International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V71I10P210

Volume 71 Issue 10, 105-114, October 2023
© 2023 Seventh Sense Research Group®

Original Article

Intelligent Cloud-Based Intrusion Detection System using Enhanced Sunflower Optimization with Deep Learning Model

Samineni Nagamani[1], S. Arivalagan[2], M. Senthil[3], P. Sudhakar[4]

[1,2,4]Department of CSE, Annamalai University, Tamilnadu, India.
[3]Department of AIML, QIS College of Engineering and Technology, Andhra Pradesh, India.

[1]Corresponding Author : maniramesh2004@gmail.com

Received: 23 June 2023 Revised: 28 August 2023 Accepted: 15 September 2023 Published: 03 October 2023

Abstract *- Cloud Computing (CC) can be used in many research areas mainly for its network capacity and high computing power. Flexibility, Data security, and cost-effectiveness of working options for remote employees made this technology grab the interest of many. Intruders find innovative attack types every day; thus, to thwart such attacks, initial accurate detection should be done using Intrusion Detection Systems (IDSs), and after that, proper responses must be given. IDSs, which serve a most significant role in the security of the network, contain 3 main elements they are decision engine, data collection, and conversion or feature selection. Currently, DL was developed as a novel technique which enables a high accuracy rate and low training time with its distinctive learning system. Therefore, this study develops an Intelligent Cloud based Intrusion Detection System using an Enhanced Sunflower Optimization with Deep Learning (CIDS-ESFODL) model. The presented CIDS-ESFODL technique focuses on the recognition and categorization of intrusions in the cloud platform. The presented CIDS-ESFODL model has a three-phase process. In the initial stage, the ESFO algorithm is applied as a feature selector, providing an optimal subset of features. Secondly, the Denoising Autoencoder (DAE) technique is implemented for classifying and recognizing intrusions. Finally, the Nadam optimizer is utilized for the adjustment of the hyperparameters. The investigational validation of the CIDS-ESFODL technique on the benchmark IDS dataset reported its significant performance over the other current models by means of distinct measures.*

Keywords *- Cloud environment, Deep Learning, Nadam optimizer, Intrusion Detection System, Feature selection.*

1. Introduction

CC is one of the latest service innovations in the IT field. The CC permits access without limitations of location and time, which is one of its major merits [1]. CC supports collaborative and mobile services or applications, allows the flexibility of governing storage capacity, and offers reasonable costs [2]. Additionally, cloud services are multisource, letting the users use many service providers depending on their necessities. The utility of CC even minimizes energy consumption, physical space and maintenance necessities for on-site storage and capital expenditures [3]. Since CC services are becoming more prevalent, governments, a large group of firms, and banks have implemented this technology. This transition even exposed such systems to several forms of cyberattacks by intruders and hackers, demanding robust security systems. While implementing CC technologies, cybersecurity should prioritize healthcare services due to the privacy of operational and patient data [4]. Prevailing IDS work on the mechanism of anomaly detection or signature. If the detection system is unsuitable, operational and patient data are at risk [5]. The cybersecurity approach helps to find and protect malevolent actors while assisting in enhancing an administration's defence from cyberattacks. There are two main techniques for detecting malicious activity in the cloud situation they are Machine Learning (ML) techniques and Non-ML (NML) methods [6]. NML techniques, a method including signature-based techniques and virtual machine-related methods, depend on the features of cloud malicious behaviours to find the assaults [7]. In recent times, ML approaches have been formulated to solve the restriction of NML techniques. Many existing IDS utilize typical ML approaches for developing detection methods. ML methods were broadly implemented to build the IDS [8]. But, because of the volume and speed of the IoT-generated data, orthodox ML approaches that want well-crafted feature engineering need intensive research efforts for extracting the representative attributes from unstructured and big data produced by IoT gadgets. Thus, classical ML–related solutions still face several complexities [9]. Currently, Deep Learning (DL) techniques are employed in detection mechanism systems. DL expedited the analysis between real and fast data streams in the extraction of appropriate data to forecast the future of the IoT sector. DL was more reliable compared to conventional learning since it easily derives data and, hence, grants better accuracy [10].

This study develops an Intelligent Cloud based Intrusion Detection System using an Enhanced Sunflower Optimization with Deep Learning (CIDS-ESFODL) model. The presented CIDS-ESFODL technique follows a three-stage process. In the initial stage, the ESFO algorithm is applied as a feature selector, providing an optimal subset of features. Secondly, the Denoising Autoencoder (DAE) technique is implemented for classifying and recognizing intrusions. Finally, the Nadam optimizer is utilized for the adjustment of the hyperparameters. The investigational validation of the CIDS-ESFODL technique on the benchmark IDS dataset reported its significant performance over the other current models by means of distinct measures.

2. Related Works

Mayuranathan et al. [11] presented an effective selection-based Feature Subset (FS) classifying technique for detecting DDoS attacks. To identify these attacks in IDS, a better FS is chosen with maximal recognition using the Random Harmony Search (RHS) optimization technique. DL-based classification method with RBM is exploited to identify the DDoS as soon as the feature is selected. Seven additional layer sets are present amidst the RBM hidden and visible layers to enhance the recognition accuracy of DDoS attacks. The authors in [12] proposed an IDS with DL method using Fuzzy Min Max NN Based IDS (FMMNN-IDS). An expansion-contraction and fuzzy min-max learning approach that could learn nonlinear class restrictions in a singular traverse dataset and provide the capability to integrate and optimize the present classes without retraining is exploited to identify min-max points.

The authors in [13] presented a newly incorporated Cloud-based IDS (CIDS) to immunize the cloud against any potential assaults. The presented CIDS comprises five major mechanisms to do the subsequent action: to capture and analyze the Intrusion Detection (ID), traffic movements, monitoring the network, extracting features, logging all activities, and taking a reaction. Moreover, an improved bagging ensemble method of three DL techniques is exploited for predicting intrusion efficiently.

Sharon et al. [14] designed a method to exploit an integration of an SAE along with Stacked Contractive AE (S-SCAE) together with the Bi-DLDA model (succeeded by the layer with attention model, dense layer, and dropout layer) to detect intrusion in a cloud atmosphere. Furthermore, a cloud IDS is modelled for collecting the data congestion from the NSL-KDD data and employs a fusion technique to define whether acquired packets are harmful.

In [15], designed an effective IDS was designed for the cloud environments with ensemble FS and classification methods. The presented model depends on the univariate ensemble FS method employed for choosing valuable decreased FSs from the given intrusion dataset. At the same time, the ensemble classifier could proficiently fuse a single classifier to produce a robust classifier by implementing the voting method. An ensemble-based presented model effectively classifies whether the networking traffic behaviours are standard or under attack. Zhang et al. [16] proposed an efficient network IDS-based DL method. The presented technique applies a DAE with a weighted loss function for selecting features that define constraints amount of features for IDS for reducing feature dimensionality. The selected dataset is later categorized by the compact MLP for IDS.

3. The Proposed Model

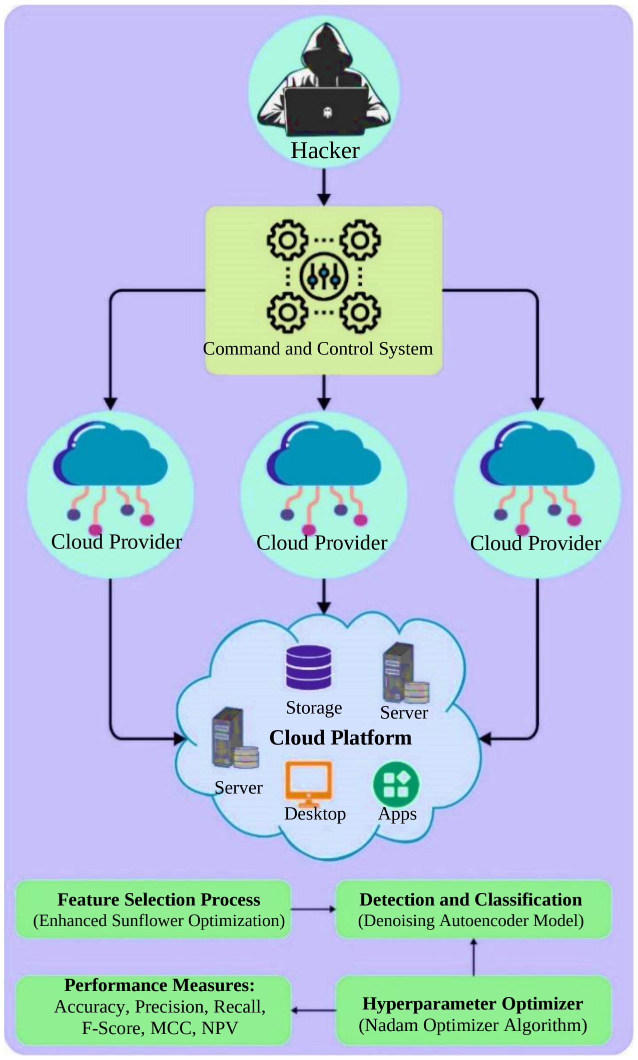
Hacker
Command and Control System
Cloud Provider
Cloud Provider
Cloud Provider
Storage
Server
Cloud Platform
Server
Desktop
Apps
Feature Selection Process (Enhanced Sunflower Optimization)
Detection and Classification (Denoising Autoencoder Model)
Performance Measures: Accuracy, Precision, Recall, F-Score, MCC, NPV
Hyperparameter Optimizer (Nadam Optimizer Algorithm)

Fig. 1 Overall flow of the CIDS-ESFODL system

In this article, a novel CIDS-ESFODL methodology for the recognition and categorization of intrusions in the cloud platform. This CIDS-ESFODL methodology utilizes a three-state process, namely Nadam optimizer, ESFO-based FS selection, and DAE-based ID. Fig. 1 portrays the comprehensive flow of the CIDS-ESFODL system.

3.1. Stage I: Feature Selection

Primarily, the ESFO algorithm is applied as a feature selector, providing an optimal subset of features. The SFO technique simulates the SF movement to absorb the sunlight [17]. The SFO has encompassed two stages: movement and pollination. In the movement stage, SF takes random steps towards the optimum SF, regarded as the sun.

In the pollination stage, SF cooperates to generate pollen gamete. Accurate solution space exploration was the major aspect of finding the global optimal of optimized problems. To realize these issues, the study introduced an updated SFO version, such as the ESFO technique. This technique can be armed with a novel pollination operator that balances exploitation and exploration competencies. Also, this technique is exploited for scheduling tasks.

Create population: The first population encompassed Z SF.

$$Pop = \{S_1, S_2, \dots, S_Z\} \tag{1}$$

All the SFs $S_i = \{s_{i,1}, s_{i,2}, \dots, s_{i,M}\}$ in the population can be encompassed by M numbers, which represent the identifier of the VM that the task runs on it.

$$s_{ij} = s_{\min} + \varphi_{ij}(s_{\max} - s_{\min}) \tag{2}$$

In Eq. (2), φ_{ij} denotes the uniformly distributed random integer ranges from zero to one, $s_{\min}$ and $s_{\max}$ are correspondingly the lower and upper boundaries of s_{ij}. For scheduling tasks, $s_{\min}$ and $s_{\max}$ are correspondingly initialized as 1 and M. There exist two VMs and six tasks $\{t_1, t_2, t_3, t_4, t_5, t_6\}$,. A potential solution is $S_i = [1,2,2,2,1,1]$ that map tasks t_1, t_5, and t_6 into the initial VMs, and t_2, t_3, and t_4 into the second VMs.

Two limitations need to be fulfilled in task allocation. All the tasks must be contained in the scheduling task, and all the tasks might appear only once. Meanwhile, the scheduling task was a distinct issue; the index of the task and VM is a positive integer. To employs ESFO in scheduling task, the element of SF is round to be a distinct number.

Power calculation: The power of all the SFs S_i is evaluated. The word "power" is associated with the solution cost in the scheduling task. Afterwards, power calculation, SF with minimal cost (S^*) was taken into account as the sun.

Pollination: To design this method, the typical SFO arbitrarily chooses $p \times Z$ sunflower from the population and later updates every chosen SF as

$$S_i^{t+1} = r_i.\left(S_i^t - S_j^t\right) + S_j^t \tag{3}$$

Where p indicates the pollination rate. S_i^t and S_j^t denotes the locations of i and j SFs at iteration t. r_i shows the random value within [0,1]. In the typical SFO, $p = 0.5$ is exploited. In ESFO, the study developed the subsequent formula for the pollination stage.

$$\begin{gathered} S_i^{t+1} = \alpha^t \times A + (1 - \alpha^t) \times B \\ A = S_j^t + U(-1,+1) \times \left(S_i^t - S_j^t\right) \\ B = S_i^t + \beta \times (S^* - S_i^t) \end{gathered} \tag{4}$$

Where α^t indicates the switching probability at t iteration controlling the local and global pollination ratio. S^* denotes the present optimum SF. $U(-1,+1)$ is a uniformly generated random integer within [1, 1] that adds some deviation to improve the search around the SF S_i, β denotes the scaling factors that control the amplitude of the searching direction $(S^* - S_i^t)$. Then, apply the value $\beta = 3.\lambda$, whereas $\lambda \in (0,1)$ indicates a uniform random integer:

$$\alpha^t = \alpha_{\max} - \frac{t}{I}(\alpha_{\max} - \alpha_{\min}) \tag{5}$$

In Eq. (5), I indicate the maximal amount of iteration. $\alpha_{\max}$ and $\alpha_{\min}$ are fixed as 0.6 and 0.4. Now, the concept is that ESFO does an enhanced global search initially at iteration and further local search as it reaches the end.

The primary motive in presenting the novel pollination operator was simultaneously enhancing exploitation and exploration. The Term A and B in Eq. (4) increases the exploitation and exploration by mimicking the local and global pollination technique.

Movement: here, $(1 - p) \times Z$ SF is chosen and upgraded as:

$$S_i^{t+1} = S_i^t + r.\frac{S^* - S_i^t}{\|S^* - S_i^t\|} \tag{6}$$

The direction changes of SF S_i toward the sun S^* is simulated by the above equation. Lastly, the suitable SF is found and replaced with the sun.

$$S^* = S_k | P(S_k) > P(S_j) \forall j = 1,2,Z \tag{7}$$

The process repeats until the maximal amount of iterations is obtained.

A1gorithm 1: The ESFO Algorithm

```
Input: ESFO parameter;
$M$ task, $T = \{t_1, t_2, t_M\}$;
$N$ virtual machine, $V = \{v_1, v_2, \dots, v_N\}$;
Output: A top solution;
Initialize the population of SF;
Assess the power of SFs; $t = 0$;
while $(t < I)$ do
    Calculate the pollination of SF;
    Calculate the movement of SF;
    Estimate the power of SF;
    Upgrade sun $S^*$;
   $t = t + 1$;
end
Return the sun $S^*$ as the better solution;
```

The Fitness Function (FF) exploited in the presented method is devised to have a balance amongst the number of selective attributes in all solutions (min), and the classifier accuracy (max) attained using selective features, Eq. (8) designates the fitness function to evaluate the solution.

$$Fitness = \alpha\gamma_R(D) + \beta\frac{|R|}{|C|} \qquad (8)$$

Where $\gamma_R(D)$ embodies the classifier error rate of a given classier (the KNN) technique is utilized here).$|C|$ is the overall attributes in the dataset, and $|R|$ is the cardinality of the selective subset, and β and α were 2 parameters corresponding to the significance of subset length and classification quality. $\in$ [1, 0] and $\beta = 1 - \alpha$.

3.2. Stage II: ID

This work employs the DAE method for classifying and recognizing intrusions. AE can be considered to be an optimization technique that is used for extracting and learning principal components in case of larger data dissemination [18]. It is commonly considered a DL algorithm as it owns the power to make a deep network that can handle the networking to coordinate with the expected environments. It is generally exploited for denoising, compression, image extraction, etc. The image compression technique is employed as a feature selection method in the AE. AE is considered the most suited pre-processing method for image classification by employing DNN. As the size is higher, one extra intermediate Hidden Layer (HL) is regarded for the encoder and decoder. The middle layer comprises the encoded image with a dimension of 64x64. Where X_i characterizes the input, H_i epitomizes HL (here I am 1 to 3) and Y_i is the outcome as shown below:

$$h_i = f_i(W_i X_i + b_i), i = 1\ to\ 4 \qquad (9)$$

Where, W_i indicates the weight vector amidst H_1 to H_2, X_i to H_1, and Y_i.

The Deep Neural Network (DNN) is relatively nonlinear, and thereby, they are not worth enough for main problems. Therefore, pretraining with the noisy dataset was very essential. This resulted in a process where noises are artificially added to all the layers to give rapid training and the best performance. An extension of typical AEA is a DAE that was proposed as a base for deep networks.

The basic concept of DAE is bare and relatively simple. The presented model reconstructs the dataset from an input of the corrupted or ruined dataset. This is a process of the effect of the force placed on the HL to recognize powerful aspects and avoid simply learning. Therefore, the AE was trained for design input from the input dataset's corrupted version. This results in a more polished output than the input dataset. This DAE is considered a stochastic version of typical AE where it performs two tasks: it loosens the effects of the corruption technique employed to the input, and it encodes the input. Fig. 2 signifies the infrastructure of DAE.

The training method of the DAE is rather a non-complex task. One means of training it is by stochastically collapsing the dataset and passing it to the NN. Based on this, the AE is trained alongside the original data. Another way is to ruin the dataset by removing part of the dataset. This might lead to an AE forecasting the missing input. The DAE is stacked on one another for the iterative learning algorithm to offer an equilibrium between input and output.

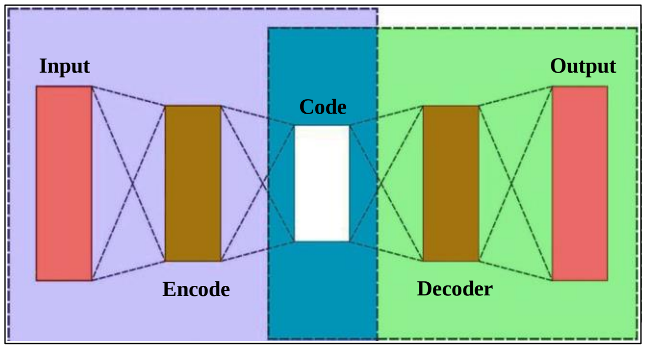

Fig. 2 Architecture of DAE

3.3. Stage III: Hyperparameter Tuning

Here, the Nadam optimizer is exploited for the adjustment of the hyperparameters. Integrating the Nesterov momentum with Adam optimization produces a novel technique termed the Nadam technique [19]. The learning procedure in the trained model was accelerated using an exponential decay dependent upon the moving gradient average through the momentum trick named Nesterov.

From the Adam optimization, the Nadam optimization connects further quickly and is further appropriate for the prior training step. The specifics of the optimizer techniques are denoted.

```
Algorithm 2: Pseudocode of Nadam optimizer
Input: Instances of the data
Output: Optimal Hyperparameters of DAE
Initialized $k \leftarrow 0, n_0 \leftarrow 0, m_0 \leftarrow 0, \tau \approx 10^{-8}$
        $\beta_1, \beta_2 \in [0,1), \eta_1 = 0.001$
while ($L(w_k^s)$ not converged) do
        $k \leftarrow k+1$
    $g_k \leftarrow \nabla_w L(w_k^s)$//gain gradients at time step $k$
    $\hat{g} \leftarrow g_k/(1-\prod_{i=1}^{k}\beta_1^i)$//correct $g_k$
    $m_k \leftarrow \beta_1 m_{k-1} + (1-\beta_1)g_k$ //the 1st moment estimation
    $\widehat{m}_k \leftarrow m_k/(1-\prod_{i=1}^{k+1}\beta_1^i)$//correct $m_k$
    $nk \leftarrow \beta_2 n_{k-1} + (1-\beta_2)g_k^2$ //the 2nd moment estimation
    $\hat{n}k \leftarrow nk/(1-\beta_2^k)$//correct $n_k$
    $w_k^s \leftarrow w_{k-1}^s - \frac{\eta_1}{\sqrt{\hat{n}k}+\tau}(\beta_1^{k+1}\widehat{m}_k + (1-\beta_1^k)\hat{g}_k)$
end while
```

The weight is advanced based on the law of recursive projected above in Algorithm 2. The parameter which requires that initialization is time step k, the 1st and 2nd order moments estimate mk, and n_k, exponential decay rates β_1 and β_2, the rate of learning η_1, and fuzz factor τ. The momentum schedule is provided as follows:

$$\beta_1^k = \beta_1\left(1 - 0.5 \times 0.96^{\frac{k}{250}}\right) \quad (10)$$

Whereas $\beta_1 = 0.99$.

Afterwards, the essential part of the technique is the iterative procedure. The primary stage is for calculating the loss function's gradient (w_{k-1}^s), whereas w_{k-1}^s signifies the weighted parameter. The secondary stage is to compute the 1st and 2nd order moments estimate mk and n_k at k time step. Next correction, the unbiased estimations $\widehat{m}k$ and $\hat{n}k$ are acquired. The final stage is to upgrade the weighted parameter. This iterative procedure then remains the optimum value attained.

4. Results and Discussion

The presented CIDS-ESFODL method is validated on the UNSW-NB15 Dataset. The existing UNSW-NB15 dataset consists of ten classes and 42 factors (labels excluded), known as DoS, Normal, Generic, Analysis, Backdoors, Fuzzers, Exploits, Reconnaissance, Shellcode, and Worms. In that, 5 classes are selected for the experimentation is given.

Table 1. Dataset details

Class	Sample Numbers
Normal	93000
Fuzzers	24246
DoS	794
Exploits	44525
Shellcode	1511
Total Number of Samples	**164076**

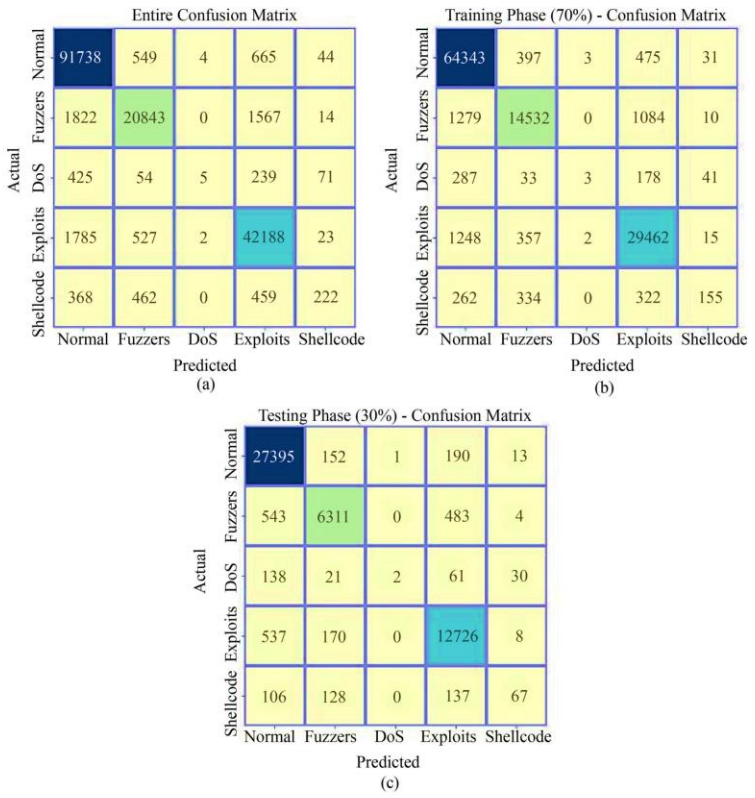

Fig. 3 Confusion matrix of CIDS-ESFODL model (a) Entire database, (b-c) 70 and 30 percent of TRS and TSS

The CIDS-ESFODL technique's confusion matrix is depicted in Fig. 3. With the overall dataset, and the CIDS-ESFODL technique identifies 91738 samples into normal class, 20843 samples into fuzzes, 5 samples into DoS, 42188 samples into exploits, and 222 samples into shellcode. Simultaneously, with 70% of TRS, the CIDS-ESFODL technique identifies 64343 samples into normal class, 14532 samples into fuzzes, 3 samples into DoS, 29462 samples into exploits, and 155 samples into shellcode. Finally, with 30% of TSS, the CIDS-ESFODL approach identifies 27395 instances into normal class, 6311 instances into fuzzes, 2 instances into DoS, 12726 instances into exploits, and 67 instances into shellcode.

Table 2. IDS outcome of CIDS-ESFODL system with discrete measures and classes

Class	Accuracy	Precision	Recall	F-Score	MCC	NPV
Dataset of Entire Class						
Normal	96.55	95.42	98.64	97.01	93.01	98.14
Fuzzers	96.96	92.90	85.96	89.30	87.62	97.60
DoS	99.52	45.45	00.63	01.24	05.31	99.52
Exploits	96.79	93.51	94.75	94.12	91.92	98.04
Shellcode	99.12	59.36	14.69	23.55	29.24	99.21
Average	**97.79**	**77.33**	**58.94**	**61.05**	**61.42**	**98.50**
Training (70%)						
Normal	96.53	95.44	98.61	97.00	92.97	98.09
Fuzzers	96.96	92.84	85.96	89.27	87.59	97.61
DoS	99.53	37.50	00.55	01.09	04.51	99.53
Exploits	96.80	93.47	94.78	94.12	91.92	98.05
Shellcode	99.12	61.51	14.45	23.40	29.53	99.20
Average	**97.79**	**76.15**	**58.87**	**60.97**	**61.30**	**98.50**
Testing (30%)						
Normal	96.59	95.39	98.72	97.02	93.10	98.26
Fuzzers	96.95	93.06	85.97	89.37	87.69	97.57
DoS	99.49	66.67	00.79	01.57	07.24	99.49
Exploits	96.78	93.59	94.68	94.13	91.92	97.99
Shellcode	99.13	54.92	15.30	23.93	28.68	99.24
Average	**97.79**	**80.72**	**59.09**	**61.21**	**61.72**	**98.51**

Table 2 offers the overall IDS results of the CIDS-ESFODL model. Fig. 4 investigates the IDS outputs of the CIDS-ESFODL model on the overall dataset in terms of accu_y, prec_n, and reca_l. The experimental results indicated that the CIDS-ESFODL model reaches enhanced results under each class. For instance, with normal class, the CIDS-ESFODL model attains accu_y, prec_n, and reca_l of 96.55%, 95.42%, and 98.64% respectively. Meanwhile, with the Fuzzers class, the CIDS-ESFODL approach attains accu_y, prec_n, and reca_l of 96.96%, 92.90%, and 85.96% correspondingly. Furthermore, with the DoS class, the CIDS-ESFODL approach attains accu_y, prec_n, and reca_l of 99.52%, 45.45%, and 00.63% correspondingly.

Fig. 5 inspects the IDS outputs of the CIDS-ESFODL technique on the entire dataset in terms of F_{score}, MCC, and NPV. The results show that the CIDS-ESFODL approach reaches enhanced results under each class. For example, with normal class, the CIDS-ESFODL method attains F_{score}, MCC, and NPV of 97.01%, 93.01%, and 98.14% correspondingly. In the meantime, with normal class, the CIDS-ESFODL approach attains F_{score}, MCC, and NPV of 89.30%, 87.62%, and 97.60% correspondingly. Additionally, with normal class, the CIDS-ESFODL approach attains F_{score}, MCC, and NPV of 1.24%, 5.31%, and 99.52% correspondingly.

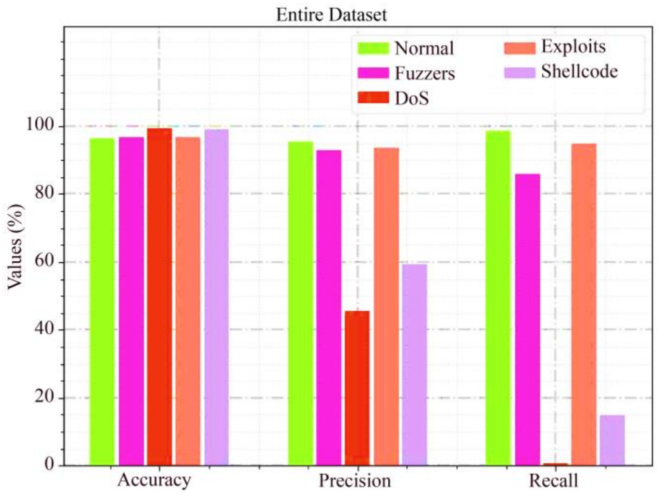

Fig. 4 $Accu_y$, $prec_n$, and $reca_l$ analysis of the CIDS-ESFODL system on the entire database

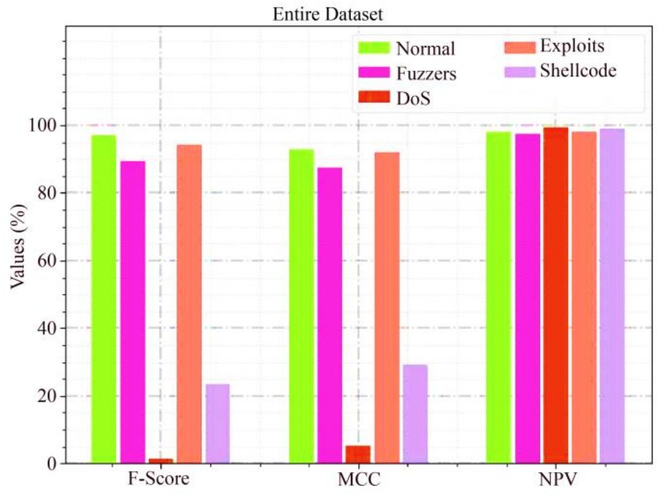

Fig. 5 F_{score}, MCC, and NPV analysis of the CIDS-ESFODL system on the entire database

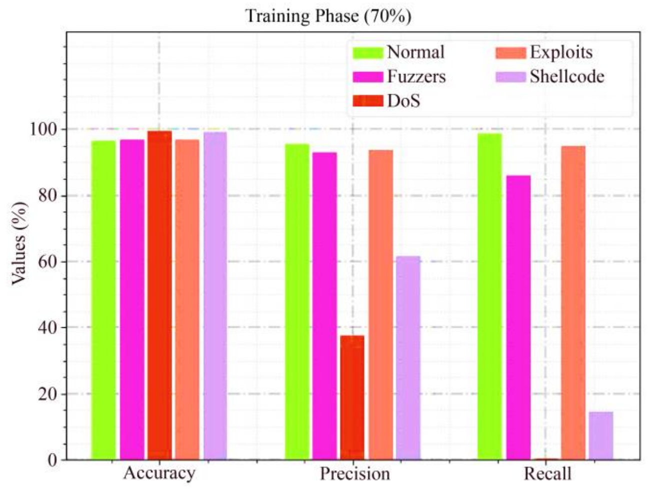

Fig. 6 $Accu_y$, $prec_n$, and $reca_l$ analysis of the CIDS-ESFODL system on 70% of TRS

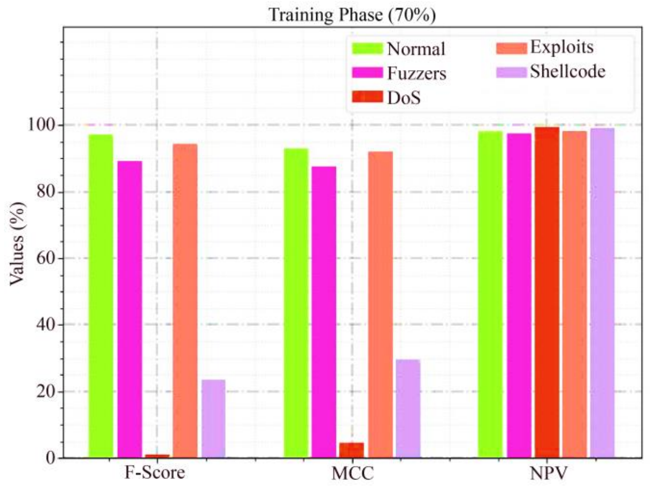

Fig. 7 F_{score}, MCC, and NPV analysis of CIDS-ESFODL system on 70% of TRS

Fig. 6 examines the IDS outcomes of the CIDS-ESFODL technique on 70% of TRS in terms of $accu_y$, $prec_n$, and $reca_l$. The results show that the CIDS-ESFODL technique reaches enhanced results under each class. For example, with normal class, the CIDS-ESFODL method attains $accu_y$, $prec_n$, and $reca_l$ of 96.53%, 95.44%, and 98.61% correspondingly. Meanwhile, with the Fuzzers class, the CIDS-ESFODL model attains $accu_y$, $prec_n$, and $reca_l$ of 96.96%, 92.84%, and 85.96% respectively. Moreover, with the DoS class, the CIDS-ESFODL approach achieves $accu_y$, $prec_n$, and $reca_l$ of 99.53%, 37.50%, and 00.55% correspondingly.

Fig. 7 examines the IDS outcomes of the CIDS-ESFODL approach on 70% of TRS in terms of F_{score}, MCC, and NPV. The figure displayed that the CIDS-ESFODL approach reaches enhanced results under each class. For cases with normal class, the CIDS-ESFODL model attains F_{score}, MCC, and NPV of 97%, 92.97%, and 98.09%, correspondingly.

Meanwhile, with normal class, the CIDS-ESFODL model attains F_{score}, MCC, and NPV of 89.27%, 87.59%, and 97.61% correspondingly. Moreover, with normal class, the CIDS-ESFODL model attains F_{score}, MCC, and NPV of 1.09%, 4.51%, and 99.53% correspondingly.

Fig. 8 examines the IDS outputs of the CIDS-ESFODL technique on 30% of TSS in terms of $accu_y$, $prec_n$, and $reca_l$. The experimental results specified that the CIDS-ESFODL technique reaches enhanced results under each class. For example, with normal class, the CIDS-ESFODL model attains $accu_y$, $prec_n$, and $reca_l$ of 96.59%, 95.39%, and 98.72% correspondingly. While, with the Fuzzers class, the CIDS-ESFODL model gains $accu_y$, $prec_n$, and $reca_l$ of 96.95%, 93.06%, and 85.97% correspondingly. Also, with the DoS class, the CIDS-ESFODL model attains $accu_y$, $prec_n$, and $reca_l$ of 99.49%, 66.67%, and 00.79% correspondingly.

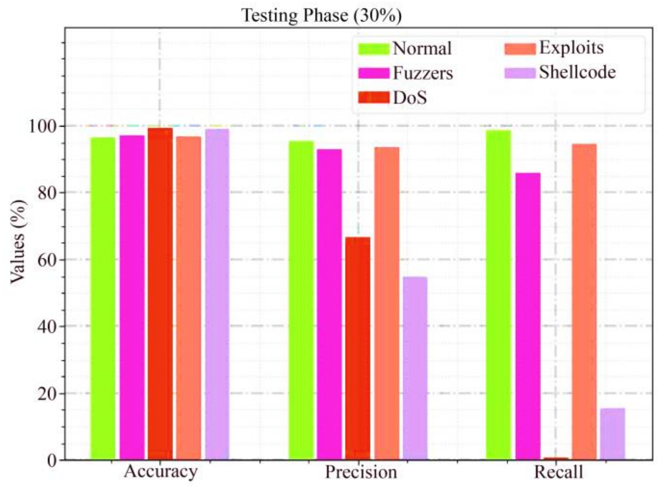

Fig. 8 $Accu_y$, $prec_n$, and $reca_l$ analysis of the CIDS-ESFODL system on 30% of TSS

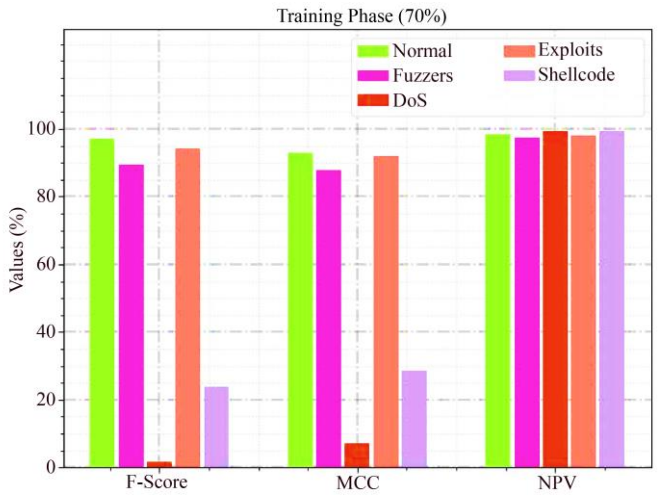

Fig. 9 F_{score}, MCC, and NPV evaluation of CIDS-ESFODL approach on 30% of TSS

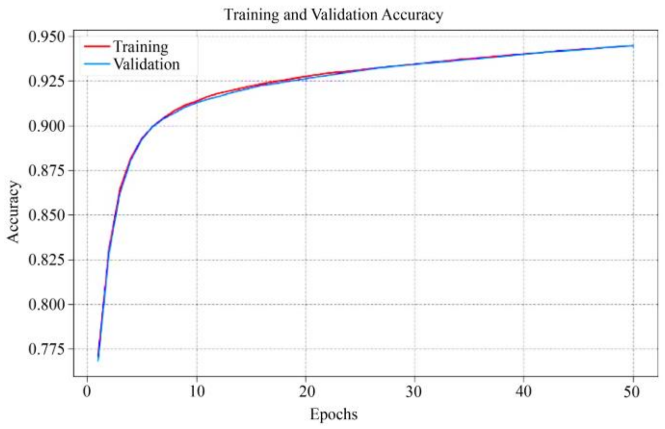

Fig. 10 TACC value and VACC value evaluation of the CIDS-ESFODL approach

Fig. 9 inspects the IDS outcomes of the CIDS-ESFODL technique on 30% of TSS in terms of F_{score}, MCC, and NPV. The results indicated that the CIDS-ESFODL method reaches enhanced results under each class. For example, with normal class, the CIDS-ESFODL model attains F_{score}, MCC, and NPV of 97.02%, 93.10%, and 98.26% correspondingly. In the meantime, with normal class, the CIDS-ESFODL model attains F_{score}, MCC, and NPV of 89.37%, 87.69%, and 97.57% correspondingly. Additionally, with normal class, the CIDS-ESFODL technique attains F_{score}, MCC, and NPV of 1.57%, 7.24%, and 99.49% correspondingly.

The TACC and VACC values of the CIDS-ESFODL method are inspected on IDS in Fig. 10. The output depicted that the CIDS-ESFODL method portrayed an enhanced achievement with higher TACC and VACC values. Also, the CIDS-ESFODL technique attained optimum TACC outputs.

The TLS and VLS values of the CIDS-ESFODL method are examined on IDS achievement in Fig. 11. The output portrays that the CIDS-ESFODL method exhibited enhanced achievement with lesser TLS and VLS values. Lastly, the CIDS-ESFODL approach has lessened VLS outputs.

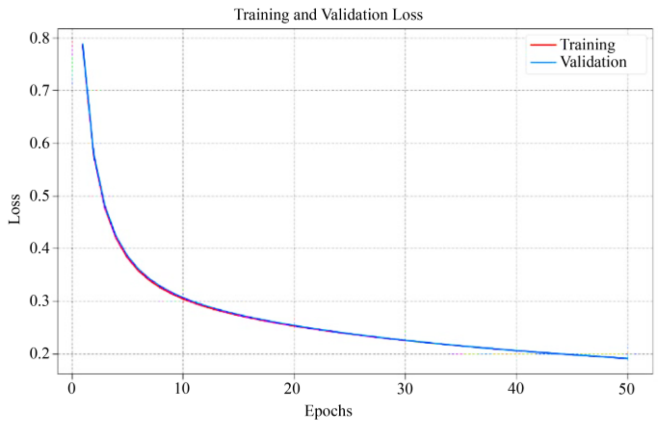

Fig. 11 TLS value and VLS value evaluation of the CIDS-ESFODL approach

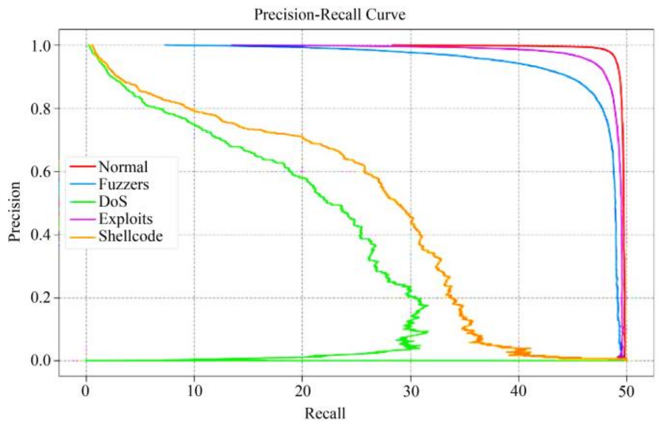

Fig. 12 Precision-recall evaluation of the CIDS-ESFODL approach

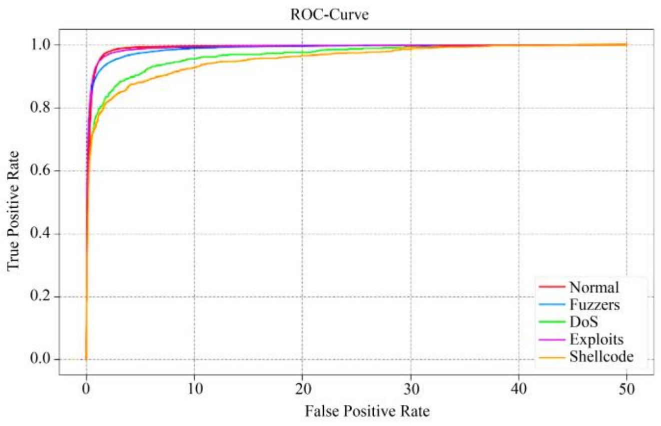

Fig. 13 ROC analysis of the CIDS-ESFODL system

A clear study of the precision-recall of the CIDS-ESFODL approach under the testing data is given in Fig. 12. The figure portrays that the CIDS-ESFODL approach has an enhanced precision-recall values under the overall classes.

An elaborate study of ROC of the CIDS-ESFODL technique under the testing data is provided in Fig. 13. The results signified that the CIDS-ESFODL technique has shown its capability in classifying discrete classes.

Table 3. $Accu_y$ analysis of the CIDS-ESFODL system with other ML techniques

Methods	**Accuracy (%)**
CIDS-ESFODL	97.79
DT Classifier	86.19
LR Model	83.93
NB Classifier	83.17
ANN-Single Layered	81.84
EM-Clustering	78.62
Ramp-KSVCR	94.23
PSI-NetVisor	94.57
SVM Model	96.08

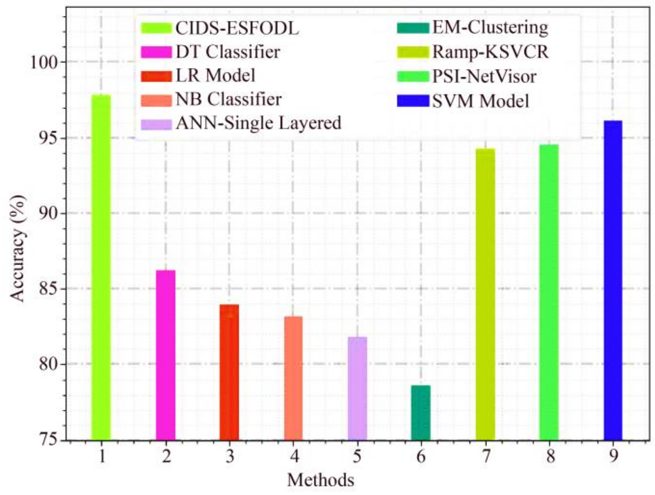

Fig. 14 $Accu_y$ analysis of the CIDS-ESFODL system with other ML techniques

Table 3 and Fig. 14 show an elaborate relative study of the CIDS-ESFODL method with current ML methods [21]. The experimental values indicated that the EM-clustering approach reaches a lower $accu_y$ of 78.62%.

Next, the DR, LR, and NB models reached slightly improved performance. Along with that, the Ramp-KSVCR, PSI-NetVisor, and SVM Models attained closer results. However, the CIDS-ESFODL model accomplishes maximum outcomes with an increased $accu_y$ of 97.79%. Therefore, the CIDS-ESFODL model is found to be an effectual tool for cloud IDS.

5. Conclusion

In this article, a novel CIDS-ESFODL methodology for the recognition and categorization of intrusions in the cloud platform. This CIDS-ESFODL methodology utilizes a three-stage process. In the initial stage, the ESFO algorithm is applied as a feature selector, providing an optimal subset of features. Secondly, the DAE algorithm is implemented for classifying and recognizing intrusions. Finally, the Nadam optimizer is utilized for the adjustment of the hyperparameters. The investigational validation of the CIDS-ESFODL technique on the benchmark IDS dataset reported its significant achievement over the other current approaches by means of diverse measures. In the future, an ensemble learning process can boost the CIDS-ESFODL technique's experimental results.

References

[1] Jan Lansky et al., "Deep Learning-Based Intrusion Detection Systems: A Systematic Review," *IEEE Access*, vol. 9, pp.101574-101599, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[2] Mesut Uğurlu, and İbrahim Alper Doğru, "A Survey on Deep Learning Based Intrusion Detection System," *4th International Conference on Computer Science and Engineering (UBMK)*, pp. 223-228, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[3] Mohamed Amine Ferrag et al., "Deep Learning-Based Intrusion Detection for Distributed Denial of Service Attack in Agriculture 4.0," *Electronics*, vol. 10, no. 11, pp. 1-26, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[4] Jiangang Shu et al., "Collaborative Intrusion Detection for VANETs: A Deep Learning-Based Distributed SDN Approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4519-4530, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[5] Junjie Cen, and Yongbo Li, "Deep Learning-Based Anomaly Traffic Detection Method in Cloud Computing Environment," *Wireless Communications and Mobile Computing*, vol. 2022, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[6] Loheswaran Karuppusamy et al., "Chronological Salp Swarm Algorithm based Deep Belief Network for Intrusion Detection in Cloud using Fuzzy Entropy," *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, vol. 35, no. 1, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[7] Parul Singh, and Virender Ranga, "Attack and Intrusion Detection in Cloud Computing using an Ensemble Learning Approach," *International Journal of Information Technology*, vol. 13, no. 2, pp. 565-571, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[8] Nishtha Srivastava et al., "A Review of Machine Learning-Based Intrusion Detection Systems on the Cloud," *Security, Privacy and Data Analytics*, vol. 848, pp. 303-317, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[9] E. Balamurugan et al., "Network Optimization using Defender System in Cloud Computing Security based Intrusion Detection System with Game Theory Deep Neural Network (IDSGT-DNN)," *Pattern Recognition Letters*, vol. 156, pp. 142-151, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[10] Vinayakumar Ravi, Rajasekhar Chaganti, and Mamoun Alazab, "Recurrent Deep Learning-Based Feature Fusion Ensemble Meta-Classifier Approach for Intelligent Network Intrusion Detection System," *Computers and Electrical Engineering*, vol. 102, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[11] M. Mayuranathan, M. Murugan, and V. Dhanakoti, "Best Features Based Intrusion Detection System by RBM Model for Detecting DDoS in Cloud Environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 3, pp. 3609-3619, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[12] Anil Kumar et al., "An Intrusion Identification and Prevention for Cloud Computing: From the Perspective of Deep Learning," *Optik*, vol. 270, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[13] Wisam Elmasry, Akhan Akbulut, and Abdul Halim Zaim, “A Design of an Integrated Cloud-Based Intrusion Detection System with Third Party Cloud Service,” *De Gruyter Open Access*, vol. 11, no. 1, pp. 365-379, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[14] Andrea Sharon et al., “An Intelligent Intrusion Detection System using Hybrid Deep Learning Approaches in Cloud Environment,” *International Conference on Computer, Communication, and Signal Processing*, pp. 281-298, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[15] S. Krishnaveni et al., “Efficient Feature Selection and Classification through Ensemble Method for Network Intrusion Detection on Cloud Computing,” *Cluster Computing*, vol. 24, no, 3, pp. 1761-1779, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[16] Hongpo Zhang et al., “An Effective Deep Learning Based Scheme for Network Intrusion Detection,” *24th International Conference on Pattern Recognition (ICPR)*, pp. 682-687, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[17] Hojjat Emami, “Cloud Task Scheduling using Enhanced Sunflower Optimization Algorithm,” *ICT Express*, vol. 8, no. 1, pp. 97-100, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[18] Pradeep Kumar Mallick et al., “Brain MRI Image Classification for Cancer Detection using Deep Wavelet Autoencoder-Based Deep Neural Network,” *IEEE Access*, vol. 7, pp. 46278-46287, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[19] Bin Xiao, Yonggui Liu, and Bing Xiao, “Accurate State-of-Charge Estimation Approach for Lithium-Ion Batteries by Gated Recurrent Unit with Ensemble Optimizer,” *IEEE Access*, vol. 7, pp. 54192-54202, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[20] Mohammad Dawood Momand, Vikas Thada, and Utpal Shrivastava, “Intrusion Detection System in IoT Network,” *SSRG International Journal of Computer Science and Engineering*, vol. 7, no. 4, pp. 11-15, 2020. [CrossRef] [Publisher Link]

[21] Muataz Salam Al-Daweri et al., “An Analysis of the KDD99 and UNSW-NB15 Datasets for the Intrusion Detection System,” *Symmetry*, vol. 12, no. 10, pp. 1-32, 2020. [CrossRef] [Google Scholar] [Publisher Link]