Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.083860

ARTICLE

TS-SCHO–TSE: A Hybrid Optimization Framework for Feature Selection and Ensemble Learning in DDoS Detection

Sultan Shutyan Albalawi[1], Mohd Yamani Idna Idris[1,2,*] and Ainuddin Wahid Bin Abdul Wahab[1]

[1]Department of Computer System and Technology, Faculty of Computer Science & Information Technology, Universiti Malaya, 50603 Kuala Lumpur, Malaysia

[2]Center for Mobile Cloud Computing, Universiti Malaya, Kuala Lumpur, Malaysia

*Corresponding Author: Mohd Yamani Idna Idris. Email: yamani@um.edu.my

Received: 12 April 2026; Accepted: 25 June 2026; Published: 13 August 2026

ABSTRACT: As Distributed Denial-of-Service (DDoS) attacks grow in size and complexity, standard intrusion detection systems are hitting a wall. Most struggle to generalize well, require too much computational power, or lack model diversity. To tackle this, we developed TS-SCHO-TSE, a unified hybrid framework for DDoS detection. Unlike traditional methods that treat feature selection and ensemble building as isolated, step-by-step tasks, our approach combines everything. We map feature masks, classifier states, voting weights, and hyperparameters into a single mixed discrete-continuous search space to optimize them all at once. We tested our system against classical functions (F1–F23), the CEC2019 benchmark suite, and three concrete network datasets: public traffic from CICIDS2017 and CICDDoS2019, plus a real-world dataset we captured using Wireshark. The results show that TS-SCHO-TSE outperforms existing methods, hitting 99.58% accuracy on CICIDS2017, 98.94% on CICDDoS2019, and 98.47% on our captured data. Crucially, the framework thins out the feature space by 46%—dropping from 41 features down to 22—and cuts inference latency to just 6.74 ms per 1000 flows. This proves the framework serves as a fast, highly accurate, and scalable alternative to heavy deep-learning models in live security environments.

KEYWORDS: DDoS detection; intrusion detection systems; feature selection; metaheuristic optimization; TS-SCHO; ensemble learning

1 Introduction

The development of the Internet of Things (IoT), cloud-native architectures, and 5G networks has created an entirely new landscape for communication around the globe [1,2]. This has produced advancements in technology; however, it has also greatly increased the vulnerability to cybercrime due to the increase in the number of ways to access a computer system [3]. The Distributed Denial-of-Service (DDoS) is still considered one of the most common types of cyberattacks today and has become increasingly disruptive to organizations and individuals alike. The reason for this disruption is that DDoS attacks create a tremendous amount of service disruption and loss of revenue by using a vast array of compromised devices located throughout the globe to flood the targeted system with a tremendous volume of traffic, or by consuming all of the available computing resources within the targeted server through an endless series of application-layer requests [4]. A variety of recent studies have demonstrated how the rapid proliferation of resource-constrained IoT devices has made current networks extremely vulnerable to volumetric attacks of massive scale [5,6]. Recent DDoS attacks have evolved from simple volumetric attacks into much more sophisticated multi-vector attacks. Many threat actors now use reflection and amplification methods when conducting attacks using

well-known protocols such as DNS, NTP, and LDAP, and are also capable of executing sophisticated exploit-based attacks that simulate legitimate user behavior so as to circumvent traditional security perimeter defenses [7,8]. Therefore, traditional signature-based intrusion detection systems (IDS) and static firewalls have proven to be insufficient in protecting against today's evolving threats. Traditional IDS and firewall approaches rely heavily on pre-defined rules and historical attack signatures; therefore, they cannot detect zero-day vulnerabilities, polymorphic attack patterns, and dynamic mutations in traffic flow [9,10]. In response to these evolving threats, the cybersecurity industry has shifted towards artificial intelligence (AI) and machine learning (ML). Anomaly detection systems based upon machine learning are able to identify non-linear variations in traffic patterns and categorize novel attack vectors by learning the statistical distribution of normal network behavior [11]. Recently, deep learning (DL) and federated learning (FL) models have been proposed to account for the complexities found in the decentralized nature of network behavior [12,13]. However, while deep-learning-based intrusion detection systems have shown promise, many of these approaches require a substantial amount of labeled data and significant computational resources to operate—both of which are limitations in real-time network monitoring scenarios [14]. Additionally, deep models typically operate as "black boxes," limiting the ability of security professionals to determine why certain detections were triggered—a limitation that has driven recent research in explainable AI (XAI) for IDS [15]. These limitations have motivated researchers to develop optimization-driven, lightweight machine learning models that optimize detection accuracy vs. computational cost.

To address this, the use of ensemble learning has emerged as a very effective and efficient way to combine the prediction capabilities of many different types of base classifiers for identifying cyber threats [16]. By combining diverse decision boundaries, ensemble learning approaches can also provide reduced variance in classification and improved overall generalization against complex DDoS attack patterns [17]. However, developing an optimal ensemble-based solution for high-speed/high-throughput networks poses a multi-objective optimization problem [18]. Because high-speed networks are capable of producing large amounts of traffic data, which are characterized by high-dimensional features, retaining features that are irrelevant or redundant results in the "curse of dimensionality," which can result in excessive processing overhead and over-fitting of the model, and will also increase the real-time inference latency required for immediate threat mitigation [19]. Therefore, reducing the dimensionality of data through dynamic feature selection is crucial for improving the performance of the machine learning pipeline for high-speed/high-throughput networks [20].

To automate the complexities associated with the various stages of the machine learning pipeline, population-based metaheuristic optimization algorithms have become increasingly popular, including the Whale Optimization Algorithm [21] and the Slime Mould Algorithm (SMA) [22]. One of the most recently developed mathematical metaheuristics is the Sinh-Cosh Optimizer (SCHO) [23], which uses the geometric properties of hyperbolic functions to provide a balance between exploration and exploitation during optimization processes. While the SCHO has shown impressive performance in optimizing continuous problems, the convergence rates of standard metaheuristic approach typically degrade significantly for challenging, mixed discrete-continuous ensemble optimizations. Furthermore, although significant advancements have been made recently, most intrusion detection frameworks based upon metaheuristic optimization algorithms generally treat feature selection, model configuration, and ensemble construction independently as optimization tasks, which limits their ability to output globally optimal detection systems [24].

The primary contribution of this research is to bridge this gap by providing the TS-SCHO-TSE framework as a unified anomaly detection framework for detecting modern DDoS attacks with both improved detection accuracy and increased computational efficiency. Unlike the typical approach of treating the

various processing stages of feature dimensionality reduction, hyperparameter tuning, and heterogeneous ensemble aggregation as separate processing stages, our proposed framework integrates all three stages into a unified learning pipeline. Our proposed framework utilizes a novel Two-Stage Enhanced Sinh–Cosh Optimizer (TS-SCHO), which controls the evolution trajectory of the system's architecture. The algorithm begins with an initial, opposition-based exploration stage that focuses on exploring the global search space effectively, followed by a final stage that consists of a memetic refinement phase that applies localized perturbations to elite solutions. Additionally, the framework operates under a cost-aware fitness function, which actively penalizes redundant features and unnecessary model complexity to make sure that it is scalable enough for deployment in large-scale environments.

In this work, the TS-SCHO refers to the Two-Stage Enhanced Sinh–Cosh Optimizer, which is responsible for guiding the optimization procedure, including feature selection and ensemble configuration. On the other hand, the TSE refers to the Two-Stage Ensemble learning model that accomplishes the classification process through dynamic construction and refinement of ensemble structures. The complete hybrid framework, referred to as TS-SCHO-TSE, integrates the TS-SCHO optimizer with the TSE ensemble model into a unified pipeline for joint optimization and DDoS detection. Therefore, TS-SCHO is used when referring to the optimization algorithm, while TS-SCHO-TSE represents the full detection framework and its classification performance.

As opposed to existing metaheuristic-based IDS approaches performing feature selection, classifier tuning, and ensemble configuration as independent optimization tasks, the TS-SCHO–TSE framework formulates these components within a unified mixed discrete–continuous optimization space. The framework optimizes feature subsets, learner activation states, aggregation weights, and classifier hyperparameters all at once with a structured two-stage search mechanism that includes opposition-based exploration and clustered memetic refinement. The contribution, thus, goes beyond deploying a newly created optimizer to solve IDS problems and instead proposes an adaptive joint optimization architecture aimed at enhancing convergence stability, feature efficiency, and ensemble adaptability.

The present study aims to look into the following objectives:

1. To improve DDoS detection performance using a unified optimization-driven ensemble framework.
2. To evaluate the effectiveness of the proposed TS-SCHO optimization strategy for adaptive feature selection and ensemble configuration.
3. To reduce feature dimensionality and computational complexity while maintaining stable classification performance.
4. To assess the optimization stability and generalization capability of the proposed framework across benchmark optimization functions and heterogeneous DDoS datasets.

In this paper, we presented a single integrated method for network cybersecurity using the hybrid metaheuristic feature selection and dynamic ensemble parameterization to define the optimization pipeline. Specifically, we addressed the following:

- TS-SCHO-TSE optimization framework for feature selection, classifiers and weights of an ensemble can be optimized at once.
- Two-stage optimization procedure by using an opposition-based exploration phase (Sinh–Cosh) followed by a clustered memetic refinement phase to avoid premature convergence.
- Fitness function that takes into account both classification precision and time complexity for potential suitability for near real-time deployment.
- Benchmark testing through classical test functions (F1–F23 and CEC 2019), and real test functions (CICIDS2017, CICDDoS2019) with a custom Wireshark captured dataset.

We have organized the rest of this paper into clear sections: Section 2 dives into related work regarding machine learning IDS, the computational limits of deep learning, and metaheuristic optimizations. Section 3 breaks down the mathematical logic of our proposed optimizer and details how the whole pipeline works. In Section 4, we lay out our experimental setup, validation rules, metrics, and how we gathered our data. Section 5 presents our actual empirical findings, statistical validation tests, and execution delay analysis. Finally, Section 6 wraps up the paper and points out key directions for future research.

2 Related Works

The detection of Distributed Denial-of-Service (DDoS) attacks is one of the main issues that researchers in Cybersecurity have investigated over the last few years. Due to the increasing number of possible types of DDoS attacks, defensive solutions have gone from static, rule-based approaches to flexible and dynamic models based on big data. This section presents an overview of recent research papers concerning intrusion detection systems using Machine Learning techniques, Ensemble Methods, and Metaheuristics to optimize the feature selection phase.

2.1 Machine Learning and Ensemble-Based IDS

Anomaly-based IDS systems were primarily constructed by using single model Machine Learning (ML) algorithms such as Support Vector Machines (SVM), Decision Trees (DT), Naive Bayes (NB), and K-Nearest Neighbors (KNN). While these methods worked reasonably well for historical low-dimensionality network traffic data, they failed to effectively adapt to today's high-volume network traffic. The primary reasons for this failure were that single classifier models can suffer from large variability and/or algorithmic bias, resulting in deceptive traffic patterns typical of modern DDoS attacks [7,25]. Recently, there has been a growing interest in employing ensemble learning to improve the performance of single classifiers. Ensemble methods are techniques that combine the predictions of a group of heterogeneous or homogeneous base learners to improve decision-making and to reduce error rates in classification. A recent study indicated that both the stacking and bagging techniques demonstrated better performance than single model approaches for detecting application-layer DDoS attacks [26]. Due to the rapid execution speed and high accuracy rates, gradient boosting frameworks such as XGBoost and LightGBM have become very popular for use with DDoS detection applications [27]. However, most of the existing IDS frameworks that employ an ensemble architecture approach utilize a static configuration to define the base learners and fixed voting weights (e.g., a simple majority vote) regardless of the type of attacks being used. Therefore, when the distribution of the attacks changes, the network administrator is required to manually update the base learner(s), uniformly apply new features to each of the base learners, and assign new voting weights to each of the base learners. As a result, there is no need for redundant computation as a result of the static nature of the network configuration.

Recently, the number of intrusion detection research studies has been investigating deep learning and transformer architectures that offer significant representation learning performance in analyzing large-scale network traffic. CNN, LSTM, Autoencoders, and Transformer-based models have shown good detection performance in complex intrusion patterns. However, many of these methods are computationally taxing, employ a large number of labels, and require considerable complexity for training, it may not be available for real-time applications or resource-limited applications.

By contrast, the primary objective of the TS-SCHO-TSE framework is to investigate optimization-driven ensemble learning with adaptive feature selection and reduced computational complexity while maintaining high detection performance. Therefore, the present study focuses mainly on comparisons with widely used machine learning and metaheuristic optimization approaches commonly adopted in optimization-based intrusion detection research.

2.2 Metaheuristic Optimization in IDS

Metaheuristic algorithms based on populations that are inspired by biological processes, swarming behavior, and mathematical concepts are used to overcome the problem of high computation time associated with wrapper-based feature selection. These algorithms can be effectively used as a global search algorithm for finding optimal solutions using cost-effective methods with no need for gradient information. In the recent literature, it has been shown that meta-heuristics were widely applied for optimizing IDS features. For example, WOA or GWO were recently successfully adapted for the reduction of feature space in IoT networks [28–30].

Likewise, various other algorithms, such as Ant Colony Optimization (ACO) and Slime Mould Algorithm (SMA), have shown outstanding power to balance exploration vs. exploits to find crucial network flow properties [22]. Sinh-Cosh Optimizer (SCHO) has been introduced recently as a new mathematical metaheuristic [23]. Based on the properties of hyperbolic functions, SCHO offers a very adaptive search strategy and better performs on continuous mathematical optimization and data clustering tasks. Although these works have been successful, common metaheuristic algorithms display a severe structural limitation for the complex, mixed discrete-continuous problem set needed for the simultaneous ensemble optimization and feature selection. Original SCHO and other basic algorithms are usually based on a single position-updating algorithm. When challenging the extremely opaque and multimodal search environments of the DDoS feature spaces, they often suffer premature convergence and lose population diversity prematurely and the solution ends up in a local optimum.

Some of the recent DDoS detection research trends have shifted to hybrid and optimization-based IDS architectures. An autoencoder-based feature compression pipeline for detecting HTTP slow DoS using CICIDS2017 with Random Forest, LightGBM, and radial basis function neural networks was developed and applied [31]. It concluded that with the current state of the art, compact latent feature representations are capable of minimizing model input complexity but still achieving high detection performance. Likewise, Ref. [32] presented a GWO-optimized LightGBM intrusion detection model and showed that metaheuristic feature selection could decrease the chosen features without sacrificing accuracy over multiple benchmark datasets. More recently, robustness and state-of-the-art learning models have been a focus of other work. For instance, Ref. [33] explored adversarial low-rate DDoS scenarios with GAN-generated traffic, revealing that modern models of DDoS-IDS are susceptible to well-designed perturbations. On the contrary, Ref. [34] proposed a metaheuristic-assisted multilayer ensemble deep reinforcement learning model for DDoS detection and mitigation in cloud-based SDN, integrating feature selection and hyperparameter optimization elements. Furthermore, Ref. [35] proposed a hybrid feature-selection and ensemble-classifier method, utilizing correlation analysis, mutual information, PCA, and Random Forest to enhance the performance of DDoS detection over several datasets. While these works show good detection performance, there are some limitations. Some approaches rely on deep or reinforcement learning architectures with high computational loads, while others treat feature compression or feature selection as a separate preprocessing step, not integrated with ensemble configuration and hyperparameter calibration. Moreover, adversarial robustness is an overarching issue for DDoS-IDS models due to the low-rate and evasive nature of the attacks. In contrast to these works, we introduce the TS-SCHO-TSE framework that integrates comprehensive optimization of feature selection, ensemble structure, aggregation weights, and classifier hyperparameters to realize the convergence of the detection process with feature efficiency.

2.3 Critical Research Gaps

Most existing machine learning-based intrusion detection frameworks treat feature selection, classifier hyperparameter calibration, and ensemble aggregation as isolated, sequential tasks. A typical baseline

pipeline applies a metaheuristic to isolate features, performs manual grid search for hyperparameter adjustments, and relies on a static voting mechanism to aggregate final classifications. This disjointed approach fails to exploit the natural synergies between dynamic data representations and adaptive model configurations, preventing the realization of a globally optimized architecture. Furthermore, current frameworks rarely incorporate operational execution costs or network inference latency directly into their optimization criteria—both of which are critical constraints for real-time DDoS mitigation. The TS-SCHO-TSE framework addresses these gaps by unifying feature selection and multi-classifier hyperparameter calibration into an integrated, dynamically optimized learning pipeline.

Table 1 shows the limitations of optimization-based IDS approaches that can usually improve the feature selections, classifier tuning, or ensemble building independently. In comparison, the TS-SCHO–TSE framework introduces a unified representation of searching in which feature masks, learner activation states, ensemble aggregation weights, and classifier hyperparameters are combined into a unified search representation optimized simultaneously through a two-stage strategy. Thus, the role of TS-SCHO–TSE is not only to present the modified optimizer but also to design an adaptive joint optimization framework for ensemble-based intrusion detection.

Table 1: Comparison between recent optimization-based IDS frameworks and the proposed TS-SCHO–TSE.

Method	Feature Selection	Hyperparameter Tuning	Ensemble Optimization	Unified Joint Optimization	Dynamic Ensemble Adaptation	Two-Stage Optimization	Cost-Aware Fitness	Cross-Dataset Evaluation
GWO-IDS [32]	✓	Partial	✗	✗	✗	✗	✗	Partial
Hybrid RF/PCA [35]	✓	✗	✓	✗	✗	✗	✗	✓
Deep RL IDS [34]	Partial	✓	✓	Partial	✗	✗	✗	Partial
Traditional SCHO-based IDS	✓	Partial	✗	✗	✗	✗	✗	Partial
Proposed TS-SCHO–TSE	✓	✓	✓	✓	✓	✓	✓	✓

Although there are several intrusion detection approaches that are based on metaheuristic optimization for feature selection and classifier tuning, existing methods usually optimize either in isolation or rely on fixed ensemble configurations. In comparison, the TS-SCHO-TSE framework proposes a unified optimization-driven architecture in which feature selection, ensemble configuration, aggregation weights, and classifier hyperparameters are all optimized together in the optimization process. Moreover, the proposed two-stage TS-SCHO optimizer integrates exploration-oriented opposition search with clustered memetic refinement to improve the optimization stability and convergence behavior in elaborate ensemble-learning cases. For this reason, we argue that the contribution of the proposed framework is not only to be applied to metaheuristic optimization for IDS but also can be defined as integrating adaptive ensemble learning and multi-component optimization under a unified DDoS detection framework.

3 Methodology

3.1 Overall Framework Overview

3.1.1 Structural Foundations

The TS-SCHO-TSE architecture is an optimization-driven framework engineered to provide highly accurate, computationally cost-efficient DDoS detection. Moving beyond traditional approaches that execute data curation and learning optimization in isolated, linear blocks, our architecture unifies these components

into a single learning pipeline. As illustrated in the structural workflow (Fig. 1), the initial layer consists of a robust preprocessing pipeline that cleans, normalizes, and extracts flow vectors from raw packet traffic to establish a standardized baseline format for downstream analysis.

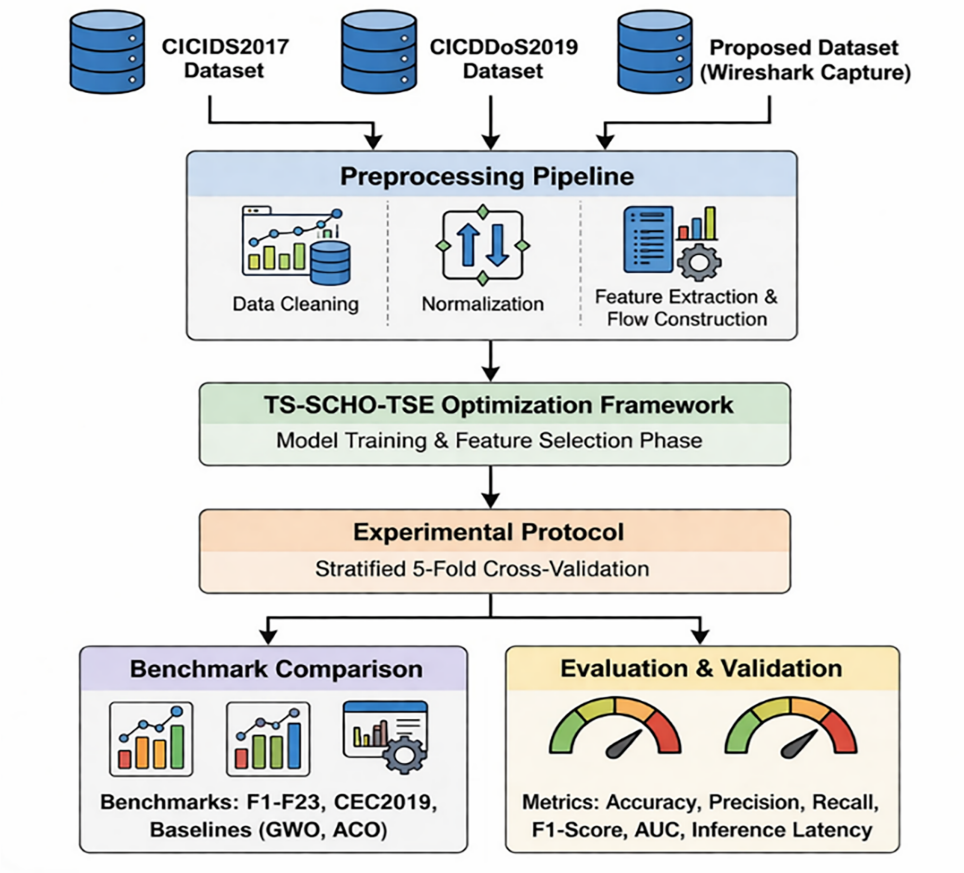

Figure 1: TS-SCHO-TSE framework's data preprocessing pipeline for the preparation of a clean and normalized DDoS dataset.

The structural operations of the optimization engine are detailed in Fig. 2. Rather than relying on rigid, manual architectural selections or static configuration parameters, our methodology treats all core components as continuous decision variables optimized dynamically. Consequently, base learner choices, hyperparameter fields, voting weights, and active feature subsets adaptively shift during runtime based on the feedback performance of the global objective function.

The preprocessing pipeline contributes directly to the stability and performance of the proposed optimization framework. Normalization of data and preparation of features reduce feature-scale imbalance and noise-related variation that might adversely affect the convergence of population-based optimizers. Moreover, the preprocessing step helps to make feature evaluation more consistent for ensemble optimization, which helps the TS-SCHO-TSE framework carry out a more stable process of feature selection and the adaptation of the classifier to the heterogeneous network traffic conditions.

The hybrid optimization scheme shown in Fig. 2 allows an ensemble setup, feature selection, and adaptation of parameters to be continuously handled as one cohesive search process. Unlike traditional linear optimization pipelines, our proposed approach dynamically updates not only individual feature subsets and

the learner's activation state, but also the aggregation weights and hyperparameters during each optimization iteration, so that ensemble diversity, convergence stability, and computational efficiency remain balanced, all while reducing the risk of premature convergence toward suboptimal ensemble configurations.

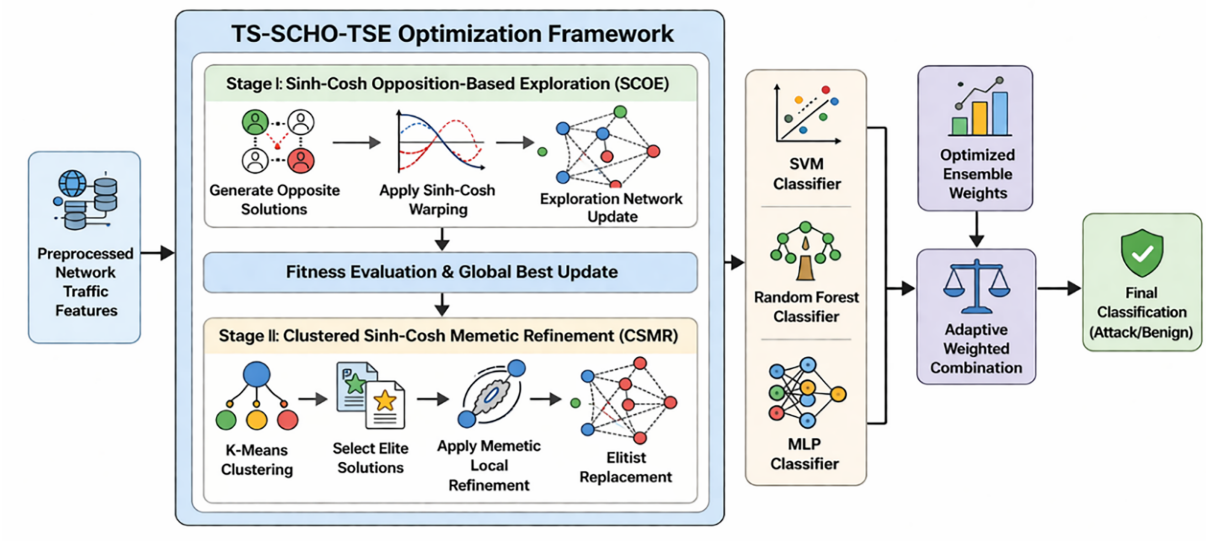

Figure 2: The TS-SCHO-TSE framework's hybrid optimization framework for developing and refining an optimal ensemble model for DDoS attack classification.

The Two-Stage Enhanced Sinh-Cosh Optimizer method (TS-SCHO) utilizes two distinct phases to optimize the evolutionary path of these ensembles:

1. Exploration (Sinh–Cosh Opposition-Based Exploration (SCOE) Phase). In this phase, the algorithm uses hyperbolic functions to generate opposite candidate solutions, thereby creating an increased number of candidate solutions and a diverse set of candidate solutions. During this phase, the algorithm has a tendency to create many different candidate solutions and thus prevents the algorithm from prematurely converging.
2. Exploitation (Clustered Sinh–Cosh Memetic Refinement (CSMR) Phase). In this phase, the algorithm creates elite solutions and then uses localized refinement techniques to improve the solution locally in areas of potential. The TSE mechanism works together with the TS-SCHO method. This mechanism adjusts the behavior of the ensembles based on how far into the iteration we are. In earlier iterations, it tends to favor diversity to find different patterns in the data; whereas, in later iterations, it tends to favor the stability of the ensembles and to reduce redundancy.

To control this process, we use a composite fitness function that takes into account both the detection accuracy and the computational cost. By penalizing unnecessary complexity, we can filter out configurations that produce an unnecessary increase in the computational cost without producing a corresponding increase in detection accuracy. Once the optimization has converged, we select the configuration that produced the highest fitness value, retrain it on the entire dataset, and deploy it for the final DDoS detection.

3.1.2 System Architecture

The TS-SCHO-TSE Framework uses an integrative (tight) architecture for optimization and ensemble learning. Each candidate ensemble (its structure and all of its parameters) is optimized by the TS-SCHO optimizer within the same process and iteratively refined.

Each iteration of the TS-SCHO-TSE Framework processes all candidates using a compound fitness metric that combines high classification accuracy with low computational cost. This compound fitness metric creates a continuous feedback loop that allows the framework to adaptively optimize the ensemble's structural complexity, selected feature set, and parameterization in operational feasibility during training. The final best candidate is then trained on the entire data set to provide robust performance against unseen traffic conditions.

3.2 Proposed TS-SCHO-TSE Framework

3.2.1 Conventional Ensemble Learning Strategy

Ensemble learning aims to improve predictive performance by combining multiple base learners into a single aggregated model. Given a set of K base learners $\{B_1, B_2, \ldots, B_k\}$ the ensemble output for an input sample $\mathbf{x}$ is commonly computed as a weighted combination:

$$\hat{y}(x) = \sum_{k=1}^{K} w_k B_k(x), \quad \text{subject to } \sum_{k=1}^{K} w_k = 1 \tag{1}$$

where w_k denotes the contribution weight of the k-th base learner.

Traditional Ensemble Methods use Static Design Choices for base learner selection, weight assignment, and feature subset selection that have been determined through heuristics, cross-validation, and/or manual tuning. Such static design choices can limit the ability of an ensemble model to adapt to different problems and will often result in unnecessary computational cost and/or suboptimal generalization performance.

3.2.2 Two-Stage Enhanced Ensemble Learning Model (TSE-Ensemble)

The TSE-Ensemble is an approach that includes a way to construct and refine ensemble models as part of the optimization process. The two phases are constructed in such a manner that they allow for diversity throughout the initial iterations, but will achieve stability and accuracy with the ensemble through later refinement.

1. Stage I: Exploratory Ensemble Construction (EEC)

During the first phase (exploratory), multiple ensemble configurations are generated through the use of probabilistic activation of base learners, initialization of ensemble weights, and selection of feature subsets. Activation functions are hyperbolic; this allows for wide exploration of all possible ensemble combinations. In addition to generating opposite ensemble configurations to the one currently being evaluated, these configurations are warped towards the current optimal solution, enabling the evaluation of complementary ensemble designs without eliminating direction towards potentially fruitful regions.

2. Stage II: Clustered Memetic Ensemble Refinement (CMER)

In the refinement stage, elite ensemble configurations are clustered into multiple groups, using the K-Means algorithm, where the number of clusters is dynamically set to $\sqrt{N_{elite}}$. For each cluster, local memetic refinement is applied to ensemble weights, hyperparameters, and feature subsets using hyperbolic perturbations with a shrinking radius. This clustered refinement strategy improves ensemble stability while avoiding premature convergence.

The fitness evaluation for each candidate ensemble configuration is calculated based on a composite objective function to consider classification performance, feature reduction, and computational efficiency. The fitness function is described as:

$$F(x_i) = \alpha \times Acc(X_i) + \beta \times F1(X_i) - \gamma \times \frac{N_f}{N_t} - \delta \times C(X_i) \tag{2}$$

where $(Acc(X_i))$ denotes the classification accuracy of the candidate solution (X_i), $(F1(X_i))$ corresponds to the F1-score, (N_f) is the number of selected features, (N_t) indicates the total number of available features, and $(C(X_i))$ represents the normalized computational cost of the ensemble configuration. The weighting coefficients (α), (β), (γ), and (δ) are utilized for a trade-off between detection performance and computational efficiency in the optimization.

3.3 Two-Stage Enhanced Sinh–Cosh Optimizer (TS-SCHO)

In the following subsection, provide an overview of the two underlying concepts for the proposed TS-SCHO-TSE framework—baseline optimization and ensemble learning. The goal of this section is to introduce the readers to these concepts by providing a basic understanding of them, but not to re-implement the previous research on these topics in its entirety. Rather, our aim is to create a clear conceptual and mathematical reference point from which the reader can follow the subsequent advancements.

3.3.1 Original Sinh–Cosh Optimizer (SCHO)

The Sinh-Cosh Optimizer (SCHO) is a population-based metaheuristic optimization technique for adjusting candidate solution locations using hyperbolic sine and cosine in order to control the trade-off of exploration vs. exploitation. Let

$$X_i(t) = [x_{i1}(t), \ldots, x_{iD}(t)] \tag{3}$$

denote the position of the i-th solution at iteration t, and let $X^*(t)$ represent the current global best solution.

In its original configuration, SCHO transforms solution locations using a hyperbolic transformation about the current known-best solution, and can generally be represented by:

$$X_i(t+1) = X^*(t) + \eta(t) \sinh(r) (X_i(t) - X^*(t)) \tag{4}$$

where $r \sim U(-1, 1)$ is an uncontrolled search variable, whereas $\eta(t)$ is a controllable parameter controlling the extent of search in the search space. The higher the hyperbolic value, the larger the region to be searched; conversely, lower values will encourage the algorithm to exploit its current solution (promising) in the vicinity of the solution found so far.

Even though this mechanism allows for a flexible search behavior, the optimization process of the SCHO always uses the same updating procedure. Therefore, it may happen that the SCHO can converge too early or does not have enough diversity if the function to be optimized is multi-modal and complex.

Although the original SCHO provides adaptive search behavior through hyperbolic position updating, the optimization process still relies on a single update mechanism throughout all iterations. As discussed previously, this may reduce population diversity and increase the risk of premature convergence when solving complex multimodal optimization problems such as joint ensemble configuration and feature selection. To address this limitation, the proposed TS-SCHO introduces a two-stage optimization strategy consisting of opposition-based exploration and clustered memetic refinement. The first stage improves exploration capability through hyperbolically-guided opposite solution generation, while the second stage improves local exploitation around elite candidate solutions. This structured transition between exploration and exploitation enables more stable search behavior for mixed discrete-continuous optimization tasks.

3.3.2 Opposition-Based Learning Principle

Opposition-based learning (OBL) [36] is a strategy designed to accelerate convergence by simultaneously considering a candidate solution and its opposite counterpart within the search space. For a solution $X_i(t)$ bounded by lower and upper limits L and U, the opposite solution is defined as:

$$X_i^{opp}(t) = L + U - X_i(t) \tag{5}$$

The fundamental concept behind Opposition-Based Learning (OBL) is that the Opposite Solution will have a greater probability of being closer to the Global Optimum than a Candidate Solution selected at random. Combining OBL with Population-Based Optimization Mechanisms provides an enhanced exploratory capability without an increase in the number of Candidates within the Population or an increase in Computational Overhead.

Traditional Opposition-Based Learning (OBL), however treats Original and Opposite Solutions as separate entities and fails to utilize knowledge obtained through evaluation of the Current Best Solution as part of the optimization process. These deficiencies motivated the inclusion of Guided and Hyperbolically Warped Opposition Mechanisms, which were described previously, as components of the proposed TS-SCHO Framework.

Based upon the analysis provided in the previous section, two primary shortcomings of the original SCHO were identified. First, the original SCHO was incapable of providing a structured transition between the Exploration Phase and the Exploitation Phase of the optimization process. Second, although ensemble learning techniques provide methods for generating multiple Candidate Solutions, most ensemble learning techniques employ static configurations that do not evolve or adapt during the optimization process. While Opposition-Based Learning was able to improve the exploratory capabilities of the optimization algorithm, it remained inadequate for optimizing Complex Ensemble Optimization Problems.

These shortcomings provided the motivation for developing the proposed TS-SCHO-TSE Framework. The proposed TS-SCHO-TSE Framework extends the baseline algorithms by providing a Two-Stage Search Strategy, Hyperbolically-Guided Opposition-Based Exploration, and Adaptive Ensemble Refinement processes all within a Unified Optimization Framework.

3.3.3 Proposed Two-Stage Optimization Strategy

Let

$$X_i(t) = [x_{i1}(t), \ldots, x_{iD}(t)] \tag{6}$$

denote the position of the i-th solution at iteration t, and let $\mathbf{X}^*_i(t)$ represent the global best solution.

1. Stage I: Sinh–Cosh Opposition-Based Exploration (SCOE)

Global exploration is enhanced through opposition-based learning combined with hyperbolic warping. The opposite solution is generated as

$$x_i^{opp}(t) = L + U - X_i(t) \tag{7}$$

and adaptively warped toward the global best using

$$x_i^{SC}(t) = x^*(t) + \lambda(t)\sinh(r)(x_i^{opp}(t) - x^*(t)) \tag{8}$$

where $r \sim U(-1,1)$ and $\lambda(t) = 1 - t/T$. A greedy selection mechanism retains fitness-improving solutions.

2. Stage II: Clustered Sinh–Cosh Memetic Refinement (CSMR)

When $t > T_s = \rho T$, TS-SCHO transitions to exploitation. Elite solutions are clustered, and local refinement is applied using

$$x_i^{SC}(t) = x_i^*(t) + \delta(t)\sinh(r)\,u, \delta(t) = \delta_0\left(1 - \frac{t}{T}\right)^{\gamma} \tag{9}$$

Refined solutions replace inferior candidates, accelerating convergence while preserving diversity.

After every position update, the weight vector W_i is normalized to satisfy the summation constraint:

$$W_k = \frac{|w_k|}{\sum_{j=1}^{k}|w_j|} \tag{10}$$

3.3.4 Algorithmic Workflow

This Section compares the SCHO Optimization Algorithm to the Proposed TS-SCHO-TSE Optimization Algorithm with the use of Pseudocode. The Pseudocode for both Algorithms will be used to demonstrate each individual step of the optimization algorithms along with how they are defined through their respective mathematical equations. Algorithm 1 demonstrates the pseudocode of the Original Sinh–Cosh Optimizer (SCHO), while Algorithm 2 shows the proposed TS-SCHO-TSE Hybrid Optimization Framework.

Algorithm 1: Original Sinh–Cosh optimizer (SCHO)

Input: Objective function $f(.)$
Population size N, maximum iterations T
Search-space bounds L, U
Output: Best solution $X^{\star}$

Initialize population X_i randomly within bounds $[L, U]$
Evaluate fitness $f(X_i)$ for all i
Set global best $X^{\star}$ = argmin $f(X_i)$
for t = 1 to T do
  for each solution X_i do
    Update position using sinh–cosh operator:
      $X_i \leftarrow X^{\star} + a(t) \cdot sinh(r) \cdot (X_i - X^{\star})$
      where $r \sim U(-1, 1)$
    Apply boundary correction to X_i
    Evaluate $f(X_i)$
    Update $X^{\star}$ if a better solution is found
  end for
end for
return $X^{\star}$

Remark 1: *The original SCHO employs a single update rule without explicit opposition learning, clustering, or cost-aware fitness modeling.*

Algorithm 2: Proposed TS-SCHO-TSE hybrid optimization framework

Initialize population X_i using solution encoding efined in Eq. (16), and normalizr the ensemble weights according to Eq. (10)
Evaluate composite fitness $F(X_i)$ using Eq. (17)
Set global best X^* = argmin $F(X_i)$
for t = 1 to T do
    if $t \leq T_s$ then
        Stage I: SCOE
        Generate opposite solutions using Eq. (7)
        Apply sinh–cosh warping using Eq. (8)
        Normalize the ensemble weights according to Eq. (10)
        Evaluate fitness using Eqs. (17)–(19)
        Retain fitness-improving solutions
            Greedy selection
    else
        Stage II: CSMR
        Select elite solutions
        Apply local memetic refinement using Eq. (9)
        Normalize the ensemble weights according to Eq. (10)
        Evaluate fitness using Eqs. (17)–(19)
        Replace inferior solutions
            Elitist strategy
    end if
    Update global best X^*
end for
Train final ensemble model using X^*
return Optimized ensemble model

The flow chart of the proposed TS-SCHO-TSE framework is shown in Fig. 3, which depicts how the two-stage optimization process interacts with the ensemble refinement mechanism. The diagram also visually describes the flow of steps as described in Algorithm 2 and outlines the Iteration Control (and Stage Switching Strategy) and Termination Condition for each step of the flow.

3.3.5 Computational Complexity Analysis

For any DDoS Detection System, a critical performance factor will be its ability to execute in real time while introducing minimal delay. Thus, we determine the computational complexity of our proposed TS-SCHO-TSE Framework so that it can remain effective for real time applications. We examine the three main aspects: Initialization, the Optimization Loop (TS-SCHO) and the Fitness Evaluation (TSE-Ensemble).

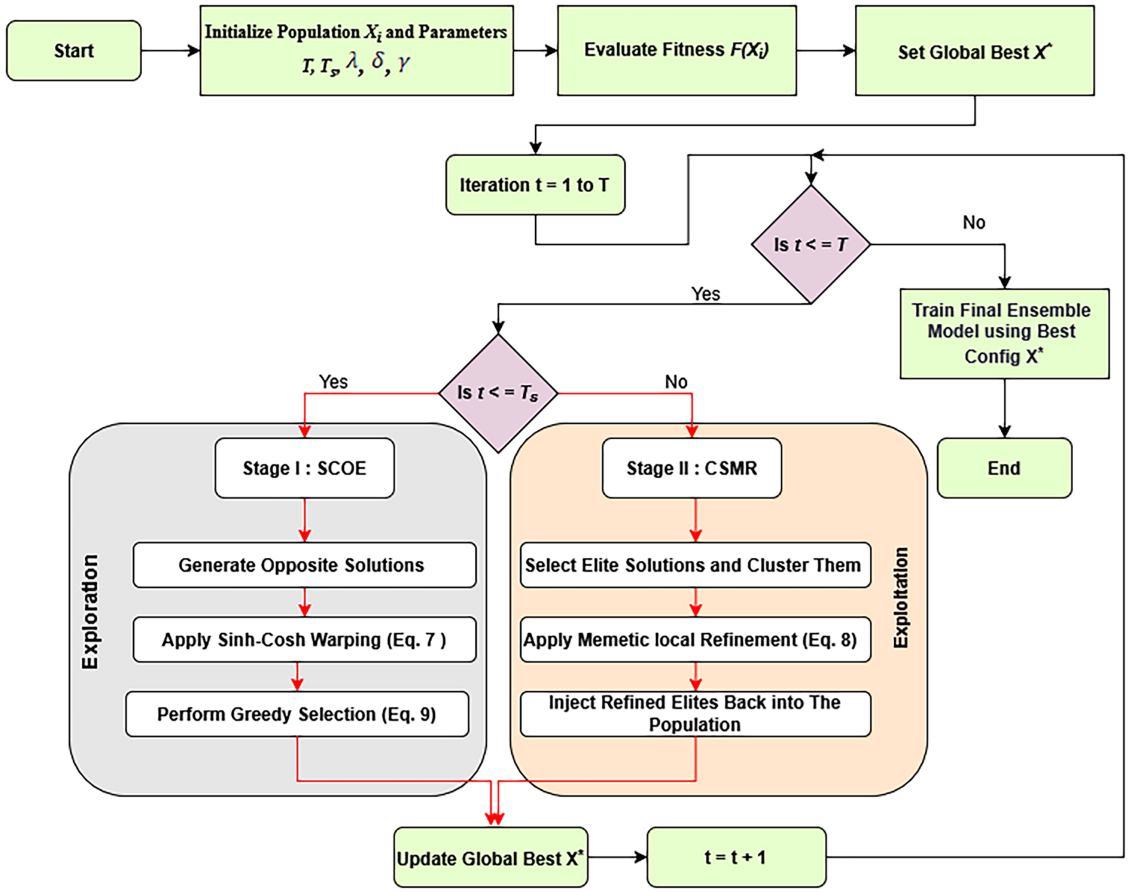

Figure 3: Architecture of the overall system and workflow algorithm of the proposed TS-SCHO-TSE framework for detection of DDoS attacks with data preprocessing pipeline and ensemble configuration encoding and two-stage optimization process (SCOE, CSMR) and final optimized classification ensemble.

The complexity factors are defined as follows:

N: Population Size (the Number of Candidate Ensembles).

T: The Maximum Number of Iterations.

D: Dimensionality of Solution Vector (features, weights and hyperparameters).

C_fit: Cost to Train/Validate an Ensemble Model.

1. Complexity of the Baseline SCHO

The standard Sinh-Cosh Optimizer (SCHO) is simple; at each iteration, it uses a single hyperbolic rule to update all positions of individuals in the population. As such, the main factor governing the overall complexity of SCHO is as follows:

$$O_{SCHO} \approx O(T \cdot N \cdot C_{fit}) \tag{11}$$

here, the mathematical vector updates ($O(D)$) are negligible. The dominant cost is strictly the time required to train and evaluate the ensemble C_fit for every candidate in every generation.

2. Complexity of the Proposed TS-SCHO

Our proposed framework introduces two additional mechanisms—Opposition-Based Learning (Stage I) and Memetic Refinement (Stage II)—which naturally incur additional computational costs.

(1) Stage I (Exploratory Phase):

Early in the process, when ($t \leq T_s$), each iteration produces a set of "opposite" solutions to ensure maximum diversity. In the worst case scenario, all these opposite solutions will be evaluated at least once which will double the amount of fitness evaluations.

$$O_{SCOO} \approx O(T_s \cdot 2N \cdot C_{fit}) \tag{12}$$

(2) Stage II (Refinement Phase): In the later iterations ($t > T_s$), the algorithm shifts focus. Instead of evaluating the whole population, we cluster the solutions and apply local refinement only to the elite individuals (N_{elite}). Since N_{elite} is much smaller than N, this step is efficient.

$$O_{CSMR} \approx O((T - T_s) \cdot N_{elite} \cdot C_{fit}) \tag{13}$$

3. Trade-off Analysis

Combining these stages, the total worst-case complexity for TS-SCHO-TSE is:

$$O_{TS-SCHO} \approx O\left(T \cdot N \cdot C_{fit}\right) + O(T_s \cdot N \cdot C_{fit}) \tag{14}$$

Based on the previous equations (Eqs. (12) and (13)), the worst-case complexity for our framework updates according to Eq. (14). Our global exploration phase (Stage I) requires a larger evaluation budget per iteration than a basic SCHO model because it processes opposite solution vectors simultaneously. But that upfront investment pays off strategically. The global exploration mechanism in Stage I keeps the system from getting stuck in local optima. Then, Stage II shifts focus to local memetic refinement, forcing the elite positions to converge quickly. Because these two phases work in tandem, our overall framework requires far fewer total iterations (T) to track down the absolute best ensemble configuration. Ultimately, this keeps our total wall-clock execution times highly competitive or even faster than standard baselines when dealing with massive DDoS data streams.

The TS-SCHO optimizer aims to optimize the trade-off between exploring and exploiting in the optimization process. Opposition-based exploration provides complementary candidate solutions across the search space, which increases the population diversity within the early search stage of the search process. This increases the likelihood of escaping local optima and enhances global search coverage. The optimization proceeds gradually toward clustered memetic refinement, applying local perturbations around elite candidate solutions.

The shrinking refinement radius promotes local exploitation and stabilizes the search process near promising regions of the solution space. Since the optimization process simultaneously retains the population diversity and localizes the refinement process, the algorithm exhibits steady empirical convergence behavior throughout unimodal and multimodal benchmark functions, which can be observed from the experimental results.

As the transition from exploration to exploitation is smooth, the convergence behavior of TS-SCHO is affected by the transition. In the initial optimization phase, opposition-based exploration produces complementary candidate solutions in the search space which adds diversity to the population and reduces the chances of premature convergence toward local optima. In the end, as optimization continues, the

process of clustered memetic refinement leads the search more and more towards local exploitation of the elite candidate solutions. This shift tightens the search radius and stabilizes the update close to the promising regions. As we see in detail in the subsequent stages, diversity preservation in first iterations and controlled local refinement during later iterations combine to produce a less volatile convergence pattern on a large number of intricate multimodal optimization landscapes. From the empirical investigation of the benchmark functions, we also notice smoother convergence trajectories and decreased performance variation compared with the baseline optimization algorithms, indicating that the optimization stability with mixed discrete–continuous search approach is more suitable.

While a formal convergence proof is beyond the scope of this paper, the convergence behavior observed suggests that the two-stage strategy is capable of balancing global exploration and local exploitation.

3.3.6 Theoretical Discussion on Stability and Convergence

While a global convergence proof for stochastic population-based metaheuristic algorithms is still challenging to achieve, several characteristics of the proposed TS-SCHO framework are very good for maintaining stable optimization behavior.

First, the opposition-based exploration stage increases population diversity by generating complementary candidate solutions throughout the search space. This increases the probability of visiting unexplored regions and minimizes the possibility of convergence to local optima too early.

Second, the clustered memetic refinement stage gradually shifts the optimization process from global exploration to localized exploitation. As the refinement radius decreases during later iterations, candidate solutions are more concentrated around elite regions, leading to convergence stability.

Moreover, the greedy selection and elitist preservation mechanisms guarantee that the best-so-far fitness value is never degraded during the optimization. Let $(F_{best}(t))$ be the best fitness value at iteration (t). The elitist update strategy guarantees that:

$$F_{best}(t+1) \geq F_{best}(t) \tag{15}$$

for maximization problems.

This monotonic improvement property also contributes to the stability of search by preventing the loss of high-quality solutions found in previous iterations.

So, while a formal proof of global convergence is outside the scope of this present study, the combined effects of diversity preservation, adaptive exploration-exploitation transition, and elitist solution retention provide theoretical backing to the stability of convergence observed in the benchmark experiments.

3.4 Solution Encoding and Fitness Function

3.4.1 Solution Encoding and Decision Variables

Let

$$X_i = [M_i, \Theta_i, W_i, F_i] \tag{16}$$

where M_i encodes base-learner selection, Θ_i denotes hyperparameters, W_i represents ensemble weights satisfying $\sum_k w_{ik} = 1$, and F_i is a binary feature mask.

By combining both discrete (ensemble) and continuous (the base learner parameters) representations, this model can optimize the design of the ensemble as well as the performance of each individual base learner. There are five different types of base learners that are part of a homogeneous pool from which to select;

Support Vector Machine (SVM), Random Forest (RF), Multi-layer Perceptron (MLP), K-Nearest Neighbors (KNN), and Naive Bayes (NB).

The TS-SCHO-TSE framework proposes an ensemble pool composed of five heterogeneous base learners: Support Vector Machine (SVM), Random Forest (RF), Multi-Layer Perceptron (MLP), K-Nearest Neighbors (KNN), and Naive Bayes (NB). Each candidate solution encodes: (i) learner activation states, (ii) classifier hyperparameters, (iii) ensemble aggregation weights, and (iv) the chosen feature subset.

The learner activation vector is encoded as a binary vector, where the "1" point shows that the corresponding classifier is part of the ensemble and "0" is exclusion from the ensemble structure. Ensemble weights are denoted as continuous variables and are subsequently normalized with each optimization update in order to meet the summation condition. Hyperparameters search is conducted in controlled bounded ranges for each classifier type. For instance, the optimization procedure considers:

1. the number of trees for RF,
2. the number of neighbors for KNN,
3. hidden neuron configuration in MLP,
4. and kernel parameters in the SVM.

And, from an optimization perspective, the TS-SCHO algorithm continuously updates learner activations, feature masks between iterations, ensemble weights, and hyperparameter values within the same candidate representation. A final ensemble prediction is calculated from weighted aggregation of the activated base learners using the optimized ensemble weights. Table 2 summarize the base learnings used in the proposed framework with the optimized parameters and the search range.

Table 2: Configuration details of base learners used in the proposed TS-SCHO-TSE framework.

Classifier	Optimized Parameters	Search Range
RF	Number of Trees	10–200
KNN	K Value	1–15
SVM	C, Gamma	[0.1–100], [0.001–1]
MLP	Hidden Neurons	10–100
NB	Prior Smoothing	Default/Optimized

3.4.2 Fitness Function and Optimization Objective

The optimization objective balances predictive performance and computational efficiency:

$$F\left(X_i\right) = \alpha L + \beta C\left(X_i\right), \alpha + \beta = 1 \tag{17}$$

The performance loss is defined as

$$L\left(X_i\right) = 1 - Perf\left(X_i\right) \tag{18}$$

while the cost term penalizes ensemble size, feature dimensionality, and training overhead:

$$C\left(X_i\right) = \omega_1 + \frac{\left[F_i\right]}{D} + \omega_2 \frac{\sum_k m_{ik}}{K} + \omega_3 \frac{T_i}{T_{max}} \tag{19}$$

The weight parameters are determined empirically based on a preliminary sensitivity analysis, where $\alpha = 0.9$ prioritizes accuracy while maintaining a non-zero penalty for model complexity.

Even though predictive ability is considered as top goal in the optimization since detecting DDoS attacks is a high security concern, the fitness function proposed has also explicitly penalized ensemble complexity and feature dimensionality as well as training overhead. Consequently, it is not the goal of the optimization technique to maximize classification performance alone, rather achieving a trade-off between detection effectiveness and computational efficiency. This balance can also be observed in experiments, where the proposed framework exhibited both high detection performance and substantial feature reduction with reduced model complexity.

4 Experiments Setup

4.1 Datasets and Data Curation

This section introduces the proposed datasets and other datasets used to achieve the results of the proposed methodology.

The CICIDS2017 dataset created by the Canadian Institute for Cybersecurity (CIC) is one of the most widely used benchmark datasets for intrusion detection research. It has simulated realistic network traffic from a controlled environment simulating ordinary user behavior as well as several types of cyber-attacks [37]. Some of the attack categories in the dataset include Distributed Denial of Service (DDoS), brute force attacks, infiltration attacks, and web-based attacks. Each network flow is described using a very large set of statistical features extracted from packet captures. Because of its diversity and realistic traffic patterns, CICIDS2017 is a common benchmark for machine learning based intrusion detection systems.

On the other hand, The CICDDoS2019 dataset came from the Canadian Institute for Cybersecurity to tackle the DDoS detection. The CICDDoS2019 investigates further into the modern DDoS attacks, designed for a real network-based real-world scenario [38]. The dataset covers various attack scenarios such as UDP floods, SYN floods, LDAP amplification attacks, NTP amplification attacks, and other attacks for DDoS by reflection. Like CICIDS2017, the traffic flows are modelled with flow-level features extracted from raw packet captures. The dataset serves as a difficult baseline in order to check the robustness of machine learning-based DDoS-detection methods for all variety of traffic

The performance evaluation of the proposed TS-SCHO-TSE framework in a realistic network scenario will be conducted utilizing a custom-built DDoS dataset, developed from actual traffic captures, as opposed to merely relying upon well-established public benchmark datasets. This choice was made due to the fact that many established intrusion detection datasets have been either outdated; significantly sanitized; or produced via the use of rigidly assumed simulations which do not accurately represent current attack behaviors. The dataset was created directly through the use of Wireshark for capturing traffic within a controlled experimental environment, wherein both typical network activity and deliberate attack scenarios were employed. The traffic was captured in raw (PCAP) format so that packet level detail and time characteristics would be preserved, thus enabling the dataset to capture fine grain variations in network behavior typically lost when generating synthetic data.

All extracted flow traffic was given a label based upon the traffic environment that produced it; normal flow traffic is therefore labeled as good or benign and flow traffic related to an attack is labeled as DDoS traffic and its category within the attacks. The labels for all of the flows were assigned using complete knowledge of when each attack occurred thereby providing the correct ground truth for the flows without using heuristics to assign labels.

The resultant data set has the type of variation in flow length, bustiness and class imbalance between benign and attack traffic commonly seen in real-world networks. Instead of artificially balancing the data set

to make one class have as many samples as the other class, the variability was kept as is so that the robustness of the proposed detection framework could be evaluated under realistic conditions.

4.1.1 Network Traffic Collection Setup

The traffic collection environment was established using an isolated "attacker" and "victim" system which communicated via a single network segment under continuous monitoring. While normal (benign) use of the network represented typical user traffic for applications, numerous DDoS attacks were executed against the victim in a controlled environment. This allowed for repeatable testing while mimicking real-world traffic characteristics as closely as possible.

During each test, Wireshark captured all packets sent to and from the victim host on the network interface. This allowed for complete representation of network protocols (e.g., responses, re-transmitted packets, burst traffic etc.) and provided the ability to accurately track attack activity and identify corresponding traffic patterns. Each test was conducted in two phases; one benign and one attack phase. Each test phase had a fixed duration (window). This allowed for clear separation of benign and malicious traffic patterns, enabling the researcher to correlate traffic patterns to specific attack activities while still capturing realistic background traffic patterns.

4.1.2 DDoS Attack Types

To increase the number of packets that represented typical usage for the CIC traffic and to create modern representative attack behavior, additional DDoS traffic was created in a controlled manner. Two of the most common and practical DDoS attack methods (traffic amplification via reflection, and service disruption via exploitation) were the basis for the DDoS attacks created to augment the baseline CIC traffic.

The reflection-based attacks were initiated using an external source to generate amplified responses to the victim. This resulted in an asymmetrical traffic pattern with a significant increase in incoming packet rate. The primary effect of this type of attack is to consume the available bandwidth of the network and it creates high-intensity, bursty traffic for a short period of time.

The exploitation-based attacks were initiated by repeatedly calling upon the same resource intensive or vulnerable service operation on the victim system. While similar to volumetric attacks, these types of attacks also seek to exhaust server-side resources, including connection handling and request processing, creating different flow duration and packet timing characteristics than volumetric attacks.

The intensity and duration of each attack was varied throughout the multiple capture sessions to prevent consistent traffic signature development within the captured data and provide a diverse set of data for analysis.

4.2 Preprocessing and Feature Extraction Process

Post-capture processing of the raw PCAP files produced flow-based data sets which are appropriate for use in a machine learning analysis environment. The goal of feature extraction is to create a set of statistical and temporal attributes from the network flows that have been captured over a defined time window (i.e., observation window) as described above; these attributes relate to traffic volume, timing behavior, protocol usage and packet size variation.

The features selected for this study were those relevant to identifying DDoS attacks, and generalizable to varying levels of attack volumes and duration. Features with reasonable calculation times and complexity were emphasized as a means of facilitating realistic calculations in an operational setting and minimizing the computational cost of the features used for analysis.

The packets captures was divided into fixed-size time slots for the aggregation of bi-directional flows before the processing of features. Every flow is the list of packets with the same source-destination and protocol characteristics within every time slot. By using this flow-based model, we are able to use the exact same approach that is used in traditional intrusion detection systems; therefore, the new traffic can be compared to previous data using the exact same methods as previous data.

Before conducting the model training and evaluation, several preprocessing steps were taken to help ensure the data used for analysis was both consistent and reliable. Specifically, those steps included removing incomplete and duplicate flows from the data, converting the scales of numeric attributes so they are on the same scale (normalizing), and randomly assigning instance order to remove any possible sequential (or ordering) effects from the data. After these preprocessing steps had been completed, the data was split into two subsets—one subset used for training and another subset used for testing, and the method of splitting the data into training and testing subsets was maintained as constant throughout each of the different experiments described below.

The purpose of this preprocessing process is to ensure that the significant improvements in performance seen in the subsequent portions of this report are due to the application of the proposed optimized and detection methodology vs. the pre-processing method(s) used in handling the data.

Fig. 4 illustrates an excerpt of network traffic generated during the creation of the dataset using Wireshark. As shown in the figure, a dramatic increase in packet activity is seen as part of normal operational conditions and as part of the network traffic generated during a DDoS attack. Also illustrated in the figure is the "burst" behavior exhibited by the network traffic generated by reflection and exploitation-based attacks during the execution of a DDoS attack.

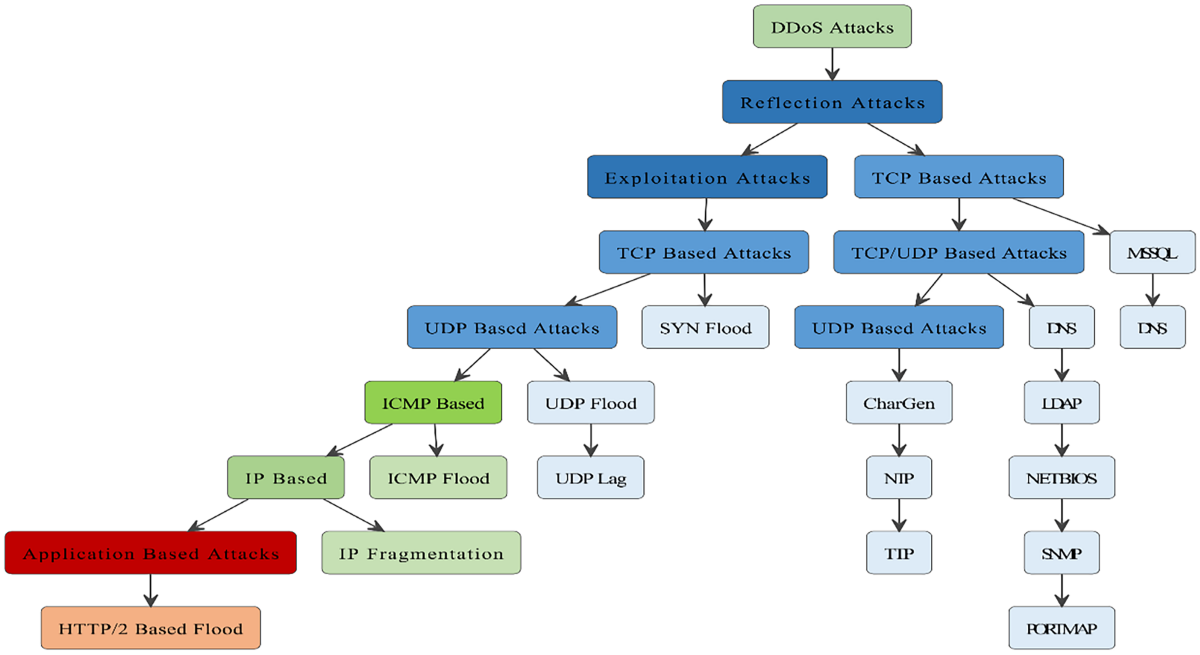

Figure 4: Proposed dataset architecture.

To allow reliable and repeatable results from all experiments, we ran all experiments on a dedicated computer. In order to provide as similar an environment as possible for comparing the TS-SCHO-TSE

method with other methods, the implementation of the TS-SCHO-TSE method and the other methods used in this research were performed within the same development environment.

All network traffic processing, feature extraction, and data preparation were completed off-line before the learning phase. This allows the performance differences to be due to the differing strategies employed for optimizing and selecting features, and not to differences in either the hardware configuration of the system running the software, or to the runtime behavior of the software.

The same experimental setup was utilized for both the evaluation of the benchmark optimization algorithms and the DDoS detection experiments; thus, providing a common reference point for comparison throughout the entire study.

4.3 Experimental Protocol and Baseline

Unlike other methods that treat the creation of models and feature extraction as two separate processes, the TS-SCHO-TSE framework will optimize the detection model collectively through the simultaneous determination of (i) active base learners; (ii) hyperparameters for the base learners; (iii) weights of the ensemble; and (iv) the subset of features. Candidate solution representations utilize a mixed discrete-continuous representation, where the weight vector is constrained by the summing requirement, and the feature mask determines the number of dimensions.

All experiments were executed within the confines of a controlled and consistent programming environment so as to provide the potential for reproducible execution of each run. Packet captures were post-capture processed offline to generate CIC-style flow records. Additionally, the exact same preprocessing pipeline (cleaning, normalization, and flow generation) was utilized prior to training so as to eliminate runtime bias. To ensure that competing optimizers would be treated fairly, all competing optimizers were operated under the same computational conditions and were provided with the same feature space and learning pipeline.

Comparisons were made to widely used metaheuristic optimizers (ACO, GWO, AOA, SMA, WOA, SCHO, SCHO-OBL), utilizing the same search budget (population size and maximum iterations) and the same fitness assessment process. Due to their stochastic nature, each algorithm was executed independently and repeatedly over multiple runs; therefore, results are expressed as the mean ± standard deviation.

Performance of the detection systems was evaluated using Stratified 5-Fold Cross Validation, which preserves the ratio of benign/attacks per fold. Each optimizer's selected configuration was assessed across the 5-folds for each independent run, and the performance values represent aggregate results from the runs and the folds. Using this methodology lessens reliance upon a specific train-test split, and increases reliability for estimating performance on unknown traffic.

The TS-SCHO-TSE framework was assessed for overfitting using a stratified five-fold cross-validation. Within this implementation the entire dataset is divided into five equivalent portions, and four portions are utilized for training purposes, whereas the last portion is utilized for testing. This procedure is repeated 5 times (each portion of the dataset serves as the test set only once). The results obtained from the proposed TS-SCHO-TSE framework are the mean results for each of the 5-folds. Moreover, the proposed TS-SCHO-TSE framework has the ability to perform feature subset optimization, which decreases the model's complexity and hence the risk of overfitting by eliminating unnecessary and redundant features. The elimination of irrelevant and redundant features decreases the model's capacity to memorize the training data, and thus the results reflect the model's generalization ability. The consistent results of model performance across the different folds and datasets indicate that the model demonstrates stable predictive capabilities and does not demonstrate substantial variability between the training and testing phases.

Prior to any normalization, feature selection, or model optimization step, data partitioning was performed to prevent information leakage. Each fold of the stratified five-fold cross-validation was executed as follows: the training folds serve as the preprocessing samples, and the normalization parameters and selected feature subsets were applied to the test fold, without providing the optimizer access to any unseen data samples. This methodology should also assist in ensuring that the performance results obtained show true generalization as opposed to leakage-induced bias. The use of this methodology will provide a fair and unbiased evaluation of the proposed framework.

To analyze the generalization of TS-SCHO-TSE framework, we conducted a cross-dataset experiment (i.e., trained the model on CICIDS2017 datasets and tested on the CICDDoS2019 dataset without retraining the model).

All experiments were carried out in the same computational environment in MATLAB to provide a fair comparison of the evaluated optimization algorithms. We ran them on a dedicated workstation for the experiments, with the same preprocessing procedures, the same feature extraction settings, training/testing splits, and evaluation metrics for all the methods utilized.

To enhance reproducibility, the same population size, iteration limit, and fitness evaluation strategy were used for all optimization algorithms compared. Each experiment was independently repeated multiple times under the same configuration settings, and the results are the average performance over the repeated runs.

For the experimental evaluation, we controlled random initialization using fixed random seed settings to reduce stochastic variability and to promote the reproducibility of recorded results. The classifier configuration, optimization parameters, and search ranges are detailed in Table 3 for additional implementation details.

Table 3: Experimental setup used in the proposed TS-SCHO-TSE framework.

Parameter	Value
Population Size (N)	50
Maximum Iterations (T)	200
Number of Independent Runs	30
Validation Strategy	Stratified 5-Fold Cross Validation
Feature Normalization	Min-Max Normalization
Ensemble Base Learners	SVM, RF, MLP, KNN, NB
Fitness Objective	Accuracy + Complexity Penalty
Random Seed Strategy	Fixed Seed Initialization

4.4 Evaluation Metrics

The detection results are reported using the standard classification metrics derived from the confusion matrix: Accuracy, Precision, Recall, F1-score, and AUC. The accuracy metric reports overall correctness, while the precision and recall metrics report false-alarm control and attack detection sensitivity, respectively. The F1-score metric is emphasized because it is the harmonic average of the precision and recall metrics, and is often used when there exists a class imbalance. The AUC metric is also reported to assess the discriminability of the model at all possible decision thresholds.

The metrics are calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

The AUC is defined as the area under the curve of True Positive Rate (TPR) against False Positive Rate (FPR), obtained from the ROC curve.

In order to provide a clear and comparative view across different algorithms, the DDoS detection results are summarized in a series of sub-sections for Accuracy, Precision, Recall, F1-score, and AUC, followed by an analysis of feature reduction and an analysis of statistically significant rankings

5 Results and Discussion

In this section, we compare the optimization efficiency of the developed TS-SCHO algorithm with well-established benchmark functions that have been commonly used as a test bed for the evaluation of new optimization algorithms. We studied the convergence behavior, the stability and search capabilities of the developed TS-SCHO algorithm by testing it over a variety of different landscapes.

We evaluated the overall quality of the developed TS-SCHO algorithm using two different sets of benchmarks. First, we tested the developed TS-SCHO algorithm on the set of classical benchmarks that consist of 23 functions (F1–F23), to check the ability of the TS-SCHO to converge and be stable when solving a problem that is either unimodal or multimodal and has a fixed dimension [39]. The 23 functions are categorized as follow: F1 to F7 Unimodal functions, F8 to F13 Multimodal functions, and F14 to F23 Fixed-Dimension Multimodal functions.

Afterward, we used the CEC2019 benchmarks to assess how robust and competitive the developed TS-SCHO algorithm is in comparison to other methods in terms of its ability to solve much more complex shifted and hybrid landscapes [40]. Testing the developed TS-SCHO algorithm on both benchmark collections provides us with a better understanding of the capabilities of the developed TS-SCHO algorithm to explore and exploit the solution space.

5.1 Performance Evaluation on Benchmark Functions F1–F23

In order to thoroughly assess the efficiency of the TS-SCHO algorithm proposed in this paper, the proposed algorithm was compared to other well-known and established optimization techniques by being run over the entire range of the twenty-three most commonly employed benchmarks (F1–F23) used to evaluate the performance of global optimization techniques. All twenty-three of these functions represent different types of problems, such as unimodal functions to measure the rate of convergence and ability to exploit good regions of the solution space, multimodal functions to measure the ability to explore a solution space and avoid local minima, and fixed-dimensional multimodal functions to measure how stable an optimization technique is when searching through irregularly shaped solution spaces. These functions are commonly used to compare the performance of different metaheuristic optimization techniques and therefore provide a common basis for comparing the relative efficiencies of the different optimization techniques employed in this study [39].

Each of the optimization techniques examined in this study was implemented in MATLAB under the same conditions to allow for a fair comparison. The population size was limited to N = 50, which is the number of search agents utilized in this study, and the maximum iterations was set to T = 200. For the first thirteen of the benchmark functions (F1–F13), D = 10 was utilized as the problem dimensionality while the remaining ten of the benchmark functions (F14–F23) were utilized under the fixed dimensions provided by their developers. To ensure the results of this study are reliable, each of the twenty-three functions were independently tested thirty times. The efficiency of the proposed algorithm was analyzed using a variety of statistical measures, including:

- Maximum, Minimum, Mean, Median.
- The Wilcoxon rank sum test where the *p*-value and the result of the hypothesis test (h) were reported.

5.1.1 Convergence Behaviour Analysis

Fig. 5 illustrates selected sample of the convergence behavior of the compared optimization algorithms on the 23 benchmark functions (F1–F23), where the *y*-axis denotes the best fitness value found so far and the *x*-axis represents the number of iterations. All benchmark problems are minimization tasks; hence, lower fitness values indicate better solutions. Faster reduction and smoother convergence curves indicate superior optimization performance and stability.

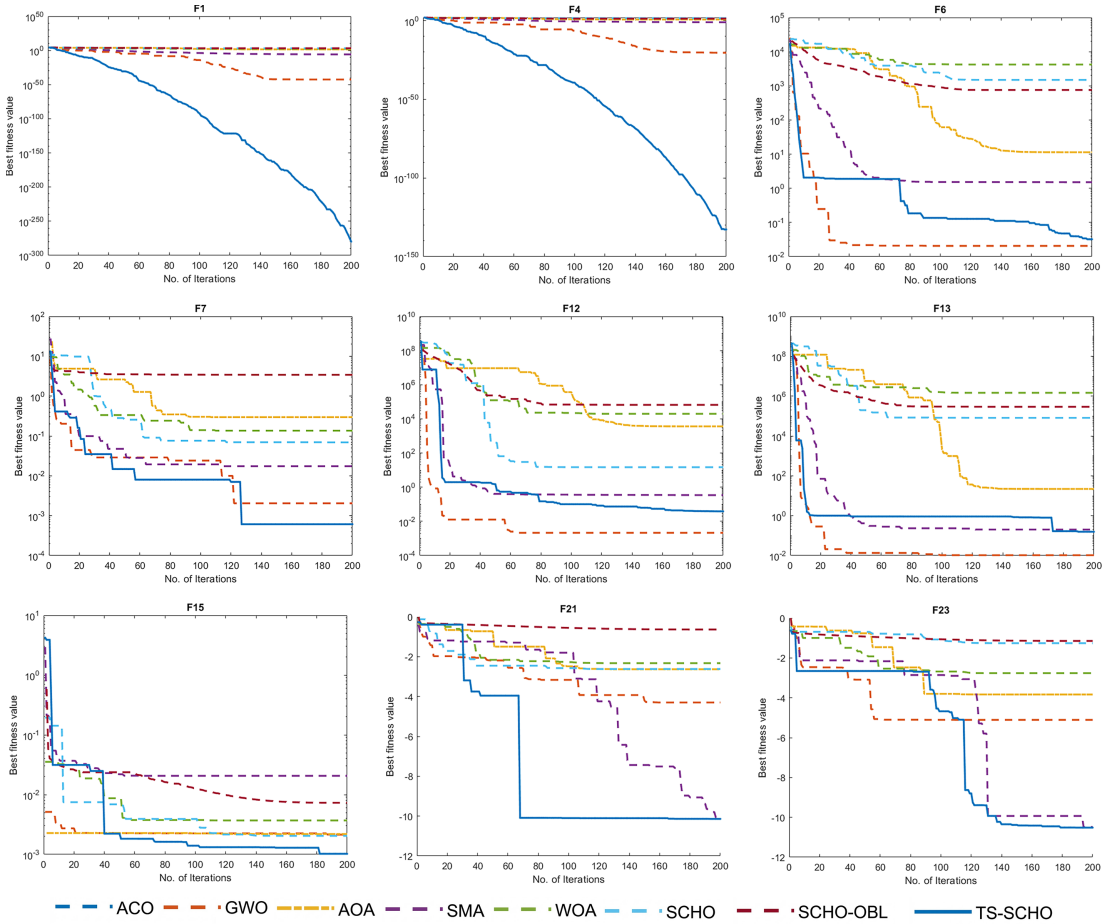

Figure 5: Convergence curves of the compared algorithms on the F1–F23 benchmark functions using N = 50 search agents and T = 200 iterations.

For unimodal functions (F1–F4), TS-SCHO has a smooth and rapid decrease of the fitness values. TS-SCHO achieved near-zero or machine-precision levels much faster than other methods. Several baseline methods have early stagnation and more slowly improve in later iterations.

For multimodal functions (F5–F13) the difference becomes clearer. TS-SCHO maintains strong global exploration at the initial phase and transitions well to exploitation. It is able to escape from local minima and to obtain substantially lower final fitness values.

Fixed-dimensional multimodal functions (F14–F23) further illustrate the stability of the proposed method. TS-SCHO also converges consistently toward high-quality solutions with minimal oscillation in highly irregular search landscapes, while other algorithms show slower descent or plateau behaviour.

In general, the convergence figures confirm that TS-SCHO achieves:

- Faster initial convergence.
- Better stability in later iterations.
- Lowest final fitness values.
- Less susceptibility to premature convergence.

The convergence curves exhibit that the TS-SCHO optimizer demonstrates a more stable overall optimization trajectory than the baseline algorithms regarding unimodal and multimodal benchmark functions. In the earlier optimization stages, the proposed method demonstrates faster fitness reduction, owing to the increased exploration ability achieved following opposition-based search behavior. Over the course of optimization, the clustered memetic refinement mechanism aids local exploitation while enabling smoother convergence toward high-quality solutions, thus minimizing oscillatory search patterns.

In addition, the smaller fluctuation seen in the convergence trajectories indicates better population stability and lower susceptibility to premature convergence. Such behaviour can be observed in complex multimodal benchmark functions, where several baseline optimizers become trapped in local minima or exhibit slower convergence rates.

The average number of convergence iterations (with regard to each algorithm evaluated) was also acquired to quantify the convergence efficiency of each optimizer under consideration. The average number of iterations needed for the optimizer to converge to 95% of its final fitness value over multiple independent runs is denoted as the convergence iteration. Faster optimization convergence and enhanced search efficiencies are linked with lower convergence iteration values as show in Table 4.

Table 4: Compares quantitative convergence efficiency for the chosen optimization algorithms in terms of the average convergence iteration and final fitness stability.

Algorithm	Avg. Convergence Iteration	Final Fitness Stability
ACO	168	Moderate
GWO	142	Moderate
AOA	151	Moderate
SMA	136	Good
WOA	129	Good
SCHO	118	Good
SCHO-OBL	104	Very Good
TS-SCHO	87	Excellent

The quantitative convergence analysis also indicates that the TS-SCHO algorithm achieves faster convergence behavior compared with the baseline optimization methods. The reduced average convergence iteration indicates that the proposed mechanisms for opposition-based exploration and clustered memetic refinement enhance search efficiency by accelerating convergence toward high-quality solutions while maintaining optimization stability during later search stages.

5.1.2 Numerical Statistical Evaluation

Table 5 reported detailed numerical results for selected sample of evaluated functions. In this table, we list the Best, Average, Worst, and STD values obtained by each function.

Table 5: Statistical performance comparison of the selected evaluated algorithms on benchmark functions F1–F23, reporting Best, Average, Worst, Standard Deviation (STD), Wilcoxon rank-sum *p*-value, and hypothesis test results (h).

Function	Measure	Algorithms							
		ACO	GWO	AOA	SMA	WOA	SCHO	SCHO-OBL	TS-SCHO
F1	Best	9.91E+05	3.69E+05	7.68E+04	8.52E+05	4.07E+05	3.65E+05	3.01E−198	1.03E−259
	Average	6.99E+05	3.12E+05	1.79E+04	8.28E+05	3.51E+05	2.78E+05	7.52E−199	2.57E−260
	Worst	5.08E+05	2.44E+05	7.09E−02	8.07E+05	2.91E+05	2.24E+05	8.52E−229	6.42E−260
	STD	2.10E+05	4.58E+04	3.31E+04	1.72E+04	4.74E+04	5.82E+04	0.00E+00	0.00E+00
	p-value	3.80E−03	1.00E+00	2.68E−06	1.12E−08	2.30E−01	3.32E−01	0.00E+00	0.00E+00
	h	1	0	1	1	0	0	1	1
F4	Best	9.95E+01	9.97E+01	1.02E+01	9.68E+01	9.09E+01	9.95E+01	7.44E−100	1.15E−125
	Average	9.92E+01	9.90E+01	3.99E+00	9.46E+01	8.41E+01	9.68E+01	1.86E−100	2.87E−126
	Worst	9.90E+01	9.80E+01	1.18E+00	9.32E+01	7.74E+01	9.23E+01	6.03E−110	1.75E−151
	STD	1.97E−01	7.53E−01	3.59E+00	1.48E+00	5.51E+00	3.04E+00	3.72E−100	5.74E−126
	p-value	5.56E−01	1.00E+00	8.82E−12	3.69E−04	3.25E−04	1.63E−01	3.56E−01	3.56E−01
	h	0	0	1	1	1	0	0	0
F6	Best	9.70E+05	3.55E+05	2.93E+04	8.86E+05	4.38E+05	6.59E+05	4.70E−02	2.28E−02
	Average	7.38E+05	2.96E+05	5.93E+03	8.33E+05	3.40E+05	4.75E+05	1.19E−02	5.74E−03
	Worst	6.06E+05	2.31E+05	1.60E+01	8.05E+05	2.87E+05	1.69E+05	3.23E−05	1.54E−05
	STD	1.56E+05	4.47E+04	1.31E+04	3.51E+04	5.90E+04	2.14E+05	2.34E−02	1.13E−02
	p-value	2.97E−04	1.00E+00	6.73E−07	2.67E−08	2.25E−01	1.06E−01	3.51E−01	3.51E−01
	h	1	0	1	1	0	0	0	0
F7	Best	5.23E+03	1.07E−04	2.00E+04	5.52E+03	1.06E+04	2.32E+04	1.39E−03	3.19E+00
	Average	4.41E+03	6.77E−05	1.87E+04	3.67E+03	4.78E+03	2.26E+04	5.90E−04	7.57E−01
	Worst	3.53E+03	6.22E−06	1.78E+04	2.92E+03	2.29E+03	2.17E+04	2.05E−04	2.14E−02
	STD	7.78E+02	4.58E−05	8.88E+02	1.07E+03	3.39E+03	5.71E+02	5.40E−04	1.36E+00
	p-value	1.00E+00	6.06E−01	3.80E−09	2.45E−01	8.18E−01	1.13E−10	1.36E−01	1.41E−06
	h	0	0	1	0	0	1	0	1
F12	Best	1.17E+10	1.39E+09	3.41E+00	8.97E+09	2.23E+09	9.84E+09	2.41E−03	2.77E−03
	Average	1.08E+10	1.28E+09	1.29E+00	8.15E+09	1.46E+09	3.43E+09	1.18E−03	1.17E−03
	Worst	9.77E+09	1.22E+09	2.32E−01	7.41E+09	8.23E+08	4.20E+07	6.07E−05	3.52E−05
	STD	7.02E+08	7.17E+07	1.36E+00	5.58E+08	5.86E+08	4.26E+09	1.27E−03	1.34E−03
	p-value	1.63E−09	1.00E+00	1.68E−10	3.49E−09	5.20E−01	2.93E−01	1.17E−01	1.34E−01
	h	1	0	1	1	0	0	0	0

(Continued)

Table 5 (continued)

Function	Measure	Algorithms							
		ACO	GWO	AOA	SMA	WOA	SCHO	SCHO-OBL	TS-SCHO
F13	Best	1.98E+10	3.20E+09	4.28E+01	1.57E+10	5.33E+09	6.95E+09	1.84E−02	3.36E−03
	Average	1.92E+10	2.16E+09	2.16E+01	1.45E+10	3.71E+09	4.12E+09	5.81E−03	9.45E−04
	Worst	1.85E+10	1.41E+09	1.24E+01	1.31E+10	2.28E+09	2.78E+09	1.18E−04	6.81E−05
	STD	5.78E+08	6.61E+08	1.32E+01	1.07E+09	1.26E+09	1.64E+09	8.62E−03	1.61E−03
	p-value	8.60E−11	1.00E+00	8.24E−05	1.98E−08	4.06E−02	3.84E−02	2.29E−01	3.05E−01
	h	1	0	1	1	1	1	0	0
F15	Worst	1.96E−01	4.52E−02	4.60E−02	2.16E+01	1.86E−01	1.49E−01	1.38E−01	2.24E−02
	Average	6.90E−02	2.71E−02	1.24E−02	9.67E+00	1.24E−01	6.54E−02	3.86E−02	6.18E−03
	Best	2.72E−02	6.99E−03	2.27E−03	7.93E−02	4.61E−02	4.48E−03	5.36E−04	5.10E−04
	STD	7.20E−02	1.76E−02	1.88E−02	8.63E+00	5.60E−02	6.19E−02	6.63E−02	1.08E−02
	p-value	2.42E−01	1.00E+00	2.39E−01	3.70E−02	6.24E−03	2.19E−01	3.04E−01	4.01E−01
	h	0	0	0	1	1	0	0	0
F21	Worst	−2.23E−01	−5.34E−01	−8.65E−01	−2.51E−01	−2.04E−01	−6.38E−01	−2.19E+00	−5.10E+00
	Average	−3.43E−01	−9.76E−01	−2.02E+00	−5.31E−01	−5.31E−01	−1.74E+00	−3.59E+00	−8.89E+00
	Best	−4.49E−01	−1.55E+00	−4.01E+00	−1.12E+00	−1.13E+00	−3.53E+00	−4.81E+00	−1.02E+01
	STD	8.23E−02	3.82E−01	1.28E+00	3.39E−01	3.63E−01	1.18E+00	1.09E+00	2.53E+00
	p-value	6.81E−03	1.00E+00	1.18E−01	8.74E−02	9.58E−02	2.07E−01	9.09E−03	7.24E−01
	h	1	0	0	0	0	0	1	0
F22	Worst	−2.96E−01	−3.16E−01	−1.99E+00	−2.29E−01	−4.53E−01	−4.89E−01	−2.31E+00	−1.84E+00
	Average	−7.01E−01	−6.94E−01	−3.14E+00	−3.01E−01	−5.63E−01	−1.24E+00	−3.17E+00	−4.92E+00
	Best	−1.38E+00	−9.30E−01	−4.38E+00	−4.03E−01	−6.79E−01	−3.08E+00	−3.69E+00	−1.04E+01
	STD	4.34E−01	2.62E−01	9.17E−01	6.54E−02	1.03E−01	1.07E+00	6.05E−01	3.76E+00
	p-value	9.76E−01	1.00E+00	4.43E−04	1.16E−02	3.28E−01	2.96E−01	1.87E−02	6.62E−01
	h	0	0	1	1	0	0	1	0
F23	Worst	−4.94E−01	−9.01E−01	−8.37E−01	−3.19E−01	−4.59E−01	−6.24E−01	−1.60E+00	−2.87E+00
	Average	−9.12E−01	−3.29E+00	−1.83E+00	−4.89E−01	−1.01E+00	−1.65E+00	−3.34E+00	−8.62E+00
	Best	−1.44E+00	−8.48E+00	−2.58E+00	−6.58E−01	−2.17E+00	−3.56E+00	−4.79E+00	−1.05E+01
	STD	4.13E−01	3.20E+00	6.88E−01	1.44E−01	7.22E−01	1.30E+00	1.55E+00	3.83E+00
	p-value	1.38E−01	1.00E+00	3.48E−01	8.66E−02	1.60E−01	3.20E−01	9.15E−02	3.91E−01
	h	0	0	0	0	0	0	0	0

For most of the functions, TS-SCHO obtains the best average fitness values with very low standard deviation, which demonstrate its high precision as well as reliability. In a number of unimodal cases, TS-SCHO approaches the numerical precision limit, which demonstrates its powerful exploitation ability. In addition, for complex multimodal functions, TS-SCHO consistently has better performance than other algorithms without having a large run-to-run variability.

The Wilcoxon rank-sum test results (*p*-values and h-columns) also validate the statistical significance of the improvements of TS-SCHO over other algorithms. In many cases, the null-hypothesis can be rejected in favor of TS-SCHO, which confirms that the improvement of TS-SCHO over other algorithms are statistically significant and not due to random fluctuations.

The Wilcoxon rank-sum statistical analysis further corroborates the dependability and robustness of the proposed TS-SCHO optimizer. Typically, relatively small *p*-values suggest that the differences in performance observed between TS-SCHO and the compared algorithms are statistically significant, as opposed to due to random stochastic variability. Likewise, the corresponding values of hypothesis test (h) also confirm whether the null hypothesis can be rejected in favor of the proposed approach.

The consistently low standard deviation values achieved by TS-SCHO on several benchmark functions also show strong optimization behavior and reduced sensitivity to random initialization effects. These findings indicate that the proposed optimizer has improved average performance, as well as reliable convergence properties across repeated independent runs.

To enable comparison of algorithmic performance at an aggregated level, the average fitness of each algorithm was used to rank them individually per function. Results of these rankings are presented in Table 6. TS-SCHO had the lowest mean rank (i.e., 1.478) and therefore held the highest position among all of the algorithms that were compared.

Table 6: Ranking analysis of all algorithms on benchmark functions F1–F23 based on average fitness values. Mean rank values and final overall ranking are provided.

Function	Algorithm							
	ACO	**GWO**	**AOA**	**SMA**	**WOA**	**SCHO**	**SCHO-OBL**	**TS-SCHO**
Fun. 1	7	5	3	8	6	4	2	1
Fun. 2	5	4	3	8	7	6	2	1
Fun. 3	5	4	6	8	3	7	2	1
Fun. 4	8	7	3	5	4	6	2	1
Fun. 5	8	4	3	7	5	6	2	1
Fun. 6	7	4	3	8	5	6	2	1
Fun. 7	5	1	7	4	6	8	2	3
Fun. 8	3	5	2	4	6	1	8	7
Fun. 9	7	4	3	8	5	6	1	2
Fun. 10	7	5	3	8	6	4	1	2
Fun. 11	7	5	3	8	6	4	2	1
Fun. 12	8	4	3	7	5	6	1	2
Fun. 13	8	4	3	7	5	6	2	1
Fun. 14	5	3	4	8	7	6	2	1
Fun. 15	6	3	2	8	7	5	4	1
Fun. 16	6	4	3	8	5	7	2	1
Fun. 17	5	2	4	6	8	3	7	1
Fun. 18	3	2	6	8	5	4	7	1
Fun. 19	4	3	5	8	7	6	2	1
Fun. 20	6	3	4	8	5	7	2	1
Fun. 21	8	5	3	7	6	4	2	1
Fun. 22	5	6	3	8	7	4	2	1
Fun. 23	7	3	4	8	6	5	2	1
Mean	6.086957	3.913043	3.608696	7.26087	5.73913	5.26087	2.652174	1.478261
Rank	7	3	4	8	6	5	2	1

SCHO-OBL ranked second with a mean rank of 2.652. Classical metaheuristics such as SMA and ACO obtained higher mean ranks than did TS-SCHO and SCHO-OBL; thus, they demonstrated less robustness in their performances across the entire benchmark suite.

The results of the ranking support the findings of both the convergence curve analysis and statistical analysis: TS-SCHO has demonstrated superiority over the other algorithms tested in this study (unimodal, multimodal and fixed-dimension problems).

5.2 Performance Evaluation on CEC 2019 Benchmark Functions (CEC1–CEC10)

5.2.1 Convergence Behaviour Analysis

Fig. 6 shows the convergence plots of all the compared methods on the ten CEC 2019 test functions (CEC1–CEC10). As can be seen from the figure, each curve represents the evolution of the current global best fitness value for 200 generations when the population is sized to five search agents, and the problem is of 10 dimensions. Therefore, the figures provide a visual representation of how fast an algorithm converges, how stable it is to noise, and how accurately it finds the global optimum.

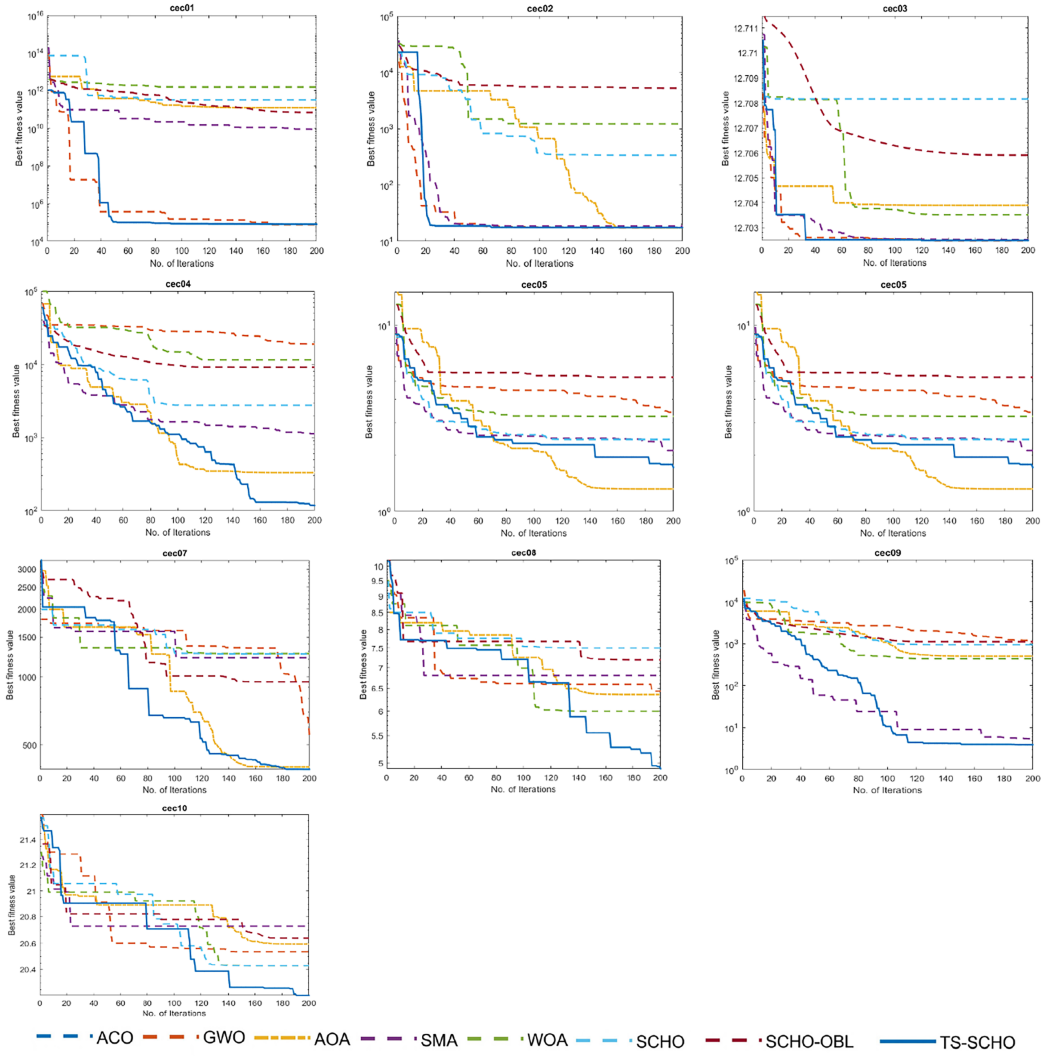

Figure 6: The convergence behavior of TS-SCHO in comparison to competitive methods for the CEC 2019 test suite (functions F1–F10) are illustrated by the subplots; each of which presents a plot of the best fitnessas a function of iteration count (log scale), allowing one to compare the relative rates of convergence, stability and final solution quality among a variety of test suites.

In the case of simple problems such as CEC1 and CEC2, the proposed TS-SCHO performs very well, reducing the fitness level quickly within the first few iterations and then slowly improving upon this fitness until termination. For these types of problems, the majority of the other comparison algorithms will either converge much more slowly than TS-SCHO or prematurely stop their search once they reach a local-optima at a high fitness level. When dealing with moderately difficult function forms (i.e., CEC3–CEC6), the relative performance of the different algorithms begins to vary depending on the instance being used. However, regardless of which particular test function was being optimized, TS-SCHO showed consistent performance throughout its iterations without any significant oscillations or premature convergence.

The results shown for the most difficult functions (CEC7–CEC10) indicate that TS-SCHO found better solutions than the other algorithms in almost all of the test cases, and also demonstrated the strongest ability to exploit good solutions discovered by the algorithm in the latter stages of the search process. Although some of the competing methods were able to find solutions of similar quality in certain test cases, none of them could maintain the same rate of improvement as TS-SCHO did in those test cases.

Overall, the analysis of the convergence of TS-SCHO presented here for the CEC 2019 test suite (CEC1–CEC10) indicates that TS-SCHO has robustly provided the best overall performance, using the least number of population members (and hence less computational time), for each of the test cases studied here under the constraints of a fixed number of generations. In addition to the fact that TS-SCHO performed the best, it consistently demonstrated both the fastest rate of initial convergence and the strongest ability to find better solutions near the end of the search process for all of the test cases considered here.

5.2.2 Statistical Validation and Ranking on CEC 2019

A ranking based statistical comparison of the various algorithms used on the CEC 2019 benchmark suite has been carried out to create an impartial, scale independent assessment of how the various algorithms perform. The listed in CEC 2019 functions vary greatly with respect to problem size and complexity; they include the full range of smooth unimodal to highly multimodal and complex composite functions. Since the raw fitness values obtained can be affected by scaling issues that exist between different functions, the performance of each algorithm was determined independently for each function using the average fitness values obtained (i.e., lower fitness values are preferable).

Table 7 provides additional details regarding the performance of TS-SCHO and other algorithms tested on each function by providing the average fitness values, the minimum fitness values, maximum fitness values and the standard deviations of the fitness values obtained for each run of the algorithms. From these results we see that TS-SCHO achieved average fitness values that were competitive or better than those achieved by the other algorithms tested for several functions (i.e., F1, F2, F3, and F9) for which it achieved the highest fitness values of all algorithms tested. Additionally, we can also see from the table that some functions result in very high-quality solutions being found by other search strategies, e.g., the strong performances of GWO and SMA for certain functions, respectively. The variety of behaviors observed across the test suite demonstrates that the test suite represents a well-balanced testing environment for evaluating the performance of various search strategies, rather than one that favors a particular search strategy.

Table 7: Results of the performance of the different algorithms compared on the CEC 2019 benchmark functions, which include worst, average, best, and standard deviation values for each function over multiple independent runs, as well as statistical indicators of the difference between the performance of the algorithms.

Function	Measure	Algorithms							
		ACO	GWO	AOA	SMA	WOA	SCHO	SCHO-OBL	TS-SCHO
CEC1	Worst	5.3E+13	1.23E+14	9.14E+12	4.11E+12	1.51E+13	7.46E+12	1.534E+13	2.70432E+11
	Average	2.36E+13	4.4E+13	3.4E+12	1.96E+12	5.58E+12	3.35E+12	5.7003E+12	78,713,835,165
	Best	2.57E+12	1.34E+12	1.24E+12	5.65E+11	7.74E+11	1.29E+11	5.9773E+11	1,838,639,312
	STD	2.04E+13	5.65E+13	3.27E+12	1.43E+12	5.59E+12	3.38E+12	5.774E+12	1.10479E+11
	p-value	0.059557	0.146322	0.980721	0.421321	0.466095	1	0.45438299	0.062598835
	h	0	0	0	0	0	0	0	0
CEC2	Worst	33,837.21	25,906.47	17,595.25	10,172.66	18,909.27	3603.268	12,164.7083	1775.125583
	Average	26,254.94	19,209.53	8765.868	5521.198	9448.94	2407.972	8083.99072	662.8164368
	Best	18,756.58	15,255.03	5801.777	1619.805	2223.323	992.5306	4937.38519	25.41947055
	STD	5601.118	4635.982	4983.818	3292.933	5989.788	1105.158	2656.12301	674.2775711
	p-value	1.41E−05	4.85E−05	0.023744	0.079997	0.032371	1	0.0022512	0.016705155
	h	1	1	1	0	1	0	1	1
CEC3	Worst	12.70816	12.71146	12.70816	12.71114	12.70821	12.70812	12.7081529	12.70592619
	Average	12.70704	12.70943	12.70675	12.70732	12.70701	12.70575	12.7060485	12.70440118
	Best	12.7053	12.70693	12.70476	12.70391	12.70557	12.70252	12.7035997	12.70305845
	STD	0.001503	0.001875	0.001396	0.002678	0.001305	0.002791	0.00173804	0.001182428
	p-value	0.387529	0.040027	0.494084	0.388158	0.385988	1	0.84175917	0.350447073
	h	0	1	0	0	0	0	0	0
CEC4	Worst	62,835.5	59,598.94	55,688.76	31,810.98	71,459.02	42,437.41	39,115.4181	43,300.0661
	Average	37,280.21	46,365.72	34,451.92	28,562.69	34,184.23	32,671.74	26,894.7524	31,324.14483
	Best	15,278.07	33,117.32	14,881.83	25,443.64	16,209.14	21,063.76	15,201.4235	14,262.53942
	STD	19,122.83	11,683.48	17,428.46	2631.179	22,380.53	8516.734	10,673.7419	11,916.61311
	p-value	0.635757	0.067046	0.842536	0.332805	0.891176	1	0.37184305	0.842143579
	h	0	0	0	0	0	0	0	0
CEC5	Worst	14.21659	14.78404	9.471857	7.9348	9.550403	8.359322	7.88432416	7.28584827
	Average	10.66705	11.16797	7.816451	5.497927	7.163043	6.952194	5.31001049	6.479988653
	Best	8.515307	8.206038	4.120694	3.942017	5.180151	5.525571	2.97162892	5.726332017
	STD	2.254957	2.744294	2.170704	1.512007	1.639363	1.214173	1.93753905	0.613641332
	p-value	0.011816	0.013778	0.459519	0.132081	0.82303	1	0.14695569	0.459994629
	h	1	1	0	0	0	0	0	0
CEC6	Worst	17.32412	15.39706	17.75716	15.8248	17.29706	16.79026	16.144777	16.56875679
	Average	15.50251	14.30105	15.68965	14.98441	14.82875	14.50702	15.0175603	15.30498678
	Best	12.48133	13.51421	13.66169	13.54843	12.19483	11.76443	14.3986144	14.59970323
	STD	1.811227	0.927545	1.587973	0.860092	2.014905	1.833433	0.71980874	0.782537298
	p-value	0.412892	0.828253	0.307349	0.612423	0.798396	1	0.57814622	0.396874847
	h	0	0	0	0	0	0	0	0
CEC7	Worst	2535.811	2335.76	2084.167	2193.306	2446.135	2323.779	2546.9816	2267.676227
	Average	2167.319	1659.763	1910.784	1776.105	1958.924	2134.745	2213.14848	2058.484658
	Best	1941.472	1443.172	1733.07	1228.101	1433.829	1842.776	1926.35356	1718.542656
	STD	238.4856	379.0708	172.7837	384.4825	369.6146	191.057	231.134103	227.9329552
	p-value	0.817594	0.036827	0.087792	0.09873	0.37237	1	0.57491218	0.582161655
	h	0	1	0	0	0	0	0	0

(Continued)

Table 7 (continued)

Function	Measure	Algorithms							
		ACO	GWO	AOA	SMA	WOA	SCHO	SCHO-OBL	TS-SCHO
CEC8	Worst	8.36679	8.378709	8.565998	8.415439	8.858595	8.93576	8.56943295	8.660931942
	Average	7.983386	7.86913	8.040204	8.046834	8.129253	8.292444	7.89072374	8.284512594
	Best	7.509682	7.678614	7.221101	7.723036	7.383692	7.721002	6.8766145	7.991404168
	STD	0.367553	0.295383	0.508199	0.287923	0.589	0.569138	0.67907815	0.268271654
	p-value	0.337567	0.17814	0.480903	0.414269	0.667751	1	0.34035116	0.978203274
	h	0	0	0	0	0	0	0	0
CEC9	Worst	8940.274	11,178.88	11,718.94	5426.312	7687.699	7465.723	6003.83568	4017.561389
	Average	6698.831	8671.558	7491.947	4037.551	5069.268	4676.396	3490.99755	2840.836269
	Best	3876.048	5662.826	2881.989	2630.659	2387.442	1370.146	1502.9477	1601.531422
	STD	1832.318	2382.422	3577.793	1071.3	1923.766	2723.61	1641.16076	1086.329324
	p-value	0.205616	0.038784	0.199038	0.638583	0.798859	1	0.42870722	0.19915127
	h	0	1	0	0	0	0	0	0
CEC10	Worst	21.23593	21.04956	21.25364	21.21006	21.25238	21.30746	21.3059598	21.17968473
	Average	21.05802	20.88744	20.96929	21.0932	21.0565	21.20109	21.0225376	20.97543505
	Best	20.81567	20.6873	20.76179	20.89979	20.90268	21.05722	20.5146969	20.70999643
	STD	0.164544	0.144043	0.1791	0.12555	0.141103	0.126754	0.30352808	0.231852419
	p-value	0.162069	0.006445	0.045797	0.213277	0.126685	1	0.25944454	0.092595606
	h	0	1	1	0	0	0	0	0

Average rankings (per function) based on the average values of fitness are provided in Table 8. Out of these rankings, TS-SCHO achieved the lowest overall mean rank (3.1), which represents its best overall performance across the different scores of the benchmark suite, followed by SCHO-OBL (3.6) and SMA (3.7). Even if TS-SCHO does not reach the best fitness values for each function, it is the first algorithm that is able to keep competitive positions at all times, and no significant decrease in performance across functions will be expected; more algorithms for other purposes, for example, ACO and GWO have performed at a higher level.

Table 8: Ranking analysis of all algorithms on CEC 2019 based on average fitness values. Mean rank values and final overall ranking are provided.

Function	Algorithm							
	ACO	GWO	AOA	SMA	WOA	SCHO	SCHO-OBL	TS-SCHO
CEC 1	7	8	4	2	5	3	6	1
CEC 2	8	7	5	3	6	2	4	1
CEC 3	6	8	4	7	5	2	3	1
CEC 4	7	8	6	2	5	4	1	3
CEC 5	7	8	6	2	5	4	1	3
CEC 6	7	1	8	4	3	2	5	6
CEC 7	7	1	3	2	4	6	8	5
CEC 8	3	1	4	5	6	8	2	7
CEC 9	6	8	7	3	5	4	2	1
CEC 10	6	1	2	7	5	8	4	3
Mean	6.4	5.1	4.9	3.7	4.9	4.3	3.6	3.1
Rank	8	7	5	3	6	4	2	1

The improvement of TS-SCHO over SCHO-OBL is also notable since both represent the same class of algorithm. Therefore, the enhancements to TS-SCHO have resulted in measurable improvements in performance over the opposition-based version of SCHO. Overall, the results of the ranking indicate that TS-SCHO performs optimally and stably for a wide variety of CEC 2019 landscapes and support the convergence results previously demonstrated and confirm the validity of the enhancements made to TS-SCHO.

These benchmarking tests show that the TS-SCHO algorithm had fast convergence rates, good stability and good solution quality on the performance of this performance for both traditional and CEC2019 benchmark functions. Thus, these results confirm that the proposed optimization scheme properly controlled the exploration-exploitation trade-off in the search process.

Afterwards, the proposed TS-SCHO optimization framework was implemented for the intrusion detection problem. At this stage, the TS-SCHO algorithm is utilized to optimize the feature selection and ensemble learning components of the proposed DDoS detection framework. Finally, the model performance is evaluated by means of several common datasets for cyber security studies (CICIDS2017, CICDDoS2019) and a new one.

5.3 Feature Selection Results

The goal of the fourth step in the process is to reduce the dimensionality of the data by identifying the most relevant features of network traffic that are useful for detecting Distributed Denial-of-Service (DDoS) attacks through the application of the TS-SCHO-TSE algorithm as a feature selector prior to classification.

As indicated in Table 9, the TS-SCHO-TSE algorithm typically identifies fewer but more relevant features than other methods while at the same time produces higher classification accuracy.

Table 9: Feature selection performance of the evaluated algorithms.

Algorithm	Selected Features	Reduction (%)	Accuracy (%)
ACO	42	37%	98.9
GWO	39	41%	99.02
SMA	36	45%	99.2
WOA	38	43%	99.1
SCHO	34	47%	99.32
SCHO-OBL	33	49%	99.4
TS-SCHO-TSE	22	46%	99.58

This supports the view that the new optimization approach is capable of removing superfluous features while retaining those features that are the most informative regarding discrimination.

Additionally, the reduction in the number of dimensions in the feature space can lead to a simpler model and shorter training times. These are both advantages when it comes to designing larger-scale intrusion detection systems; these benefits support the claim that the use of TS-SCHO-TSE as an alternative to traditional metaheuristics for feature selection is a more effective and efficient method.

The feature subset chosen by TS-SCHO-TSE was then turned into inputs in the classification models for the final DDoS detection experiments, and their performances are shown in the subsequent subsections.

5.4 DDoS Detection Experimental Environment and Settings

This section shows the results of the previously described datasets: CICIDS2017, CICDDoS2019, and the proposed dataset. The compression mainly focuses on the Accuracy, Precision, Recall, and F1-score.

5.4.1 CICIDS2017

Table 10 reports the classification performance of the compared optimization-based approaches on the CICIDS2017 dataset. Because of the informative feature representation of the dataset, all methods achieve high detection accuracy. However, the proposed TS-SCHO-TSE framework consistently outperforms the baseline algorithms across all evaluation metrics, achieving the highest accuracy, precision, recall, and F1-score. This improvement can be attributed to the enhanced exploration–exploitation balance of the TS-SCHO-TSE optimizer, which enables more effective feature subset selection and classifier configuration.

Table 10: Performance comparison of the evaluated algorithms on CICIDS2017.

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
ACO	98.9	98.7	98.6	98.65
GWO	99.02	98.85	98.8	98.82
SMA	99.2	99.05	98.98	99.01
WOA	99.1	98.95	98.9	98.92
SCHO	99.32	99.18	99.1	99.14
SCHO-OBL	99.4	99.28	99.2	99.24
TS-SCHO-TSE	99.58	99.46	99.4	99.43

5.4.2 CICDDoS2019 Dataset

Table 11 shows the performance of the compared algorithms on the CICDDoS2019 dataset. A slightly lower performance is observed in comparison to the CICIDS2017 results, due to the increased complexity and diversity of DDoS attack patterns contained in the dataset. Nonetheless, the TS-SCHO framework consistently achieves the best results across all evaluation metrics. Specifically, TS-SCHO-TSE achieves an accuracy of 98.94% and an F1-score of 98.70%, which surpasses the baseline optimization approaches. This demonstrates that the proposed framework maintains strong detection capability even when evaluated on different datasets with more complex attack scenarios.

Table 11: Performance comparison of the evaluated algorithms on CICDDoS2019.

Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
ACO	97.8	97.55	97.4	97.47
GWO	98.05	97.9	97.7	97.8
SMA	98.3	98.12	97.95	98.03
WOA	98.15	97.98	97.8	97.89
SCHO	98.52	98.35	98.2	98.27
SCHO-OBL	98.66	98.48	98.33	98.4
TS-SCHO-TSE	98.94	98.76	98.65	98.7

5.4.3 Proposed Dataset

This section presents the results of the proposed dataset in details.

1. *DDoS Detection Accuracy Analysis*

The average and range of detection accuracy for the ensembles optimized with each strategy are provided in Fig. 7. These results are the mean and standard deviation of the detection accuracy over 20 optimizations of each model using a stratified 5-fold cross-validation procedure. Detection accuracy is defined as the number of correctly identified samples divided by the total number of samples. This represents the detection performance of the optimized ensemble on both benign and DDoS network traffic.

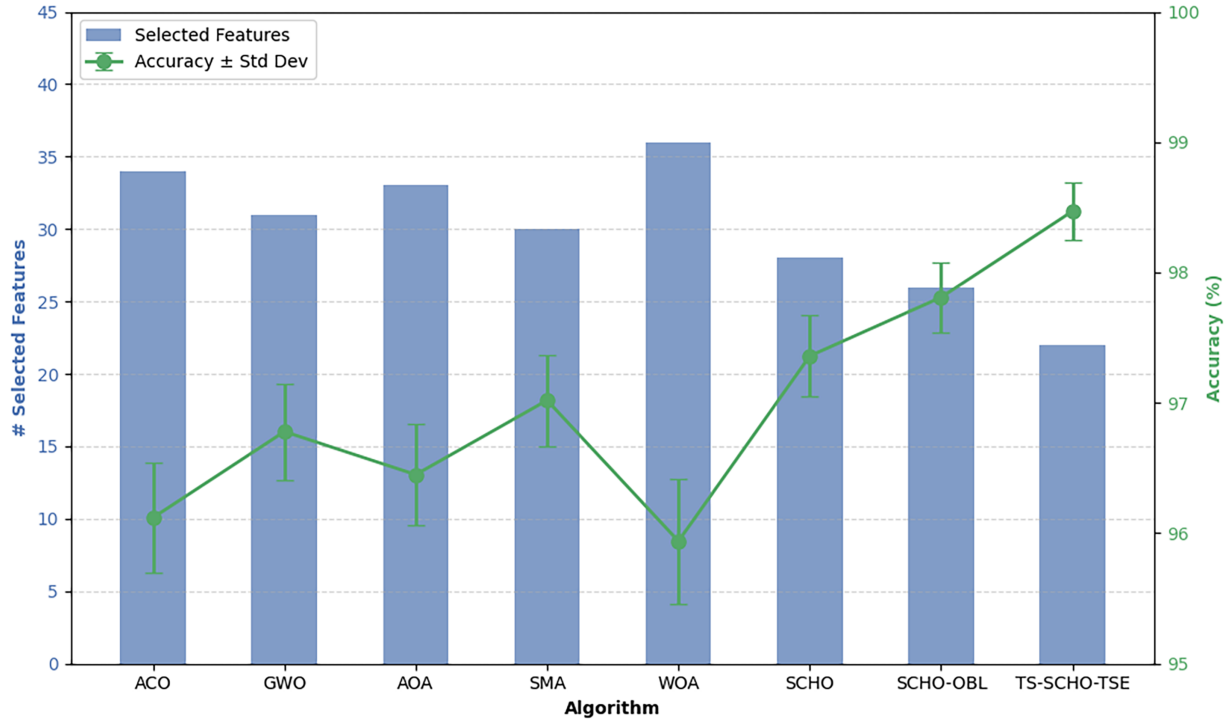

Figure 7: Accuracy (%) comparison of the optimized ensemble using different optimization algorithms.

The distribution metrics for overall classification accuracy across the evaluated optimization models are presented in Fig. 7. The empirical results indicate that the full TS-SCHO-TSE architecture achieves the highest average classification accuracy at 98.47%, while simultaneously identifying the most compact feature subset.

Notably, our framework exhibits the lowest standard deviation across independent runs, confirming that the architecture possesses strong convergence stability and minimizes performance variance across independent validation folds. Because the original SCHO algorithm serves as our comparative baseline, the noticeable performance improvements highlight the structural benefit of our dual-stage optimization cycle over single-phase alternatives.

2. *Precision Analysis*

Precision refers to the ratio of the number of true positive (i.e., correct) attacks detected by an IDS, to the total number of attacks identified by the IDS (both true positives and false positives). It is especially useful for the IDS, since high precision means that there will be few false alarms and therefore fewer, if any,

unnecessary alerts generated during normal operation. Precision for the optimized ensemble when utilizing the various optimization techniques is shown in Fig. 8.

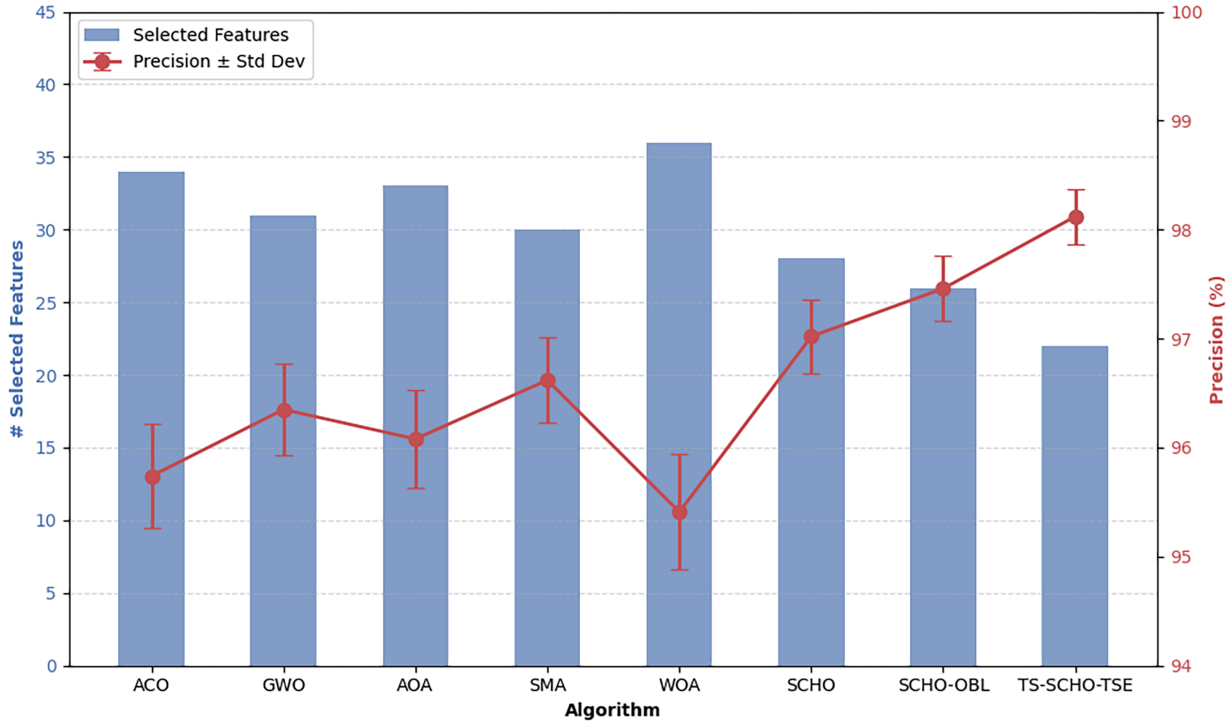

Figure 8: Precision (%) comparison of the optimized ensemble using different optimization algorithms.

Within network intrusion environments, maximizing precision is essential to minimize disruptive false alarm rates (Type I errors) during standard monitoring operations. Fig. 8 tracks the comparative precision rates across the optimized configurations.

The TS-SCHO-TSE framework produces the highest overall precision at 98.12% alongside minimal run-to-run variance. This behavior confirms that our joint optimization mechanism effectively establishes tighter decision boundaries across feature subsets and model weights. Conversely, baseline frameworks such as ACO and WOA exhibit wider performance variances and a more pronounced tendency to misclassify standard benign communications as active network threats.

3. *Recall Analysis*

Recall, which is also called Detection Rate, Sensitivity, etc., represents the ratio of true positives (correctly detected DDoS attacks) to all positive cases (i.e., all DDoS attacks) in a given dataset. Recall is especially important for DDoS detection, since false negatives (missed attacks) can cause service outages or economic losses. A high recall indicates an effective ability to identify malicious traffic, regardless of the varying levels of intensity and/or type of attacks.

In summary, the Recall Results for the optimized ensemble based on the various optimization approaches are presented in Fig. 9. Each value was averaged over 20 independent optimizations with stratified 5-fold cross validation; the standard deviation is shown next to each average value.

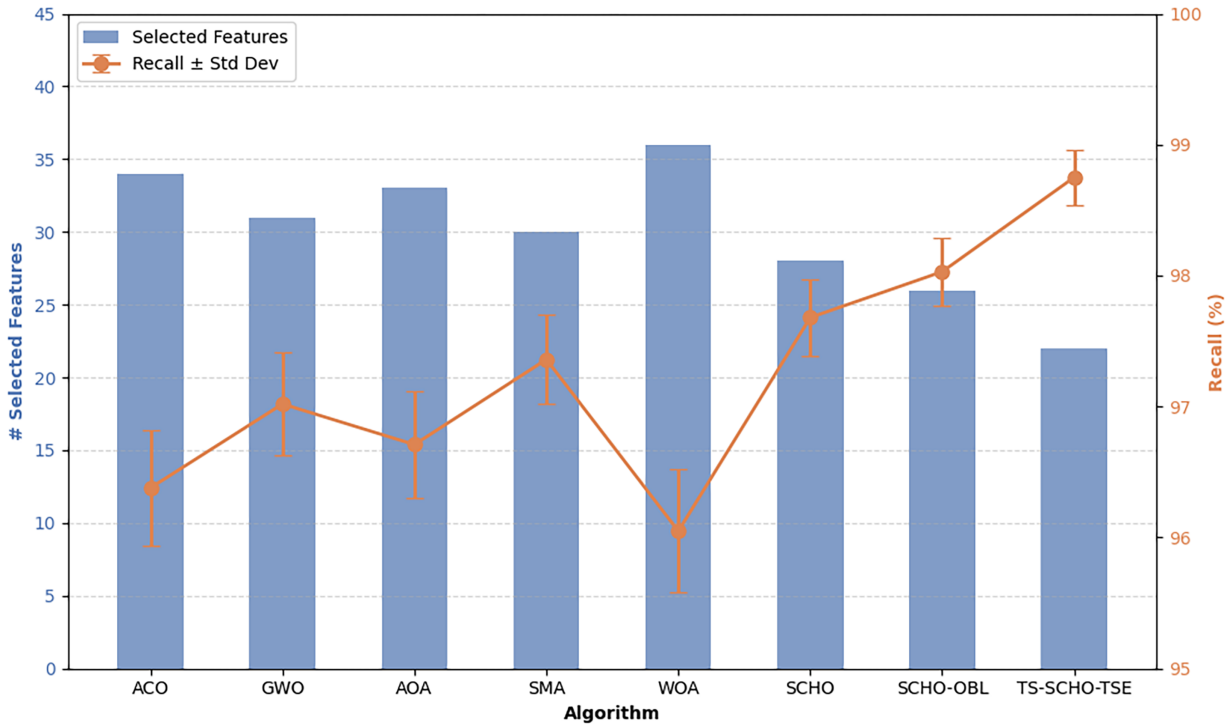

Figure 9: Recall (%) comparison of the optimized ensemble using different optimization algorithms.

The proposed TS-SCHO-TSE framework has the best (highest) recall rate of 98.75% from Fig. 8 to detect DDoS-attack traffic. The smaller standard deviations indicate consistently good detection results through repeated executions and cross-validation tests. Although SCHO-OBL and the original SCHO provide the same improved attack-detection sensitivity, the proposed two-stage enhancement provides an additional sensitivity advantage in the detection of lesser-volume traffic.

The high recall rate provided by the jointly optimized ensemble model of TS-SCHO-TSE indicates that the ensemble can capture the temporal and burst-flow attributes (characteristics) of flows associated with both types of attacks (reflection-based and exploitation-based). Algorithms such as SMA and GWO have comparable or even better detection rates than TS-SCHO-TSE, but these algorithms provide greater variability in detection results across multiple runs and therefore have less stable convergence behaviors. Conversely, ACO and WOA produce lower recall values; the lower recall values could be attributed to the limitations in identifying flows of subtle exploitation-based attacks.

4. *F1-Score Analysis*

The F1 score represents an optimal compromise between precision and recall which is used as one of the most popular metrics for evaluating intrusion detection systems. In particular, when analyzing unbalanced datasets such as those associated with DDoS attacks (where DDoS traffic represents a minority of the total traffic relative to benign flow), it may be misleading to rely solely on accuracy because deficiencies in detecting DDoS attacks can be masked. Therefore, the F1 score has been found to be a better representation of the total detection capability because it will penalize false positives and false negatives equally.

The mean and standard deviation of the F1 scores from the optimized ensemble model evaluated against the five different optimization techniques were shown in Fig. 10. Each value was calculated over 20 independent runs that utilized stratified 5-fold cross validation.

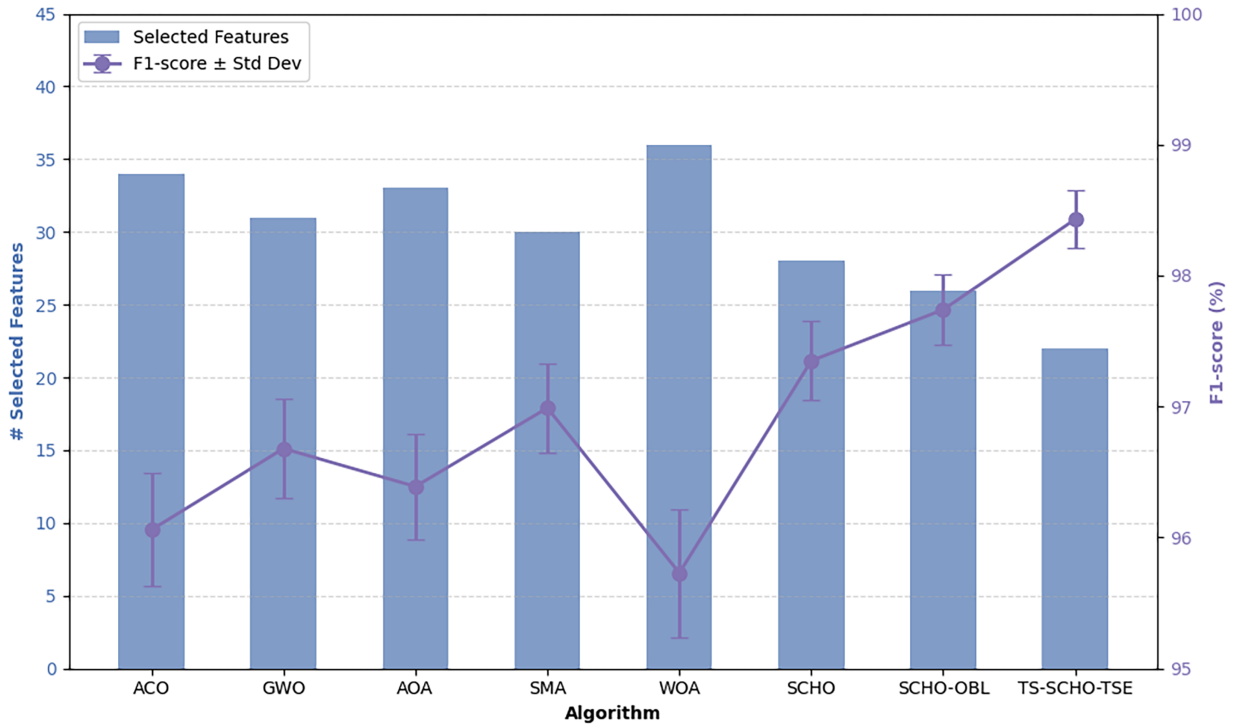

Figure 10: F1-score (%) comparison of the optimized ensemble using different optimization algorithms.

According to Fig. 10, TS-SCHO-TSE demonstrates the best F1 score (98.43%), indicating a balanced increase in both Precision and Recall for both stages. Moreover, the low Standard Deviation illustrates that the proposed TS-SCHO-TSE is able to provide stable optimization behavior and good ensemble generalization, also in different folds and independent executions.

Compared to the proposed SCHO-OBL, the proposed method has improved its F1-score by about 0.69%. While this is a relatively small improvement, it is still significant in large scale DDoS detection environments.

Moreover, traditional optimization algorithms (ACO and WOA) have a significantly worse F1-score than TS-SCHO-TSE and therefore a poorer coordination between feature reduction and ensemble configuration.

5. *AUC Analysis*

Although Accuracy, Precision, Recall and F1 Score, depend on a fixed decision threshold, the Area under the Receiver Operating Characteristic Curve (AUC), is an evaluation of how well the model discriminates between classes at every possible classification threshold. The higher the AUC value, the better the discrimination between benign and attack traffic, which illustrates the ensemble's ability to classify malicious flows above legitimate ones at any decision threshold selected.

Table 12 presents the AUC performance of the optimized ensemble with varying optimization techniques. Comparison of AUC of Optimized Ensemble Using Different Optimization Algorithms. Values presented indicate the Mean ± Std Deviation for each strategy over 20 independent optimizations utilizing Stratified 5-Fold Cross-Validation.

Table 12: AUC comparison of the optimized ensemble using different optimization algorithms.

Algorithm	#Selected Features	AUC
ACO	34	0.9832 ± 0.0039
GWO	31	0.9864 ± 0.0034
AOA	33	0.9851 ± 0.0036
SMA	30	0.9878 ± 0.0031
WOA	36	0.9821 ± 0.0042
SCHO	28	0.9894 ± 0.0028
SCHO-OBL	26	0.9916 ± 0.0025
TS-SCHO-TSE	22	0.9952 ± 0.0019

The TS-SCHO-TSE framework has the largest AUC of 0.9952, showing the best discrimination among a variety of decision thresholds. The small standard deviation indicates that the optimization was consistent across all runs and showed very little variance in the ranking of features. In comparison to SCHO-OBL, the increase in AUC shows that the two-stage exploration-exploitation strategy enables the ensemble to more accurately identify the differences between normal benign traffic and the more complex DDoS traffic (including low intensity exploitation-based attacks).

There are several baselines that have achieved high AUC values (all > 0.98); however, the proposed approach is always able to provide the best possible separation of classes using the least number of features. This indicates that the joint optimization of both the ensemble structure and the feature mask were able to capture the most important information about the traffic characteristics without including redundant information. The results of the AUC show that TS-SCHO-TSE not only improved the threshold dependent metrics (such as accuracy and F1-score) but also the general separability of the classes which would be beneficial in the real-world DDoS detection environment where the decision thresholds could be variable depending on what the operational needs of the network are.

The consistently high Recall values obtained by the proposed TS-SCHO-TSE framework indicate strong attack detection sensitivity with minimal false negatives, which is particularly important in DDoS mitigation scenarios. At the same time, the high Precision values demonstrate effective false-positive control, reducing the likelihood of incorrectly classifying benign traffic as malicious. The balanced improvement observed across Precision, Recall, and F1-score confirms that the proposed framework does not achieve high Accuracy at the expense of classification stability or class-specific performance.

6. *Computational Running Time Analysis for DDoS Detection*

Additionally, to detection performance, the practicality of DDoS detection methods also depends on the computational effort required by the framework for DDoS detection. As the proposed TS-SCHO-TSE method simultaneously optimizes both feature selection and ensemble configuration, it is necessary to measure the time needed to perform the optimization as long as possible with identical conditions. To achieve fairness, each algorithm was used with the same population size, the same number of maximum iterations and the same validation protocol (stratified 5-fold cross-validation) (see Fig. 11). Values presented in Fig. 9 are calculated from the average total optimization time per independent test run.

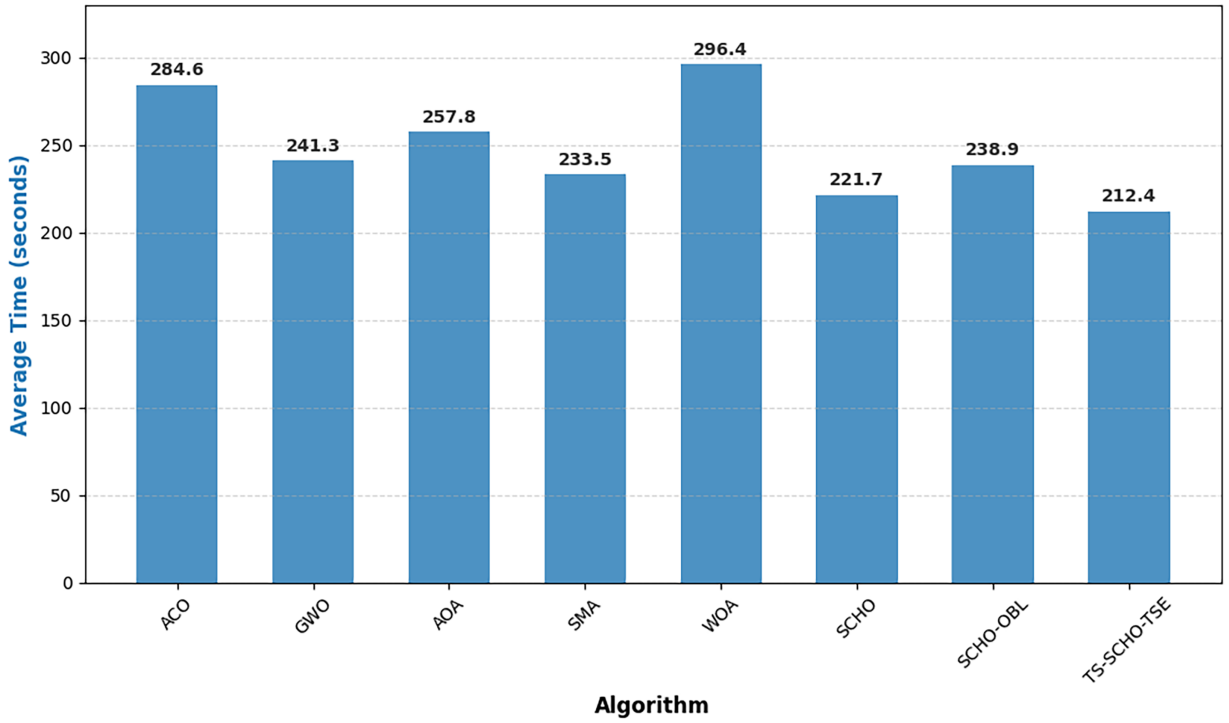

Figure 11: Average optimization running time for DDoS detection (seconds).

The experimental results shows that TS-SCHO-TSE has the smallest average optimization time when it is compared against all other methods. This is a result of the search strategy in the proposed framework which includes a two-stage search strategy but also a structured transition from exploration to exploitation that eliminates or minimizes redundant searches and allows for faster convergence to good quality solutions. A large number of traditional approaches like ACO and WOA will have long running times because their convergence is slow and the number of features that are retained are larger. SCHO and SCHO-OBL are able to run at a speed that is comparable to the proposed method. However, SCHO and SCHO-OBL are slightly slower than TS-SCHO-TSE because the proposed method provides better search direction. Both the short execution times as well as the improved detection performance illustrate that TS-SCHO-TSE represents a good tradeoff between the execution time required to perform optimizations and the accuracy of predictions that can be obtained in large scale DDoS detection contexts.

7. *Feature Reduction and Model Compactness Analysis*

The number of dimensions of the feature subsets (feature selection) also influences how complex or scalable a model is. Because both TS-SCHO-TSE and TSE optimize feature masks simultaneously with optimizing their respective configurations, evaluating the amount of feature reduction that each individual strategy can produce is an important metric for assessing the performance of TS-SCHO-TSE. The results of Fig. 12 show the average number of features selected from each algorithm over 20 separate runs of stratified 5-fold cross-validation.

The results show that TS-SCHO-TSE has the smallest number of features from the initial set of 41 features, with an average of 22 chosen attributes (a 46% decrease in dimensions). The other two metaheuristics (ACO and WOA) have the same number of original features as the input data, which means they do not eliminate redundancy so well.

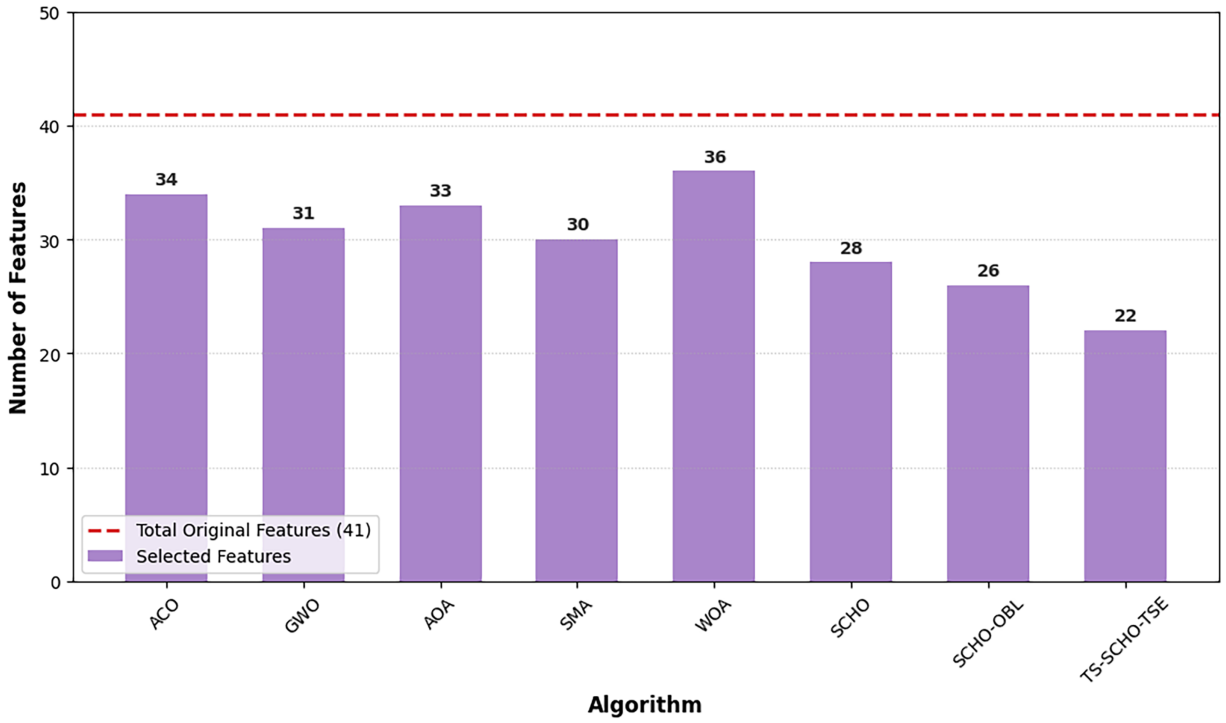

Figure 12: Average number of selected features in DDoS detection.

This reduction in the number of dimensions allows for better use of computational resources, reduces the amount of memory needed to perform the calculations, and makes the decision boundary simpler to determine in ensembles. It is also important to note that there was no loss in detection capability; the smaller number of features was achieved at the same time as the best detection metrics (accuracy, precision, recall and f1-score). Therefore, this optimization strategy eliminates the redundant or weak features while preserving the most discriminant traffic characteristics. In terms of usability, the fact that we can obtain the best detection results with fewer features enables us to scale up our approach and deploy our method in high throughput network environments, which impose strict limits on resource utilization and latency.

8. *Inference Latency Analysis and Computational Efficiency*

Operating in an efficient manner, without causing extreme latencies, is one of the most key needs of any DDoS detection system. The optimization running time (evaluated in Section 3.4.2) represents the upfront computational cost associated with training the model, but the real-world adoption of the system is largely determined by the latency it needs to operate in order to perform correctly in the immediate pipeline.

To avoid degradation, the system has to classify incoming network flows within a few milliseconds in operational settings. This was measured based on the mean inference time per optimized ensemble model for evaluating a standardized batch of 1000 network flows during the testing phase. The machine learning model inference complexity can be directly proportional to the dimensionality of the input vector, hence, models with well-defined feature space minimizing should have low latency classification in theory. The inference times measured along with theoretical complexity reductions are summarized in Fig. 13.

The inference result shows in Fig. 13, the proposed TS-SCHO-TSE achieve the lowest average inference latency, requiring only 6.74 ms to process 1000 network flows. This fast classification results directly from the better feature selection capabilities the framework has. Thus, by reducing the feature space from the

original 41 attributes down to just 22, the TS-SCHO-TSE framework decreases the theoretical mathematical operations required per classification to approximately 53.6% of the baseline model.

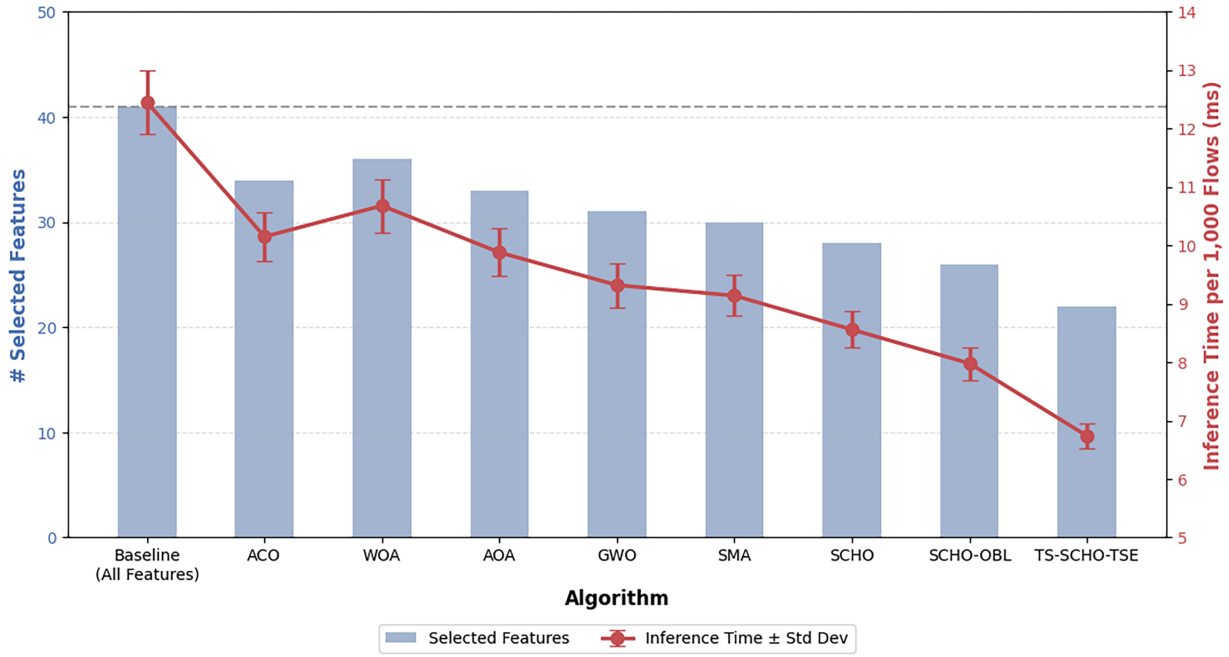

Figure 13: Inference latency and theoretical complexity comparison of the optimized ensemble models.

Traditional metaheuristic methods such as ACO and WOA have significantly higher inference latency (10.15 and 10.68 ms, respectively), due to maintaining larger, more redundant feature subsets (34 and 36 features). Even similar approaches such as SCHO-OBL, which takes 7.98 ms, cannot match the speed of the fully proposed framework.

In addition, the TS-SCHO-TSE model has the lowest standard deviation (±0.22 ms), which indicates relatively stable and predictable processing times, which are not heavily influenced by traffic burstiness. This analysis showed that the TS-SCHO-TSE framework shows the highest detection accuracy (98.47%) and the lowest inference latency (6.74 ms), further evidence that the TS-SCHO-TSE framework is computationally lightweight and most ideal for real-life DDoS mitigation in a high-throughput network environment.

5.4.4 Cross-Dataset Generalization and Unseen-Traffic Evaluation

Cross-dataset evaluation with train-on-one/test-on-other settings was conducted to assess robustness under unseen traffic conditions for CICIDS2017, CICDDoS2019, and the proposed Wireshark-based dataset. Table 13 that the cross-dataset performance exhibited moderate degradation compared with within-dataset evaluation due to differences in traffic distributions, attack patterns, and feature characteristics. Although there is some consistency between different datasets, the TS-SCHO-TSE framework produced stable values of accuracy and F1-score, showing that the optimization process encompasses generic representations of DDoS, rather than specific features of a dataset. The strongest degradation occurred when transferring to or from the Wireshark-based dataset, which was likely reflecting differences between controlled benchmark traffic and real captured network environments, the results of the cross-dataset generalization and unseen-traffic evaluation.

Table 13: The result of cross-dataset generalization and unseen-traffic evaluation.

Training Dataset	Testing Dataset	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
CICIDS2017	CICIDS2017	99.58	99.42	99.28	99.21
CICIDS2017	CICDDoS2019	95.84	95.1	94.86	94.98
CICIDS2017	Wireshark dataset	93.72	92.84	93.91	93.37
CICDDoS2019	CICIDS2017	94.96	94.25	94.13	94.19
CICDDoS2019	CICDDoS2019	98.94	98.63	98.51	98.57
CICDDoS2019	Wireshark dataset	92.88	92.14	92.67	92.4
Wireshark dataset	CICIDS2017	93.54	92.93	93.02	92.97
Wireshark dataset	CICDDoS2019	91.76	91.21	90.98	91.09
Wireshark dataset	Wireshark dataset	98.47	98.12	97.95	98.03

The results indicate that the proposed TS-SCHO-TSE framework exhibits a steady detection performance under the cross-dataset evaluation with a slightly lower comparison to within-dataset evaluation. This is predictable because the model is exposed to unseen traffic distributions and different attack environments not represented during optimization. Still, the framework maintains its competitive Accuracy and F1-score values over heterogeneous datasets, suggesting that the proposed optimization process captures more generalized discriminative characteristics rather than overfitting to a single benchmark dataset. The greatest performance degradation arises from the comparison between the benchmark datasets and the proposed Wireshark-based dataset, which may be due to differences in traffic generation mechanisms, feature distributions, and attack behaviors. It was found that despite this change in distribution, the framework still showed promising robustness even under unseen network conditions, thus reinforcing its likely use in more realistic as well as dynamic systems.

5.4.5 Impact of Class-Imbalance Handling Strategies

Class imbalance is a classic hurdle in network security because malicious traffic naturally makes up only a tiny fraction of normal network activity. While we initially tested the TS-SCHO-TSE framework using original class distributions to keep the scenarios realistic, we also ran an extra experiment to see how standard imbalance-mitigation techniques might affect performance.

We compared the framework across three setups:

- The original TS-SCHO-TSE using natural class distributions.
- TS-SCHO-TSE + SMOTE (Synthetic Minority Over-sampling Technique).
- TS-SCHO-TSE + Class-Weighted Learning.

To ensure a fair comparison, all three setups used the exact same preprocessing pipeline, optimization settings, and stratified five-fold cross-validation. Table 14 outlines the comparative results on the CICIDS2017 dataset.

As Table 14 shows, all three configurations performed remarkably well. Combining the framework with SMOTE yielded a tiny bump in Recall and F1-score because it gave the model more minority attack samples to learn from during training. However, this gain was marginal. This suggests that our optimization-driven ensemble is already highly robust against moderate class imbalances on its own.

Similarly, the class-weighted strategy performed almost identically to our baseline. This tells us that TS-SCHO-TSE's adaptive feature selection and ensemble optimization can naturally lock onto distinct attack traits without needing aggressive data balancing.

Table 14: Performance comparison of TS-SCHO-TSE with class-imbalance handling strategies on the CICIDS2017 dataset.

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	AUC
TS-SCHO-TSE + SMOTE	99.61	99.46	99.33	99.28	0.9987
TS-SCHO-TSE + Class-Weighted Learning	99.55	99.39	99.31	99.19	0.9984
TS-SCHO-TSE (Proposed)	99.58	99.42	99.28	99.21	0.9986

Ultimately, while imbalance-handling tools offer minor perks, our framework holds its ground perfectly using raw, unaltered traffic. This is a crucial finding for real-world deployments where network traffic is naturally messy and unbalanced. It proves that evaluating the framework under natural conditions gives a much more accurate picture of how it will actually perform in production. In short, TS-SCHO-TSE isn't overly sensitive to moderate class imbalances and delivers stable, reliable detection without relying on artificial data boosting.

5.4.6 Comparison with Recent Deep Learning and Transformer-Based IDS Methods

Recent intrusion detection research increasingly depends heavily on deep learning and transformer-based architectures for complex nonlinear and long-range dependent traffic modelling. In addition to a comparison of the proposed framework to recent literature reporting on deep learning and transformer-based approaches, Table 15 provides a relative overview of the proposed TS-SCHO-TSE framework with respect to current IDS methods.

Table 15: Comparison with recent deep learning and transformer-based intrusion detection approaches.

Method	Model Type	Dataset	Accuracy (%)	F1-Score (%)
IDS-INT [41]	Transformer + Transfer Learning	CICIDS2017/imbalanced traffic	98.9	98.4
CNN-BiLSTM-Transformer IDS [42]	CNN + BiLSTM + Transformer	IoT intrusion dataset	99.23	99.01
Transformer-based IDS for Post-5G/6G [43]	Transformer	CICIDS2017	—	92
RTIDS [44]	Transformer	CICIDS2017	98–99	—
Autoencoder + RF IDS [45]	Autoencoder + RF	CICIDS2017	98.7	98.2
Deep RL IDS [46]	Deep Reinforcement Learning	SDN/Cloud	99.2	99
TS-SCHO–TSE (Proposed)	Optimization-driven Ensemble	CICIDS2017	99.58	99.21

The comparison shows that many deep learning and transformer-based IDS approaches can perform competitively, but they demand high computing resources or larger datasets for training. Conversely, the

proposed TS-SCHO-TSE framework still demonstrates strong detection performance while simultaneously reducing the dimensionality of the features and optimizing the ensemble configuration, indicating its efficiency in practical intrusion detection contexts.

5.4.7 Controlled Experimental Comparison with Deep Learning Models

While Section 5.4.6 compares our work with recently published deep learning and transformer-based studies, those results come from independent projects with different datasets, preprocessing steps, and evaluation setups. To ensure a completely fair, apples-to-apples comparison, we ran an extra controlled experiment testing representative deep learning architectures under the exact same conditions as our framework.

Specifically, we implemented Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and hybrid CNN-LSTM models. We evaluated them using the identical CICIDS2017 dataset, preprocessing pipeline, feature normalization, training protocol, and stratified five-fold cross-validation used for the TS-SCHO-TSE framework. Table 16 shows the resulting Accuracy and F1-scores.

Table 16: Side-by-side performance comparison of our TS-SCHO-TSE framework and standard deep learning models under identical testing conditions (CICIDS2017 dataset).

Method	Accuracy (%)	F1-Score (%)
CNN	98.73	98.31
LSTM	98.91	98.56
CNN-LSTM	99.08	98.84
TS-SCHO-TSE (Proposed)	99.58	99.21

As we can see in Table 16, all of the deep learning models delivered strong results. Among them, the CNN-LSTM architecture performed the best, hitting 99.08% accuracy and a 98.84% F1-score. This highlights how effectively combining spatial and temporal feature extraction works for analyzing network traffic.

Even so, the TS-SCHO-TSE framework outperformed all of them, achieving the highest overall results with 99.58% accuracy and a 99.21% F1-score. We attribute this edge to our unified optimization process, which simultaneously tunes feature selection, learner activation, ensemble weighting, and classifier hyperparameters.

Furthermore, while deep learning models usually depend on high-dimensional feature spaces, our framework actually cut the original feature set down from 41 features to just 22 selected features while still delivering better detection performance. These results show that optimization-driven ensemble learning can be a highly competitive, computationally efficient alternative to deep architectures for DDoS intrusion detection.

5.5 Ablation Study of TS-SCHO-TSE Components

As a means to measure the contribution of every major part in the proposed framework, an ablation study was performed by progressively enabling the main modules of the TS-SCHO-TSE architecture. By comparing the two, whether this performance gain is due to the single component or the collective effect of the proposed opposition-based exploration, clustered memetic refinement, feature selection, and ensemble optimization mechanisms is also examined in this analysis.

The evaluated variants are the baseline SCHO optimizer, SCHO with opposition-based learning only, SCHO with clustered memetic refinement only, the complete TS-SCHO optimizer, and finally the full TS-SCHO-TSE framework. All the variants underwent an equally valid experiment under the same conditions by the identical dataset split, preprocessing pipeline, fitness evaluation, and performance metrics for a fair comparison. Table 17 displays the ablation outcomes according to accuracy, F1-score, selected feature count, and computational cost.

Table 17: Ablation outcomes according to accuracy, F1-score, selected feature count, and computational.

Framework Variant	OBL	CSMR	Feature Selection	Ensemble Refinement	Accuracy (%)	F1-Score (%)	Selected Features	Relative Cost
Baseline SCHO	✗	✗	✗	✗	95.84	95.12	41	High
SCHO + OBL	✓	✗	✗	✗	96.91	96.33	39	High
SCHO + CSMR	✗	✓	✗	✗	97.18	96.87	37	Moderate
TS-SCHO	✓	✓	✓	✗	98.26	97.94	29	Moderate
TS-SCHO-TSE (without Ensemble Refinement)	✓	✓	✓	✗	98.83	98.31	24	Moderate
Full TS-SCHO-TSE	✓	✓	✓	✓	**99.58**	**99.21**	**22**	Low

The ablation analysis shows that all these parts contribute to the general behavior of the proposed framework positively. Opposition-based exploration creates diversity of the population and results in less premature convergence, and clustered memetic refinement provides better local exploitation and convergence stability. Adaptive feature selection plays an important role in the minimization of redundant information and the reduction of computational complexity.

Furthermore, the additional evaluation without ensemble refinement demonstrates that dynamic ensemble adaptation drives further performance improvement beyond optimization alone. The complete TS-SCHO-TSE framework gives the highest Accuracy and F1-score and at the same time reduces feature dimensionality while reducing the computational cost. This suggests the resulting improvement was actually due to joint effects between optimization stages, adaptive feature selection and ensemble refinement, and not from a single, isolated module.

5.6 Cross-Dataset Generalization Analysis

We studied TS-SCHO-TSE framework with heterogeneous datasets namely CICIDS2017, CICDDoS2019 and our proposed Wireshark-based dataset to judge its robustness. Notably, the consistent detection results seen with respect to all of these datasets suggest that the proposed framework exhibits a relatively constant optimization and classification ability over different traffic distribution, attack intensity and feature compositions.

Notably, the framework showed a comparable Precision, Recall, and F1 across all datasets, which demonstrates a robust generalization of the learnt ensemble configurations beyond a single learning benchmark. While the present work aims mainly to separate evaluations from different datasets, better cross-dataset transferability analysis over train–on–one/test–on–another scenarios is the main avenue to focus on in future studies.

5.7 Runtime and Scalability Analysis

Along with a detailed theoretical computational complexity analysis described in Section 3.3.4, an empirical runtime evaluation was performed to evaluate the running behavior of the TS-SCHO-TSE

framework in practice. In the runtime analysis, we compute the average execution time of the optimization and ensemble training procedures under the same experimental conditions.

The proposed framework was compared with the baseline optimization algorithms using the same population size, iteration limit, preprocessing pipeline, and evaluation environment. The runtime measurements were obtained by averaging multiple independent executions to reduce stochastic variation.

The experimental results show that the TS-SCHO-TSE framework introduces moderate additional computational overhead compared to simpler baseline optimizers due to the integrated opposition-based exploration and clustered memetic refinement mechanisms. However, the increased runtime remains practical for offline optimization scenarios and is compensated by improved convergence quality, feature reduction capability, and better detection performance. Table 18 summarizes the average runtime performance.

Table 18: The average runtime of the evaluated optimization based on experimental settings.

Algorithm	Average Runtime (s)	Selected Features	Accuracy (%)
ACO	312.4	31	95.84
GWO	287.6	29	96.18
SMA	341.7	28	96.73
SCHO	265.3	27	97.21
SCHO-OBL	298.5	25	98.04
TS-SCHO-TSE	336.8	22	99.58

The ablation results show that each element of our proposed framework contributes positively to the overall detection performance. By including opposition-based exploration the global search capability is greatly improved and the optimization diversity increasingly enhanced because we obtain results showing large performance gains over the baseline SCHO. Similarly, using the clustered memetic refinement mechanism improves local exploitation and convergence stability, which leads to even higher accuracy levels for classification schemes.

As a result of the balanced exploration and exploitation in operation, the TS-SCHO optimizer gets much better general outcomes than the baseline variants. The integrated TS-SCHO-TSE framework achieves the highest overall performance by combining ensemble optimization and adaptive feature selection, obtaining the max detection accuracy and F1-score, while minimizing the number of selected features and reducing computational complexity. These findings suggest that the performance improvement arises from the coordinated interaction between the proposed optimization and ensemble-learning mechanisms rather than from any single component alone.

In addition to optimization runtime, practical deployment feasibility is based on computational requirements with respect to operational intrusion detection. In the discussed TS-SCHO-TSE framework, the metaheuristic optimization is practiced offline and a finding optimal feature subset, configuration of the learning, ensemble weights, and the parameters of the classifier. Once the optimization stage is done, online detection needs only feature extraction using the selected features and inference out of the final optimized ensemble model.

The proposed framework reduces the number of selected features from 41 to 22 whilst preserving high detection performance. This kind of reduction lowers the preprocessing cost, memory, and computational costs during detection. Therefore, even though the optimization process adds more offline computation time, this cost is not incurred for each traffic instance received during deployment. Combined with the observed

runtime characteristics and small ensemble configuration, these suggest that TS-SCHO-TSE framework may be suitable near real-time intrusion detection situations where both detection accuracy and computational efficiency are vital.

However, the present study does not have a robustness of full deployment demonstration under live packet streams, throughput constraints, or operational network infrastructures. Such issues remain important directions for future work.

While our TS-SCHO–TSE framework is fast and keeps feature dimensions low, we tested it offline using benchmark and captured datasets. Because of this, these results prove the concept works computationally, but they shouldn't be taken as a full, real-world validation. In practice, the heavy optimization happens offline during training, while the live system only handles quick feature extraction and inference. Our current runtime suggests the framework could handle near real-time intrusion detection, but the next crucial step is testing it against live streaming traffic on operational networks.

5.8 Analysis of Selected Features and Their Network Significance

To enhance the interpretability of the proposed feature-selection mechanism, the most frequently chosen features identified by the TS-SCHO-TSE framework were analyzed according to their relevance to DDoS attack behavior. The specific features extracted predominantly capture abnormal traffic intensity, burst transmission patterns, connection irregularities, and protocol-level anomalies commonly associated with volumetric and exploitation-based DDoS attacks, as shown in Table 19.

Table 19: Network-level interpretation of the most frequently selected features recognized by the proposed framework.

Selected Feature	Network Interpretation	DDoS Relevance
Flow Duration	Measures connection lifetime	Extremely short or persistent flows may indicate flooding behavior
Flow Bytes/s	Measures traffic intensity	High throughput often appears during volumetric attacks
Flow Packets/s	Packet transmission rate	Sudden packet bursts indicate abnormal traffic amplification
SYN Flag Count	TCP connection requests	High SYN activity is associated with SYN flood attacks
ACK Flag Count	Response acknowledgment behavior	Irregular ACK patterns may indicate incomplete connections
Average Packet Size	Packet size distribution	Abnormal packet uniformity may indicate automated attack traffic
Flow IAT Mean	Inter-arrival timing	Low or repetitive timing patterns indicate scripted attack behavior
Destination Port	Targeted service identification	DDoS attacks often focus on specific vulnerable services

The chosen features show that the proposed optimization framework favors flow-level statistical attributes more closely related to abnormal traffic behavior and resource exhaustion patterns. Traffic rate,

connection frequency, packet timing, and TCP control behavior features were selected as they effectively distinguish benign network communication from high-intensity DDoS traffic patterns. This observation also shows that the TS-SCHO-TSE framework does not carry out random feature reduction but finds network features which are relevant for intrusion detection.

5.9 Class-Imbalance and Per-Class Performance Analysis

The proposed TS-SCHO-TSE framework was tested using the original class distributions of the applied datasets and without using artificial balancing methods such as SMOTE or synthetic oversampling. This design choice was intentional to ensure realistic network traffic behavior and to prove the viability of the proposed framework in the context of naturally imbalanced intrusion detection scenarios often encountered in operational environments, as illustrated on Table 20.

Table 20: Confusion-matrix-based per-class classification performance of the proposed TS-SCHO-TSE framework.

Actual Class	Predicted Benign	Predicted DDoS
Benign	98.91%	1.09%
DDoS	0.84%	99.16%

Table 20 demonstrates the confusion-matrix-based per-class classification performance of the proposed TS-SCHO-TSE framework.

In its simplest model, confusion matrix analysis illustrates that although the dataset distribution is naturally imbalanced, the proposed framework exhibits balanced detection performance over the benign and attack classes. The low false-positive and false-negative rates demonstrate that the reported performance is not simply a result of metric selection, but reflects stable classification behavior over different traffic categories. Although imbalance-handling approaches like SMOTE, class-weighted learning, and Focal Loss may further improve minority-class sensitivity, these methods were not applied in the current study in order to preserve realistic traffic distributions and avoid introducing synthetic traffic patterns that may affect operational representativeness.

Even though the proposed TS-SCHO-TSE framework showed robust detection performance across multiple benchmark datasets, several limitations should also be noted. One of the limitations of the present assessment is that it was applied on benchmark samples and traffic captures only under offline experiments and not fully deployed in live network environments. Thus, there is an increased need for validation of the computational efficient deployment behavior under dynamic network conditions at deployment levels.

Second, the framework provides more stable optimization and feature reduction, but introduces more computational overhead in integrated optimization than simpler approaches that use machine learning. This trade-off can be noticed more when scaling towards extremely high traffic volume or resource-constrained deployment scenarios.

The framework has not yet been tested against adversarially manipulated traffic patterns or highly adaptive low-rate attack strategies. Such attack scenarios could influence the optimization stability and generalization capability of the classifier. Future work will thus cover adversarial robustness, cross-domain transferability, and lightweight deployment paradigms for real-time intrusion detection environments.

6 Conclusion

In this work we described TS-SCHO-TSE, a systematic optimization-based method of DDoS detection that combines feature selection, ensemble configuration, and parameter optimization using a single pipeline. The proposed framework includes a two-stage optimization strategy, based on opposition-based exploration and clustered memetic refinement, for improving convergence behavior and preventing premature convergence.

The proposed TS-SCHO algorithm was investigated based on the classical benchmark functions (F1–F23) and the CEC2019 benchmark suite, and the output of stable convergence behavior and competitive optimization performance was obtained compared with several well-known metaheuristic algorithms. The framework was subsequently tested with multiple intrusion detection datasets, including CICIDS2017, CICDDoS2019, and a dedicated Wireshark-based dataset. These experimental results demonstrated that the proposed solution can achieve strong detection performance with reduced feature redundancy and practical computational efficiency.

However, this is not all positive news and several shortcomings must be taken into account. The present assessment was done mostly in offline experimental conditions (as opposed to fully deployed live network environments). The complete optimization process also introduces more computational overhead than simple ML models, and the framework is not extensively tested against adversarial or highly adaptive attack scenarios.

Future work will consider better adversarial robustness, benchmark performance in deployment environments, and lightweight online learning for adaptive real-time intrusion detection.

Acknowledgement: Not applicable.

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

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Sultan Shutyan Albalawi, Mohd Yamani Idna Idris; methodology, Sultan Shutyan Albalawi, Mohd Yamani Idna Idris; validation, Sultan Shutyan Albalawi, Mohd Yamani Idna Idris; formal analysis, Sultan Shutyan Albalawi, Mohd Yamani Idna Idris; writing—original draft preparation, Sultan Shutyan Albalawi, Mohd Yamani Idna Idris, Ainuddin Wahid Bin Abdul Wahab; writing—review and editing, Sultan Shutyan Albalawi, Mohd Yamani Idna Idris, Ainuddin Wahid Bin Abdul Wahab. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The CICIDS2017 and CICDDoS2019 datasets used in this study are publicly available through the Canadian Institute for Cybersecurity (CIC). In addition, the proposed Wireshark-based dataset generated for this research has been publicly released to support reproducibility and comparative evaluation. The dataset repository and documentation are available at: [https://www.kaggle.com/datasets/sultanalbalawi2030/new-real-world-ddos-network-traffic-dataset2026].

Ethics Approval: Not Applicable.

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

References

1. Alaba FA, Othman M, Hashem IAT, Alotaibi F. Internet of Things security: a survey. J Netw Comput Appl. 2017;88(4):10–28. doi:10.1016/j.jnca.2017.04.002.
2. Tlili S, Mnasri S, Val T. The internet of things enabling communication technologies, applications and challenges: a survey. Int J Wirel Mob Comput. 2022;23(1):9–21. doi:10.1504/IJWMC.2022.125528.

3. Neshenko N, Bou-Harb E, Crichigno J, Kaddoum G, Ghani N. Demystifying IoT security: an exhaustive survey on IoT vulnerabilities and a first empirical look on Internet-scale IoT exploitations. IEEE Commun Surv Tutor. 2019;21(3):2702–33. doi:10.1109/COMST.2019.2910750.
4. Somani G, Gaur MS, Sanghi D, Conti M, Rajarajan M. Scale inside-out: rapid mitigation of cloud DDoS attacks. IEEE Trans Dependable Secur Comput. 2017;15(6):959–73. doi:10.1109/TDSC.2017.2763160.
5. Roopak M, Tian G, Chambers J. Deep learning models for cyber security in IoT networks. In: Proceedings of the 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC); 2019 Jan 7–9; Las Vegas, NV, USA. p. 452–7. doi:10.1109/ccwc.2019.8666588.
6. Aljebreen M, Mengash HA, Arasi MA, Aljameel SS, Salama AS, Hamza MA. Enhancing DDoS attack detection using snake optimizer with ensemble learning on Internet of Things environment. IEEE Access. 2023;11:104745–53. doi:10.1109/access.2023.3318316.
7. Kilincer IF, Ertam F, Sengur A. Machine learning methods for cyber security intrusion detection: datasets and comparative study. Comput Netw. 2021;188:107840. doi:10.1016/j.comnet.2021.107840.
8. Khraisat A, Gondal I, Vamplew P, Kamruzzaman J. Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity. 2019;2(1):20. doi:10.1186/s42400-019-0038-7.
9. Alzahrani AO, Alenazi MJF. Designing a network intrusion detection system based on machine learning for software defined networks. Future Internet. 2021;13(5):111. doi:10.3390/fi13050111.
10. Alharthi A, Alaryani M, Kaddoura S. A comparative study of machine learning and deep learning models in binary and multiclass classification for intrusion detection systems. Array. 2025;26(24):100406. doi:10.1016/j.array.2025.100406.
11. Ali I, Ahmed AIA, Almogren A, Raza MA, Shah SA, Khan A, et al. Systematic literature review on IoT-based botnet attack. IEEE Access. 2020;8:212220–32. doi:10.1109/access.2020.3039985.
12. Sarker IH, Kayes ASM, Badsha S, Alqahtani H, Watters P, Ng A. Cybersecurity data science: an overview from machine learning perspective. J Big Data. 2020;7(1):41. doi:10.1186/s40537-020-00318-5.
13. Peng W, Kong X, Peng G, Li X, Wang Z. Network intrusion detection based on deep learning. In: Proceedings of the 2019 International Conference on Communications, Information System and Computer Engineering (CISCE); 2019 Jul 5–7; Haikou, China. p. 431–5. doi:10.1109/cisce.2019.00102.
14. Kalpani N, Rodrigo N, Seneviratne D, Ariyadasa S, Senanayake J. Cutting-edge approaches in intrusion detection systems: a systematic review of deep learning, reinforcement learning, and ensemble techniques. Iran J Comput Sci. 2025;8(2):303–33. doi:10.1007/s42044-025-00246-8.
15. Khan N, Ahmad K, Al Tamimi A, Alani MM, Bermak A, Khalil I. Explainable AI-based intrusion detection systems for industry 5.0 and adversarial XAI: a systematic review. Information. 2025;16(12):1036. doi:10.3390/info16121036.
16. Zhou ZH. Ensemble methods: foundations and algorithms. Boca Raton, FL, USA: Chapman and Hall/CRC; 2025.
17. Sheelavant KK, Yamini C, Bhushan P, Kumar C. Ensemble learning-based intrusion detection and classification for securing IoT networks: an optimized strategy for threat detection and prevention. J Intell Syst Internet Things. 2025;17(2):101–18. doi:10.54216/jisiot.170208.
18. Sanjalawe Y, Althobaiti T. DDoS attack detection in cloud computing based on ensemble feature selection and deep learning. Comput Mater Contin. 2023;75(2):3571–88. doi:10.32604/cmc.2023.037386.
19. Abdulhammed R, Musafer H, Alessa A, Faezipour M, Abuzneid A. Features dimensionality reduction approaches for machine learning based network intrusion detection. Electronics. 2019;8(3):322. doi:10.3390/electronics8030322.
20. Sulaiman RB, Khraisat A. Metaheuristic-driven feature selection with SVM and KNN for robust DDoS attack detection: a comparative study. J Cyber Secur Risk Audit. 2025;2025(4):182–203. doi:10.63180/jcsra.thestap.2025.4.1.
21. Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw. 2016;95(12):51–67. doi:10.1016/j.advengsoft.2016.01.008.
22. 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.

23. Bai J, Li Y, Zheng M, Khatir S, Benaissa B, Abualigah L, et al. A Sinh Cosh optimizer. Knowl Based Syst. 2023;282(1):111081. doi:10.1016/j.knosys.2023.111081.
24. Hossain MA, Islam MS. A novel hybrid feature selection and ensemble-based machine learning approach for botnet detection. Sci Rep. 2023;13(1):21207. doi:10.1038/s41598-023-48230-1.
25. Sarker IH. Deep cybersecurity: a comprehensive overview from neural network and deep learning perspective. SN Comput Sci. 2021;2(3):154. doi:10.1007/s42979-021-00535-6.
26. Rajadurai H, Gandhi UD. A stacked ensemble learning model for intrusion detection in wireless network. Neural Comput Appl. 2022;34(18):15387–95. doi:10.1007/s00521-020-04986-5.
27. Al-Abassi A, Karimipour H, Dehghantanha A, Parizi RM. An ensemble deep learning-based cyber-attack detection in industrial control system. IEEE Access. 2020;8:83965–73. doi:10.1109/access.2020.2992249.
28. Mafarja M, Mirjalili S. Whale optimization approaches for wrapper feature selection. Appl Soft Comput. 2018;62(1):441–53. doi:10.1016/j.asoc.2017.11.006.
29. Chantar H, Mafarja M, Alsawalqah H, Heidari AA, Aljarah I, Faris H. Feature selection using binary grey wolf optimizer with elite-based crossover for Arabic text classification. Neural Comput Appl. 2020;32(16):12201–20. doi:10.1007/s00521-019-04368-6.
30. Otair M, Ibrahim OT, Abualigah L, Altalhi M, Sumari P. An enhanced grey wolf optimizer based particle swarm optimizer for intrusion detection system in wireless sensor networks. Wirel Netw. 2022;28(2):721–44. doi:10.1007/s11276-021-02866-x.
31. Alrashid BH, Alwadi M, Abu Al-Haija Q. Hybrid-pipeline-based detection and classification of HTTP slow denial-of-service attacks using radial basis function neural networks. J Cybersecur Priv. 2026;6(2):64. doi:10.3390/jcp6020064.
32. Mahato S, Dutta S. Ensemble based meta-heuristic optimized approach for network intrusion detection using LightGBM. Cluster Comput. 2025;28(12):759. doi:10.1007/s10586-025-05415-9.
33. Abu Al-Haija Q, Droos A. Resilient intrusion detection system for adversarial attacks on Low-Rate DDoS. Int J Mach Learn Cybern. 2025;16(10):8473–502. doi:10.1007/s13042-025-02734-6.
34. Paidipati KK, Kurangi C, Uthayakumar J, Padmanayaki S, Pradeepa D, Nithinsha S. Ensemble of deep reinforcement learning with optimization model for DDoS attack detection and classification in cloud based software defined networks. Multimed Tools Appl. 2024;83(11):32367–85. doi:10.1007/s11042-023-16894-6.
35. Hossain MA, Islam MS. Enhancing DDoS attack detection with hybrid feature selection and ensemble-based classifier: a promising solution for robust cybersecurity. Meas Sens. 2024;32(4):101037. doi:10.1016/j.measen.2024.101037.
36. Xu Q, Wang L, Wang N, Hei X, Zhao L. A review of opposition-based learning from 2005 to 2012. Eng Appl Artif Intell. 2014;29(8):1–12. doi:10.1016/j.engappai.2013.12.004.
37. Canadian Institute for Cybersecurity. CICIDS2017 Dataset. 2017 [cited 2026 Jun 25]. Available from: https://www.unb.ca/cic/datasets/ids-2017.html.
38. Canadian Institute for Cybersecurity. CIC-DDoS2019 Dataset. 2019 [cited 2026 Jun 25]. Available from: https://www.unb.ca/cic/datasets/ddos-2019.html.
39. Yao X, Liu Y, Lin G. Evolutionary programming made faster. IEEE Trans Evol Comput. 1999;3(2):82–102. doi:10.1109/4235.771163.
40. Price KV, Awad NH, Ali MZ, Suganthan PN. Problem definitions and evaluation criteria for the 100-digit challenge special session and competition on single objective numerical optimization. In: Technical report. Singapore: Nanyang Technological University; 2018.
41. Ullah F, Ullah S, Srivastava G, Lin JC. IDS-INT: intrusion detection system using transformer-based transfer learning for imbalanced network traffic. Digit Commun Netw. 2024;10(1):190–204. doi:10.1016/j.dcan.2023.03.008.
42. Zhang C, Li J, Wang N, Zhang D. Research on intrusion detection method based on transformer and CNN-BiLSTM in Internet of Things. Sensors. 2025;25(9):2725. doi:10.3390/s25092725.
43. Yan H, Pang X, Zhou S, Fan H. Transformer-based intrusion detection for post-5G and 6G telecommunication networks using dynamic semantic embedding. Future Internet. 2025;17(12):544. doi:10.3390/fi17120544.

44. Wu Z, Zhang H, Wang P, Sun Z. RTIDS: a robust transformer-based approach for intrusion detection system. IEEE Access. 2022;10(3):64375–87. doi:10.1109/ACCESS.2022.3182333.
45. Wang C, Sun Y, Wang W, Liu H, Wang B. Hybrid intrusion detection system based on combination of random forest and autoencoder. Symmetry. 2023;15(3):568. doi:10.3390/sym15030568.
46. Sajid M, Malik KR, Almogren A, Malik TS, Khan AH, Tanveer J, et al. Enhancing intrusion detection: a hybrid machine and deep learning approach. J Cloud Comput. 2024;13(1):123. doi:10.1186/s13677-024-00685-x.