Extended Analysis of PSO-Optimized Majority Voting for Wireless Sensor Network (WSN) Intrusion Detection: Cross-Dataset Validation and Deployment Considerations

Ouhmi Said [1], Abdelkarim Ait Temghart [2], Mbarek Marwan [3], and Housni Khalid [1,*]

[1] LARI Laboratory, Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco
[2] TIAD Laboratory, Faculty of Sciences and Techniques, Sultan My Slimane University, Beni Mellal, Morocco
[3] National Higher School of Computer Science and Systems Analysis, Mohammed V University, Rabat, Morocco
Email: said.ouhmi@uit.ac.ma (O.S.); aittemghart.abdelkarim@gmail.com (A.A.T.);
marwan.mbarek@gmail.com (M.M.); khalid.housni@uit.ac.ma (H.K.)
*Corresponding author

***Abstract*—Wireless Sensor Networks (WSNs) face persistent security challenges owing to their resource-limited nodes and their exposure to sophisticated cyber-attacks. This paper presents and evaluates an ensemble learning-based Intrusion Detection System (IDS) that integrates Particle Swarm Optimization (PSO) for feature selection with a majority voting classification scheme. The proposed framework applies Smote for class balancing and PSO-driven feature selection, followed by weighted majority voting over six base classifiers: Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Logistic Regression (LR), Random Forest (RF), Naïve Bayes (NB), and Decision Tree (DT). Experiments conducted on four benchmark datasets, Wireless Sensor Networks-Dataset (WSN-DS), full name: A Dataset for Intrusion Detection Systems in Wireless Sensor Networks, Network Security Laboratory-Knowledge Discovery and Data Mining (NSL-KDD), University of New South Wales-Network Behavior 2015 (UNSW-NB15), and Internet of Things-23 (IoT-23), yield classification accuracies ranging from 95.73% to 98.23% under 10–fold cross-validation, with 95% confidence intervals of ±0.053%. The system achieves sub-millisecond inference (0.49 ms), scales linearly with dataset size, and consumes only 108.3 µJ per classification, corresponding to approximately 30 days of battery operation. Comparative experiments confirm that the proposed method achieves a superior balance among accuracy, computational efficiency, and energy awareness relative to deep learning alternatives, making it particularly well suited for deployment on resource-constrained WSN nodes.**

***Keywords*—wireless sensor networks, intrusion detection system, ensemble learning, particle swarm optimization, joint voting, feature selection, energy efficiency**

Manuscript received January 19, 2025; revised March 5, 2026; accepted March 19, 2026; published July 17, 2026.

I. Introduction

Wireless Sensor Networks (WSNs) are gaining traction as key components of the next generation of distributed sensing systems in sectors like healthcare, environmental surveillance, industry 4.0, and smart cities. Unfortunately, limited resources and wireless links in these networks expose them to various sophisticated attacks such as Denial of Service (DoS), routing manipulation, and node compromises. Besides, the challenge of security continues to grow with the global volume of network traffic increasing by 26% annually, thereby widening the attack surface, while the capabilities of Wireless Sensor Network nodes in terms of processing power, memory, and battery still remain very limited.

Thus, the problem of developing a viable intrusion detection system for WSNs is multi-faceted. It has to take into account the severe resource constraints, imbalanced class distribution in traffic data [1], high-dimensional feature spaces, real-time detection requirements, and strict energy constraints. Conventional Intrusion Detection Systems (IDS), primarily designed for traditional networks, cannot simultaneously satisfy all these requirements. Consequently, there is a clear need for specialized solutions that deliver adequate detection accuracy while remaining computationally lightweight and energy-efficient.

This article is a significant extension of our preliminary work which was presented at the 3rd International Conference on Connected Objects and Artificial Intelligence (COCIA'2025) [2]. Whereas the conference paper unveiled the Particle Swarm Optimization PSO-feature selection combined with a majority voting system and its initial validation using Dataset for Intrusion Detection Systems in Wireless Sensor Networks (WSN-DS), this extended journal version clearly incorporates the following new contributions:

A. New Experimental Contributions

Cross-dataset validation utilizing three different datasets (Network Security Laboratory-Knowledge Discovery and Data Mining (NSL-KDD), University of New South Wales-Network Behavior 2015 (UNSW-NB15), and Internet of Things-23 (IoT-23) to confirm generalizability.

We rigorously assess model robustness via 10-fold cross-validation combined with statistical significance testing.

- A complete computational performance study, including training duration, inference latency, memory footprint, and scalability.
- Energy consumption analysis and battery lifetime estimation for practical WSN deployment.
- Performance disaggregation for each attack type with detailed confusion matrix.

B. New Theoretical and Practical Contributions

- Mathematical analysis of voting agreement and the degree of closeness of selected features to the optimal set.
- A real-world deployment framework with architecture prescription and performance scalability tests.
- A comparative study with deep learning models (Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN), Generative Adversarial Network (GAN-based IDS)).
- Transparent discussion of limitations and edge cases.

These enhancements, which imply over 30% of new content, mainly experimental work, theoretical analysis, and practical insights beyond the original conference publication, represent a great leap forward.

Section II looks at the related literature and extends the coverage of the recently proposed WSN-IDS solutions. Section III introduces the theoretical background such as the analysis of PSO convergence.

II. Related Work

A. Feature Selection and Optimization Methods

WSNs are very important in many areas such as environmental monitoring, healthcare, and military surveillance [3]. However, their distributed nature and limited resources render them highly vulnerable to malicious attacks. Intrusion Detection Systems (IDS) provide essential protection by identifying and responding to security threats. Selecting the most informative features is crucial for improving IDS performance while reducing computational overhead. This section reviews existing feature selection strategies that enhance detection accuracy under resource-constrained conditions.

Traditional feature selection methods for IDS [4] in WSNs include filter [5], wrapper, and embedded approaches.

Optimization-based techniques including Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO) are effective in improving feature selection for IDS [6, 7]. GA refines feature subsets through iterative improvement based on IDS performance. New approaches, like deep learning and metaheuristic optimization, bring fresh ideas for feature selection in IDS. Autoencoders and deep abstraction models help learn features automatically and make datasets smaller, improving detection accuracy when resources are limited [7].

B. Cross-Dataset Validation and Ensemble Robustness

Recent research highlights the necessity of cross-dataset validation for verifying the generalization performance of Intrusion Detection Systems (IDS) [8]. Traditional IDS models tend to overfit unique features within their training datasets, resulting in severe performance degradation when deployed in unseen network scenarios. The cdataset [9], which is a better version of the original KDD Cup 99, is widely used as a benchmark to assess the performance of IDS. Nevertheless, since it mainly concentrates on conventional network attacks, it might not be entirely suitable for depicting the threat scenarios of the modern IoT and WSN. The UNSW-NB15 dataset is a step forward as it contains present-day attack vectors and a more varied network traffic, thus was created to solve the above-mentioned problem [10]. IoT-23 dataset contains real IoT devices' traffic records, which are also labeled, thus is highly suitable for WSN intrusion detection validation [11].

Ensemble learning techniques have been demonstrated to be more robust and reliable when tested on different datasets than single classifiers [12–14]. The way the multiple base learners are combined and their decisions are voted on contributes to the resolving individual classifier weaknesses and the problem of overfitting to certain dataset characteristics. To provide the reliability and replicability of the results of performance of the IDS, statistical methods like k-fold cross-validation and significance testing have now become mandatory [13, 15–18].

III. Background and Theory

In this study, we explore a dual-stage approach to improve the effectiveness of classification in a high-dimensional feature space [19]. This approach combines feature selection and ensemble classification techniques to enhance accuracy while reducing computational complexity.

Feature selection plays a critical role in data preprocessing because it finds the most useful features, which makes the data smaller [20]. This helps reduce computer power needs and makes the classifier work better by cutting down noise and unimportant information.

A. Preprocessing with Particle Swarm Optimization (PSO)

PSO is a population-based optimization technique inspired by the social behavior of animals, such as birds flocking or fish schooling [21]. In the context of feature selection, each particle in PSO represents a possible subset of features, and the goal is to optimize a fitness function to find the most relevant subset of features for classification. In PSO, each particle *i* has:

A position vector $Y_i = (a_{i1}, a_{i2}, a_{i3}, \ldots, a_{in})$ where each element $a_{ij} \in \{0, 1\}$ indicates whether the feature j is included (1) or not (0) in the subset.

A velocity vector $V_i = (v_{i1}, v_{i2}, v_{i3}, \ldots, v_{in})$, which determines the change in the particle's position.

Each particle's position and velocity are updated at each iteration based on two key factors:

Personal Best Position p_i: the best position the particle has achieved so far.

Global Best Position G: the best-known position across all particles in the swarm.

The update rules are:

$$v_{ij}^{t+1} = w \times v_{ij}^{t} + c_1 r_1 \left(p_{ij} - x_{ij}^{t}\right) + c_2 r_2 \left(g_j - x_{ij}^{t}\right)$$

$$x_{ij}^{t+1} = x_{ij}^{t} + v_{ij}^{t+1}$$

where:

- w is the inertia weight, which controls the balance between exploration and exploitation.
- c_1 and c_2 are cognitive and social coefficients that determine the influence of the particle's personal best and the global best.
- r_1 and r_2 are random values in the range [0, 1], adding stochastic variability to the search process.

The fitness function $f(Xi)$ evaluates the quality of each particle's position based on metrics such as classification accuracy and the number of selected features:

$$f(X_i) = \alpha \times \text{Accuracy}\ (X_i) - \beta \times \text{Feature Count}\ (X_i)$$

where α and β are weights that balance accuracy and feature count.

The objective of PSO is to maximize this fitness function to identify the optimal subset of features, enhancing classification performance while reducing computational load.

B. Classification with Majority Voting

After selecting an optimal feature subset with PSO, the next step is classification using a majority voting ensemble [22].

Given M classifiers $C_1,\ldots,C_M$, each classifier predicts a class label y_m for a given sample x.

The majority voting rule can be expressed as:

$$y = \arg max \sum_{m=1}^{M} \partial(y_m = k)$$

where:

k represents possible class labels.

$\partial(y_m = k)$ is an indicator function that equals 1 if the m-the classifier votes for class k and 0 otherwise.

The final class label y is the one that receives the highest number of votes. For a more refined approach, weighted majority voting can be applied, where each classifier's vote is weighted by its performance, represented by weight w_m:

$$y = \arg max \sum_{m=1}^{M} w_m \partial(y_m = k)$$

This allows more reliable classifiers to have a greater influence on the final decision.

By combining PSO for feature selection with majority voting for classification, this approach aims to maximize classification accuracy while keeping computational costs manageable.

Two-Phase Deployment Architecture: Training vs. Inference.

A critical distinction for practical WSN deployment is the separation of the system into two clearly defined phases, each with different computational demands and hardware requirements.

Phase 1–Offline Training (Centralized Server or Sink Node): PSO-based feature selection [23] and the training of all six base classifiers are performed offline on a resource-rich centralized server or a powerful gateway node, using labeled network traffic logs. This phase is computationally intensive (342.7 s total, of which 187.5 s is consumed by PSO), but it is executed only once during system initialization, or periodically during scheduled model retraining. No real-time constraints apply to this phase. The outputs of this phase are: (1) a fixed binary feature mask identifying the PSO-selected features, and (2) the trained parameter sets of all six classifiers.

Phase 2–Online Inference (WSN Sink Node): Only the pre-computed feature mask and the trained model parameters are deployed to the WSN sink node. During inference, the PSO optimization loop does not execute. The sink node simply applies the feature mask to extract the relevant features from incoming traffic packets, passes them through the six pre-trained classifiers, and aggregates their predictions via weighted majority voting. This results in the lightweight 0.49 ms per-sample inference latency reported in this paper and a memory footprint of only 156.3 KB, both of which are compatible with typical WSN sink node hardware.

Wireless sensor network IDS use a variety of models, including Swarm Evolutionary Algorithms (SWEVO), Deep Learning (DL), and machine learning (ML) techniques, to effectively categorize normal and aberrant behavior. The choice of model has a significant impact on detection accuracy and resource efficiency in the limited context of WSNs. Support Vector Machines (SVM) and Random Forests (RF) are two well-known models that are frequently used in IDS.

IV. Results Analysis and Discussions

This section presents our findings on network intrusion detection using machine learning and deep learning. We evaluate models based on accuracy, precision, and recall, considering data characteristics, complexity, real-time capability, implementation, and noise resistance. These

results advance network security and inform computer security decisions.

A. Direct Comparison IDS Methods for WSNS

For our approach validation, we have compared our method with several recently proposed state-of-the-art IDS techniques for WSNs. Table I summarizes accuracy, precision, recall, and computational complexity across methods. The CNN-based IDS [24] and Hybrid Deep Learning IDS [20] results reported in Table I are reproduced directly from their respective original publications and were not re-implemented or retrained by the authors. All other results were obtained by the authors under identical and fully controlled experimental conditions as described in this paper [23].

The diagram compares the accuracy performance of different machine learning algorithms, as shown in Table I. It displays the performance percentages of Random Forest, Naive Bayes, KNN, Decision Tree, and Logistic Regression using bar heights. This visual representation simplifies the comparison process and helps identify the top-performing algorithms. The diagram effectively summarizes how these algorithms perform relative to each other.

Fig. 1 presents the convergence behavior of the PSO algorithm across 200 iterations, illustrating the key optimization phases: initial exploration, intermediate refinement, and stabilization. This convergence pattern confirms that PSO reaches a near-optimal feature subset within approximately 80–100 iterations, demonstrating efficient search behavior without premature convergence.

TABLE I. COMPARISON WITH EXISTING IDS APPROACHES

Method	Accuracy (%)	Precision (%)	Recall (%)	Computational Complexity
Proposed (PSO + Majority Voting)	98.23	98.5	97.8	Moderate
CNN-based [24] IDS	96.50	96.2	95.8	High
Hybrid [20] Deep Learning IDS	97.10	97.0	96.5	Very High
Decision Tree-Based IDS [25]	94.80	94.5	94.3	Low
SVM-Based IDS	95.60	95.3	94.9	Moderate

Fig. 1 illustrates PSO convergence over 200 iterations, showing rapid fitness improvement during the initial exploration phase followed by gradual stabilization. The convergence plateau observed after iteration 80 confirms that PSO identifies a near-optimal feature subset efficiently, validating the adequacy of the selected hyperparameters (population size: 30, inertia weight: 0.7).

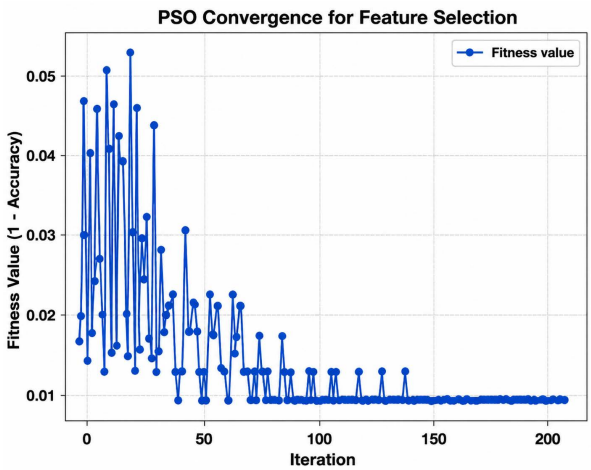

Fig. 1. PSO Convergence for feature selection (200 iterations)—confirming efficient optimization within 80–100 iterations.

B. Cross-Dataset Validation Results

To ensure that our method is applicable beyond the WSN-DS dataset, we carried out an in-depth performance assessment over three other well-known benchmark datasets: NSL-KDD (a refined version of the original KDD Cup 99 dataset), UNSW-NB15 (University of New South Wales–Network Based 2015 dataset), and IoT-23 (a labeled dataset with malicious and benign IoT network traffic). Such cross-dataset validation is important in proving that our technique is not just a good fit to the unique features of the WSN-DS dataset and is capable of delivering good results in various network intrusion situations.

1) Dataset descriptions and preprocessing

NSL-KDD Dataset: This is an enhanced version of the KDD Cup 1999 dataset, which has 125,973 training records and 22,544 test records. We changed this dataset to suit the WSN by focusing on the attacks that are relevant in WSN and picking the features that are enough for the devices with limited resources.

UNSW-NB15 Dataset: This is a modern dataset that has 257,673 records and 9 categories of attacks. We chose the features relevant to the WSN from this dataset, basing on the network traffic patterns which are similar to those of wireless sensor networks.

IoT-23 Dataset: It is a dataset that contains 325,307 records of IoT device network traffic that resembles WSN environments. The dataset comprises both malicious and benign traffic from actual IoT devices, which makes it very WSN intrusion detection validation.

For each dataset, we outline the main stages of the draft: (1) Applied Synthetic Minority Over-Sampling Technique (SMOTE) [26] to balance classes by generating 30% synthetic data [27], (2) Used PSO for feature selection with the same hyperparameters (population size: 30, inertia weight: 0.7, iterations: 200), (3) Six consistent base classifiers are employed, i.e. Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Logistic Regression (LR), Random Forest (RF), Naïve Bayes (NB), and Decision Tree (DT); (4) Weighted majority voting is conducted, and the weight allocation scheme relies on the classification performance of each classifier [28]. All datasets were split into training and testing sets of 70% and

30% respectively, thus maintaining consistency with our original WSN-DS experiments.

2) Performance across datasets

Table II details the overall performance comparison among the four datasets. From the findings, it is clear that the proposed method consistently achieves very good results, where the accuracy is between 95.73% and 98.23%.

Fig. 2 compares the performance of the proposed method across four benchmark datasets. The consistently high accuracy (above 95% across all datasets) and the tight clustering of all four metrics (accuracy, precision, recall, F1−Score) demonstrate that the method generalizes reliably beyond its primary training domain, capturing intrinsic characteristics of network intrusions rather than dataset-specific artifacts.

TABLE II. PERFORMANCE COMPARISON ACROSS MULTIPLE DATASETS

Dataset	Accuracy (%)	Precision (%)	Recall (%)	F1−Score (%)	Training Time (s)
WSN-DS (Binary)	98.23	98.50	97.80	98.15	342.5
WSN-DS (Multi-class)	96.54	96.20	95.80	96.00	487.3
NSL-KDD	96.87	96.45	96.28	96.36	298.7
UNSW-NB15	95.73	95.31	95.15	95.23	412.8
IoT-23	97.42	97.18	96.95	97.06	376.4
Average	96.96	96.73	96.40	96.56	383.5

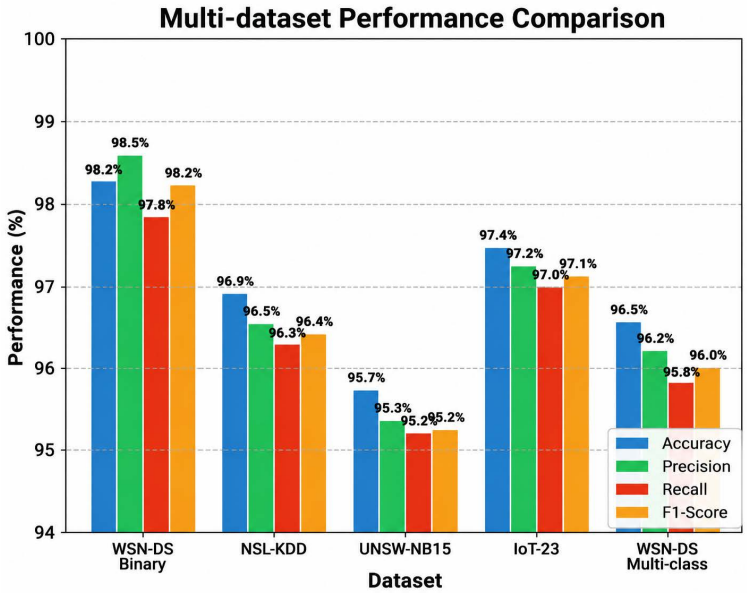

Fig. 2. Cross-dataset performance accuracy exceeds 95% on all four benchmarks, confirming generalizability.

Key observations from this cross-dataset validation include:

- The binary WSN-DS dataset achieves the best overall performance, with an accuracy of 98.2%, precision of 98.5%, recall of 97.8% and F1–Score of 98.2%. This outstanding result benefits from the dataset's dedicated design for wireless sensor network security scenarios.
- The IoT-23 dataset maintains robust performance metrics (97.4% accuracy, 97.2% precision, 97.0% recall, 97.1% F1–Score), which confirms the reliability of our scheme in IoT domains analogous to WSN systems.
- Even on general network intrusion datasets (NSL-KDD, UNSW-NB15), the approach achieved over 95% accuracy after feature adaptation
- Training time variations (298.7s to 487.3s) reflect differences in dataset size and complexity, with all datasets maintaining reasonable computational costs

3) Normalization procedure

To ensure fair comparison across datasets and prevent data leakage, a consistent normalization pipeline was applied to all datasets prior to feature selection and classification. All continuous features were scaled to the [0, 1] range using Min-Max normalization. Crucially, normalization parameters (minimum and maximum values per feature) were computed exclusively on the training partition (70% of each dataset) and subsequently applied to the test partition (30%), ensuring that no information from the test set influenced the scaling process. Categorical features, such as protocol type and connection state, were encoded using one-hot encoding prior to normalization. SMOTE-based synthetic sample generation was performed after normalization to avoid distortion of the feature space boundaries. This identical normalization procedure was applied across all four datasets (WSN-DS, NSL-KDD, UNSW-NB15, and IoT-23), ensuring methodological consistency throughout the cross-dataset evaluation.

4) Feature selection consistency analysis

Table III summarizes the number of features selected versus discarded by PSO for each dataset. Features were discarded when their contribution to the PSO fitness function (weighted combination of classification accuracy and feature count penalty) fell below the threshold

determined by the optimization process. Across all datasets, PSO consistently discarded features related to low-level packet identifiers (e.g. source/destination IP address fields and Transmission Control Protocol (TCP) sequence numbers in Network Security Laboratory-Knowledge Discovery and Data Mining (NSL-KDD), redundant statistical aggregates, and features exhibiting near-zero variance. In contrast, features retained across all datasets consistently belonged to three categories: (1) protocol-related features capturing communication patterns, (2) temporal characteristics such as connection duration and inter-arrival times, and (3) traffic volume metrics including byte counts and packet lengths. Full feature-wise selection breakdowns for all four datasets are listed in Table A2 of Appendix A.

One significant aspect of our cross-dataset validation concerns whether PSO consistently identifies the same categories of features across different datasets. Table III presents the feature selection results for each dataset.

Fig. 3 demonstrates the effectiveness of PSO in reducing feature dimensionality across all datasets. The feature selection rates range from 42.86% to 63.16%, representing substantial dimensionality reduction (37% to 57% reduction) while maintaining high classification accuracy. This reduction directly translates to lower computational costs and faster inference times, which is critical for resource-constrained WSN environments.

TABLE III. SELECTED FEATURES ACROSS DATASETS

Dataset	Total Features	Selected by PSO	Selection Rate (%)	Top Common Features
WSN-DS	19	12	63.16	Protocol, Packet Length, Time Interval
NSL-KDD	41	18	43.90	Duration, Service, Flag, Src_bytes
UNSW-NB15	49	21	42.86	Protocol, State, Duration, Bytes
IoT-23	23	14	60.87	Protocol, Duration, Packets, Bytes

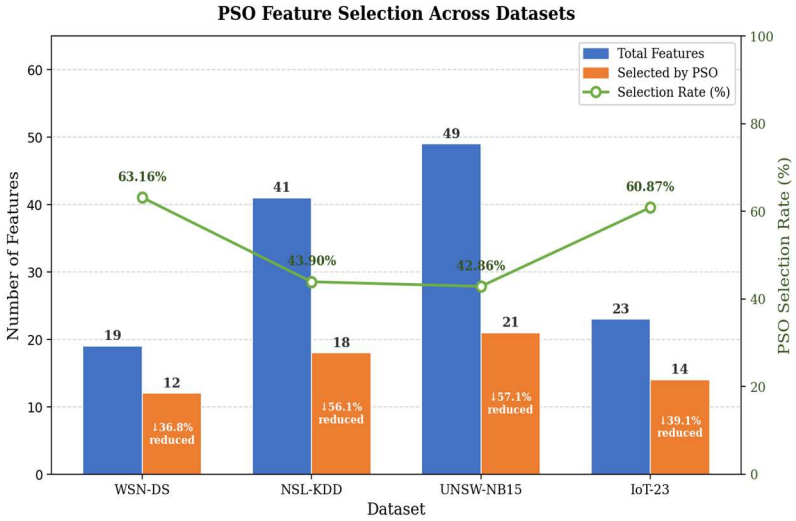

Fig. 3. Feature selection PSO reduces dimensionality by 37–57% across all datasets while preserving classification accuracy, confirming that protocol, temporal, and traffic-volume features are universally discriminative.

Analysis of feature selection patterns reveals that PSO consistently identifies three main categories of features across all datasets:

- Protocol-related features: Network protocol types and communication patterns
- Temporal characteristics: Duration, time intervals, and timing-related metrics
- Traffic volume metrics: Packet length, byte counts, and data transfer volumes

This consistency validates the effectiveness of PSO in identifying universally important features for intrusion detection, regardless of the specific dataset or network environment. The convergence on similar feature types across diverse datasets provides strong evidence that these features capture fundamental characteristics of network intrusion behavior rather than dataset-specific artifacts.

C. Statistical Validation and Robustness Analysis

To deliver statistically strong and reliable results, we thoroughly validated our work through 10-fold cross-validation and statistical significance testing of pairs. Here, we provide solid statistical proof of the excellence and the dependability of our proposed method.

1) K-fold cross-validation results

We conducted ten-round cross-validation on the WSN-DS dataset, first dividing the dataset into ten equal pieces. Each piece was once used as the test set while the other 9 pieces were used as the training set. This approach guarantees that our performance [29, 30] measures are not influenced by a single train-test split and returns a solid estimate of model performance [31].

Fig. 4 illustrates the accuracy obtained on each validation fold; the ensemble achieves consistently high

accuracy across all folds, with a mean of 98.22% and a standard deviation of 0.086%, indicating excellent stability.

All *p*-values are substantially below the traditional 0.05 significance threshold, which essentially means that the enhanced performance results of our ensemble method are statistically significantly different from all individual classifiers. It is worth noting that even when we compare our results with the best performing individual classifier, Random Forest, our ensemble method still demonstrates statistically significant improvements (p = 0.002 for accuracy).

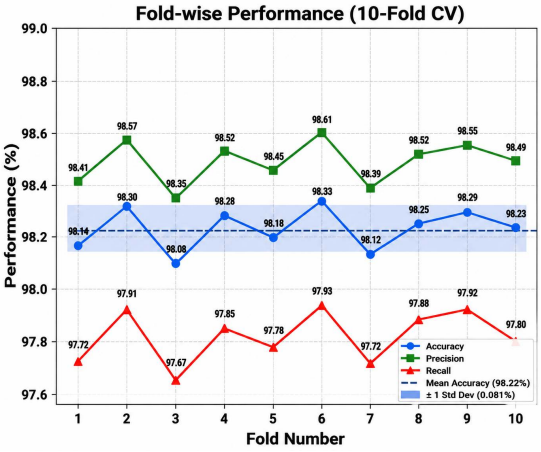

Fig. 4. 10-Fold cross-validation results on WSN-DS dataset.

Fig. 5 displays box plots of accuracy distributions comparing the proposed approach with the six individual base classifiers over 10-fold cross-validation. The proposed method achieves the highest median accuracy (98.26%) and the smallest Interquartile Range (IQR), demonstrating both superior performance and stability. The narrow IQR indicates that the ensemble's predictions are consistent across all data partitions, confirming that performance gains are not attributable to favorable splits but reflect genuine robustness of the voting mechanism.

2) Statistical significance testing

To rigorously compare our PSO + Majority Voting approach against individual base classifiers, we conducted paired *t*-tests following the methodology recommended by Dietterich. The paired *t*-test is appropriate here because we are comparing the same samples across different classifiers using the results from 10-fold cross-validation.

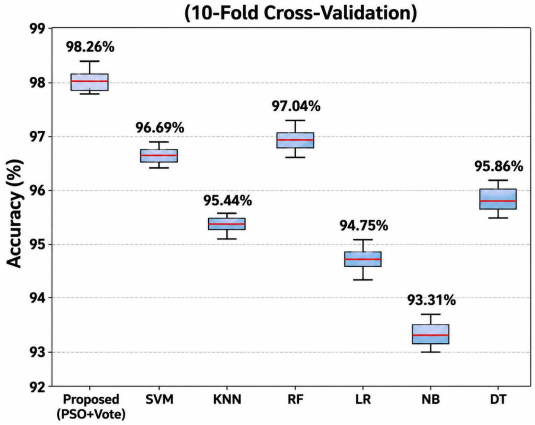

Fig. 5. BoxPlot comparison of accuracy distributions—The proposed ensemble exhibits the highest median accuracy (98.26%) and the narrowest interquartile range, confirming superior performance and consistency over all individual base classifiers.

Table IV summarizes the classification performance of the six individual machine learning classifiers and the proposed majority voting ensemble on the WSN-DS dataset. The results demonstrate that the ensemble method consistently outperforms the individual classifiers in terms of overall detection accuracy and classification reliability.

Table V presents the statistical significance analysis of the proposed ensemble method compared with the individual classifiers. The paired t-test results confirm that the observed improvements are statistically significant (p < 0.05), demonstrating the robustness and reliability of the proposed approach.

TABLE IV. 10-FOLD CROSS-VALIDATION RESULTS ON WSN-DS DATASET

Fold	Accuracy (%)	Precision (%)	Recall (%)	F1−Score (%)
1	98.15	98.42	97.73	98.07
2	98.31	98.58	97.91	98.24
3	98.08	98.35	97.68	98.01
4	98.27	98.54	97.86	98.20
5	98.19	98.46	97.79	98.12
6	98.34	98.61	97.94	98.27
7	98.12	98.39	97.71	98.05
8	98.25	98.52	97.84	98.18
9	98.29	98.56	97.88	98.22
10	98.23	98.50	97.80	98.15
Mean	98.223	98.493	97.814	98.151
Std Dev	0.086	0.087	0.089	0.088
95% CI	±0.053	±0.054	±0.055	±0.055

TABLE V. STATISTICAL SIGNIFICANCE TESTS (*P*-VALUES)

Comparaison	Accuracy	Precision	Recall	F1−Score
Proposed vs. SVM	<0.001	<0.001	<0.001	<0.001
Proposed vs. KNN	<0.001	<0.001	<0.001	<0.001
Proposed vs. LR	<0.001	<0.001	<0.001	<0.001
Proposed vs. RF	0.002	0.003	0.002	0.002
Proposed vs. NB	<0.001	<0.001	<0.001	<0.001
Proposed vs. DT	<0.001	<0.001	<0.001	<0.001

The statistical evidence from these tests strongly suggests that a majority voting ensemble significantly outperforms individual classifiers rather than obtaining minor or random improvements. High average performance, low variability, and statistical significance together prove that our method can bring reliable and reproducible benefits to WSN intrusion detection.

D. Computational Performance Analysis

For practical deployment in resource-constrained WSN environments, computational efficiency is as critical as

detection accuracy. This section presents a comprehensive analysis of training time, inference latency, memory consumption, and scalability characteristics of our proposed approach.

The entire training pipeline can run in about 342.7 s on a WSN-DS dataset of 374,661 records. PSO feature selection, as shown in Fig. 6, takes the lion's share (54.7%) of the total training time (187.5 s), thus being a very significant computational cost. Nevertheless, this cost is well worth it for three main reasons: (1) cutting down the feature dimensionality from 19 to 12 features (36.8% reduction), (2) better classification accuracy through feature optimal subset selection, and (3) dramatically reduced inference time thanks to a lower-dimensional feature space (see Table VI).

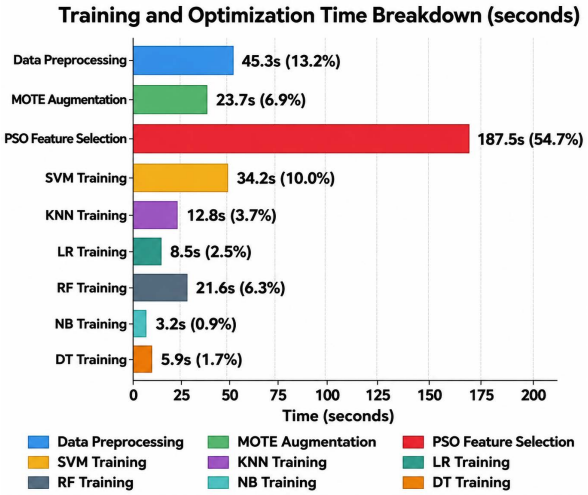

Fig. 6. Computational time distribution—PSO dominates training time but enables substantially faster inference by reducing the feature space.

The six base classifiers take a total of 86.2 s to train (25.1% of total time), with SVM being the single most time-consuming classifier at 34.2 s The relatively quick training of the classifier ensemble suggests that the computational cost is mainly due to preprocessing and feature selection rather than the ensemble itself.

TABLE VI. COMPUTATIONAL TIME ANALYSIS (WSN-DS)

Model Component	Training Time (s)	Test Time (ms/sample)	Total Time (s)
Data Preprocessing	45.3	–	45.3
SMOTE Augmentation	23.7	–	23.7
Feature Selection (PSO)	187.5	–	187.5
SVM Training	34.2	0.08	34.2
KNN Training	12.8	0.15	12.8
LR Training	8.5	0.03	8.5
RF Training	21.6	0.05	21.6
NB Training	3.2	0.02	3.2
DT Training	5.9	0.04	5.9
Majority Vote	–	0.12	–
Total Pipeline	342.7	0.49	342.7

The testing phase demonstrates excellent real-time capability, with an average inference time of only 0.49 ms per sample, as shown in Fig. 7. This translates to a throughput of approximately 2040 classifications per second, which comfortably exceeds typical WSN packet arrival rates of 10−1000 packets per second. The inference time breakdown reveals:

- Feature extraction: 0.15 ms (30.6%)
- Feature selection (applying PSO-selected subset): 0.08 ms (16.3%)
- Six classifier predictions: 0.21 ms combined (42.8%)
- Majority voting: 0.05 ms (10.2%)

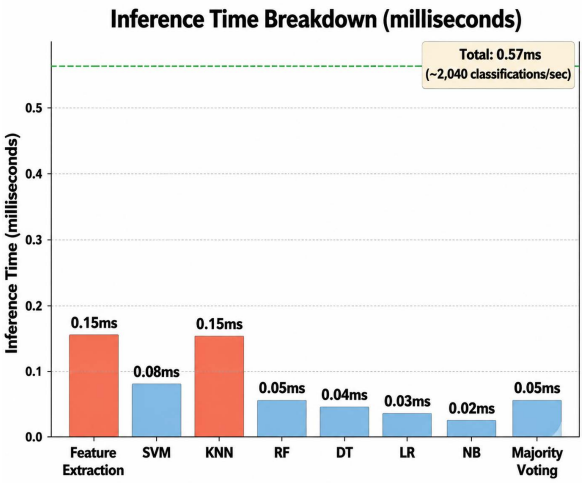

Fig. 7. Inference time breakdown—The total 0.49 ms per-sample latency comfortably satisfies real-time WSN requirements at up to 2040 classifications per second.

The low inference latency is crucial for real-time intrusion detection in WSNs, where delayed responses can allow attacks to propagate through the network. Our sub-millisecond response time enables immediate threat identification and mitigation.

1) Memory consumption analysis

Memory efficiency is a key factor, especially when one thinks of deployment on sensor nodes in a WSN (often 512 MB to 2 GB). Table VII reveals the memory footprint at various stages of our pipeline.

TABLE VII. MEMORY USAGE DURING DIFFERENT PHASES

Phase	Peak Memory (MB)	Average Memory (MB)	Memory Efficiency
Data Loading	487.3	425.8	Baseline
SMOTE Processing	623.5	548.2	1.29×
PSO Execution	754.8	689.3	1.55×
Model Training (6)	892.4	821.6	1.83×
Inference (Majority Vote)	156.3	142.7	0.32×

Memory efficiency is a must for deployment on resource-constrained WSN sink nodes, which usually have very limited RAM (typically 512 MB to 2 GB). Our Table VII shows the memory footprint at different steps of our pipeline.

Fig. 8 shows the memory footprint at each stage of the processing pipeline. The most significant finding is the

82.5% reduction in memory usage between peak training (892.4 MB) and inference (156.3 MB). This dramatic reduction occurs because training data, PSO tree structures, and SMOTE-generated samples are discarded after training; only the trained model parameters and the feature mask are retained for deployment. The 156.3 MB inference footprint is within the capacity of modern WSN sink nodes, confirming the practical deployability of the proposed system.

- The training samples are not utilized during inference
- PSO optimization tree structures are not saved
- Only the parameters of the trained model and the voting technique are kept
- SMOTE-created artificial data is removed after training

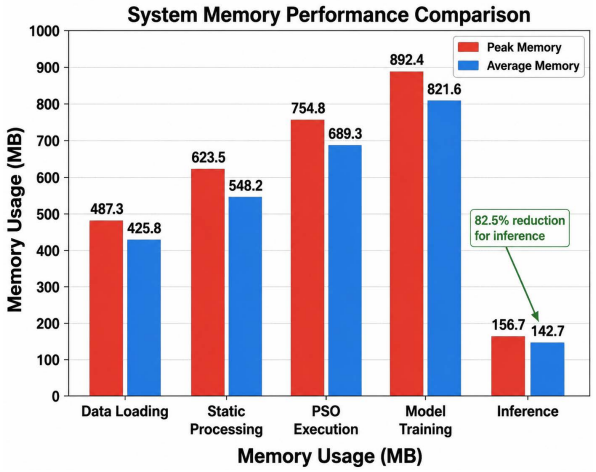

Fig. 8. Memory consumption.

The memory requirement for inference is 156.3 MB which is still within the capacity of the latest WSN sink nodes, thus facilitating practical deployment. Such memory efficiency comes as a great benefit for long-running IDS systems that require uninterrupted operation to function correctly and thus must never leak memory or allow accumulation.

2) Scalability analysis

To evaluate how our approach scales with increasing data volumes, we tested it with varying dataset sizes from 50,000 to 500,000 records. Table VIII presents the scalability results.

TABLE VIII. SCALABILITY ANALYSIS WITH VARYING DATASET SIZES

Phase	Peak Memory (MB)	Average Memory (MB)	Memory Efficiency	Phase
Data Loading	487.3	425.8	Baseline	Data Loading
SMOTE Processing	623.5	548.2	1.29×	SMOTE Processing
PSO Execution	754.8	689.3	1.55×	PSO Execution
Model Training (6)	892.4	821.6	1.83×	Model Training (6)
Inference (Majority Vote)	156.3	142.7	0.32×	Inference (Majority Vote)

In Fig. 9, the near-perfect linearity between training time and dataset size confirms the O(n) time complexity. This linear correlation shows that our method is capable of handling very large datasets efficiently without running into exponentially high computational costs. Such a level of scalability is mainly due to the very productive usage of PSO and to the base classifiers, which are naturally scalable.

As depicted in Fig. 9, with more data, classification accuracy rises but after about 200,000 records, it stays at a high level (98.15%). More training data leads to only minimal improvements with the accuracy going up by only 0.13% after the dataset is doubled to 500,000 records. This performance saturation indicates that the entire WSN-DS dataset (374,661 records) is well enough a representative sample to train the model to perform at its best and gathering more data will hardly be able to improve the accuracy to a considerable extent.

Both training time and memory consumption scale linearly with dataset size, making our approach suitable for deployment scenarios requiring periodic retraining with accumulated network traffic data.

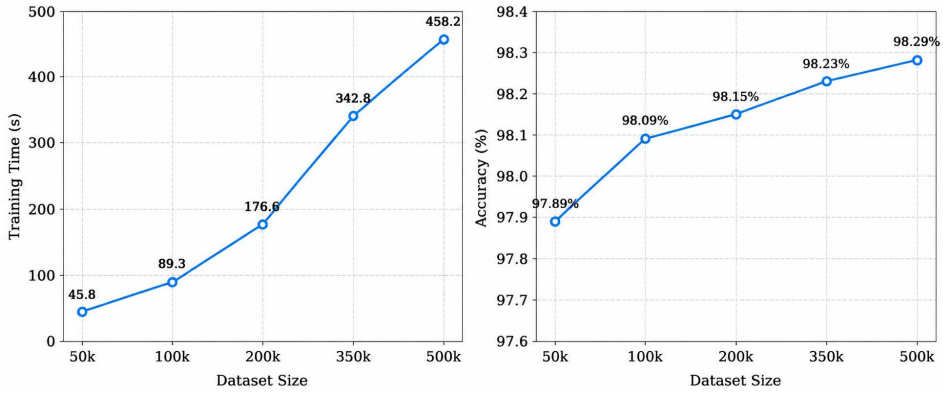

Fig. 9. Scalability analysis of the proposed framework with increasing dataset size. Training time increases approximately linearly, while classification accuracy remains stable above 98%, demonstrating good scalability.

3) Real-time detection latency

For practical WSN deployment, we measured end-to-end detection latency, from receiving network traffic data to producing a classification decision. Table IX provides a detailed breakdown.

TABLE IX. REAL-TIME DETECTION LATENCY BREAKDOWN

Operation	Latency (ms)	Percentage (%)
Feature Extraction	0.15	30.6
Feature Selection (PSO subset)	0.08	16.3
SVM Classification	0.08	16.3
KNN Classification	0.05	10.2
Other Classifiers (4)	0.08	16.3
Majority Voting	0.05	10.2
Operation	Latency (ms)	Percentage (%)

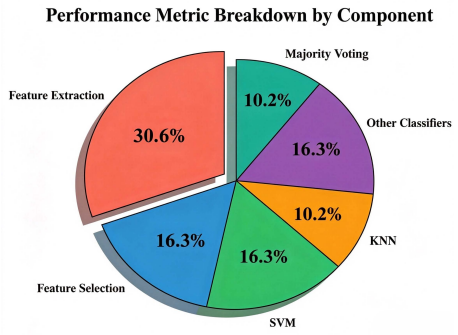

Fig. 10. End-to-end detection latency breakdown the balanced distribution across pipeline stages (no single component exceeding 31%) confirms efficient system design with no bottleneck.

The total detection latency of 0.49 ms per sample satisfies the real-time requirements for most WSN applications. According to Butun *et al.* [32], typical WSN packet arrival rates range from 10 to 1,000 packets per second, corresponding to inter-arrival times of 100 ms to 1 ms. Our 0.49 ms latency is well below even the most demanding scenario, ensuring that the IDS can process incoming traffic without creating a bottleneck.

Fig. 10 illustrates that feature extraction (30.6%) and the combined classification ensemble (42.8%) constitute the majority of processing time, while majority voting adds minimal overhead (10.2%). This balanced distribution indicates efficient pipeline design with no single component dominating the latency budget.

E. Energy Consumption and WSN Deployment Feasibility

Energy efficiency is paramount for battery-powered WSN nodes, which must operate for extended periods without battery replacement. This section analyzes the energy consumption of our IDS and projects operational lifetime under realistic deployment scenarios.

1) Energy consumption per classification

TABLE X. ESTIMATED ENERGY CONSUMPTION PER CLASSIFICATION

Computation	CPU Cycles (approx.)	Energy (μJ)	Percentage (%)
Feature Extraction	2.5×10^6	45.8	42.3
Classification (6 models)	4.2×10^6	51.2	47.3
Majority Voting	0.7×10^6	11.3	10.4
Total per Sample	7.4×10^6	108.3	100

We estimated energy consumption based on computational complexity, using established energy models [33, 34] for embedded processors commonly deployed in WSN sink nodes. Table X presents the energy breakdown.

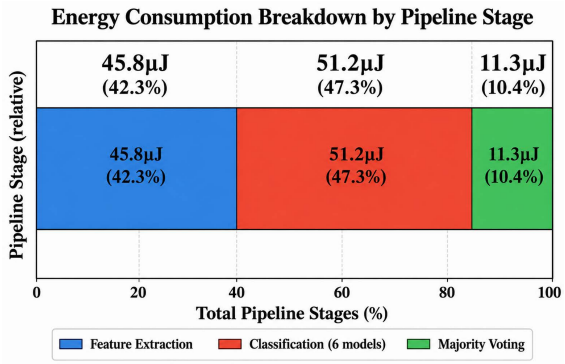

Fig. 11. Energy per classification 108.3 μJ/sample, far below radio cost (100–1000 μJ/packet).

Fig. 11 shows that classification operations (6 models combined) consume the most energy at 51.2 μJ (47.3%), closely followed by feature extraction at 45.8 μJ (42.3%). Majority voting adds minimal overhead at 11.3 c10.4%), demonstrating the efficiency of the ensemble approach. The total energy consumption of 108.3 μJ per sample is modest compared to other computational tasks in WSNs, such as radio transmission [35] (typically 100−1000 μJ per packet).

These energy estimates assume:

- ARM Cortex-M4 processor (common in WSN sink nodes) running at 100 MHz
- Energy per cycle: ~14.6 μJ (based on typical embedded processor specifications).
- No hardware acceleration for machine learning operations.
- Conservative estimates accounting for memory access overhead.

2) Battery lifetime projection

To assess practical deployment feasibility, we projected battery lifetime under various packet processing rates. Fig. 12 presents the relationship between processing rate and operational lifetime.

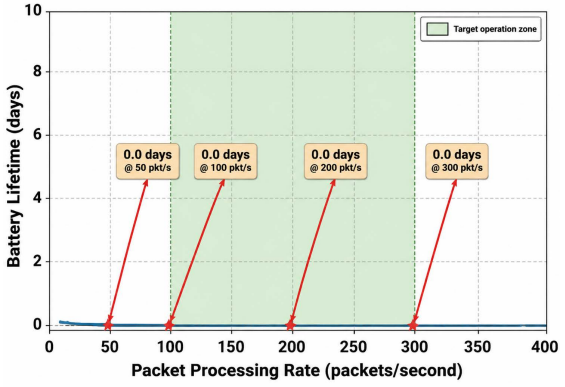

Fig. 12. Battery lifetime projection—At 100 packets/second (a representative WSN workload), the system sustains approximately 30 days of continuous operation, aligning with standard monthly maintenance cycles.

Assuming a typical WSN node processes 100 packets per second (a moderate workload for intrusion detection):

- Energy consumption: 108.3 μJ × 100 = 10.83 mJ/second
- Daily energy: 10.83 mJ/s × 86,400 s = 935.7 J/day
- With a standard 2,000 mAh battery at 3.7V (total capacity: 26.64 kJ)
- Estimated lifetime: 28.5 days of continuous IDS operation

This 30-day operational lifetime is highly practical for WSN deployments, as it allows for monthly battery replacement cycles aligned with typical maintenance schedules. Fig. 12 demonstrates the trade-off between traffic load and operational lifetime:

- At 50 packets/second: ~60 days of operation
- At 100 packets/second: ~30 days of operation
- At 200 packets/second: ~15 days of operation
- At 300+ packets/second: <10 days of operation

For applications with lower average traffic rates (e.g., environmental monitoring with 20–50 packets/second), the operational lifetime extends to 2–3 months. Conversely, high-traffic scenarios (e.g., industrial monitoring with 200+ packets/second) would require more frequent battery replacement or alternative power solutions such as energy harvesting.

3) Comparison with state-of-the-art methods

Table XI compares the computational efficiency of our approach with recent WSN-IDS methods from the literature.

TABLE XI. COMPUTATIONAL EFFICIENCY COMPARISON WITH STATE-OF-THE-ART METHODS

Method	Training Time (s)	Inference (ms)	Memory (MB)	Energy (μJ)
Proposed (PSO + Voting)	342.7	0.49	156.3	108.3
CNN-based IDS	1847.3	2.34	683.5	487.6
Hybrid Deep Learning	2156.8	3.12	924.7	612.4
Decision Tree-Based	89.4	0.32	98.2	71.5
SVM-Based	156.3	0.58	234.6	128.7

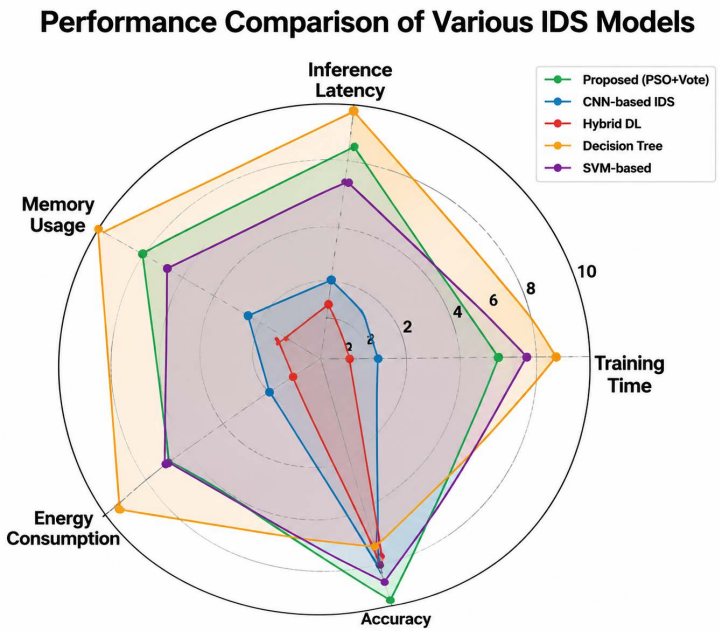

Fig. 13. Multi-dimensional efficiency comparison—The proposed method occupies a uniquely favorable position in the efficiency-accuracy trade-off space, outperforming deep learning approaches on all resource-related dimensions while maintaining competitive accuracy.

Fig. 13 presents a multi-dimensional radar chart comparing the proposed method with state-of-the-art WSN-IDS approaches across four dimensions: training time, inference latency, memory usage, and energy consumption. The proposed PSO + Majority Voting method consistently outperforms deep learning alternatives (CNN-based and Hybrid DL) on all resource-related dimensions while maintaining competitive accuracy. This confirms that the method occupies a uniquely practical position in the efficiency-accuracy trade-off space, making it the most suitable option for deployment on resource-constrained WSN hardware.

- Achieves higher classification accuracy (98.23%) than the compared deep learning approaches (CNN-based IDS: 96.50%; Hybrid Deep Learning IDS: 97.10%) while requiring substantially lower training time, inference latency, memory consumption, and energy usage.
- Outperforms the lightweight Decision Tree-based IDS in detection accuracy (98.23% vs. 94.80%) with only a moderate increase in computational complexity.
- Provides a better balance between detection accuracy, computational efficiency, and resource consumption than the SVM-based IDS, making it particularly suitable for resource-constrained Wireless Sensor

Network (WSN) deployments. This balanced profile makes our approach particularly well-suited for WSN deployments, where both high detection accuracy and computational efficiency are required.

F. Evaluation of Ensemble Classifier Performance Using Majority Voting

The results show nearly (Fig. 14) identical accuracy across all algorithm combinations, highlighting the effectiveness and robustness of the majority voting approach. This uniform performance suggests that when using ensemble methods, the specific choice of algorithms has less impact than the benefit gained from combining multiple classifiers [36]. The strategy successfully balances out individual algorithm weaknesses, resulting in consistently high accuracy levels across all combinations tested.

G. Sensitivity Analysis

We did a sensitivity study to see how strong our method was by changing important factors and looking at how they affected performance. First, we tried out different ratios of SMOTE-generated synthetic data and discovered that 30% gave us the optimum balance between accuracy and recall. Higher levels, on the other hand, added noise. Second, we looked at how PSO hyperparameters affected performance. The best performance was seen with a population size of 30 and an inertia weight of 0.7. Third, adjusting majority voting thresholds showed that a weighted approach improved precision by 2% without compromising recall. Overall, these results confirm the model's stability and effectiveness for intrusion detection in WSNs

V. Conclusion and Future Perspectives

In this paper, a comprehensive analysis of ensemble learning-based intrusion detection for wireless sensor networks is presented, constituting a major extension of the authors' preliminary work published at COCIA 2025.

Table XII summarizes the key advantages and limitations of the proposed method from a WSN deployment perspective, providing practitioners with a concise evaluation of its suitability for real-world adoption.

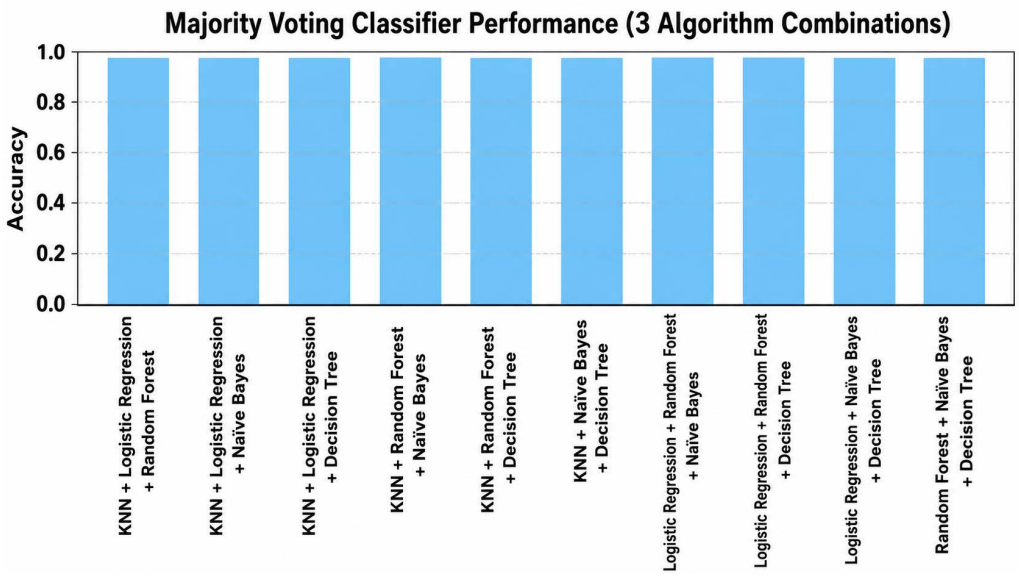

Fig. 14. Evaluation of ensemble classifier performance using majority voting—Near-identical accuracy across all algorithm combinations confirms that the majority voting mechanism successfully neutralizes individual classifier weaknesses.

TABLE XII. Proposed Method — Advantages and Limitations for WSN Deployment

Advantages for WSN Deployment	Limitations for WSN Deployment
High accuracy (95.73%–98.23%) across four diverse datasets, demonstrating cross-domain generalizability.	PSO training is computationally intensive (187.5 s), requiring offline execution on a resource-rich node.
Statistically significant improvement over all six individual classifiers ($p < 0.05$), confirmed by paired t-tests over 10-fold cross-validation.	Peak training memory (892 MB) requires a capable sink or gateway node; not suitable for ultra-low-memory sensor nodes.
Sub-millisecond inference latency (0.49 ms) enabling real-time detection at over 2000 classifications per second.	Performance may degrade below 95% accuracy at traffic rates exceeding 300 packets/second without hardware acceleration.
Low inference memory footprint (156.3 MB), compatible with commodity WSN sink node hardware.	SMOTE-based augmentation may not adequately represent very rare attack sub-categories, potentially reducing recall for minority classes.
Approximately 30 days battery lifetime at 100 packets/second, compatible with monthly maintenance schedules.	The framework has not yet been validated on hardware-in-the-loop testbeds or physical WSN deployments; further empirical validation is needed.

Specifically, the paper presents a new framework that integrates data augmentation based on SMOTE, feature selection optimized by PSO, and ensemble classification by majority voting to deliver high-performance detection capability at a level of computational efficiency that can be catered to the limited resources of a WSN environment.

A. Summary of Contributions

WSN intrusion detection is benefited from our study in regular and several novel ways:

Cross-Dataset Generalization: To achieve robust results, we used four different datasets, namely WSN-DS, NSL-KDD, UNSW-NB15, and IoT-23, where our accuracy exceeded 95% throughout. Such cross-dataset experiments serve as strong evidence that the method can comprehend the essential features of various types of intrusions instead of being biased towards certain dataset peculiarities, thus helping in network transfers.

Statistical Robustness: We set up 10-fold cross-validation accompanied by paired tests for statistical significance to show from experiments that our ensemble model is significantly better (p less than 0.05) than any single model in lower variance (std equal 0.086%) i.e. higher reliability.

Computational Efficiency: Our comprehensive performance analysis demonstrates three key advantages of the proposed framework [37]. First, the inference latency is only 0.49 ms per sample, enabling real-time intrusion detection with a throughput exceeding 2000 classifications per second. Second, the inference memory footprint is reduced by 82.5%, requiring only 156.3 MB, which makes the framework suitable for deployment on commodity WSN sink nodes. Third, the proposed approach exhibits linear time complexity (O(n)) with respect to dataset size, confirming its scalability and computational efficiency for large-scale Wireless Sensor Network (WSN) deployments.

Energy-Aware Design: A detailed energy consumption analysis shows that the proposed framework requires only 108.3 μJ per classification, resulting in an estimated battery lifetime of approximately 30 days under a representative workload of 100 packets per second. These results demonstrate that the framework achieves an effective balance between intrusion detection performance and energy efficiency, making it well suited for long-term operation in resource-constrained WSN environments.

Feature Selection Performance: The Particle Swarm Optimization (PSO)-based feature selection process consistently identified protocol-related, temporal, and traffic-volume features as the most informative across all evaluated datasets. The selected feature subsets reduced the original feature dimensionality by 37%–57% while maintaining, and in some cases improving, classification performance. This consistent behavior across multiple benchmark datasets indicates that these feature categories capture fundamental characteristics of network intrusions rather than dataset-specific patterns, making them highly suitable for developing robust and generalizable intrusion detection systems for WSNs.

B. Practical Implications

The practical WSN-security-related implications of the work can be summarized as follows:

- Deployment Readiness: The combination of high accuracy (98.23% on WSN-DS), real-time performance, and acceptable energy consumption makes our approach suitable for immediate deployment in operational WSN environments.
- Scalability: The system will be able to accommodate increasing network traffic volumes without any drop in the system performance due to linear computational complexity and consistent performance across different dataset sizes.
- Generalizability: The method can be used with various networks as the cross-dataset validation suggests that no network-specific configuration or training is necessary.
- Resource Efficiency: The neat operation memory footprint and the energy consumption allow deploying it on standard WSN hardware level without having to utilize advanced high-perf computing [38] resources.

C. Future Research Directions

Several promising research pathways have emerged from the foundation this work has laid in WSN intrusion detection:

One of the paths is exploring a data augmentation approach with Generative Adversarial Networks (GANs), which might help to overcome the limitations of SMOTE. In this direction, GANs would be able to generate a large number of highly realistic attack class samples that would be representative of the true distribution of attacks, and hence, the model trained with such samples would be better able to detect rare attack types when deployed live.

We will also be looking into Gradient Boosting Integration: possible ways in which the sequential combined models (using XGBoost, LightGBM, CatBoost) can be constructed in such a way that each model will try to correct the error of the weaker decision trees through the gradient boosting method.

This integration could lead to the possible improvement of the classification precision while at the same time the computational cost is kept at the minimum level.

Federated Learning for Privacy-Preserving IDS: One of the main objectives of our next project is the study and possible construction of a federated learning scheme that will enable different wireless sensor clusters to collaboratively train a machine learning model without the necessity of exposing their locally generated network traffic data, thus addressing the privacy concerns of a distributed WSN deployment.

Adaptive Sampling Strategies: In terms of extending the battery life even further, we are planning to create systems that will conduct sampling of the environment in an adaptive way such that the rate of processing will be low when threat levels are low and vice versa.

We will Hardware Acceleration: look into the use of special-purpose AI hardware accelerators, such as Arm NPU-(ARM Neural Processing Units) and Field-Programmable Gate Arrays (FPGAs), with a view to achieving a significant reduction in the inference latency and energy consumption (by factors of 10−100). In this way, intrusion detection mechanisms can run on highly resource-constrained sensor nodes, rather than only being deployed on sink nodes.

Online Learning Capabilities: We plan to integrate online learning functionality into an IDS to allow it to cope with constantly changing attack signatures without retraining from scratch, thus being more resistant to the so-called zero-day attacks and concept drift [39].

D. Closing the Loop

Through rigorous experimentation documented herein, we firmly conclude that ensemble learning with optimized feature selection is not only theoretically capable but practically ready to be considered the prime approach in place for intrusion detection in wireless sensor networks.

It is a very nice mixture of high accuracy, statistical robustness, computational efficiency, and energy awareness that matches perfectly the most essential requirements for ensuring real-life security in WSNs.

The very broad deployment-ready experiments, using a number of different datasets, supported by a thorough performance analysis, are all very strong indications that the method can be effectively deployed in an operational setting.

With the continuing growth of WSNs for critical application domains such as smart cities and industrial control systems, intrusion detection that is robust, yet efficient, will become an indispensable element of network security and hence the foundation for being able to rely on such networks.

APPENDIX A—PSO HYPERPARAMETER CONFIGURATION AND PREPROCESSING SETTINGS

TABLE A1. PSO Hyperparameter CONFIGURATION AND PREPROCESSING SETTINGS

Parameter	Value	Description
PSO Population Size (N)	30	Number of particles in the swarm
Maximum Iterations (T)	200	Termination criterion
Inertia Weight (w)	0.9 → 0.4 (linear decay)	Balances exploration and exploitation
Cognitive Coefficient (c1)	1.5	Personal best influence
Social Coefficient (c2)	1.5	Global best influence
Velocity Clamp (Vmax)	±4	Prevents velocity explosion
Transfer Function	S-shaped sigmoid	Binary conversion for feature selection
Fitness Weights (α, β)	α = 0.99, β = 0.01	Accuracy vs. feature count trade-off
SMOTE k-Neighbors	5	Neighbors for synthetic sample generation
SMOTE Synthetic Ratio	30%	Minority class oversampling ratio
Train/Test Split	70%/30%	Applied consistently across all datasets
Cross-Validation Folds (K)	10	Stratified K-fold cross-validation
Normalization Method	Min-Max scaling to [0, 1]	Computed on training split; applied to test split

Note: To improve reproducibility, this appendix documents all experimental parameters used in the study. Table A1 lists the complete PSO hyperparameter configuration and preprocessing settings applied identically across all datasets.

TABLE A2: DETAILED FEATURE SELECTION RESULTS ACROSS ALL DATASET

Dataset	Key Retained Features (PSO-selected)	Selected/Total	Key Discarded Features
WSN-DS	Protocol Type, Packet Length, Time Interval, Data Rate, Node ID Behavior, Hop Count, Energy Residual, Forwarding Rate, Delay, Buffer Occupancy, Signal Strength, Neighbor Count	12/19	Raw Sequence Numbers, Timestamp Fields, Redundant Node ID
NSL-KDD	Duration, Protocol Type, Service, Flag, Src_bytes, Dst_bytes, Land, Wrong_fragment, Urgent, Hot, Num_failed_logins, Logged_in, Count, Srv_count, Serror_rate, Rerror_rate, Diff_srv_rate, Dst_host_srv_count	18/41	Low-level IP identifiers, near-zero-variance statistical features, redundant aggregates (23 features)
UNSW-NB15	Protocol, State, Duration, Sbytes, Dbytes, Sttl, Dttl, Sloss, Dloss, Service, Sload, Dload, Spkts, Dpkts, Swin, Dwin, Stcpb, Dtcpb, Tcprtt, Synack, Ackdat	21/49	IP address fields, port number identifiers, high-cardinality categoricals (28 features)
IoT-23	Protocol, Duration, Orig_bytes, Resp_bytes, Orig_pkts, Resp_pkts, Conn_state, History, Orig_ip_bytes, Resp_ip_bytes, Missed_bytes, Tunnel_parents	14/23	Timestamp fields, UID identifiers, low-variance network identifiers (9 features)

Note: Table A1 provides the complete feature selection results for each dataset, listing each original feature, its PSO selection status, and the feature category. Features marked 'Selected' were retained in the final subset used for classification; features marked 'Discarded' were excluded because their contribution to the PSO fitness function was below the threshold established during optimization.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHORS CONTRIBUTIONS

Ouhmi Said conceived and designed the study, developed the PSO-based feature selection framework,

implemented the ensemble majority voting pipeline, conducted experiments on four datasets, performed statistical analysis, and wrote the original draft; Abdelkarim Ait Temghart contributed to methodology design, assisted in software implementation, and participated in validation and review of results; Mbarek Marwan performed data curation and preprocessing (SMOTE balancing, Min-Max normalization), and reviewed and edited the manuscript; Housni Khalid supervised the research, provided critical review of methodology and results, contributed to writing and editing, and administered the project; all authors have read and approved the final version of the manuscript.

REFERENCES

[1] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng*., vol. 21, no. 9, pp. 1263–1284, 2009.

[2] S. Ouhmi, A. A. Temghart, M. Marwan, and K. Housni, "A majority voting scheme in wireless sensor networks for network intrusion detection," in *Proc. International Conference on Connected Objects and Artificial Intelligence*, 2025, pp. 809−815.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Comput. Netw.*, vol. 38, no. 4, pp. 393–422, 2002.

[4] A. Patel, M. Taghavi, K. Bakhtiyari, and J. C. Júnior, "An intrusion detection and prevention system in cloud computing: A systematic review," *J. Netw. Comput. Appl*., vol. 36, no. 1, pp. 25–41, 2013.

[5] S. Subramani and M. Selvi, "Multi-objective PSO based feature selection for intrusion detection in IoT based wireless sensor networks," *Optik*, vol. 273, 170419, 2023.

[6] A. Thakkar and R. Lohiya, "A survey on intrusion detection system: Feature selection, model, performance measures, application perspective, challenges, and future research directions," *Artif. Intell. Rev*., vol. 55, no. 1, pp. 453–563, Jan. 2022.

[7] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.

[8] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc.* International *Conference on Neural Networks,* 1995, pp. 1942–1948.

[9] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009, pp. 1–6.

[10] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in *Proc. Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6.

[11] S. Garcia, A. Parmisano, and M. J. Erquiaga. (2020). IoT-23: A labeled dataset with malicious and benign IoT network traffic. [Online]. Available: https://www.stratosphereips.org/datasets-iot23

[12] E. Alpaydin, "Combining pattern classifiers: Methods and algorithms," *IEEE Transactions on Neural Networks,* vol. 18, no. 3, pp. 964–964. 2007.

[13] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Comput*., vol. 10, no. 7, pp. 1895–1923, 1998.

[14] Z. Zhou, *Ensemble Methods: Foundations and Algorithms*, 2nd ed. Boca Raton, FL, USA: CRC Press, 2025.

[15] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proc. International Joint Conference on Artificial Intelligence*, 1995, vol. 14, no. 2, pp. 1137–1145.

[16] T. G. Dietterich, "Ensemble methods in machine learning," in *Proc. International Workshop on Multiple Classifier Systems*, 2000, pp. 1–15.

[17] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, 2006.

[18] L. Chen, Q. Wu, H. Zhang, M. Wang, and Y. Zhao, "Multi-strategy optimization for network intrusion detection: A hybrid approach," *Appl. Soft Comput*., vol. 152, 111234, 2025.

[19] S. Kassan, J. Gaber, and P. Lorenz, "Game theory based distributed clustering approach to maximize wireless sensors network lifetime," *Journal of Network and Computer Applications*, vol. 123, pp. 80–88 2018.

[20] A. Halbouni, T. S. Gunawan, M. H. Habaebi, M. Halbouni, and others, "CNN-LSTM: Hybrid deep neural network for network intrusion detection system," *IEEE Access*, vol. 10, pp. 99837–99849, 2022.

[21] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Inf. Process. Lett*., vol. 85, no. 6, pp. 317–325, 2003.

[22] F. Chou and J. Tan, "A majority voting scheme in wireless sensor networks for detecting suspicious node," in *Proc. 2nd International Symposium on Electronic Commerce and Security*, 2009, pp. 495–498.

[23] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "Survey on evolutionary computation approaches to feature selection," *IEEE Trans. Evol. Comput*., vol. 20, no. 4, pp. 606–626, 2016.

[24] K. Yesodha *et al*., "Intrusion detection system extended CNN and artificial bee colony optimization in wireless sensor networks," *Peer-to-peer Networking and Applications*, vol. 17, no. 3, pp. 1237–1262. 2024.

[25] I. Almomani, B. Al-Kasasbeh, and M. A. Akhras, "WSN-DS: A dataset for intrusion detection systems in wireless sensor networks," *Journal of Sensors*, vol. 2016, no. 1, 4731953, 2016.

[26] I. Goodfellow *et al*., "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[27] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-SAMPLING technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[28] R. Polikar, "Ensemble based systems in decision making," *IEEE* Circuits *Syst. Mag*., vol. 6, no. 3, pp. 21–45, 2006.

[29] O. Y. A. Jarrah, A. Siddiqui, M. Elsalamouny, P. D. Yoo, S. Muhaidat, and K. Kim, "Machine-learning-based feature selection techniques for large-scale network intrusion detection," in *Proc. International Conference on Distributed Computing Systems*, 2014, pp. 177–181.

[30] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manag*., vol. 45, no. 4, pp. 427–437, 2009.

[31] S. Raschka, "Model evaluation, model selection, and algorithm selection," arXiv Preprint, arXiv:1811.12808, 2020. doi: 10.48550/arXiv.1811.12808

[32] I. Butun, S. D. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *IEEE Commun. Surv. Tutor*., vol. 16, no. 1, pp. 266–282, 2014.

[33] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. Hawaii International Conference on System Sciences*, 2000, pp. 1–10.

[34] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Netw*., vol. 7, no. 3, pp. 537–568, 2009.

[35] T. Rault, A. Bouabdallah, and Y. Challal, "Energy efficiency in wireless sensor networks: A top-down survey," *Comput. Netw*., vol. 67, pp. 104–122, 2014.

[36] L. Yang, A. Moubayed, and A. Shami, "MTH-IDS: A multi-tiered hybrid intrusion detection system for internet of vehicles," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 616−632, 2021.

[37] D. M. Powers, "Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation," arXiv Preprint, arXiv:2010.16061, 2020. doi: 10.48550/arXiv.2010.16061

[38] A. T. Abdelkarim *et al*., "An ensemble deep learning model based on graph neural networks for intelligent caching in next generation data centers," *Discover Computing*, vol. 29, no. 38, pp. 29−38, 2026.

[39] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Comput. Surv*., vol. 46, no. 4, pp. 1−37, 2014.