Current Applied Science and Technology e-ISSN: 2586-9396

Curr. Appl. Sci. Technol. 2025, Vol. 25 (No. 3), e0262276

Research article

Intrusion Detection System: An Ensemble Deep Learning Approach for Cloud Computing Using EBWO

Vinolia Alexander Moudiappa*, Kanya Nataraj and Veeramalai Natarajan Rajavarman

Department of Information Technology, Dr. M.G.R Educational and Research Institute, Chennai, India

Received: 23 February 2024, Revised: 11 July 2024, Accepted: 29 November 2024, Published: 22 January 2025

Abstract

Cloud computing is the industry standard for data storage, sharing, processing, and other services. It experienced numerous security problems as a result of the regular attacks. These security issues are worsened by the variety of attack situations that exist. One of the most established safety measures applied to cloud computing is the intrusion detection system (IDS). An effective security model is necessary for the IDS system, though, to increase cloud security. In this study, we used ensemble categorization methods and a feature selection algorithm to construct an effective IDS for the cloud environment. The proposed BOT-IOT, CSE-CIC-IDS 2018, and Ciciddos datasets were pre-processed, which involved cleaning the data, applying one hot encoding, and normalizing steps. The Enhanced Black Widow Optimization (EBWO) algorithm was employed to choose the most advantageous reduced feature sets from the provided incursion datasets. We used an ensemble of Hierarchical Multi-scale LSTM (HMLSTM) and Darknet Convolutional Neural Network (DNetCNN) to categorize the attacks. The combination of DNetCNN and HMLSTM was used to identify intrusions, effectively classifying attacks, lowering false alarm rates, and increasing detection rates. Simulation research showed that the proposed strategy performed better than the baseline in terms of F-Score, DR, and FPR, as well as accuracy, detection rate, and precision.

Keywords: cloud; intrusion detection system; enhanced black widow optimization algorithm; darknet convolutional neural network; Hierarchical Multi-scale LSTM

1. Introduction

As cloud computing evolves and derives from distributed computing, its use is expanding, and numerous clients can utilize the exact cloud resources by utilizing logical isolation methods (Du et al., 2023). Since destructive network attacks are getting more frequent and advanced, they can increase network traffic. The most important benefit of cloud computing is that it allows organizations to save money by using on-demand services virtually over the internet (Ravi et al., 2022; Hnamte & Hussain, 2023). This benefit outweighs other

*Corresponding author: E-mail: vinolia@dhtepdy.edu.in

https://doi.org/10.55003/cast.2025.262276

advantages like scalability, rapid flexibility, and measurable capabilities. One of the most challenging tasks is protecting cloud assets and the information flow between them because the number of invasions grows daily (Gupta et al., 2022).

The scientific community has developed a wide range of machine learning-based IDS approaches (Alqahtani & Kumar, 2022; Kim& Pak, 2022; Kasongo, 2023). Traditional machine learning techniques help to enhance the categorization of small and low-dimension datasets. However, these algorithms' classification reliability deteriorates when dealing with circumstances demanding significant dimensionality and nonlinearity (Imran et al., 2022). As a result, intrusion detection models have become increasingly necessary to address the effectiveness of categorization problems as AI advances. The problem with deep learning approaches, such as CNN and LSTM, is how well they adapt to high-dimensional and nonlinear input (Thirimanne et al., 2022; Soltani et al., 2023). CNN and LSTM provide nonlinearity solution strategies for modelling nonlinear systems (Thakkar & Lohiya, 2023).

In past works, CNN and LSTM were employed to tackle high-dimensional data issues using the deep learning approach (Rahman et al., 2020; Saba et al., 2022). Automated machine learning (AutoML), a relatively new field of data science and machine learning, is a subset of these disciplines. The adaptability of AutoML makes it useful for machine learning researchers, data scientists, and engineers (Kunang et al., 2021). Research studies show that AutoML can revolutionize ML model construction even without technical expertise or ML proficiency. AutoML architectures produce a code pipeline by selecting a model from input datasets based on machine learning model recommendations (Thakkar & Lohiya, 2021; Azzaoui et al., 2022; Vishwakarma & Kesswani, 2022;). The selection is carried out based on the precision of these ML methods (Mighan & Kahani, 2021). Finding the best-performing model pipeline using manual settings of the model parameters is very difficult; AutoML solves this problem (Wang et al., 2021; Saheed et al., 2022).

We proposed ensemble approaches based on Darknet Convolutional Neural Networks (DNetCNN) and Hierarchical Multiscale LSTM (HMLSTM) to classify the attack categories in this research.

Our research introduces several novel contributions to the field of network intrusion detection, with novelty at its core. First, our preprocessing methodology incorporates state-of-the-art methods which are essential for improving the quality of data for the analysis that follows. We use a powerful Conditional Generative Adversarial Network (CGAN) to enhance minority samples to address the problem of imbalanced datasets and increase the accuracy and robustness of the model. Second, we present the Enhanced Black Widow Optimization (EBWO) algorithm, designed to simplify intrusion detection model training, to maximize learning efficiency and minimize computational complexity. Finally, we present and put into practice two hybrid deep learning architectures specifically created to categorize network intrusion attacks: HMLSTM and DNetCNN. The ensemble goal of these techniques is to increase detection rates, reduce false alarms, and improve attack classification accuracy.

The following are the paper's main contributions:

- We apply a one hot encoding model, cleaning the data, and normalizing the data in the pre-processing step. To increase the minority samples, an effective conditional Generative Adversarial Network (CGAN) is used.
- To reduce the computation complexity and improve the learning algorithm efficiency, we utilize the Enhanced Black Widow optimization (EBWO) algorithm.

- Finally, two hybrid deep learning techniques (HMLSTM and DNetCNN) are used for classifying the network intrusion attacks. So, our approach can efficiently classify the attacks, reduce the number of false alarms, and improve detection rates.
- Several experiments have been run on the Ciciddos, CES-CIC-IDS 2018, and BOT-IOT datasets. The findings from experiments demonstrate that our proposed network was effective and outperformed all previous methods.

The field of study surrounding Network Intrusion Detection System (NIDS) is quite extensive. CNN, RNN, machine learning, and hybrid models are examples of current approaches utilized in IDS. The issues of low detection accuracy and difficulties in identifying specific types of occurrences have been addressed by researchers in the field of IDS using various methods. A few recent, relevant articles are discussed below.

Khan (2021), presented a system for detecting network intrusions that use a CRNN. Employing CRNN and DL-based hybrid recognition, a system is developed for anticipating and classifying destructive cyber-attacks on the network. While CNN employs convolution to capture local information, the RNN gathers temporal characteristics to improve the efficiency and predictions of the IDS in the HCRNNIDS. The suggested method reliably identified malicious attacks up to 97.75% of the time with tenfold cross-validation.

Devan& Khare (2020) introduced the XGBoost-DNN-based classification algorithm for the IDS. With XGBoost, regularization and overfitting issues are resolved. XGBoost allows quicker detection of network intrusions than the current models employed for binary classification of intrusion detection. DNN was used to classify network intrusions. Compared to the earlier approaches, the strategy achieved 97% accuracy on the NSL-KDD dataset.

A NIDS was developed by Liu & Hu (2021) using LightGBM and an adaptive synthetic (ADASYN) oversampling technique. The training data disparity is fixed by employing the ADASYN oversampling approach, which boosts the minority samples. The LightGBM hybrid learning algorithm was applied to significantly reduce the system's time complexity while preserving the precision of detection. On datasets from NSL-KDD, UNSW-NB15, and CICIDS 2017, the suggested approach achieved accuracies up to 92.57%, 89.56%, and 99.91%.

Chiba et al. (2019) combined the Classical Auto Encoder (CAE) technique with a deep neural network for a NIDS. Due to the properties of the presented method, network anomalies can be found in two steps. In the initial phase, feature engineering was done using a CAE. In the second phase, categorization was done using a DNN. The method obtained an accuracy of 91.29% on the UNSW-NB15 dataset.

Babu & Rao (2023) presented an effective approach per an improved conditional generative adversarial network (MCGAN) for solving the unbalanced problem by balancing minority and majority classes. To effectively categorize the multi-class ID, the Bidirectional-Long Short-Term Memory method was eventually included. The proposed approach performed better when measured against current approaches. The overwhelming majority of relevant publications rely on outdated NSL-KDD and KDD datasets. These datasets are of minimal use to a modern IDS. Since the creation of these datasets in 1999, both normal and malicious network traffic have undergone considerable changes, rendering the bulk of the conclusions drawn from them negligible in value.

To solve several shortcomings of previously recommended systems, such as low attack detection for irregular attacks, and incorrect classification of assaults, we provide an ensemble IDS that combines various categorization techniques, namely HMLSTM and

DNetCNN. To assess its performance in identifying network intrusions, we used three standard datasets, which we divided into training and testing datasets. We also compared it to other ML and DL-based techniques.

2. Materials and Methods

IDS have attracted a lot of focus because it can instantly react to intrusions from the inside and outside. A detection-based approach for identifying malicious was conducted and featured intrusion detection that can distinguish between an attack and a non-attack to increase detection precision and decrease false alarm rates.

In the initial stages of preprocessing, the data were cleaned, hot encoded, and normalized. Second, the feature selection process was carried out to prevent the high dimensionality curse. Unneeded features were eliminated using the Enhanced Black Widow Optimization Algorithm (EBWO) based on the feature relevance score. Third, the combined DNetCNN and HMLSTM classifiers are constructed and trained. The assaults are categorized using the ensemble classifier. Three datasets were used to illustrate the utility of the NIDS. The proposed strategy's primary goal is to prevent unauthorized access and data hacking. The study can be helpful to businesses that use a cloud and have to cope with various forms of malicious activity by intruders. The architecture of the proposed methodology is illustrated in Figure 1.

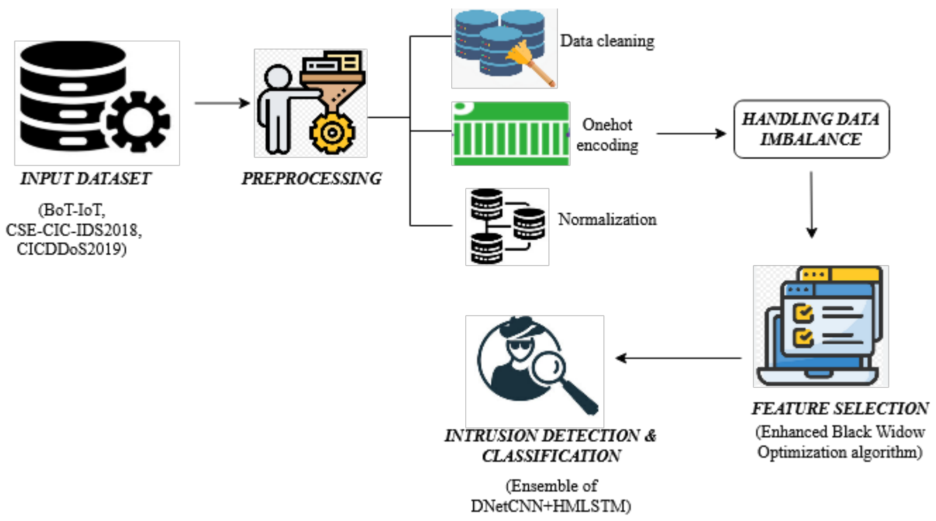
INPUT DATASET
(BoT-IoT, CSE-CIC-IDS2018, CICDDoS2019)
PREPROCESSING
Data cleaning
Onehot encoding
Normalization
HANDLING DATA IMBALANCE
FEATURE SELECTION
(Enhanced Black Widow Optimization algorithm)
INTRUSION DETECTION & CLASSIFICATION
(Ensemble of DNetCNN+HMLSTM)

Figure 1. The overall framework of the proposed methodology

2.1 Problem statement

This research discusses some of the significant research issues that necessitate careful consideration. Using a proper dataset is a challenging issue in the development of an IDS. The primary data sources utilized by existing IDS systems, such as the KDD-99 or NSL-KDD benchmark datasets, are untrustworthy in terms of efficiency findings since they contain old traffic, do not adequately reflect current attack situations, and lack real-time characteristics. This issue was solved by analyzing more recent datasets, such as the

UNSW-NB15, CICIDS-2017, Mawilab, and IoT-23 datasets. IDS utilizes machine learning techniques that are Data-Fit. Even though the data was collected from a single network, a comparable IDS needed to be constructed on a different network. Finding attacks hidden by evasion tactics is another challenging issue for an intrusion detection system. How resistant IDS is to various evasion techniques needs further investigation.

2.2 Data preprocessing

Dataset processing is a crucial technique that is applied to every dataset employed in this work.

2.2.1 Cleaning the data

To deal with missing or damaged data, we looked at entire datasets. We began by identifying all instances of missing data and data that had inadequate quantities, such as -inf, +inf, nan, and so forth, to accomplish this. Because the dataset contained a sizable amount of data, any specimens with incorrect or missing entries were removed.

2.2.2 One-hot processing

The method was employed to translate the dataset's symbolic properties into numerical attributes. It is a practical and attractive encoding method and is a widely utilized approach for addressing the ordinal property's neutralization. Ordinal properties are converted into a binary vector, where one unit is given a value of 1, while the remaining units are given values of 0. A value of 1 in an entity's integer denotes the possibility that there are numbers that could fit the category feature.

2.2.3 Data normalization

To balance the wide variety of data properties throughout the normalization process and speed up the proposed categorization strategy's ability to discover the ideal solution, data scaling was used. To scale the attribute values, we applied the maximum-minimum normalization approach. Equation 1 states that all values for attributes are normalized to fall between a predefined range of (0, 1).

$$\boldsymbol{y'} = \frac{y - y_{min}}{ymin_{max}} \tag{1}$$

While y represents the initial value and y' represents normalized gain. The data collection's minimum and maximum values are denoted as min and max. The values range from 0 to 1. With the help of the minimum and maximum values calculated for every column, the data in the training portion was normalized.

2.3 Imbalanced data handling using CGAN

A dataset imbalance between the CICDDoS2019, BOT-IOT, and CSE-CIC-IDS 2018 datasets resulted in 2 extensive class descriptions for approximately 50% of the dataset. As a result, before categorization, the dataset needed to be balanced. In order to balance the datasets, a conditional Generative Adversarial Network (GCN) based oversampling strategy was used. The GCN architecture, an extension of the GAN architecture, puts a producer G against a discriminator D to determine which can perform better. It is constructed as a function to confuse the discriminator. The discriminator's objective is to distinguish between samples generated by the generator and instances from the provided database. The generator module G, denoted as G: Z→ X, generates noise. Z is a noise space with arbitrary dimension dz. D: X (0, 1) that denotes the discriminative module, and an estimate of the probability that a specimen will emerge from a distribution of data rather than a general distribution G is provided. The CGAN architecture expands the probabilistic method G to incorporate the additional space Y, designating external data from the training data as shown in equation (2):

$$G: Z \times Y \rightarrow X \tag{2}$$

In the same manner as G, the discriminator D is upgraded which is illustrated in equation (3):

$$D: X \times Y \rightarrow (0, 1) \tag{3}$$

The function is written as follows in equation (4) for this two-method MLP with two players:

$$min_G \, max_D \, V\,(D, G) = E_D + E_G \tag{4}$$

While,

$$E_D = E_{x,y\sim p_{data}(x,y)}{}^{(log\,D(x,y))}$$
$$E_G = E_{z\sim p_z(z),y\sim p(y)}{}^{(log(1-D(G(z,y),y)))}$$

The (x, y) X→Y values originate from the data distribution data (x, y), the audio signal $p_z(z)$, and the Y values originate from provisional eigenvectors in the training phase and are determined by the probability density $_{by}$ (y).

2.4 Feature selection

To select the most important features, the Enhanced Black Widow Optimization Algorithm (EBWO) was used in this phase. The FS technique seeks the fewest features that satisfy a given criterion to produce a model for prediction with the highest degree of accuracy. Benefits of feature selection include decreased computational efficiency, enhanced learning performance, the removal of redundant data, and increased generalization and comprehension of the data.

The Enhanced Black Widow Optimization Algorithm (EBWO) was chosen for its adaptive and competitive behavior. It efficiently explores and exploits the search space, making it well-suited for the complex nature of network intrusion detection. EBWO is designed to have faster convergence rates than traditional swarm intelligence algorithms, crucial for real-time intrusion detection where timely responses are essential. Its effective balance between exploration and exploitation reduces the likelihood of getting stuck in local optima, enabling the identification of diverse and sophisticated attack patterns. Additionally, EBWO is flexible and robust, ensuring consistent performance across various datasets and attack scenarios. When compared to other swarm intelligence algorithms, EBWO is expected to demonstrate superior detection accuracy, lower false alarm rates, faster convergence, and better scalability. These advantages stem from EBWO's adaptive nature and efficient handling of large datasets. Consequently, EBWO's robustness and versatility make it a reliable choice for enhancing network intrusion detection systems, outperforming other algorithms in key performance metrics.

2.4.1 Background of black widow spider

The spider, a small-sized member of the Orygiidae family, is most commonly seen in European countries that border the Mediterranean Sea. Female spiders spend most of their time on webs doing activities other than eating, mating, or giving birth. The four main steps of the BWO algorithm are mutation, reproduction, cannibalism, and population initialization.

1) Initialization

An individual spider is represented as $W_{N\times D} = (X_1, X_2, \cdots, X_N)$ with *N* widows $X_1, X_2, \dots, X_N$. The dimension of an optimization issue is represented by D. In the individual, *an i*-th widow is represented by$X_i = (x_{i,1}, x_{i,2}, \cdots, x_{i,D})(1 \leq i \leq N)$. Every individual prospect is depicted in equation (5).

$$x_{i,j} = l_j + rand(0,1) \cdot (u_j - l_j), 1 \leq j \leq D, \qquad (5)$$

The upper and lower bound of the variables are represented by$U = (u_1, u_2, \cdots, u_D), L = (l_1, l_2, \cdots, l_D)$.

2) Procreation

The special way that black widows mate allows them to create a new generation. To begin mating, a population of spiders designated as the mother and father spiders is randomly chosen, and they are then paired up based on procreating rate (Pp). By using equation (6), the progeny is created.

$$\begin{cases} Y_i = \alpha X_i + (1-\alpha) X_j \\ Y_j = \alpha X_j + (1-\alpha) X_i' \end{cases} \qquad (6)$$

Mother and father spiders are identified as X_i and X_j. By mating, Y_i and Y_j are produced. Additionally, the solution vector is a D-dimensional array that contains random numbers.

3**) Cannibalism**

Three types of cannibalism are present in this step: sexual cannibalism, cannibalism among siblings, and cannibalism among offspring and mothers. Excellent spiders can be kept alive by removing the weak ones.
Sexual cannibalism: Female black widows eat their partners either during or shortly after mating.
Sibling cannibalism: The spiders struggle to locate or identify enemies, and their siblings are often eaten by more dominant spiders. According to biological competition theories, a spider's strength is determined by its ability to survive and reproduce, which directly correlates with its fitness. The number of survivors is calculated using a cannibalism rating (CR).

Cannibalism between offspring and mother: Spiders might even consume the mothers of some offsprings who are incredibly powerful. In other words, if parents create a solution with a high fitness value, the result will swap out its mother and enter the following generation. Figure 2 displays a mutation operator.

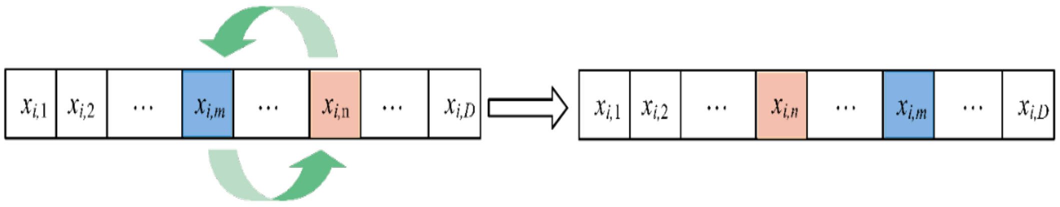

Figure 2. Mutation method

4) Mutation

The mutation rate (Pm), a known constant in this stage, determines how many members of population will mutate.

5) Modified black widow optimization algorithm

By mimicking the behavior of the black widow, the spider individuals in the BWO algorithm look for the best people within the search space. Its simplicity in use and comprehension is a benefit. The original BWO algorithm's convergence rate and precision, meanwhile, are still insufficient for dealing with actual optimization issues. In this work, an enhanced BWO called MBWO was proposed. However, three key parameters are adaptively estimated to reduce their impact on the overall process.

6) Strategy for selection

In the BWO, when the female is anxious to mate, a male widow is chosen at random. This means that the development of new solutions happens independently of the design of the parents. The offspring acquires too many genes that are identical to those of its parents, which is akin to the behavior of inbreeding. Numerous subpar solutions are generated in this situation, resulting in slow convergence and low precision. A fresh approach is therefore suggested to address the drawback. Every female's weight is used in the method to determine the pheromone concentration using equation (7).

$$w_i = \frac{f(X_i)-f_{worst}}{f_{best}-f_{worst}}. \tag{7}$$

While the values of f_{orest} and $_{best}$ correspond to the population's worst and best fitness, respectively. Priority in selecting the male spider belongs to the female with more weight. The Euclidean distance as well as the similarity among spiders, can be calculated using equation (8).

$$d_{ij} = \left(\sum_{k=1}^{D}(x_{i,k} - x_{j,k})^2\right)^{1/2}. \tag{8}$$

The least comparable male in a population of spiders is chosen for mating to prevent inbreeding. Once a person is chosen, they cannot be chosen again.

7) Mutation operator

When an optimization model has multiple local optimal solutions, the BWO is more likely to select the local solution than the best outcome due to a basic mutation mechanism called premature convergence. Spiders are chosen at random to experience the following modifications based on the mutation rate (Pm), as indicated by equation (9).

$$X_{new}^{t+1} = X_{best}^{t} + S \cdot (X_{r1}^{t} - X_{r2}^{t}), \quad (9)$$

Among the current population, where X_{r1}^{t}, X_{r2}^{t} are randomly chosen. X_{best}^{t} represents the best individual. $S = t/T$ is the mutation operator that gets more powerful as iterations. Population diversity is estimated here using equation (10).

$$Diversity = \frac{1}{|N|\cdot|S|}\sum_{i=1}^{N}\sqrt{\sum_{j=1}^{D}(x_{ij} - \bar{x}_j)^2}, \quad (10)$$

Where population size and dimension are denoted by N and D, respectively. The search space size is denoted by S., $\bar{x}_j$ representing the population's jth dimension's average value. Finding the ideal solution becomes more challenging because all of the chosen functions.

8) Adaptive parameters

Three characteristics in the original BWO are crucial for getting better results. In the enhanced black widow (EBW) applied for feature selection, the procreation rate (Pp) is set to 0.6, the cannibalism rate (CR) to 0.44, and the mutation rate (Pm) to 0.4 to balance the different phases of the optimization process. The procreation rate of 0.6 ensures that 60% of the population will be new offspring, facilitating adequate exploration of the search space while preserving the integrity of existing solutions. The cannibalism rate of 0.44 indicates that 44% of the population is eliminated, promoting the survival of stronger candidates and maintaining population diversity. The mutation rate of 0.4 signifies that 40% of the offspring will undergo mutations, introducing genetic diversity and aiding in the exploration of new areas in the search space, thereby avoiding local optima. These specific parameter values were chosen to achieve an optimal balance between exploration and exploitation, ensuring that the EBWO algorithm effectively identifies the most relevant features for classifying network intrusion attacks. Whether these values were determined through empirical experimentation or referenced from existing literature, they were selected to enhance the algorithm's overall performance in terms of convergence speed, accuracy, and robustness. This strategic use of EBWO for feature selection significantly contributes to the improvement of detection rates, reduction of false alarms, and efficiency of the learning algorithm in handling network intrusion detection.

In differential evolution (DE), the control factors were directly linked to people and changed to produce better people. Additionally, artificial neural networks were used to choose the ideal parameters. Therefore, it was a challenge to think about how to alter these parameters to increase the algorithm's success. To balance exploitation and exploration and then make the computation simpler, parameters had to be adjusted as the number of iterations increased rather than being set to constant values.

2.4.2 Computational complexity

Regarding the enhanced algorithm upgrades, the new selection strategy's primary factor in increasing the algorithm's temporal complexity is the separation between mother spiders and candidate individuals. The distance matrix indicated that M_{dist} (which is employed to calculate the distance among both gender spiders in the enhanced choosing technique), N, D, and T together determine the computational complexity of EBWO. Equations (11) and (12) illustrate the computational complexity formula.

$$O(EBWO) = O(T(O(M_{dist})+)O(procreate + cannibalis) + O(mutation)) \quad (11)$$

$$= O(T(N^2 + ND + N)) \quad (12)$$

2.5 Classification

A new classifier ensemble that was a combination of DNetCNN and HMLSTM was created to generate greater performance results and attain the significant efficiency of IDS. In the research on classifying attack categories, the choice of models like HMLSTM and DNetCNN was driven by their specific strengths in handling the complexities of attack data. HMLSTM was selected for its capability to capture hierarchical temporal dependencies, essential for identifying multiscale patterns in attacks that evolve over different time scales. This model efficiently processes data at various resolutions, balancing memory usage while effectively discerning both short-term fluctuations and long-term trends in attack behaviours. DNetCNN, on the other hand, excels in extracting spatial features from structured data, making it particularly suitable for tasks where identifying intricate spatial patterns in network traffic or other structured attack data is crucial. In contrast, Recurrent Neural Networks (RNNs) combined with LSTM were not chosen due to their limitations in hierarchical processing and spatial feature extraction. While RNNs with LSTM are adept at capturing temporal dependencies, they lack inherent support for multiscale temporal analysis and may face challenges with computational efficiency and spatial feature learning compared to more specialized models like HMLSTM and DNetCNN. Therefore, the selection of HMLSTM and DNetCNN was based on their suitability for handling the specific characteristics and challenges posed by attack data classification in the research context.

2.5.1 Darknet Convolutional Neural Network (DNetCNN)

Artificial neural networks and deep learning are crucial components of the present research's regression and categorization techniques. Convolutional neural networks were shown to be the most effective classifiers among the various deep learning models. Although the data used was clean and free of noise, the traditional convolutional neural network (CNN) classified the features as valuable. However, if the quality of the data was poor, the accuracy of the CNN's classification was negatively impacted. To increase the categorization accuracy of IDS, the DNetCNN framework was included as the initial layer of CNN. Darknet-19 is a classifier based on a deep learning model that uses the YOLO (You Only Look Once) detection mechanism to identify items in real time. The activation function is responsible for activating five pooling layers, 19 convolutional layers, and the

Darknet structure as a whole. The function of sigmoid activation was utilized for binary categorization. Softmax activation was employed for multi-categorization. Employing equation (13), a 2D convolutional process was carried out for the input picture X and kernel K.

$$C(X,K)_{(i,j)} = \sum_r \sum_c K(r,c) \times X \times (i-r, j-c) \tag{13}$$

The input matrix for K is parameterized in steps. Since the testing dataset for IDS involves binary categorization, equation (14) is applied.

$$sigmoid(h) = \frac{1}{1+e^{-Xi}} \tag{14}$$

Sixteen convolution layers were present in the proposed DNetCNN. Every Darknet layer had a convolutional layer for convolution and activation operations. The same three sequential shapes were applied every four convolution layers. The input data was standardized via the normal functioning of the convolution layers, which also helped reduce training time. The neuron will not deactivate due to the activation function. Max pooling was performed using a 2x2 filter in the pooling layer, maximizing the area covered by the filters. In the Darknet model, the convolution layers used filters of varying sizes: 8, 16, 32, and 64. The initial layer was a Darknet layer with a 3x8 filter. Following that, DN, pool, and CNN were the subsequent tiers. A 256-valued filter was used in the final convolution layer.

2.5.2 Hierarchical Multi-scale LSTM (HMLSTM)

LSTM neural networks are widely used to retrieve the time-domain aspects of time-series data. In this study, an upgraded LSTM known as HMLSTM was employed for this. There are several different iterations of the HMLSTM model. The HMLSTM was modified in this study to meet the criteria of the IDS model. The proposed approach features an introduced parameterized boundary detector that generates binary output values for every layer to learn the condition of termination and produce the temporal features. This enables layer ℓ feature maps as input from all preceding layers and generates a feature map concatenation. Using this method, the LSTM model's learning property for spatial features is enhanced, resulting in a classifier that is more effective at detecting intrusions. Gates and the candidate values are represented as follows. These common functions are enhanced by the border detector variable (z_t) which is shown in equation (15)

$$\begin{bmatrix} i_t \\ f_t \\ u_t \\ o_t \\ z_t \end{bmatrix} = Wx_t + Uh^1_{t-1} + z_{t-1}Vh^2_{t-1} + b \tag{15}$$

In the HMLSTM framework, an upgrade procedure is performed at every level, as described by equation (16).

$$h^l_t, c^l_t, z^l_t = f^l_{HMLSTM}(c^l_{t-1}, h^l_{t-1}, h^{l-1}_t, h^{l+1}_{t-1}, z^l_{t-1}, z^{l-1}_t) \tag{16}$$

The function f_{HMLSTM}^{l}represents the forget gate of HMLSTM; the two boundary states z_t^{l-1} are used for this determination. The upgraded cell states are expressed as per equation (17)

$$c_t^l = \begin{cases} f_t^l \Theta c_{t-1}^l + i_t^l \Theta g_t^l if\ z_{t-1}^l = 0 \quad and \quad z_t^{l-1} = 1(Update) \\ c_{t-1}^l \quad if\ z_{t-1}^l = 0 \quad and \quad z_t^{l-1} = 0(Copy) \\ i_t^l \Theta g_t^l \quad if\ z_{t-1}^l = 1(Flush) \end{cases} \tag{17}$$

In this case, g stands for the vector of cell proposal, in contrast to an ordinary LSTM.

If the boundary is discovered at the bottom layerz_t^{l-1} but was absent in the preceding time step, an upgrade operation is carried out to rectify the layer l summary representation. The HMLSTM gating operations in Figure 3 illustrates the information flow between the model's low, middle, and high hierarchical layers. Information flow throughout and between layers is controlled by input, forget, and output gates. The model is able to analyze sequences at different levels of abstraction because boundary detectors determine when to update or maintain states.

Equation (18) illustrates the HM gate operation. The expression can be updated using the equations (19), (20) and (21).

As illustrated in Figure 4, one can obtain an HMLSTM learning temporal characteristic's output module with three layers.

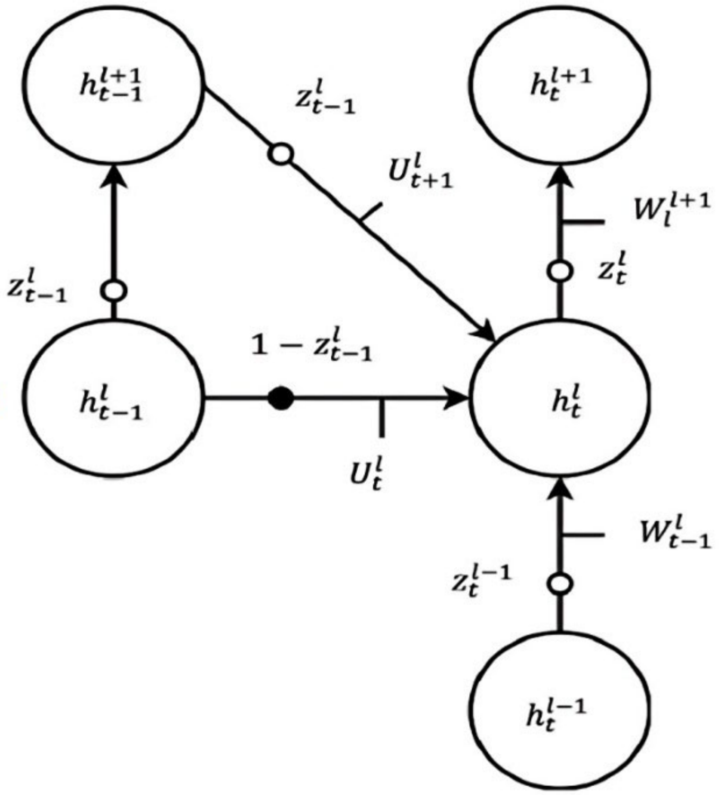

Figure 3. HMLSTM model's Gating operation

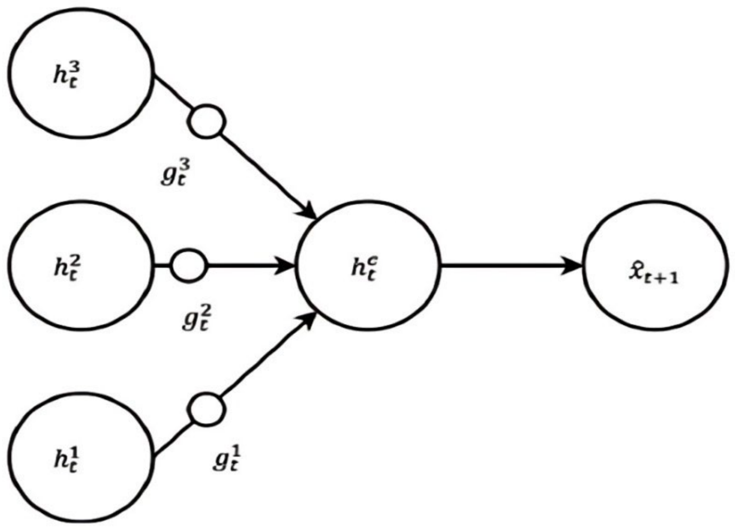

Figure 4. Output module of HMLSTM

$$\begin{bmatrix} f_t^l \\ i_t^l \\ o_t^l \\ g_t^l \\ \tilde{z}_t^l \end{bmatrix} = \begin{bmatrix} sigm & \\ sigm & \\ sigm & ()_{slice}{}_t^{recuurent(l)}{}_t^{top-down(l)}{}_t^{bottom-up(l)} \\ tanh & \\ hardsigm & \end{bmatrix} \quad (18)$$

Here,

$$s_t^{recuurent(l)} = U_l^l h_{t-1}^l \quad (19)$$

$$s_t^{top-down(l)} = z_{t-1}^l U_{l+1}^l h_{t-1}^{l+1} \quad (20)$$

$$s_t^{bottom-up(l)} = z_t^{l-1} W_{l-1}^{ll-1} ht \quad (21)$$

To identify the binary boundary state z_t^l, the equation (22) illustrates that,

$$z_t^l = f_{bound}(\tilde{z}_t^l) \quad (22)$$

It is possible to represent it using the deterministic step function is possible and is expressed in equation (23).

$$z_t^l = \begin{cases} 1 if\ \tilde{z}_t^l > 0.5 \\ 0 otherwise \end{cases} \quad (23)$$

3. Result and Discussions

This study validates three datasets to determine the effectiveness of our proposed method. Two groups of data samples are created, one of which is utilized as the training dataset to create a classifier. The classifier is evaluated using the testing dataset in the second stage. We carried out two trials to evaluate how well the model worked. In the first investigation, multiple categorizations were used, and in the second, it was contrasted with current techniques.

3.1 Experimental setup

Using an Intel CoreTM i5 processor with 8 GB of RAM installed, the proposed approach was carried out employing the Python programming language and Scikit-learn machine learning tools.

3.2 Dataset description

The NSL-KDD dataset has been widely used for IDS, although it contains obsolete traffic and may not adequately reflect modern attack scenarios or traffic trends. Real-time functions are also lacking. In this study, the most recent benchmark datasets, CSE-CIC-IDS2018, BOT-IOT, and Cicddos2019, are used.

CSE-CIC-IDS2018 (Dataset 1): a dataset made public in 2018 as part of a joint project. The victim network comprised a further server room and five other organizational departments. The harmless packets were created by network events employing the impersonal actions of users. One or more computers outside of the target network ran the malware scenarios. The original dataset had 75 features extracted using the CICFlowMeter-V3 initiative.

BoT-IoT (Dataset 2): A network environment including both regular and botnet traffic was formed in 2018. For the generation of non-IoT and IoT traffic, the Ostinato and Node-red tools were used. The Argus program was employed to retrieve the dataset's original 42 features from the collected 69.3GB of pcap files.

CICDDoS2019 (Dataset 3): This study makes use of the CICDoS2019 dataset, which has been widely used for DDoS attack detection and classification. The collection contains a substantial number of samples from recent, actual DDoS attacks and benign instances. Figure 5 displays the proposed dataset's class distribution.

3.3 Performance measures

Based on the findings of F-measure, precision, recall, and accuracy, the performance of every category is assessed. Four crucial states of the confusion matrix must be measured.

- True Positive (TP): This shows that the approach is reliable, consistent, and capable of foreseeing successful results.
- False negative (FN): Inaccurate prediction is a feature of FN. It properly predicts unfavorable outcomes but recognizes malicious incidents with certainty as natural occurrences.
- False positive (FP): Although the number of attacks reported is typical, the model forecasts favorable results.
- True negative (TN): TN classifies incidents that are well-tracked as attacks and foresees unfavorable outcomes.

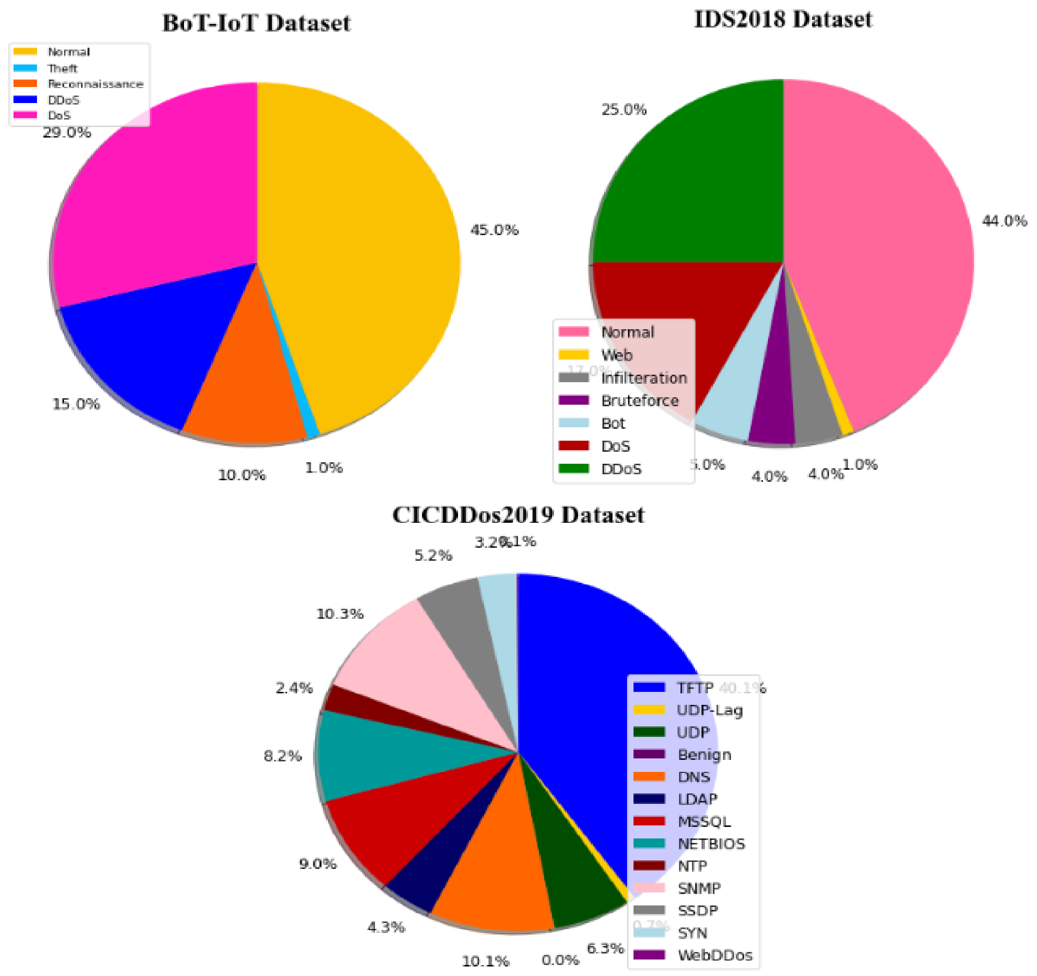

Figure 5. Class distribution of proposed datasets

Accuracy (AC): A model's prediction accuracy is determined by calculating the accuracy rate, which is the proportion of correct predictions throughout the entire data set. The accuracy formula is displayed in equation (24).

$$AC = \frac{TN+TP}{FP+TP+FN+TN} \tag{24}$$

Detection rate (DR): The DR, also called the recall rate, is the proportion of initially authentic assault instances that were identified as assaults within the dataset. This indicator is a key component that shows how well the NIDS can identify attacks. The DR formula is displayed in equation (25).

$$DR = \frac{TP}{FN+TP} \tag{25}$$

Precision (PR): The percentage of anticipated assaults records is known as the precision. The PR formula is displayed in equation (26).

$$PR = \frac{TP}{TP+FP} \tag{26}$$

False Alarm Rate (FAR): The percentage of the sample's normal traffic that is considered to be an assault is known as the FAR. The FAR formula is displayed in equation (27).

$$FAR = \frac{FP}{FP+TN} \tag{27}$$

F1-score: It provides an effective evaluation by measuring the precision correlation coefficient and recall limit. The F1-score formula is displayed in equation (28).

$$F1 = 2 * \frac{RC+PR}{RC+PR} \tag{28}$$

The efficiency of intrusion detection systems is thoroughly assessed by combining these metrics. They shed light on multiple aspects of system efficiency, including the system's capacity to consistently maintain overall classification accuracy (AC), prevent false alarms (FAR), and detect intrusions accurately (DR). When both high recall and precision are required, the F1 Score is especially helpful in ensuring a fair assessment of the system's efficacy.

3.3.1 Evaluation of CSE-CIC-IDS2018 dataset

This portion includes a performance summary of the dataset 1. The proposed methodology is contrasted with existing methods, including LR, XGB, DT, and HCRNN. The comparison of the effectiveness of suggested and existing techniques is shown in Table 1.

The performance of our proposed strategy is strong when compared to other machine learning methods. The dataset from CSE-CIC-IDS2018 is shown in Figure 6. We evaluated the performance of our proposed approach using several metrics, comparing it to prior models: XGBoost (XGB), Decision Trees (DT), Logistic Regression (LR), and Hierarchical Convolutional Recurrent Neural Network (HCRNN). Though displaying different false alarm rates (FAR), Decision Trees, XGBoost, and Logistic Regression showed good performance in terms of precision (87.33% to 78.01%), recall (88.5% to 80.1%), and F1-score (87.9% to 79.01%). With a significantly lower FAR (2.5%) and significantly higher precision (96.33%), recall (97.12%), and F1-score (97.6%), the Hierarchical Convolutional Recurrent Neural Network (HCRNN) outperformed these conventional models. The results of our suggested technique were 99.02% precision, 98.92% recall, 99% F1-Score, and 99.13% DR. The current approach, HCRNN, achieved a similar second-best performance with 96.33% precision, 97.12% recall, 97.6% F1-Score, and 97.86% DR. These outcomes highlight the proposed approach's efficacy in reducing false alarms and improving intrusion detection accuracy, indicating its potential for reliable implementation in actual cloud computing settings.

Table 1. Evaluation performance on the dataset 1

Approaches	Precision (%)	F1-Score (%)	Recall (%)	DR (%)	FAR
DT	87.33	87.9	88.5	88.16	7.8
XGB	84.5	83.9	83.4	83.12	9.13
LR	78.01	79.01	80.1	80	11.50
HCRNN	96.33	97.6	97.12	97.86	2.5
Proposed	99.02	99	98.92	99.13	1.3

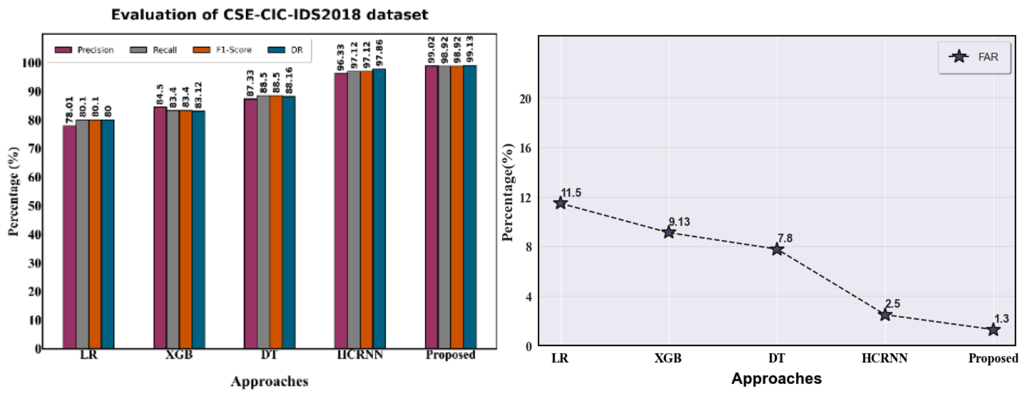

Figure 6. Performance comparison using the CSE-CIC-IDS2018 dataset

Table 2 and Figure 7 compare the suggested DL techniques with currently used approaches. The performance metrics of different approaches in the context of intrusion detection are compiled in the Table. Accuracy and False Alarm Rate (FAR) were used to assess each method, which included DNN (Deep Neural Network), CNN (Convolutional Neural Network), DBN (Deep Belief Network), and a proposed approach. With a 96.2% accuracy rate and an 8.6% false alarm rate, LSTM performed well in accurately classifying instances with a moderate false alarm rate. DNN attains a 90.25% accuracy rate. With a FAR of 9.8% and an accuracy of 95%, DBN exhibits strong performance with somewhat increased false alarms. CNN's accuracy rate was 96%. By contrast, the proposed approach achieved the highest accuracy of 99.12% and the lowest FAR of 1.3%, outperforming all other methods. These findings demonstrate the proposed approach's superior efficacy in minimizing false alarms and achieving high accuracy, indicating its potential to improve intrusion detection systems in real-world applications.

Table 3 provides the multiple categorizations of the CSE-CIC-IDS2018 dataset. The accuracy rate for the infiltration, DoS, and BOT classes was greater than 99%. Values of 1.89% and 1.76% were obtained for FPR and FNR, respectively, with SQL injection.

Table 2. Comparison of existing approaches with proposed approach

Methods	Accuracy	FAR
LSTM	96.2%	8.6
DNN	90.25%	-
DBN	95%	9.8
CNN	96%	-
Proposed	99.12%	1.3

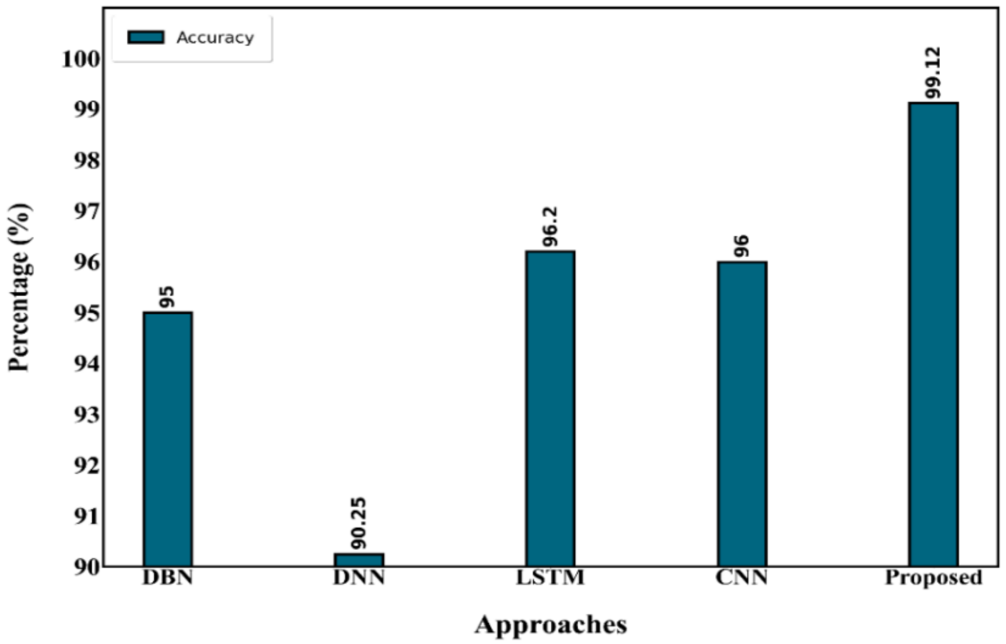

Figure 7. Accuracy comparison on the dataset 1

Table 3. Multi-categorization performances on the CSE-CIC-IDS2018 dataset

Class	Accuracy (%)	FPR (%)	FNR (%)
Benign	98.76±0.05	2.32±0.01	2.84±0.04
DDos	96.15±0.01	4.03±0.02	4.06±0.01
Bot	99.62±0.03	2.68±0.03	2.47±0.03
Brute Force	97.89±0.07	3.45±0.05	2.17±0.05
DoS	99.71±0.01	2.45±0.01	3.10±0.02
Infiltration	99.56±0.05	2.03±0.05	2.13±0.03
SQL injection	98.72±0.04	1.89±0.04	1.76±0.05

3.3.2 Evaluation of BOT-IOT dataset

Here, we provide a summary of the BOT-IOT dataset's results. The results of the multiple classifications on the BOT-IOT dataset are shown in Table 4. Figure 8 compares the performance of each class using the BOT-IoT dataset. Based on the results, the theft class had higher performance compared to other classes.

Table 4. Multi-categorization performances on the BOT-IOT dataset

Class	Accuracy (%)	FPR (%)	FNR (%)
Normal	99.03	2.36	3.69
DoS	98.56	3.01	2.86
DDoS	99.13	2.59	4.06
Reconnaissance	97.56	2.16	1.98
Theft	99.09	1.74	2.11

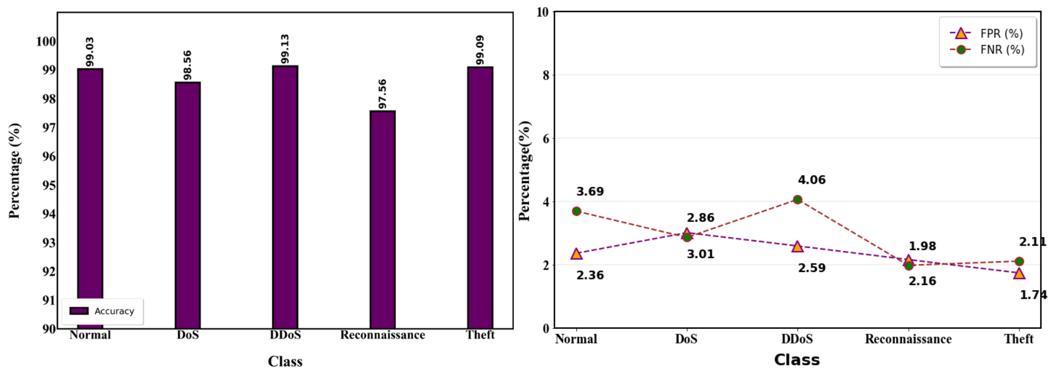

Figure 8. Class-wise performance comparison on the BOT-IOT dataset

3.3.3 Evaluation of Ciciddos2019

To evaluate the efficacy of the proposed method, multiple tests were run on the dataset 3. Table 5 displays the suggested method's results for multi-class classification.

Attack-wise performance comparison on the Ciciddos2019 dataset is shown in Figure 9. For most classes, accuracy rates are greater than 99%.

Table 5. Multi-class categorization on the Cicddos2019 dataset

Attack types	Accuracy	FNR	FPR
NTP	98.89	2.17	1.15
DNS	99.08	3.04	2.24
LDAP	99.06	1.07	0.84
MSSQL	98.92	0.87	1.79
NetBIOS	99.01	1.28	0.63
SNMP	97.65	2.14	0.72
SSDP	98.73	0.99	1.51
UDP	96.26	0.83	0.96
UDP-Lag	98.43	0.71	1.64
WebDDoS	96.64	1.06	0.87
SYN	99.77	2.29	2.31
TFTP	94.43	3.02	0.49

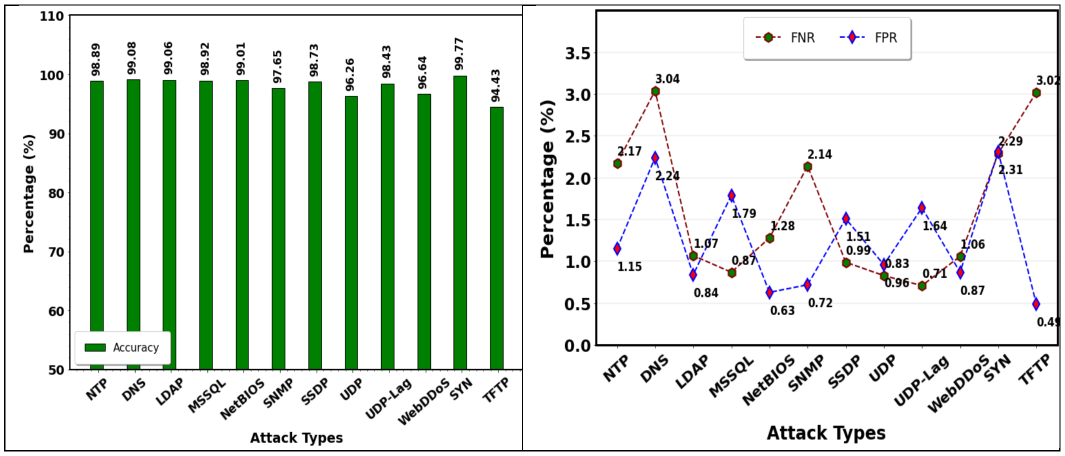

Figure 9. Attack-wise performance comparison on the Ciciddos2019 dataset

The differentiation of the suggested model with prior approaches on dataset 3 is shown in Table 6 and Figure 10. With a remarkable 97% accuracy rate, the Extended Decision Tree led, bolstered by strong precision (99%) and recall (97%). With 98.18% accuracy, bi-directional LSTM came in second, with 97.93% precision and 99.84% recall. The accuracy of the Convolutional Neural Network (CNN) was 95.4%, and its precision-recall scores were balanced (93.3% and 92.4%, respectively). With an accuracy of 92.5%, the Multi-layer Perceptron (MLP) exhibited good overall performance; however, its precision was relatively low at 84.4%. The robust feature representation capabilities of autoencoder-based approaches, AE+Regression and AE+MLP, were demonstrated with accuracy levels of 88.39% and 98.34%, respectively. With remarkable results in all categories: accuracy of 99.32%, 99.06% precision, 99.03% recall, and 98.99% F1-score, the suggested method surpassed the challenge and demonstrated its efficacy for precise and dependable classification tasks.

Table 6. Performance of proposed model vs prior approaches on the Cicddos2019 dataset

Approaches	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Random forest	-	77	56	62
Extended DT	97	99	97	97.8
Extended NB	96.25	96	96	96
Bi-LSTM	98.18	97.93	99.84	-
CNN	95.4	93.3	92.4	92.8
MLP	92.5	84.4	94.2	89
AE+Regression	88.39	85.44	95.95	90.4
AE+MLP	98.34	97.91	98.48	98.18
Proposed	99.32	99.06	99.03	98.99

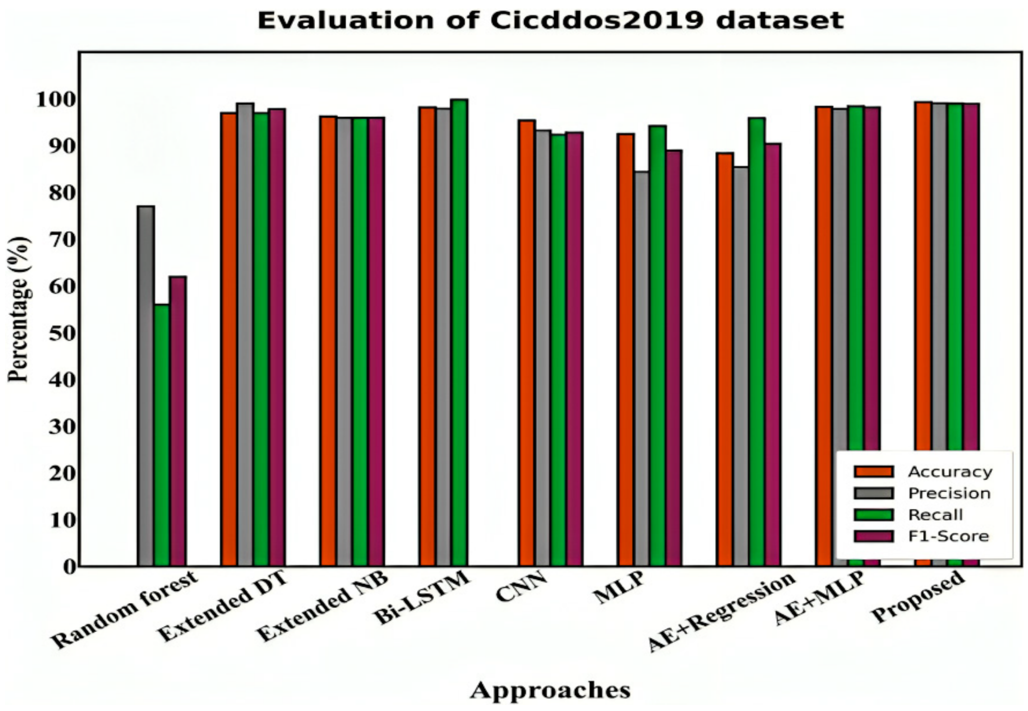

Figure 10. Performance comparison of proposed approach vs existing approaches

3.3.4 Evaluation of overall performances

Using various datasets, the proposed method against the current methods was evaluated. The proposed method is assessed in comparison to similar past attempts, including SVM, DT, RF, and DNN.

The assessment of overall effectiveness is displayed in Table 7 and Figure 11. Table 7 presents a comparison of the performance metrics of several approaches to classification including the SVM, DT, RF, DNN, and the proposed approach. With corresponding precision and recall of 67.50% and 69.85% and an F-score of 56.36%, SVM attained a modest accuracy of 65.95%. With an accuracy of 77.99%, DT outperformed SVM, exhibiting balanced recall (75.59%) and precision (75.37%), yielding an F-score of 71%. With an accuracy of 81.43% and a recall of 74.76%, RF improved efficiency even more and yielded an F-score of 68.40% while retaining high precision (80.10%). This underscored its capacity to effectively handle intricate patterns. The highest accuracy of 99.16%, precision of 98.87%, recall of 99.03%, and F-score of 98.65% were achieved by the proposed approach, which surpassed all other approaches. These findings highlight the proposed approach's superior performance and robustness in precisely classifying instances while preserving high precision and recall, highlighting its potential for applications that require effective and dependable classification.

Table 7. Overall evaluation performance

Approaches	Accuracy (%)	Precision (%)	Recall (%)	F-Score (%)
SVM	65.95	67.50	69.85	56.36
DT	77.99	75.37	75.59	71
RF	81.43	80.10	74.76	68.40
DNN	95.53	94.65	91.92	92.43
Proposed	99.16	98.87	99.03	98.65

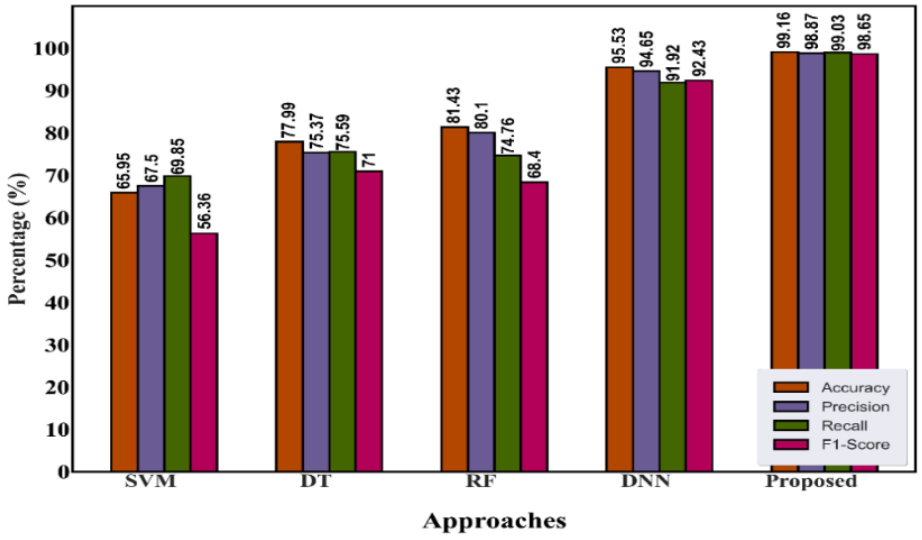

Figure 11. Overall performance comparison

In the present paper, we employed the Nemenyi post-hoc test for post-hoc analysis. The critical value that represents the average sequence variance is determined by the Nemenyi test in the following manner:

$$CD = q_{\alpha}\sqrt{\frac{R(R+1)}{6N}}$$

where qα denotes the Tukey distribution's critical value. Based on computation, 7.822 is the threshold CD for α = 0.05.

Table 8 illustrates that while the proposed approach differed significantly from the other algorithms, algorithms like RF, LightGBM, and Adaboost performed fairly close to each other in terms of accuracy. Table 9 shows that while the average ordinal variance of this algorithm and other methods was high, the effectiveness of RF, GBDT, and LightGBM was comparatively close to the proposed in terms of FAR. When examining these two indicators alone, it is evident that the proposed method has comparatively more benefits even though the efficiency of GBDT and LightGBM was comparable to that of the proposed method.

Table 8. Statistical test for accuracy

	DT	Adaboost	RF	NB	SVM	KNN	LR	ANN	GBDT	LightGBM
Adaboost	3.45	-	-	-	-	-	-	-	-	-
RF	4.21	1.68	-	-	-	-	-	-	-	-
NB	2.68	0	1.53	-	-	-	-	-	-	-
SVM	5.04	3.97	1.50	2.47	-	-	-		-	
KNN	0.96	2.31	2.64	3.17	1.52	-	-	-	-	-
LR	2.87	1.34	4.87	5.23	6.01	4.01	-	-	-	-
ANN	3.68	7.67	2.57	3.61	3.17	2.00	1.99	-	-	-
GBDT	2.67	2.50	1.34	5.06	3.88	7.03	0.67	4.72	-	-
LightGBM	3.66	7.61	3.81	4.32	5.06	6.03	2.67	3.67	0.84	-
Proposed	4.86	3.21	0.93	8.55	7.63	5.02	6.21	2.65	1.16	4.86

Table 9. Statistical test for FAR

	DT	Adaboost	RF	NB	SVM	KNN	LR	ANN	GBDT	LightGBM
Adaboost	4.12	-	-	-	-	-	-	-	-	-
RF	4.35	0.87	-	-	-	-	-	-	-	-
NB	1.94	1.05	2.51	-	-	-	-	-	-	-
SVM	3.64	1.00	1.36	0.00	-	-	-		-	
KNN	1.23	2.41	5.66	0.22	1.67	-	-	-	-	-
LR	2.67	3.80	5.17	5.83	2.22	1.65	-	-	-	-
ANN	3.09	4.83	2.33	2.50	3.16	2.33	1.56	-	-	-
GBDT	0.67	1.00	1.17	4.85	3.81	0.57	2.87	3.62	-	-
LightGBM	0.91	3.35	4.09	3.64	2.33	4.55	0.89	2.19	3.65	-
Proposed	7.02	4.83	6.01	7.32	2.18	2.17	4.85	3.47	2.17	3.89

3.3.5 Evaluation of training and testing

The iteration steps are supplied along with a visual that shows the accuracy and loss value for IDS categorization. Figures 12, 13, and 14 illustrate the beneficial effect that the study's suggested technique had on convergence. Training and testing sessions were held for the proposed dataset. The training phase of the experiment utilizes 75% of the data, while the testing phase uses the remaining 25%. Training the suggested model involved 200 iterations. A learning rate of 0.1 was found to exist. The training vs. testing datasets for accuracy and loss are shown in Figures 12-14.

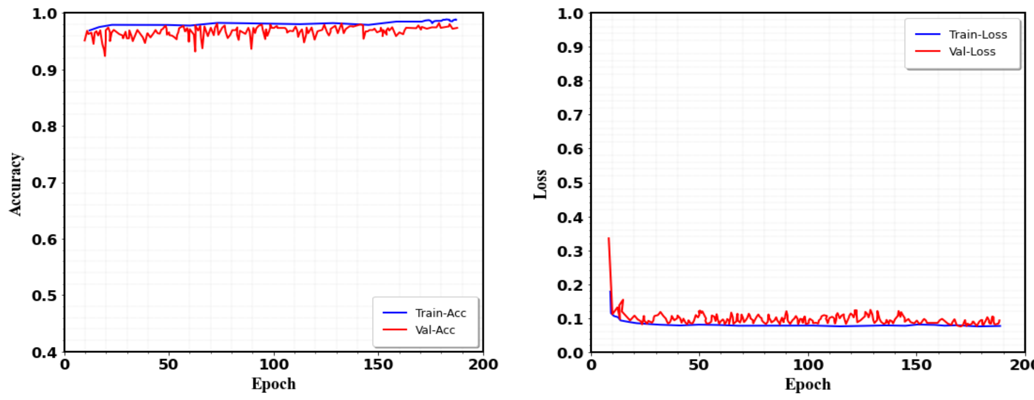

Figure 12. Evaluation of training vs testing performance for dataset 1

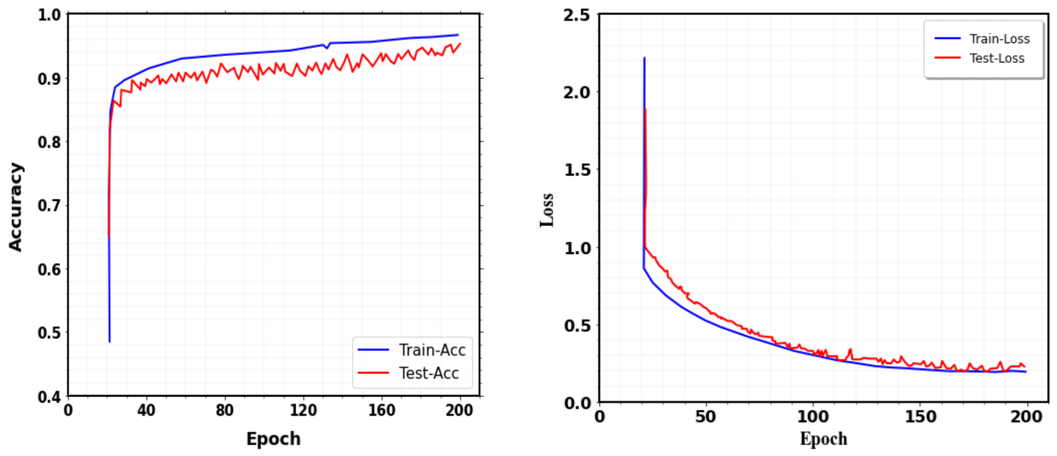

Figure 13. Evaluation of training vs testing performance for dataset 2

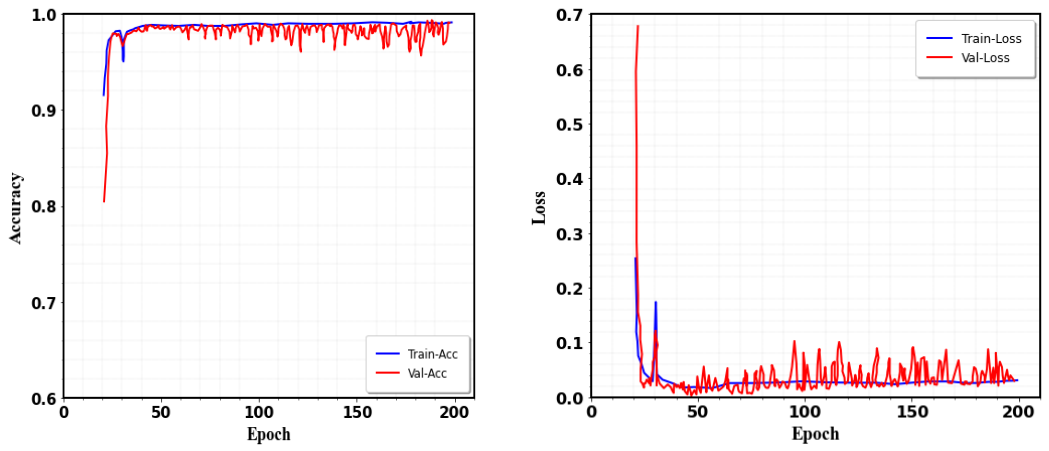

Figure 14. Evaluation of training vs testing performance for dataset 3

4. Conclusions

The selection and categorization of ensemble method feature sets were used to create an effective intrusion detection system for the cloud environment. This EBWO strategy, which is our proposed feature selection method, was utilized to choose a useful reduced feature set from provided intrusion datasets. DNetCNN and HMLSTM were used as part of an ensemble for categorizing different assault types. The main objective of the ensemble approach was to improve prediction accuracy over all other classifiers. Additionally, we assessed this suggested study using three separate intrusion datasets and applied FPR, FNR, precision, recall, DR, and accuracy to determine the effectiveness of classification. Thus, we established that the suggested strategy significantly improved the accuracy and endurance of a variety of categorization assignments, thereby supporting FS and important IDS processes. The results of our suggested method were 99.16% accuracy, 98.87% precision, 99.03% recall, and 98.65% F-Score.

However, the model proposed in this study had some limitations, such as a large number of parameters and a lengthy running time, both of which had an impact on the detection accuracy. As we continue to explore the model light-weighting, we will be able to further enhance the minority sample detection accuracy, the method's overall categorization effect, and the running time cost.

5. Conflicts of Interest

We declare that this manuscript is original, has not been published before and is not currently being considered for publication elsewhere.

ORCID

Vinolia Alexander Moudiappa https://orcid.org/0009-0002-3212-6364

Kanya Nataraj https://orcid.org/0009-0007-6473-0829

Veeramalai Natarajan Rajavarman https://orcid.org/0009-0000-8977-0770

References

Alqahtani, H., & Kumar, G. (2022). A deep learning-based intrusion detection system for in-vehicle networks. *Computers and Electrical Engineering*, 10(4), Article 108447. https://doi.org/10.1016/j.compeleceng.2022.108447

Azzaoui, H., Boukhamla, A. Z. E., Arroyo, D., & Bensayah, A. (2022). Developing new deep-learning model to enhance network intrusion classification. *Evolving Systems*, 13(1), 17-25. https://doi.org/10.1007/s12530-020-09364-z

Babu, K. S., & Rao, Y. N. (2023). MCGAN: Modified conditional generative adversarial network (MCGAN) for class imbalance problems in network intrusion detection. *Applied Sciences*, 13(4), Article 2576. https://doi.org/10.3390/app13042576

Chiba, Z., Abghour, N., Moussaid, K., El Omri, A., & Rida, M. (2019). New anomaly network intrusion detection system in cloud environment based on optimized back propagation neural network using improved genetic algorithm. *International Journal of Communication Networks and Information Security*, 11(1), 61-84. https://doi/org/10.17762/ijcnis.v11i1.3764

Devan, P., & Khare, N. (2020). An efficient XGBoost–DNN-based classification model for network intrusion detection system. *Neural Computing and Applications*, 32(16), 12499-12514. https://doi.org/10.1007/s00521-020-04708-x

Du, J., Yang, K., Hu, Y., & Jiang, L. (2023). NIDS-CNNLSTM: Network intrusion detection classification model based on deep learning. *IEEE Access*, 11(1), 24808-24821. https://doi.org/ 10.1109/ACCESS.2023.3254915

Gupta, S. K., Tripathi, M., & Grover, J. (2022). Hybrid optimization and deep learning-based intrusion detection system. *Computers and Electrical Engineering*, 100(1), Article 107876. https://doi.org/10.1016/j.compeleceng.2022.108156

Hnamte, V., & Hussain, J. (2023). DCNNBiLSTM: An efficient hybrid deep learning-based intrusion detection system. *Telematics and Informatics Reports*, 10(1), Article 100053. https://doi.org/10.1016/j.teler.2023.100053

Imran, M., Haider, N., Shoaib, M., & Razzak, I. (2022). An intelligent and efficient network intrusion detection system using deep learning. *Computers and Electrical Engineering*, 9(9), Article 107764. https://doi.org/10.1016/j.compeleceng.2022.107764

Kasongo, S. M. (2023). A deep learning technique for intrusion detection system using a Recurrent Neural Networks-based framework. *Computer Communications*, 19(9), 113-125. https://doi.org/10.1016/j.comcom.2022.12.010

Khan, M. A. (2021). HCRNNIDS: hybrid convolutional recurrent neural network-based network intrusion detection system. *Processes*, 9(5), Article 834. https://doi.org/10.3390/pr9050834

Kim, T., & Pak, W. (2022). Robust network intrusion detection system based on machine-learning with early classification. *IEEE Access*, 10(1), 10754-10767. https://doi.org/ 10.1109/ACCESS.2022.3145002

Kunang, Y. N., Nurmaini, S., Stiawan, D., & Suprapto, B. Y. (2021). Attack classification of an intrusion detection system using deep learning and hyperparameter optimization. *Journal of Information Security and Applications*, 5(8), Article 102804. https://doi.org/10.1016/j.jisa.2021.102804

Liu, J., Gao, Y., & Hu, F. (2021). A fast network intrusion detection system using adaptive synthetic oversampling and LightGBM. *Computers & Security*, 10(6), Article 102289. https://doi.org/10.1016/j.cose.2021.102289

Mighan, S. N., & Kahani, M. (2021). A novel scalable intrusion detection system based on deep learning. *International Journal of Information Security*, 20(3), 387-403. https://doi.org/10.1007/s10207-020-00508-5

Rahman, M. A., Asyhari, A. T., Leong, L. S., Satrya, G. B., Tao, M. H., & Zolkipli, M. F. (2020). Scalable machine learning-based intrusion detection system for IoT-enabled smart cities. *Sustainable Cities and Society*, 6(1), Article 102324. https://doi.org/10.1016/j.scs.2020.102324

Ravi, V., Chaganti, R., & Alazab, M. (2022). Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system. *Computers and Electrical Engineering*, 10(2), Article 108156. https://doi.org/10.1016/j.compeleceng.2022.108156

Saba, T., Rehman, A., Sadad, T., Kolivand, H., & Bahaj, S. A. (2022). Anomaly-based intrusion detection system for IoT networks through deep learning model. *Computers and Electrical Engineering*, 99(1), Article 107810. https://doi.org/10.1016/j.compeleceng.2022.107810

Saheed, Y. K., Abiodun, A. I., Misra, S., Holone, M. K., & Colomo-Palacios, R. (2022). A machine learning-based intrusion detection for detecting internet of things network attacks. *Alexandria Engineering Journal*, 61(12), 9395-9409. https://doi.org/10.1016/j.aej.2022.02.063

Soltani, M., Ousat, B., Siavoshani, M. J., & Jahangir, A. H. (2023). An adaptable deep learning-based intrusion detection system to zero-day attacks. *Journal of Information Security and Applications*, 7(6), Article 103516. https://doi.org/10.1016/j.jisa.2023.103516

Thakkar, A., & Lohiya, R. (2021). Analyzing fusion of regularization techniques in the deep learning-based intrusion detection system. *International Journal of Intelligent Systems*, 36(12), 7340-7388. https://doi.org/10.1002/int.22590

Thakkar, A., & Lohiya, R. (2023). Fusion of statistical importance for feature selection in Deep Neural Network-based Intrusion Detection System. *Information Fusion*, 90(1), 353-363. https://doi.org/10.1016/j.inffus.2022.09.026

Thirimanne, S. P., Jayawardana, L., Yasakethu, L., Liyanaarachchi, P., & Hewage, C. (2022). Deep neural network based real-time intrusion detection system. *SN Computer Science*, 3(2), Article 145. https://doi.org/10.1007/s42979-022-01031-1

Vishwakarma, M., & Kesswani, N. (2022). DIDS: A deep neural network based real-time Intrusion detection system for IoT. *Decision Analytics Journal*, 5(1), Article 100142. https://doi.org/10.1016/j.dajour.2022.100142

Wang, Z., Liu, Y., He, D., & Chan, S. (2021). Intrusion detection methods based on integrated deep learning model. *Computers & Security*, 10(3), Article 102177. https://doi.org/10.1016/j.cose.2021.102177