


Article

Enhanced Chimp Optimization-Based Feature Selection with Fuzzy Logic-Based Intrusion Detection System in Cloud Environment

Manal Abdullah Alohalı¹, Muna Elsadıg¹, Fahd N. Al-Wesabi² , Mesfer Al Duhayyim³, Anwer Mustafa Hilal^{4,*} and Abdelwahed Motwakel⁵ 

¹ Department of Information Systems, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia

² Department of Computer Science, College of Science & Art at Mahayil, King Khalid University, Abha 62529, Saudi Arabia

³ Department of Computer Science, College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, Al-Kharj 16273, Saudi Arabia

⁴ Department of Computer and Self Development, Preparatory Year Deanship, Prince Sattam Bin Abdulaziz University, Al-Kharj 16278, Saudi Arabia

⁵ Department of Information Systems, College of Business Administration in Hawtat bani Tamim, Prince Sattam Bin Abdulaziz University, Al-Kharj 16278, Saudi Arabia

* Correspondence: a.hilal@psau.edu.sa

Abstract: Cloud computing (CC) refers to an Internet-based computing technology in which shared resources, such as storage, software, information, and platform, are offered to users on demand. CC is a technology through which virtualized and dynamically scalable resources are presented to users on the Internet. Security is highly significant in this on-demand CC. Therefore, this paper presents improved metaheuristics with a fuzzy logic-based intrusion detection system for the cloud security (IMFL-IDSCS) technique. The IMFL-IDSCS technique can identify intrusions in the distributed CC platform and secure it from probable threats. An individual sample of IDS is deployed for every client, and it utilizes an individual controller for data management. In addition, the IMFL-IDSCS technique uses an enhanced chimp optimization algorithm-based feature selection (ECOAFS) method for choosing optimal features, followed by an adaptive neuro-fuzzy inference system (ANFIS) model enforced to recognize intrusions. Finally, the hybrid jaya shark smell optimization (JSSO) algorithm is used to optimize the membership functions (MFs). A widespread simulation analysis is performed to examine the enhanced outcomes of the IMFL-IDSCS technique. The extensive comparison study reported the enhanced outcomes of the IMFL-IDSCS model with maximum detection efficiency with accuracy of 99.31%, precision of 92.03%, recall of 78.25%, and F-score of 81.80%.

Keywords: cloud security; metaheuristics; optimization algorithm; feature selection; fuzzy logic; intrusion detection



Citation: Alohalı, M.A.; Elsadıg, M.; Al-Wesabi, F.N.; Al Duhayyim, M.; Mustafa Hilal, A.; Motwakel, A. Enhanced Chimp Optimization-Based Feature Selection with Fuzzy Logic-Based Intrusion Detection System in Cloud Environment. *Appl. Sci.* **2023**, *13*, 2580. <https://doi.org/10.3390/app13042580>

Academic Editors: Arcangelo Castiglione and Habib Hamam

Received: 5 January 2023

Revised: 26 January 2023

Accepted: 13 February 2023

Published: 16 February 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Cloud computing (CC) is familiar and is obtaining attention from individual users and organizations. However, the transition to the cloud platform could be more direct, and there are many operational and security difficulties [1]. Guaranteeing the security of information outsourcing to the cloud is significant because of the trend of storing more data in the cloud. Virtual machines (VMs) and hypervisor technologies are also a security threat, as these VM and hypervisor technologies are prone to VM-level attacks [2,3]. Such a mechanism has some on-site computer institutions with more software and hardware systems [4,5]. Susceptibilities in the VM structure are used by invaders for exfiltrating data or conducting assaults, such as distributed denial of service (DDoS). This occurs because of the inherent weaknesses in the internet protocol stack. Moreover, numerous novel assaults have recently

appeared that use metamorphisms and polymorphism to avoid recognition [6]. In an IaaS cloud atmosphere, data regarding the victims' machines are easily exploited and acquired, enabling assaults on VMs.

The cloud structure is open and fully dispersed, making it a vulnerable target for invaders [7,8]. Thus, the security of cloud platforms is at risk, where conventional network assaults, along with cloud-specific assaults, threaten cloud users (may be organizations or individuals). Conventional network security measures and firewalls are better for stopping several outsider assaults. Still, assaults within the network, along with certain complex outsider assaults cannot be managed efficiently through these systems [9]. This is where intrusion detection systems (IDSs) come into play. The role of IDS in the security of the cloud is highly significant, as it serves as an additional defensive layer of safety. Apart from identifying only familiar assaults, it can find variants of unknown and familiar assaults [10,11]. An intrusion is an attempt to bypass the security mechanism or to negotiate the confidentiality, integrity, and availability (CIA) of a network or computer [12]. Intrusions are activated by aggressors trying to access cloud resources over the Internet, legal users trying to obtain privileges not provided to them formally, and users who misuse their rights to access resources [13]. Intrusion detection (ID) observes the actions in a network or computer and scrutinizes them for signs of intrusion. IDS could be hardware or software, or an integration of both to automate the ID process [14]. It captures data from the network or under observation and informs the network manager by logging or mailing intrusion events. However, the warnings made by IDS are rarely related to actual intrusion because false positives and negatives influence the IDS performance [15]. Though several IDS models are available in the literature, improvement is still needed for the detection efficiency. In addition, feature selection (FS) techniques are essential to eliminate irrelevant features that raise false alarms and increase the accuracy of the system.

This paper presents improved metaheuristics with a fuzzy logic-based intrusion detection system for the cloud security (IMFL-IDSCS) technique. The IMFL-IDSCS technique uses the enhanced chimp optimization algorithm-based feature selection (ECOAFS) method for choosing optimal features. The adaptive neuro-fuzzy inference system (ANFIS) method is enforced to recognize intrusions. Finally, the hybrid jaya shark smell optimization (JSSO) algorithm is used to optimize the membership functions (MFs). A widespread simulation analysis is performed to examine the enhanced outcomes of the IMFL-IDSCS technique. In short, the major contributions of the study are given as follows:

- An intelligent IMFL-IDSCS technique comprising ECOAFS, ANFIS classification, and JSSO-based parameter optimization is presented for intrusion detection. To the best of our knowledge, the presented IMFL-IDSCS technique does not exist in the literature.
- A new ECOAFS technique is designed for the selection of the optimal subset of features to accomplish enhanced classification performance.
- JSSO is employed with the ANFIS model for the classification of intrusions into different class labels.
- The performance of the proposed model is validated on the benchmark NSL-KDD dataset.

The rest of the paper is organized as follows. Section 2 provides the related works, and Section 3 introduces the proposed model. Later, Section 4 offers experimental validation and Section 5 draws the conclusions.

2. Related Works

A DNN with game theory for cloud security (GT-CSDNN) was modeled in [16]. The proposed method includes defender and attacker techniques when utilizing the game theory method. Moreover, the deep neural network (DNN) network leverages the devised game theory method to categorize regular data and attacks. The GT-performance CSDNNs are evaluated by using the CICIDS—2017 data. The collected data can be normalized, and the improved whale algorithm (IWA) was employed to evaluate the attack's optimal points and normal data. Shyla and Sujatha [17] suggested an innovative IDS recognized on an integration of a leader-oriented k-means clustering (LKM), an optimal fuzzy logic (FL) system. Initially, input data were assembled into clusters by using the LKM. Afterwards,

cluster datasets were afforded to the FL system (FLS). In this study, abnormal and normal data were examined by the FLS, and FLS training can be performed through grey wolf optimizer (GWO) by maximizing the rules.

Mishra et al. [18] devised a VM Introspection (VMI) related security structure model for fine granular monitoring of VMs for identifying renowned assaults and their variants. VMGuard enforces the software breakpoint injection approach, which is agnostic and is employed for trapping the accomplishment of programs. Inspired by text mining methods, VMGuard provides a 'Bag of n-grams (BonG)' method compiled with the term frequency-inverse document frequency (TF-IDF) approach for selecting and extracting attack features and normal traces. After implementing the random forest (RF) method for generating generic conduct for distinct groups of intrusion of the monitored VM, Aoudni et al. [19] introduced HMM_TDL, a DL method for preventing and detecting assault in the cloud environment. The devised approach can be performed in three states, incorporating a hidden Markov model (HMM) for attack detection. The HMM method sends hyper-alerts to the database for assault prevention.

In Ref. [20], an anomaly-related network IDS (NIDS) was modeled, analyzing and monitoring the network traffic flow that targeted a cloud platform. A network administrator must be informed regarding the nature of such traffic to block and drop any intrusive network connection. The binary-related particle swarm optimization (BPSO) was implemented to select the most related network features, whereas the standard-based PSO (SPSO) was enforced for tuning support vector machine (SVM) control parameters. Velliangiri and Premalatha [21] evaluated and presented a radial basis function neural network (RBF-NN) detector for detecting DDoS assaults. The resultant network can be frequently insufficient or needlessly intricate, and a suitable network structure is configured merely by the trial-and-error technique. This study devises the bat algorithm (BA) for configuring RBF-NN mechanically.

Sathiyadhas and Soosai Antony [22] presented an effectual dragonfly-improved invasive weed optimizer-based Shepard CNN (DIIWO-based ShCNN) for detecting the intruder and for mitigating the attack from the cloud paradigm and it can further detect the intruder with ShCNN. In [23], an effectual IDS was suggested by the presented sail fish dolphin optimizer-based Deep RNN (SFDO-based Deep RNN) that was employed for identifying an anomaly from the cloud framework. The established SFDO was formed by combining sail fish optimizer (SFO) and the dolphin echolocation (DE) technique. Virtual machine (VM) migration and cloud data management can be obtained utilizing ChicWhale technique.

Geetha and Deepa [24] examined a Fisher kernel-based PCA dimensional reduction technique and GWO-based weight dropped Bi-LSTM technique (FKPCA-GWO WDBiLSTM) for IDS. Primarily, combined with the data record for PCA, the Fisher kernel with Fisher score was offered as input for achieving linearly separable dimensionality reduction. Secondly, the WDBiLSTM network was utilized for retaining the long-term dependency but removing the feature from forward and backward directions. A novel DL approach dependent upon CNNs and RNNs for IDS was established for cloud security in [25]. With our DL technique, some detected and not accepted traffic was prevented in obtaining the server from the cloud.

3. The Proposed Model

In this study, a novel IMFL-IDSCS method was derived to detect intrusions in the distributed CC platform and secure it from probable threats. Figure 1 represents the workflow of the IMFL-IDSCS approach. IDS was deployed in the CC platform. There is an individual IDS controller for the cloud service provider (CSP). It is noticeable that if there exists a single IDS in the whole network, the load on it rises as the number of hosts rises. Besides that, it can be challenging to monitor dissimilar types of intrusions or assaults that act on every host presented in the network. To resolve these limitations, the study presents a framework, where the mini-IDS instance is deployed between every user of the cloud and the CSP. Consequently, the load on every IDS sample would be lower than that

of an individual IDS; thus, smaller IDS instances would be capable of conducting their work in a special way. The number of packets dropped would be decreased because of the decreased load that individual IDS instances would have. The IDS controllers would give all those IDS instances. If the client wants to access any services granted by CSP, IDS controllers must give IDS instances to that client. That instance would monitor all user activity, and once the client ends the session, a log of that session would be transmitted by the IDS instance to the IDS controller, which would be saved in cloud logs. Knowledge Base was saved in cloud logs and has data regarding the paradigm of user activity based on the data saved in the cloud. Whenever an IDS instance was granted to a specific user, IDS controllers from Knowledge Base would query data regarding the last activity. These activity patterns detect intrusion from the user's working pattern. Additionally, it uses various rules for multiple users such that each feature of IDS does not need to be positioned if specific users need only a few. The abovementioned IDS instance should be positioned on the application, system, and platform layers.

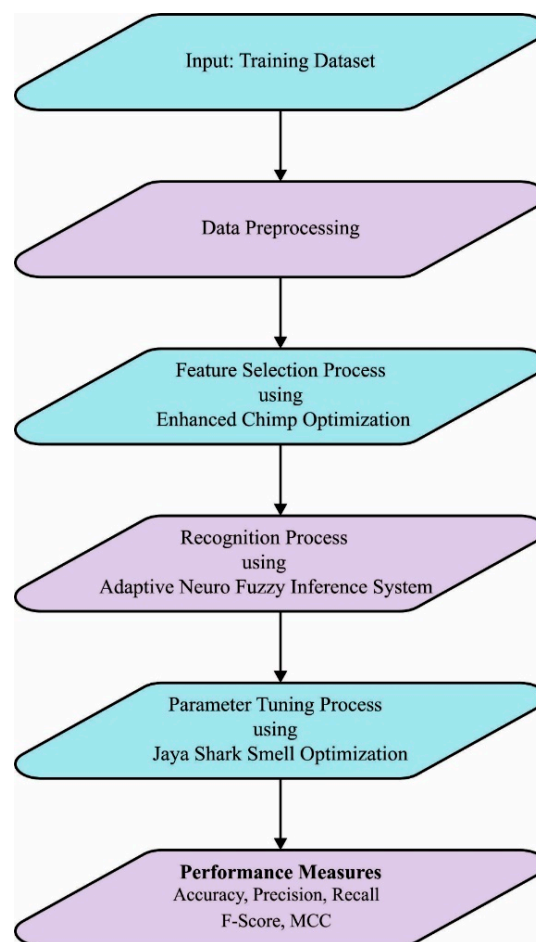


Figure 1. Workflow of IMFL-IDSCS system.

3.1. Process Involved in ECOA-FS Technique

Initially, the IMFL-IDSCS technique used the ECOA-FS approach for choosing optimal features. COA is a mathematical model that can be reliant on intelligent diversity [26]. Chase, drive, attack, and block are the distinct strategies of chimps that can realize obstacles, the attacker, the driver, and the chaser. These hunting behaviors are concluded in two stages. Exploration and exploitation are the two stages of COA. The initial stage comprises

chasing, driving, and blocking the prey. The next phase has attacked the prey. At the same time, the driving and chasing behaviors are illustrated in Equations (1) and (2):

$$d = |c \cdot x_{prey}(t) - m \cdot x_{chimp}(t)| \tag{1}$$

$$x_{chimp}(t + 1) = x_{prey}(t) - a \cdot d \tag{2}$$

Now, X_{prey} stands for the vector of prey location, x_{chimp} indicates the vector of chimp place, t represents the number of current iterations, and $a, c,$ and m imply the coefficient vector. This is accomplished in Equations (3)–(5):

$$a = 2 \cdot fc \cdot r_1 - fc \tag{3}$$

$$c = 2 \cdot r_2 \tag{4}$$

$$m = chaotic_value \tag{5}$$

Whereas fc indicates the non-linear decrease in 2.5 to 0, r_1 and r_2 indicate arbitrary value ranges from zero to one, and m denotes chaotic vector. The dynamic coefficient fc prefers dissimilar curves and slopes. Hence, the chimps used different abilities to search for prey. They upgraded the location dependent upon other chimps, and the mathematical modeling can be given in Equations (6)–(8):

$$\begin{aligned} d_{Attacker} &= |c_2 x_{Attacker} - m_1 x| \\ d_{Barrier} &= |c_2 x_{Barrier} - m_2 x| \\ d_{Chaser} &= |c_3 x_{Chaser} - m_3 x| \\ d_{Driver} &= |c_4 x_{Driver} - m_4 x| \end{aligned} \tag{6}$$

$$\begin{aligned} \chi_2 &= x_{Barrier}^{-a_2(d_{Barrier})} \chi_1 = x_{Attacker}^{-a_1(d_{Attacker})} \\ x_4 &= x_{Driver} - a_4(d_{Driver}) x_3 = x_{Chaser} - a_3(d_{Chaser}) \end{aligned} \tag{7}$$

$$\begin{aligned} x(t + 1) &= \frac{x_1 + x_2 + x_3 + x_4}{4} \\ x_1 &= x_{Attacker} - a_1(d_{Attacker}) \\ x_2 &= x_{Barrier} - a_2(d_{Barrier}) \\ x_3 &= x_{Chaser} - a_3(d_{Chaser}) \\ x_4 &= x_{Driver} - a_4(d_{Driver}) \end{aligned} \tag{8}$$

In ECOA, the extremely disruptive polynomial mutation was a revised edition of the polynomial mutation method. It resolves the constraint that the polynomial mutation method fell as the local optima when the parameter was near the boundary, as shown in Equations (9) and (12):

$$\delta_1 = \frac{\chi_i - lb}{ub - lb} \tag{9}$$

$$\delta_2 = \frac{ub - \chi_i}{ub - lb} \tag{10}$$

$$\delta_k = \begin{cases} [(2r) + (1 - 2r) * (1 - \delta_1)^{\eta_m + 1}]^{\frac{1}{\eta_m + 1} - 1}, & r \leq 0.5 \\ 1 - [2(1 - r) + 2(r - 0.5) * (1 - \delta_2)^{\eta_m + 1}]^{\frac{1}{\eta_m + 1}}, & otherwise \end{cases} \tag{11}$$

$$x_i = x_i + \delta_k(ub - lb) \tag{12}$$

Whereas ub and lb determine the upper and lower limits of search spaces, r indicates the arbitrary value ranges within $[0, 1]$ and η_m indicates the distribution exponential that can be a non-negative number.

Unlike the traditional ECOA, the solution update occurs in the search region in the direction of the continual value position. However, in BECOA, the search region is determined as the n dimension Boolean lattice. Moreover, the solution is upgraded through the corner of the hypercube. In addition, for choosing features, 1 characterizes the feature

selection, or 0. Furthermore, the BECOA derives a fitness function from determining solutions for retaining tradeoffs amongst a pair of objectives, as defined in Equation (13):

$$fitness = \alpha \Delta_R(D) + \beta \frac{|Y|}{|T|} \tag{13}$$

$\Delta_R(D)$ shows the classifier’s error, $|Y|$ indicates the subset size, and $|T|$ represents the overall features present in the data. In addition, α shows a parameter $\in [0, 1]$ associated with the weight of the classifier error level, and $\beta = 1 - \alpha$ indicates the importance of the reduction feature.

3.2. ANFIS-Based Intrusion Detection Model

In this phase, the ANFIS model is applied to recognize intrusions. ANFIS, developed by Jang, integrates the benefits of neural networks (NNs) and fuzzy systems [27]. It employs an NN learning algorithm to automatically extract the input and rules of the sample dataset automatically, thus forming an ANFIS that includes five layers. Figure 2 demonstrates the infrastructure of ANFIS. ANFIS is a kind of artificial intelligence (AI) method which integrates the strength of fuzzy logic or artificial neural network (ANN) techniques. ANFIS is a hybrid method which utilizes a fuzzy inference system (FIS) embedding in an NN structure. The FIS module of ANFIS was utilized for representing the linguistic rules which describe the connections amongst inputs and outputs in a provided system. The NN module was utilized for adjusting the parameter of the FIS, namely the rule weights and the membership function parameters, based on the input–output training dataset. This enables the ANFIS to “learn” the relation between inputs and outputs, and to make prediction regarding new input value. ANFIS is used for a large number of tasks, involving time series prediction, control, and function approximation. It is an advanced type of fuzzy logic system that is able to learn and adapt by utilizing a learning mechanism. The learning method adjusts the parameter of the fuzzy system to enhance the performance based on the training dataset. The adaptive components of ANFIS make it especially pertinent for the application where the fundamental system is subjected to changes over time, viz., in modeling the dynamic system.

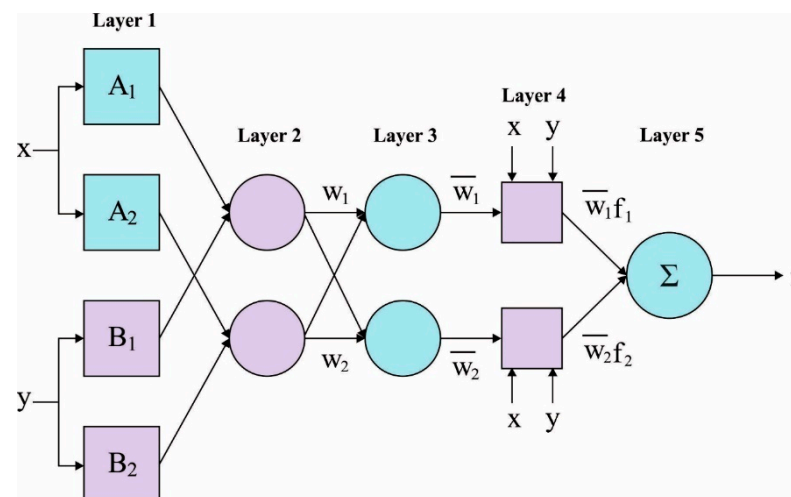


Figure 2. Structure of ANFIS [27].

It is an extension of the conventional fuzzy system and has an analogous structure that comprises the fuzzy inference, input and output layers. Briefly, the ANFIS integrates the capability to characterize vague or uncertain knowledge of the FIS with the capability to learn from information of the NNs. Therefore, it leverages the knowledge of human expertise and data from the environments to increase performance.

Layer 1: Each node is an adaptive node. The output of layer 1 is the fuzzy membership grade of inputs, as shown in Equations (14) and (15):

$$O_i^1 = \mu_{A_i}(x) \quad i = 1, 2 \tag{14}$$

$$O_i^1 = \mu_{B_{i-2}}(y) \quad i = 3, 4 \tag{15}$$

where $\mu_{A_i}(x), \mu_{B_{i-2}}(y)$ adopts the fuzzy membership function.

Layer 2: The nodes are fixed. They are labeled with M , specifying that they play simple multipliers as given in Equation (16):

$$O_i^2 = \omega_i = \mu_{A_i}(x)\mu_{B_i}(y) \quad i = 1, 2 \tag{16}$$

Layer 3: The nodes are fixed nodes. They are labeled with N , representing that they execute the normalization to the firing strength from the preceding layer as defined in Equation (17):

$$O_i^3 = \omega_i = \frac{\omega_i}{\omega_1 + \omega_2} \quad i = 1, 2 \tag{17}$$

Layer 4: The node is adaptive. The output of all nodes is the product of the normalized firing strength and the first-order polynomial (first-order Sugeno model), as given in Equation (18):

$$O_i^4 = \bar{\omega}_i f_i = \bar{\omega}_i(p_i x + q_i y + r_i) \quad i = 1, 2 \tag{18}$$

Layer 5: There is one fixed node labeled with S . This node is played as a summation of each incoming signal:

$$O_i^5 = \sum_{i=1}^2 \bar{\omega}_i f_i = \frac{\sum_{i=1}^2 \omega_i f_i}{\omega_1 + \omega_2} \tag{19}$$

The final output of ANFIS is defined in Equation (20):

$$\begin{aligned} f_{out} &= \bar{\omega}_1 f_1 + \bar{\omega}_2 f_2 = \frac{\omega_1}{\omega_1 + \omega_2} f_1 + \frac{\omega_2}{\omega_1 + \omega_2} f_2 \\ &= (\bar{\omega}x)p_1 + (\bar{\omega}x)q_1 + (\bar{\omega}x)r_1 + (\bar{\omega}x)p_2 + (\bar{\omega}x)q_2 + (\bar{\omega}x)r_2 \end{aligned} \tag{20}$$

3.3. Parameter Tuning Using JSSO Algorithm

Finally, the JSSO algorithm is used to optimize the MFs. The traditional SSO technique is found to be effective in resolving real-time optimization challenges. The presented method gives the productive means of changing parameters [28]. It gives the best exploration capability for the primary searching method. However, it has the limitation that it suffers from lower convergence speed and requires higher convergence time. This challenge minimizes the population range to escape from the local optima. The abovementioned problems are considered to develop the new method with the integration of JA. The traditional jaya algorithm (JA) has different features; it can be simple for the developer, and only one phase has fewer variables. The traditional JA is exploited for multi-objective optimized problems. It is capable of finding the optimal value in a lesser period. This feature is taken into account to perform the new technique named *J-SSO*. Rather than rotational movement, JA updating is used. In this work, the new equation is expressed for the gradient of the main function of SSO represented by $\nabla(of)$, as provided in Equation (21):

$$\nabla(of) = \frac{abs(of_{g_{best}} - of(j))}{of_{g_{best}}} \tag{21}$$

The term $of_{g_{bet}}$ represents a better solution’s fitness function, and the existing solution’s fitness function is denoted as 0. Therefore, if the attained solution reaches the values of $\nabla(of) < 0.5$, then the location updating can be performed by SSO forward movement. Otherwise, the location updating can be performed using *JA*. The pseudocode of *J-SSO* is demonstrated in Algorithm 1. The hybrid optimized technique is stimulated by integrating

different optimized techniques or rules. It is proved that the best outcomes are for the fixed search problem. The hybrid optimized algorithm is demonstrated to attain the best convergence using different optimization principles.

Algorithm 1: J-SSO

```

Initialization of the shark population
Set the determined user variables
Initialization phase counter as i = 1
for i = 1:i_max
    if  $\nabla(of) < 0.5$ 
        Upgrade velocity vector following SSO
        Upgrade location following SSO
    Else
        Upgrade solution following JA
    end if
end for i
Set i = i + 1
Selection of the better location of shark at a final phase that has the maximum value
End

```

The JSSO technique derives a fitness function to attain improved classification performance. It determines a positive integer to represent the better performance of the candidate solutions. In this study, the minimization of the classification error rate is considered the fitness function, as given in Equation (22):

$$fitness(x_i) = \frac{\text{number of misclassified samples}}{\text{Total number of samples}} * 100 \quad (22)$$

4. Results and Discussion

The proposed model is simulated using Python 3.6.5 tool on PC i5-8600k, GeForce 1050Ti 4GB, 16GB RAM, 250GB SSD, and 1TB HDD. The parameter settings are given as follows: learning rate, 0.01; dropout, 0.5; batch size, 5; epoch count, 50; and activation, ReLU. In this section, the intrusion classification performance of the IMFL-IDSCS model is inspected on the NSL-KDD dataset. Table 1 defines the detailed description of the dataset.

Table 1. Details of the dataset.

Class	No. of Samples
Denial of service attack (Dos)	45,927
Unauthorized access from a remote host (R2l)	995
Port monitoring or scanning (Probe)	11,656
Unauthorized local super user privileged access (U2r)	52
Not an Attack (Normal)	67,343
Total number of samples	125,973

4.1. Result Analysis

The confusion matrices of the IMFL-IDSCS model on intrusion classification are demonstrated in Figure 3. The outcomes exhibit that the IMFL-IDSCS approach properly categorized all intrusions and normal samples. With 80% of the training (TR) set, the IMFL-IDSCS model identified 36,302 into DoS, 549 into R2l, 8768 into Probe, 2 into U2R, and 53,414 into normal. Eventually, with 20% of testing (TS) set, the IMFL-IDSCS approach identified 9130 into DoS, 127 into R2l, 2324 into Probe, 3 into U2R, and 13,177 into normal. Meanwhile, with 70% of TR set, the IMFL-IDSCS technique identified 31,838 into DoS, 479 into R2l, 8007 into Probe, 0 into U2R, and 46,461 into normal. Finally, with 30% of the

TS set, the IMFL-IDSCS method identified 13,581 into DoS, 212 into R2L, 3407 into Probe, 0 into U2R, and 20,004 into normal.

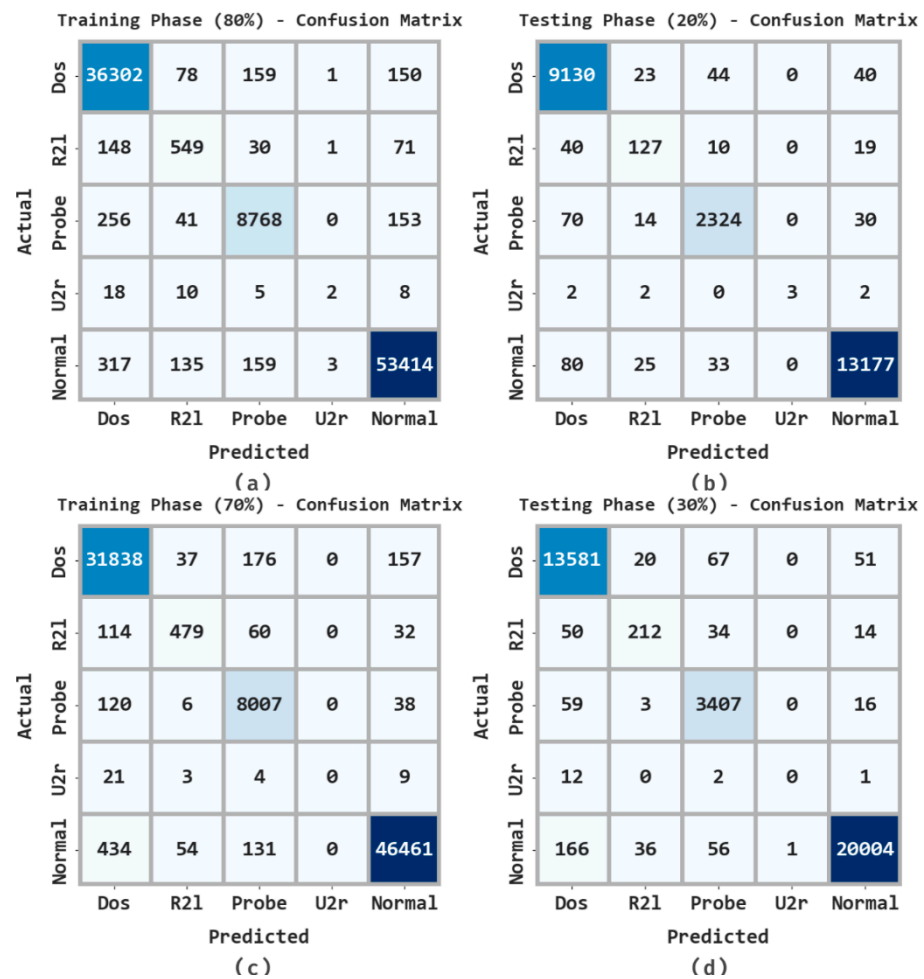


Figure 3. Confusion matrices of IMFL-IDSCS system (a,b) TR and TS databases of 80:20 and (c,d) TR and TS databases of 70:30.

Table 2 and Figure 4 provide the comprehensive intrusion recognition results of the IMFL-IDSCS method on 80% of TR and 20% of TS databases.

The figure demonstrates that the IMFL-IDSCS model showed maximum outcome under all classes. For sample, on 80% of the TR database, the IMFL-IDSCS approach provided an average $accu_y$ of 99.31%, $prec_n$ of 77.90%, $reca_l$ of 73.26%, F_{score} of 73.86%, and Mathew correlation coefficient (MCC) of 74.03%. Moreover, on 20% of the TS database, the IMFL-IDSCS approach rendered an average $accu_y$ of 99.31%, $prec_n$ of 92.03%, $reca_l$ of 78.25%, F_{score} of 81.80%, and MCC of 82.83%.

Table 3 and Figure 5 provide the overall intrusion recognition outcomes of the IMFL-IDSCS model on 70% of TR and 30% of TS databases. The outcomes exhibit that the IMFL-IDSCS method showed maximum outcome under all classes. For the sample, on 70% of the TR database, the IMFL-IDSCS approach presented an average $accu_y$ of 99.37%, $prec_n$ of 75.14%, $reca_l$ of 73.09%, F_{score} of 74%, and MCC of 73.56%. On 30% of the TS database, the IMFL-IDSCS technique offered an average $accu_y$ of 99.38%, $prec_n$ of 74.26%, $reca_l$ of 72.77%, F_{score} of 73.45%, and MCC of 73%.

The training accuracy (TACC) and validation accuracy (VACC) of the IMFL-IDSCS technique are inspected on ID performance in Figure 6. The figure displays that the IMFL-IDSCS approach showed improved performance with increased values of TACC and VACC. It is seen that the IMFL-IDSCS algorithm reached maximum TACC outcomes.

Table 2. Result analysis of IMFL-IDSCS system under 80:20 of TR/TS databases.

Class	$Accu_y$	$Prec_n$	$Reca_l$	F_{score}	MCC
Training Phase (80%)					
Dos	98.88	98.00	98.94	98.47	97.59
R2l	99.49	67.53	68.71	68.11	67.86
Probe	99.20	96.13	95.12	95.62	95.18
U2r	99.95	28.57	04.65	08.00	11.51
Normal	99.01	99.29	98.86	99.08	98.01
Average	99.31	77.90	73.26	73.86	74.03
Testing Phase (20%)					
Dos	98.81	97.94	98.84	98.39	97.45
R2l	99.47	66.49	64.80	65.63	65.37
Probe	99.20	96.39	95.32	95.85	95.42
U2r	99.98	100.00	33.33	50.00	57.73
Normal	99.09	99.31	98.96	99.14	98.18
Average	99.31	92.03	78.25	81.80	82.83

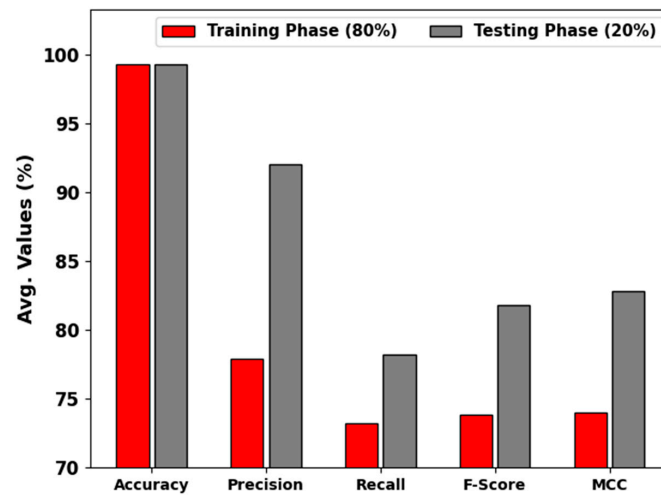


Figure 4. Average outcome of IMFL-IDSCS system under 80:20 of TR/TS databases.

Table 3. Result analysis of IMFL-IDSCS system under 70:30 of TR/TS databases.

Class	$Accu_y$	$Prec_n$	$Reca_l$	F_{score}	MCC
Training Phase (70%)					
Dos	98.80	97.88	98.85	98.36	97.42
R2l	99.65	82.73	69.93	75.79	75.89
Probe	99.39	95.57	97.99	96.77	96.44
U2r	99.96	00.00	00.00	00.00	00.00
Normal	99.03	99.49	98.69	99.09	98.06
Average	99.37	75.14	73.09	74.00	73.56
Testing Phase (30%)					
Dos	98.88	97.93	98.99	98.46	97.58
R2l	99.58	78.23	68.39	72.98	72.94
Probe	99.37	95.54	97.76	96.64	96.30
U2r	99.96	00.00	00.00	00.00	−0.01
Normal	99.10	99.59	98.72	99.15	98.19
Average	99.38	74.26	72.77	73.45	73.00

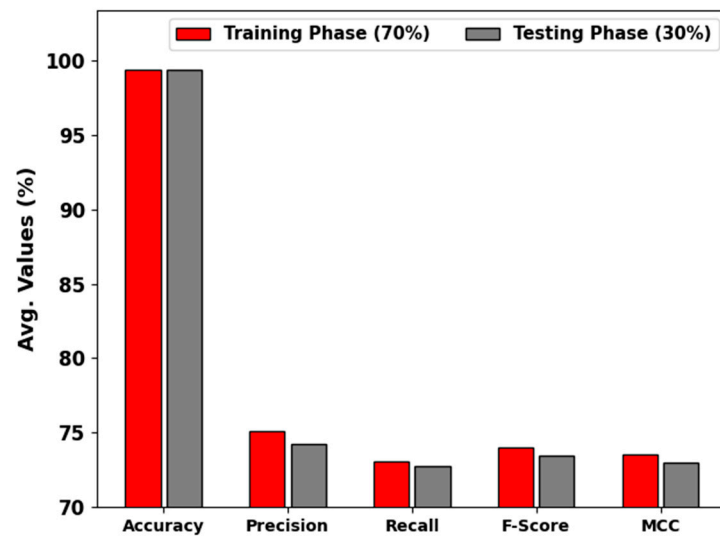


Figure 5. Average outcome of IMFL-IDSCS system under 70:30 of TR/TS databases.

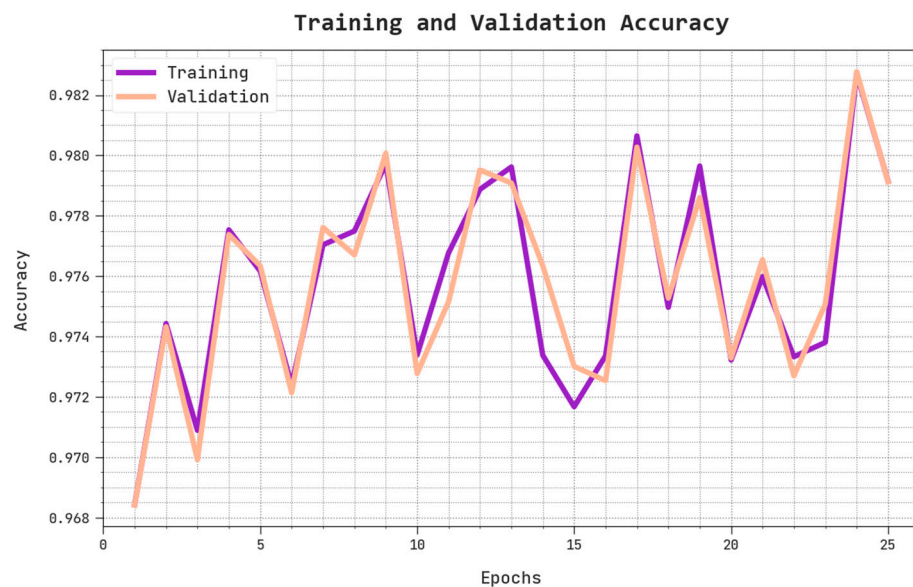


Figure 6. TACC and VACC analysis of IMFL-IDSCS system.

The training loss (TLS) and validation loss (VLS) of the IMFL-IDSCS methodology are tested on ID performance in Figure 7. The figure displays that the IMFL-IDSCS algorithm revealed better performance with the least values of TLS and VLS. It is seen that the IMFL-IDSCS technique reduced the VLS outcomes.

A clear precision–recall examination of the IMFL-IDSCS method in the test database is portrayed in Figure 8. The results specified that the IMFL-IDSCS methodology had enhanced precision–recall values in class labels.

The detailed ROC study of the IMFL-IDSCS method under the test database is given in Figure 9. The outcomes denoted by the IMFL-IDSCS approach showed its ability to classify different classes under the test database.

4.2. Discussion

A widespread comparative analysis of the IMFL-IDSCS approach with other CAD techniques is presented in Table 4 [17]. In Figure 10, a close inspection of the IMFL-IDSCS model with other models, such as LKM with optimal FLS, K-means-OFLS, fuzzy c-means with OFLS (FCM-OFLS), multilayer perceptron (MLP), and principal component analysis with NN (PCA-NN) in terms of $accu_y$ and F_{score} , is given. The results imply the enhanced

outcome of the IMFL-IDSCS model. For instance, based on $accu_y$, the IMFL-IDSCS model provided a maximum $accu_y$ of 99.31%, whereas the LKM-OFLS, K-means-OFLS, FCM-OFLS, MLP, and PCA-NN models gained reduced performance with $accu_y$ of 89.34%, 91.43%, 93.38%, 91.46%, and 90.08%, respectively. Additionally, based on F_{score} , the IMFL-IDSCS approach presented a maximum F_{score} of 81.80%, whereas the LKM-OFLS, K-means-OFLS, FCM-OFLS, MLP, and PCA-NN approaches gained reduced performance with F_{score} of 78.26%, 78.33%, 75.68%, 74.99%, and 77.54%, correspondingly.



Figure 7. TLS and VLS analysis of IMFL-IDSCS system.

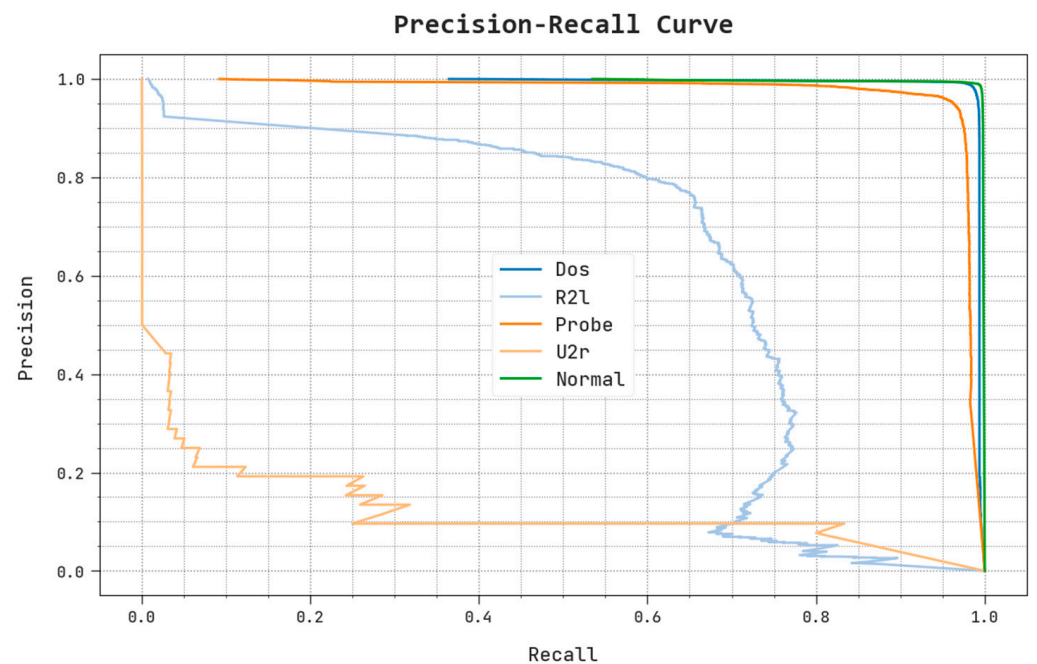


Figure 8. Precision–recall analysis of IMFL-IDSCS system.

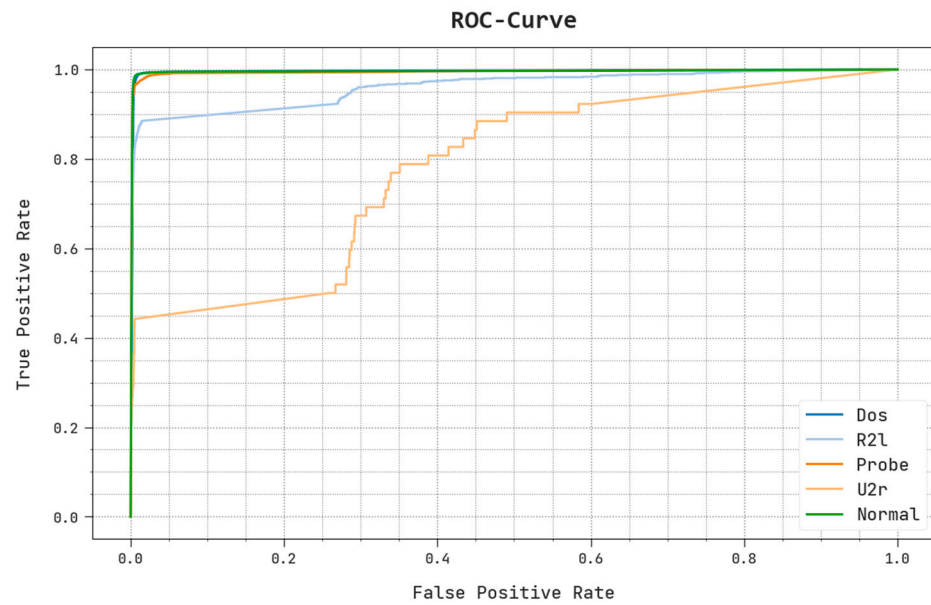


Figure 9. ROC curve analysis of IMFL-IDSCS system.

Table 4. Comparative analysis of IMFL-IDSCS system with other approaches.

Methods	$Accu_y$	$Prec_n$	$Reca_l$	F_{score}
IMFL-IDSCS	99.31	92.03	78.25	81.80
LKM-OFLS [17]	89.34	84.64	74.68	78.26
K-means-OFLS [17]	91.43	85.74	75.51	78.33
MLP [29]	91.46	86.61	76.76	74.99
PCA-NN [30]	90.08	84.56	76.06	77.54
FCM-OFLS [31]	93.38	82.74	74.43	75.68

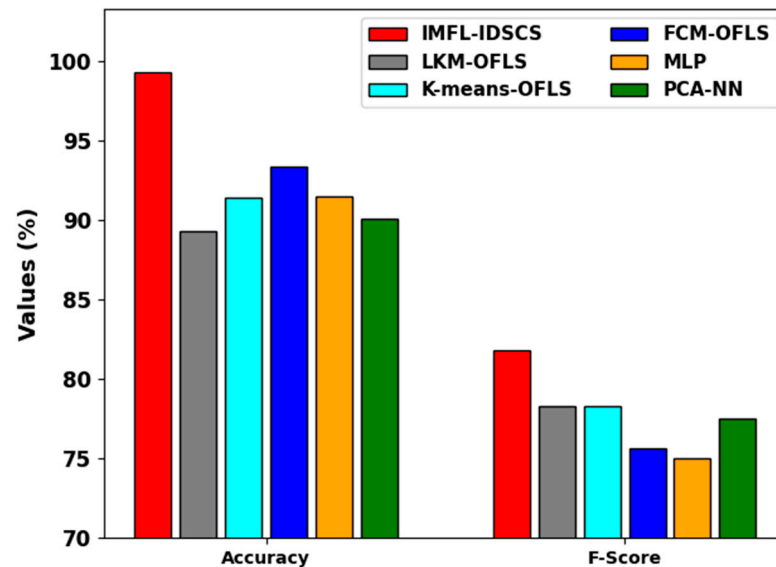


Figure 10. $Accu_y$ and F_{score} analysis of IMFL-IDSCS system with other approaches.

In Figure 11, a close inspection of the IMFL-IDSCS method with other ML methods in terms of $prec_n$ and $reca_l$ is given. The outcomes exhibit the enhanced outcome of the IMFL-IDSCS approach. For instance, based on $prec_n$, the IMFL-IDSCS methodology presented maximum $prec_n$ of 92.03%, whereas the LKM-OFLS, K-means-OFLS, FCM-OFLS, MLP, and PCA-NN methods obtained reduced performance with $prec_n$ of 84.64%, 85.74%, 82.74%, 86.61%, and 84.56%, correspondingly. Similarly, based on $reca_l$, the IMFL-IDSCS approach

rendered maximum $reca_1$ of 78.25%. In contrast, the LKM-OFLS, K-means-OFLS, FCM-OFLS, MLP, and PCA-NN techniques achieved reduced performance with $reca_1$ of 74.68%, 75.51%, 74.43%, 76.76%, and 76.06% correspondingly. These results highlight the superior outcomes of the IMFL-IDSCS model. The enhanced performance of the proposed model is due to the inclusion of the ECOA-FS model, which decreases the computational burden and increases the classification rate. In addition, the design of the JSSO algorithm for the ANFIS model helps in attaining maximum detection performance.

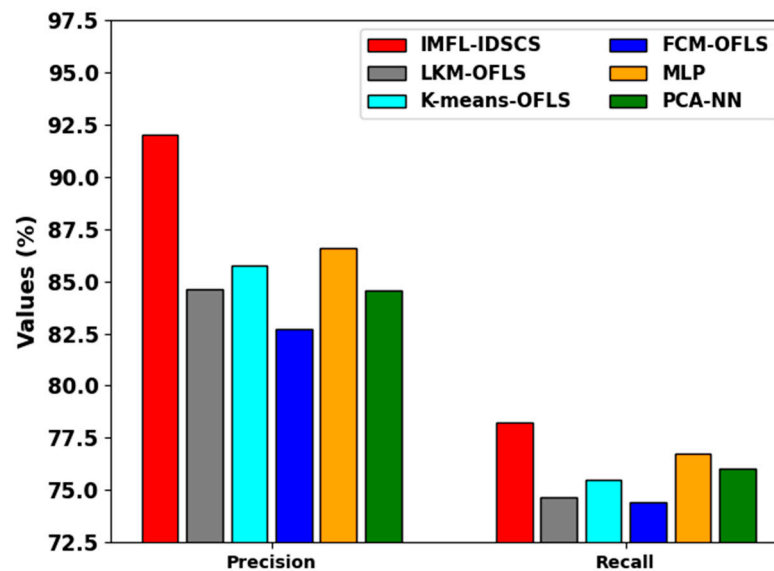


Figure 11. $Prec_n$ and $reca_1$ analysis of IMFL-IDSCS system with other approaches.

5. Conclusions

In this study, a new IMFL-IDSCS method was derived to detect intrusions in the distributed CC platform and secure it from probable threats. An individual sample of IDS was deployed for every client, and an individual controller was utilized for data management. In addition, the IMFL-IDSCS technique used the ECOA-FS method for choosing the best features. This was followed by the ANFIS model, which was applied to recognize intrusions. Finally, the JSSO algorithm was used to optimize the MFs. A widespread simulation analysis was performed to examine the enhanced outcomes of the IMFL-IDSCS technique. The extensive comparison study reported the enhanced outcomes of the IMFL-IDSCS model with maximum detection efficiency with accuracy of 99.31%, precision of 92.03%, recall of 78.25%, and F-score of 81.80%. In the future, the performance of the IMFL-IDSCS technique can be improvised by outlier removal approaches. In addition, the proposed model can be extended to the detection of malware, ransomware, and other kinds of attacks in the Android environment.

Author Contributions: Conceptualization, M.A.A. and M.E.; methodology, F.N.A.-W.; software, A.M.; validation, M.A.A., F.N.A.-W. and M.A.D.; formal analysis, A.M.H.; investigation, A.M.H.; resources, M.A.D.; data curation, A.M.; writing—original draft preparation, M.A.A., F.N.A.-W., M.A.D., M.E. and A.M.H.; writing—review and editing, M.A.A., A.M. and M.A.D.; visualization, M.A.D.; supervision, M.A.A.; project administration, A.M.H.; funding acquisition, M.A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Deanship of Scientific Research at Princess Nourah bint Abdulrahman University through the Research Groups Program Grant no. (RGP-1443-0051).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare that they have no conflict of interest to report regarding the present study.

References

1. Negi, P.S.; Garg, A.; Lal, R. Intrusion detection and prevention using honeypot network for cloud security. In Proceedings of the 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 29–31 January 2020; pp. 129–132.
2. Meryem, A.; Ouahidi, B.E. Hybrid intrusion detection system using machine learning. *Netw. Secur.* **2020**, *2020*, 8–19. [[CrossRef](#)]
3. Achbarou, O.; El Kiram, M.A.; Bourkhouk, O.; Elbouanani, S. A new distributed intrusion detection system based on multi-agent system for cloud environment. *Int. J. Commun. Netw. Inf. Secur.* **2018**, *10*, 526. [[CrossRef](#)]
4. Singh, D.A.A.G.; Priyadharshini, R.; Leavline, E.J. Cuckoo optimisation based intrusion detection system for cloud computing. *Int. J. Comput. Netw. Inf. Secur.* **2018**, *11*, 42–49.
5. Hatem, M.A.; Shaker, V.; Jabbarpour, M.R.; Jung, J.; Zarrabi, H. HIDCC: A hybrid intrusion detection approach in cloud computing. *Concurr. Comput. Pract. Exp.* **2018**, *30*, e4171. [[CrossRef](#)]
6. Ma, X.; Fu, X.; Luo, B.; Du, X.; Guizani, M. A design of firewall based on feedback of intrusion detection system in cloud environment. In Proceedings of the IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 9–13 December 2019; pp. 1–6.
7. Fontaine, J.; Kappler, C.; Shahid, A.; Poorter, E.D. Log-based intrusion detection for cloud web applications using machine learning. In *Advances on P2P, Parallel, Grid, Cloud and Internet Computing, Proceedings of the 14th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC-2019), Antwerp, Belgium, 7–9 November 2019*; Barolli, L., Hellinckx, P., Natwichaivon, J., Eds.; Springer: Cham, Switzerland, 2019; pp. 197–210.
8. Chang, V.; Golightly, L.; Modesti, P.; Xu, Q.A.; Doan, L.M.T.; Hall, K.; Boddu, S.; Kobusińska, A. A survey on intrusion detection systems for fog and cloud computing. *Future Internet* **2022**, *14*, 89. [[CrossRef](#)]
9. Luo, G.; Chen, Z.; Mohammed, B.O. A systematic literature review of intrusion detection systems in the cloud-based IoT environments. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6822. [[CrossRef](#)]
10. Krishnaveni, S.; Sivamohan, S.; Sridhar, S.; Prabhakaran, S. Network intrusion detection based on ensemble classification and feature selection method for cloud computing. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6838. [[CrossRef](#)]
11. Kannadhasan, S.; Nagarajan, R.; Thenappan, S. Intrusion detection techniques based secured data sharing system for cloud computing using msvm. In Proceedings of the 9th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 23–25 March 2022; pp. 50–56.
12. Guezzaz, A.; Asimi, A.; Asimi, Y.; Azrou, M.; Benkirane, S. A.; Asimi, A.; Asimi, Y.; Azrou, M.; Benkirane, S. A distributed intrusion detection approach based on machine learning techniques for a cloud security. In *Intelligent Systems in Big Data, Semantic Web and Machine Learning*; Gherabi, N., Kacprzyk, J., Eds.; Springer: Cham, Switzerland, 2021; Volume 1344, pp. 85–94.
13. Elmasry, W.; Akbulut, A.; Zaim, A.H. A design of an integrated cloud-based intrusion detection system with third party cloud service. *Open Comput. Sci.* **2021**, *11*, 365–379. [[CrossRef](#)]
14. Raj, M.G.; Pani, S.K. A meta-analytic review of intelligent intrusion detection techniques in cloud computing environment. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 206–217. [[CrossRef](#)]
15. Aldallal, A.; Alisa, F. Effective intrusion detection system to secure data in cloud using machine learning. *Symmetry* **2021**, *13*, 2306. [[CrossRef](#)]
16. Varun, P.; Ashokkumar, K. Intrusion detection system in cloud security using deep convolutional network. *Appl. Math. Inf. Sci.* **2022**, *16*, 581–588.
17. Shyla, S.I.; Sujatha, S.S. Cloud security: LKM and optimal fuzzy system for intrusion detection in cloud environment. *J. Intell. Syst.* **2020**, *29*, 1626–1642. [[CrossRef](#)]
18. Mishra, P.; Varadharajan, V.; Pilli, E.S.; Tupakula, U. Vmguard: A vmi-based security architecture for intrusion detection in cloud environment. *IEEE Trans. Cloud Comput.* **2018**, *8*, 957–971. [[CrossRef](#)]
19. Aoudni, Y.; Donald, C.; Farouk, A.; Sahay, K.B.; Babu, D.V.; Tripathi, V.; Dhabliya, D. Cloud security based attack detection using transductive learning integrated with Hidden Markov Model. *Pattern Recognit. Lett.* **2022**, *157*, 16–26. [[CrossRef](#)]
20. Sakr, M.M.; Tawfeeq, M.A.; El-Sisi, A.B. Network intrusion detection system based PSO-SVM for cloud computing. *Int. J. Comput. Netw. Inf. Secur.* **2019**, *11*, 22–29. [[CrossRef](#)]
21. Velliangiri, S.; Premalatha, J. Intrusion detection of distributed denial of service attack in cloud. *Clust. Comput.* **2019**, *22*, 10615–10623. [[CrossRef](#)]
22. Sathiyadhas, S.S.; Soosai Antony, M.C.V. A network intrusion detection system in cloud computing environment using dragonfly improved invasive weed optimization integrated Shepard convolutional neural network. *Int. J. Adapt. Control. Signal Process.* **2022**, *36*, 1060–1076. [[CrossRef](#)]
23. Srinivas, B.V.; Mandal, I.; Keshavarao, S. Virtual machine migration-based Intrusion Detection System in cloud environment using deep recurrent neural network. *Cybern. Syst.* **2022**, 1–21. [[CrossRef](#)]
24. Geetha, T.V.; Deepa, A.J. A FKPCA-GWO WDBILSTM classifier for intrusion detection system in cloud environments. *Knowl.-Based Syst.* **2022**, *253*, 109557.
25. Hizal, S.; Çavuşoğlu, Ü.; Akgün, D. A new deep learning based Intrusion Detection System for Cloud Security. In Proceedings of the 2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), Istanbul, Turkey, 11–13 June 2021.

26. Jia, H.; Sun, K.; Zhang, W.; Leng, X. An enhanced chimp optimization algorithm for continuous optimization domains. *Complex Intell. Syst.* **2022**, *8*, 65–82. [[CrossRef](#)]
27. Iwendi, C.; Mahboob, K.; Khalid, Z.; Javed, A.R.; Rizwan, M.; Ghosh, U. Classification of COVID-19 individuals using adaptive neuro-fuzzy inference system. *Multimed. Syst.* **2022**, *28*, 1223–1237. [[CrossRef](#)] [[PubMed](#)]
28. Ahamad, D.; Alam Hameed, S.; Akhtar, M. A multi-objective privacy preservation model for cloud security using hybrid Jaya-based shark smell optimization. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 2343–2358. [[CrossRef](#)]
29. Moradi, M.; Zulkernine, M. A neural network based system for intrusion detection and classification of attacks. In Proceedings of the IEEE International Conference on Advances in Intelligent Systems—Theory and Applications, Kirchberg, Luxembourg, 15–18 November 2004; pp. 15–18.
30. Mahmood, Z.; Agrawal, C.; Hasan, S.S.; Zenab, S. Intrusion detection in cloud computing environment using neural network. *Int. J. Res. Comput. Eng. Electron.* **2012**, *1*, 1–4.
31. Hajimirzaei, B.; Navimipour, N.J. Intrusion detection for cloud computing using neural networks and artificial bee colony optimization algorithm. *J. ICT Exp.* **2019**, *5*, 56–59. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.