Received: July 20, 2023. Revised: October 19, 2023.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Lightweight Hybrid CAE-ELM and Enhanced Smote Based Intrusion Detection for Networks

D. Shankar[1]* **G.Victo Sudha George[2]** **N. Kanya[1]** **S. Saraswathi[3]**

[1]Department of Information Technology,
Dr. M.G.R. Educational and Research Institute, Chennai-95, Tamil Nadu, India
[2]Department of Computer Science and Engineering,
Dr. M.G.R Educational and Research Institute, Chennai-95, Tamil Nadu, India
[3]Department of Computer Science and Engineering,
Sri Sivasubramaniya Nadar College of Engineering, Chennai-603110, Tamil Nadu, India
* Corresponding author's Email: shankarmtechit@gmail.com

Abstract: The intrusion detection system (IDS) plays an imperative role in defending the network from attacks. But, the IDS data is imbalanced, making the process complex for detecting the attacks accurately. According to these problems, this study proposes a network intrusion detection system based on an enhanced synthetic minority oversampling technique (SMOTE) and lightweight hybrid CAE (convolutional auto encoder)-ELM (extreme learning machine) model. Initially, the normalization of the original data is performed to avoid the influence of the maximum and minimum values. Secondly, an enhanced SMOTE is employed for solving the issues of the less detection rate of minority-attacks because of less training data. Here, the war strategy optimization (WSO) is incorporated with the SMOTE to design a novel WSO-based SMOTE technique for balancing the dataset. The three major steps utilized in WSO-based SMOTE are WSO based clustering, filtering and over-sampling. Then, the Information gain and fisher score based features extraction is employed for the reduction of computational complexity prior to the intrusion detection. Finally, the lightweight hybrid CAE-ELM model is executed for attack detection. CAEs are one of the most commonly used learning models because of their ability to construct a higher-level feature representation from the input data. Another model used in intrusion detection is the ELM, which provides an acceptable discrimination performance as well as a fast speed of learning. The performance of the proposed NIDS model is tested on the two benchmark datasets and achieved better accuracies of 99.21% and 99.15% on the UNSW-NB15 and CSE-CIC-IDS2018 datasets.

Keywords: Network intrusion detection systems, War strategy optimization, Enhanced SMOTE, Clustering, Convolutional auto encoder.

1. Introduction

The advancement of information technology and the materialization of cyberspace contribute to the social and economic prosperity of the technology world. However, the major concern is about security risks to the networks. Thus, the network's security is essential and is employed through detection. The network defense is assured by analyzing the incoming data traffic of the network based on the signs and behavior [1]. The activities of the networks, like dynamic tendency, convolution, heterogeneity, and vagueness, are analyzed to identify the malicious activity of the network continuously [2, 3]. Here, the detection of malicious activity or marking of the network intrusion is termed Intrusion detection and utilizes two different criteria in its detection process. In the first category, the detection of the misuse node is employed. Here, the characteristics of the incoming data traffic are compared with the attacker's history stored in the network and used to detect network intrusion.

In the second category, the detection of network intrusion is employed by making a connection

DOI: 10.22266/ijies2023.1231.83

between the current data traffic and the normal characteristics of the node. If there is any change overcomes in the behaviors of the data compared to the normal node, then the particular traffic is considered an intrusion [4-7]. The detection of the intrusion based on the misuse detection stores the known attacks; still, the learning capability is lower. Hence, the constant updating of the database is employed by the misuse detection technique with a minimal false alarm rate. When the network encounters a new attack model, the methods provide poor detection capability because of the matching criteria with the old attacker models. In contrast, the detection of an intrusion based on the anomaly criteria effectively detects the unknown intruder, but the elevated false alarm rate limits the performance [8].

While considering the intrusion into the network, the detection of the network traffic is initially evaluated for the prevention of abnormal traffic. Thus, the detection of the intrusion in the network is employed, and then the type of attack is stored in the host for the enhancement of the network's own defenses. The detection of malicious traffic in the network is considered a binary and multi-classification issue. The AI (artificial intelligence) based models detect the intrusion in the network through various criteria like network based model, host based model, anomaly based model and signature based model [9, 10, 28]. Among these, signature-based network intrusion detection is widely utilized by creating a set of rules for identifying the network pattern. The administrator's actions are tracked for raising the alarm of the model while detecting the intrusion into the computer system. The role of attack detection is essential for networks because an intruder in the networks disables the entire network and damages the resources of the network [11, 12]. The detection of intrusion in the network using the data mining technique provided positive results based on the attribute extraction and the data selection criteria, which limited the performance due to insignificant attribute consideration. Therefore, there is a need for a data processing approach with better accuracy in detecting network intruders [13].

Several researchers devised the detection of intrusion in the network based on machine learning and deep learning. The deep learning (DL) methods can learn the data and make the generalization more effective while performing network intrusion detection using the raw data [14-17]. The most utilized unsupervised model for detecting intrusions in the network is the convolutional auto encoder (CAE), which offers promising performance through the learning criteria. The self-learning capability, along with the adaptive concept of the CAE, enhances the performance of intrusion detection in the network [18-20]. In addition, extreme learning machine (ELM) is utilized in several application domains due to their good generalization with fast convergence and higher speed of data learning. Besides, the implementation of the ELM is simpler as it utilizes only one hidden layer. The random assignments of the ELM's input weights make learning very fast [21, 22].

Motivation and problem statement

Network intrusion detection is useful for protecting information from various attacks and preventing them with effective control methods. The application domains of the intrusion detection mechanisms are signature matching, anomaly detection, threat reporting, network traffic processing, threat classification, prevention systems and several other domains. There are several methods for the detection of intrusion in the network; still, accurate detection is a tedious task. Some of the challenges that the previous intrusion detection mechanisms faced are: The unbalanced dataset used to evaluate the performance of the developed model affects the model's performance by providing lower detection accuracy. The utilization of balanced data enhances the performance by elevating the count of minority attacks. The design of intrusion detection mechanisms with minimal training time and computation overhead is still challenging and limits performance. The introduction of lightweight mechanisms overcomes the challenges through the limited computation overhead. Choosing the appropriate attributes for detecting a network intrusion determines the accuracy of the detection. The failure to select the significant attributes limits the detection accuracy. The design of methods with offline learning using all gathered samples through the batch learning process is a time consuming process that makes the computation time overhead. The objectives of the proposed intrusion detection based on the lightweight hybrid CAE-ELM model are:

- To balance the dataset using the proposed WSO-SMOTE for solving the class imbalance problem to improve detection accuracy.
- To design the lightweight CAE-ELM for enhancing intrusion detection accuracy with minimal computation complexity.
- To extract the most relevant features from the incoming traffic for the reduction of

computation overhead.

- To evaluate the performance assessment based on the assessment measures like accuracy, precision, recall, f1-score and area under the curve (AUC).
- To show the performance enhancement by comparing the proposed method with other baseline techniques.

The rest part of the work is arranged as follows: Section 2 presents the recent literary works based on intrusion detection; section 3 presents the proposed intrusion detection model with a detailed description; section 4 presents the results analysis of the proposed and conventional models; section 5 depicts the conclusion of the overall work.

2. Literature review

Prior methods related to IDS based on DL models are detailed here. The optimized extreme learning based intrusion detection was suggested by Balasundaram [23], in which both the time-independent and dependent features were extracted from the incoming traffic prior to the classification. The inefficiency in handling the predispositions has been adjusted using the optimization strategy for acquiring increased accuracy. The optimization strategy used to tune the bias and weights of the ELM provided accurate detection by improving the global minima. The faster response was achieved in attack detection due to minimal computational effort and a simple architecture model.

An optimized ELM was designed by Tang, Y. and Li, C [24] to detect intrusions in the network through sequential learning. In this, the issue concerning the initialization of the deviations and weights of the online regularized ELM was optimally solved using the optimization approach. The sequential learning of the network by replacing batch learning enhances the effectiveness of the model through its better generalization capability. Besides, the designed model applied to the unbalanced dataset and showed elevated detection accuracy for the minority samples. The method was applicable to the real-time scenario with better classification accuracy.

An improved kernel-based ELM for detecting intrusions in the network was devised by Wang Z et al. [13], in which the back propagation was replaced using the enhanced grey wolf optimization for initializing the kernel's parameters. The hunting strategy employed by the gray wolf is improved by incorporating the exterior and interior hunting criteria in terms of feature space to increase optimization ability and search ability. The capability of high-dimensional detection was accomplished through the quick selection of the optimal parameters using the introduced optimization. The feasible solution for network intrusion detection was acquired through the designed framework in terms of assessment measures.

An optimized deep learning was developed by Sekhar, R et al. [25] for network intrusion detection, which used fuzzy-based imputation of missing data to solve the problem of inaccuracy in the dataset. Besides, the auto-encoder based attribute extraction was employed prior to the attack detection to minimize the computational complexity, where the optimal tuning of the neurons was devised while extracting the attributes. The evaluation of the presented method shows high accuracy in detecting the intrusion, and the analysis with different data sets shows the validity of the model.

Deep learning with the auto-encoder for the detection of intrusion in the network was developed by Andresini, G et al. [26], in which flow-based attribute extraction was employed on the incoming traffic. The combination of the triplet network and the auto-encoders was employed to detect intrusions in the network. The embedding-based learning criteria were used to predict the attack on the network. Here, the issue concerning the convergence during the learning of the triplet was solved through the inclusion of the auto-encoders in the detection model.

The time taken for the network training and testing was analyzed for the evaluation of the computation time of the model. It acquired a minimal time in milliseconds for testing and minutes for training. In addition, the ablation study of the model illustrates the capability of the introduced model. Table 1 presents the comparative analysis of the existing works.

Problem statement: Thus, from the review of existing methods, issues like vanishing gradient issues, non-capability of identifying some attacks, higher computation complexity, biased outcome due to the imbalanced dataset and inaccurate detection limit the model's performance. In the proposed attack detection method, WSO-SMOTE is proposed for balancing the dataset to avoid a biased outcome. Besides, the attack detection using the proposed hybrid CAE-ELM model provides computational efficient detection because the ELM is a computational efficient algorithm. Also, the consideration of CAE for the mapping of spatial features assists in enhancing the classification accuracy. The consideration of fewer layers of the model solves the vanishing gradient issues that happen in conventional deep learning models.

Table 1. Literature review

Author	Techniques	Benefits	Challenges
Balasundaram [23]	Optimized ELM	Accomplished higher exactness even with the increase in the number of malicious attacks.	The slower convergence of the optimization technique, along with the local optimal convergence limits the performance.
Tang, Y. and Li, C [24]	Optimized ELM	The computation complexity of the method was minimal, with a shorter training time.	The method failed to recognize some attacks in the dataset that limited the performance of the model.
Wang, Z et al. [13]	Improved kernel based ELM	The designed framework for detecting network intrusion was not sensitive to the dataset utilized that depicts the stability of the model.	A failure is considered for the significant attribute selection that leads to high dimensional data, which makes the computation overhead of the model.
Sekhar, R et al. [25]	Optimized deep learning	The quantitative analysis of the method with the standard detection algorithms offered better performance based on evaluation measures due to the robust attribute extraction criteria.	The accuracy and the error estimated by the designed method were not applicable for detecting the intrusion in the real time networks due to the vanishing gradient issue of the attack detector.
Andresini, G et al. [26]	Auto-encoder based deep learning	The triplet based learning, along with the auto-encoders, helps solve the issue concerning the imbalanced data and makes the detection more effective. Besides, the triplet convergence issues were solved using the auto-encoders during the random sampling strategy during the construction of the triplet.	The learning of the designed model was employed based on batch learning, in which the fitting issues exist due to the failure to consider the drift detection criteria while considering the streaming applications.

3. Proposed methodology

The NIDS plays a major role in protecting the network from intrusion. But, the conventional NIDS data is not balanced and is complex for correctly detecting minority attacks. This makes the training process complex, and the recognition time of the DL model is long. To overcome these challenges, this work presents the data balancing using WSO based SMOTE for NIDS. After the process of normalization, the data is divided into 70% (training) and 30% (testing). Then, the features are reduced using Information gain (IG) and fisher score (FS). Finally, the reduced features are classified by CAE with ELM, and this classifier improved the detection accuracy. The entire workflow of the proposed NIDS model is presented in Fig. 1, and the notation list is presented in Table 2.

3.1 Pre-processing

The dataset must be restructured before processing because the classification process takes more time. Here, the data is normalized in the range of [0, 1] using the min-max technique for transforming the input into a linear form. This preserves the relationship of the original data for enhancing the effectiveness of the intrusion detection technique. The min-max normalization is expressed as:

$$v_a = \frac{u_a - min(u)}{max(u) - min(u)} \tag{1}$$

3.2 Data balancing using WSO based SMOTE

In NIDS, numerous models are introduced for the classification and prediction of data. The DL models perform better than ML models when the process requires sufficient data. However, the performance is degraded when the data is not enough for the process. The synthetic minority oversampling technique (SMOTE) is used to tackle the data imbalance; however, the class generates noise since it doesn't

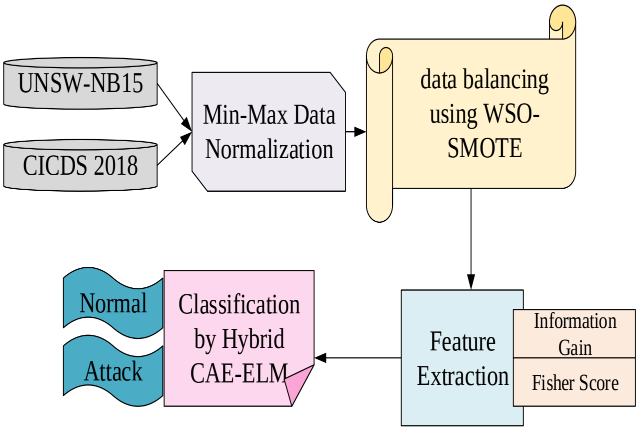

UNSW-NB15
CICDS 2018
Min-Max Data Normalization
data balancing using WSO-SMOTE
Feature Extraction
Information Gain
Fisher Score
Classification by Hybrid CAE-ELM
Normal
Attack

Figure. 1 Proposed NIDS model

Table 2. Notation list

Notation	Detail
ED	Euclidean distance
C_i	center of cluster point
Y_k	center of data point
m	Total samples
N	Number of clusters
$Z_i(t)$	Previous position
$Z_i(t+1)$	New position
L	Position of leader
K	Position of king
S_i	score
M_{iter}	Maximum iteration
β	exponential term
c	head of the army
$Z_w(t)$	weak soldiers
$med(Z)$	median of the overall army
C	penalty matrix
B	output matrix
Z_{dj} and Z_{dj+1}	input and output of the j^{th}layer of the decoding network
Z_{ej} and Z_{ej+1}	input and output of the j^{th}layer of the encoding network
W_{ej}^T	weight
b_{ej}	bias
g_{ej}	Activation function

consider the position of data near the rare data. Hence, in this work, WSO based SMOTE is presented for balancing the data. These data balancing stages are clustering, filtering and over-sampling.

In clustering, the input data is divided into K-groups using KMC (K-means clustering). The filtering process chooses the clusters for the over-sampling and retains the large part of the minority class samples. Then, it allocates the synthetic samples for generating and providing more samples for clustering. At last, in the over-sampling process, SMOTE is provided in every chosen cluster for achieving the minority and majority instances.

KMC is one of the iterative models to find natural circumstances set in data, and it is indicated in Euclidean distance. The KMC algorithm has the stages like (a) number of clusters, (b) initialization of cluster and (c) Euclidean distance. The stages involved in KMC are:

Stage 1: Choose a random K initial center of clusters $(y_1, y_2, \ldots, y_k)$ from m samples $(z_1, z_2, \ldots, z_m)$

Stage 2: Distribute the sample z_i to the cluster C_i.

Stage 3: Find Kclusters from the data on the basis of the objective function L, and it is given as:

$$L = \sum_{i=1}^{m} \sum_{k=1}^{N} ED^2 \,(C_i - Y_k) \tag{2}$$

Where $ED$is the Euclidean distance, C_i and $Y_k$are the center of cluster and data points. N is the number of clusters.

According to the distance attained, the data points are allocated to the cluster with a minimal distance from the centroid. Then, the data points are clustered, and the average of every point belongs to the clusters. The average value is defined as the new centroid of the cluster for the next iteration. This procedure is continued until the previous iteration and centroid attained are equal.

3.2.1. WSO

To improve the clustering quality of KMC, this optimization is utilized. The number of clusters to be created is found in the first stage. After that, each data point is clustered based on the minimum achieved Euclidean distance. The next stage is to compute the optimal centroid of the cluster for every cluster. Each cluster is randomly initialized as a king and war population for the optimization process. After that, the value of fitness of every king and leader is computed using the objective function of KMC. When the sum of the mean of the intra-cluster distances must be reduced, the kind with a minimum fitness value is set as the best solution. The WSO procedure is performed for every cluster to obtain the best position of the cluster's centroid.

This WSO optimization [27] mimics the strategic behavior of a group of soldiers in war. Here, the war is considered an optimizer model in which all soldiers move statically to the optimal value. This optimization has two major mechanisms, the attacking mechanism and defense mechanism. Based on the strategy, the soldier's positions on the battlefield are estimated.

Attacking (exploitation) mechanism: There are two war mechanisms; in the initial mechanism, each soldier’s position is updated based on the position of the king and the leader. The king considers the powerful position for launching an enormous attack on the opponent. Due to that, the soldiers with high attacking force are considered the king. Every soldier has a similar weight and score during the war. When the soldier completes the strategy successfully, his score increases. Based on the success, the weight and score are updated, and it is given as follows:

$$Z_i(t+1) = Z_i(t) + 2 \times \rho \times (L - K) + r \times (U_i \times L - Z_i(t)) \quad (3)$$

Where $Z_i(t)$ and$Z_i(t+1)$ are the previous and new positions, $L$and $K$are the positions of leader and king. $r$is the random number,ρ and U_i are the density and weights.

Updation of score and weight: The position of all search agents is based on the interconnection of the position of the leader, king and the score of the soldiers. The weight of the soldiers is based on the score, and it is given as follows:

$$U_i = U_i \times \left(1 - \frac{S_i}{M_{iter}}\right)^{\beta} \quad (4)$$

Where S_i is the score, weight is represented as U_i, M_{iter} and β are the maximum iteration and exponential term.

Defense (Exploration) mechanism: This mechanism is on the basis of the positions of the random soldier, the head of the army and the king. This mechanism is given as:

$$Z_i(t+1) = Z_i(t) + 2 \times \rho \times (K - Z_r(t)) + r \times (U_i \times c - Z_i(t)) \quad (5)$$

Where c is the head of the army, and $Z_r(t)$ is the random value at the previous position.

Replacing the weak soldiers: The soldiers with the worst fitness value are identified for all iterations. The weak soldiers are replaced with random soldiers, and it is given as:

$$Z_w(t+1) = LL + r \times (UL - LL) \quad (6)$$

Relocation of the weak soldiers near the median of the overall army is given as:

$$Z_w(t+1) = -1(1-r) \times (Z_w(t) - med(Z)) + K \quad (7)$$

Where $Z_w(t)$ is the weak soldiers and $med(Z)$ is the median of the overall army. The pseudocode of the clustering based WSO is given in Algorithm 1.

Algorithm 1: Pseudocode of the clustering based WSO

```
Input: Number of clusters, size of soldier, positions of king and leader, population
Output: Optimal centroid of cluster
Choose the random points as center of clusters
for every data point
Find the Euclidean distance using Equation (2)
Identify the cluster with minimal Euclidean distance
end for
Compute the average of all data points in every cluster
Declare the average values as the new center of clusters
end while
for every cluster
Initialize rank, weights and initial population of king
In the war space, soldiers are allocated equally and randomly
for 1: Size of soldier
    For every soldier, get the attack force
end
Arrange the fitness of every soldier
Choose the soldier with high value is considered as king and the next fitness as the leader
                $while\ t < max\_iter$
for 1: Size of soldier
                $\rho = r$
    Update the exploration and exploitation stage
end
    Compute the attacking force of all soldiers.
     Arrange the fitness of every soldier
end
Find the weak soldier having the worst fitness
Replacing the weak soldiers using Equation (6).
Update the king and leader’s positions
                $t = t + 1$
Choose the king as the new cluster center
end
```

3.3 Feature extraction

This feature extraction process is essential for classification to eliminate redundant features and retain useful features. Further, this model is used for retrieving the sub-set of the original features without modifications. The next stage is to compress the dimensionality of the two datasets using IG and FS.

3.3.1. IG

The major aim of this IG is to rank the attributes by computing the IG entropy for all attributes. The attributes have a gain value of most relevant (one) to least relevant (zero). The attributes having high rank are set as the input subset of features. $IG(a,b)$ shows the minimization of uncertainty and is defined as:

$$IG(a,b) = H(a) - H(a|b) \tag{8}$$

Where $H(a)$ is the entropy of the data and shows the uncertainty in predicting the random parameter. $H(a|b)$ is the conditional entropy and shows the uncertainty on the basis of the known parameter b. $H(a)$ and $H(a|b)$ are represented as:

$$H(a) = -\sum p(a)\, log\, p\,(a) \tag{9}$$

$$H(a|b) = \sum p(b) H(a|b) \tag{10}$$

The order of each attribute is sorted using the IG value and the attribute with the high rank is considered the input feature.

3.3.2. FS

The features selected from IG are used for the classification process, but the IG will be baized to the features with a high range of feasible values. Therefore, the features return a nonzero entropy value and increase the value of gain than the remaining features. To address this challenge, the features selected from IG are further reduced using FS.

The FS model efficiently manages the irrelevant features, and it is used for the classification of attacks. It selects features separately on the basis of the fisher strategies and obtains the optimal set of features. Moreover, the FS model finds the feature subset. Hence, the lower limit of the FS is increased. The FS model is expressed as follows:

$$FS(K_s) = M\left\{(\vec{E_s})(\vec{E_r} + \chi A)^{-1}\right\} \tag{11}$$

Where $M$is the matrix, $\vec{E_s}$matrix of class scatter, $\vec{E_r}$is the overall scatter matrix, $\chi$is the regularization variable and A is the controlling variable.

3.4 Classification using CAE-ELM

This work utilizes the CAE-ELM classifier for the NIDS, which integrates the single classifiers CAE and ELM to produce a robust classification. Further, it classifies the attacks on the network. The major aim of this classifier is to enhance the accuracy, which is higher than the single classifiers.

3.4.1. CAE

Auto-encoders (AEs) are self-supervised classifier that utilizes neural networks for representation learning. The term representation learning is a model in which the system has encoding input data. AE is used for mapping the input data for compressing domain representation. CAE utilizes convolutional and pooling (down sampling) layers to encode the input data for compressing domain representation. Then, the convolutional and up sampling layers are utilized for the reconstruction of the original data.

CAE combines the advantages of CNN (convolutional neural network) and AE for attaining better representation of features, and they are more essential for intrusion detection. Moreover, CAE shares weight between the inputs and preserves the spatial features. Hence, the total number of parameters required for training is reduced; this minimizes the computational overhead and memory requirements. CAE has a convolutional layer for encoding and a deconvolutional layer for decoding. CAE has four major parts: Encoding network, Bottleneck layer, decoding network and reconstruction loss. The encoding network encodes the input data to the compressing domain. The final layer of the encoding network is the bottleneck layer, and the result obtained is encoded data. The mathematical computation for representing the process of every layer in the encoder is given as follows:

$$Z_{ej+1} = g_{ej}\left(W_{ej}^T Z_{ej} +\right) b_{ej} \forall_j = 0,1,2,\ldots.,N \tag{12}$$

Where $Z_{ej}$and $Z_{ej+1}$are the input and output of the j^{th}layer of the encoding network. W_{ej}^T, $b_{ej}$and $g_{ej}$are the weight, bias and activation functions of the encoding network.

The decoding network obtains the output from the bottleneck layer and is used for reconstructing the original data. The number of layers in encoding and decoding is the same but in reversible order. The final layer of the decoding network generates the input data reconstruction.

The mathematical computation for representing the process of every layer in the encoder is given as follows:

$$Z_{dj+1} = g_{dj}\left(W_{dj}^T Z_{dj} +\right) b_{dj} \forall_j = 0,1,2,\ldots.,N \tag{13}$$

Where $Z_{dj}$and $Z_{dj+1}$are the input and output of the j^{th}layer of the decoding network. W_{dj}^T, $b_{dj}$and $g_{dj}$are the weight, bias and activation functions of the decoding network. The difference between the Z^O (original data) and Z^R (reconstructed data) is called the reconstruction loss. The AE is trained by back propagation to minimize the reconstruction. MSE (mean squared error) and BCE (binary cross entropy) are two loss functions utilized for computing the reconstruction loss.

$$MSE(Z^o, Z^R) = \frac{1}{D}\sum_j^D (Z_j^o - Z_j^R)^2 \quad (14)$$

$$BCE(Z^o, Z^R) = -\frac{1}{D}\sum_j^D Z_j^o \log Z^R + (1 - Z_j^o)\log(1 - Z_j^R) \quad (15)$$

3.4.2. ELM

This classifier is stated as the least square based SLFN (single layered feed network) and is used for classification and regression issues. There are three layers in ELM: input layer (IL), hidden layer (HL) and output layer (OL). For the training set M, the hidden neurons $H$and the activation function $f(y)$ is defined as:

$$e_k = \sum_j^H \alpha_j f\left(w_j, c_j, x_k\right) k = 1,2,\ldots,N \quad (16)$$

Where w_j and α_j are the weight vectors of the input with HL and the output with HL. $x_k$is the input parameter, c_j is the hidden bias of the j^{th} hidden neuron and $e_k$is the output.

The expression for computing the output weight is given as:

$$\beta = B^+Y \quad (17)$$

Where $B$is the output matrix of HL, $B^+$is the Moore-Penrose generalization inverse of $B$and $Y$is the target value of ELM. The aim of ELM is to obtain the minimal training error of the output weights, and it is expressed as:

$$min \frac{1}{2}||\beta||^2 + \frac{C}{2}||H\beta - Y||^2 \quad (18)$$

Where C is the penalty matrix.

3.4.3. CAE-ELM

This hybrid network integrates CAE and ELM, as shown in Fig. 2. The CAE is designed, and the parameters like the number of filters and the size of convolutional kernels are defined. Similarly, the model ELM uses a penalty matrix and hidden nodes. The training process of this network undergoes three major stages:

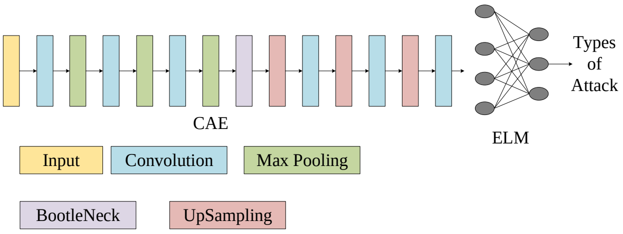


Figure. 2 Structure of CAE-ELM

Table 3. Hyper-parameters of the proposed CAE-ELM

Hyper-parameters	Encoder	Decoder
Convolutional layer	10	10
Up and down samplings	5	5
Shape of input	225,225,1	7,7,256
Shape of output	7,7,256	225, 225, 1
Activation function	ReLU	ReLU
Learning rate	0.01	
momentum	0.90	
Size of batch	128	
Optimizer	SGD	

(a) Generation of convolution feature maps where CAE-ELM randomly generates convolution kernels and arrives at a convolution feature map using random kernels and input data. A pooling operation is performed on the maps to preserve the overall invariance.

(b) The weights are randomly generated using AE, and the weights of ELM are calculated. In ELM, the features generated with AE are integrated into the vector.

(c) In the testing phase, the test patterns are deployed in the CAE-ELM network to get the attacks in the network.

4. Results analysis

For developing and evaluating the performance of the proposed NIDS, the system configuration Intel core I7, 3.2 GHz processor and 16 GB RAM is utilized. The Python programming language is utilized for the evaluation. Table 3 presents the hyper-parameters of the proposed CAE-ELM.

4.1 Dataset details

UNSW-NB15: The network packets of this dataset are produced using the IXIA PerfectStorm tool. The tcpdump device was evaluated for capturing 100 GB of unprocessed files. This data has 9 kinds of attacks, and this network is generated using the IXIA perfectstorm tool. There are 1, 75,341 records (training set) and 82,332 records (testing set).

CSE-CIC-IDS2018: This dataset was introduced in 2018 by a collaborative project among the CSE (communications security establishment) and CIC (Canadian Institute for Cybersecurity). The benign packets are produced using the real network events by the abstract characteristics of human users. The attacking strategies are developed by various machines on the target of the network. This dataset contains a total of 16,232,943 flows; among them, 83.07% (13, 474,708) begin flows and 16.93% (2,748,235) attack flows.

4.2 Performance measures

This work is performed with various evaluation measures for computing the performance and compared with the following measures accuracy, F1-measure, precision, recall, specificity, false negative rate (FNR), confusion matrix and ROC (region of characteristics) analysis.

Accuracy: It is one of the major measures for estimating the accurate classification. This metric estimates the entire performance of the classifier, and it is expressed as:

$$A = \frac{T_p + T_n}{T_p + T_n + F_p + F_n} \tag{19}$$

F-Measure: This measure is used for combining precision and recall, and it is expressed as:

$$F1 - measure = \frac{2T_p}{2T_p + F_p + F_n} \tag{20}$$

Precision: It is the capacity of the classifier for classifying the important data points in the data. It is the portion of the expected positives to the entire dataset, and it is given as follows:

$$P = \frac{T_p}{T_p + F_p} \tag{21}$$

Recall: It is the capacity of the classifier to discover the important data points in the data. It is the ratio of the expected positives to an entire positive sample, and it is given as:

$$R = \frac{T_p}{T_p + F_n} \tag{22}$$

Specificity: It is the ratio of true negative outcomes to the total number of negatives, and it is expressed as:

$$Sp = \frac{T_n}{T_n + F_p} \tag{23}$$

FNR: It is the ratio of false negative cases to the overall positive cases, and it is expressed as:

$$FNR = \frac{F_p}{T_n + F_p} \tag{24}$$

4.3 Comparative analysis of the CSE-CIC-IDS2018 dataset

The following analysis of results shows the comparative analysis of the various models on the CSE-CIC-IDS2018 dataset. The performance of the classification is an essential measure for NIDS. The performance of the classifiers is carried out with and without SMOTE NIDS.

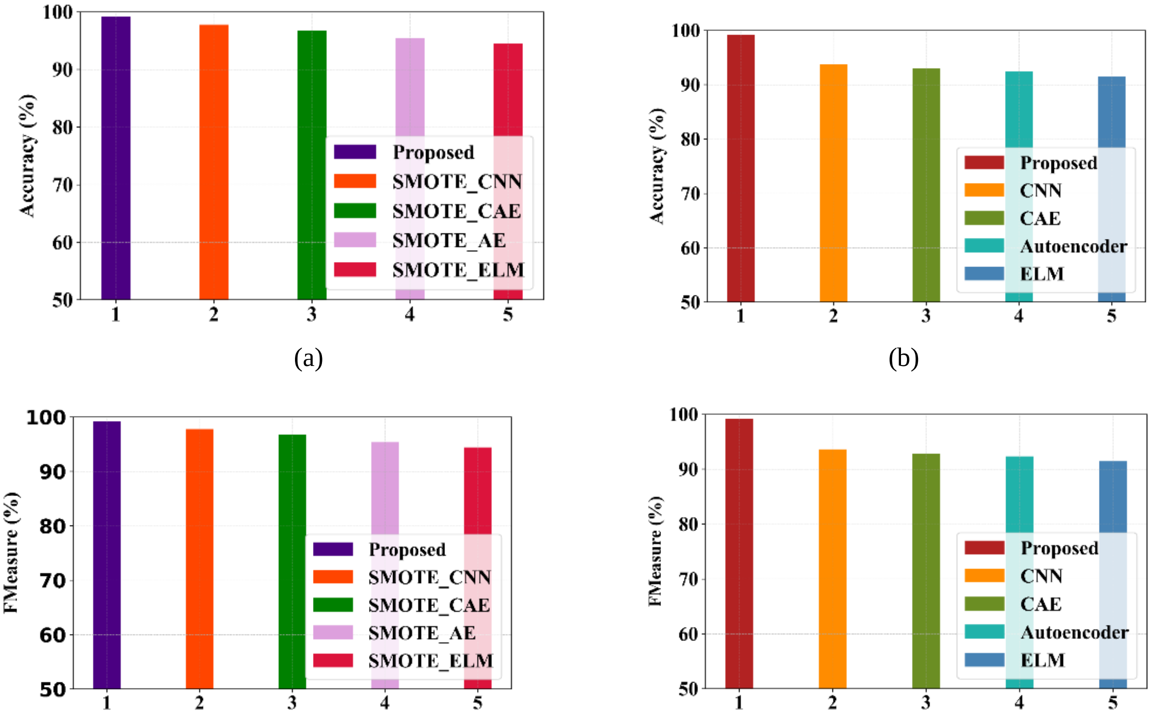
Fig. 3 compares the classification performance of accuracy with SMOTE, accuracy without SMOTE, F-measure using SMOTE and F-measure without SMOTE. The accuracy performance achieved by the proposed model is 99.21%, SMOTE-CNN, SMOTE-CAE, SMOTE-AE, and SMOTE-ELM are 97.78%, 96.79%, 95.4% and 94.4% on the CSE-CIC-IDS2018 dataset. Similarly, the F-measure performance achieved by the proposed model is 99.22%, SMOTE-CNN, SMOTE-CAE, SMOTE-AE, and SMOTE-ELM are 97.7%, 96.77%, 95.35% and 94.4%. Further, it is observed that when the SMOTE is not applied to the dataset, these models attain poor results.

Fig. 4 compares the classification performance of precision and recall using SMOTE and without SMOTE. The obtained precision and recall values of the proposed model are 99.14% and 99.29%, respectively. However, the precision and recall values are very low for the classifiers like SMOTE-CNN, SMOTE-CAE, SMOTE-AE and SMOTE-ELM. The proposed NIDS model achieved better outcomes due to the SMOTE based WSO. The clustering based weight selection model provided better results.

Fig. 5 defines the comparison of classification performance of specificity and FNR using SMOTE and without SMOTE. The value of the specificity must be high for the better classifier, and the value of the FNR must be low for the better classifier. The experimental demonstration shows that the proposed NIDS model achieved better specificity, and FNR are 99.14% and 0.007, respectively. Hence, the enhanced SMOTE with a hybrid classifier yielded better results.

Fig. 6 presents the ROC curves of with SMOTE and without SMOTE models. This curve is a useful measure to compare multi-classifiers. This curve provided the balancing between FPR (false negative rate) and TPR (true positive rate). The accuracy of the classifier using the ROC curve is computed based on the AUC (area under the curve). The AUC value achieved by the proposed model is 0.992 on the

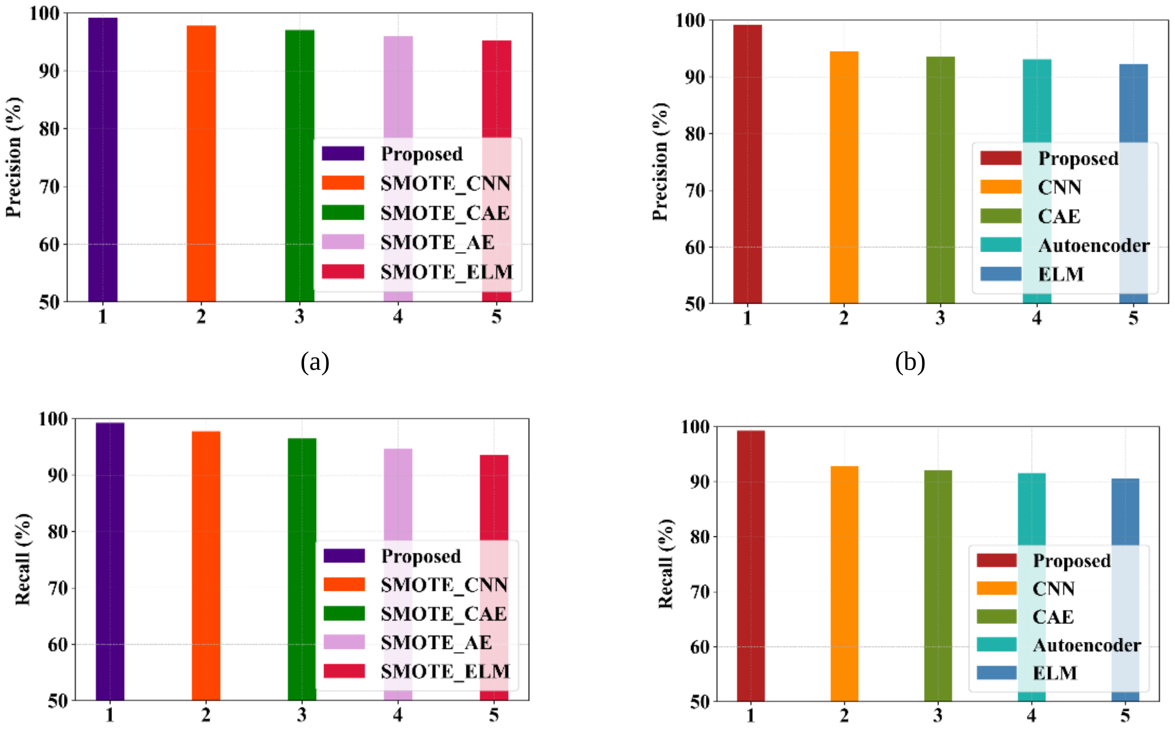
Received: July 20, 2023. Revised: October 19, 2023.



(a) (b)

(c) (d)

Figure. 3 Comparison of classification performance (a) accuracy using SMOTE, (b) accuracy without SMOTE, (c) F-measure using SMOTE, and (d) F-measure without SMOTE

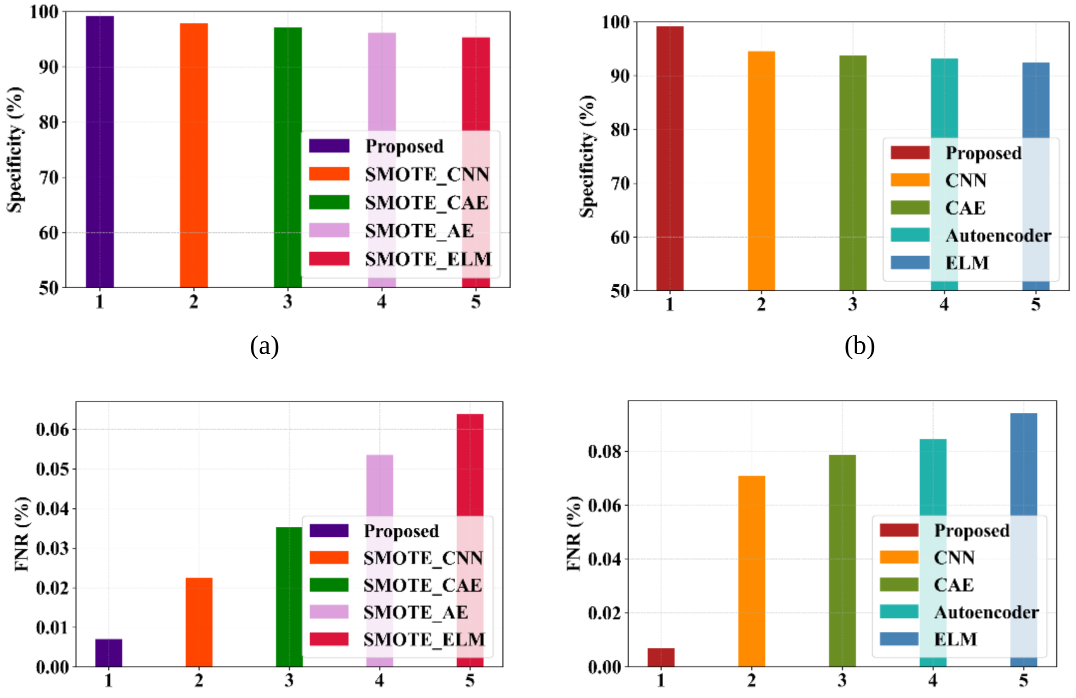


(a) (b)

(c) (d)

Figure. 4 Comparison of classification performance: (a) Precision using SMOTE, (b) Precision without SMOTE, (c) Recall using SMOTE, and (d) Recall without SMOTE

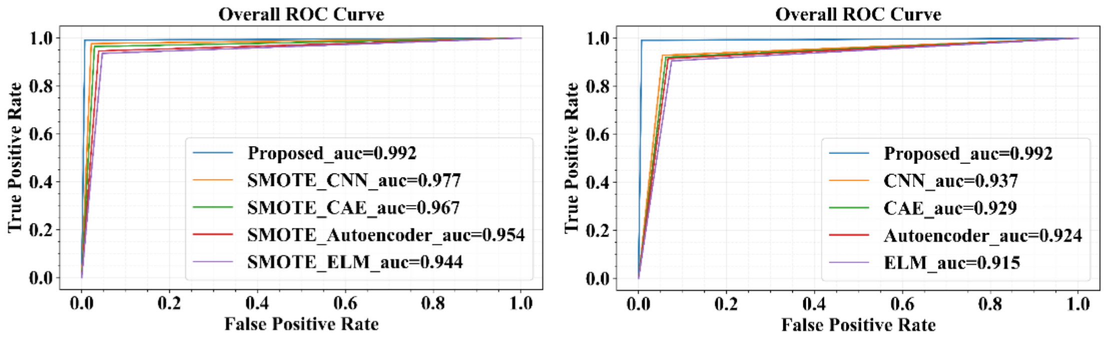
Received: July 20, 2023. Revised: October 19, 2023.



(a) (b)

(c) (d)

Figure. 5 Comparison of classification performance: (a) Specificity using SMOTE, (b) Specificity without SMOTE, (c) FNR using SMOTE, and (d) FNR without SMOTE



(a) (b)

Figure. 6 ROC curve: (a) with SMOTE and (b) without SMOTE

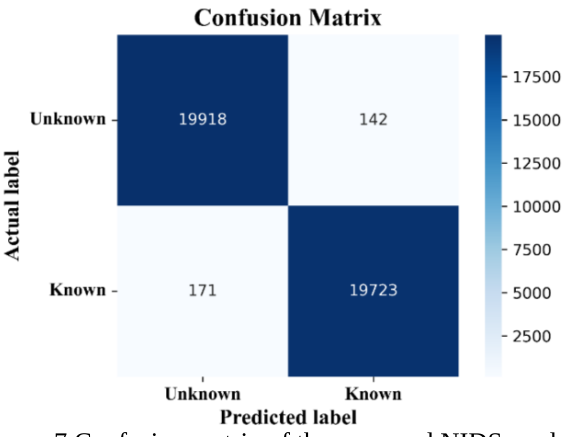


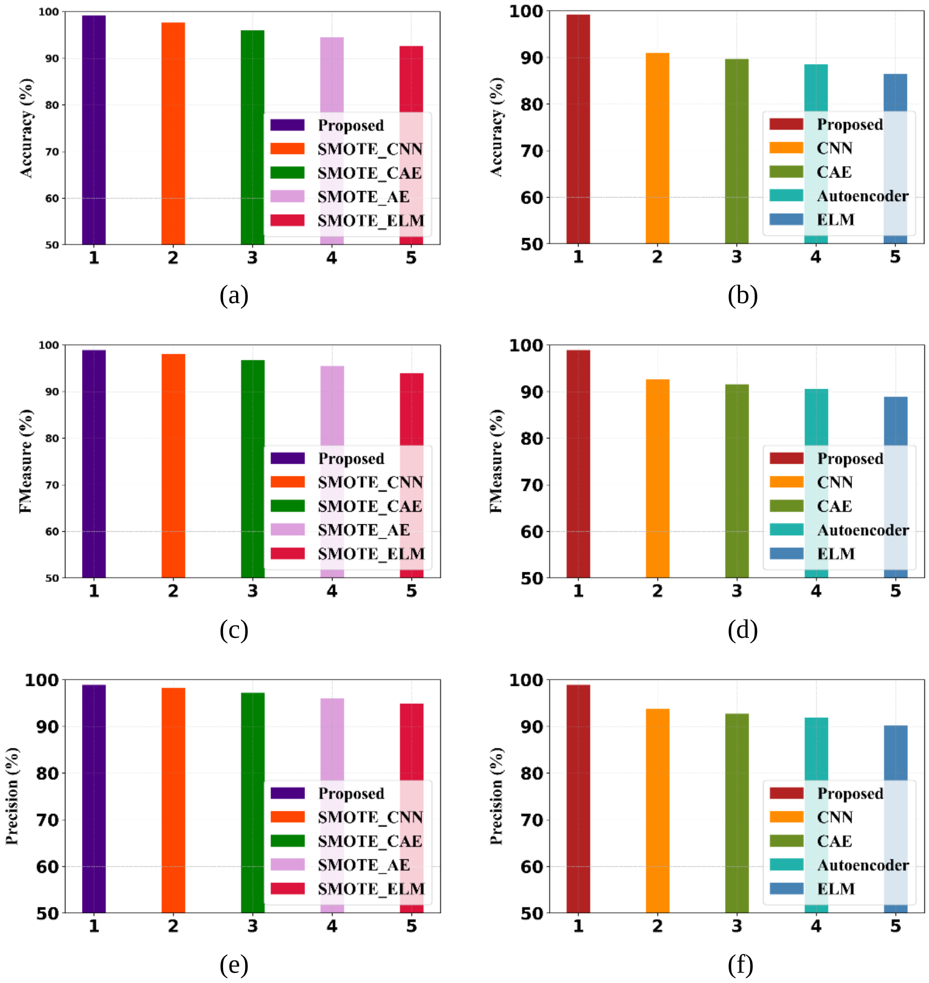
Figure. 7 Confusion matrix of the proposed NIDS model

CSE-CIC-IDS2018 dataset.

Fig. 7 represents the confusion matrix of the proposed NIDS model on the CSE-CIC-IDS2018 dataset. This matrix depicts the capacity of the proposed CAE-ELM classifier for detecting and classifying the attacks. Further, this matrix provides information about the classifiers' classifications of actual and prediction. In this matrix, a total of 19,918 samples are classified as unknown (attacks), and 142 samples are misclassified. Similarly, 19,723 known (normal) and 171 samples are misclassified.

4.4 Comparative analysis of the UNSW-NB15 dataset

The following results analysis shows the comparative analysis of the various models on the



(a) (b) (c) (d) (e) (f)

Figure. 8 Comparison of classification performance: (a) accuracy using SMOTE, (b) accuracy without SMOTE, (c) F-measure using SMOTE, (d) F-measure without SMOTE, (e) Precision using SMOTE, and (f) Precision without SMOTE

UNSW-NB15 dataset. The robustness of the proposed model is compared for with and without SMOTE based classifiers.

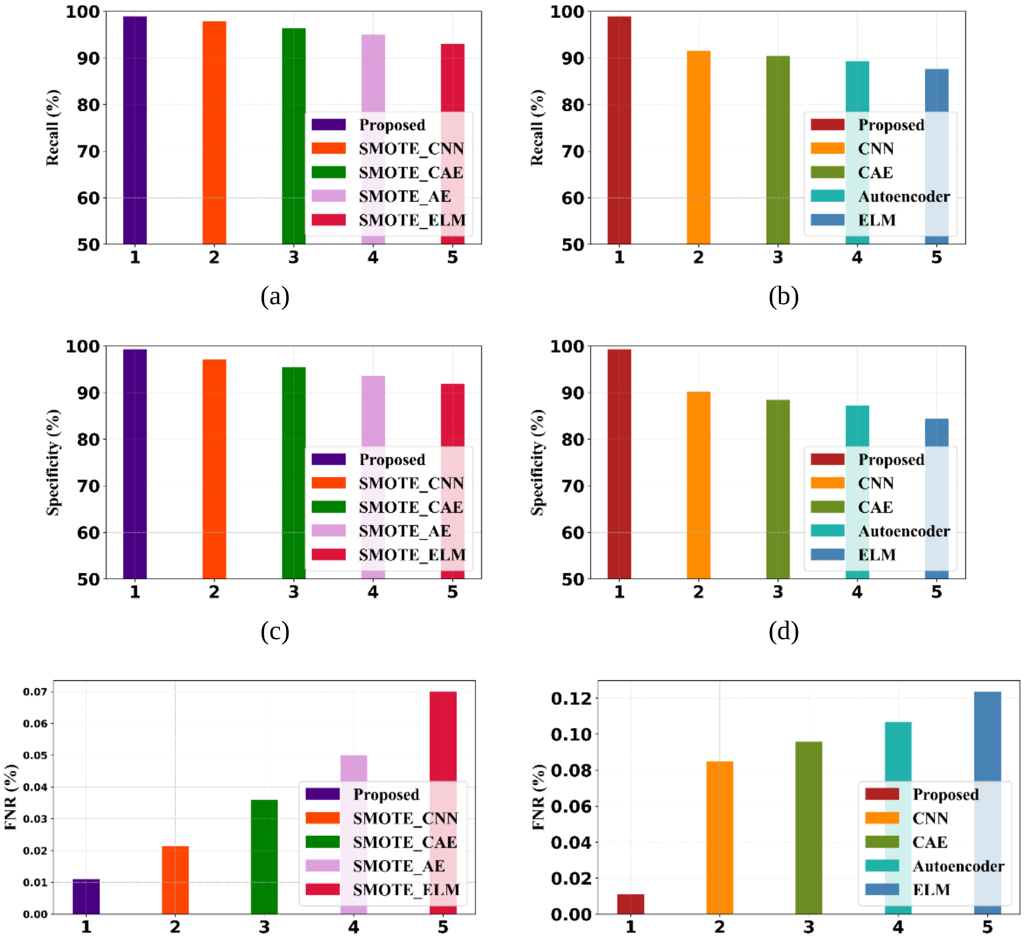
Fig. 8 defines the comparison of classification performance for accuracy, F-measure and precision using SMOTE and without SMOTE. The proposed SMOTE is used for solving the class imbalance problem. Fig. 8 presents the distribution of data and results in various classes before and after providing SMOTE. The accuracy, F-measure and precision of the proposed model are 99.15%, 98.8% and 98.86% on the UNSW-NB15 dataset.

Fig. 9 depicts the comparative analysis of the recall, specificity and FNR using SMOTE and without SMOTE. From the experimental observation, it is found that the proposed model attained better results in all the cases. The conventional models like CNN, CAE, AN, and ELM attained poor results because of the data imbalanced issues. The proposed model overcomes the data imbalanced issues by clustering using KMC and weight selection by WSO. Therefore, the proposed NIDS model attained a very less FNR value and high recall and specificity.

Fig. 10 depicts the division of T_p to F_p , and here, the graph is plotted for with and without SMOTE on the UNSW-NB15 dataset. This analysis provides the efficiency of predicted information from the classifier. In this curve, the AUC shows how the classifier is effective in the discrimination of classes and defines the measures of differentiating classes. The ROC value achieved by the proposed model is 0.991, SMOTE-CAE, SMOTE-ELM, and without SMOTE ELM achieved less AUC values of 0.959, 0.924 and 0.860, respectively.

Fig. 11 indicates the confusion matrix of the proposed NIDS model on the UNSW-NB15 dataset. Here, the confusion matrix is given for the binary classification. The graph is plotted between the predicted and actual labels. In this matrix, a total of 6,820 samples are classified as unknown (attacks), and 76 samples are misclassified. Similarly, 11,159 known (normal) and 78 samples are misclassified.

Fig. 12 shows the convergence analysis on CSE-CIC-IDS2018 and UNSW-NB15 datasets of various optimization techniques. This Figure is presented to

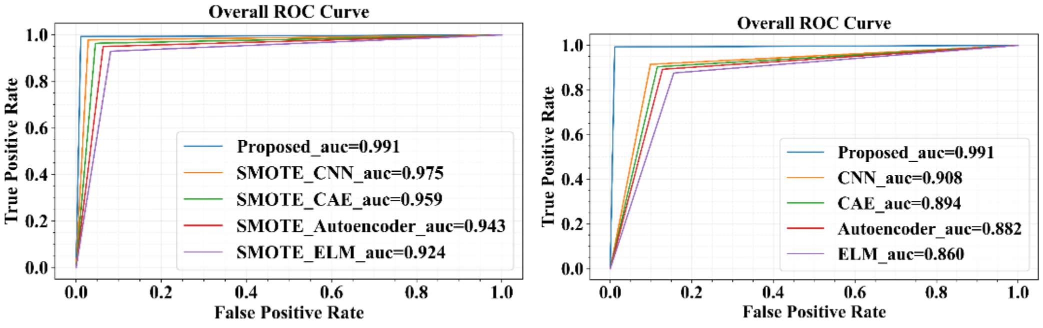


(a) (b)

(c) (d)

(e) (f)

Figure. 9 Comparison of classification performance: (a) Recall using SMOTE, (b) Recall without SMOTE, (c) Specificity using SMOTE, (d) Specificity without SMOTE, (e) FNR using SMOTE, and (f) FNR without SMOTE



(a) (b)

Figure. 10 ROC curve: (a) with SMOTE and (b) without SMOTE

analyze the efficiency of the proposed optimizer clustering based WSO. The optimization techniques like Aquila optimization (AO), Ebola optimization (EO), hunger games optimization (HGO) and Gorilla Troop's optimization (GTO) are compared with the proposed optimizer clustering based WSO. Here, the performance is carried out for 100 iterations and for all the iterations, the proposed optimization attained a better fitness value. Hence, it is proved that this optimization is not trapped by local optima and slow convergence. Table 4 presents the comparison of the proposed intrusion detection model with the OptiBiNet_GRU model. Here, the accuracy attained by OptiBiNet_GRU, Proposed (Without SMOTE) and Proposed (With SMOTE) is 99%, 89.67% and 99.15%. The proposed model outperformed the without SMOTE and OptiBiNet_GRU models.

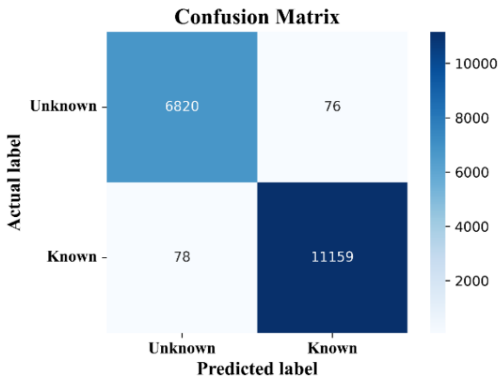
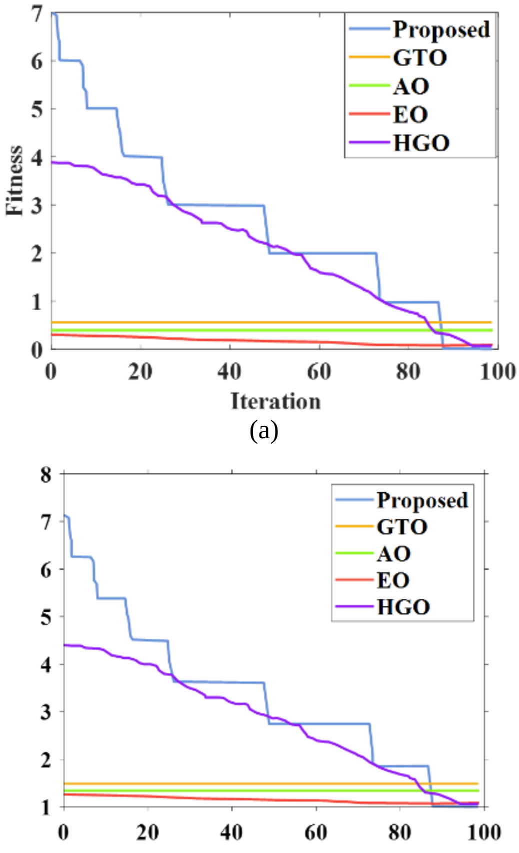


Figure. 11 Confusion matrix



(a)

(b)

Figure. 12 Convergence analysis on: (a) CSE-CIC-IDS2018 and (b) UNSW-NB15 dataset

Table 4. Comparative analysis of the UNSW-NB15 dataset

Methods	Accuracy	Precision	Recall	F-measure
OptiBiNet_GRU [29]	99	95	98	97
Proposed (Without SMOTE)	89.67	92.72	90.42	91.56
Proposed (With SMOTE)	99.15	98.86	98.89	98.88

Table 5. Comparative analysis

Metrics/ Methods	Optimized ELM	Improved Kernel based ELM	Optimized Deep Learning	Proposed
UNSW-NB15 dataset				
Accuracy	94.56	86.74	90.86	99.15
Recall	93.25	84.47	87.34	98.9
Precision	92.73	85.14	89.41	98.87
F-Measure	94.75	86.43	92.58	98.88
CSE-CIC-IDS2018				
Accuracy	95.89	86.72	91.72	99.21
Recall	93.75	85.32	88.12	99.29
Precision	92.96	85.36	90.25	99.14
F-Measure	95.21	87.29	93.58	99.22

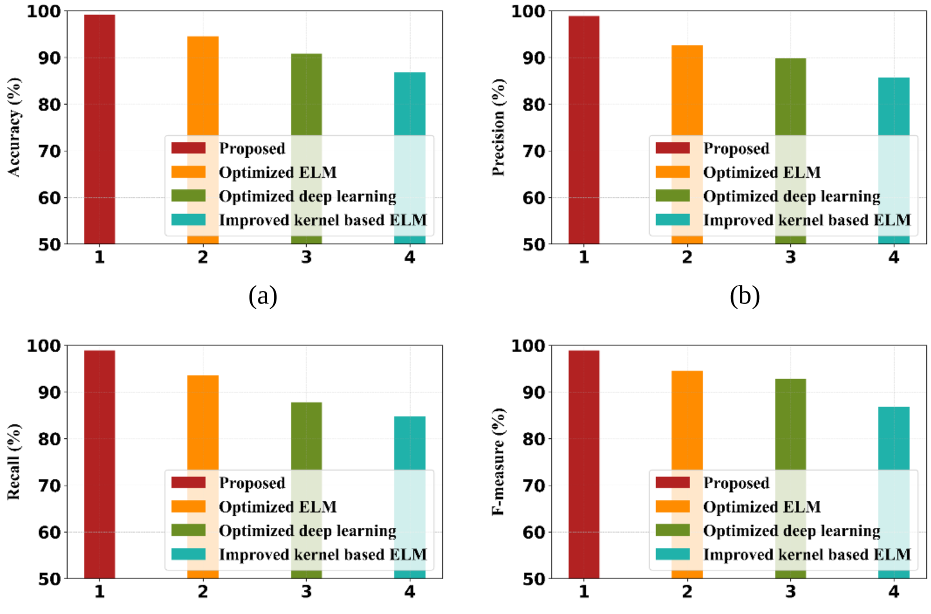
4.5 Comparative analysis with state of the art methods

The comparative analysis of the proposed method with the conventional intrusion detection methods like Optimized ELM [24], Improved Kernel based ELM [13] and Optimized Deep Learning [25] using the UNSW-NB15 is depicted in Figure 13. The detailed analysis using the UNSW-NB15 and CSE-CIC-IDS2018 is depicted in Table 5.

Here, the analysis depicts that the proposed method accomplished a superior outcome compared to the existing methods while analyzing using the CSE-CIC-IDS2018 and UNSW-NB15 datasets.

5. Conclusion

The detection of the intrusion and prevention of the same in the network is carried out using artificial intelligence (AI) based techniques that increase the detection accuracy of the model. An efficient NIDS model was developed and designed using the hybrid classifier. The major aim of this work was to increase the data for enhancing detection accuracy and reducing overfitting issues. The proposed WSO-based SMOTE undergoes the stages like WSO based clustering, filtering and over-sampling. Then, the features are extracted using IF and FS and classified using the hybrid classifier CAE-ELM. The performance was carried out on the two benchmark datasets, UNSW-NB15 and CSE-CIC-IDS2018 for with and without SMOTE. The experimental analysis proved that the proposed WSO-based SMOTE achieved better accuracy of 99.21% and 99.15% on the CSE-CIC-IDS2018 and 99.14% and 98.86% on the UNSW-NB15 dataset. In future, this proposed model will be applied for the accuracy detection of each type



(a)

(b)

(c)

(d)

Figure. 13 Comparative analysis of UNSW-NB15: (a) Accuracy, (b) Precision, (c) Recall, and (d) F-Measure

of attack. Further, it will be applied to the recent datasets for covering the new types of attacks.

Conflicts of interest

Authors have no conflict of interest.

Author contributions

Data collection & references are prepared by D.Shankar & Dr. S.Saraswathi; Literature survey done by D.Shankar, & Dr. S.Saraswathi; Implementation part done by Dr. N.Kanya; Manuscript prepared by D.Shankar, Dr. G.Victo Sudha George; Proof read & technical check done by Dr. G.Victo Sudha George.

Reference

[1] J. Park, Y. Park, C. I. Kim, “TCAE: Temporal Convolutional Autoencoders for Time Series Anomaly Detection”, In: *Proc. of 2022 Thirteenth International Conference on Ubiquitous and Future Networks*, pp. 421-426, 2022.

[2] E. K. Boahen, S. A. Frimpong, M. M. Ujakpa, R. N. Sosu, O. L. Siaw, E. Owusu, J. K. Appati, and E. Acheampong, “A Deep Multi-architectural Approach for Online Social Network Intrusion Detection System”, In: *Proc. of 2022 IEEE World Conference on Applied Intelligence and Computing (AIC)*, pp. 919-924, 2022.

[3] A. Singh and J. J. Jaccard, “Autoencoder-based Unsupervised Intrusion Detection using Multi-Scale Convolutional Recurrent Networks”, *arXiv Preprint arXiv:2204.03779*. 2022.

[4] A. A. Salih and A. M. Abdulazeez, “Evaluation of classification algorithms for intrusion detection system: A review”, *Journal of Soft Computing and Data Mining*, Vol. 2, No. 1, pp. 31-40, 2021.

[5] Z. Ahmad, A. S. Khan, C. W. Shiang, J. Abdullah, and F. Ahmad, “Network intrusion detection system: A systematic study of machine learning and deep learning approaches”, *Transactions on Emerging Telecommunications Technologies*, Vol. 32, No. 1, p. e4150, 2021.

[6] D. K. Singh and M. Shrivastava, “Evolutionary Algorithm-based Feature Selection for an Intrusion Detection System”, *Engineering, Technology & Applied Science Research*, Vol. 11, No. 3, pp. 7130-4, 2021.

[7] J. Lansky, S. Ali, M. Mohammadi, M. K. Majeed, S. H. Karim, S. Rashidi, M. Hosseinzadeh, and A. M. Rahmani, “Deep learning-based intrusion detection systems: a systematic review”, *IEEE Access*. Vol. 9, pp. 101574-99, 2021.

[8] Y. Wang, J. Ma, A. Sharma, P. K. Singh, G. S. Gaba, M. Masud, and M. Baz, “An exhaustive research on the application of intrusion detection technology in computer network security in sensor networks”, *Journal of Sensors*, Vol. 2021,

p. 1, 2021.

[9] D. Kumar and R.S. Pippal, “Analysis of Data Normalization with Multilevel Classifiers for Intrusion Detection”, *International Journal of Advanced Trends in Computer Science and Engineering*, Vol. 7, No. 5, 2018.

[10] M. A. Janabi and M. A. Ismail, “Improved intrusion detection algorithm based on TLBO and GA algorithms”, *Int. Arab J. Inf. Technology*, Vol. 18, No. 2, pp. 170-9, 2021.

[11] R. Meddeb, F. Jemili, B. Triki, and O. Korbaa, “A Deep Learning based Intrusion Detection Approach for MANET”, *Soft Computing*, pp. 1-15, 2023.

[12] I. Idrissi, M. Azizi, and O. Moussaoui, “An unsupervised generative adversarial network based-host intrusion detection system for internet of things devices”, *Indones. J. Electr. Eng. Comput. Sci*, Vol. 25, No. 2, pp. 1140-50, 2022.

[13] Z. Wang, Y. Zeng, Y. Liu, and D. Li, “Deep belief network integrating improved kernel-based extreme learning machine for network intrusion detection”, *IEEE Access*. Vol. 9, pp. 91-16062, 2021.

[14] Y. Song, S. Hyun, and Y. G. Cheong, “Analysis of autoencoders for network intrusion detection”, *Sensors*, Vol. 21, No. 13, p. 4294, 2021.

[15] F. Aloul, I. Zualkernan, N. Abdalgawad, L. Hussain, and D. Sakhnini, “Network intrusion detection on the IoT edge using adversarial autoencoders”, In: *Proc. of 2021 International Conference on Information Technology (ICIT) IEEE*, pp. 120-125, 2021.

[16] R. Halder, K. J. Fidkowski, and K. J. Maki, “Non-intrusive reduced-order modeling using convolutional autoencoders”, *International Journal for Numerical Methods in Engineering*, Vol. 123, No. 21, pp. 90-5369, 2022.

[17] R. Kalpana, “Recurrent nonsymmetric deep auto encoder approach for network intrusion detection system”, *Measurement: Sensors*. Vol. 24, p. 100527, 2022.

[18] V. Dutta, M. Pawlicki, R. Kozik, and M. Choraś, “Unsupervised network traffic anomaly detection with deep autoencoders”, *Logic Journal of the IGPL*, Vol. 30, No. 6, pp. 25-912, 2022.

[19] S. Zhang, E. J. Carranza, H. Wei, K. Xiao, F. Yang, J. Xiang, S. Zhang, and Y. Xu, “Data-driven mineral prospectivity mapping by joint application of unsupervised convolutional auto-encoder network and supervised convolutional neural network”, *Natural Resources Research*, Vol. 30, pp. 31-1011, 2021.

[20] X. Luo, Y. Jiang, E. Wang, and X. Men, “Anomaly detection by using a combination of generative adversarial networks and convolutional autoencoders”, *EURASIP Journal on Advances in Signal Processing*, Vol. 2022, No. 1, p. 112, 2022.

[21] Y. Xue, “Research on Network Security Intrusion Detection with an Extreme Learning Machine Algorithm”, *International Journal of Network Security*, Vol. 24, No. 1, pp. 29-35, 2022.

[22] H. Lin, Q. Xue, J. Feng, and D. Bai, “Internet of things intrusion detection model and algorithm based on cloud computing and multi-feature extraction extreme learning machine”, *Digital Communications and Networks*, Vol. 9, No. 1, pp. 24-111, 2023.

[23] J. Balasundaram, “Retracted: A novel optimized Bat Extreme Learning intrusion detection system for smart Internet of Things networks”, *International Journal of Communication Systems*, Vol. 34, No. 7, p. e4729, 2021.

[24] Y. Tang and C. Li, “An online network intrusion detection model based on improved Regularized Extreme learning machine”, *IEEE Access*, Vol. 9, pp. 44-94826, 2021.

[25] R. Sekhar, K. Sasirekha, P. S. Raja, and K. Thangavel, “A novel GPU based intrusion detection system using deep autoencoder with Fruitfly optimization”, *SN Applied Sciences*, Vol. 3, No. 6, p. 594, 2021.

[26] G. Andresini, A. Appice, and D. Malerba, “Autoencoder-based deep metric learning for network intrusion detection”, *Information Sciences*, Vol. 569, pp. 27-706, 2021.

[27] T. S. Ayyarao, N. S. Ramakrishna, R. M. Elavarasan, N. Polumahanthi, M. Rambabu, G. Saini, B. Khan, and B. Alatas, “War strategy optimization algorithm: a new effective metaheuristic algorithm for global optimization”, *IEEE Access*, Vol. 10, pp. 105-25073, 2022.

[28] D. Shankar, G.V.S. George, J.N.S.S.J. Naidu, and P.S. Madhuri, “Deep Analysis of Risks and Recent Trends towards Network Intrusion Detection System”, *International Journal of Advanced Computer Science and Applications*, Vol. 14, No. 1, 2023.

[29] D. Shankar, G.V.S. George, and N. Kanya, “OptiBiNet_GRU: Robust Network Intrusion Detection System Using Optimum Bi-Directional Gated Recurrent Unit”, *International Journal of Intelligent Engineering and Systems*, Vol. 16, No. 3, 2023, doi: 10.22266/ijies2023.0630.06.