International Journal of Basic and Applied Sciences, 14 (7) (2025) 508-519

International Journal of Basic and Applied Sciences

Website: www.sciencepubco.com/index.php/IJBAS
https://doi.org/10.14419/h1qhyd19
Research paper



An Optimization of The Hyperparameters of Multi-Valued Neutrosophic ConvLSTM for Intrusion Detection

Veni Chinnasamy [1] *, Selvi Sellappan [2]

[1] *Research Scholar, Department of Computer Science, PSG College of Arts and Science, Coimbatore, Tamil Nadu, India*
[2]*Associate Professor & Head, Department of Computer Science, PSG College of Arts and Science, Coimbatore, Tamil Nadu, India*
**Corresponding author E-mail: venivijayan.c.phd@gmail.com*

Received: September 11, 2025, Accepted: October 22, 2025, Published: November 23, 2025

Abstract

In today's world, cybersecurity has become indispensable as networks play an integral role in daily life. The increasing size and complexity of these networks heighten the risk of new types of attacks, making it crucial to design robust and efficient Intrusion Detection Systems (IDS). Numerous Deep Learning (DL) techniques have been developed to automate IDS and detect abnormal network behaviors. But certain models might be inaccurate during the training phase due to the presence of numerous irrelevant features. To solve these problems, Feature Multi-Valued Neutrosophic Convolutional Long Short-Term Memory (FMVN-ConvLSTM) was created. But the hyperparameters of ConvLSTM were not optimized, which limits its intrusion accuracy and increases model complexity. This paper leverages the Dove Swarm Optimization (DSO) model to fine-tune ConvLSTM hyperparameters for enhancing IDS accuracy and reducing computational complexity. Inspired by the flocking behavior of doves, DSO dynamically adjusts key ConvLSTM parameters like rate of learning, size of batch, weight decay, dropout rate, and optimizer to achieve an ideal balance between performance and efficiency. DSO navigates the hy-hyperparameter space by mapping initial dove positions to starting configurations, refining them iteratively to facilitate efficient model learning. Unlike other optimization strategies, DSO enhances population diversity and streamlines model complexity, allowing ConvLSTM to capture essential temporal and spatial patterns. This approach produces a robust IDS capable of identifying various network threats with high accuracy and reduced computational overhead. The complete model is termed an Optimized FMVN-ConvLSTM (OFMVN-Con-vLSTM). In the final analysis, the experiment results show that compared to other current models on three distinct datasets, CIC-IDS2018, WSN-DS, and UNSW-NB15, the OFMVN-ConvLSTM achieves an accuracy of 97.48%, 96.35%, and 96.85%.

Keywords: *IDS; Deep Learning; ConvLSTM; Dove Swarm Optimization; Hyperparameter Tuning.*

1. Introduction

There is a pressing need for strong network security due to the exponential growth of technologies like cloud computing, big data, and the Internet of Things (IoT). These innovations have a significant influence on the day-to-day interactions. Traditional network security sys-tems like firewalls and data encryption are becoming more ineffective as hackers advance. Cybersecurity experts have stressed the value of current network IDS in preventing unlawful behavior. Information & communication system security is enhanced, and data integrity, confidentiality, and availability are protected by IDS. Identity and access management systems use two main detection strategies—abuse detection and anomaly detection—to enhance network security and prevent damaging attacks. Abuse detection, which uses previously identified patterns of intrusions and attacks, is also known as signature-based detection. By contrast, anomaly detection looks for network activity that doesn't seem to fit the usual. More sophisticated networks will challenge the effectiveness of current security measures with the emergence of new threats. IDS relies on rapid and accurate threat detection and mitigation to ensure network security. Anomaly detec-tion is essential in this process, as it can detect both common and rare cyberattacks. DL models use complex neural network topologies to automatically identify and categorize harmful activity in network data for enhancing the accuracy of IDS. These systems can adapt to evolving threats by detecting both known and unknown attacks through automated feature extraction and real-time analysis. An integrated model that uses a combination of Convolutional Neural Networks (CNNs) and LSTM networks to identify harmful actions was created in [1] for IDS. It takes data from networks and extracts features based on their location and time. Batch normalization, as well as dropout, are additional components that enhance the model's performance. However, this model faces epistemic uncertainty, which influences DL performance in IDS applications. Addressing the uncertainty in incursion datasets needs a more advanced DL architecture and novel feature extraction approaches.

Copyright © Veni Chinnasamy, Selvi Sellappan. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In the recent past, neutrosophic logic has been successfully incorporated into machine learning and deep learning models in various applications, such as pattern recognition and sentiment analysis. Such neutrosophic representations can easily manage indeterminacy and uncertainty across a wide range of different types of data and hence increase the interpretability and robustness of models. This broader view supports the generality and generalizability of neutrosophic theory when it comes to IDS. Accordingly, the Multivalued Neutrosophic Convolutional LSTM (MVN-ConvLSTM) model [2] was developed to efficiently manage uncertainty in IDS by extracting deep features that improve predicted accuracy. In this approach, the Neutrosophic Set (NS) domain is utilized to categorize incursion features into six groups: Positive Truth-membership, Positive Falsity membership, Positive indeterminate toward falsity and truth membership, Negative Truth-membership, Negative Falsity membership, and Negative indeterminate toward falsity and truth membership. This model uses four parallel processing routes, and backpropagation to update weights results in better FAR and Detection Rate (DR). Nonetheless, its accuracy during training suffers when dealing with datasets containing a large number of irrelevant attributes. To overcome these constraints, the Feature Prioritized MVN-ConvLSTM (FMVN-ConvLSTM) model [3] enhances IDS performance by prioritizing feature relevance and reducing complexity. An in-model Feature Ranking (FR) algorithm was applied for efficient processing and interpretation. It utilizes ConvLSTM's intrinsic dynamics to evaluate feature relevance without external methods and applies the L1 norm to promote sparsity by assigning near-zero values to irrelevant features. Furthermore, ConvLSTM with weighted softmax enhances feature prioritization and reduces misclassification rates in IDS tasks.

1.1. Problem definition

Although both the MVN-ConvLSTM and FMVN-ConvLSTM models can successfully address the issue of uncertainty through neutrosophic representations, their empirical effectiveness is hampered by poorly tuned hyperparameters and redundant features in large datasets of intrusions. Poor tuning is often the cause of overfitting, slow convergence, and increased computational cost, and adding irrelevant attributes reduces the classification accuracy. As a result, there is a pressing need to apply an optimization mechanism to automatically optimize the parameters of ConvLSTM models, which leads to an optimal trade-off between predictive accuracy, model generalization, and computational efficiency. To address this, this paper suggests a hybrid approach involving the combination of DSO and the FMVN-ConvLSTM framework, enabling the adaptive optimization of the hyperparameters and increasing the capability of the model to identify a diverse range of dynamic cyberattacks in real time.

1.2. Contributions

In this paper, OFMVN-ConvLSTM is developed to lower the computational complexity and enhance the accuracy performance in IDS. This model applies the DSO model, which is inspired by the collective behavior of doves, utilizing their flocking mechanisms to explore the search space intelligently and converge on the optimal solutions. Neuron count, hidden unit count, rate of learning, weighting decay, epoch count, batch size, dropout rate, partition count, cluster count per batch, momentum, optimizer, and loss function are the critical parameters of ConvLSTM. By simulating the flight patterns of doves, DSO dynamically adjusts key parameters of ConvLSTM to achieve an optimal configuration that balances accuracy and computational efficiency. The initial group of doves signifies the starting hyperparameters, while their subsequent crumb positions reflect the search for optimal hyperparameter values, facilitating effective model learning. For instance, DSO tunes the learning rate and momentum to stabilize the training process, avoiding overshooting and ensuring faster convergence. It also optimizes the batch size and number of epochs to find an effective trade-off between training time and model generalization. The dropout rate is optimized to prevent overfitting, while weight decay is adjusted to regulate model complexity. Additionally, DSO configures the number of partitions and clusters per batch, which enhances ConvLSTM's ability to manage data more efficiently, especially in high-dimensional intrusion datasets. By selecting the best optimizer and loss function, DSO further refines the training process to minimize errors effectively. The adaptive parameter tuning by DSO improves ConvLSTM's ability to capture relevant temporal and spatial patterns for intrusion detection systems (IDS), maximizing accuracy while minimizing computational load. Compared to other optimization strategies, DSO enhances population diversity and simplifies the model, leading to higher accuracy rates in identifying various hostile activities within the network.
Here is how the remainder of the paper is structured: The research on several IDS models based on DL is summarized in Section 2. Section 3 outlines the OFMVN-ConvLSTM paradigm, while Section 4 proves its efficacy. Section 5 presents findings and suggestions for further study.

2. Literature Survey

This section presents the IDS models based on different approaches, categorized as feature-based, DL-based, and hybrid DL with optimization algorithms.

2.1. DL-based IDS models

Kasongo [4] suggested a DL method for IDS using the Recurrent Neural Networks (RNN) framework. In this model, the gathered dataset was pre-processed, normalized, and feature extraction was performed using the XGBoost model. Then, the RNN with the Gated Rectified Unit (GRU) model was employed for the detection of intrusion. However, this model results in high training time and memory space. Shobana & Sai [5] constructed an IDS network using a step-based DL model. In this method, a two-initiative network outage recognition is based on GoogLeNet Inception and CNN models. This model was leveraged using the GoogLeNet to detect the parallel problem from organizational packaging to distinguish between the quality of raw parcel and traffic data. Multiclass breakdowns and organizational package elements were also distinguished using the CNN model. However, this model results in a slower convergence rate.
Hnamte & Hussain [6] presented a dependable IDS using a Deep Convolutional Neural Network (DCNN). This method identifies valuable features in network traffic data, enhancing its accuracy and robustness in IDS. To train the DCNN to distinguish between normal and abnormal behavior, a large-scale dataset was used. This dataset includes both benign and malicious network traffic. However, this model was ineffective and achieved higher FAR results. Mohammadian et al. [7] proposed a gradient-driven method for launching adversarial attacks against DL-based network IDS. The Jacobian Saliency Map was used to identify the optimal features for generating adversarial instances in network data, analyzing adversarial attacks in network IDS. Ultimately, CNN was employed for classifying malicious network activity. But the feature of adversarial attacks could vary significantly across different contexts, which might affect classification accuracy.

To improve NIDS's multi-classification features, Ayantayo et al. [8] proposed a DL model that makes use of feature fusion techniques. To learn the correlations between various input features more efficiently, this method utilized feature fusion mechanisms such as early fusion, late fusion, and late ensemble learning fusion using fully linked deep networks. Additionally, it reduces the possibility of bias associated with a certain feature type to generalize the behavior for intrusion detection systems. But this model takes a long time to detect the intrusion and other attacks. To aid in the identification of harmful assaults, Silivery et al. [9] built a DL-based trustworthy IDS system. Datasets were first numericalized and normalized as part of the preprocessing phase. The LSTM-RNN model was fed preprocessed training and testing data. To enhance network security, LSTM-RNN was employed for the prediction and classification of multi-class malicious. Qazi et al. [10] designed a hybrid DL-driven network IDS (HDLNIDS). The method involves preprocessing and classifying network traffic data to address imbalance issues. A convolutional recurrent neural network is used to predict network attacks, collecting local features for improved efficiency and predictability. A deep-layered RNN retrieves these features for IDS. However, the model should focus on advanced models to address data imbalance issues.

In summary, DL-based IDS models have demonstrated superior feature extraction and classification performance, but most of the current systems have limited scalability and generalizability, especially in the presence of large or imbalanced datasets. Furthermore, their inappropriate hyperparameter settings cause overfitting issues, longer training times, and lower classification accuracy. To address these issues, there is a need to employ optimization techniques for hyperparameter tuning, which achieves more reliable and effective intrusion detection.

2.2. Hybrid DL with optimization-based IDS models

For IDS prediction, Turukmane et al. [11] proposed a Mud ring-assisted Multilayer Support Vector Machine (M-MultiSVM). This approach used min-max normalization and null value handling for data preprocessing. The features were retrieved using Modified Singular Value Decomposition (M-SVD), while class imbalance and overfitting were addressed by the Advanced Synthetic Minority Oversampling Technique (ASMoT), with optimization performed through Opposition-based Northern Goshawk Optimization (ON-gO). MultiSVM handled attack classification with Mud Ring optimization, tuning the hyperparameters. However, it adds significant computational layers, potentially leading to increased processing time and resource demands, especially when handling large datasets. Aljehane et al. [12] developed the Global Jackal Optimization Algorithm with DL-assisted IDS for Network Security (GJOADL-IDSNS), an efficient network security algorithm, with the use of DL-assisted intrusion detection systems. To make the input data more usable, our model used data normalization. Afterwards, GJOA was used to pick features. The next step in IDS prediction was to use the Attention-Based Bi-Directional LSTM (A-BiLSTM) model. Lastly, the Salp Swarm Algorithm (SSA) was used to optimize the A-BiLSTM hyperparameters. However, this model leads to enhanced computational cost. The Gated Attention Dual LSTM (Dugat-LSTM) was proposed by Devendiran and Turukmane [13] as an IDS method. As a preprocessing step, this model employs data cleansing as well as M-squared normalization. To extract features from the imbalanced datasets, Kernel-assisted Principal Component Analysis (K-PCA) was used after Extended Synthetic Sampling. Chaotic Honey Badger was the optimization algorithm that chose the best characteristics. Following feature extraction, the Dugat-LSTM model is employed for attack classification.

In summary, studies have demonstrated that hybrid DL and optimization algorithms improve IDS performance by optimizing feature selection and hyperparameter tuning. However, conventional optimization algorithms often have a high computational cost, slower convergence speed, and poor tradeoff between exploration and exploitation. To address these issues, this study introduces the DSO algorithm that accelerates the convergence speed, reduces the computational complexity of hyperparameter tuning, and increases the detection accuracy.

3. Proposed Methodology

The suggested OFMVN-ConvLSTM model is explained here. A diagrammatic representation of the suggested approach is illustrated in Figure 1.

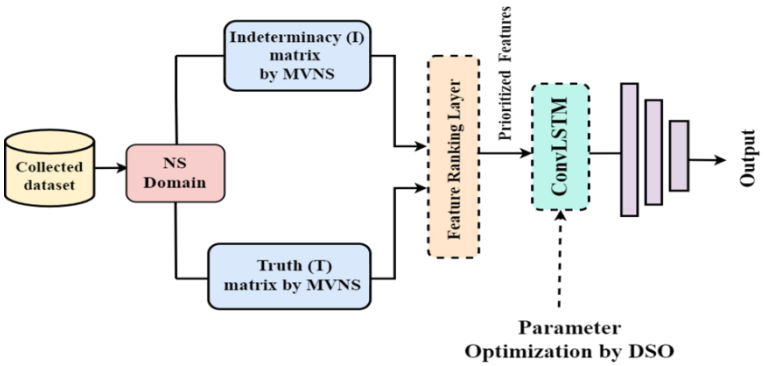


Fig. 1: Pipeline of the Proposed Model.

3.1. Deep feature extraction and feature prioritization

As mentioned in Figure 1, the proposed IDS framework leverages the NS domain to manage uncertainty in intrusion datasets. It uses the Multi-Valued Neutrosophic (MVN) method [3] to glean deep features from the data that has been collected. A feature prioritization algorithm [4] is then developed to establish an FR layer, which prioritizes and selects the most important features. These extracted features are subsequently input into a ConvLSTM [2] model, enabling the detection of abnormal activities within the network and enhancing its security through efficient IDS capabilities. Since the parameters of ConvLSTM are not wisely optimized, this article utilizes the DSO to enhance the accuracy rate in IDS.

3.2. Optimizing ConvLSTM hyperparameters using dove swarm optimization

DSO is a bio-inspired algorithm that mimics the social foraging behavior of doves to find optimal solutions to complex problems. Each dove represents a potential solution and moves through the solution space based on interactions with other doves and the best-performing leader dove. Doves adjust their positions by following leaders and adaptively balancing exploration and exploitation of the space. The algorithm iterates until convergence, aiming to locate the highest crumb (optimal solution) based on a defined objective. Dove foraging behavior has prompted various applications such as feature selection, parameter tuning, or model optimization in machine learning models. In this framework, the DSO algorithm can optimize the hyperparameters of the ConvLSTM model by dynamically adjusting the parameters based on the efficiency of the best dove in finding crumbs. Each dove (agent) explores potential hyperparameter settings in the solution space by foraging for the configuration that maximizes performance. Key steps of DSO applied to fine-tune the hyperparameters of ConvLSTM are listed below.

3.2.1. Define the hyperparameter search space and objective function

Maximizing the fitness of a specific set of ConvLSTM hyperparameters is the goal of DSO. The objective function $F(X) = [f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}]$ Represents this fitness. Parameters such as neuron count (f_1), total number of hidden units (f_2), rate of learning (f_3), weight decay (f_4), epoch count (f_5), size of batch (f_6), rate of dropout (f_7), partition count (f_8), number of clusters in each batch (f_9), momentum (f_{10}), optimizer (f_{11}), and loss function (f_{12}) are all part of the hyperparameter set, which determines each optimal location X. Every point X is considered to have crumbs, and the total amount of crumbs at all points X is equal to F(X) crumbs. Finding the area with the most crumbs is the best option.

3.2.2. Dove initialization

Consider n As the number of doves that are determined to be deployed in the solution space, which are ideally scattered evenly across a rectangular surface. Start with an initial epoch count of 0 and modify the level of satiety. $\mathcal{S}_D^e$ for each dove $D = 1, \dots, n$ .The ConvLSTM hyperparameters, denoted by the dove's location vector $X_D \subset r^k$, can be initialized randomly or employing lattice initialization. The most straightforward approach is to dynamically generate the X_D Around the solution space. The lattice-initialized approach is another option. The appropriate stages are shown below.

1) To accelerate training and establish a topologically ordered feature map, two effective weight initialization strategies are utilized to configure weight vectors. The novel initialization technique is particularly suitable for this algorithm, which is developed using an initialization strategy. It assumes a minimal hyper-rectangle in parameter space containing valid values for hyperparameters f_1 to f_{12}, represented by $[L_1, U_1], \dots, [L_k, U_k]$ where L_k and U_k Denote the bounds for each dimension in the solution space. Each boundary contains valid parameter values for ConvLSTM.
2) The proposed initialization method relies on the idea of reducing the N-dimensional hyper-rectangle to a 2D plane such that a 2D-net can encapsulate the solution space. So that everything is very clear, the rectangular cells are indexed from 1 to U*V using a and b. Below are the steps that need to be followed.

3.2.2.1. Position adjustment

Construct the cells in the four corners. Also, adjust the four neurons on the network's corners with the following weight vectors (W) as given in Eqns. (1.1 – 1.4).

$$W_{1,1} = (L_1, L_2, \dots, L_k)^T \tag{1.1}$$

$$W_{U,V} = (U_1, U_2, \dots, U_k)^T \tag{1.2}$$

$$W_{1,V} = (L_1, L_2, \dots, L_{\lfloor k/2 \rfloor}, U_{\lfloor k/2 \rfloor + 1}, \dots,)^T \tag{1.3}$$

$$W_{U,1} = (U_1, U_2, \dots, U_{\lfloor k/2 \rfloor}, L_{\lfloor k/2 \rfloor + 1}, \dots, L_k)^T \tag{1.4}$$

The above-mentioned weight vectors $W_{1,1}$, $W_{U,V}$, $W_{1,V}$, $W_{U,1}$ Adjust the initial extreme configurations of ConvLSTM hyperparameters.

3.2.2.2. Constructing the edges of the solution space

Set up the cells on the four edges. The Eqns. (2.1 – 2.4) represents the initialled cell values on the four edges,

$$W_{1,b} = \frac{W_{1,V} - W_{1,1}}{V-1}(b-1) + W_{1,1}$$

$$= \frac{b-1}{V-1} W_{1,V} + \frac{V-b}{V-1} W_{1,V} \quad b = 2, \dots, V-1 \tag{2.1}$$

$$W_{U,b} = \frac{W_{U,V} - W_{U,1}}{V-1}(b-1) + W_{U,1}$$

$$= \frac{b-1}{V-1} W_{U,V} + \frac{V-b}{V-1} W_{U,1} \quad b = 2, \dots, V-1 \tag{2.2}$$

$$W_{a,1} = \frac{W_{U,1} - W_{1,1}}{U-1}(a-1) + W_{1,1}$$

$$= \frac{a-1}{V-1} W_{U,1} + \frac{U-a}{U-1} W_{1,1} \quad a = 2, \dots, U-1 \tag{2.3}$$

$$W_{a,V}= \frac{W_{U,V}-W_{1,V}}{U-1}(a-1)+W_{1,V}$$
$$= \frac{a-1}{U-1} W_{U,V}+\frac{V-a}{V-1} W_{1,V} \; a= 2,\ldots, U-1 \quad (2.4)$$

Every one of the four nodes in the network has its own unique weight vector that is created. Following a left-to-right and top-to-bottom organization scheme characterizes the remaining neurons. Apply Eqns. (2.1 – 2.4) to interpolate values along the boundaries, allowing for a structured, initialized grid where doves are arranged based on ConvLSTM configurations close to initial estimates. The following is a pseudo-code explanation of the initialization procedure for residual neurons:

1) Start
2) For b=2 to V-1
3) For a=2 to U-1
4) Determine

$$W_{a,b}= \frac{W_{U,b}-W_{1,b}}{U-1}(a-1)+W_{1,b}\frac{a-1}{U-1}.W_{U,b}+\frac{U-a}{U-1}.W_{1,b}$$
$$\frac{a-1}{U-1}\left(\frac{b-1}{V-1}.W_{U,V}+\frac{V-b}{V-1}.W_{U,1}\right)+\frac{U-a}{U-1}\left(\frac{b-1}{V-1}.W_{1,V}+\frac{V-b}{V-1}.W_{1,1}\right) \quad (3)$$
$$= \frac{((b-1)(a-1)W_{U,V}+(b-1)(U-a)W_{1,V}+(V-b)(a-1)W_{U,1}+(V-b)(U-a)W_{1,1}}{(V-1)(U-1)}$$

5) End For
6) End For
7) End

3.2.3. Dove fitness evaluation

At each epoch e, determine the position of each dove D based on the dove's fitness $F(W_b^e)$, where b =1,….n . This fitness evaluates the effectiveness of the dove's position in influencing the ConvLSTM configuration concerning the chosen objective, like classification accuracy.

3.2.4. Identifying the optimal dove position

Identify the dove D_b^e That is closest to the maximum crumbs according to the highest criterion at epoch. e, as determined by Eq. (4),

$$D_b^e=\text{argmax}\{F(W_b^e)\}, b=1,\ldots n \quad (4)$$

By using Eq. (4), the doves with the maximum fitness score are determined. This dove represents the configuration with the highest crumbs (best performance) and also serves as a reference for other doves to follow each other.

3.2.5. Update Dove Satiety

Apply Eq. (5) to calculate the satiety degree. $\mathcal{S}_b^e$ for each dove

$$S_b^e= \lambda S_b^{e-1}+e^{(F(W_b)-F(W_{D_F}))}, b=1,\ldots n \quad (5)$$

This update motivates doves with higher fitness to keep exploring their current region, while guiding others toward areas with potential improvements.

3.2.6. Identifying a high degree of satiety

Determine the highest satisfied dove. D_S^e with the greatest degree of satiety (leader) by utilizing the following maximum criteria, b=1,2,….n

$$D_S^e= \arg \max_{1\le b\le n}\langle S_b^e\rangle \quad (6)$$

D_S, chosen using Eq. (6), signifies the dove that demonstrates the best foraging behavior and should be emulated by the rest of the flock. This provides the optimal ConvLSTM configuration. All other doves update their position vectors to approximate. $W_{D_s}^e$.

3.2.7. Position update

Change the position vector of each dove D using the following maximum criterion.

$$W_b^{e+1}= W_b^e+1\,\delta_b^e\,(W_{D_s}^e- W_b^e) \quad (7)$$

Where,

$$\delta_b^e= \left(\frac{\delta_{b_s}^e-\delta_{b_s}^e}{\delta_{b_s}^e}\right)\left(1-\frac{\|W_b^e- W_{D_s}^e\|}{\text{Max_Dis}}\right) \quad (8)$$

$$\text{Max}_{\text{Dis}} = \max_{1\le b\le n_} \|W_b - W_a\| \quad (9)$$

Where, Max_Dis Is the maximum distance. The learning rate l It is modified as per the dove position vector. δ_b^e The scaling factor assists in balancing exploitation and exploration, refining ConvLSTM parameters by gradually converging toward the best-performing configurations.

3.2.8. Termination criterion

Make sure to increase the total number of epochs (for example, set e = e+1) and repeat steps in sections 3.2.3–3.2.7 until the terminate condition is met. Here are the rules for ending a contract:

$$\left|F_{D_S}^e - T(e)\right| \leq \gamma \text{ or } e \leq \max \text{ epoch} \quad (10)$$

The DSO approach has a complexity order of $Onn_D{}^e$, therefore, there will be n_D data points within the dataset, where n represents the number of doves as well as e Represents the number of epochs. The optimal ConvLSTM hyperparameters correspond to the dove configuration that achieves maximum satisfaction after convergence. If the optimization criteria include determining the smallest W_b^e, the sequence of Eqns. (4) and (5) may be changed appropriately.

$$D_b^e = \text{argmin}\{F(W_b^e)\}\ b = 1,\ldots,n \quad (11)$$

$$S_b^e = \begin{cases} \lambda S_b^{e+1} + e^{\left(F(W_b) - F(W_{D_F})\right)} & F(W_{D_F}) \neq 0 \\ \lambda S_b^{e-1} + 1, & F(W_{D_F}) = 0 \end{cases} \quad b = 1,\ldots,n \quad (12)$$

For better clarity, the updating rules are interpreted as given in Eqns. (7 – 9) as follows:

1) Doves in a flock are inspired by the success of the greatest individual and want to replicate it. They follow the dove exhibiting the highest level of satisfaction to discover more food sources. This form of social learning is simulated by updating the position vector W_b^e to better align with the position vector of the most satiated dove, $W_{D_S}^e$.

$$W_b^{e+1} = W_b^e + l\delta_b^e \left(W_{D_S}^e - W_b^e\right) \quad (13)$$

2) A dove with higher satiety is more cautious and hesitant to change its foraging strategy, while a dove with lower satiety is more likely to modify its strategy and emulate the best individual behavior. This societal effect is demonstrated by comparing modifications to the first term value on the right-hand side of Eq. (9) is stated as follows:

$$\left\| W_b^e - W_{D_s}^e \right\| = \left(\frac{S_{j_S}^e - S_j^e}{\delta_{j_s}^e}\right) \quad (14)$$

3) The level of social influence diminishes with increasing distance, suggesting that a dove's influence is directly related to its proximity to the best performing in the flock. This social influence is modelled by making the adjustment magnitude proportional to the third term on the right-hand side of Eq. (8), i.e., $\left((1 - \frac{\left\| W_b^e - W_{D_s}^e \right\|}{\text{Max_Dis}}\right)$.

3.3. Model training

The ConvLSTM for IDS prediction is trained with a set of optimal hyperparameters chosen by the DSO. The selected hyperparameter values are listed in Table 1.

Table 1: Hyperparameter Optimization Data Set for ConvLSTM Models

Parameters	Search Space	Optimal Range
ConvLSTM		
Neuron count	[32, 64, 96, 128, 160]	128
Rate of Learning	[0.01, 0.001, 0.0001]	0.001
Loss Function	[Cross-entropy, Mean Squared Error]	Cross-entropy
Batch size	[16, 32, 64, 128]	64
Rate of dropout	[0.2, 0.4, 0.6, 0.8]	0.6
Partition count	[100, 200, 300, 400]	300
Number of groups per batch	[1, 2, 3, 4]	3
Momentum	[0, 1]	0.9
Hidden unit count	[64, 128, 256, 512]	256
Weight decay	[0.0001, 0.01]	0.01
Optimizer	[Stochastic Gradient Descent, RMSprop, Adam]	Adam
Number of epochs	[50, 100, 150, 200]	100
DSO		
Number of populations	[80, 90, 100, 110, 120]	90
Maximum Number of Iterations	[60, 70, 80, 90]	70
Step size	[0.55, 0.65, 0.75]	0.65
λ (Weighting Factor)	[0.5, 0.6, 0.7, 0.8, 0.9]	0.8
ℓ (Learning Rate)	-	0.18~0.375

Hence, the hyperparameters of ConvLSTM are optimized using DSO, which intends to eliminate the computational complexity and enhance the classification accuracy in the application of IDS. Algorithm 1 describes the DSO pseudocode for hyperparameter adjustments in ConvLSTM.

Algorithm 1: DSO-based Hyperparameter Optimization in ConvLSTM
Input: These are the model's hyperparameters: neuronal density, hidden unit density, rate of learning, weighting decay, epoch count, batch size, dropout rate, partition count, cluster count per batch, momentum, optimizer, and loss function.
Output: Adjusting the hyperparameters for the optimal solution
1) Begin
2) Set the number of doves n.
3) Initialize dove positions X_D randomly or using lattice initialization within defined bounds $[L_k, U_k]$ For each hyperparameter.
4) Initialize the degree of satiety S_D^e For each dove.
5) Set initial epoch e.
6) While ($e<e_m$)
7) Compute the fitness value $F(W_b^e)$ For all doves based on the performance of the ConvLSTM model using the current hyperparameters.
8) Identify the dove D_b^e Closest to the highest crumbs as in Eq. (11).
9) Update the satiety degree for each dove S_D^e As in Eq. (12).
10) Using Eq. (6), choose the dove that has the greatest degree of satiety.
11) Update the position of each dove based on the leader as per Eq. (7).
12) If convergence criteria are met or e reaches e_{max} Exit the loop.
13) Increase the epoch count e=e+1
14) End while
15) Return the best dove D_b^e representing optimal hyperparameters for ConvLSTM and the corresponding optimal fitness value $F(W_b^e)$.
16) End

Figure 2 displays a flowchart of DSO for optimizing the ConvLSTM's hyperparameters.

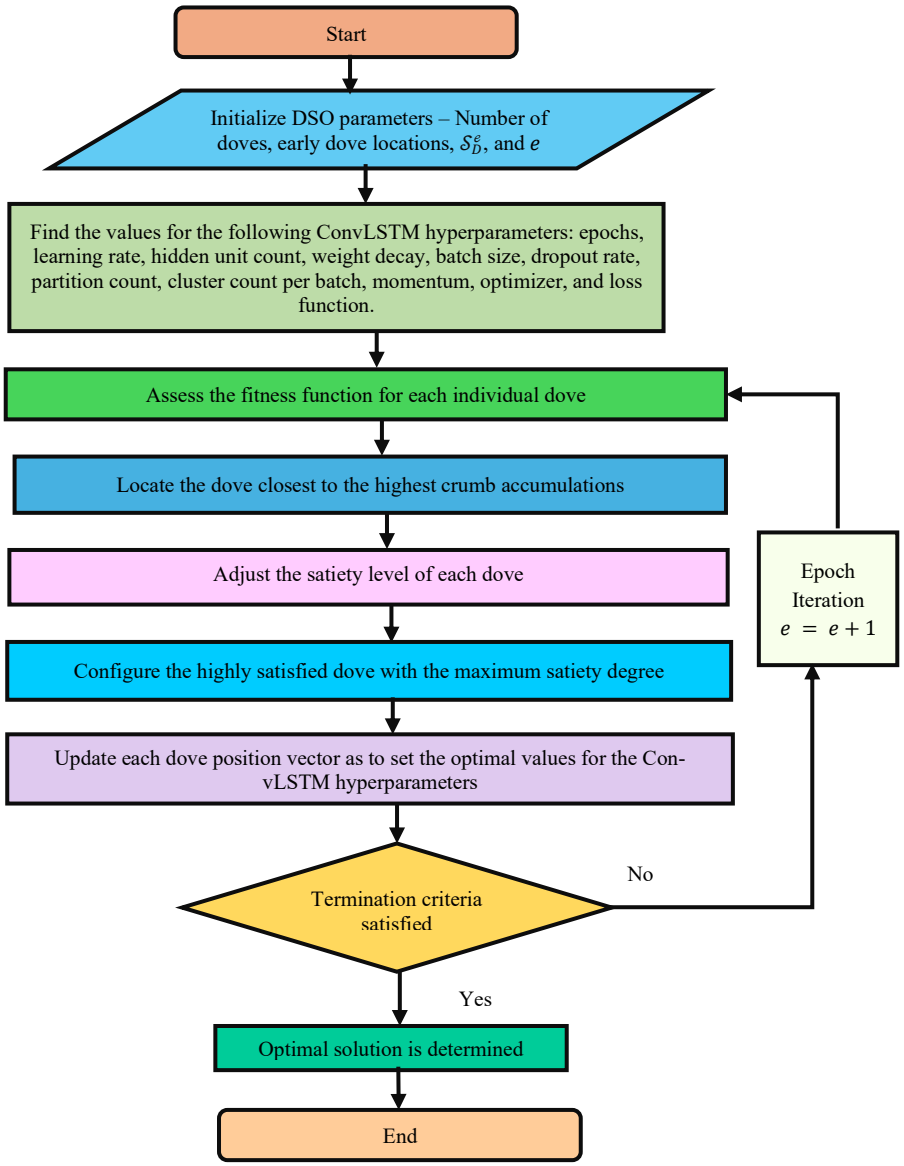


Fig. 2: Flowchart of DSO for Hyperparameter Selection in ConvLSTM.

4. Results and Discussion

4.1. Dataset description

CIC-IDS2018: A comprehensive cybersecurity dataset incorporating profiles was generated by the Communications Security Establishment as well as the Canadian Institute for Cybersecurity. Application, protocol, and lower-level network entity distribution models are included, along with extensive descriptions of invasions. Some of the attack scenarios included in the dataset are brute-force, Heartbleed, botnet, DoS, DDoS, and web attacks, as well as network infiltration. Accompanying the victim organization's log files and network traffic are eighty aspects of the traffic that were retrieved using CICFlowMeter-V3.
WSN-DS: The 2016-developed WSN-DS monitors sensor nodes to identify traffic in wireless networks. Utilizing the LEACH routing procedure, it extracts 170 entries with 23 characteristics. In addition to the usual records, it contains four kinds of attacks: DoS, flooding, grayhole, and blackhole, as well as TDMA details.
UNSW-NB15: In 2015, the ACCS compiled data from 162 real-world websites, such as BID, CVE, and MSD, to generate a dataset that included both benign traffic and records of nine different types of assaults. These attacks included fuzzers, analysis, backdoors, and DoS, as well as exploits. For experimental purposes, the samples and features collected from the above three datasets are listed in Table 2 below.

Table 2: Utilized Samples and Features

Datasets	Examples	Characteristics
WSN-DS	68602	18
UNSW-NB15	175341	45
CIC-IDS2018	261147	30

4.2. Experimental result

This section evaluates the OFMVN-ConvLSTM model's performance by simulating it with Python 3.7.8 across three different datasets (discussed in section 4.1). For experimental purposes, the collected dataset has been split into 80% training and 20% testing sets. A comparison study is also carried out using these datasets with the proposed OFMVN-ConvLSTM and other algorithms such as CNN-LSTM [1], MVN-ConvLSTM [2], FMVN-ConvLSTM [3], HDLNIDS [10], GJOADL-IDSNS [12], and Dugat-LSTM [13]. Below is a brief illustration of the evaluation measures that are used to assess the efficiency of both the proposed and existing models.

4.2.1. Accuracy

To determine the accuracy, they divide the whole number of data instances that were fed into the model (as indicated in Eq. (15)) by the number of instances where the intrusion data was correctly classified as either regular records or attack type.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{15}$$

Here, True Positive (TP) denotes normal records that are appropriately classified as normal, False Positive (FP) denotes normal records that are wrongly categorized as malicious, True Negative (TN) denotes normal records that are erroneously classified as malicious, and False Negative (FN) denotes malicious records that are incorrectly identified as normal.

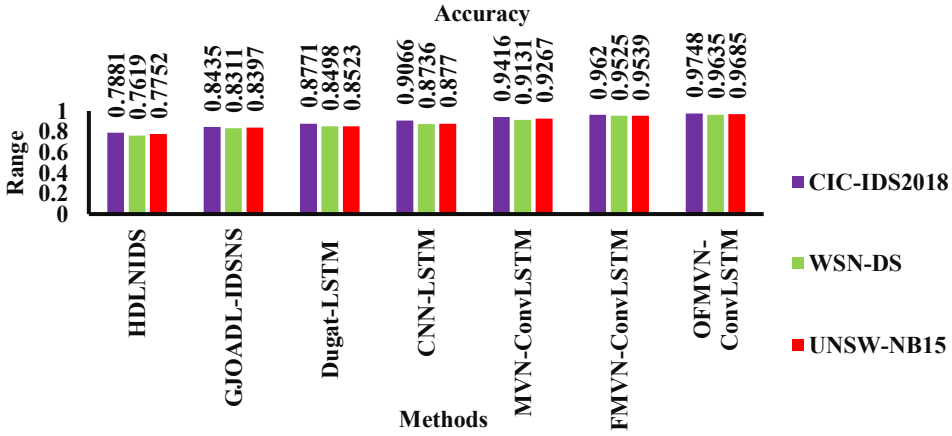


Fig. 3: Comparison of Accuracy.

Figure 3 displays the accuracy findings of both the suggested and existing IDS for various datasets. It shows that the accuracy of the proposed method using the CIC-IDS2018 dataset is 23.69%, 15.57%, 11.14%, 7.52%, 3.53%, and 1.33% greater than HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM. On WSN-DS, OFMVN-ConvLSTM outperforms HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM models in terms of accuracy by 26.46%, 15.93%, 13.38%, 10.29%, 5.52%, and 1.15%, respectively. Compared to HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM models on UNSW-NB15, MVN-ConvLSTM achieves an accuracy of 24.94%, 15.34%, 13.63%, 10.43%, 4.51%, and 1.53% higher, respectively. The results show that compared to other models currently available for detecting network intrusions, the suggested OFMVN-ConvLSTM performs better in terms of accuracy.

4.2.2. Precision

The fraction of samples correctly identified as intrusions as a proportion of all samples received. The precision formula is shown in Eq. (16).

$$\text{Precision} = \frac{TP}{TP + FP} \tag{16}$$

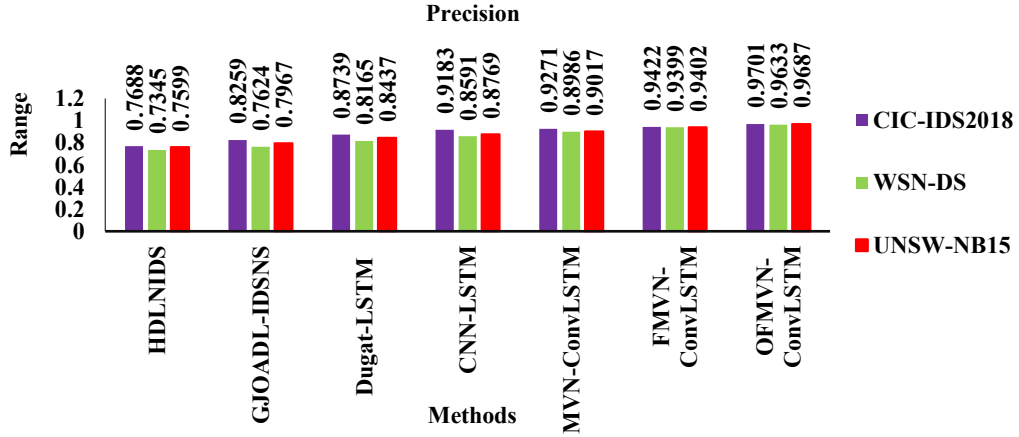


Fig. 4: Comparison of Precision.

Figure 4 displays the precision findings of both the suggested and existing IDS on various datasets. The data indicate that the precision of OFMVN-ConvLSTM using the CIC-IDS2018 dataset is significantly higher than that of HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM by percentages of 26.18%, 17.46%, 11.01%, 5.64%, 4.64%, and 2.96%, respectively. Similarly, on the WSN-DS dataset, OFMVN-ConvLSTM outperforms HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM by 31.15%, 26.35%, 17.98%, 12.13%, 7.2%, and 2.49%. Furthermore, on the UNSW-NB15 dataset, OFMVN-ConvLSTM shows precision improvements of 27.48%, 21.59%, 14.82%, 10.47%, 7.43%, and 3.03% compared to HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM. This analysis demonstrates that the proposed OFMVN-ConvLSTM model enhances precision performance compared to other existing models for identifying intrusions in network systems.

4.2.3. Recall

It is the percentage of intrusion samples that are accurately classified relative to the total number of intrusion samples. Eq. (17) defines the recall formula.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{17}$$

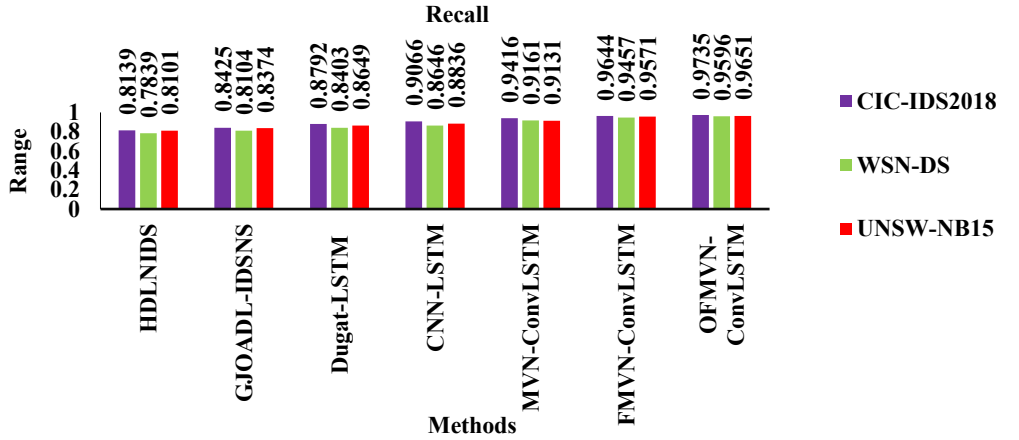


Fig. 5: Comparison of Recall.

Figure 5 shows the recall findings of the suggested and current IDS for various datasets. The analysis shows that OFMVN-ConvLSTM using the CIC-IDS2018 dataset has a much greater recall than HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM by 19.61%, 15.55%, 10.73%, 7.38%, 3.39%, and 0.94%, respectively. Similarly, OFMVN-ConvLSTM performs better than HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM on the WSN-DS dataset by 22.41%, 18.41%, 14.19%, 10.99%, 4.75%, and 1.47%. Furthermore, when compared to HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM, OFMVN-ConvLSTM exhibits recall increases of 19.13%, 15.25%, 11.59%, 9.22%, 5.69%, and 0.84% on the UNSW-NB15 dataset. In comparison to other models already in use for detecting intrusions in network systems, this investigation shows that the suggested OFMVN-ConvLSTM model improves recall performance.

4.2.4. F1-score

According to Eq. (18), the F1-score is the harmonic average of the recall and precision.

$$F1 - Score = 2 * \frac{Precision.Recall}{Precsison+Recall} \quad (18)$$

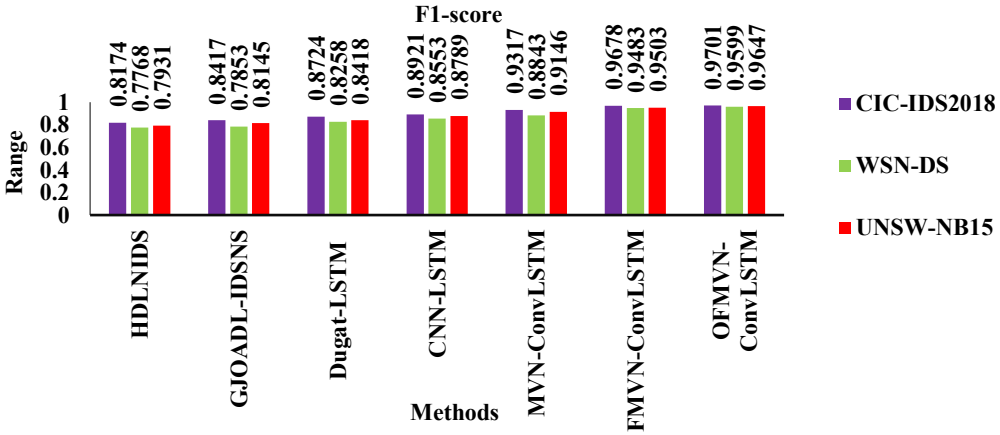


Fig. 6: Comparison of F1-Score.

Figure 6 illustrates the F1-score results for the proposed and existing IDS across various datasets. The analysis indicates that OFMVN-ConvLSTM significantly outperforms HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM, achieving greater F1-scores by percentages of 18.68%, 15.25%, 11.19%, 8.74%, 4.12%, and 0.24%, respectively. On the WSN-DS dataset, OFMVN-ConvLSTM continues to excel, showing improvements over the same models by 23.57%, 22.23%, 16.24%, 12.23%, 8.55%, and 1.22%. Additionally, for the UNSW-NB15 dataset, OFMVN-ConvLSTM demonstrates F1-score enhancements of 21.64%, 18.44%, 14.59%, 9.76%, 5.48%, and 1.51% compared to HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM. This study confirms that the proposed OFMVN-ConvLSTM model enhances recall performance relative to other current models for intrusion detection in network systems.

4.2.5. Training time

It is the total time taken to train the model on a given dataset. Figure 7 illustrates the training time results for the proposed and existing IDS across various datasets. The analysis indicates that OFMVN-ConvLSTM using the CIC-IDS2018 dataset significantly reduces the training time by 41.57%, 35.96%, 25.01%, 21.14%, 15.42%, and 8.14% compared to the HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM, respectively. For the WSN-DS dataset, the training time is 43.84%, 40.94%, 37.87%, 34.03%, 25.43%, and 13.49% lower than that of the same models. For the UNSW-NB15 dataset, the training time is decreased by 33%, 27.74%, 23.91%, 19.49%, 14.83%, and 8.64% compared to the same models. Thus, it is revealed that the OFMVN-ConvLSTM model efficiently reduced the training time compared to the other previous IDS models across different datasets.

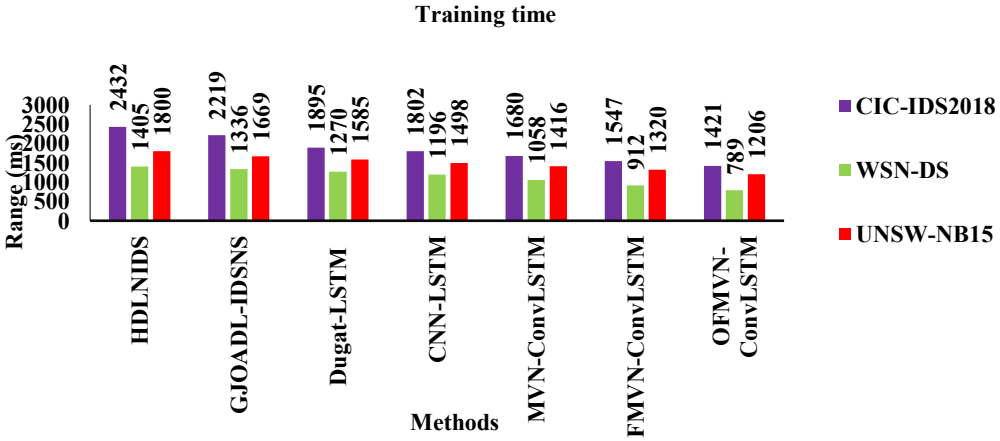


Fig. 7: Comparison of Training Time.

4.2.6. Inference time

It refers to the time taken to process a single data instance. Figure 8 illustrates the inference time results for the proposed and existing IDS across various datasets. The analysis indicates that OFMVN-ConvLSTM, using the CIC-IDS2018 dataset, significantly reduces inference time by 42.28%, 38.26%, 33.64%, 28.28%, 21.11%, and 13.41% compared to HDLNIDS, GJOADL-IDSNS, Dugat-LSTM, CNN-LSTM, MVN-ConvLSTM, and FMVN-ConvLSTM, respectively. For the WSN-DS dataset, the inference time is 57%, 51.14%, 45.57%, 38.57%, 27.12%, and 15.69% lower than that of the same models. For the UNSW-NB15 dataset, the inference time is decreased by 46.3%, 41.41%, 33.33%, 28.4%, 20.55%, and 12.12% compared to the same models. Thus, it is revealed that the OFMVN-ConvLSTM model efficiently reduced the inference time compared to the other previous IDS models across different datasets.

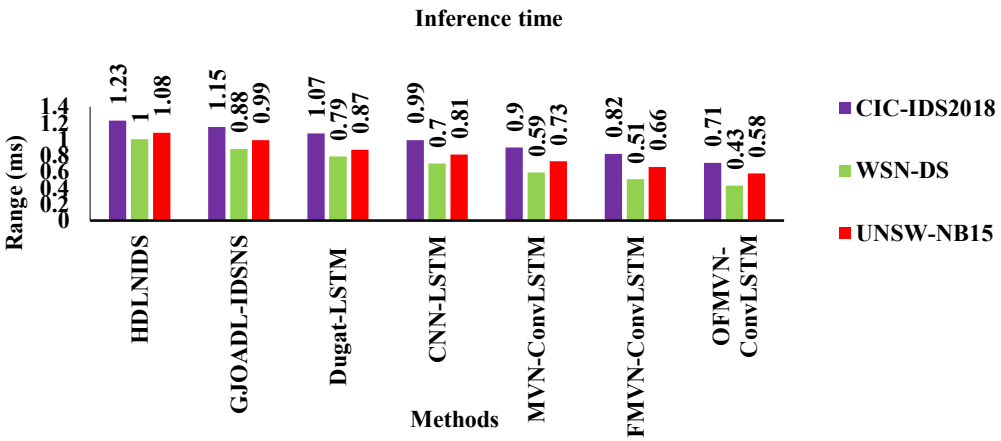


Fig. 8: Comparison of Inference Time.

4.3. Discussion

The better performance of the OFMVN-ConvLSTM model can be explained by the synergistic complementary effect of the multi-valued neutrosophic framework and the DSO algorithm. In comparison to traditional optimizers such as PSO and GA, DSO dynamically resets exploration and exploitation by balancing hyperparameters through adaptive satiety-based learning, which prevents early convergence and results in more stable hyperparameter settings. This process enables the ConvLSTM to learn more vivid spatiotemporal correlations between intrusion data and effectively handle uncertainty about feature relevance. Additionally, DSO reduces overfitting and speeds up convergence by self-adjusting various parameters (including learning rate, dropout rate, and batch size) and, therefore, explains the decreased training and inference time of the model. The good performance of the model in generalization across three different datasets, CIC-IDS2018, WSN-DS, and UNSW-NB15, is an indication that the model is resistant to other forms of traffic and attack patterns.
Nevertheless, as network environments continue to grow in size and diversity, further research and testing are necessary to assess the scalability of the OFMVN-ConvLSTM for high-velocity streaming conditions and large-scale IoT applications. So, future studies can also focus on variant forms of the DSO-based optimization process, such as distributed or federated, thus retaining performance efficiency in a resource-constrained, real-time condition.

5. Conclusion

In this article, the OFMVN-ConvLSTM model is proposed by leveraging the DSO to fine-tune the hyperparameters of ConvLSTM, significantly improving IDS performance and computational efficiency. To find the optimal balance between efficiency and performance, DSO dynamically adjusts important parameters, including optimizer, learning rate, size of batch, dropout rate, and weight decay. It also optimizes parameters like the number of clusters and partitions per batch, improving ConvLSTM's ability to process high-dimensional intrusion data. DSO minimizes overfitting and accelerates convergence, enhancing population diversity and streamlining model complexity. This approach produces a robust IDS capable of identifying network threats with high accuracy and reduced computational overhead. Based on experimental findings, OFMVN-ConvLSTM outperforms other models with accuracy scores of 0.9748, 0.9635, and 0.9685 on the CIC-IDS2018, WSN-DS, and UNSW-NB15 datasets, respectively.
The OFMVN-ConvLSTM model's applicability and resistance can be enhanced through future studies. Testing on larger datasets and evaluating their vulnerability to adversarial and evasion attacks can improve cybersecurity in hostile environments. The hybrid optimization-oriented framework can also be applied to other fields of time-series anomaly detection, such as fault detection in industrial systems and sensor monitoring IoT devices, expanding its advantages in various areas. This will enhance the model's applicability and resistance in complex real-world network settings.

References

[1] Halbouni A, Gunawan TS, Habaebi MH, Halbouni M, Kartiwi M, & Ahmad R (2022), CNN-LSTM: hybrid deep neural network for network intrusion detection system. *IEEE Access* 10, 99837-99849. https://doi.org/10.1109/ACCESS.2022.3206425
[2] Veni C & Sridevi R (2023), Multi-valued neutrosophic convolutional LSTM for intrusion detection, *International Journal of Intelligent Engineering and Systems* 16(5), 364-375. https://doi.org/10.22266/ijies2023.1031.31.
[3] Veni C & Sridevi R (2024), A novel feature prioritization algorithm in multi-valued neutrosophic ConvLSTM for improving intrusion detection, *Journal of Electrical Systems* 20(3), 7203-7213. https://doi.org/10.52783/jes.7391.
[4] Kasongo SM (2023), A deep learning technique for intrusion detection system using a recurrent neural networks based framework, *Computer Communications* 199, 113-125. https://doi.org/10.1016/j.comcom.2022.12.010.
[5] Shobana G & Sai NR (2023), Network intrusion detection using a step-based deep learning approach, *Research Advancements and Innovations in Computing, Communications and Information Technologies* 2796(1), 130002. https://doi.org/10.1063/5.0149139.
[6] Hnamte V & Hussain J (2023), Dependable intrusion detection system using deep convolutional neural network: a novel framework and performance evaluation approach, *Telematics and Informatics Reports* 11, 100077. https://doi.org/10.1016/j.teler.2023.100077.
[7] Mohammadian H, Ghorbani AA & Lashkari AH (2023), A gradient-based approach for adversarial attack on deep learning-based network intrusion detection systems, *Applied Soft Computing* 137, 110173. https://doi.org/10.1016/j.asoc.2023.110173.
[8] Ayantayo A, Kaur A, Kour A, Schmoor X, Shah F, Vickers I, ... & Abdelsamea MM (2023). Network intrusion detection using feature fusion with deep learning, *Journal of Big Data* 10(1), 167. https://doi.org/10.1186/s40537-023-00834-0.
[9] Silivery AK, Kovvur RMR, Solleti R, Kumar LS & Madhu B (2023), A model for multi-attack classification to improve intrusion detection performance using deep learning approaches, *Measurement: Sensors* 30, 100924. https://doi.org/10.1016/j.measen.2023.100924
[10] Qazi EUH, Faheem MH & Zia T (2023), HDLNIDS: hybrid deep-learning-based network intrusion detection system, *Applied Sciences* 13(8), 4921. https://doi.org/10.3390/app13084921.

[11] Turukmane AV & Devendiran R (2024), M-MultiSVM: an efficient feature selection assisted network intrusion detection system using machine learning, *Computers & Security* 137, 103587. https://doi.org/10.1016/j.cose.2023.103587.
[12] Aljehane NO, Mengash HA, Eltahir MM, Alotaibi FA, Aljameel SS, Yafoz A, ... & Assiri M (2024), Golden jackal optimization algorithm with deep learning assisted intrusion detection system for network security, *Alexandria Engineering Journal* 86, 415-424. https://doi.org/10.1016/j.aej.2023.11.078.
[13] Devendiran R & Turukmane AV (2024), Dugat-LSTM: Deep learning based network intrusion detection system using chaotic optimization strategy, *Expert Systems with Applications* 245, 123027. https://doi.org/10.1016/j.eswa.2023.123027.