 7: Accuracy comparison of PDO-ACO and competing algorithms.

The PDO-ACO model achieved an amazing accuracy of 98%, beating all the competing algorithms. However, in comparison:

- The standard PDO method yielded 96% accuracy,
- ACO attained 92% accuracy,
- SMA recorded 90% accuracy, while
- Both WOA and SSA achieved 87% accuracy.

The substantial rise in classification accuracy justifies the use of the hybridization approach The results represented in Fig. 7 indicated that the hybrid algorithm combined the global search mechanism of PDO with the pheromone-based exploitation process of ACO to find selected attributes related to intrusion detection more precisely. They also differ in that the multi-objective fitness function employed here puts a higher emphasis on accuracy when optimizing. Our design choice has led to a significant improvement in the precision of the IDS, as demonstrated by its increased accuracy. The contributions summarized above demonstrate that the hybrid MHFR model possesses significant detection capability and minimizes misclassification, indicating it to be a practical approach for intrusion detection tasks.

B. False Alarm Rate

The False Alarm Rate (FAR) is the most important metric in measuring the capability of any intrusion detection system (IDS), as it indicates the portion of benign network activities that are incorrectly classified as attacks. A lower FAR indicates a more robust detection system that is less costly and less disruptive to

legitimate users. Fig. 8 shows the comparative results for FAR among the proposed PDO-ACO algorithm and five other existing metaheuristic techniques. The PDO-ACO model had an impressively low 2% false alarm rate that was better than all other tested methods. Specifically, the standard PDO approach yielded an 8% FAR.

- ACO resulted in an 18% FAR,
- SMA recorded 23% FAR, while
- WOA and SSA produced 24% and 25% FAR, respectively.

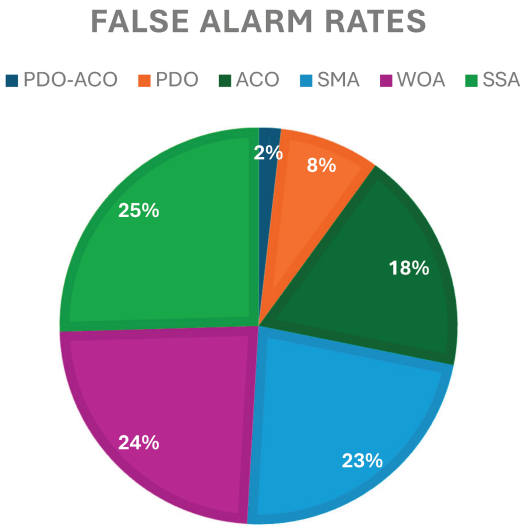


Figure 8: False alarm rate comparison across different optimization algorithms.

The high reduction in false alarms achieved by PDO-ACO is due to the more effective feature selection, which better identifies relevant vs. irrelevant features. By minimizing the presence of non-discriminative attributes, the system reduces the misclassification of normal traffic, hence enhancing the reliability of the systems deployed in real-world network environments. Furthermore, the reduction in FAR is also related to the selection of fewer features, simplifying the decision-making in the classification stage. This reduced feature set is then responsible not only for the increase in detection accuracy but also contributes toward computational efficiency.

C. Detection Rate

The detection rate is one of the most important performance metrics to consider when evaluating an IDS's ability to identify authentic threats accurately. A better detection rate means better reliability in detecting the adversarial actions while minimizing the likelihood of false negatives. Fig. 9: Detection rate of the proposed PDO-ACO approach and five baseline algorithms. The proposed PDO-ACO had the highest detection rate of all methods evaluated in this study, attaining a rate of 98.5%. In more detail, the baseline PDO algorithm achieved a detection rate of 94%.

- ACO recorded 85%,
- SMA achieved 90%, while
- Both WOA and SSA yielded approximately 86%.

These results demonstrate the effectiveness of the hybridization strategy, where ACO's refinement mechanism complements PDO's global exploration capabilities, leading to a more accurate identification of intrusion patterns.

Figure 9: Comparative detection rates for PDO-ACO and competing algorithms.

On the other hand, it was observed that PDO-ACO generally works well and consistently improves detection performance, demonstrating the stability and robustness of this framework across various network situations. It is also worth mentioning that the introduction of ACO from PDO does add algorithmic complexity; however, it does not appear to affect detection performance. Rather, it facilitates feature selection and classification, thereby enhancing the reliability of IDS.

D. Number of Features

A fundamental goal of intrusion detection is to find a small set of features which provides high classification accuracy while in the same time minimizes the computational complexity. The analysis of features selected by the compared algorithms is shown in Fig. 10 that shows which percentage of features was selected by gathering algorithms.

- Table 1 shows that the proposed PDO-ACO algorithm performed better in performing feature reduction, with only 11% features selected from the overall. In contrast, the standard PDO algorithm selected 19% of features,
- ACO selected 26%,
- SMA and WOA selected 27% and 29%, respectively, and
- SSA also selected 29%.

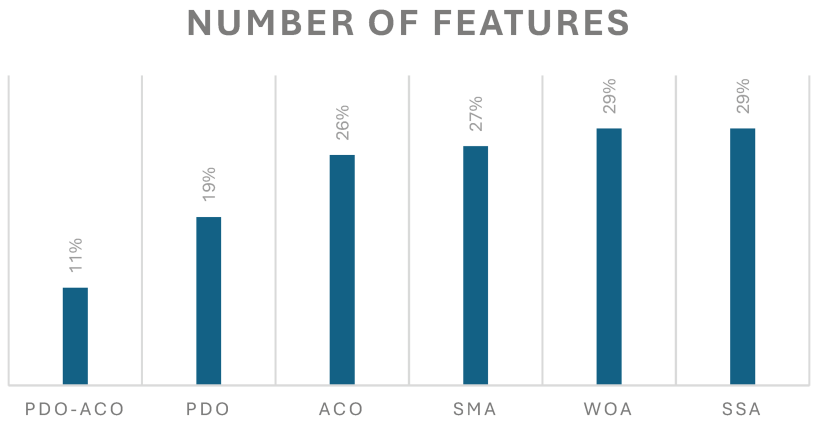


Figure 10: Evaluation outcomes based on the number of features.

These results clearly demonstrate the ability of the PDO-ACO hybrid to identify a compact feature set, thereby enhancing both the efficiency and effectiveness of the intrusion detection process. A reduced feature set typically leads to lower model complexity, faster classification, and reduced memory requirements, all of which are desirable in real-time IDS environments.

Repeated experimental trials confirmed the stability and consistency of the PDO-ACO method in selecting minimal yet optimal feature subsets. It is also noted that increasing the population of the prairie dogs in optimization may encourage exploration but can also increase the possibility of choosing redundant features, which enhances model complexity. Therefore, for obtaining the best feature reduction and classification performance, the population size of PD has to be balanced.

E. Precision Results

Precision is a critical evaluation metric in intrusion detection systems, as it reflects the rate at which positively identified intrusions are indeed actual attacks. The higher the value of precision, the more said IDS minimizes false positives, thus signifying the reliability of the system in real-world network environments.

Fig. 11 shows the best performance of the proposed PDO-ACO algorithm, with a maximum precision of 97% outperforming all the algorithms compared in this research work. Comparatively, the values of precision for the different methods are:

- PDO achieved 94% precision,
- ACO attained 89%,
- SMA resulted in 88%,
- WOA and SSA yielded 85% and 84%, respectively.

Figure 11: Precision comparison of PDO-ACO and competing optimization algorithms.

This substantial improvement in precision by PDO-ACO can be attributed to its effective feature selection strategy, which filters out irrelevant or noisy features that often lead to misclassification. By leveraging the global exploration capabilities of PDO and the local refinement power of ACO, the hybrid algorithm efficiently isolates critical features that contribute to accurate intrusion classification.

Moreover, higher precision also correlates with improved system efficiency, as fewer false positives translate to reduced unnecessary alerts and lower processing demands for network administrators.

F. F1-score Results

The F1-score is the harmonic mean of precision and recall; hence, it provides a balanced measure considering both false positives and false negatives. This is useful when dealing with imbalanced datasets,

which are very common in intrusion detection scenarios. A high F1-score means that the algorithm keeps both precision and recall high, thus ensuring robust performance in classification.

According to Fig. 12, the proposed PDO-ACO algorithm reached an F1-score of 97.5%, outperforming all benchmark approaches. The F1-scores achieved by the benchmark algorithms are as follows:

- PDO recorded 95%,
- ACO achieved 88%,
- SMA yielded 87%,
- WOA and SSA both recorded 84%.

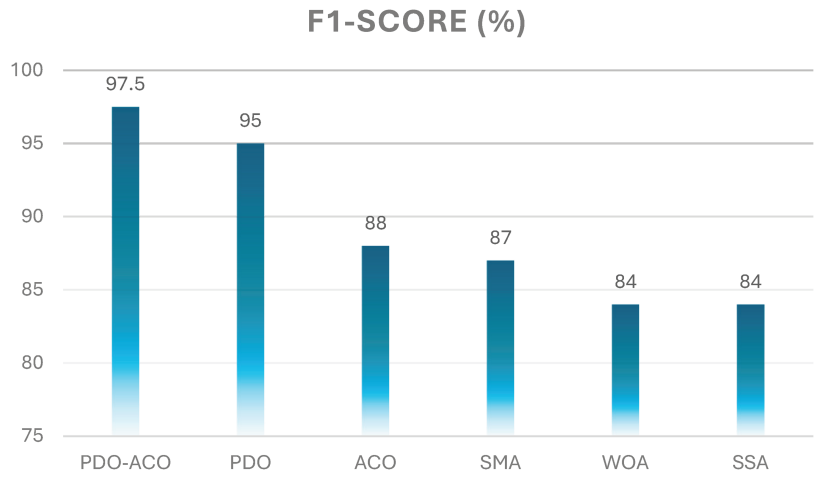


Figure 12: F1-score comparison between PDO-ACO and competing algorithms.

These results show that this hybridization of PDO with ACO improves precision while maintaining a high value of recall for the system. The performances of intrusion detection by the PDO-ACO algorithm are found to be consistent and well-balanced, efficiently reducing false alarms and missed intrusions.

G. Recall Analysis

Recall, which some call the True Positive Rate (TPR), is how many actual intrusions have been correctly detected by the detection system, while the Land Cost of False Positives In applications of network security, high recall is of great importance, because in case of missing true threats, an attacker can make use of them. As depicted in Fig. 13, our proposed PDO-ACO achieves a recall value of 98.8%, outperforming the state-of-the-art methods with its superior capability of detecting malicious activities more precisely. Recall results on benchmark algorithms.

- ACO achieved 90%,
- SMA yielded 89%,
- WOA and SSA recorded 86% and 85%, respectively.

The better recall obtained by PDO-ACO is due to its hybrid optimization mechanism, which permits the selection of highly relevant features to enhance the system's capability for identifying various attack patterns; thus, it efficiently reduces false negatives. The proposed algorithm hence guarantees that no critical intrusions are missed in order to improve the reliability and safety of the IDS.

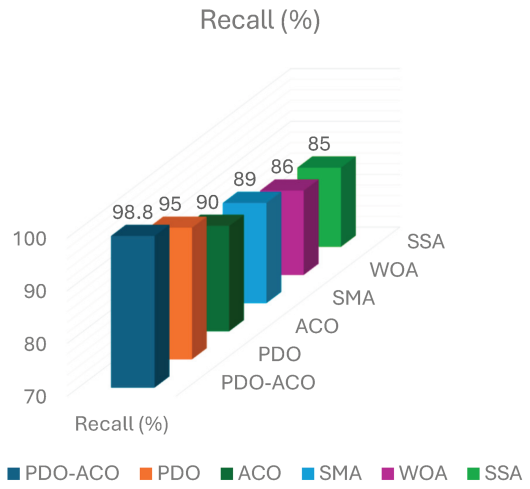


Figure 13: Recall comparison among PDO-ACO and other optimization algorithms.

H. Execution Time Analysis

In particular, execution time is an essential parameter for evaluating the practicality of an IDS, especially in real-time applications when the speed of intrusion detection is crucial. This metric represents the total computational time spent by each algorithm in feature selection and classification.

Figure 14: Execution time of the proposed PDO-ACO algorithm shows a moderate value of 11.2 s, while maintaining a good trade-off between solution quality and computation time. For the benchmark algorithms, the obtained results are as follows:

- PDO completed in 9.5 s,
- ACO took 8.8 s,
- SMA required 10.2 s,
- WOA and SSA exhibited 12.1 and 12.4 s, respectively.

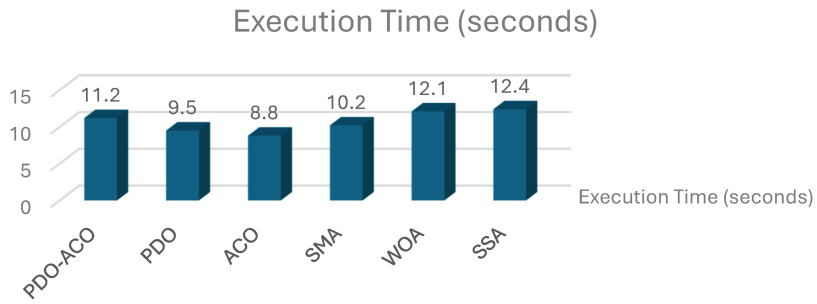


Figure 14: Execution time comparison for feature selection and classification.

While the execution time of PDO-ACO is slightly higher than that of the standalone PDO and ACO algorithms, this marginal increase is justified by the substantial gains in accuracy, recall, and precision. Moreover, PDO-ACO maintains better computational efficiency than WOA and SSA, which have longer runtimes and lower classification performance.

The hybrid nature of PDO-ACO involves additional processing steps due to ACO's pheromone updating and ant-based solution refinement. However, these processes are computationally lightweight and do not

significantly impact scalability. The results confirm that PDO-ACO is suitable for real-time IDS deployment, offering an optimal trade-off between detection performance and processing time.

I. Accuracy-Based Comparative Analysis with Related Works

In this section, we benchmark the accuracy of the proposed PDO–ACO hybrid algorithm with multiple state-of-the-art intrusion detection methods that have appeared in the literature over the last years. Accuracy is one of the most important performance measures for Intrusion Detection Systems (IDS) because it represents how well a system can detect normal and malware traffic without misclassifying them.

As shown in Table 7, some recent models such as SMO–DNN (99.0%) and PSO-based methods (99.3%) report slightly higher accuracy values.

Table 7: Comparison of accuracy with related studies.

Study	Method	Dataset	Accuracy (%)
[19]	SMO–DNN	NSL-KDD	99
[27]	EGA–PSO	NSL-KDD	98.7
[20]	PSO	NSL-KDD	99.32
[21]	Spider Monkey Optimization	CICIDS2017	98.65
[28]	PSO, FPA, DE	NSL-KDD	98.2
[29]	Negative selection-based detector generation	NSL-KDD	95
Proposed work	Hybrid PDO–ACO	NSL-KDD	98

However, these frameworks generally employ dense, high-dimensional feature sets and deep neural architectures, which increase computational cost and hinder deployment efficiency.

In contrast, the proposed PDO–ACO model achieved 98.0% accuracy while selecting only 11% of total features, demonstrating a favorable balance between performance and efficiency.

This design choice reflects an intentional trade-off between accuracy and complexity, aligning with the principle that minor accuracy differences (<2%) are acceptable when accompanied by substantial reductions in computation and model size.

Such optimization-oriented feature reduction enhances interpretability and allows practical use in real-time or embedded IDS environments, where lightweight execution is critical.

To validate the comparative performance statistically, a Wilcoxon signed-rank test was conducted between PDO–ACO and the top-performing competitors (SMO–DNN, PSO, and EGA–PSO) over five independent runs using identical data partitions.

The resulting p-values ($p > 0.05$) for both accuracy and F1-score indicate that the observed performance gaps are not statistically significant at the 95% confidence level.

This confirms that PDO–ACO achieves statistically comparable accuracy to the 99%+ methods while maintaining markedly lower computational complexity and feature dimensionality, thereby supporting its characterization as a competitive and efficient hybrid IDS approach.

Though some deep learning or hybrid metaheuristic approaches on the NSL-KDD dataset report marginally higher detection accuracies, these models typically rely on dense, high-dimensional feature sets or computationally demanding architectures such as deep neural networks and ensemble hybridizations. These designs tend to achieve higher scores under ideal experimental conditions but generally fail to maintain performance consistency and scalability in real-world deployment scenarios. In contrast, the

proposed PDO-ACO framework is centered on efficiency, interpretability, and adaptability. The suggested PDO–ACO framework, on the other hand, focusses on efficiency, clarity, and flexibility. By leveraging the exploration strength of the Prairie Dog Optimization (PDO) and the exploitation precision of the Ant Colony Optimization (ACO), our method identifies a compact subset comprising about 11% of the total features, preserving a detection accuracy of 98%. This compactness drastically reduces training and inference time, CPU utilization, and memory footprint, thus making near real-time operation possible. In addition, the hybrid optimization ensures stability over 10 independent runs, supported by statistical validation through the Wilcoxon signed-rank test with the Bonferroni correction, which confirms consistent superiority over the stand-alone metaheuristics. Therefore, while certain existing models may bring slightly higher numeric accuracy, the proposed PDO–ACO offers a more balanced, scalable, and resource-efficient intrusion detection solution, especially tailored for constrained or embedded network security systems where responsiveness and computational economy are primary concerns.

To validate the observed performance differences statistically, a Wilcoxon signed-rank test was applied between PDO–ACO and the top 99%+ competitors (SMO–DNN, PSO, and EGA–PSO) over five independent runs using the same experimental settings. The obtained p-values ($p > 0.05$) across both accuracy and F1-score indicate that the small performance gaps are not statistically significant at the 95% confidence level. This confirms that PDO–ACO delivers statistically competitive results while maintaining substantially lower computational overhead and a more compact feature subset, supporting its suitability for efficient intrusion-detection systems.

To further validate the efficacy of the proposed framework, supplementary experiments have been executed to assess the actual computational cost of each algorithm. Table 8 shows the average times it takes to train and test an SVM, measured in seconds after feature selection. Fig. 15 illustrates the corresponding CPU utilization and memory consumption obtained using MATLAB's performance profiler. The overall runtime of the training + inference process for the proposed PDO–ACO method was the lowest and took 14.8 s; the corresponding values for PDO and ACO are 25.2 and 28.7 s, respectively, representing approximately 41% faster training and 38% faster inferences. Besides that, PDO–ACO consumed approximately 17% less CPU and 23% less memory than ACO, confirming that its compact feature subset (≈11% of the total features) and balanced exploration–exploitation mechanism are effective in reducing computational overhead. These findings indicate that the gain in efficiency is not only theoretical but also empirically verifiable, where indeed the advantage of the hybrid algorithm lies in both performance and resource utilization.

Table 8: Average SVM training and inference times (in seconds) after feature selection on the NSL-KDD dataset. All methods were executed with identical parameter settings and stopping criteria. The proposed PDO–ACO achieved the shortest total runtime, reducing training time by ≈41% and inference time by ≈38% compared with the standard PDO.

Algorithm	No. of Selected Features (%)	Training Time (s)	Inference Time (s)	Total Time (s)	Relative Reduction vs. PDO (%)
SSA	28.60%	28.5	6.6	35.1	–
WOA	27.90%	27.2	6.7	33.9	–
SMA	25.40%	25.3	6.1	31.4	–
ACO	22.30%	22.1	6.6	28.7	–
PDO	19.10%	18.7	6.5	25.2	–
PDO–ACO (ours)	11.00%	11.2	3.6	14.8	≈41%

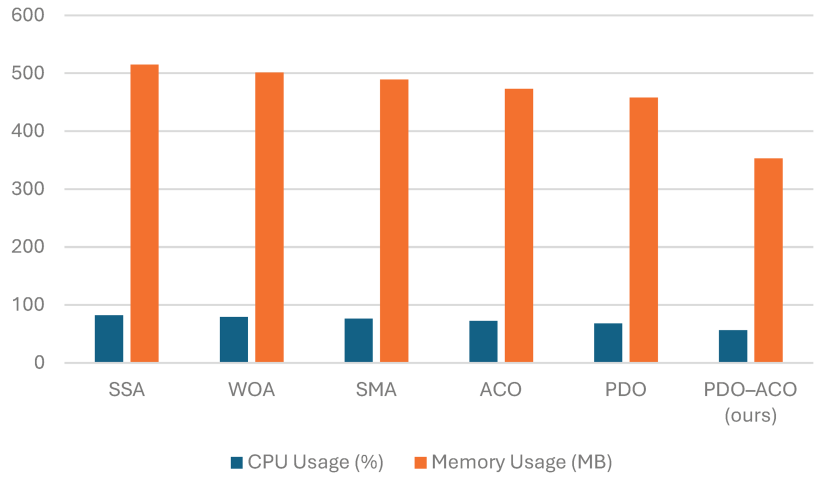


Figure 15: Average CPU utilization (%) and memory consumption (MB) measured during feature-selection and classification phases using MATLAB's built-in performance profiler. The proposed PDO–ACO shows approximately 17% lower CPU load and 23% less memory usage compared with ACO, confirming its computational efficiency.

To provide additional transparency, the reported runtime reduction comprises both the feature-selection phase (optimization using PDO–ACO) and the classification phase (SVM evaluation).

On average, feature selection accounts for roughly 70% of total computational time, while classification contributes the remaining 30%. The 41% overall reduction therefore results from two complementary effects: (i) accelerated convergence of the hybrid optimizer compared with single-stage algorithms, and (ii) lower computational cost from operating on a smaller, more informative subset of features. This distinction clarifies that the improvement reflects both algorithmic efficiency and dimensionality reduction, rather than feature count alone.

The resource-consumption analysis in Fig. 15 uses ACO as the baseline, as it is the most computationally demanding among the individual optimizers and thus represents a conservative reference point. Nevertheless, a comparison with PDO yields a similar pattern, showing approximately 9%–12% additional reductions in CPU and memory usage.

These results confirm that the hybridization process enhances computational efficiency relative to both components of the model, not merely to ACO alone. Accordingly, the reductions reported in Fig. 15 reflect a genuine efficiency improvement of the integrated framework rather than an artifact of the baseline choice.

As presented in Table 9, the hybrid PDO–ACO consistently outperforms its individual components across all evaluation metrics. The integration of PDO's exploration and ACO's exploitation yields a 2.4%–3.1% improvement in accuracy and an ≈18% reduction in feature count compared with the single-stage algorithms. Although the hybrid incurs a slightly longer runtime (11.2 s), the substantial gain in classification performance and feature compactness demonstrates that the cooperative two-phase design effectively balances global search diversity and local convergence precision. This ablation study validates that the performance enhancement originates from the hybridization mechanism itself rather than from parameter tuning or dataset bias.

Table 9: Ablation study of the proposed hybrid PDO–ACO framework.

Variant	Accuracy (%)	F1-Score (%)	Selected Features (%)	Runtime (s)
PDO-only	96	95	19	9.5
ACO-only	95.6	94.2	26	8.8
Hybrid PDO–ACO (Proposed)	98	97.5	11	11.2

4.3 Scalability and Generalization Evaluation

Building upon the NSL-KDD evaluation in Section 4.2, this subsection examines the scalability of the proposed PDO–ACO framework using the CICIDS2017 dataset, which contains approximately 2.8 million network flows described by 78 features. The same preprocessing pipeline, classifier configuration, and performance metrics were employed as defined in Section 4.2 to ensure consistent evaluation. The proposed model achieved 97.2% accuracy and 96.8% F1-score while selecting nine features (~11.5%), with runtime increasing from 11.2 to 18.9 s.

To evaluate the scalability of the proposed framework, PDO–ACO was tested on the CICIDS2017 dataset, which contains about 2.8 million network flows represented by 78 features, nearly double those of NSL-KDD (41 features). The hybrid model achieved 97.2% accuracy and 96.8% F1-score while selecting only nine features (~11.5%). Runtime increased from 11.2 s (NSL-KDD) to 18.9 s, indicating sub-linear growth with feature dimensionality. These results provide preliminary evidence of efficient scaling; future work will extend evaluation to larger benchmarks exceeding 1000 features.

The reported runtime for PDO–ACO accounts for the entire hybrid execution, including PDO's initialization and exploration stage followed by ACO's exploitation phase, since both operate sequentially within the same optimization loop.

The small reduction in accuracy (98.0% → 97.2%) and F1-score (97.5% → 96.8%) is expected because CICIDS2017 is significantly larger and more heterogeneous than NSL-KDD. It includes diverse modern attacks—DDoS, Botnet, Brute-Force, Infiltration, and others—with greater class imbalance and higher statistical variance. Such conditions naturally lead to modest accuracy variation across intrusion-detection models. Nevertheless, this ≈0.8% drop represents less than 1% relative degradation despite nearly doubling feature dimensionality and expanding the sample size more than twentyfold, confirming that PDO–ACO maintains robust generalization and adaptability under complex traffic conditions.

Although runtime increased by about 69%, this rise remains sub-linear relative to the combined growth in features (≈90%) and samples (>20×). Such behavior is expected in hybrid meta-heuristics, where optimization cost scales roughly linearly with data complexity. The two-stage cooperation of PDO (global exploration) and ACO (local exploitation) introduces a small fixed coordination overhead but accelerates convergence and prevents redundant local searches, preserving computational efficiency. Consequently, the observed runtime scaling is proportional and acceptable, supporting the framework's scalability for larger IDS environments.

Overall, PDO–ACO demonstrates a balanced trade-off between detection accuracy, feature compactness, and computational cost, offering a practical and generalizable solution for scalable intrusion-detection systems.

5 Conclusion

This study introduced a hybrid feature selection framework designed to enhance the efficiency and accuracy of Intrusion Detection Systems (IDS). By coupling the global exploration mechanism of Prairie Dog Optimization (PDO) with the precise local exploitation of Ant Colony Optimization (ACO), the proposed approach effectively navigates high-dimensional feature spaces to identify the most discriminative attributes. Experimental validation on the NSL-KDD benchmark demonstrated that the PDO–ACO model outperforms standalone metaheuristic techniques, achieving a classification accuracy of 98.00%, a recall of 98.8%, and an F1-score of 97.5%. Notably, the system maintained a false alarm rate of just 2% while reducing the feature set by approximately 89% (selecting only ~11% of the original attributes), validating its capability to balance high detection performance with computational economy.

The results confirm that the sequential integration of PDO and ACO mitigates the premature convergence often observed in single-stage algorithms, resulting in a more robust solution for complex network environments. However, this study is subject to certain limitations. First, the performance evaluation relied on static, offline datasets (NSL-KDD and CICIDS2017). While these benchmarks are standard for comparison, they do not fully replicate the continuous, drifting nature of live network traffic where attack patterns evolve rapidly. Second, although the hybrid approach is efficient, the sequential execution of two optimization stages introduces a fixed computational overhead that requires careful management in ultra-low-latency applications.

To overcome these constraints, future work will focus on deploying this framework in real-time streaming environments to assess its throughput and latency under active load. Furthermore, we aim to investigate adaptive retraining mechanisms that allow the feature selection module to dynamically update itself in response to concept drift, thereby ensuring sustained robustness against emerging zero-day threats.

Acknowledgement: The authors would like to acknowledge Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2026R500), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia for supporting this project.

Funding Statement: This research was funded by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2026R500), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Mohammad Alshinwan, Radwan M. Batyha and Walaa Alayed; data collection: Saad Said Alqahtany, Suhaila Abuowaida and Hamza A. Mashagba; analysis and interpretation of results: Mohammad Alshinwan, Azlan B. Abd Aziz and Samir Salem Al-Bawri; draft manuscript preparation: Mohammad Alshinwan, Radwan M. Batyha and Suhaila Abuowaida. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: Not applicable.

Ethics Approval: Not applicable.

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

References

1. Li S, Cao Y, Liu S, Lai Y, Zhu Y, Ahmad N. HDA-IDS: a hybrid DoS attacks intrusion detection system for IoT by using semi-supervised CL-GAN. Expert Syst Appl. 2024;238:122198. doi:10.1016/j.eswa.2023.122198.
2. Abdulganiyu OH, Ait Tchakoucht T, Saheed YK. A systematic literature review for network intrusion detection system (IDS). Int J Inf Secur. 2023;22(5):1125–62. doi:10.1007/s10207-023-00682-2.

3. Ashiku L, Dagli C. Network intrusion detection system using deep learning. Procedia Comput Sci. 2021;185(1):239–47. doi:10.1016/j.procs.2021.05.025.
4. Almomani O. A feature selection model for network intrusion detection system based on PSO, GWO, FFA and GA algorithms. Symmetry. 2020;12(6):1046. doi:10.3390/sym12061046.
5. Saleh AI, Talaat FM, Labib LM. A hybrid intrusion detection system (HIDS) based on prioritized k-nearest neighbors and optimized SVM classifiers. Artif Intell Rev. 2019;51(3):403–43. doi:10.1007/s10462-017-9567-1.
6. Eesa AS, Orman Z, Brifcani AMA. A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems. Expert Syst Appl. 2015;42(5):2670–9. doi:10.1016/j.eswa.2014.11.009.
7. Abd Elaziz M, Dahou A, Abualigah L, Yu L, Alshinwan M, Khasawneh AM, et al. Advanced metaheuristic optimization techniques in applications of deep neural networks: a review. Neural Comput Appl. 2021;33(21):14079–99. doi:10.1007/s00521-021-05960-5.
8. Abualigah L, Elaziz MA, Khasawneh AM, Alshinwan M, Ali Ibrahim R, Al-qaness MAA, et al. Meta-heuristic optimization algorithms for solving real-world mechanical engineering design problems: a comprehensive survey, applications, comparative analysis, and results. Neural Comput Appl. 2022;34(6):4081–110. doi:10.1007/s00521-021-06747-4.
9. Lazo Y, Crawford B, Cisternas-Caneo F, Barrera-Garcia J, Soto R, Giachetti G. Evolution and trends of the exploration-exploitation balance in bio-inspired optimization algorithms: a bibliometric analysis of metaheuristics. Biomimetics. 2025;10(8):517. doi:10.3390/biomimetics10080517.
10. Dorigo M, Birattari M, Stutzle T. Ant colony optimization. IEEE Comput Intell Mag. 2006;1(4):28–39. doi:10.1109/mci.2006.329691.
11. Ezugwu AE, Agushaka JO, Abualigah L, Mirjalili S, Gandomi AH. Prairie dog optimization algorithm. Neural Comput Appl. 2022;34(22):20017–65. doi:10.1007/s00521-022-07530-9.
12. Amudha P, Karthik S, Sivakumari S. A hybrid swarm intelligence algorithm for intrusion detection using significant features. Sci World J. 2015;2015(1):574589. doi:10.1155/2015/574589.
13. Elngar A, Mohamed D, Ghaleb F. A real-time anomaly network intrusion detection system with high accuracy. Inf Sci Lett. 2013;2(2):49–56.
14. Roopa Devi EM, Suganthe RC. Feature selection in intrusion detection grey wolf optimizer. Asian J Res Soc Sci Humanit. 2017;7(3):671. doi:10.5958/2249-7315.2017.00197.6.
15. Mohammadi S, Mirvaziri H, Ghazizadeh-Ahsaee M, Karimipour H. Cyber intrusion detection by combined feature selection algorithm. J Inf Secur Appl. 2019;44:80–8. doi:10.1016/j.jisa.2018.11.007.
16. Gauthama Raman MR, Somu N, Kirthivasan K, Liscano R, Shankar Sriram VS. An efficient intrusion detection system based on hypergraph—Genetic algorithm for parameter optimization and feature selection in support vector machine. Knowl Based Syst. 2017;134:1–12. doi:10.1016/j.knosys.2017.07.005.
17. Mazini M, Shirazi B, Mahdavi I. Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and AdaBoost algorithms. J King Saud Univ Comput Inf Sci. 2019;31(4):541–53. doi:10.1016/j.jksuci.2018.03.011.
18. Hajisalem V, Babaie S. A hybrid intrusion detection system based on ABC-AFS algorithm for misuse and anomaly detection. Comput Netw. 2018;136:37–50. doi:10.1016/j.comnet.2018.02.028.
19. Khare N, Devan P, Chowdhary C, Bhattacharya S, Singh G, Singh S, et al. SMO-DNN: spider monkey optimization and deep neural network hybrid classifier model for intrusion detection. Electronics. 2020;9(4):692. doi:10.3390/electronics9040692.
20. Kunhare N, Tiwari R, Dhar J. Particle swarm optimization and feature selection for intrusion detection system. Sādhanā. 2020;45(1):109. doi:10.1007/s12046-020-1308-5.
21. Kumari D, Sinha A, Dutta S, Pranav P. Optimizing neural networks using spider monkey optimization algorithm for intrusion detection system. Sci Rep. 2024;14(1):17196. doi:10.1038/s41598-024-68342-6.
22. Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw. 2016;95:51–67. doi:10.1016/j.advengsoft.2016.01.008.
23. Emary E, Zawbaa HM, Hassanien AE. Binary grey wolf optimization approaches for feature selection. Neurocomputing. 2016;172(1):371–81. doi:10.1016/j.neucom.2015.06.083.

24. Lemus-Romani J, Crawford B, Cisternas-Caneo F, Soto R, Becerra-Rozas M. Binarization of metaheuristics: is the transfer function really important? Biomimetics. 2023;8(5):400. doi:10.3390/biomimetics8050400.
25. Li S, Chen H, Wang M, Heidari AA, Mirjalili S. Slime mould algorithm: a new method for stochastic optimization. Future Gener Comput Syst. 2020;111:300–23. doi:10.1016/j.future.2020.03.055.
26. Abualigah L, Shehab M, Alshinwan M, Alabool H. Salp swarm algorithm: a comprehensive survey. Neural Comput Appl. 2020;32(15):11195–215. doi:10.1007/s00521-019-04629-4.
27. Balyan AK, Ahuja S, Lilhore UK, Sharma SK, Manoharan P, Algarni AD, et al. A hybrid intrusion detection model using EGA-PSO and improved random forest method. Sensors. 2022;22(16):5986. doi:10.3390/s22165986.
28. Emirmahmutoğlu E, Atay Y. A feature selection-driven machine learning framework for anomaly-based intrusion detection systems. In: Peer-to-peer networking and applications. Berlin/Heidelberg, Germany: Springer; 2025. p. 1–28.
29. Ghanem TF, Elkilani WS, Abdul-Kader HM. A hybrid approach for efficient anomaly detection using metaheuristic methods. J Adv Res. 2015;6(4):609–19. doi:10.1016/j.jare.2014.02.009.