IOT INTRUSION DETECTION USING MIXED DUNG BEETLE OPTIMISATION-BASED FEATURE SELECTION AND ATTENTION-BASED BIRNN WITH HYPERPARAMETER TUNING

P Devabalan[1], S Meenakshi Sundaram[2], T Sangeetha[3], L Rajesh[4*]

[1] Department of Information Technology, AAA College of Engineering and Technology (Autonomous), Sivakasi, Tamil Nadu, India.

[2] Department of Computer Science and Engineering, AAA College of Engineering and Technology (Autonomous), Sivakasi, Tamil Nadu, India.

[3] Department of Biomedical Engineering, Karpaga Vinayaga College of Engineering and Technology, Karpaga Vinayaga Educational Lore for Learning (Deemed to be University), Chengalpattu – 603 308, Tamil Nadu, India.

[4*] Department of Mechatronics Engineering, SRM Institute of Science and Technology, SRM Nagar, Kattankulathur – 603 203, Chengalpattu, Tamil Nadu, India.

Email: devabalanme@gmail.com[1], bosemeena@gmail.com[2], sangee4contact@gmail.com[3], rl8367@srmist.edu.in[4]* (Corresponding author)

Abstract

The exponential growth of the Internet of Things (IoT) has led to new cybersecurity threats from the rise in the number of various and resource-limited connected devices. Conventional intrusion detection systems (IDSs) may not be efficient in dealing with real-time, high-dimensional, class-imbalanced nonstationary IoT traffic data. To overcome these limitations, in this paper, an IoT intrusion detection framework is proposed based on Mixed Dung Beetle Optimisation with Deep Learning Cyber Security (MDBODR-DLCS). To this end, Z-score normalisation is first performed to normalise the features of network traffic, which enhances the stability of subsequent models. Then, Mixed Dung Beetle Optimisation (MDBO) is used for optimal feature selection and dimensionality reduction to remove redundant and irrelevant attributes. To employ the established temporal dependency information and contextual traffic patterns as features for intrusion classification, our proposed method utilises an Attention-Based Bidirectional Recurrent Neural Network (ABiRNN). Besides, Artificial Rabbits Optimisation (ARO) is employed to fine-tune hyperparameters to improve classification accuracy and convergence speed. We conduct an experimental evaluation on the WSNDS dataset that shows that our framework is able to outperform state-of-the-art intrusion detection solutions, with over 99% accuracy and very high sensitivity, specificity, F-score, and AUC for multiple attack classes. The comparative analyses showed that the proposed method outperforms other existing methods using machine learning and deep learning-based IDS structures while maintaining low computational complexity with an average execution time of 0.60 seconds per instance. The results were found to demonstrate that the developed method MDBODR-DLCS efficiently achieves high accuracy performance in real time and scalability, while being resilient due to its ability to detect even the variations an attacker might introduce, whilst maintaining very low overhead, making it appropriate for resource-constrained environments such as IoT.

Keywords: Internet of Things (IoT); Intrusion Detection System (IDS); Mixed Dung Beetle Optimisation (MDBO); Attention-Based BiRNN; Artificial Rabbits Optimisation (ARO); WSNDS Dataset, Imbalanced Classification; Feature Selection; Attention Mechanism; Deep Learning.

1.0 INTRODUCTION

IT security is one of the key challenges of today's information technology (IT), particularly in the ever-expanding field of IoT [1]. And the secure communication and data protection are even more sticky issues when it comes to billions of networked devices spread in application areas like industrial automation, smart homes, intelligent transportation and smart cities [2–5]. The estimated volume of 'Internet of Things' (IoT) devices was over 50 billion in 2020 [6], indicating the scale of connectivity anticipated in the digital world. However, this evolution brings new threats to the confidentiality of the data, the integrity of the system, and the availability of the services: indeed, IoT devices can be subject to malicious intrusions [7].

Ensuring security in IoT scenarios is a non-trivial problem due to the presence of numerous devices that are resource-constrained, geographically dispersed, and often rely on wireless communications, which are easily susceptible to sophisticated cyber-attacks [8–10]. Possible attacks are privilege escalation, denial of service (DoS), eavesdropping and web-based injection, which pose risks of data leakage, tampering and privacy abuse. As such, there is an urgent demand for strong intrusion detection systems (IDSs) that can serve as the last line of defence to detect and combat these threats [11–13]. Unlike traditional security approaches like encryption or authentication only, IDSs are capable of making a dynamic distinction between malicious and benign traffic based on behavioural deviation and the attack-specific signature [14–15]. However, it is challenging to develop effective ID devices for IoT, because of the high-dimensional traffic data, constrained device resources, and dynamic threats of cyber-attack [16–18].

Machine learning (ML) and deep learning (DL) are important topics because they can be used for large-scale network traffic analysis, pattern recognition and intelligent IDS [19–21]. DL approaches are able to model complex data relationships and are capable of detecting zero-day attacks when well-optimised feature representation and hyperparameters are selected [22–23]. Furthermore, standard benchmark datasets are invaluable for the validation of IDS solutions, which need to encompass benign traffic and a variety of attack data, maintain a real-time representation of the environment, and must remain publicly available for reasons of reproducibility [24–25]. In this study, we present a hybrid algorithmic framework for IOT, referred to as MDBO with dimensionality reduction using a deep learning-based cybersecurity solution (MDBODR-DLCS) to address these challenges. The method integrates several complementary strategies:

- Pre-processing technique of z-score normalisation is applied to make feature scaling the same, which in effect increases predictive performance by normalising the input distribution [26].
- Core component of our proposed HBD-SVN model is the discarding of irrelevant features and preserving of the most discriminative ones through Mixed Dung Beetle Optimisation (MDBO) based feature selection technique [27].
- An Attention-based Bidirectional Recurrent Neural Network (ABiRNN) is used to catch the temporal dependencies in the IoT traffic data for efficient intrusion detection [28].

Finally, we introduce the Artificial Rabbits Optimisation (ARO) for hyperparameter optimisation to enable model flexibility and smooth training [29]. The amalgamation of these approaches provides an integrated and adaptive IDS system, ultimately improving classification performance, reducing algorithm complexity, and effectively addressing dynamic IoT attacks [30–31]. Experimental results on benchmark IDS datasets validate the efficiency of the proposed MDBODR-DLCS that achieves better correctness, exactness, recollection and F1-score than several existing approaches [32–33]. This renders our solution to be a formal-proved cybersecurity to protect the IoT infrastructure.

Several important research aspects still require addressing, despite recent progress on IoT intrusion detection systems:

- Due to high-dimensional and redundant characteristics of IoT traffic, existing IDS approaches also suffer excessive computation overhead.
- Most deep-learning-based IDS models do not work well for handling class imbalance and minority attack detection on realistic IoT datasets.
- Though traditional optimisation algorithms are more susceptible to early convergence and less pliable in dynamic conditions of cyber-attack strategies.
- The existing approaches provide high accuracy but require significant computational power, hence not ideal for real-time and resource-constrained IoT environments.
- Few-level interpretations of the IDS rely on unified structures in which feature selection, temporal learning and hyperparameter optimisation are fused.

The primary purpose of this work is to design a generalised, effective intrusion detection system for IoT platforms employing deep learning and metaheuristic optimisation strategies. The aim of the framework is to find a better way of feature selection, minimising computational complexity and maximising identification susceptibility in defence against high-dimensionality and unbalanced IoT traffic data for delivering better accuracy while improving real-time cybersecurity performance with an optimal attention-based learning module.

We present a new MDBODR-DLCS framework, which combines Applied Mixed Dung Beetle Optimisation (MDBO), Attention-Based BiRNN, and Artificial Rabbits Optimisation (ARO) for IoT intrusion detection. This model achieves feature selection, intrusion classification and hyperparameter optimisation together under a common framework. Experimental results on the WSNDS dataset show a remarkable improvement in accuracy, sensitivity, F-score and computational efficiency over other state-of-the-art approaches for intrusion detection.

Section I presents emerging cybersecurity issues in IoT networks and motivates the need for sophisticated IDS. In Section 2, we discuss related works, existing deep learning and metaheuristic-based IDS approaches and their drawbacks. Section 3 describes the proposed MDBODR-DLCS, including pre-processing, feature selection,

deep learning classification and hyperparameter optimisation approaches. The experimental results and discussion are shown in Section 4 to verify the superiority and efficiency of the proposed algorithm with comparison. The conclusions and future work section round off the paper to highlight contributions, bring forward practical implications, and set the stage for future research in the field of IoT security.

2.0 RELATED WORKS

Lately, the mixed metaheuristic algorithms and deep learning frameworks are used in the development of intrusion detection systems (IDS) for IoT networks to deal with complexities, high-dimensionality and the dynamic nature of traffic data [6, 11, 19]. For example, Potnurwar et al. [34] proposed a deep learning-based IDS for Industrial IoT, in which a DFFNN and mixed rule-based feature selection (encapsulated within an unsupervised approach) are used in order to increase the classification accuracy in the detection of attacks (e.g., in NSL-KDD and UNSW-NB15 datasets). The presented method showed superior detection performance while relying on rule-based traffic selection; it may be hard to adapt to new attack behaviours [20, 22].

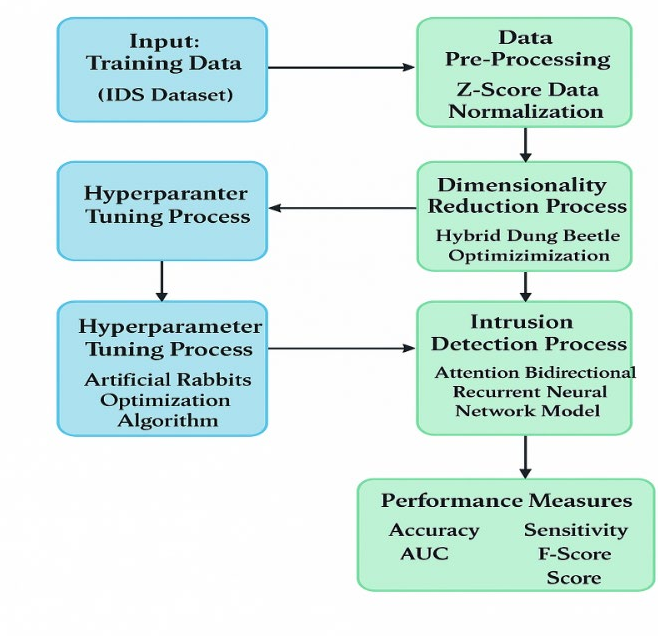

Fig. 1: Architecture of AI-Enabled Intrusion Detection Systems

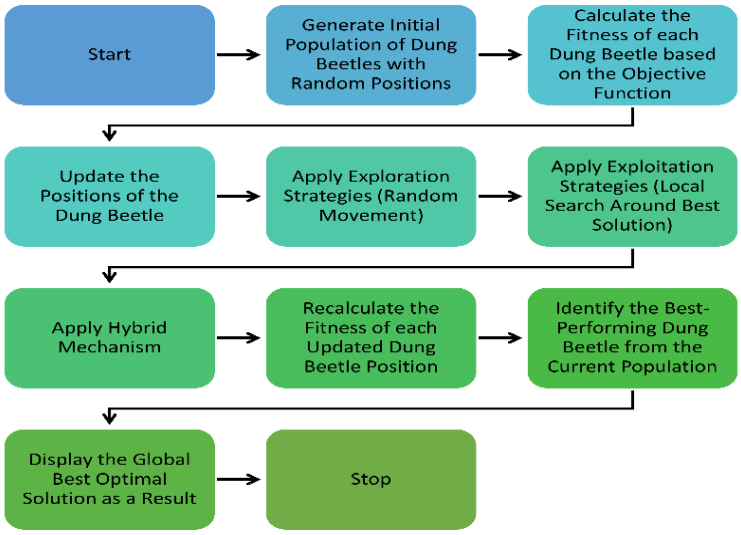

Fig. 2: Schematic Representation of the MDBO Algorithm

A self-attention progressive GAN (SAPGAN) based architecture for an IoT intrusion detection system was reported in Nature (2024), which adopted the recursive feature elimination (RFE) method for feature selection, and classification was conducted using a convolutional neural network (CNN). Experimental results on BoT-IoT and other standard datasets demonstrated excellent accuracy at the expense of F-score in some cases [23]. The system is based on strong pre-processing and requires superior computation for GAN-based training. Fig. 1 provides an overview of the AI-enabled IDS architecture with the whole process, including IoT traffic collection, preprocessing, feature extraction, classification and finally intrusion detection decision-making, provided as a part of the proposed cybersecurity framework. In the example scenario, different behavioural strategies of the dung beetle are utilised to train and adjust an adaptive feature selection with dimensionality reduction structure for a proposed IDS pipeline (as shown in Fig. 2).

Pure and mixed metaheuristic-based techniques are still successfully being employed for feature selection in the context of IDS [24, 25]. Exhaustive experiments on the ToN-IoT dataset achieved 98.11% accuracy at the highest, yet the authors reported the remaining issues of accommodating the scale of raw data, real-time analysis, model generalizability for production, etc.

In another recent study, deep learning models and Ant Colony Optimisation (ACO) were combined for the design of IDS. The chosen best features of the NSL-KDD dataset enabled the neural model to classify more effectively with increased accuracy to reduce the training time [26]. However, the effectiveness of the method is dataset-specific, and the adaptation to evolving IoT attack vectors is an essential requirement for wider application. Meanwhile, additional systematic reviews and surveys, such as that published in 2024, indicate that metaheuristic feature selection (e.g., evolutionary, swarm, physics-based algorithms) continues to be extensively employed to optimise selectivity and efficiency in ML/DL-based IDSs [27–28]. Nonetheless, these still result in computational overhead/energy restrictions trade-offs and resilience performance trade-offs in resource-scarce IoT environments [29–30].

But despite these bright spots, there is much that has been missing. A great number of mixed and metaheuristic-based approaches present high accuracy on public datasets, yet the generalisation of such approaches to dynamical IoT in real-world contexts is limited [31]. Finally, generalisation to new or advanced cyber-attacks in a changing threat environment without continuously retraining is a key research issue to be addressed [32–33]. The above open questions underscore the necessity of IDS systems that provide a trade-off between the effectiveness of feature selection and detection performance on one hand, and resource constraints on the other in realistic IoT environments.

3.0 THE PROPOSED METHOD

Within the structure of this problem, we propose a new MDBODR-DLCS for the identification problem in IoT. By combining a deeper data normalisation process, new feature synthesis and selection techniques, deep learning-based classification and metaheuristic search of hyperparameters, MDBODR-DLCS aims to achieve the best possible performance in attack detection. Fig. 3 illustrates the complete end-to-end workflow of the proposed MDBODR-DLCS framework, including Z-score normalisation, MDBO-based feature selection, ABiRNN classification, and ARO-based hyperparameter tuning for efficient IoT intrusion detection.

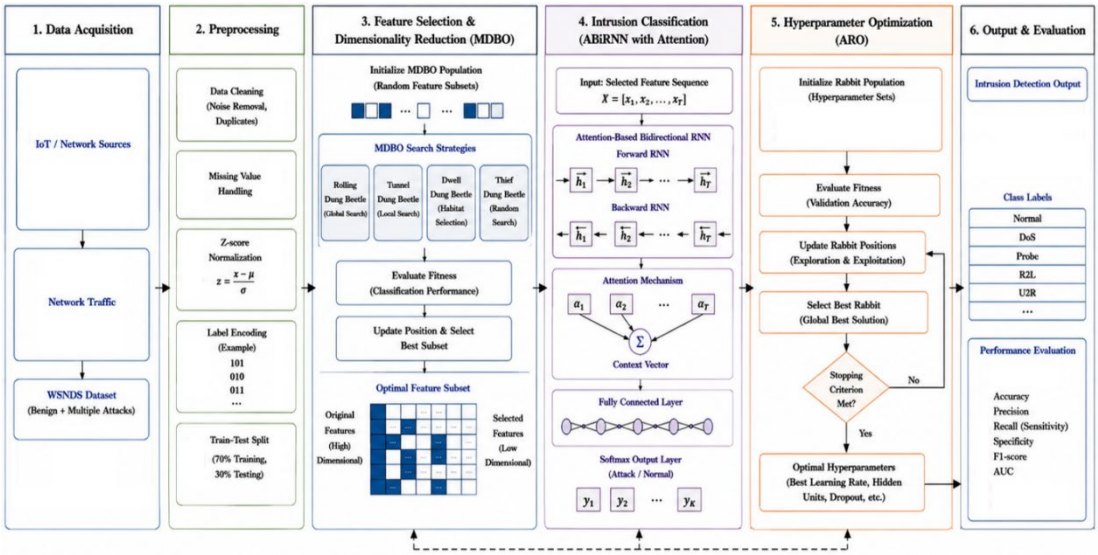

Fig. 3: End-to-End Workflow of the Proposed MDBODR-DLCS Framework for IoT Intrusion Detection

3.1 Data Standardisation through Z-Scores

At the preprocessing stage, the Z-score normalisation is applied on the WSNDS intrusion detection dataset in the MDBODR-DLCS framework. This normalisation approach normalises network traffic features to diminish the variation in scale among attributes for improved stability and convergence performance during classification. This method is especially useful to manage the differences between feature scales, not allowing any of the characteristics to have more importance in the model selection. To ensure appropriate normalisation for such scale-sensitive algorithms as neural networks and the K-nearest neighbour algorithm, this normalisation is quite practical. For Z-score normalisation, unlike min–max normalisation, there is the possibility of point distortion if outliers are present in the data. This is because this method scales data by centring features on the mean and normalising to the standard deviation.

This aids gradient-based optimisation by reducing the amount of "saddle" points and also stably and quickly converges the training process. For instance, in the field of intrusion detection systems (IDS), these standardisations are crucial as they deliver a common base to merge heterogeneous entities such as system logs and network traffic observables. It enables the IDS to compare anomaly scores among different dimensions and to detect any anomalous behaviour that can be related to potential threats. In the end, Z-score normalisation improves both the robustness and interpretability of the model, providing strong support for effective detection of anomalies.

3.1.1 Dataset Description and Preprocessing Strategy

In this section, WSNDS is a well-known public dataset for an intrusion detection task that has 384,936 traffic instances classified into five classes: Normal, Blackhole Attack, Grayhole Attack, Flooding Attack and Scheduling Attacks. The proposed MDBODR-DLCS framework was evaluated on the WSNDS intrusion detection dataset. The dataset has many features related to packet transmission behaviour, routing information and is used for attack characterisation for intrusion classification.

In the first stage, data preprocessing, missing and inconsistent values were verified and filtered to improve data quality. Then we applied Z-score normalisation to normalise the distributions of features as well as reduce scale disparity between attributes. This preprocessing step enhances the robustness and convergence performance of the deep learning innovator.

Two train–test split strategies were applied to evaluate the robustness of the proposed framework: 80:20 and 70:30. Such split configurations allow for thorough performance verification against one another's quality, while also investigating the generalisation potential of the proposed model.

Given the significant class imbalance observed within the WSNDS dataset, where samples corresponding to normal traffic significantly outnumber minority attack classes, we further present an attention (ADB) driven MDBO-based learning framework denoted as MDBODR-DLCS, which is designed to enhance detection performance against minority attacks. Furthermore, to ensure a balanced evaluation of classification performance in various attack types, the sensitivity/F-score/AUC evaluation metrics were also provided.

3.2 MDBO-Driven Feature Engineering Approach

To solve the problem of dimensionality reduction, the model induces the Mixed Dung Beetle Optimisation (MDBO) for feature selection. The intuitive rationale of this approach is to keep the most representative features and drop redundant or irrelevant ones. What is special in MDBO is that it quite nicely balances exploration and exploitation, which is vital for high-dimensional data. Inspired by the dung beetle's natural foraging mode, this algorithm can not only effectively avoid the local and global optimal solutions, but also converge rapidly, and it outperforms the GA and PSO. The adaptive property of MDBO makes it work well in a noisy and large-scale feature set environment, which is more suitable for the complex networks' cybersecurity data. Through optimisation of the feature subset, we not only improve the accuracy of the classification, but also reduce the computation cost in the process of classification to make the model have better generality.

The MDBO framework, which we proposed to balance exploration and exploitation features selection by using multiple adaptive search operators. Here, the rolling strategy contributes to global exploration while the tunnel and dwelling strategies allow local refinement. Also, the perturbation based on the thief enhances the population diversity and prevents early convergence in optimisation.

Algorithm 1: MDBO-Based Feature Selection
Input: Normalised IoT traffic dataset

Output: Optimal feature subset

Step 1: Initialise MDBO population with random feature subsets
Step 2: Evaluate the classification fitness of each solution
Step 3: Apply rolling beetle strategy for global exploration
Step 4: Apply tunnel and dwelling strategies for local exploitation
Step 5: Apply the thief beetle strategy to improve diversity
Step 6: Update feature subsets based on fitness values
Step 7: Select the best feature subset
Step 8: Terminate when the maximum iterations are reached

3.2.1 Performance Validation Using the Ball-Roller DBs

Ball-roller dung beetles dynamically adjust their movement directions in response to environmental cues and adaptive search behaviours while performing global exploration. This strategy is mathematically modelled to enhance the exploration ability and prevent being trapped in local optimal points during the feature selection process within the proposed MDBO framework. To illustrate the movement update mechanism of ball-roller beetles, we get

$$\begin{cases} x_{t1}(s+1) = x_{t1}(s) + a * k * x_{t1}(s-1) + b.\Delta x, & \text{if } 0.9 < \mathrm{R1} \\ x_{t1}(s+1) = x_{t1}(s) + \tan(\theta) * | x_{t1}(s) - x_t(s-1) |, & \text{if } 0.9 \leq \mathrm{R1} \end{cases} \quad (1)$$

$$\Delta \mathrm{x1} = | x_{t1}(\mathrm{s}) - \mathrm{X1^w} | \quad (2)$$

This parameter b controls the randomness of the environment and k determines how bad random deflection vertical during wandering. The Adaptive coefficient a is the movement behaviour that balances exploration and exploitation. The global worst solution X_w and the deviation factor help the optimisation process to move out of bad regions in the search space.

3.2.2 Classification Accuracy Assessment on Brood Ball DBs

Brood ball dung beetles, for instance, exploit both the search space more broadly and individuals with potential within their patches. It uses this behaviour to improve candidate feature subsets close to local optimal solutions within the framework proposed. The adaptive search boundaries are updated as needed to stabilise the convergence process and achieve better classification performance.

$$\begin{cases} L'_{bb1} = \max(X1' * (1 - R1), L'_{b1}) \\ U'_{bb1} = \min(X1^1 * (1 + R1), U^1_{b1}) \end{cases} \quad (3)$$

$$\mathrm{R1} = 1 - \frac{t1}{T1_{max}} \quad (4)$$

$$x_{t1}(s+1) = x^{opt} + d_1 \cdot (x_{t1}(s) - L_{opt}) + d_2 \cdot (x_{t1}(s) - U_{opt}) \quad (5)$$

As iterations update the search range, it shrinks gradually in a way so that global exploration can be switched to local exploitation gradually. It works towards the refinement of feature subsets and boosts the discriminative power of selected features.

3.2.3 Performance Evaluation of Classification Models on Small DBs

Neighbourhood exploration and adaptive search diversity in small dung beetles. We model this behavior in the proposed MDBO framework, which is aimed at preserving population diversity and enhancing exploration across high-dimensional feature spaces.

$$\begin{cases} L_{bb1}^{1} = \max(X^{1} * (1 - R), L_{b1}^{1}) \\ U_{bb1}^{1} = \min(X^{1} * (1 + R), U_{b1}^{1}) \end{cases} \tag{6}$$

$$x_{t1}(s+1) = x_{t1}(s) + D_1 \cdot \left(x_{t1}(s) - L_{b1}^{reg}\right) + D_2 \cdot \left(x_{t1}(s) - U_{b1}^{reg}\right) \tag{7}$$

The adaptive upper and lower bounds will restrict the search space but still allow stochastic usage of neighbourhood direction based on potential solutions. Similarly, this strategy avoids early stopping and enhances the robustness to optimise features.

3.2.4 Exploitation of Thief DBs for Adversarial Attacks

There is also a thieving behaviour where the beetles will steal food balls from one another. They are scavenger races who live by engaging their opponents. The synthesis (movement) of thief DBs is formally expressed as:

$$x_{t1}(s+1) = X1' + S.g.(|x_{t1}(s) - X1^{best}| + |x_{t1}(s) - X1'|) \tag{8}$$

$$Z_{s+1} = \begin{cases} \dfrac{Z_s}{c}, & 0 < Z_s < c \\ \dfrac{1 - Z_s}{1 - c}, & c < Z_s < 1 \end{cases} \tag{9}$$

$$x_{t1}(s) = X'(U_b' - L_b') + L_b' \tag{10}$$

$$x_{start}(s) = \begin{cases} L_b', & \text{if } x_t(s) < L_b' \\ U_b', & \text{if } x_t(s) > U_b' \end{cases} \tag{11}$$

.

$$Score = \lambda \eta_Q(S) + \mu \frac{|\mathrm{Q}|}{|\mathrm{S}|} \tag{12}$$

The biologically inspired search behaviours are mathematically abstracted into adaptive optimisation operators for efficient feature selection in IoT intrusion detection. In addition, a tent chaotic mapping is added to MDBO to encourage more diverse populations at initialisation. Chaotic initialisation spreads candidate solutions more evenly across the feasible search space, which leads to higher optimisation efficiency and quality of convergence.

The score function assesses candidate feature subsets against a balance between classification results and the simplicity of the subset itself. This fitness formulation allows for the search to optimise predictive accuracy while penalising redundancy in feature selection to improve computational efficiency and model interpretability.

3.3 Employing ABiRNN for Enhanced Classification Accuracy

In this layer, the ABiRNN model is used to forecast the past incursions that have occurred. This method is appropriate for an intrusion detection system since it can keep the time dependency and the content information of the input. More importantly, different from typical RNNs, it makes the model have the ability to take both previous and future states into consideration. This way, the model has a better chance of learning the real correlations that we can see, which are indicative of an intrusion. By handling the attention type mechanism in the model, the model can pay more attention to a single feature in order to improve its accuracy and reduce the noise due to the irrelevant input. Because of this inherent fusion, the ABiRNN model can have an easy chance of finding the new potential fine-grained types of attacks which seem difficult for other schemes, including the similarity-based or the traditional RNN-based classifiers, to catch while training. Furthermore, the possibility to run this model in real time and to generalise across different kinds of traffic to a network setting, gives grounds to consider it as a robust candidate to deal dynamically with security settings. Overall, the ABiRNN model is a better choice in terms of intrusion detection for high-dimensional data and feature extraction capability. ABiRNN architecture consisting of forward and backward recurrent layers to capture temporal dependencies in IoT traffic data, whereas the attention mechanism highlights the most relevant features related to IS attacks, as shown in Fig. 4.

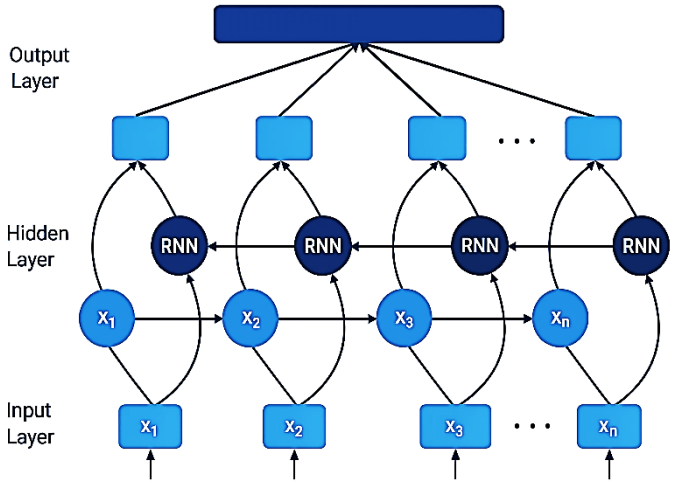

Fig. 4: Architecture of the BiRNN model structure.

$$h_s = F(Ux + Wh_{s-1} + \eta) \tag{13}$$

Equation 13. is the general formula of the recurrence relation for RNN layer. where h_s is the s^{th} hidden state, x is the input at time step s, U_x and W are the weight matrix of the input and of the previous hidden state, η is a bias. The nonlinearity activation function F(·) (e.g., tanh or ReLU) is then used for the weighted sum of these components.

$$\vec{h}_s = F\left(\vec{U}_x + \vec{W}\,\vec{h}_{s-1} + \vec{\eta}\right) \tag{14}$$

Equation 14. represents the forward hidden state update. The arrows $\overrightarrow{h_s}$ denote that the state is computed traversing the sequence from start to end. Parameters $\overrightarrow{U_x}$,$\overrightarrow{W}$ and $\overrightarrow{\eta}$ are specifically for the forward pass, to capture sequential context from past to future.

$$\overleftarrow{h}_s = \overleftarrow{F}\left(\overleftarrow{U}_x + \overleftarrow{w}\overleftarrow{h}_{s-1} + \overleftarrow{\eta}\right) \tag{15}$$

Equation 15. provides the reverse hidden state update, where $\overleftarrow{h_s}$ is calculated by reversing the sequence when: from end to start. Params $\overleftarrow{U_x}$,$\overleftarrow{w}$ and $\overleftarrow{\eta}$ areactor-specific and ensure that the network is extracting future-looking information at each time tick.

$$s_p = \left[\vec{s}_p^{\,T}\,;\,\overleftarrow{s}_p^{\,T}\,\right] \tag{16}$$

This equation 16. forms the hidden representation at position p of the sequence by concatenating the hidden states produced by the forward pass $\vec{s}_p^{\,T}$ and the backward pass $\overleftarrow{s}_p^{\,T}$ of the BiRNN. The consequence, s_p, mixes information from the past and future context inside the position p, allowing for a more comprehensive, position-specific feature representation. The superscript ⊤ means vector transposition, sorting the representations for the following operation.

$$u_p = \sum_{m=1}^{P} \beta_p^m s_m \tag{17}$$

Here, up is the context vector for position p generated by the attention mechanism. The sum runs over all sequence positions m with each hidden state s_m weighted by an attention coefficient β_p^m that captures the relative importance of each position's representation when combining information for downstream tasks (e.g., prediction or classification). The proposed attention-based aggregation can attend to essential structural patterns in the data selectively.

$$\beta_p^m = \frac{\exp(\hat{\beta}_p^m)}{\sum_{i=1}^{P_y} \exp(\hat{\beta}_p^i)} \tag{18}$$

The attention coefficient β_p^m is calculated using a softmax function which normalizes the raw score $\hat{\beta}_p^m$ for position m at p, such that the produced weights sum to one. This normalization improves that every attention score is viewed as a probability, and the mechanism can allocate the concentration of focus proportionally to all positions in the input. The denominator is an accumulator over all P_y sequence position scores while weighting, which stabilises the weighting between positions and improves that a single position cannot dominate unless strongly favoured by the scoring function $\hat{\beta}_p^i$.

3.3.1 ABiRNN Architecture and Training Configuration

The ABiRNN model includes bidirectional recurrent layers with an attention mechanism designed to capture temporal dependencies in both past and future directions of IoT traffic sequences. Bidirectional learning strategy enhances context feature extraction, and the attention mechanism highlights the critical intrusion-related patterns when classifying.

The ABiRNN network was trained for 100 epochs using the Adam optimiser with an initial learning rate of 0.001. Epochs and batch size nStay for 100 epochs, with a batch size of 64 to achieve stable convergent and learning performance. A dropout layer with a 0.5 dropout rate was used after two 2D-convolution layers, to reduce the overfitting and improve the generalisation capability of our model, as shown in Table 1.

Moreover, a learning rate scheduling strategy was applied to decrease the learning rate on each occasion where validation performance levelled off during training. The multi-class intrusion classification could be acquired from the categorical cross-entropy loss function. Moreover, the comprehensive training setup enhances both convergence stability and classification accuracy while also improving model robustness towards complicated IoT attack behaviours.

Table 1: ABiRNN Training Parameters

Parameter	Value
Optimizer	Adam
Initial Learning Rate	0.001
Batch Size	64
Epochs	100
Dropout Rate	0.5
Loss Function	Categorical Cross-Entropy
Learning Rate Scheduler	Reduce-on-Plateau
Classification Type	Multi-Class

3.4 Enhancing Model Performance through ARO-Guided Hyperparameter Tuning

Eventually, from the ARO algorithm, a hyperparameter search is performed from the best overall classification. Two reasons make the ARO model attractive as a hyperparameter tuning (model of behaviour cancer that was formulated based on physiology, where b design, one may make a comparison to the behaviour of rabbits when they are foraging for food. The method can explore the hyperparameter space well and get out of local optima, which are the problems that may be found when employing standard strategies such as grid or random search. This approach can accurately and efficiently search/exploit. In addition, ARO aggression is population-based and allows for the adaptability of learning to a broad spectrum of models and datasets as compared to crash-recovery. Also, being able to choose among multiple optimisation methods and to add some randomness helps the method to look for several solutions and hence to enhance the system's performance. In the hyperparameter search, ARO provides an advantageous way, particularly for the difficult machine learning problems, as evidenced by the enhancements in the model accuracy and rate of convergence. Fig. 5 shows the ARO hyperparameter and its overall structure. The figure describes how exploration and exploitation strategies are utilised to tune the learning parameters of the proposed deep learning architecture for better performance in intrusion detection.

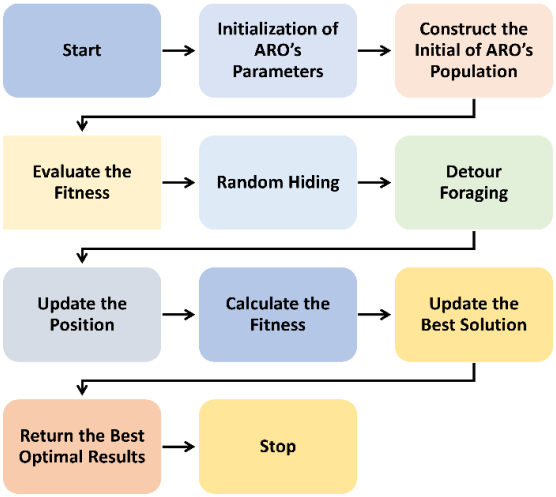

Fig. 5: Overview of the ARO Architecture through a Graphical Representation

In ARO, adaptive exploration and exploitation strategies are employed in order to search the hyperparameter space more efficiently. The three most dynamic update operators also improve convergence stability and allow effective optimisation of the ABiRNN learning parameters for intrusion detection. Table 2 illustrate Dataset composition across different traffic categories.

Table 2: Dataset Composition Across Different Traffic Categories.

Methods	Number of Samples
Blackhole	11,059
Flooding	3,414
Scheduling Attacks	6,739
Gray hole	13,656
Normal	350,068
Total Number of Samples	384,936

There is another aspect we should consider, namely that the rabbit is losing strength and has to de-automatise from the detour-picking and focus on the changeover, which we could have termed random hiding. The seeking agent is not the only actor capable of dragging occupied locations along the diversion to random individuals to increase the extent of the induced damage, as can be seen in the case of actively searching along the diversion. Here you will find a form of a model that provides a formal description of the detour foraging (and that has been recommended): The update of the agent k at the step s+1 is defined as:

$$\vec{U}_k(s+1) = \vec{y}_q(s) + P \cdot \left(\vec{\psi}(s) - \vec{y}_k(s)\right) + \text{round}(0.025 + 0.5q1) \cdot m1 \qquad (19)$$

Where, $p,q=1,\ldots,M$ integrates several sources of randomness and directional shifts. The position $\vec{y}_q(s)$ is a reference from agent q, while P scales the exploration based on the vector difference between a population attractor $\vec{\psi}(s)$ and the current state $\vec{y}_k(s)$. The rounding term, controlled by the random variable q1, introduces integer-valued stochastic jumps, further multiplied by m1 to vary step size. Agents p,q vary from 1 to M, indicating all possible pairs in the population.

Where, p,q=1,…,M encompasses multiple sources of randomness and directionality. The position. $\vec{y}_q(s)$ is a reference term from agent q and P is scaling the exploration by the vector difference of a population attractor $\vec{\psi}(s)$ and the current state $\vec{y}_k(s)$. The rounding term introduces integer-valued stochastic jumps, further scaled by m1 to vary step size. Agents p,q belong to 1 to M agents and represent any possible pair in the population.
The exploration factor P is computed as

$$P = H \cdot d \qquad (20)$$

where H encodes time-dependent cyclic variation:

$$H = \left(f - f^{\left(\frac{s-1}{S}\right)^2}\right) \cdot \sin(2\pi q_2) \tag{21}$$

Herein, f is the scaling factor of magnitude, S is the maximum iterations, ss is the present iteration, and q2 is a random uniform number. This equation suppresses H with an exponentially decreasing term and a sinusoid term, so that the search pressure fluctuates and changes gradually from exploration to exploitation. Random selection operations are modelled by the indexing function,

$$d(u) = \begin{cases} 1 & \text{if } u == h(v) \\ 0 & \text{else} \end{cases} \tag{22}$$

where $h(v)$ is a randomised mapping, typically drawn from a random permutation,

$$h = \text{randperm}(Q) \tag{23}$$

over Q elements. This function specifies binary selection or activation, ensuring the chosen indices for update or activity are diversified at each step.

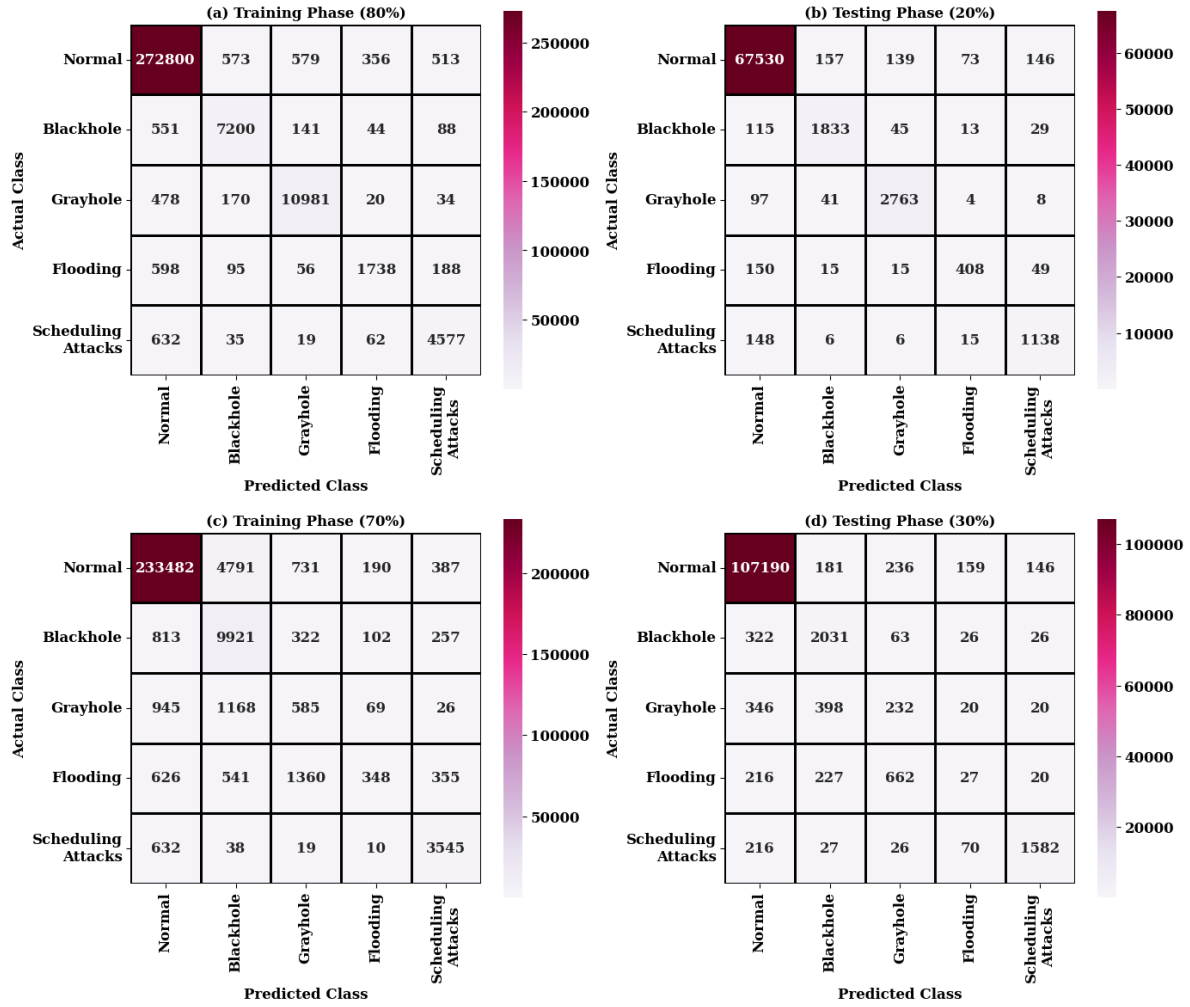

Fig. 6: The confusion matrices.

Algorithm 2: ARO-Based Hyperparameter Optimisation
Input: Selected feature vectors and ABiRNN parameters
Output: Optimised hyperparameters

Step 1: Initialise the rabbit population with random parameter values
Step 2: Evaluate fitness using validation accuracy
Step 3: Perform exploration search phase
Step 4: Perform the exploitation refinement phase
Step 5: Update rabbit positions dynamically
Step 6: Select the best global solution
Step 7: Repeat until the stopping criterion is satisfied

Table 3: Results of the intrusion detection using an 80:20 split of TRAINING PHASE and TESTING PHASE

Metric (80% / 20%)	Normal value	Blackhole value	Grayhole value	Flooding value	Scheduling Attacks value	Average value
Accuracy Value	98.56 / 98.62	99.33 / 99.44	99.51 / 99.52	99.52 / 99.54	99.47 / 99.46	99.20 / 99.32
Sensitivity Value	99.25 / 99.23	89.72 / 90.51	93.98 / 94.84	64.96 / 64.06	85.94 / 86.64	86.77 / 87.06
Specificity Value	91.84 / 92.50	99.61 / 99.60	99.71 / 99.71	99.82 / 99.85	99.70 / 99.61	98.16 / 98.31
F-Score Value	99.20 / 99.24	89.45 / 89.91	93.61 / 93.95	71.00 / 70.95	85.34 / 85.14	87.72 / 87.85
AUC Score Value	95.54 / 95.91	94.71 / 95.10	96.85 / 97.27	82.41 / 81.94	92.82 / 93.18	92.46 / 92.68

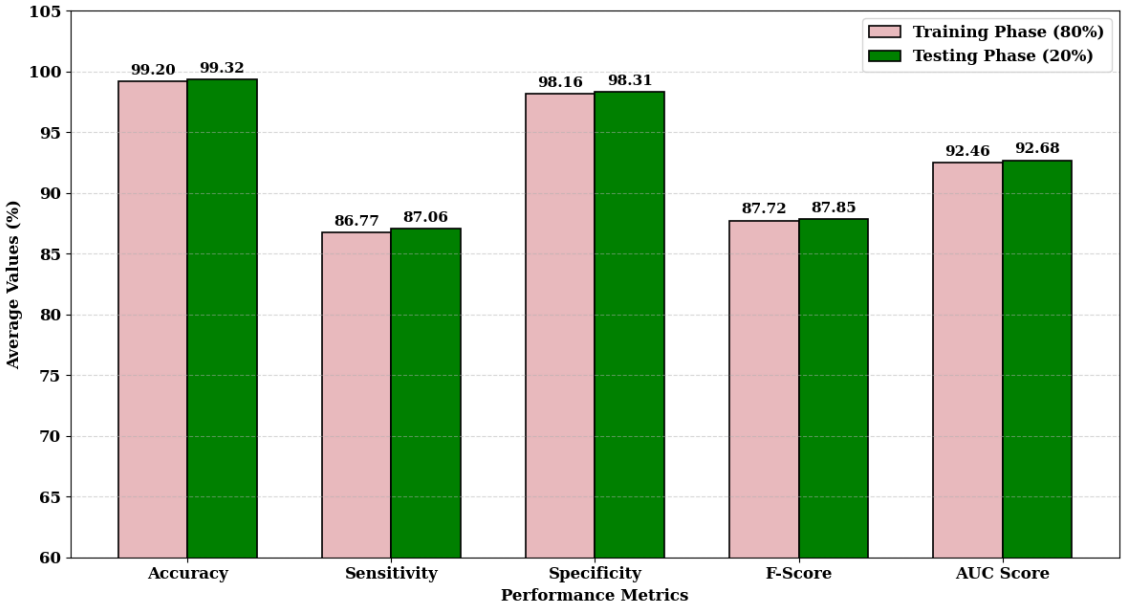

Fig. 7: Performance comparison of the proposed MDBODR-DLCS framework under the 80:20 training–testing split using multiple evaluation metrics, including accuracy, sensitivity, specificity, F-score, and AUC score.

Table 4: Results of intrusion detection using a 70:30 split of TRAINING PHASE and TESTING PHASE.

Metric (70% / 30%)	Normal Value	Blackhole Value	Grayhole Value	Flooding Value	Scheduling Attacks Value	Average Value
Accuracy Value	98.15 / 98.16	99.34 / 99.37	99.34 / 99.32	98.56 / 99.56	99.36 / 99.34	99.15 / 99.15
Sensitivity Value	99.31 / 99.23	84.91 / 85.67	90.00 / 89.62	67.36 / 65.19	77.14 / 77.67	83.75 / 83.58
Specificity Value	86.65 / 87.06	99.76 / 99.75	99.71 / 99.61	99.84 / 99.83	99.76 / 99.73	97.14 / 97.21
F-Score Value	98.98 / 98.97	87.80 / 88.21	91.45 / 90.26	72.96 / 72.05	81.22 / 80.81	86.47 / 86.20
AUC Score Value	92.98 / 93.17	92.32 / 92.66	94.86 / 94.72	83.62 / 82.86	88.47 / 88.52	90.46 / 90.40

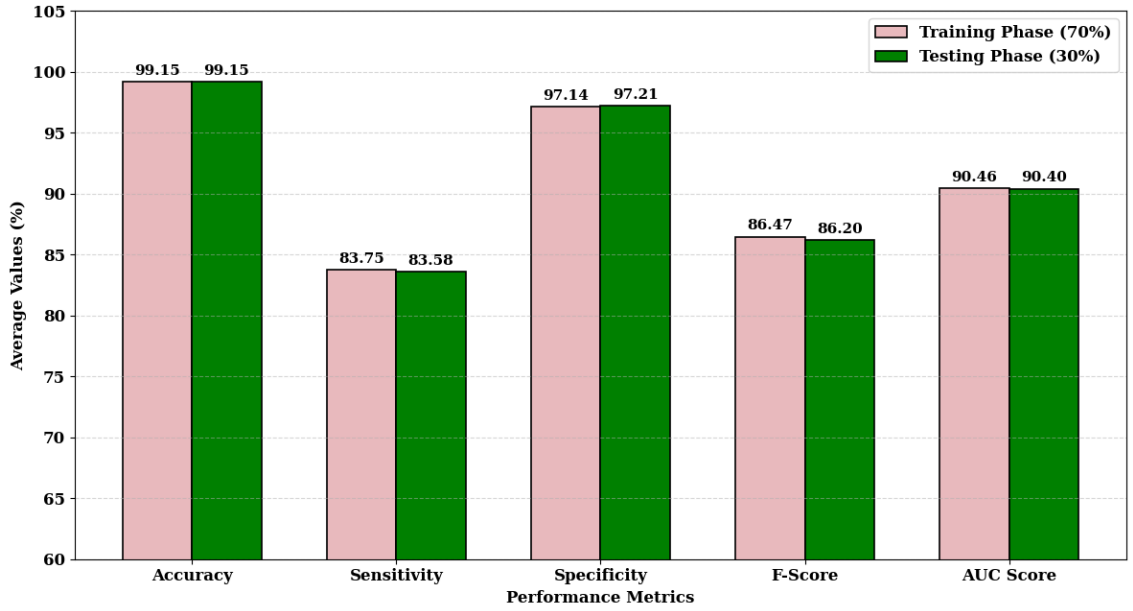

Fig. 8: Performance comparison of the proposed MDBODR-DLCS framework under the 70:30 training–testing split using multiple evaluation metrics, including accuracy, sensitivity, specificity, F-score, and AUC score.

As presented in Tables 3 and 4, along with Figs. 6–8 illustrate that the proposed MDBODR-DLCS framework provides consistently superior intrusion detection accuracy within both the 80:20 and 70:30 train–test split groups. The model achieved accuracy values over 99% on average, and with high performance for specificity and the F-score in different types of attacks. In contrast, the overall results verify that the proposed framework presents a robust and effective IoT intrusion detection approach, despite lower sensitivity to Flooding attacks from overlapping traffic characteristics and cross-class imbalance.

$$\vec{a}_{pq}(s) = \vec{Y}_p(s) + Q \cdot h \cdot \vec{Y}_p(s), q = 1, \dots, Q \quad (24)$$

Equation 24. creates a new candidate vector of the agent p at the q index, where the perturbation is multiplied by the Q, which is a time variant factor (a dynamic factor that depends on the population context and the number of iterations) and h, a random scale factor. It improves that the agents receive personalised, stochastic updates to better explore the search space.

$$Q = \frac{S-s+1}{S} q_5 \quad (25)$$

Equation 25. computes the global dynamic scaling factor Q that varies with optimisation. It squashes the range between 1 and 0, and multiplies the result by q_5, a random variable forcing magnitude, decreasing or as a set of iterations to control the convergence of the solution.

$$\ell(u) = \begin{cases} 1 & \text{if } u == v \\ 0 & \text{else} \end{cases} \quad v = 1, \dots, Q \tag{26}$$

$\ell(u)$=1 if selection matches index; 0, otherwise.

These equations 26. randomly choose activations between 0 and 1, choosing which ones are turned on/up or mutated over random mappings such as permutations.

$$\vec{Z}_k(s+1) = \vec{y}_k(s) + P \cdot \left(q_5 \cdot \vec{a}_{pq}(s) - \vec{y}_k(s)\right) \tag{27}$$

In this update step the updated candidate $\vec{Z}_k(s+1)$ is generated by moving $\vec{y}_k(s)$ towards the perturbed vector $\vec{a}_{pq}(s)$ with a step size being controlled by the parameters P and q_5. This is an attempt to implement a competitive, random exploration of the search space.

$$\ell(u) = \begin{cases} 1 & \text{if } u == [q_6 \cdot Q] \\ 0 & \text{else} \end{cases} \quad u = 1, \dots, Q \tag{28}$$

$$\vec{a}_{pq}(s) = \vec{y}_p(s) + Q \cdot h_p \cdot \vec{y}_p(s) \tag{29}$$

$$\vec{y}_k(s+1) = \begin{cases} \vec{y}_k(s) & h\left(\vec{y}_k(s)\right) \le h\left(\vec{Z}_k(s+1)\right) \\ \vec{Z}_k(s+1) & h\left(\vec{y}_k(s)\right) > h\left(\vec{Z}_k(s+1)\right) \end{cases} \tag{30}$$

$$B(s) = 4\left(1 - \frac{s}{S}\right)\ln\frac{1}{q} \tag{31}$$

This parameter governs the switch from wide to deep exploration. As s approaches the maximum iteration S, the scaling will decay, and local search and refinement solutions will be more preferred.

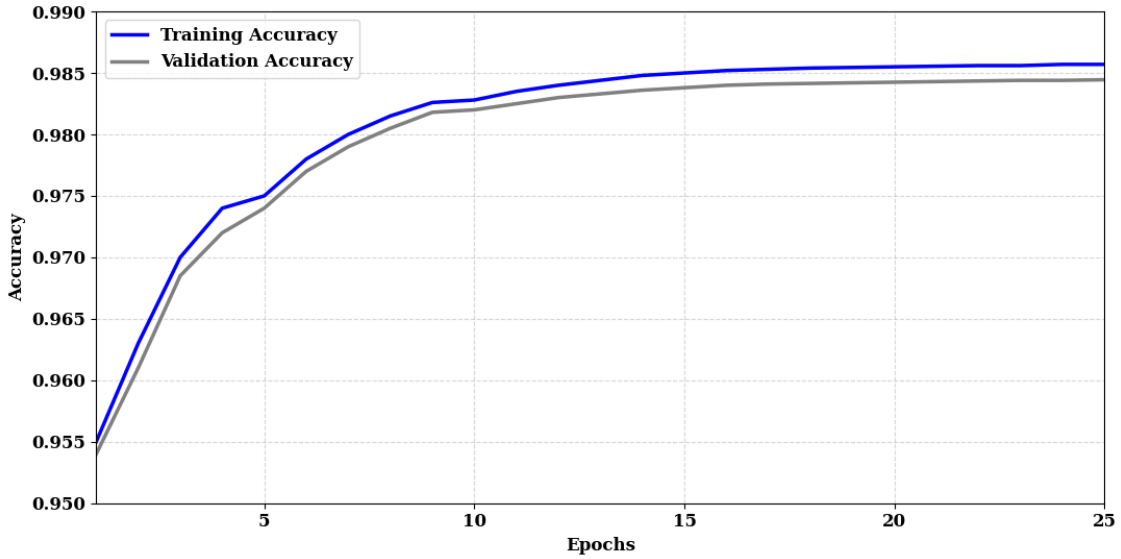

Fig. 9: Average training and validation accuracy curves of the proposed MDBODR-DLCS framework under the 70:30 training–testing split across different training epochs.

$$score(y_k) = ErrorRate(y_k) = \frac{\text{number of incorrect classifications}}{\text{Total cases}} \times 100 \qquad (32)$$

This ultimate criterion also gives the classification error rate to ease comparison of solutions for a selection of the most accurate ones. Fig.9 Training and validation accuracy curves of the proposed MDBODR-DLCS framework over a 70:30 train–test split, showing stable convergence and consistent learning behaviour across training epochs.

3.4.1 MDBO and ARO Parameter Configuration

Both the MDBO feature selection procedure and the ARO-based hyperparameter optimisation strategy are crucial to the performance of the proposed MDBODR-DLCS framework. In the previously defined MDBO algorithm, we initialised each search process with different candidate feature subsets for a balanced exploitation-oriented optimization. The search space of the feature selection used is all the normalised attributes of network traffic in the WSNDS dataset.

In the same vein, critical ABiRNN hyperparameters, such as learning rate, hidden neuron count, dropout rate and batch size were also optimised using the ARO algorithm. The lower and upper bounds were defined for the hyperparameter search space to guarantee stable convergence and effective training time performance. The optimisation algorithms were performed iteratively until the stopping criteria or maximum iteration limits were fulfilled. Tables 5 and 6 show the parameter configurations and search spaces used for MDBO feature selection and ARO-based hyperparameter optimisation processes. The previously presented parameter settings allow a suitable balance between exploration and exploitation in the optimisation process, while the predefined hyperparameters modified the range of ABiRNN, allowing stable convergence, efficient learning and a higher quality classifier regarding the intrusion detection performance.

Table 5: MDBO and ARO Parameter Settings

Parameter	MDBO	ARO
Population Size	30	30
Maximum Iterations	100	100
Search Dimension	Feature Count	Hyperparameter Count
Exploration Factor	0.5	0.6
Exploitation Factor	0.7	0.8
Fitness Function	Classification Accuracy	Validation Accuracy
Search Space	Feature Subsets	Hyperparameter Space
Stopping Criterion	Maximum Iterations	Maximum Iterations

Table 6: Hyperparameter Search Space for ABiRNN

Hyperparameter	Search Range
Learning Rate	0.0001 – 0.01
Hidden Units	32 – 256
Batch Size	32 – 128
Dropout Rate	0.1 – 0.5
Number of Epochs	50 – 150

4.0 RESULT AND DISCUSSION

The experiments were conducted using stratified train–test splitting to preserve the original class distribution across training and testing subsets. The performance of the proposed Mixed Dung Beetle Optimisation Dimensionality reduction, along with the Deep Learning for Cyber Security (MDBODR-DLCS) framework, is evaluated with the popular WSNDS dataset that is employed in the intrusion detection field. The dataset is depicted in Table 2 and includes 384,936 traffic samples belonging to five categories: Normal, Blackhole, Grayhole, Flooding, and Scheduling Attacks. An important issue of this dataset is the class imbalance; the Normal class occupies a large portion (350,068) of the 20M samples, and the minority attacks (Flooding and Scheduling) have 3,414 and 6,739 samples, respectively. This uneven distribution is representative of IoT traffic distributions

observed in practice and renders the dataset a good candidate for testing IDS robustness, as models must have the capacity to make highly accurate predictions for the majority classes without bias.

To measure the performance of the MDBODR-DLCS feature set, we evaluated it with two different train sets. The confusion matrices are shown in Fig 6, from which one can see the effectiveness of our method in distinguishing benign and attack samples. From the 80:20 split, Table 3 displays the classification outcomes for the prototypical with an average accuracy of 99.2%. The sensitivity is 99.25% for Normal traffic, 93.98% for Grayhole traffic, and 64.96% for Flooding attacks detected in the most difficult way. An average F-score of 87.72% and AUC of 92.46% across the 143 runs shows good general performance of the model. In the cutoff value was used in the 70:30 ratio, Table 4 summarises the evidence again at a 70:30 ratio, the performance parameter is maintained almost the same: the mean of accuracy is 99.15%, sensitivity is 83.75%, specificity is above 97%, and AUC is around 90.5%. These findings reveal that the proposed method generalises well from various training–testing splits and is not too sensitive to the ratio of training data.

Table 7: Performance Comparison Between Different Approaches

Metrics Value	**MDBODR-DLCS VALUE**	**RKOA-AEID VALUE**	**AdaBoost VALUE**	**GB VALUE**	**XG-Boost VALUE**	**KNN-AOA VALUE**	**KNN-PSO VALUE**
Accuracy_Value	99.34	98.96	95.70	94.60	96.83	97.21	92.90
Sensitivity_Value	87.06	75.35	69.24	71.04	71.52	70.17	71.32
Specificity_Value	98.32	96.46	95.01	94.11	94.46	96.05	95.08
F-Score_Value	87.84	79.55	76.18	71.96	71.02	73.89	70.51

The performance comparison of our MDBODR-DLCS is demonstrated in Figs .7 to 9 through performance metrics on two splits, and it shows that MDBODR-DLCS sustains the higher accuracy, sensitivity, specificity and F-score even with the unbalanced dataset. In all partitions, accuracy was above 99%, and specificity was between 97% and 98%, demonstrating the model's great performance in detecting patients without diseases. This is of special interest in IoT security cases since too many false alarms may cause abnormal operations. Sensitivity values, being lower than those of accuracy, illustrate the system's ability to detect minority attacks. The AUC 90–93% supports the robust discriminative capacity of the model, as shown in Table 7.

In addition to aggregate statistics, precision–recall and receiver-operating characteristic (ROC) analyses lend further insights as to the behavior of the model. We observe that the PR curve in Fig. 10 demonstrates the model achieving high precision across all classes, faring well-above the baseline random classifier with 0.5 PR points. This means that MDBODR-DLCS can detect attack traffic without significantly decreasing recall, which is crucial to intrusion detection, where the costs of false alarms and missed detections are both large. In the same vein, the ROC curve in Fig. 11 reflects that the TPR and FPR have a perfect trade-off. The curve is continually high in the top-left corner of the plot, indicating that the classifier is highly discriminative at all thresholds.

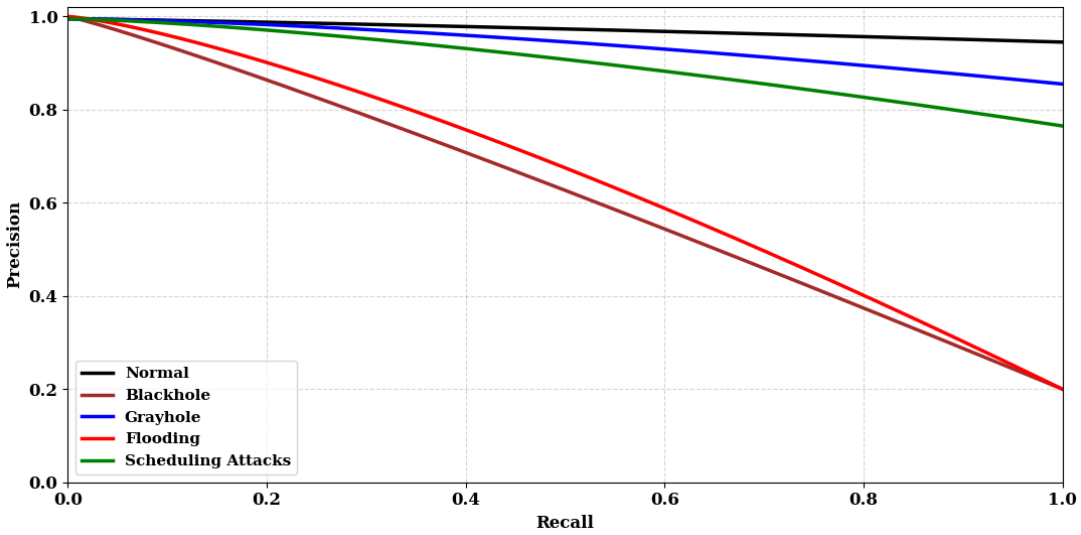

Fig. 10: Precision–Recall curve analysis of the proposed MDBODR-DLCS framework for different attack classes under the 80:20 training–testing split.

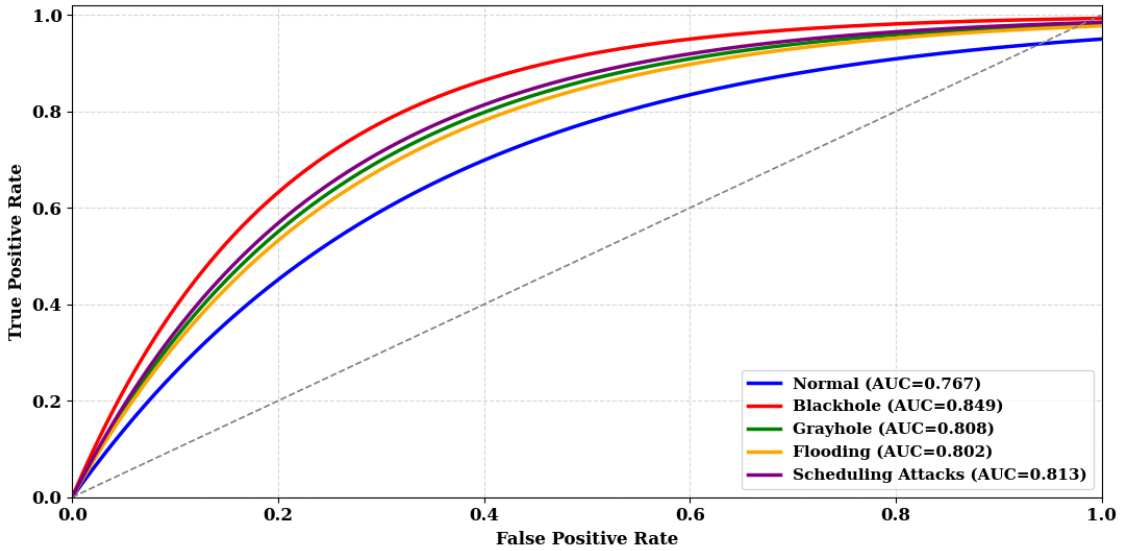

Fig. 11: ROC curve analysis of the proposed MDBODR-DLCS framework for different attack classes under the 80:20 training–testing split.

Comparison between the projected outline and the up-to-date IDS representations also demonstrates the superiority of the proposed framework. Values of the performance of MDBODR-DLCS and the comparative image in Fig. 12 (the table and the comparative image in the following text). Values of the performance of MDBODR-DLCS compared with RKOA-AEID, AdaBoost, Gradient Boosting, XGBoost, KNN-AOA, and KNN-PSO. MDBODR-DLCS surpasses all the baselines, for the highest accuracy (99.34%), F-score (87.84%) and better sensitivity (87.06%). In comparison, RKOA-AEID performed moderately strongly, with a sensitivity of 75.35% and an F-score of 79.55%, whilst traditional machine learning (AdaBoost and Gradient Boosting) had an accuracy below 96% and a sensitivity of around 70%. The comparative findings verify that the combination of Mixed Dung Beetle Optimisation for feature selection, attention-based Bidirectional Recurrent Neural Network for temporal pattern learning, and ARO for hyperparameter optimisation is significantly better than other state-of-the-art approaches.

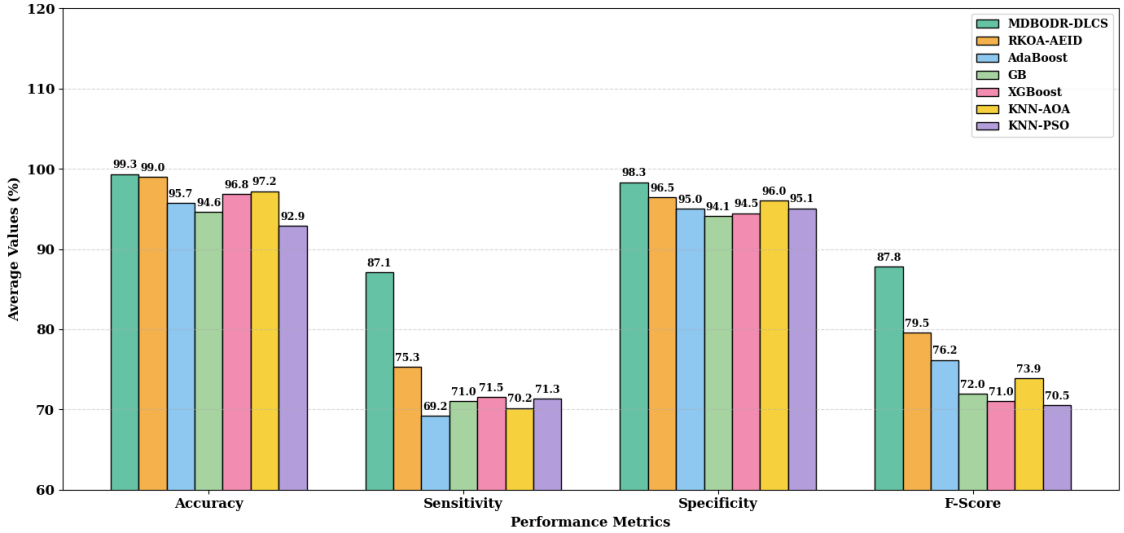

Fig. 12: Comparative performance evaluation of the proposed MDBODR-DLCS framework against existing baseline methods using accuracy, sensitivity, specificity, and F-score metrics.

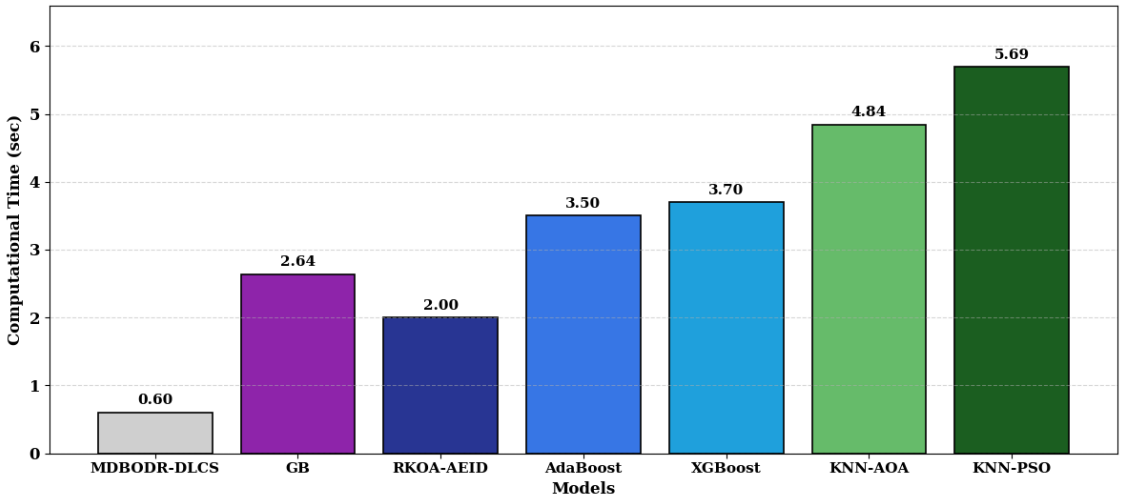

Fig. 13: Comparative computational time analysis of the proposed MDBODR-DLCS framework and existing baseline methods.

Computational efficiency is as important as classification accuracy, especially in IoT, where intrusion detection should be done near real-time. Time consumption analysis. The computation time (CT) analysis of the planned model is shown in the Table 8, and Fig. 13, it is evident that the proposed model is lightweight.

Table 8: Computational Time (CT) Chart Compared with Various Methods.

Methods	Computational Time (seconds)
MDBODR DLCS_Value	0.60
RKOA-AEID_Value	2.00
AdaBoost_Value	3.50
GB_Value	2.64
XGBoost_Value	3.70
KNN-AOA_Value	4.84
KNN-PSO_Value	5.69

MDBODR-DLCS takes an average time of 0.60 seconds on classification and outperforms all the baselines. For example, RKOA-AEID 2.00 seconds, AdaBoost 3.50 seconds, and KNN-PSO 5.69 seconds. This is a ∼ 3–10× reduction in runtime vs competing methods, demonstrating that the proposed framework is well-suited for real-time IoT applications and situations in which speed and responsiveness are important.

The summary of both Figs .6 to 13 and Tables 2–5 shows that the proposed MDBODR-DLCS is effective. The proposed framework was not only consistently accurate (>99%) but was also sensitive across diverse attack classes, proving robust to class imbalance. Precision, recall, and ROC analyses further verified the discriminative power of the model, and comparative experiments demonstrated that the proposed model outperformed the existing methods in terms of accuracy and sensitivity. Finally, the time difficulty examination of the planned work demonstrated that it is appropriate to be used in real-time resource-constrained IoT environments. Together, these observations validate MDBODR-DLCS as an effective, scalable and efficient intrusion detection framework for future IoT networks.

4.1 Experimental Setup and Runtime Environment

The implementation of the proposed MDBODR-DLCS framework was accomplished with Python, TensorFlow, and Keras deep learning libraries. All experiments were performed on a workstation with Intel Core i7, 16 GB RAM and NVIDIA GPU acceleration in the Windows environment. Since the MDBO feature selection and ARO hyperparameter optimisation procedures were combined into the ABiRNN classification framework, it was utilised to assess intrusion detection performance on the WSNDS dataset. The runtime performance reported in this study refers to the average execution times from different runs of the experiments, carried out under exactly the same computational settings. We trained and tested using stratified train–test splitting strategies that help

preserve the class distribution inherent in the data. The steps of the proposed framework are preprocessing, feature selection, classification and optimisation.

4.2 Statistical Validation and Robustness Analysis

In order to assess the performance stability of the whole MDBODR-DLCS framework, all experiments were repeated several times using the same experimental conditions and the mean performances were recorded. Reported results correspond to mean classification performance recorded over repeated experimental runs, whereas standard deviation values were used for measuring how consistent the performance was across different experimental conditions and model stability.

Moreover, statistical significance analysis was conducted to evaluate the superiority of the proposed MDBODR-DLCS framework compared to existing state-of-the-art approaches. The classification accuracy values were obtained over repeated runs, and a paired t-test was conducted on them. The framework validation through statistical analysis revealed that the proposed method presents notably better performance than traditional intrusion detection methods with p-values lower than 0.05, confirming the reliability and effectiveness of the optimisation and classification strategy proposed in this study, as shown in Table 9. Each experiment was repeated ten times to reduce randomness and improve evaluation reliability.

Table 9: Statistical Performance Analysis Across Multiple Runs

Method	Accuracy (%)	Standard Deviation	p-value
MDBODR-DLCS	99.32	±0.12	<0.05
RKOA-AEID	98.96	±0.25	<0.05
AdaBoost	95.70	±0.41	<0.05
GB	94.60	±0.46	<0.05
XGBoost	96.83	±0.37	<0.05
KNN-AOA	97.21	±0.31	<0.05
KNN-PSO	92.90	±0.54	<0.05

4.3 Ablation Study of MDBODR-DLCS Components

An ablation study is performed to investigate the individual components of MDBODR-DLCS. Various forms of the proposed model were implemented, including the removal of MDBO-based feature selection, attention mechanism and ARO-based hyperparameter optimisation. The content of the performance comparison shows how each module supports improving the results with respect to intrusion detection effectiveness.

The experimental results show that eliminating the MDBO feature selection stage will result in a higher representation of redundant features, which reduces the classification accuracy. In the same way, without the attention mechanism that enhances an ABiRNN model's memory of intrusion-related patterns, leads to a deterioration of its attention enablement capacity, which negatively affects both sensitivity and F-score performance. Also, full elimination of the ARO-based hyperparameter optimisation stage significantly harms convergence stability and efficiency in classification.

Overall, on the complete MDBODR-DLCS framework, it consistently outperformed all reduced variants, which further verifies how multimedia data in the form of time series signals can be fully exploited using optimal direction and attention-based learning within the proposed architecture for intrusion detection. The ablation analysis confirms that each optimisation and learning component contributes significantly to the overall performance improvement of the proposed framework, as shown in Table 10.

Table 10: Ablation Study of Proposed MDBODR-DLCS Framework

Model Variant	Accuracy (%)	Sensitivity (%)	Specificity (%)	F-Score (%)
Full MDBODR-DLCS	99.32	87.06	98.31	87.85
Without MDBO	97.84	81.25	96.12	82.44
Without Attention Mechanism	98.15	82.61	96.84	83.17
Without ARO	98.42	83.94	97.21	84.30
ABiRNN Only	96.75	78.12	94.63	79.08

4.4 Per-Class Performance Analysis

As a more comprehensive assessment of the proposed MDBODR-DLCS framework, per-class analyses on precision, recall and F1-score for all intrusion types identified in the WSNDS dataset were conducted. The results show consistently high classification performance of the proposed framework regarding Normal, Blackhole and Grayhole traffic classes as they have relatively distinct traffic characteristics with enough training representation.

Flooding and Scheduling Attack classes with relatively lower sensitivity values. This decrease in performance is primarily caused by class imbalance, overlapping distributions of features, and the fact that attack traffic patterns somewhat closely resemble normal communication behaviour. Flooding attacks in particular have extremely uneven traffic fluctuations, making them more complex to classify and reducing minority class separability.

Even with these challenges, the detection capability of minority attack categories based on optimised features and temporal learning by attention within MDBODR-DLCS is preserved well. Improving the ability to spotlight distinguishing intrusion patterns in the attention mechanism and feature optimisation stages enhances the relevance of features and robustness of classification across adversarially challenging attack classes. The per-class evaluation confirms that the proposed framework achieves balanced intrusion detection performance while effectively handling challenging minority attack categories, as shown in Table 11.

Table 11: Per-Class Precision, Recall, and F1-Score Analysis

Class	Precision (%)	Recall (%)	F1-Score (%)
Normal	99.24	99.23	99.24
Blackhole	89.91	90.51	89.91
Grayhole	93.95	94.84	93.95
Flooding	70.95	64.06	70.95
Scheduling Attacks	85.14	86.64	85.14

5.0 CONCLUSION AND FUTURE WORK

We proposed an MDBODR DLCS (Mixed Dung Beetle Optimisation Dimensionality Reduction with the Deep Learning Cyber Security) framework in this work to deal with intrusion detection efficiently in IoT networks. The framework integrated Z-score normalisation as preprocessing with Mixed Dung Beetle Optimisation for feature selection, Attention-based Bidirectional Recurrent Neural Network for classification, and Artificial Rabbits Optimisation for hyperparameter tuning, with the aim of providing a solution that can balance between accuracy and efficiency, particularly in the context of large-scale IoT. A detailed test conducted on the WSNDS dataset verified that the proposed method consistently delivered the classification accuracy of >99% with high sensitivity and specificity for various attack classes, including minority categories such as Flooding and Scheduling. The framework also showed high discriminative performance in precision–recall and ROC curve analysis, outperforming previous algorithms with respect to accuracy, F-score, and sensitivity. Especially, the time of 0.60 s. demonstrates that the proposed method is applicable for real-time application of intrusion detection in the restricted resources IoT environment.

In general, MDBODR DLCS may be a discernible, scalable, and efficient IDS scheme that balances detection performance and computational simplicity. With a good capability in dealing with class imbalance, robustness to different data partitions and superior performance than traditional and hybrid intrusion detection methods, it is a promising tool for a real-life IoT security task. The resulting framework thus provides an important stepping stone toward next-generation IoT security solutions for which high detection accuracy and low computational overhead must go hand-in-hand to secure modern IoT environments with hundreds of thousands of devices.

5.1 Limitations and Future Scope

While the new MDBODR-DLCS framework shows impressive performance, it still has some drawbacks. In the case of very imbalanced traffic distributions and quickly evolving zero-day attack scenarios that are not suitably well represented at training, it still may underperform. Moreover, computational complexity during computing metaheuristic optimisation and training of deep learning parameters in streaming data may prolong execution time for very large-scale real-time IoT environments.

Moreover, different attack classes that behave within similar traffic characteristics, for example, Flooding and Scheduling Attacks, are likely to have lower sensitivity in comparison with each other due to the similarity of their behaviour compared to normal operation. Also, the efficiency of the framework may vary due to the diversity and quality of data, as well, which is used for building the model.

Our future work will concern lightweight optimisation techniques, real-time distributed intrusion detection deployment and adaptive iterative learning mechanisms to ensure scalability and camouflage of learned knowledge while being robust against new unseen cyber-attacks in DIoT environments. The work done is a basic framework to prove the concept, and it can be further extended to generalise more using cross-dataset validation and real-time streaming traffic environments.

Declarations of Interest:
This is to certify that all authors have seen and approved the manuscript being submitted & have no conflict of interest. We would like to submit the paper entitled "Cyber Security Improvement in Networks of Internet of Things using Deep Learning and Mixed Feature Selection Mean Dung Beetle Optimisation" for possible evaluation in the Ionic Journal. We affirm that the manuscript has been prepared according to the Journal's instructions and that the content of the manuscript has not been published in any refereed journal.

Author Contributions: P. Devabalan and S Meenakshisundaram: Conceptualisation of the work, Writing – Original draft, Writing – review and editing. T.Sangeetha and L.Rajesh : Writing review and editing of the paper, Supervision. All authors have read and agreed to the published version of the manuscript.

Acknowledgements: The authors would like to thank SRM Institute of Science and Technology, Kattankulathur, India, for providing the necessary facilities to complete the project.

Funding: There is no need to report any funding for this project.

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

Institutional Review Board Statement: Not applicable.

REFERENCES

[1] H. Alkahtani and T. H. H. Aldhyani, "Intrusion detection system to advance Internet of Things infrastructure-based deep learning algorithms," *Complexity*, vol. 2021, pp. 1–18, 2021.

[2] Z. Ali Abbood, D. Çağdaş Atilla, and Ç. Aydin, "Intrusion detection system through deep learning in routing MANET networks," *Intelligent Automation & Soft Computing*, vol. 37, no. 1, pp. 269–281, 2023.

[3] S. S. Alotaibi, A. Sayed, E. S. Abd Elhameed, O. Alghushairy, M. Assiri, and S. S. Ibrahim, "Enhancing security in IoT-assisted UAV networks using adaptive mongoose optimization algorithm with deep learning," *IEEE Access*, vol. 12, pp. 63768–63776, 2024.

[4] R. Aluvalu, V. N. S. Kumaran, M. Thirumalaisamy, S. Basheer, E. A. Aldhahri, and S. Selvarajan, "Efficient data transmission on wireless communication through a privacy-enhanced blockchain process," *PeerJ Computer Science*, vol. 9, p. e1308, 2023.

[5] F. Y. Assiri and M. Ragab, "Optimal deep-learning-based cyberattack detection in a blockchain-assisted IoT environment," *Mathematics*, vol. 11, no. 19, p. 4080, 2023.

[6] H. Benmeziane, *Comparison of Deep Learning Frameworks and Compilers*, Master's thesis, École Nationale Supérieure d'Informatique, Oued Smar, Algeria, 2022.

[7] K. N. Dattatraya and K. R. Rao, "Hybrid-based cluster head selection for maximizing network lifetime and energy efficiency in WSN," *Journal of King Saud University – Computer and Information Sciences*, vol. 31, no. 4, pp. 504–511, 2019.

[8] R. Fang, T. Zhou, B. Yu, Z. Li, L. Ma, and Y. Zhang, "Dung beetle optimization algorithm based on improved multi-strategy fusion," *Electronics*, vol. 14, no. 1, p. 197, 2025.

[9] Y. Fu, Y. Du, Z. Cao, et al., "A deep learning model for network intrusion detection with imbalanced data," *Electronics*, vol. 11, no. 6, p. 898, 2022.

[10] Z. Gui, Y. Sun, L. Yang, et al., "LSI-LSTM: An attention-aware LSTM for real-time driving destination prediction by considering location semantics and location importance of trajectory points," *Neurocomputing*, vol. 440, pp. 72–88, 2021.

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

[12] A. Halbouni, T. S. Gunawan, M. H. Habaebi, et al., "Machine learning and deep learning approaches for cybersecurity: A review," *IEEE Access*, vol. 10, pp. 19572–19585, 2022.

[13] K. Hussain, S. J. Hussain, N. Jhanjhi, et al., "SYN flood attack detection based on Bayes estimator (SFADBE) for MANET," in *Proc. 2019 Int. Conf. Computer and Information Sciences (ICCIS)*, Apr. 2019.

[14] M. A. Jabbar, R. Aluvalu, S. S. S. Reddy, "RFAODE: A novel ensemble intrusion detection system," *Procedia Computer Science*, vol. 115, pp. 226–234, 2017.

[15] K. Jin, L. Zhang, Y. Zhang, D. Sun, and X. Zheng, "A network traffic intrusion detection method for industrial control systems based on deep learning," *Electronics*, vol. 12, no. 20, p. 4329, 2023.

[16] M. P. Kantipudi, R. Aluvalu, and S. Velamuri, "An intelligent approach of intrusion detection in mobile crowd sourcing systems in the context of IoT-based smart city," *Smart Science*, vol. 11, no. 1, pp. 234–240, 2022.

[17] S. Laqtib, K. E. Yassini, and M. L. Hasnaoui, "A deep learning method for intrusion detection systems based on machine learning in MANET," in *Proc. 4th Int. Conf. Smart City Applications (SCA 2019)*, Oct. 2019.

[18] Y. Li, J. Zhang, Y. Yan, Y. Lei, and C. Yin, "Enhancing network intrusion detection through the application of the dung beetle optimized fusion model," *IEEE Access*, vol. 12, pp. 9483–9496, 2024.

[19] L. A. Maghrabi, et al., "Enhancing cybersecurity in the Internet of Things environment using bald eagle search optimization with hybrid deep learning," *IEEE Access*, vol. 12, pp. 8337–8345, 2024.

[20] D. Mahesh and T. S. Kumar, "OptCosineDCNN: An advanced network attack detection and mitigation in SDN environments utilizing enhanced deep learning-based approaches," *Cybernetics and Systems*, pp. 1–43, 2025.

[21] R. Meddeb, F. Jemili, B. Triki, et al., "A deep learning based intrusion detection approach for MANET," *Research Square*, Aug. 2022.

[22] S. B. Ninu, "An intrusion detection system using exponential Henry gas solubility optimization-based deep neuro fuzzy network in MANET," *Engineering Applications of Artificial Intelligence*, vol. 123, p. 105969, 2023.

[23] S. Nalluri and K. S. Archana, "Performance analysis of machine-learning-based detection of sinkhole network layer attack in MANET," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 12, 2022.

[24] V. Ponnusamy, M. Humayun, N. Z. Jhanjhi, A. Yichiet, and M. F. Almufareh, "Intrusion detection systems in Internet of Things and mobile ad-hoc networks," *Computer Systems Science and Engineering*, vol. 40, no. 3, pp. 1199–1215, 2022.

[25] M. Prasad, S. Tripathi, and K. Dahal, "An intelligent intrusion detection and performance reliability evaluation mechanism in mobile ad-hoc networks," *Engineering Applications of Artificial Intelligence*, vol. 119, p. 105760, 2023.

[26] S. K. Prashanth, H. Iqbal, and B. Illuri, "An enhanced grey wolf optimisation–deterministic convolutional neural network (GWO–DCNN) model-based IDS in MANET," *Journal of Information & Knowledge Management*, vol. 22, no. 4, 2023.

[27] S. Rahman, M. Irfan, M. Raza, K. M. Ghori, S. Yaqoob, and M. Awais, “Performance analysis of boosting classifiers in recognizing activities of daily living,” *International Journal of Environmental Research and Public Health*, vol. 17, no. 3, p. 1082, 2020.

[28] O. Sbai and M. Elboukhari, “Deep learning intrusion detection system for mobile ad hoc networks against flooding attacks,” *IAES Int. J. Artif. Intell.*, vol. 11, no. 3, pp. 878–885, 2022.

[29] M. S. Shaik and F. Mira, “A comprehensive mechanism of MANET network layer-based security attack prevention,” *International Journal of Wireless and Microwave Technologies*, vol. 10, no. 1, pp. 38–47, 2020.

[30] P. Sun, P. Liu, Q. Li, et al., “DL-IDS: Extracting features using CNN-LSTM hybrid network for intrusion detection system,” *Security and Communication Networks*, vol. 2020, pp. 1–11, 2020.

[31] S. Venkatasubramanian, “Multistage optimized fuzzy-based intrusion detection protocol for NIDS in MANET,” *International Journal of Innovative Research in Technology*, vol. 8, no. 6, pp. 301–311, 2021.

[32] J. Xue and B. Shen, “Dung beetle optimizer: A new meta-heuristic algorithm for global optimization,” *The Journal of Supercomputing*, vol. 79, no. 7, pp. 7305–7336, 2022.

[33] Y. Yang and P. Zhao, “Research on dung beetle optimization-based stacked sparse autoencoder for network situation element extraction,” *IEEE Access*, vol. 12, pp. 24014–24026, 2024.

[34] Potnurwar, et al., "Deep Learning-Based Rule-Based Feature Selection for Intrusion Detection in Industrial Internet of Things Networks," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 11, no. 5s, 2023.