Computers, Materials & Continua
DOI: 10.32604/cmc.2023.034176
Article

Tech Science Press

Optimal Deep Learning Driven Intrusion Detection in SDN-Enabled IoT Environment

Mohammed Maray[1], Haya Mesfer Alshahrani[2], Khalid A. Alissa[3], Najm Alotaibi[4], Abdulbaset Gaddah[5], Ali Meree[1,6], Mahmoud Othman[7] and Manar Ahmed Hamza[8,*]

[1]Department of Information Systems, College of Computer Science, King Khalid University, Abha, Saudi Arabia
[2]Department of Information Systems, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh, 11671, Saudi Arabia
[3]SAUDI ARAMCO Cybersecurity Chair, Networks and Communications Department, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam, 31441, Saudi Arabia
[4]Prince Saud AlFaisal Institute for Diplomatic Studies, Riyadh, Saudi Arabia
[5]Department of Computer Sciences, College of Computing and Information System, Umm Al-Qura University, Saudi Arabia
[6]Department of Computer Science, Warwick University, Covetry, UK
[7]Department of Computer Science, Faculty of Computers and Information Technology, Future University in Egypt New Cairo, 11835, Egypt
[8]Department of Computer and Self Development, Preparatory Year Deanship, Prince Sattam bin Abdulaziz University, AlKharj, Saudi Arabia
*Corresponding Author: Manar Ahmed Hamza. Email: ma.hamza@psau.edu.sa
Received: 07 July 2022; Accepted: 03 October 2022

Abstract: In recent years, wireless networks are widely used in different domains. This phenomenon has increased the number of Internet of Things (IoT) devices and their applications. Though IoT has numerous advantages, the commonly-used IoT devices are exposed to cyber-attacks periodically. This scenario necessitates real-time automated detection and the mitigation of different types of attacks in high-traffic networks. The Software-Defined Networking (SDN) technique and the Machine Learning (ML)-based intrusion detection technique are effective tools that can quickly respond to different types of attacks in the IoT networks. The Intrusion Detection System (IDS) models can be employed to secure the SDN-enabled IoT environment in this scenario. The current study devises a Harmony Search algorithm-based Feature Selection with Optimal Convolutional Autoencoder (HSAFS-OCAE) for intrusion detection in the SDN-enabled IoT environment. The presented HSAFS-OCAE method follows a three-stage process in which the Harmony Search Algorithm-based FS (HSAFS) technique is exploited at first for feature selection. Next, the CAE method is leveraged to recognize and classify intrusions in the SDN-enabled IoT environment. Finally, the Artificial Fish Swarm Algorithm (AFSA) is used to fine-tune the hyperparameters. This process improves the outcomes of the intrusion detection process executed by the CAE algorithm and shows the work's novelty. The proposed HSAFS-OCAE technique was experimentally validated under different aspects, and the comparative analysis results established the supremacy of the proposed model.

Keywords: Internet of things; SDN controller; feature selection; hyperparameter tuning; autoencoder

1 Introduction

Internet of Things (IoT) refers to a dynamic network that contains smartphones, sensor nodes, software, switches/routers and servers [1]. IoT was developed as a phenomenon. In this dynamic network, real-time data movements or activities are processed and sensed. The IoT network acts as a common platform for data transmission between the physical world and the Internet of conventional things. The idea of the IoT network has resulted in extensive consumption, production and processing of information [2]. The number of devices connected via the Internet has surpassed the global population and is expected to increase considerably in a few years [3,4].

On the other hand, the devices with constrained resources contribute to the security issues in the IoT network, which considerably augments it in terms of risks, vulnerabilities and threats [5]. In this background, an appropriate analysis of the information recorded in the IoT platforms assists in predictions and early detection of the threats [6]. The Intrusion Detection Systems (IDSs) can rectify malicious activities in the IoT network through real-time traffic analysis. The IDS takes measures to protect the system from getting damaged through attack detection and classification processes. Software-Defined Network (SDN) is a naïve concept in the networking domain that decouples the packet-forwarding plane and the control plane. The SDN approach provides a global view of the network and its centralized control [7,8]. Different authors have focused on developing novel IDSs for conventional networks in literature. These studies focused on the constrained devices in the IoT networks and the recognition of malicious activities in them [9]. Fig. 1 depicts the processes involved in the SDN environment. The concept of IDS-based software defines network systems as unique, especially in the IoT environment [6].

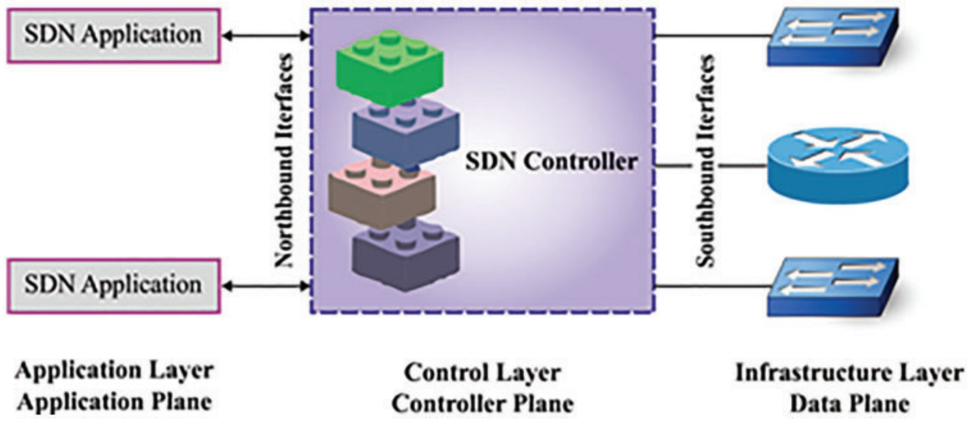

Figure 1: SDN environment

In conventional networks, network devices such as routers or switches can forward the data through different control mechanisms [10]. However, the SDN model conceived the network as a programmable entity and decouples both the control plane and the data plane. Here, the routers or switches simply act as a forwarding devices since the control module is operated from the centralized control. The SDN approach deals with the networks based on the abstraction of low-level functionality and maintains an Application Programmable Interface (API) to control the low-level devices [11]. In general, the SDN controller has a global view of the network, making it easy to configure it as and when required.

Furthermore, if there are any modifications to be done to the networking systems, the programmability feature of the SDN makes it relatively straightforward [9]. The security system and the associated features are programmed via an API module and are performed in a network by following the flow rules. These rules are administered through an OpenFlow protocol [12]. The programmability feature increases the flexibility of the network. As mentioned earlier, in case of a modification in the networking system, the control plane performs it instead of reconfiguring every device in the network individually.

In the study conducted earlier [13], an SDN-enabled deep-learning-driven structure was suggested for attack detection in the IoT environment. The existing classifiers such as the Cuda-Deep Neural Network (DNN), Cuda-Bidirectional Long Short Term Memory (Cu-BLSTM) and the Gated Recurrent Unit (Cu- DNNGRU) were efficiently leveraged for attack detection. In this study, a tenfold Cross-Validation (CV) was executed to display the unbiased results. Shu et al. [14] used a Deep Learning (DL) technique with generative adversary networks. They explored the distributed-SDN to devise a Collaborative IDS (CIDS) for the Vehicular Adhoc Network (VANET) model. In this model, multiple SDN controllers are allowed to train a global ID method mutually for the entire network without any direct interchange among the sub-network flows. Shrestha et al. [15] suggested a satellite-related, Unmanned Aerial Vehicle (UAV) 5G-network security method in which a Machine Learning (ML) technique was used to identify the cyberattacks and other vulnerabilities efficiently. The solution had two major phases: the model created for the intrusion detection process using several ML techniques and the application of the ML-related techniques in satellite or terrestrial gateways.

Aslam et al. [16] suggested an Adaptive ML-related SDN-enabled Distributed Denial of Service (DDoS) attack Detection and Mitigation (AMLSDM) structure. The suggested AMLSDM structure was an SDN-enabled security system for the IoT gadgets. An adaptive ML classification method was utilized to achieve fruitful mitigation and the detection of DDoS attacks. The presented structure used ML methods in an adaptive multi-layered feed forwarding method to identify the DDoS attacks in a successful manner. This was accomplished by probing the static attributes of the network traffic under review. Derhab et al. [17] suggested a security architecture in which the Software-Defined Network (SDN) model and the Blockchain technology were integrated. The suggested IDS security structure was created by integrating the K-Nearest Neighbor (KNN) technique and the Random Subspace Learning (RSL) technique. This was done to defend the forged commands that target the industrial control procedures and the Blockchain-related Integrity Checking System (BICS) that prevents the misrouting assault that meddles with the SDN-enabled industrial's OpenFlow regulations IoT mechanisms.

The current study devises a Harmony Search algorithm-based Feature Selection with Optimal Convolutional Autoencoder (HSAFS-OCAE) for the purpose of intrusion detection in the SDN-enabled IoT environment. The presented HSAFS-OCAE method follows a three-stage process in which the Harmony Search algorithm-based FS (HSAFS) technique is first exploited for feature selection. Next, the CAE algorithm is used for the recognition and classification of the intrusions in the SDN-enabled IoT environment. Finally, the Artificial Fish Swarm Algorithm (AFSA) is used to fine-tune the hyperparameters to boost the intrusion detection performance of the CAE algorithm.

The proposed HSAFS-OCAE technique was experimentally validated, and the results were assessed under different measures.

2 The Proposed Model

In this study, a new HSAFS-OCAE model has been proposed for a proficient recognition of intrusions in the SDN-enabled IoT environment. The presented HSAFS-OCAE model follows a three-stage process in which the HSAFS technique is exploited at first for feature selection. Next, the CAE approach is leveraged to recognize and classify intrusions in the SDN-enabled IoT environment. Finally, the AFSA-based hyperparameter fine-tuning process is executed to boost the intrusion detection performance of the CAE model. Fig. 2 shows the overall block diagram of the HSAFS-OCAE approach.

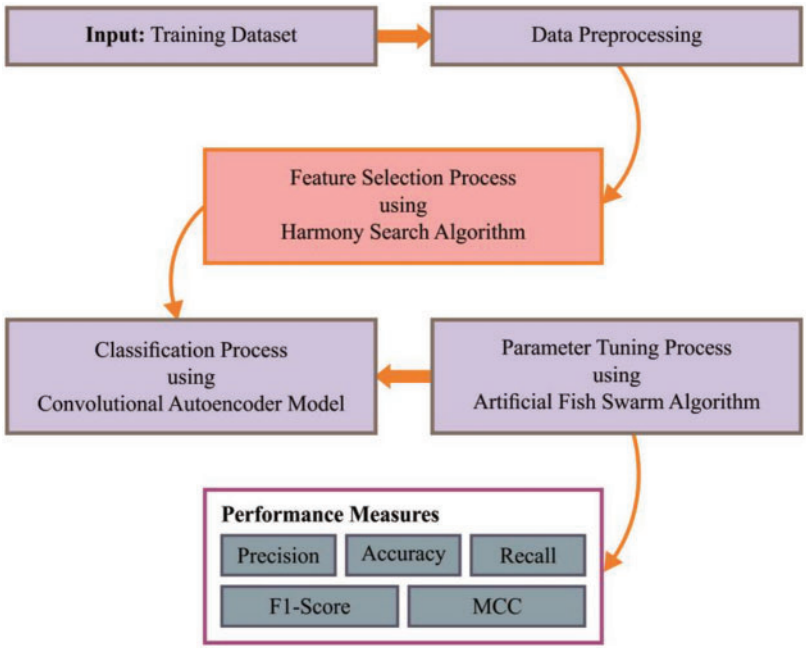
Input: Training Dataset
Data Preprocessing
Feature Selection Process using Harmony Search Algorithm
Classification Process using Convolutional Autoencoder Model
Parameter Tuning Process using Artificial Fish Swarm Algorithm
Performance Measures
Precision
Accuracy
Recall
F1-Score
MCC

Figure 2: Overall block diagram of the HSAFS-OCAE approach

2.1 *Design of HSAFS Technique*

In this study, the HSAFS technique is exploited for feature selection. In the HSA method, the optimal solution (or component from the solution space) is named 'harmony' viz., n-dimensional real vector. An arbitrary value is assigned to the initial population and is loaded from the Harmony Memory (HM) [18]. Then, a novel candidate (following iteration or generation) is evaluated and the harmony is generated based on the component from HM, either by altering the pitch or through an arbitrary selection of the element from HM. Afterwards, the component from the harmony memory and the newly-evaluated candidate harmony are correlated with the least HM vector. This process is repeated to satisfy the ending criteria. The parameters of the HS optimization approach are as follows (i) Pitch Adjusting Rate (PAR) (ii) the size of the HM (iii) distance bandwidth (BW) (iv) HM Consideration Rate (HMCR), and (v) the number of iterations or improvisations (NI). It is crucial to configure the HM module (HMS Vector). Assume that $xi = \{xi(1), xi(2), \ldots xi(n)\}$ characterizes an arbitrarily-evaluated HM vector: $xi(k) = X_l(k)+(X_u(k) - X_l(k))*rand\,(0,1)\ \mathit{for}\ k = 1, 2, \ldots, nandi = 1, 2, \ldots, HMS$ i.e., the length of the HM. Hence, the upper and the lower bounds of the searching space are characterized by $X_l(k)$ and $X_u(k)$, respectively. The HM matrix for every component is a harmony vector.

$$HM = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{HMS} \end{bmatrix} \tag{1}$$

In HM matrix, i.e., x_{new}, a harmony vector is created by three functions such as (i) the pitch adjustment (ii) memory consideration and (iii) an arbitrary re-initialization. Initially, a decision value $x_{new}(1)$ is arbitrarily chosen in the harmony i.e., $\{x_1(1), x_2(1) \ldots x_{HMS}(1)\}$. To select $x_{new}(1)$, an arbitrary number $r1$ is used in the range of 0 and 1. Once an arbitrary number is selected within the HMCR, $x_{new}(1)$ is generated by considering the memory. Else $x_{new}(1)$ is captured within the searching range $[X_l(k), X_u(k)]$. Likewise, the values are elected for the series, $x_{new}(2), x_{new}\,(3), \ldots, x_{new}(n)$. The two functions, such as (i) an arbitrary re-initialization and (ii) the memory consideration, are defined using the following equation.

$$x_{new}(k) = \begin{cases} x_i \in \{x_1(k) \cdots\cdots\cdots\cdots .x_{HMS}(k)\} \\ X_l(k) + (X_u(k) - X_l(k)) * rand\,(0,1) \\ \mathit{with\ probability\ of}\ 1{-}HMCR \end{cases} \tag{2}$$

The newly-generated x_{new} value is inspected while, on the other hand, the required value is either pitch-adjusted or not. To address these problems, a PAR entity is proposed by combining the Bandwidth Factor (BF) and the frequency. These two factors are adjusted to get a novel x_{new} value for the selected HM solution during the local search process. The pitch-adapted novel solution x_{new} is calculated as $x_{new}(k) + / - rand\,(0,1).\ BW$ with a probability of PAR. This PAR is mostly similar to the mutation process from the evolutionary bio-inspired technique. The range of the PAR is limited to $[X_l(k), X_u(k)]$.

At last, x_{new}, i.e., the newly-generated harmony vector, is upgraded or estimated as a novel component to fit itself between the x_{new} value and the worst harmony vector x_w in the HM. Consequently, x_w is altered by x_{new}, whereas the established part of the HM is identified to maximize the objective function (inter-class variance) for which the HSA is employed.

Either harmony or the solution employs a k element to decide the optimization system. The threshold value th_k is applied upon a multi-level segmentation process as formulated herewith.

$$HM = [x_1, x_2 \cdots ., x_{1HMS}]^T, \quad x_i = [th_1,\ th_2 \cdots\cdots th_k] \tag{3}$$

Here, T characterizes the transpose of the matrix, and the maximum size of the HM is denoted by the HMS technique. Each element from the HM is denoted by xi which lies in the range of $[0, k]$. The HSAFS method derives a Fitness Function (FF) to handle the trade-off between the chosen feature count and the classification accuracy with the help of the selected features. FF is determined as given below.

$$Fitness = \alpha\gamma_R\,(D) + \beta\frac{|R|}{|C|} \tag{4}$$

Here, $\gamma_R\,(D)$ represents the error rate of the presented classifier. $|R|$ indicates the selected subset and $|C|$ denotes the total feature count and α and β correspond to the constants.

2.2 *Design of CAE Classification Model*

In this stage, the CAE technique is used to recognize and classify intrusions in the SDN-enabled IoT environment. Autoencoder is a self-supervised learning mechanism that exploits the Neural Network (NN) for representative learning [19]. Representative learning is a method in which a scheme learns how to encode the input dataset. The Autoencoder (AE) approach maps the input datasets to a compressed domain demonstration or a low-dimension space. In the current study, a bottleneck is proposed in which an algorithm is enforced to learn how to demonstrate the compressed domains of the input dataset. In general, the AE approach encompasses four elements such as the Reconstruction Loss, Encoder Network (EN), Bottleneck Layer and Decoder Network (DN). Encoder Network is a NN system that encodes the input dataset to a compressed domain. The bottleneck layer is the final layer of the Encoder Network, and its output is called the encoded input data.

$$X_{ei+1} = f_{e_j}\left(WV_{e_j}^{T} X_{e_i} + b_{e_j}\right) \quad \forall i = 0, 1, 2, \ldots, N \tag{5}$$

In Eq. (5), X_{ei} corresponds to the input for the i^{th} layer, X denotes the output of the i^{th} layer of EN, WV_{ei} shows the weight vector for the i^{th} layer, b_{ei} indicates the bias for the i^{th} layer, and f_{ei} indicates the activation function for the i^{th} layer.

$$X_{di+1} = f_{d_i}\left(1WT_{d_i} X_{d_i} + b_{d_i}\right) \quad \forall i = 0, 1, 2, \ldots, N \tag{6}$$

In Eq. (6), X_{di} denotes the input for the i^{th} layer of the DN method, $Xdi + 1$ signifies the output of the i^{th} layer, w_{di} shows the weight vector for the i^{th} layer, b_{di} refers to the bias of the i^{th} layer, and f_{di} illustrates the activation function for the i^{th} layer. The variance between the original dataset X^O and the reconstructed dataset X^R is called the Reconstructed Loss. On the other hand, the Binary Cross-Entropy (BCE) and the Mean Squared Error (MSE) Loss are the two commonly-applied loss functions in the calculation of Reconstructed Loss. Here, D indicates the count of the instances in a dataset in which the AE is employed.

$$MSE\left(X^O, X^R\right) = \frac{1}{D}\sum\nolimits_{j=1}^{D}\left(X_j^O - X_j^R\right)^2 \tag{7}$$

$$BCE\left(X^O, X^R\right) = -\frac{1}{D}\sum\nolimits_{j=1}^{D} X_j^O \cdot \log X_j^R + \left(1 - X_j^O\right) \cdot \log\left(1 - X_j^R\right) \tag{8}$$

2.3 *Algorithmic Process of AFSA Based Hyperparameter Tuning*

Finally, the AFSA hyperparameter tuning process is performed to boost the intrusion detection performance of the CAE model. The AFSA model is a kind of swarm intelligence technique that is simulated from the behaviour of the animals [20]. In this method, the fish's clustering, collision and foraging behaviours are simulated along with its collective support in a fish swarm to realize the optimum global point. In this Artificial Fish (AF) technique, the maximum distance that passes through is referred to as *Step*. The apparent distance that passes over the AF is referred to as *Visual*.

Further, the retry quantity is characterized by $Try - Number$. The crowd quantity factor is characterized by η. The place of a particular AF is referred to as the resultant vector, $X = (X_1, X_2, \ldots, X_n)$ and the distance between i and j i.e., AF is determined using $d_{ij} = \left|X_i - X_j\right|$. The performance function of the AF is described in the following order i.e., prey, random, follow and swarm.

Given that a fish observes the food via its eyes, the existing position is denoted by X_i along with an arbitrarily-chosen position i.e., X_j within a perceptive range as given below.

$$X_j = X_i + Visual \times rand\,(0 \sim 1) \tag{9}$$

In Eq. (9), *rand* (0-1) characterizes an arbitrary value between [0, 1]. If $Y_i > Y_j$, the fish moves in the direction. Then, the technique arbitrarily chooses a novel position X_j to check whether it fulfils the motion condition as follows.

$$X_i^{t+1} = X_i^t + \frac{X_j - X_i^t}{\|X_j - X_i^t\|} \times Step \times rand\,(0 \sim 1) \tag{10}$$

If it does not fulfil the motion condition *Try_Number* times, an arbitrary motion is created as given below.

$$X_i^{t+1} = X_i^t + Visual \times rand\,(0 \sim 1) \tag{11}$$

In order to avoid the over-crowding issues, an artificial existing location X_i is set. Followed by, the fish amount in the n_f company and X_c center in the area ($d_{ij} < Visual$) are determined. If $Y_c/n_f < \eta \times Y_i$, the companion's location is characterized by the optimal food count and low-crowding. Consequently, the fish moves towards the companion area i.e., centre of the location.

$$X_i^{t+1} = X_i^t + \frac{X_c - X_i^t}{\|X_c - X_i^t\|} \times Step \times rand\,(0 \sim 1) \tag{12}$$

Then, it begins to follow the prey's behaviour.

The existing place of the AF swarm is referred to as X_i. The swarm describes the company Y_j as X_j in the area ($d_{ij} < Visual$). If $Y_j/n_f < \eta \times Y_i$, then the location of the company embodies an optimal food count with a less crowd.

$$X_i^{t+1} = X_i^t + \frac{X_j - X_i^t}{\|X_j - X_i^t\|} \times Step \times rand\,(0 \sim 1) \tag{13}$$

It allows the AF to accomplish the company as well as the food over a large region. The location is chosen in a random manner based on which the AF moves towards them.

Using the search region of D dimension, the extremely-possible distance between the two AFs is applied to vigorously limit the *Visual* & *Step* of the AF as given below.

$$MaxD = \sqrt{(x_{\max} - x_{\min})^2 \times D} \tag{14}$$

In Eq. (14), $x_{\min}$ and $x_{\max}$ signify the lower and the upper limits. D denotes the dimension of the searching region.

3 Results and Discussion

The performance of the proposed HSAFS-OCAE model was experimentally validated using a dataset with 84,792 samples under six class labels as tabulated in Table 1.

Table 1: Dataset details

Label	Class	No. of instances
C-1	Benign	69654
C-2	Bot	2977
C-3	Brute force-FTP	3066
C-4	DDoS-Loic-UDP	3015
C-5	DDoS-Hoic	3037
C-6	Infiltration	3043
Total number of instances		84792

The classification results accomplished by the proposed HSAFS-OCAE model are portrayed as confusion matrices in Fig. 3. The figure reports that the proposed HSAFS-OCAE model effectually recognized and classified all the input data under six class labels.

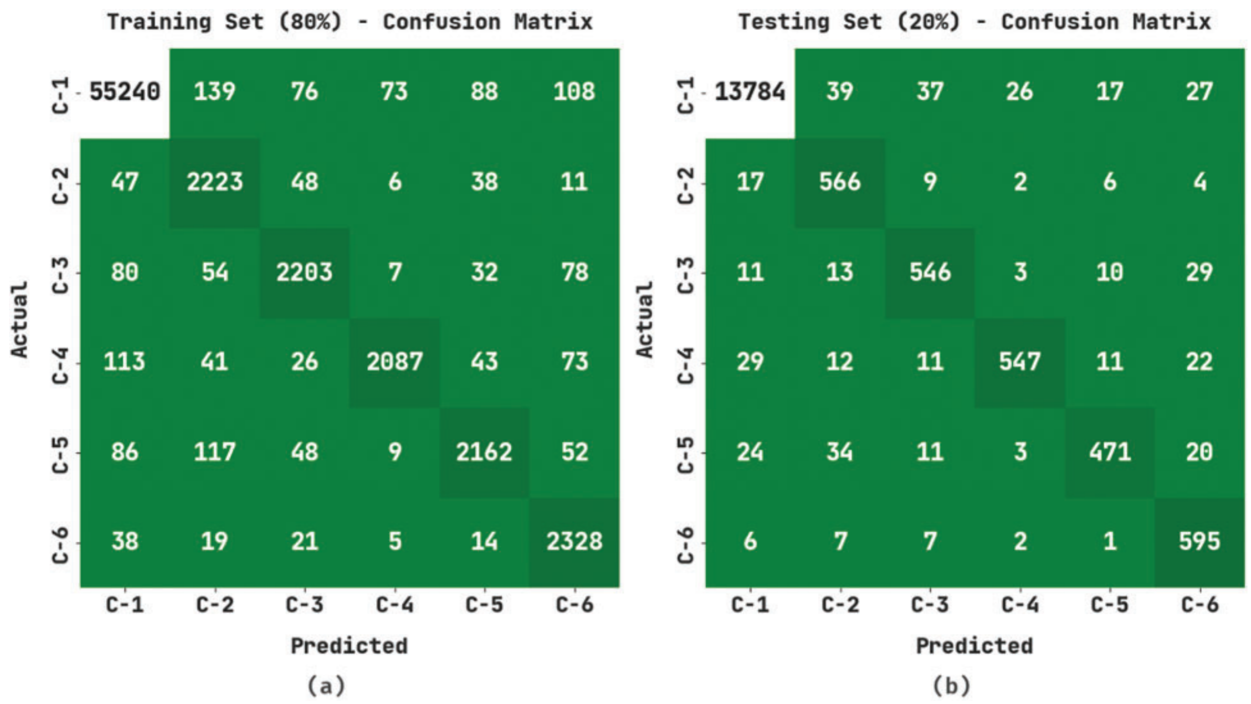

Figure 3: (Continued)

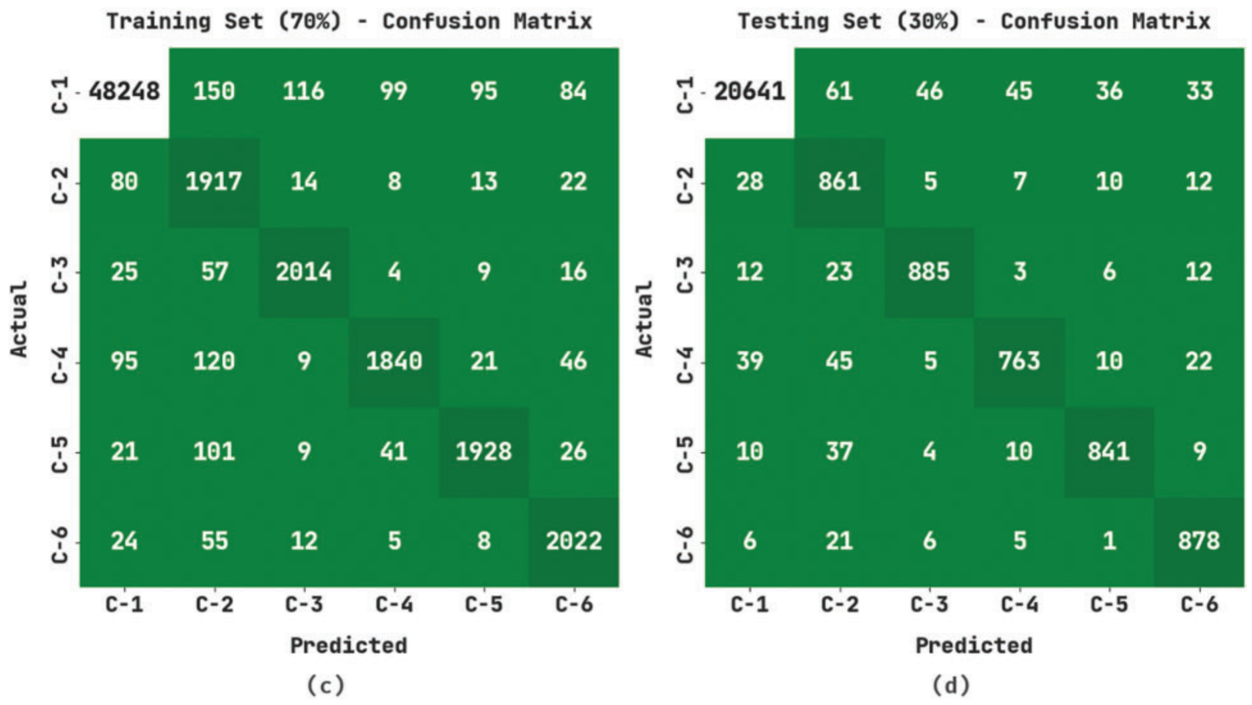

Figure 3: Confusion matrices of the HSAFS-OCAE approach (a) 80% of TR data, (b) 20% of TS data, (c) 70% of TR data, and (d) 30% of TS data

Table 2 and Fig. 4 highlight the classification outcomes achieved by the proposed HSAFS-OCAE model on 80% of the TR data. The results imply that the proposed HSAFS-OCAE model accomplished effectual outcomes under each class. For instance, on 80% of the TR data, the HSAFS-OCAE model gained an $accu_y$ of 98.75%, $prec_n$ of 99.35%, $reca_l$ of 99.13%, F_{score} of 99.24% and an MCC of 95.76%. Then, when using 80% of the TR data, the presented HSAFS-OCAE method obtained an $accu_y$ of 99.31%, $prec_n$ of 90.96%, $reca_l$ of 89.77%, F_{score} of 90.36% and an MCC of 90%. When using 80% of the TR data, the proposed HSAFS-OCAE model attained an $accu_y$ of 99.38%, $prec_n$ of 87.85%, $reca_l$ of 96%, F_{score} of 91.74% and an MCC of 91.52%.

Table 2: Analytical results of the HSAFS-OCAE approach on 80% of the TR dataset and 20% of the TS dataset

Labels	Accuracy	Precision	Recall	F-Score	MCC
		Training set (80%)			
C-1	98.75	99.35	99.13	99.24	95.76
C-2	99.23	85.73	93.68	89.53	89.23
C-3	99.31	90.96	89.77	90.36	90.00
C-4	99.42	95.43	87.58	91.33	91.12
C-5	99.22	90.95	87.39	89.14	88.75
C-6	99.38	87.85	96.00	91.74	91.52
Average	99.22	91.71	92.26	91.89	91.06

(Continued)

Table 2: Continued

Labels	Accuracy	Precision	Recall	F-Score	MCC
Testing set (20%)					
C-1	98.63	99.37	98.95	99.16	95.36
C-2	99.16	84.35	93.71	88.78	88.48
C-3	99.17	87.92	89.22	88.56	88.14
C-4	99.29	93.83	86.55	90.04	89.75
C-5	99.19	91.28	83.66	87.30	86.97
C-6	99.26	85.37	96.28	90.49	90.29
Average	99.12	90.35	91.39	90.72	89.83

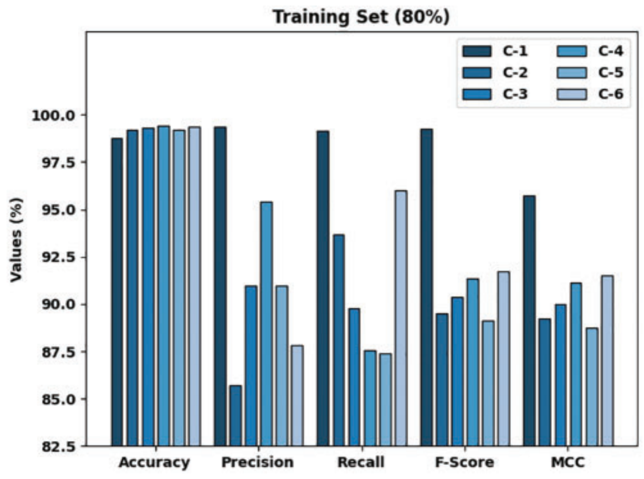

Figure 4: Analytical results of the HSAFS-OCAE approach on 80% of the TR data

Fig. 5 signifies the classification outcomes attained by the proposed HSAFS-OCAE technique on 20% of the TS data. The results denote that the HSAFS-OCAE method accomplished effectual outcomes under each class. For example, on 20% of the TS data, the HSAFS-OCAE model gained an $accu_y$ of 98.63%, $prec_n$ of 99.37%, $reca_l$ of 98.95%, F_{score} of 99.16% and an MCC of 95.36%. Also, on 20% of the TS data, the proposed HSAFS-OCAE technique gained an $accu_y$ of 99.17%, $prec_n$ of 87.92%, $reca_l$ of 89.22%, F_{score} of 88.56% and an MCC of 88.14%. Conversely, on 20% of the TS data, the proposed HSAFS-OCAE model gained an $accu_y$ of 99.26%, $prec_n$ of 85.37%, $reca_l$ of 96.28%, F_{score} of 90.49% and an MCC of 90.29%.

Table 3 and Fig. 6 highlight the classification outcomes of the proposed HSAFS-OCAE model on 70% of the TR dataset. The results infer that the proposed HSAFS-OCAE method accomplished effectual outcomes under each class. For example, on 70% of the TR dataset, the presented HSAFS-OCAE model attained an $accu_y$ of 98.67%, $prec_n$ of 99.49%, $reca_l$ of 98.89%, F_{score} of 99.19% and an MCC of 95.52%. Also, on 70% of the TR dataset, the proposed HSAFS-OCAE model gained an $accu_y$ of 99.54%, $prec_n$ of 92.64%, $reca_l$ of 94.78%, F_{score} of 93.70% and an MCC of 93.47%. Conversely, on 70% of the TR dataset, the proposed HSAFS-OCAE model reached an $accu_y$ of 99.50%, $prec_n$ of 91.25%, $reca_l$ of 95.11%, F_{score} of 93.14% and an MCC of 92.90%.

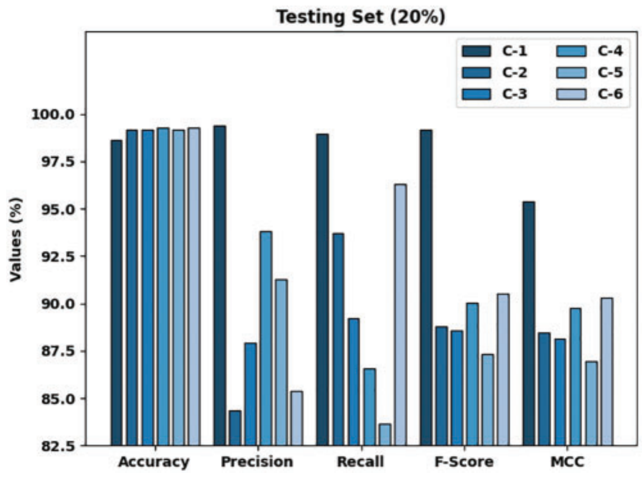

Figure 5: Analytical results of the HSAFS-OCAE approach on 20% of the TS data

Table 3: Analytical results of the HSAFS-OCAE approach on 80% of the TR dataset and 20% of the TS dataset

Labels	Accuracy	Precision	Recall	F-Score	MCC
Training set (70%)					
C-1	98.67	99.49	98.89	99.19	95.52
C-2	98.96	79.87	93.33	86.08	85.82
C-3	99.54	92.64	94.78	93.70	93.47
C-4	99.25	92.14	86.34	89.15	88.81
C-5	99.42	92.96	90.69	91.81	91.52
C-6	99.50	91.25	95.11	93.14	92.90
Average	99.22	91.39	93.19	92.18	91.34
Testing set (30%)					
C-1	98.76	99.54	98.94	99.24	95.85
C-2	99.02	82.16	93.28	87.37	87.05
C-3	99.52	93.06	94.05	93.55	93.30
C-4	99.25	91.60	86.31	88.88	88.53
C-5	99.48	93.03	92.32	92.67	92.40
C-6	99.50	90.89	95.75	93.26	93.03
Average	99.25	91.71	93.44	92.49	91.69

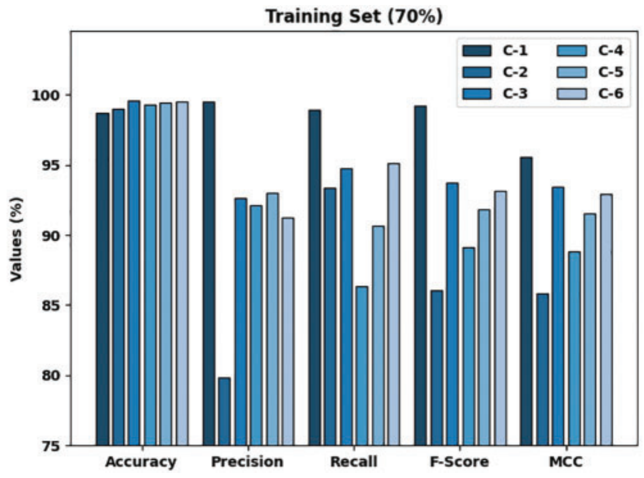

Figure 6: Analytical results of the HSAFS-OCAE approach on 70% of the TR dataset

Fig. 7 highlights the classification outcomes achieved by the proposed HSAFS-OCAE model on 30% of the TS dataset. The results infer that the proposed HSAFS-OCAE model established effectual outcomes under each class. For example, on 30% of the TS dataset, the HSAFS-OCAE algorithm reached an $accu_y$ of 98.76%, $prec_n$ of 99.54%, $reca_l$ of 98.94%, F_{score} of 99.24% and an MCC of 95.85%. Then, on 30% of the TS data, the proposed HSAFS-OCAE model gained an $accu_y$ of 99.52%, $prec_n$ of 93.06%, $reca_l$ of 94.05%, F_{score} of 93.55% and an MCC of 93.30%. When using 30% of the TS dataset, the presented HSAFS-OCAE model obtained an $accu_y$ of 99.50%, $prec_n$ of 90.89%, $reca_l$ of 95.75%, F_{score} of 93.26% and an MCC of 93.03%.

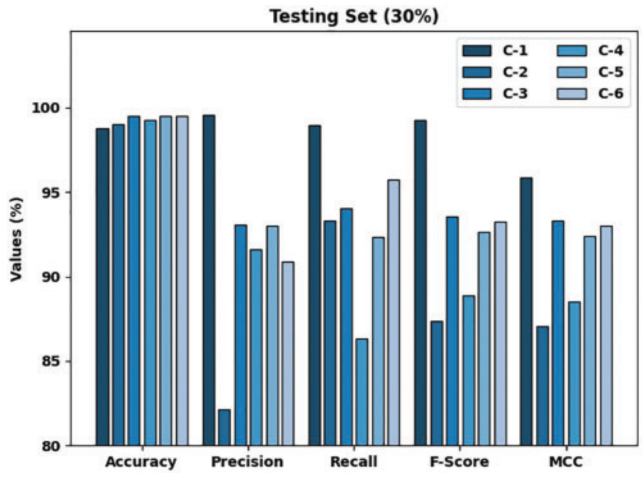

Figure 7: Analytical results of the HSAFS-OCAE approach on 30% of the TS data

Both Training Accuracy (TA) and Validation Accuracy (VA) values, acquired by the proposed HSAFS-OCAE method on test dataset, are shown in Fig. 8. The experimental outcomes imply that the proposed HSAFS-OCAE technique gained the maximal TA and VA values while the VA values were higher than the TA values.

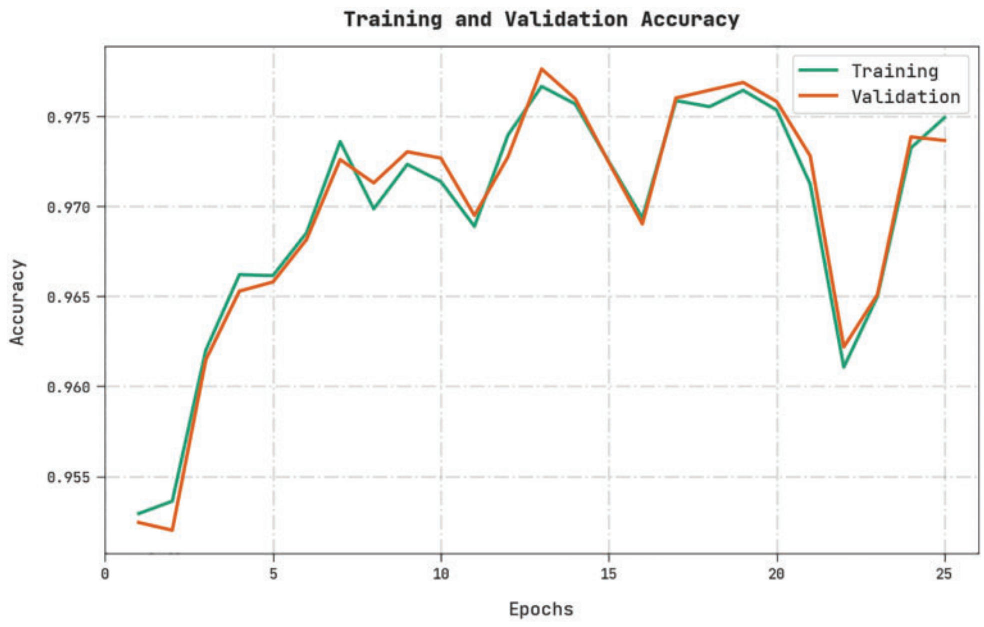

Figure 8: TA and VA analyses results of the HSAFS-OCAE methodology

Both Training Loss (TL) and Validation Loss (VL) values, attained by the HSAFS-OCAE approach on test dataset, are displayed in Fig. 9. The experimental outcomes denote that the proposed HSAFS-OCAE algorithm exhibited the least TL and VL values while the VL values were lower than the TL values.

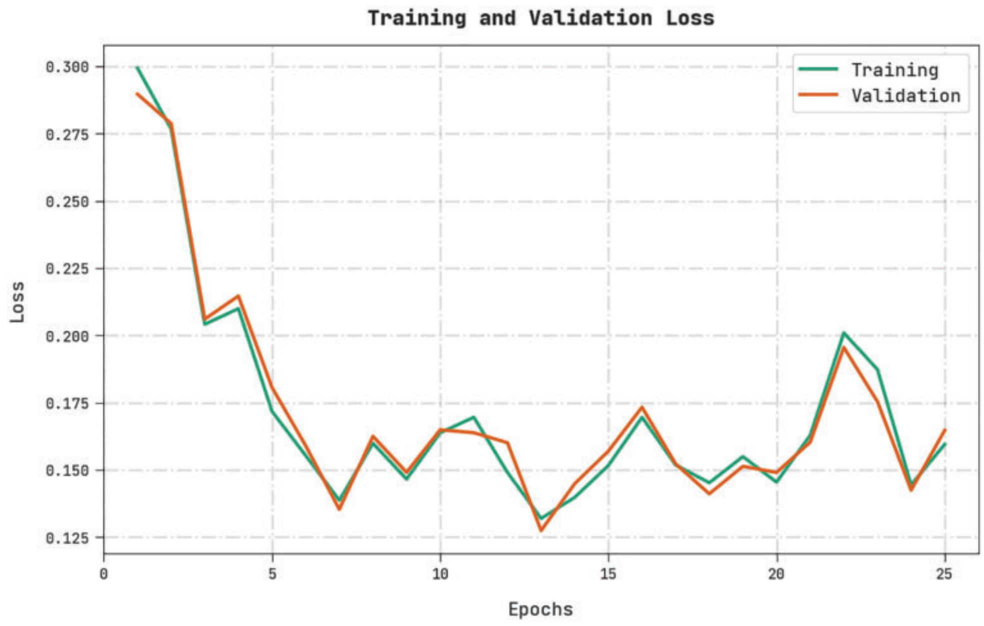

Figure 9: TL and VL analyses results of the HSAFS-OCAE methodology

A clear precision-recall analysis was conducted upon the HSAFS-OCAE method using the test dataset and the results are shown in Fig. 10. The figure represents that the proposed HSAFS-OCAE method produced enhanced precision-recall values under all the classes.

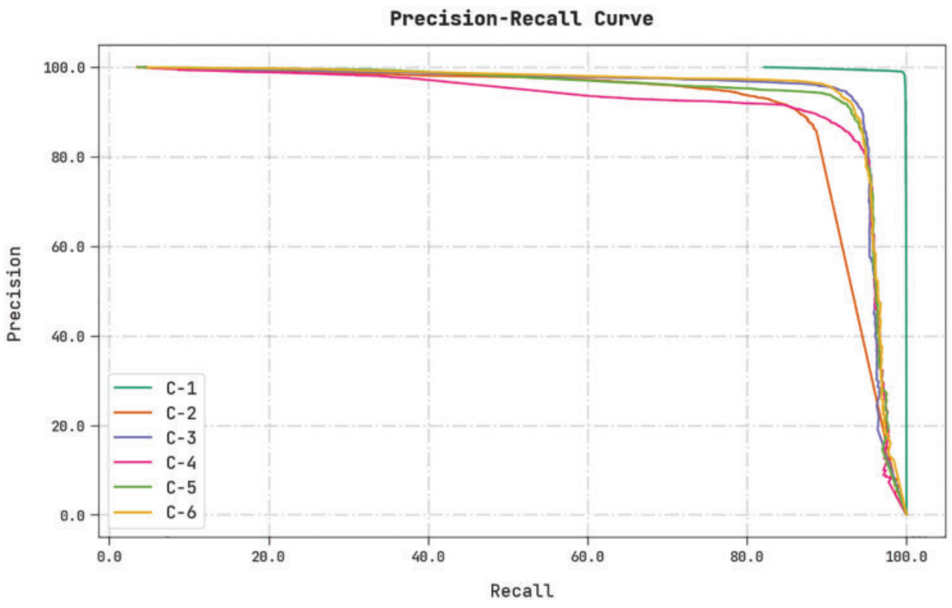

Figure 10: Precision-recall curve analysis results of the HSAFS-OCAE methodology

A brief ROC analysis was conducted upon the HSAFS-OCAE method using the test dataset and the results are depicted in Fig. 11. The results infer that the proposed HSAFS-OCAE technique established its supremacy in categorizing the test dataset under distinct classes.

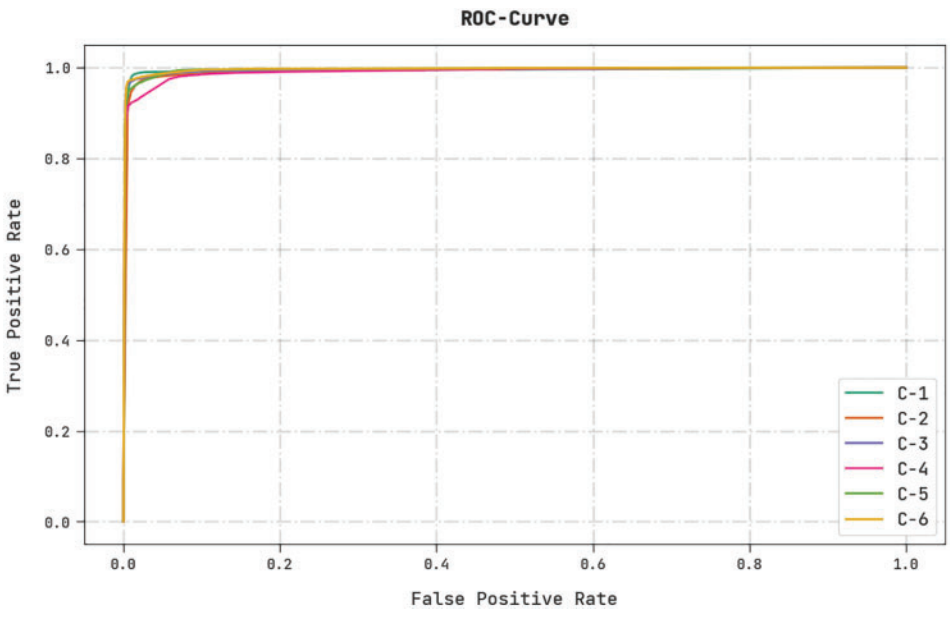

Figure 11: ROC curve analysis results of the HSAFS-OCAE methodology

To showcase the supremacy of the proposed HSAFS-OCAE model, a comparative study was conducted, and the results are shown in Table 4 [11].

Table 4: Comparative analysis results of the proposed HSAFS-OCAE approach and other existing algorithms

Methods	Accuracy	Precision	Recall	F-score
HSAFS-OCAE	99.25	91.71	93.44	92.49
Cu-DNNGRU + Cu-BLSTM	99.11	90.27	91.99	90.11
CNN	97.75	86.53	87.28	87.28
GRU-RNN	95.91	87.42	87.55	86.97
LSTM-CNN	94.33	87.52	87.85	87.44
2L-ZED-IDS	96.94	86.75	87.48	87.14
ANN	95.29	87.73	87.34	86.90
Bi-LSTM	94.50	86.20	86.81	86.61

Fig. 12 demonstrates the comparison study results of the proposed HSAFS-OCAE model and other existing models in terms of $accu_y$. The figure infers that the LSTM-CNN model and the Bi-LSTM model reported low $accu_y$ values such as 94.33% and 94.50%, respectively. In contrast, the CNN, Gated Recurrent Unit (GRU)-Recurrent Neural Network (RNN), 2L-ZED-IDS and the ANN models certainly produced increased $accu_y$ values such as 97.75%, 95.91%, 96.94% and 95.29% respectively. On the other hand, the Cu-DNNGRU + Cu-BLSTM model obtained a near-optimal $accu_y$ of 99.11%. But, the proposed HSAFS-OCAE model achieved the maximum classification performance with an $accu_y$ of 99.25%.

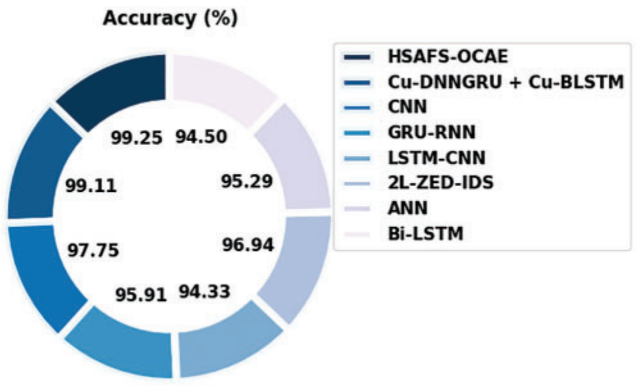

Figure 12: $Accu_y$ analysis results of the HSAFS-OCAE approach and other existing methodologies

Fig. 13 illustrates the comparative analysis results accomplished by the proposed HSAFS-OCAE model and other existing models in terms of $prec_n$. The figure infers that the LSTM-CNN model and the Bi-LSTM model reported the least $prec_n$ values such as 87.52% and 86.20%, respectively, where as the CNN, GRU-RNN, 2L-ZED-IDS and the ANN models certainly produced enhanced $prec_n$ values such as 86.53%, 87.42%, 86.75% and 87.73% correspondingly. Meanwhile, a combination of Cu-DNNGRU + Cu-BLSTM models achieved a near-optimal $prec_n$ of 90.27%. But, the proposed HSAFS-OCAE model achieved the maximal classification performance with a $prec_n$ of 91.71%.

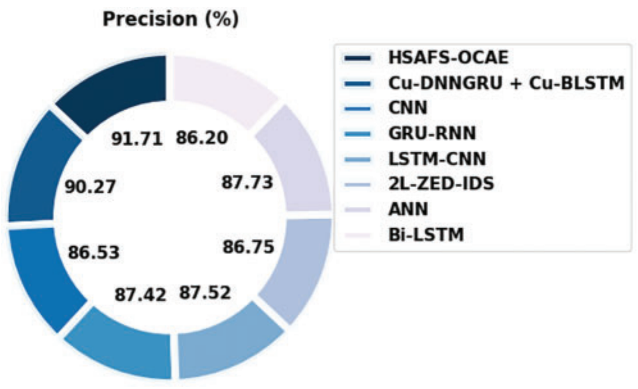

Figure 13: $Prec_n$ analysis results of the HSAFS-OCAE approach and other existing methodologies

Fig. 14 portrays the comparative analysis results achieved by the proposed HSAFS-OCAE method and other existing models in terms of $reca_l$. The figure implies that the LSTM-CNN model and the Bi-LSTM model produced the least $reca_l$ values such as 87.85% and 86.81%, respectively, whereas the CNN, GRU-RNN, 2L-ZED-IDS and the ANN models certainly achieved high $reca_l$ values such as 87.28%, 87.55%, 87.48% and 87.34% respectively. Further, a combination of Cu-DNNGRU + Cu-BLSTM models achieved a near-optimal $reca_l$ of 91.99%. But, the proposed HSAFS-OCAE method accomplished the maximal classification performance with a $reca_l$ of 93.44%. Based on these results and the discussion, it can be inferred that the proposed HSAFS-OCAE model achieved the maximum intrusion detection outcomes in the IoT-enabled SDN environment.

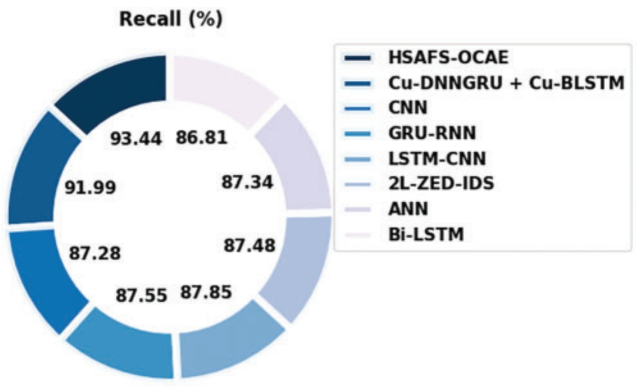

Figure 14: $Reca_l$ analysis results of the HSAFS-OCAE approach and other existing methodologies

4 Conclusion

In this study, a new HSAFS-OCAE model has been devised to recognize intrusions in the SDN-enabled IoT environment proficiently. The presented HSAFS-OCAE model follows a three-stage process in which the HSAFS technique is exploited for feature selection. Next, the CAE methodology is leveraged to recognise and classify intrusions in the SDN-enabled IoT environment. Finally, the AFSA-based hyperparameter tuning process is performed to boost the intrusion detection performance of the CAE approach. The proposed HSAFS-OCAE methodology was experimentally validated under several aspects. The comparison study outcomes established the improved outcomes of the HSAFS-OCAE model over other techniques. In the future, the HSAFS-OCAE model's performance can be improved using hybrid metaheuristic approaches.

Funding Statement: The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work through Small Groups Project under Grant Number (168/43). Princess Nourah bint Abdulrahman University Researchers Supporting Project Number (PNURSP2022R237), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. The authors would like to thank the Deanship of Scientific Research at Umm Al-Qura University for supporting this work by Grant Code: (22UQU4320484DSR01).

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

References

[1] N. Sultana, N. Chilamkurti, W. Peng and R. Alhadad, "Survey on SDN based network intrusion detection system using machine learning approaches," *Peer-to-Peer Networking and Applications*, vol. 12, no. 2, pp. 493–501, 2019.

[2] J. C. S. Sicato, S. K. Singh, S. Rathore and J. H. Park, "A comprehensive analyses of intrusion detection system for IoT environment," *Journal of Information Processing Systems*, vol. 16, no. 4, pp. 975–990, 2020.

[3] S. K. Dey and M. Rahman, "Effects of machine learning approach in flow-based anomaly detection on software-defined networking," *Symmetry*, vol. 12, no. 1, pp. 7, 2019.

[4] A. A. Albraikan, S. B. H. Hassine, S. M. Fati, F. N. Al-Wesabi, A. Mustafa Hilal *et al.,* "Optimal deep learning-based cyberattack detection and classification technique on social networks," *Computers, Materials & Continua*, vol. 72, no. 1, pp. 907–923, 2022.

[5] A. Wani, S. Revathi and R. Khaliq, "SDN-based intrusion detection system for IoT using deep learning classifier (IDSIoT-SDL)," *CAAI Transactions on Intelligence Technology*, vol. 6, no. 3, pp. 281–290, 2021.

[6] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, M. Ghogho *et al.,* "DeepIDS: Deep learning approach for intrusion detection in software defined networking," *Electronics*, vol. 9, no. 9, pp. 1533, 2020.

[7] A. O. Alzahrani and M. J. F. Alenazi, "Designing a network intrusion detection system based on machine learning for software defined networks," *Future Internet*, vol. 13, no. 5, pp. 111, 2021.

[8] M. A. Alohali, F. N. Al-Wesabi, A. M. Hilal, S. Goel, D. Gupta *et al.,* "Artificial intelligence enabled intrusion detection systems for cognitive cyber-physical systems in industry 4.0 environment," *Cognitive Neurodynamics*, 2022, https://doi.org/10.1007/s11571-022-09780-8.

[9] P. Manso, J. Moura and C. Serrão, "SDN-based intrusion detection system for early detection and mitigation of ddos attacks," *Information*, vol. 10, no. 3, pp. 106, 2019.

[10] M. A. Ferrag, L. Shu, H. Djallel and K. K. R. Choo, "Deep learning-based intrusion detection for distributed denial of service attack in agriculture 4.0," *Electronics*, vol. 10, no. 11, pp. 1257, 2021.

[11] A. S. Reddy, B. R. Reddy and A. S. Babu, "An improved intrusion detection system for sdn using multi-stage optimized deep forest classifier," *International Journal of Computer Science & Network Security*, vol. 22, no. 4, pp. 374–386, 2022.

[12] P. R. Grammatikis, P. Sarigiannidis, G. Efstathopoulos and E. Panaousis, "ARIES: A novel multivariate intrusion detection system for smart grid," *Sensors*, vol. 20, no. 18, pp. 5305, 2020.

[13] D. Javeed, T. Gao, M. T. Khan and I. Ahmad, "A hybrid deep learning-driven sdn enabled mechanism for secure communication in internet of things (IoT)," *Sensors*, vol. 21, no. 14, pp. 4884, 2021.

[14] J. Shu, L. Zhou, W. Zhang, X. Du and M. Guizani, "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.

[15] R. Shrestha, A. Omidkar, S. A. Roudi, R. Abbas and S. Kim, "Machine-learning-enabled intrusion detection system for cellular connected uav networks," *Electronics*, vol. 10, no. 13, pp. 1549, 2021.

[16] M. Aslam, D. Ye, A. Tariq, M. Asad, M. Hanif *et al.,* "Adaptive machine learning based distributed denial-of-services attacks detection and mitigation system for sdn-enabled IoT," *Sensors*, vol. 22, no. 7, pp. 2697, 2022.

[17] A. Derhab, M. Guerroumi, A. Gumaei, L. Maglaras, M. A. Ferrag *et al.,* "Blockchain and random subspace learning-based ids for sdn-enabled industrial IoT security," *Sensors*, vol. 19, no. 14, pp. 3119, 2019.

[18] L. Abualigah, A. Diabat and Z. W. Geem, "A comprehensive survey of the harmony search algorithm in clustering applications," *Applied Sciences*, vol. 10, no. 11, pp. 3827, 2020.

[19] M. S. Seyfioglu, A. M. Ozbayoglu and S. Z. Gurbuz, "Deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 4, pp. 1709–1723, 2018.

[20] A. M. Kabir, M. Kamal, F. Ahmad, Z. Ullah, F. R. Albogamy *et al.,* "Optimized economic load dispatch with multiple fuels and valve-point effects using hybrid genetic–artificial fish swarm algorithm," *Sustainability*, vol. 13, no. 19, pp. 10609, 2021.