Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.081953

ARTICLE

A Novel Adaptive Deep Learning-Based Intrusion Detection System Using Particle Swarm Optimization

Soukaina Mjahed[1] **and Ouail Mjahed**[2,*]

[1]Department of Computer Sciences, Faculty of Sciences Semlalia, Cadi Ayyad University, Marrakech, Morocco
[2]Department of Computer Sciences, Faculty of Sciences and Technology, Cadi Ayyad University, Marrakech, Morocco

*Corresponding Author: Ouail Mjahed. Email: ouail.mjahed@ced.uca.ma

Received: 11 March 2026; Accepted: 08 April 2026; Published: 08 May 2026

ABSTRACT: The rapid emergence of sophisticated, dynamic, and rare or previously unseen attack pattern exposes fundamental limitations of conventional intrusion detection systems (IDS) based on static learning architectures. While deep learning (DL) models have demonstrated strong performance by capturing complex spatial and temporal traffic patterns, existing DL-based IDS largely rely on fixed decision structures, restricting adaptability to evolving threats. Furthermore, current hybrid DL-metaheuristic approaches typically use such metaheuristics as offline or auxiliary optimizers, without interacting with the deep model's internal latent representations. This paper introduces a novel co-evolutionary IDS that establishes a tight, bidirectional coupling between DL and Particle Swarm Optimization (PSO) through latent-space-guided structural adaptation. A CNN-LSTM (Convolutional Neural Networks-Long Short-Term Memory) encoder learns discriminative spatial–temporal representations of network traffic, which dynamically guide PSO to select and optimize Adaptive Decision Blocks during training. Unlike prior hybrid methods, the proposed framework enables continuous co-evolution of both representation learning and decision structure, allowing the IDS to adapt its internal architecture in response to uncertain, rare, and previously unseen attack patterns. Comprehensive evaluations on UNSW-NB15, CICIDS2017, and ToN-IoT demonstrate statistically significant improvements over state-of-the-art DL and hybrid IDS approaches, achieving over *99.97% accuracy, recall* and F_1-score, and low-latency inference suitable for near real-time deployment.

KEYWORDS: Classification; CNN; PSO; intrusion detection system; adaptive deep learning

1 Introduction

Modern network environments are increasingly exposed to sophisticated and evolving cyber threats, including distributed denial-of-service attacks, advanced persistent threats, and rare or previously unseen attack patterns. Traditional Intrusion Detection Systems (IDS) based on static rules or conventional machine learning methods often struggle to capture the complex spatial and temporal dependencies present in network traffic. As a result, these systems frequently suffer from limited detection capability for emerging attacks and elevated false positive rates.

Recent advances in deep learning (DL) have significantly improved intrusion detection performance. Architectures such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory (LSTM) networks have demonstrated strong capability in learning hierarchical representations from network traffic data [1,2]. In particular, hybrid CNN-LSTM architectures effectively

capture both spatial feature correlations and temporal dependencies across traffic flows, achieving promising performance on benchmark datasets such as UNSW-NB15, CICIDS2017, and ToN-IoT.

Despite these advances, most DL-based IDS rely on fixed architectures and static decision pathways. Consequently, their ability to adapt to evolving attack behaviors or rare intrusion patterns remains limited. In addition, optimization procedures such as feature selection or hyperparameter tuning are commonly performed offline, without interaction with the internal representation learning process of the deep model.

To address these limitations, recent research has explored hybrid frameworks combining deep learning with metaheuristic optimization techniques such as Particle Swarm Optimization (PSO) and Genetic Algorithms (GA) [3,4]. These methods are typically used to improve feature selection, hyperparameter tuning, or architecture configuration. Although such hybrid approaches can enhance detection performance, the metaheuristic component is generally applied in a static or preprocessing manner, preventing dynamic interaction between the optimizer and the deep learning model during training.

This limitation highlights a key research gap: most existing DL–metaheuristic IDS frameworks do not allow the optimization process to adapt the internal decision structure of the deep model during learning. Consequently, the optimizer remains disconnected from the evolving latent representations produced by the network.

Motivated by this limitation, this paper proposes a DL-PSO Co-Evolution (DL-PSO-CE) framework that tightly integrates a CNN–LSTM deep encoder with PSO-driven Adaptive Decision Blocks (ADBs). Unlike existing hybrid approaches, the proposed framework establishes a bidirectional co-evolutionary loop, in which latent representations learned by the deep model guide PSO-based structural adaptation during training. This mechanism enables continuous refinement of the IDS decision structure in response to ambiguous or evolving traffic patterns.

The main contributions of this work are summarized as follows:

(i) Adaptive Decision Blocks (ADBs) are introduced to enable modular and flexible IDS decision pathways.
(ii) A latent-space-guided PSO mechanism dynamically selects and adapts internal decision structures during training.
(iii) A co-evolutionary learning framework is proposed to jointly optimize deep representations and decision topology.
(iv) Extensive experiments on UNSW-NB15, CICIDS2017, and ToN-IoT datasets demonstrate improved detection accuracy, reduced false positive rates, and low inference latency.
(v) A comprehensive statistical significance analysis validates the robustness and practical relevance of the proposed approach.

The remainder of this paper is structured as follows. Section 2 reviews related work, while Section 3 details the proposed algorithms. Section 4 describes the experimental design and implementation parameters. Section 5 presents the main results, including performance comparisons between benchmark datasets, statistical significance analysis, and a comparison with state-of-the-art methods. Finally, Section 6 concludes this paper.

2 Related Works

Deep learning has become a dominant paradigm for intrusion detection due to its ability to model complex and nonlinear patterns in network traffic. CNN-based IDS models capture spatial correlations among traffic features, while recurrent architectures such as RNNs and LSTMs effectively model temporal dependencies across sequential traffic flows [1,2]. Hybrid CNN–LSTM architectures further enhance

detection performance by jointly exploiting spatial and temporal information, particularly in large-scale and IoT-based network environments [1].

To further improve IDS performance, metaheuristic optimization algorithms inspired by natural and evolutionary processes have been increasingly integrated into deep learning pipelines. These methods are commonly employed for feature selection, hyperparameter optimization, and dimensionality reduction. In particular, Particle Swarm Optimization (PSO) has been widely used to optimize feature subsets and deep learning parameters, leading to improved detection accuracy and reduced false alarm rates [5,6]. Similarly, Genetic Algorithms (GA) have been applied to optimize network structures, select relevant features, and tune deep learning hyperparameters [4,7].

Other metaheuristic approaches, including Grey Wolf Optimizer (GWO), Differential Evolution, Dragonfly optimization, Firefly algorithms, and Cybersecurity Optimizer, have also demonstrated effectiveness in IDS optimization tasks [8–12]. Recent studies have further explored the use of swarm-based optimization for cyber-attack detection in complex infrastructures. For example, PSO-based attention temporal graph convolutional networks have been proposed for detecting dummy data injection attacks in power systems [13].

Beyond classical metaheuristics, alternative optimization strategies such as evolutionary strategies, Bayesian optimization, and differentiable architecture search (DARTS) have been investigated for neural architecture optimization [14]. However, these approaches typically focus on global architecture search and often require substantial computational resources.

Recent IDS studies report high detection performance on benchmark datasets such as CICIDS2017, UNSW-NB15, and ToN-IoT through the combination of deep learning and optimization strategies. On CICIDS2017, recurrent and hybrid deep models achieve high detection performance, with LSTM-based and DNN–LSTM architectures reporting accuracies above 99% [15,16]. Metaheuristic-assisted CNN models further improve results, with GA- and PSO-optimized CNN architectures reaching accuracies above 99.7% and F_1-scores close to 99% [17].

For the UNSW-NB15 dataset, CNN–LSTM architectures report moderate detection performance around 92.9% accuracy [1], whereas evolutionary optimization significantly improves results. GA- and PSO-optimized CNN models achieve accuracies above 99% and detection rates near 98% [17], while GWO-CNN approaches report balanced accuracy and F_1-scores exceeding 97% [3]. Similar trends are observed on the ToN-IoT dataset, where deep learning models such as LSTM achieve accuracies close to 99% [18], and classical learning methods including XGBoost and ANN report accuracies between 98.3% and 99.4% [19,20].

Despite these advances, most hybrid DL–metaheuristic IDS frameworks apply optimization in a static or peripheral manner, either before training (e.g., feature selection or hyperparameter tuning) or as a one-time architectural optimization step. Consequently, the internal structure of the deep model remains fixed during training, and no iterative feedback exists between the learned latent representations and the optimization process. This limitation restricts the ability of IDS models to adapt to rare, evolving, or previously unseen attack patterns.

Furthermore, even approaches based on metaheuristic architecture search do not implement a *true co-evolutionary learning process*, as the optimizer and the deep model typically operate in separate stages rather than interacting continuously during training.

Unlike neural architecture search approaches such as NAS or DARTS [14], which explore global network architectures through computationally intensive search procedures, the proposed DL-PSO-CE framework performs *lightweight structural adaptation during training*. PSO dynamically controls the activation of

Adaptive Decision Blocks within the network, enabling continuous interaction between representation learning and structural optimization.

By integrating latent-space-guided optimization into the learning process, the proposed DL-PSO-CE framework enables *dynamic, data-driven structural adaptation*, fundamentally distinguishing it from existing static or offline hybrid IDS approaches.

3 Methodology

This section presents the proposed methodology, combining deep learning with metaheuristic optimization for intrusion detection. After briefly introducing the core components (CNN, LSTM, and PSO), five DL-based algorithmic variants are defined to systematically analyze the contribution of PSO.

The variants range from a pure DL baseline to a fully co-evolutionary DL-PSO framework, with each configuration isolating a specific PSO role, including feature selection, hyper-parameter tuning, structural optimization, and co-evolution.

For clarity, the evaluated algorithms are denoted as follows:

- DL-B (baseline deep learning),
- DL-PSO-FS (PSO-based feature selection),
- DL-PSO-HP (PSO-based hyper-parameter optimization),
- DL-PSO-S (static structural optimization), and
- DL-PSO-CE (proposed co-evolutionary framework).

3.1 Convolutional Neural Networks (CNN)

CNNs are employed as the spatial feature extraction component of the proposed framework. CNN layers capture local correlations among network traffic attributes, such as protocol interactions, flow statistics, and packet-level patterns. Given an input feature vector x, a convolutional layer computes a feature map h_k as follows [21]:

$$h_k = \sigma\left(W_k * x + b_k\right) \tag{1}$$

where $*$ denotes convolution, W_k and b_k represent the weight and bias of the k-th convolutional filter, respectively, and $\sigma(\cdot)$ is a nonlinear activation function. Stacking multiple convolutional layers enables the model to learn increasingly abstract and discriminative representations of complex attack behaviors.

3.2 Long Short-Term Memory Networks (LSTM)

LSTM networks are adopted to model temporal dependencies in network traffic sequences. In intrusion detection, such temporal modeling is crucial for identifying time-dependent and multi-stage attacks.

An LSTM cell updates its internal memory state using gated mechanisms, expressed as in (2) [22]:

$$f_t = \sigma\left(W_f x_t + U_f\, h_{t-1} + b_f\right), \quad i_t = \sigma\left(W_i x_t + U_i\, h_{t-1} + b_i\right), \quad o_t = \sigma\left(W_o x_t + U_o\, h_{t-1} + b_o\right), \tag{2}$$

where f_t, i_t, o_t denote the forget, input, and output gates at time t, respectively. W_f, W_i, W_o denote the input weight matrices associated with the forget, input, and output gates, respectively, U_f, U_i, U_o are the corresponding recurrent weight matrices, and b_f, b_i, b_o are bias terms. This architecture allows effective learning of long-term dependencies while mitigating vanishing gradient issues.

3.3 Particle Swarm Optimization (PSO)

PSO is a population-based metaheuristic inspired by collective swarm intelligence [23]. It searches for optimal solutions by iteratively updating a set of particles according to both individual experience and global knowledge. At each iteration t, the velocity v_i and position x_i, of a particle i, are updated according to (3):

$$\begin{cases} v_i^{t+1} = \omega\, v_i^t + c_1 r_1 \left(p_i^{best} - x_i^t\right) + c_2 r_2 \left(g^{best} - x_i^t\right) \\ x_i^{t+1} = x_i^t + v_i^{t+1} \end{cases} \tag{3}$$

where ω is the inertia weight, c_1 and c_2 are acceleration coefficients, and $r_1, r_2 \in [0, 1]$ are random variables, p_i^{best} is the personal best, and g^{best} is the global best position.

Based on the optimization principles of PSO, four deep learning-based intrusion detection algorithms are introduced in the following subsections, in which PSO is progressively integrated, ranging from static offline optimization schemes to a fully co-evolutionary learning framework.

3.4 DL-B: Pure DL Baseline

The DL-Base model is a hybrid CNN–LSTM, trained conventionally. CNN layers capture spatial correlations among traffic features. LSTM layers model temporal dependencies across flows. Training uses back-propagation on a fixed architecture. This baseline serves as the reference for evaluating the impact of PSO-based enhancements. Algorithm 1 describes the steps involved in the DL-B model.

Algorithm 1: DL-B

Input: Dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$, Epochs T, batch size
Output: Trained deep model parameters θ^*
Initialize CNN, LSTM and fully connected classifier (FC) parameters (θ_{CNN}, θ_{LSTM}, θ_{FC}), Set Adam optimizer
for epoch = 1 to T do
  Shuffle D and Split into mini-batches B
  for each mini-batch B **do**
    Extract spatial features: $F_{CNN} = CNN(B, \theta_{CNN})$
    Extract temporal features: $F_{LSTM} = LSTM(F_{CNN}, \theta_{LSTM})$
    Compute class probabilities: $\hat{y} = Softmax(FC(F_{LSTM}, \theta_{FC}))$
    Compute loss L using cross-entropy $L(y, \hat{y})$
    Update θ_{CNN}, θ_{LSTM}, θ_{FC}
  end for
end for
Return θ^*, and performance metrics on training, validation, and test sets

3.5 DL- PSO-FS: Static Feature Selection

In the first proposed Algorithm, PSO is applied offline to select a subset of input features. Each particle encodes a binary feature mask. Fitness is measured via classification performance on a lightweight evaluator. The CNN–LSTM is then trained on the reduced feature set, with no further PSO interaction, reflecting standard preprocessing-based PSO usage in IDS literature.

Algorithm 2 details the DL-PSO-FS steps.

Algorithm 2: DL-PSO-FS

Input: Dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$, Epochs T, batch size, PSO Parameters
Output: Trained deep model parameters θ^*
1: Initialize PSO particles encoding feature masks
2: **for** PSO iteration = 1 to I_{pso} **do**
3: Evaluate particle fitness using lightweight classifier
4: Update particle velocities and positions
5: **end for**
6: Select optimal feature subset D_{FS}
7: Initialize $(\theta_{CNN}, \theta_{LSTM}, \theta_{FC})$
8: Train DL model on D_{FS} following Algorithm 1 (steps 2–11)
9: Return θ^*, and performance metrics on training, validation, and test sets.

3.6 DL-PSO-HP: Hyper-Parameter Optimization

In this Algorithm, PSO optimizes hyper-parameters (learning rate, number of CNN filters, LSTM units) offline. Selected hyper-parameters are fixed during training, improves convergence and stability but does not interact dynamically with latent representations. The steps of the DL-PSO-HP method are summarized in Algorithm 3.

Algorithm 3: DL-PSO-HP

Input: Dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$, Epochs T, batch size, PSO Parameters
Output: Trained parameters θ^*
1: Initialize PSO particles encoding hyper-parameters
2: **for** PSO iteration = 1 to I_{pso} **do**
3: Train DL model with candidate hyper-parameters (few epochs)
4: Evaluate validation fitness
5: Update particle velocities and positions
6: **end for**
7: Select optimal hyper-parameters
8: Train DL model on D using Algorithm 1 (steps 2–11)
9: Return trained model θ^*, and performance metrics on training, validation, and test sets.

3.7 DL-PSO-S (Static Optimization)

In DL-PSO-S Algorithm, PSO is used to preselect the structure of ADBs. Each particle represents a candidate decision block configuration. The structure is fixed after optimization, providing greater expressiveness than DL-Base but lacking dynamic adaptation to evolving attacks. The different phases of the DL-PSO-S are shown in Algorithm 4.

Algorithm 4: DL-PSO-S

Input: Dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$, Epochs T, batch size, number of ADBs, PSO Parameters
Output: Trained parameters θ^*
1: Initialize PSO particles encoding ADB activation vectors
2: **for** PSO iteration = 1 to I_{pso} **do**

(Continued)

Algorithm 4 (continued)

3: Evaluate fitness of candidate ADB structures
4: Update particle velocities and positions
5: **end for**
6: Fix optimal ADB configuration
7: Initialize CNN–LSTM and selected ADBs
8: Train DL model using Algorithm 1 (steps 2–11)
9: Return θ^*, and performance metrics on training, validation, and test sets.

3.8 Proposed DL–PSO–CE: Co-Evolutionary Framework

The proposed DL–PSO–CE framework establishes a bidirectional co-evolutionary interaction between deep representation learning and swarm-based structural optimization. A CNN–LSTM encoder continuously learns latent representations of network traffic, while Particle Swarm Optimization (PSO) dynamically adapts the model structure by controlling the activation of Adaptive Decision Blocks (ADBs).

Each ADB consists of a lightweight fully connected module followed by a nonlinear activation function. Multiple ADBs operate in parallel, forming modular decision pathways whose outputs are aggregated before the final classification layer. Their activation is governed by a binary vector encoded by PSO particles, enabling dynamic and compact adaptation of the decision structure during training.

Unlike conventional hybrid DL–PSO approaches with offline optimization, PSO is integrated directly into the training loop. Latent representations produced by the CNN–LSTM are used to estimate latent-space uncertainty, which is incorporated into the PSO fitness evaluation. PSO then updates ADB activation patterns, modifying the model structure before subsequent gradient-based updates.

Each particle encodes a binary activation vector, where each dimension corresponds to one ADB. This iterative interaction forms a closed feedback loop between representation learning and structural adaptation, enabling progressive refinement of decision boundaries and improved detection of rare or unseen attacks.

To guide the structural optimization process, an uncertainty-aware fitness signal is incorporated into the PSO evaluation. The uncertainty component is derived from the predictive entropy of the class posterior probabilities produced by the CNN–LSTM encoder. This entropy value quantifies classification ambiguity and is combined with performance metrics to guide the selection of ADB configurations that simultaneously improve classification accuracy and reduce uncertainty.

Algorithm 5 summarizes the overall co-evolutionary training procedure.

Algorithm 5: DL–PSO-CE

Input: Dataset D, Epochs T, number of ADBs, PSO Parameters
Output: Trained deep model parameters θ^*
1: Initialize CNN–LSTM parameters and PSO particles encoding ADB activation vectors
2: **for** epoch = 1 to T **do**
3: Shuffle D and split into mini-batches B
4: **for** each B **do**
5: Extract latent features F_{CNN}, F_{LSTM}
6: Compute latent-space uncertainty from current representations
7: Evaluate PSO fitness using performance and uncertainty signals
8: Update PSO particles and ADB activation vectors

(Continued)

Algorithm 5 (continued)

```
            Forward propagate through the selected ADBs
            Compute classification loss and update CNN–LSTM parameters
        end for
    end for
    Return θ*, and performance metrics on training, validation, and test sets.
```

To ensure methodological consistency across PSO-based variants, a unified fitness formulation is adopted:

$$f = a_1 Acc + a_2 F_1 - a_3 U \tag{4}$$

where Acc, F_1 and U denote Accuracy, F_1-score and latent-space uncertainty, respectively, and a_1, a_2 and a_3 are weighting coefficients. The performance components of the fitness function (Acc and F_1-score) are defined in Section 4.3. The latent-space uncertainty U is estimated using the mean predictive entropy of the class posterior distributions derived from the CNN–LSTM latent representations:

$$U = \frac{1}{N} \sum_{i=1}^{N} \left(- \sum_{c=1}^{C} p_{i,c} \log p_{i,c} \right) \tag{5}$$

where $p_{i,c}$ denotes the predicted probability of class c for sample i, C is the number of classes, and N is the number of samples.

Predictive entropy is employed as an uncertainty proxy due to its computational efficiency and stability during training. Entropy-based uncertainty estimation has been widely adopted in deep learning models to identify ambiguous or poorly classified samples. Although alternative estimators such as Monte Carlo dropout or ensemble-based uncertainty could be considered, they introduce additional computational overhead that may limit their integration within iterative optimization loops.

4 Experimental Design

This section describes the datasets used, the experimental setup, and evaluation metrics.

4.1 Datasets Description

To comprehensively evaluate the proposed intrusion detection framework under diverse and realistic network conditions, experiments were conducted on three publicly available and widely adopted benchmark datasets: UNSW-NB15, CICIDS2017, and ToN-IoT. These datasets differ in traffic characteristics, attack diversity, and temporal complexity, enabling a robust assessment of model generalization.

UNSW-NB15: 2.5M instances, 49 features, nine modern attack categories [24].

CICIDS2017: 2.8M instances, 80 flow-based features covering 15 attacks classes [25].

ToN-IoT: IoT and industrial telemetry, including up to 10 classes, described by about 43 attributes [26].

All datasets were subjected to a unified preprocessing pipeline, including the removal of missing and infinite values, one-hot encoding of categorical features, elimination of duplicate and incomplete flows, and min–max normalization of continuous attributes.

To ensure label consistency, related attack subtypes were grouped into unified classes, resulting in 7 classes for each Dataset. Class imbalance, particularly severe in CICIDS-2017 and ToN-IoT, was mitigated using SMOTE [27]. Since uncertainty signals are derived from model predictions rather than directly from

input samples, the influence of SMOTE-generated synthetic instances on uncertainty estimation remains limited. Consequently, the use of SMOTE does not introduce significant bias into the co-evolutionary optimization process.

Approximately 230k samples were retained from each Dataset, then split into training (70%), validation (15%), and test (15%) sets using stratified sampling. The normalized class distributions of the resulting datasets are reported in Table 1.

Table 1: Characteristics of the used datasets.

Class	Label			Number of Samples		
	UNSW-NB15	CICIDS2017	ToN-IoT	Train	Validation	Test
C_1	Normal	Normal	Normal	70,000	15,000	15,000
C_2	DoS	DoS/DDoS	DoS/DDoS	35,000	7510	7490
C_3	Exploit	PortScan	Scanning	21,000	4505	4495
C_4	Reconnaissance	Web	XSS	14,000	2985	3015
C_5	Fuzzers	Botnet	Injection	10,500	2245	2255
C_6	Generic	Infiltration	Password	7000	1509	1491
C_7	Analysis/Others	Heartbleed	Backdoor/Others	3500	754	746
All	230,000	230,000	230,000	161,000	34,508	34,492

To address the high dimensionality of the datasets, statistical correlation analysis and SHAP-based feature attribution were jointly employed. Highly correlated features ($r > 0.9$) were removed, and post-training SHAP analysis [28] was used to rank feature importance, retaining the top 24 features to balance detection performance and interpretability.

4.2 Experimental Setup

All experiments were conducted on an NVIDIA RTX A6000 GPU (48 GB VRAM) using PyTorch 2.2. Hyper-parameters were determined via ten-fold cross-validation.

All experiments are conducted under multi-class intrusion detection settings, consistent with the class distributions of the CICIDS2017, UNSW-NB15, and ToN-IoT datasets, provided in Table 1.

The proposed IDS employs a hybrid CNN–LSTM architecture composed of two one-dimensional convolutional layers with 64 and 128 filters (kernel size = 3), followed by max-pooling (pool size = 2) and ReLU activation. Temporal dependencies are modeled using two stacked LSTM layers with 128 hidden units and a dropout rate of 0.3. The classification layer is fully connected with Softmax activation.

Model training is performed using the Adam optimizer with a learning rate of 0.001, a batch size of 128, and categorical *cross-entropy loss*.

For PSO-based variants, a swarm size of 40 particles is used over 50 iterations, with inertia weight $\omega = 0.7$ and cognitive and social coefficients set to $c_1 = c_2 = 1.5$.

The fitness weights in Eq. (4) were fixed to $a_1 = a_2 = 0.5$ for all variants. The latent-space uncertainty term was deactivated ($a_3 = 0$) for all static PSO-based algorithms and enabled only in DL-PSO-CE ($a_1 = 0.5$, $a_2 = 0.4$, $a_3 = 0.1$).

4.3 Evaluation Metrics

To evaluate the proposed methods, precision rate Acc_i, Recall R_i and F_{1i}-score are computed, for each class C_i, as well as global rates such as accuracy Acc, recall R and F_1-score.

$$Acc_i = \frac{C_{ii}}{\sum_j C_{ji}}, \quad R_i = \frac{C_{ii}}{\sum_j C_{ij}}, \quad F_{1,i} = \frac{2Acc_i R_i}{Acc_i + R_i} \tag{6}$$

$$Acc = \frac{\sum_i C_{ii}}{\sum_{i,j} C_{ij}}, \quad R = \frac{1}{K}\sum_{i=1}^{K} R_i, \quad F_1 = \frac{1}{K}\sum_{i=1}^{K} F_{1,i} \tag{7}$$

where C_{ij} denotes the number of samples belonging to class C_i predicted as class C_j and K the number of classes. In addition, AUC (Area Under roc Curves), Inference latency (ms/packet), and training time per epoch are considered. Paired t-test and *Wilcoxon signed-rank test* [29] are used to assess significance of observed improvements. Note that the paired *t-statistic* and the Wilcoxon Z *statistic* are computed as follows:

$$t = \frac{\overline{d}}{s_d/\sqrt{n}}, \quad Z = \frac{W - \mu_W}{\pi_W}, \tag{8}$$

where $\overline{d}$, s_d, and n are the mean, standard deviation, and number of paired differences, respectively; W, μ_W and π_W are the sum of signed ranks, its expected mean, and standard deviation under the null hypothesis.

5 Results and Analysis

This section presents the experimental results of all algorithmic variants, analyzing classification performance, statistical significance, and comparative evaluation against state-of-the-art IDS approaches.

5.1 Classification Performance

The five algorithmic variants were evaluated on UNSW-NB15, CICIDS2017, and ToN-IoT using Accuracy (Acc), F_1-score, and AUC. The best test performances are summarized in Table 2.

Table 2: Test attacks detection performance.

Method	UNSW-NB15			CICIDS2017			ToN-IoT		
	***Acc* (%)**	**F_1 (%)**	***AUC***	***Acc* (%)**	**F_1 (%)**	***AUC***	***Acc* (%)**	**F_1 (%)**	***AUC***
DL-B	95.56	95.21	0.961	95.71	95.73	0.953	95.38	95.43	0.965
DL-PSO-FS	96.87	96.98	0.974	96.97	96.97	0.967	96.98	96.02	0.971
DL-PSO-HP	97.53	97.92	0.978	97.71	97.41	0.971	97.61	97.32	0.981
DL-PSO-S	98.65	98.78	0.981	98.13	98.77	0.983	98.03	98.42	0.988
DL-PSO-CE	99.98	99.99	0.999	99.98	99.97	0.995	99.97	99.99	0.999

Static PSO-based variants provide consistent but limited gains over DL-B (1%–3%), whereas DL-PSO-CE achieves substantially larger improvements (4%–5% over DL-B and 1%–3% over the best static PSO variant), while reaching near-perfect AUC values (≥0.995) across all datasets. The strongest gains are observed on ToN-IoT, highlighting the robustness and generalization capability of the proposed co-evolutionary mechanism under heterogeneous and imbalanced traffic conditions.

In addition, the per-class performance shown in Fig. 1 confirms that DL-PSO-CE achieves consistently higher per-class precisions, recall and F_1, with gains of up to 1%–5% on attack classes compared to the baseline

and static PSO-based variants. The proposed DL-PSO-CE consistently achieves the highest discrimination performance, with *AUC* values above 0.995 on UNSW-NB15, CICIDS2017, and ToN-IoT. While static PSO-based variants improve upon the baseline, their ROC curves degrade in low false positive rate regions. In contrast, DL-PSO-CE maintains high true positive rates, particularly on ToN-IoT, demonstrating superior robustness in heterogeneous and imbalanced traffic scenarios.

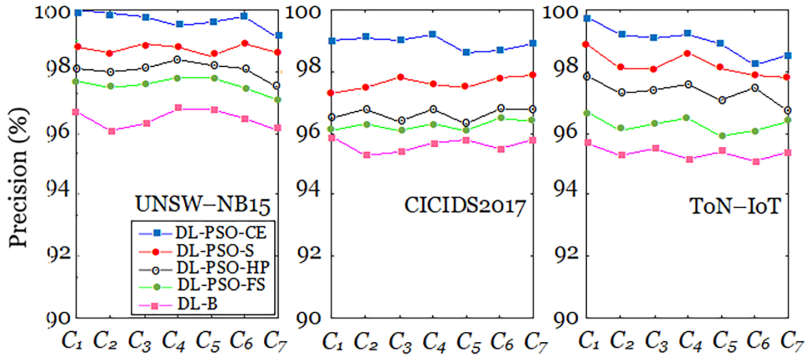

Figure 1: Per-class detection performance.

Fig. 2 presents the ROC curves for all datasets. The proposed DL-PSO-CE consistently achieves the highest discrimination performance, with *AUC* values above 0.995 on UNSW-NB15, CICIDS2017, and ToN-IoT. While static PSO-based variants improve upon the baseline, their ROC curves degrade in low false positive rate regions. In contrast, DL-PSO-CE maintains high true positive rates, particularly on ToN-IoT, demonstrating superior robustness in heterogeneous and imbalanced traffic scenarios.

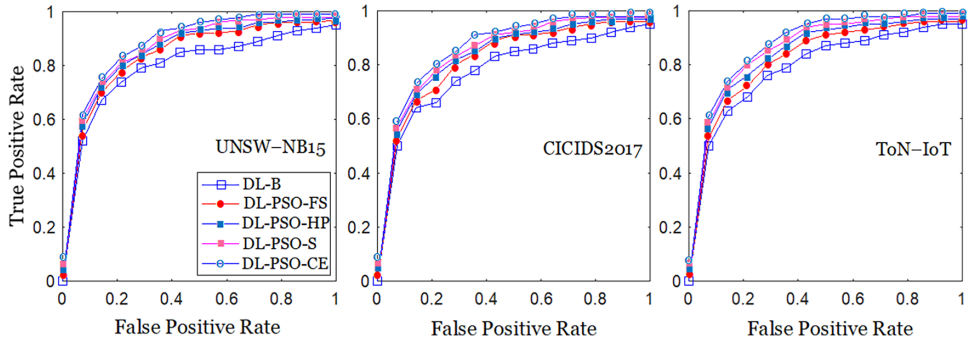

Figure 2: ROC curves.

The high detection performance may raise concerns regarding overfitting or data leakage. To mitigate this risk, strict preprocessing was applied, including removal of duplicate flows and stratified splitting into non-overlapping training, validation, and test sets. Furthermore, evaluation on three heterogeneous datasets (UNSW-NB15, CICIDS2017, and ToN-IoT) supports the generalization capability of the proposed framework.

5.2 Statistical Significance

Statistical significance was assessed using both paired *t*-tests and *Wilcoxon signed-rank tests* on UNSW-NB15, CICIDS2017, and ToN-IoT datasets.

To control for multiple comparisons, the *Holm–Bonferroni* correction was applied to all statistical tests. All comparisons remain statistically significant after adjustment, confirming that the observed performance improvements are robust and not due to multiple testing effects.

As summarized in Tables 3 and 4, the proposed DL-PSO-CE framework consistently outperforms all baseline and static PSO-based variants. The paired t-test results show t-values ranging from 3.87 to 9.91, with all corresponding p-values below 0.01, indicating statistically significant improvements in accuracy across all datasets. These findings are further confirmed by the *Wilcoxon signed-rank test*, where Z-scores range from −2.04 to −3.92, with all comparisons remaining significant ($p < 0.05$).

Table 3: Paired t-test results (accuracy%) across multiple datasets.

Comparison	UNSW-NB15			CICIDS2017			ToN-IoT		
	Mean	t-Stat	p-Value	Mean	t-Stat	p-Value	Mean	t-Stat	p-Value
DL-PSO-CE vs. DL-B	+2.31	7.84	0.001	+3.05	9.91	0.001	+2.67	8.46	0.001
DL-PSO-CE vs. DL-PSO-FS	+1.62	6.19	0.001	+2.14	8.02	0.001	+1.83	6.78	0.001
DL-PSO-CE vs. DL-PSO-HP	+1.08	4.91	0.002	+1.56	6.43	0.001	+1.29	5.11	0.001
DL-PSO-CE vs. DL-PSO-S	+0.71	3.87	0.009	+1.02	4.72	0.003	+0.86	3.95	0.008

Table 4: Wilcoxon signed-rank test results (accuracy%) across multiple datasets.

Comparison	UNSW–NB15			CICIDS2017			ToN-IoT		
	W	Z	p-Value	W	Z	p-Value	W	Z	p-Value
DL-PSO-CE vs. DL-B	0	−3.71	0.001	0	−3.92	0.001	1	−3.58	0.001
DL-PSO-CE vs. DL-PSO-FS	4	−3.12	0.002	2	−3.54	0.001	3	−3.09	0.002
DL-PSO-CE vs. DL-PSO-HP	6	−2.67	0.008	4	−3.01	0.003	5	−2.71	0.007
DL-PSO-CE vs. DL-PSO-S	10	−2.04	0.041	7	−2.33	0.020	9	−2.12	0.034

Overall, the statistical analysis demonstrates that the performance gains achieved by DL-PSO-CE are consistent, robust across datasets, and not attributable to random variation, thereby supporting the effectiveness and generalizability of the proposed co-evolutionary framework.

5.3 Training Time and Inference Latency

Table 5 summarizes the computational performance of the evaluated methods. As expected, PSO-assisted variants require longer training time due to the additional optimization steps. Among them, the proposed DL-PSO-CE framework exhibits the highest training cost, as it involves iterative PSO updates and dynamic structural adaptation through ADB activation. Nevertheless, the additional overhead remains moderate.

Table 5: Computational performance comparison on the CICIDS2017 dataset (161K samples).

Method	DL-B	DL-PSO-FS	DL-PSO-HP	DL-PSO-S	DL-PSO-CE
Training Time (CPU, s)	621	734	798	827	883
Training Time (GPU, s)	184	213	229	238	251
Avg. PSO iteration time (s)	–	1.21	1.34	1.42	1.55
ADB switching overhead (ms)	–	–	–	3.8	4.6
Inference Latency (ms/sample)	0.85	0.88	0.87	0.90	0.92
Latency (batch = 32)	0.41	0.43	0.42	0.44	0.46
Latency (batch = 128)	0.23	0.25	0.24	0.26	0.28

Importantly, inference latency remains below 1 ms per sample across all configurations, indicating that the proposed method remains suitable for near real-time deployment. Further analysis shows that the computational overhead associated with PSO iterations and ADB switching is relatively small compared with the overall training cost, while GPU acceleration significantly reduces training time.

The additional computational overhead introduced by PSO iterations and ADB structural adaptation remains moderate compared to the overall training cost. It should be noted that the reported training time includes both gradient-based model training and PSO optimization steps.

Although the co-evolutionary framework increases training time due to PSO-driven structural optimization, the additional overhead remains moderate relative to the overall training process. Considering the significant improvement in detection performance and the low inference latency (<1 ms per sample), the proposed framework provides a favorable trade-off between computational cost and detection accuracy.

5.4 Sensitivity Analysis

Sensitivity was evaluated for the four algorithms across three PSO parameters (swarm size, iteration count, inertia weight), each assessed on a different dataset (Fig. 3).

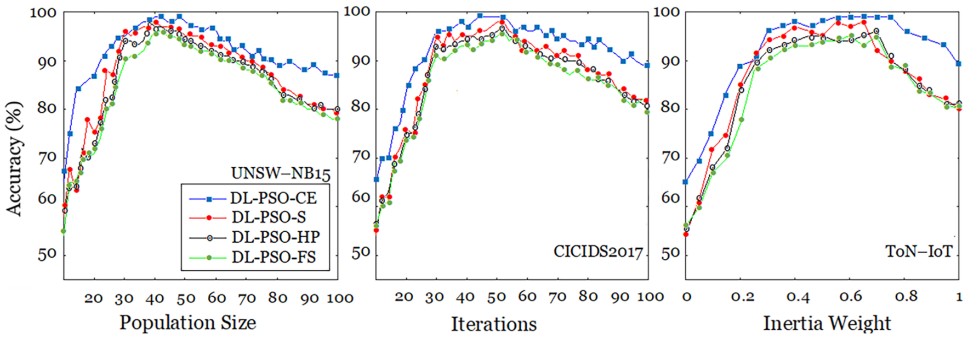

Figure 3: Sensitivity of the proposed algorithms performance to PSO parameters.

The accuracy-based sensitivity curves in Fig. 3 show that the proposed DL-PSO-CE framework consistently outperforms all baseline and static hybrid variants. While DL-B and offline PSO-based approaches exhibit strong sensitivity to inertia weight, population size, and iteration count, the co-evolutionary model maintains stable and high accuracy across a wide parameter range.

This robustness is attributed to the continuous interaction between deep latent representations and PSO-driven structural adaptation, which enables dynamic self-correction during training. These results confirm that performance gains arise from co-evolution rather than from isolated or static optimization. The results further validate the selected PSO parameter (swarm size = 40, 50 iterations, inertia weight $\omega = 0.7$).

5.5 Cross-Dataset Generalization

To evaluate robustness under distribution shift, a cross-dataset experiment was conducted using a *binary setting* (*Normal* vs. *DoS/DDoS*) consistent with Table 1. Models were trained on one dataset and directly tested on another without fine-tuning, ensuring a strict generalization assessment.

Results in Table 6 show that DL-PSO-CE maintains strong cross-dataset generalization (96.11%–97.12% Accuracy), with limited performance degradation compared to within-dataset results (2.86%–3.87%) confirming its robustness under distribution shifts.

Table 6: Cross-dataset accuracy (%) of DL-PSO-CE (binary classification: normal vs. Dos/DDos).

	Test Dataset		
Train Dataset	**CICIDS2017**	**UNSW-NB15**	**ToN-IoT**
CICIDS2017	99.98	96.21	96.11
UNSW-NB15	97.12	99.98	96.79
ToN-IoT	97.10	96.67	99.97

5.6 Comparison with State-of-the-Art Works

Table 7 compares the proposed framework with several recent intrusion detection methods evaluated on the CICIDS2017, UNSW-NB15, and ToN-IoT datasets.

Table 7: Comparison with some previous works on CICIDS2017, UNSW-NB15 and ToN-IoT datasets.

Reference	Method	Dataset	*Acc* (%)	*R* (%)	F_1 (%)
[14]	NAS-Net	CICIDS 2017	–	–	99.57
[15]	LSTM		99.10	–	–
[16]	DNN-LSTM		99.25	–	–
[17]	GA-1D-CNN		99.71	99	99.00
This work	**DL-PSO-CE**		**99.98**	**99.98**	**99.97**
[1]	CNN-LSTM	UNSW-NB15	92.90	–	–
[3]	GWO-CNN		97.08	97.21	97.08
[12]	Cyber–MLP		98.20	–	–
[17]	PSO-1D-CNN		99.28	98.00	98.00
This work	**DL-PSO-CE**		**99.98**	**99.98**	**99.99**
[18]	LSTM	ToN-IoT	98.76	–	–
[19]	XGBoost		98.30	98.30	98.30
[20]	ANN		99.44	–	–
This work	**DL-PSO-CE**		**99.97**	**99.96**	**99.99**

On CICIDS2017, the proposed DL-PSO-CE framework achieves 99.98% accuracy and 99.97% F_1-score, outperforming several deep and hybrid IDS models, including LSTM-based approaches [15], DNN-LSTM architectures [16], and GA-optimized CNN models [17]. It also improves upon more recent architectures such as NAS-Net [14], which report F_1-scores below 99.6%.

For UNSW-NB15, existing approaches including CNN-LSTM [1], GWO-CNN [3], Cyber-MLP [12] and PSO-1D-CNN [17], achieve accuracies between 92.90% and 99.28%. In comparison, DL-PSO-CE reaches 99.98% accuracy and 99.99% F_1-score, indicating improved detection performance in heterogeneous network environments.

On ToN-IoT, machine learning and deep learning methods such as LSTM [18], XGBoost [19], and ANN [20] report accuracies between 98.30% and 99.44%. The proposed framework consistently achieves 99.97% accuracy and 99.99% F_1-score, demonstrating strong robustness for large-scale IoT traffic analysis.

Overall, the results suggest that integrating deep representation learning with PSO-based structural co-evolution provides improved and consistent detection performance compared with conventional deep learning models and static optimization strategies.

5.7 Discussion

The proposed DL-PSO-CE framework achieves state-of-the-art detection performance across the UNSW-NB15, CICIDS2017, and ToN-IoT datasets, while maintaining low inference latency (below 1 ms per sample). Compared with baseline deep learning models, static PSO-based variants, and conventional hybrid approaches, the proposed method consistently demonstrates improved detection capability and robustness. Sensitivity analyses further confirm the stability of the framework with respect to variations in PSO parameters, while paired t-tests and *Wilcoxon signed-rank tests* indicate that the observed performance improvements are statistically significant ($p < 0.01$) across datasets and evaluation metrics.

Unlike static feature selection or hyperparameter optimization strategies, the proposed co-evolutionary mechanism dynamically refines the internal decision structure of the model during training. This adaptive behavior allows the IDS to better capture complex and rare attack patterns by continuously adjusting the activation of Adaptive Decision Blocks based on latent representation signals.

Despite these advantages, several limitations should be acknowledged. First, the co-evolutionary mechanism introduces a moderate training overhead due to PSO-driven structural optimization. Second, the effectiveness of the approach depends on the quality of the latent representations learned by the deep encoder. In addition, the current fitness formulation relies on a weighted-sum aggregation of objectives. Future work could explore alternative multi-objective optimization strategies, such as Pareto-based PSO, to explicitly balance competing objectives during structural adaptation.

Finally, practical deployment in real-world network infrastructures may require addressing additional challenges such as streaming traffic processing, concept drift, and memory constraints in large-scale or resource-constrained environments.

6 Conclusion

In this work, a DL–PSO Co-Evolution intrusion detection framework was presented, in which deep representation learning and metaheuristic-driven structural adaptation are tightly integrated. A CNN-LSTM encoder was coupled with PSO-controlled Adaptive Decision Blocks, allowing the limitations of conventional hybrid IDS models based on static or offline optimization to be overcome. Through a latent-space-guided co-evolutionary loop, continuous and data-driven refinement of the decision structure was

enabled during training, thereby improving robustness against complex, rare or previously unseen attack patterns while maintaining architectural compactness.

Extensive experiments conducted on UNSW-NB15, CICIDS2017, and ToN-IoT datasets, and state-of-the-art performance was consistently achieved, with detection accuracies, recall and F_1-scores exceeding *99.97%*, and AUC values greater than *0.995*.

In comparison with baseline deep learning models and static PSO-assisted variants, substantial and consistent improvements were observed, particularly in highly imbalanced and IoT-oriented traffic scenarios. Sensitivity analyses further demonstrated that stable performance was maintained under variations of PSO and architectural parameters, confirming the robustness of the proposed framework.

Despite the introduction of a co-evolutionary optimization mechanism, low inference latency (less than 1 ms per sample) was preserved, indicating suitability for near real-time deployment. Statistical validation using paired *t*-tests and *Wilcoxon signed-rank tests* confirmed that the observed performance improvements were both statistically significant and practically meaningful. In addition, a preliminary cross-dataset evaluation further demonstrated the robustness of the proposed framework under distribution shifts, highlighting its potential for real-world deployment across heterogeneous network environments

Future work will focus on extending the framework to distributed and federated IDS settings, integrating explainable learning mechanisms, exploring alternative multi-objective optimization strategies, and evaluating scalability on resource-constrained edge and IoT environments.

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: Conception and design, data curation, literature review, analysis and interpretation of results: Soukaina Mjahed and Ouail Mjahed; draft manuscript preparation: Ouail Mjahed, writing—review and editing, supervision: Soukaina Mjahed. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data supporting the conclusions of this study are freely available at the websites cited in references [24–26].

Ethics Approval: Not applicable.

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

Abbreviations

ADB	Adaptive Decision Block
ANN	Artificial Neural Network
CICIDS2017	Canadian Institute for Cybersecurity Intrusion Detection System dataset 2017
CNN	Convolutional Neural Network
DARTS	Differentiable Architecture Search
DL	Deep Learning
DL-B	Baseline Deep Learning
GA	Genetic Algorithm
GWO	Grey Wolf Optimizer
IDS	Intrusion Detection System
LSTM	Long Short-Term Memory
NAS	Neural Architecture Search

PSO	Particle Swarm Optimization
PSO-CE	PSO-based Co-Evolutionary framework
PSO-FS	PSO-based Feature Selection
PSO-HP	PSO-based Hyper-Parameter optimization
PSO-S	PSO-based Static structural optimization
RNN	Recurrent Neural Network
ToN-IoT	Telemetry of Network and Internet of Things Dataset
UNSW-NB15	University of New South Wales-Network Based dataset 2015
XGBoost	eXtreme Gradient Boosting

References

1. Altunay HC, Albayrak Z. A hybrid CNN-LSTM-based intrusion detection system for industrial IoT networks. Int J Eng Sci Technol. 2023;38:101322. doi:10.1016/j.jestch.2022.101322.
2. Alashjaee AM. Deep learning for network security: an Attention-CNN-LSTM model for accurate intrusion detection. Sci Rep. 2025;15(1):21856. doi:10.1038/s41598-025-07706-y.
3. Kaissar A, Nassif AB, Soudan B, Injadat MN. Enhancing CNN-based network intrusion detection through hyperparameter optimization. Intell Syst Appl. 2025;26(2):200528. doi:10.1016/j.iswa.2025.200528.
4. Van LTH, Thuan LD, Huong PV, Minh NH. An overall optimization model using metaheuristic algorithms for the CNN-based IoT attack detection problem. Comput Mater Contin. 2026;87(1):81. doi:10.32604/cmc.2025.075027.
5. Kan X, Fan Y, Fang Z, Cao L, Xiong NN, Yang D, et al. A novel IoT network intrusion detection approach based on adaptive particle swarm optimization convolutional neural network. Inf Sci. 2021;568(5):147–62. doi:10.1016/j.ins.2021.03.060.
6. Bahaa A, Sayed A, Elfangary L, Fahmy H. A novel hybrid optimization enabled robust CNN algorithm for an IoT network intrusion detection approach. PLoS One. 2022;17(12):e0278493. doi:10.1371/journal.pone.0278493.
7. Nguyen MT, Kim K. Genetic convolutional neural network for intrusion detection systems. Future Gener Comput Syst. 2020;113:418–27. doi:10.1016/j.future.2020.07.042.
8. Mjahed O, El Hadaj S, El Guarmah E, Mjahed S. Improved supervised and unsupervised metaheuristic-based approaches to detect intrusion in various datasets. Comput Model Eng Sci. 2023;137(1):265–98. doi:10.32604/cmes.2023.027581.
9. Rajesh KP, Santhi P. Unified deep learning approach for efficient intrusion detection system using integrated spatial-temporal features. Knowl-Based Syst. 2021;226(1):107132. doi:10.1016/j.knosys.2021.107132.
10. Bala R, Nagpal R. Performance evaluation of firefly optimization for intrusion detection system. Int J Sci Technol Res. 2020;9(03):4754–8. doi:10.15623/ijret.2018.0710003.
11. Bhuvaneshwari KS, Venkatachalam K, Hubalovský S, Trojovský P, Prabu P. Improved dragonfly optimizer for intrusion detection using deep clustering CNN-PSO classifier. Comput Mater Contin. 2022;70(3):5949–65. doi:10.32604/cmc.2022.020769.
12. Al Hwaitat AK, Fakhouri HN. Adaptive cybersecurity neural networks: an evolutionary approach for enhanced attack detection and classification. Appl Sci. 2024;14(19):9142. doi:10.3390/app14199142.
13. Xinyu W, Yifan G, Xiaoyuan L, Xinping G. Detection of dummy data injection attacks by using particle swarm optimization-attention temporal graph convolutional network model in power system. Eng Appl Artif Intell. 2026;171(5):114259. doi:10.1016/j.engappai.2026.114259.
14. Lyu R, He M, Zhang Y, Jin L, Wang X. Network intrusion detection based on an efficient neural architecture search. Symmetry. 2021;13(8):1453. doi:10.3390/sym13081453.
15. Sarhan M, Layeghy S, Portmann M. Towards a standard feature set for network intrusion detection system datasets. Mob Netw Appl. 2022;27(1):357–70. doi:10.1007/s11036-021-01843-0.
16. Shukla AK, Dwivedi S, Mishra A. An effective hybrid deep learning metaheuristic model for robust IoT intrusion detection. Discov Comput. 2025;28(1):200. doi:10.1007/s10791-025-09708-w.
17. Kilichev D, Kim W. Hyperparameter optimization for 1D-CNN-based network intrusion detection using GA and PSO. Mathematics. 2023;11(17):3724. doi:10.3390/math11173724.

18. Devendiran R, Turukmane AV. Dugat-lstm: deep learning based network intrusion detection system using chaotic optimization strategy. Expert Syst Appl. 2024;245(2):123027. doi:10.1016/j.eswa.2023.123027.
19. Gad AR, Nashat AA, Barkat TM. Intrusion Detection system using machine learning for vehicular *ad hoc* networks based on ToN-IoT dataset. IEEE Access. 2021;9:142206–17. doi:10.1109/ACCESS.2021.3120626.
20. Kumar P, Tripathi R, Gupta GP. P2IDF: a privacy-preserving based intrusion detection framework for software defined Internet of Things-Fog (SDIoT-Fog). In: Proceedings of the 2021 International Conference on Distributed Computing and Networking; 2021 Jan 5; New York, NY, USA. p. 37–42. doi:10.1145/3427477.3429989.
21. LeCun Y, Hinton G. Deep learning. Nature. 2015;521:436–44. doi:10.1038/nature14539.
22. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735–80. doi:10.1162/neco.1997.9.8.1735.
23. Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of the ICNN'95-International Conference on Neural Networks; 1995 Nov 27; Perth, Australia. p. 1942–8. doi:10.1109/ICNN.1995.488968.
24. UNSW-NB15 Dataset. [cited 2025 Nov 2]. Available from: https://research.unsw.edu.au/projects/unsw-nb15-dataset.
25. CICIDS2017 Dataset. [cited 2025 Dec 12]. Available from: https://www.unb.ca/cic/datasets/ids-2017.html.
26. TON_IoT Dataset. [cited 2025 Dec 15]. Available from: https://research.unsw.edu.au/projects/toniot-datasets.
27. Chawla NV, Bowyer KW, Hall L, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. J Artif Intell Res. 2002;16:321–57. doi:10.1613/jair.953.
28. Lundberg SM, Lee SI. A unified approach to interpreting model predictions. In: Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17); 2017 Dec 4–9; Long Beach, CA, USA. p. 4766–77.
29. Rajić V. Statistical hypothesis testing: a comprehensive review of theory, methods, and applications. Mathematics. 2026;14(2):300. doi:10.3390/math14020300.