Journal of Intelligent Systems and Internet of Things
Vol. 15, No. 02, PP. 76-90, 2025
DOI: https://doi.org/10.54216/JISIoT.150206

Ensemble of Machine Learning Model with Tuna Swarm Optimization-Driven Feature Selection for Cybersecurity Threat Detection and Classification Approach

K. Anitha[1,*], K. Rajiv Gandhi[2]

[1]Research Scholar, Department of Computer Applications, Alagappa University, Karaikudi, 630003, Tamilnadu, India

[2]Assistant Professor, Department of Computer Science , Government Arts and Science College For Women, Paramakudi, 623707, Tamilnadu, India

Emails: anitha.kalimuthu@gmail.com; dr.krajiv84@gmail.com

Abstract

The initial identification of cybersecurity events like attacks is challenging provided the continuously growing threat environment. Despite state-of-the-art surveillance, advanced attackers can apply for more than 100 days in a system before being detected. Guaranteeing cyber security is a composite task that depends on area of interest and needs cognitive capabilities to control possible threats from larger quantities of network data. The most important task of a cyber-security analyst is to safeguard a network from damage. Numerous technological developments in network and information security have enabled progressive monitoring and threat detection for the predictors, but the responsibilities they carried out could not be automated completely. Hence, in recent times' Artificial intelligence (AI), mainly deep learning (DL) and machine learning (ML) algorithms, has been utilized to expand a beneficial data-driven intrusion detection system (IDS). Many standard ML classification methods provide intelligent facilities in the area of cyber-security, mainly for intrusion detection. This study develops a Tuna Swarm Optimization-Driven Feature Selection with Ensemble of Machine Learning Models for Cybersecurity Threat Detection and Classification (TSOFSEML-CTDC) technique. The proposed TSOFSEML-CTDC model concentrates on detecting and classifying intrusions on the network. Initially, the TSOFSEML-CTDC algorithm performs data preprocessing using min-max normalization to convert an input data into a beneficial format. Then, the feature selection process has been carried out using tuna swarm optimization (TSO) algorithm. For the classification of intrusion detection, ensemble of ML techniques was employed such as support vector regression (SVR) approach, least-square support vector machines (LSSVM) method, and modified extreme learning machine (MELM) technique. At last, the hyperactive parameter optimization process is executed by using the coati optimization algorithm (COA). The experimental evaluation of the TSOFSEML-CTDC model occurs using a benchmark dataset. The stimulated results emphasized the enhanced performance of the TSOFSEML-CTDC method compared to existing approaches.

Received: September 28, 2024 Revised: November 25, 2024 Accepted: January 14, 2025

Keywords: Tuna Swarm Optimization; Ensemble of Machine Learning; Cybersecurity Threat Detection; Coati Optimization Algorithm; Data Preprocessing

1. Introduction

Intrusion detection is monitoring events in a computer network or system and examining them for intrusion signs. It is described as endeavors to cooperate with the integrity, availability, and confidentiality, or bypass the safety mechanisms of a network or computer [1]. Attackers retrieving the methods from the Internet, legal consumers of the schemes who endeavour to get other rights to which they are not permitted, trigger intruders and legal consumers who exploit the honors provided to them [2]. Intrusion detection systems (IDSs) are hardware or software systems that spontaneously monitor the events arising in a computer network or system [3]. As system, threats have raised in number and severity over the past decades, IDS has become essential to add the safety

infrastructure of most of the administrations. An IDS is a security technology endeavoring to isolate and identify computer systems intrusions [4]. IDSs can be utilized to identify diverse kinds of malicious network communications and usage of computer systems, while the traditional firewall cannot accomplish this challenge. Intrusion detection takes on the behaviors of intruders dissimilar from a legal user [5].

Generally, IDSs can be separated into dual kinds like misuse and anomaly signature recognition depending on their recognition methods [6]. Anomaly recognition determines whether a variation from the well-known normal pattern's usage can be flagged as intrusion. Alternatively, misuse recognition utilizes patterns of well-known threats or weak spots of the method to recognize intrusions [7]. Anomaly detection depends on the normal behavior of a subject for example, a system or a user in any action that extensively diverges from the normal behavior is measured as intrusive. Misuse recognition catches intrusions regarding the features of known systems or threat susceptibilities; any action that matches the pattern of a known vulnerability or threat is taken as intrusive [8]. IDS usually plays a major part in network safety, as its primary aim is to stop interruptions in the communication method. ML Static probability, DL, and neural network methods are applied to assess whether the data is malicious or not [9]. To overwhelm the drawbacks of signature-based threat identification, the investigators primarily concentrate on dynamic models based on DL or ML. The system activities were exposed utilizing the data recorded to classify new threats [10]. IDS is a major investigation topic owing to the substantial violation effects and threats in consumer applications.

This study develops a Tuna Swarm Optimization-Driven Feature Selection with Ensemble of Machine Learning Models for Cybersecurity Threat Detection and Classification (TSOFSEML-CTDC) technique. Initially, the TSOFSEML-CTDC algorithm performs data preprocessing using min-max normalization to convert an input data into a beneficial format. Then, the feature selection process is carried out using tuna swarm optimization (TSO) algorithm. For the classification of intrusion detection, ensemble of ML techniques was employed such as support vector regression (SVR) method, least-square support vector machines (LSSVM) method, and modified extreme learning machine (MELM) technique. At last, the hyperparameter optimization process is executed by using the coati optimization algorithm (COA). The experimental evaluation of the TSOFSEML-CTDC technique takes place using a benchmark dataset.

2. Literature of Works

In [11], an advanced method is presented to improve web server security by applying an IDS-enabled ML model. In search of this improving web server security, this study investigates designing an IDS enabled by ML. The initiation of this method hinges on clear public database, exposed to intense pre-processing stages like normalization, feature selection, and data cleaning. Han et al. [12] developed a new Binary Particle Swarm-Wrapped feature Selection Optimizer Framework (BPSWO) method. The enhanced transfer function is utilized to create the approach focusing on the global optimum element. Then, the Hamming distance and chaotic transformation are utilized to resolve the premature difficulty of the conventional PSO. Afterward, the diverse classifiers are fixed in the particle swarm to train. The BPSWO trains the classifier where feature selection and the last training classifiers are utilized for IDS.

In [13], an effective automated IDS utilizing the ML method is introduced. Min-max normalization and handling Null values are utilized for preprocessing. The class imbalanced difficulty is minimized by utilizing the Advanced Synthetic Minority Oversampling Technique (ASMOTE). Then Modified Singular Value Decomposition (M-SvD) has implemented the feature extraction. The Opposition-based Northern Goshawk Optimization model (ONgO) enhances the feature extraction. The diverse kinds of threats are classified by a hybrid ML method named Mud Ring aided multi-layer Support Vector Machine (M-MultiSVM) and lastly, the hyper-parameters are tuned by the Mud Ring optimizer model. Chirra [14] explores the possible ML-driven methods for developing cyber defense, aiming at next-generation IDS that can intelligently mitigate and identify a broad range of cyber-attacks. By leveraging the power models like unsupervised learning, DL, and supervised learning, these methods can autonomously identify hidden patterns, investigate huge quantities of network data, and adapt to new threat approaches. This research reviews main ML methods and intrusion detection applications. In [15], an Anomaly-based IDS (AIDS) can be presented to mitigate and identify DDoS risks. AIDS applies DL and ML approaches for threat mitigation.

Attou et al. [16] implement cloud-based IDS depending on feature engineering and RF. IDS is an improved mechanism utilized to identify abnormal activities and control traffic inside the systems. In particular, the RF classifier has been incorporated and attained to improve the accuracy (ACC) of the projected recognition method. In [17], the new method presented in this study incorporates ML into IDSs, providing a strong solution to recognize zero-day ransomware threats in hidden datasets. Over rigorous investigation with several ML models containing NNs, SVM, and RF, this research represents that ML methods can efficiently identify formerly unidentified ransomware behavior across the investigation of system anomalies.

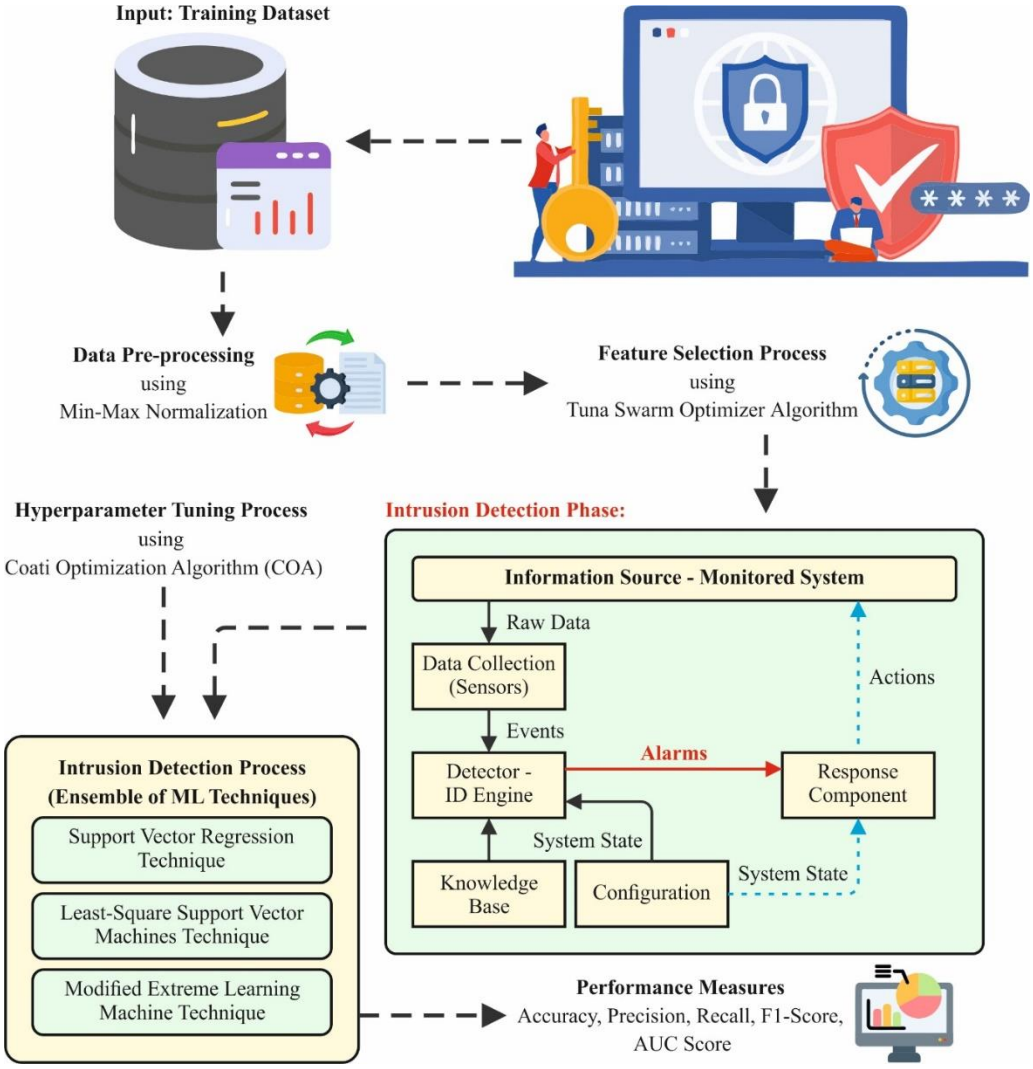

Figure 1. Overall Workflow of TSOFSEML-CTDC model

3. Proposed Methodology

This study develops a TSOFSEML-CTDC technique. The proposed TSOFSEML-CTDC model concentrates on detecting and classifying intrusions on the network. To achieve this, the TSOFSEML-CTDC model involves various stages like data pre-processing, classification model, feature selection, and hyperparameter tuning process. Fig. 1 characterizes the overall flow of the TSOFSEML-CTDC method.

A. Data Preprocessing

Initially, the TSOFSEML-CTDC algorithm performs data preprocessing using min-max normalization to convert input data into a beneficial design. Min-Max normalization is generally utilized in IDS to pre-process network traffic data previously supplying it into ML methods [18]. Scaling the feature values to a normal range, normally among (0,1), aids in guaranteeing that not one feature unequally affects the method because of varying scales. This can be mainly significant in IDS, whereas characteristics such as duration, packet size, or byte count might differ extensively. The normalization method increases the performance of models namely SVM or NN, making them more effective in identifying attacks or anomalies. Nevertheless, it is significant to process anomalies sensibly, as maximum values can alter the scaling procedure.

B. TSO-based Feature Selection

Furthermore, the process of feature selection is performed using TSO algorithm. This study applies a new swarm-based meta-heuristic method called TSO, primarily presented [19]. This model has been applied to enhance the BPNN in the upcoming state prediction unit of the dataset inside our algorithm. Dual different swarm foraging behaviors experimental within the species of tunas stimulates the TSO method. This method outlines stimulation

from the group travel predatory tactic used by tunas. Another tactic is spiral foraging. Such as tunas forage, they make a spiral and drive their prey into water's surface whereas they turn out to be simpler to assault. The next tactic is parabolic foraging. All tuna obey the one before that, making a parabolic form to encircle and catch their prey. The optimizer procedure starts by consistently making a first population within the search space randomly:

$$X_i^{int} = Rand \cdot (b_u - b_\iota) + b_\iota, i = 2,3, \dots, N_p \quad (1)$$

Here, N_p denotes the individual counts in the population of tuna, X_i^{int} symbolizes i^{th} first individual, b_l and b_u represents lower and upper limits of the search space, and $Rand$ means random vector distributed at a uniform range between (0-1). In the stage of spiral, foraging, prior individuals transfer information to the next one, ending in the formation of a constant foraging form. Nevertheless, it is mistaken to instinctively observe the best individual to forage after that an individual cannot find food. In compensation, a random coordinate has been made inside the search area to help as a suggestion idea for spiral searching, awarding the TSO using global searching abilities. This search approach can be associated with a wide global exploration of an accurate pursued search. For example, iterations improve, and the TSO transitions the point of reference for spiral foraging from an arbitrary individual to the best individual. Finally, the last mathematic approach for the spiral foraging tactic can be obtainable as:

$$X_i^{t+1} = \begin{cases} \alpha_1 \cdot (X_{rand}^t + \beta \cdot |X_{rand}^t - X_i^t|) + \alpha_2 \cdot X_i^t, i = 1 \\ \alpha_1 \cdot (X_{rand}^t + \beta \cdot |X_{rand}^t - X_i^t|) + \alpha_2 \cdot X_{i-1}^\tau, i = 2,3, \dots, N_p \end{cases}, if\ rand < \frac{t}{t_{\max}} \quad (2)$$

$$X_i^{t+1} = \begin{cases} \alpha_1 \cdot (X_{best}^t + \beta \cdot |X_{rand}^t - X_i^t|) + \alpha_2 \cdot X_i^t, i = 1 \\ \alpha_1 \cdot (X_{best}^t + \beta \cdot |X_{rand}^t - X_i^t|) + \alpha_2 \cdot X_{i-1}^t, i = 2,3, \cdots, N_p \end{cases} \ if\ rand \geq \frac{t}{t_{\max}} \quad (3)$$

$$\alpha_1 = a + (1 - a) \cdot \frac{t}{t_{\max}} \quad (4)$$

$$\alpha_2 = (1 - a) - (1 - a) \cdot \frac{t}{t_{\max}} \quad (5)$$

$$\beta = e^{bI} \cdot \cos(2\pi b) \quad (6)$$

$$l = e^{3\cos\left(\left(\left(t_{\max} + \frac{1}{t}\right) - 1\right)\pi\right)} \quad (7)$$

Whereas X_i^{t+1} indicates the i^{th} individual toward $t + 1$ iterations. X_{rand}^t refers to a point of reference randomly produced inside the search space. x_{best}^t characteristics of the present optimum individual. α_1 and α_2 are weighted coefficients leading to the tendency of the individual to draw near to the best and the previous one correspondingly. The a constant defines the level such that the tuna observes the best individual and its precursor in the first stage. t represents the present iteration number, even though t_{max} denotes the maximal iteration count. b refers to randomly generated numbers uniformly distributed among (0, 1).

In addition to spiral feeding developments, tuna additionally apply a parabolic feeding design using the food as a benchmark. In Addition, tuna guide their instant environments for nutrition. Both tactics are performed concurrently, all by a fifty percent probability. The calculated approach can be specified below:

$$X_i^{t+1} = \begin{cases} X_{best}^t + rand \cdot (X_{best}^t - X_i^t) + TF \cdot p^2 \cdot (X_{best}^t - X_i^t), if\ rand < 0.5 \\ TF \cdot p^2 \cdot X_i^t, if\ rand \geq 0.5 \end{cases} \quad (8)$$

TF means random value of both 1 or -1. The fitness function (FF) takes into account the classifier precision and the selected feature counts. It increases the classifier accuracy and lessens the set dimensions of the designated features. As a result, the subsequent FF has been applied to assess individual solutions, as exposed in Eq. (9).

$$Fitness = \alpha * ErrorRate + (1 - \alpha) * \frac{\#SF}{\#All\_F} \quad (9)$$

Here $ErrorRate$ represents the classifier error rate using the chosen features. $ErrorRate$ is computed as the incorrect percentage classified to the number of classifications completed, specified as a value amongst (0,1). $\#SF$ denotes selected feature counts and $\#All\_F$ signifies total attribute counts in the novel dataset. α has been applied for controlling the importance of subset length and classifier qualities. In our research, α is fixed to 0.9.

C. Ensemble of ML Models

For the classification of intrusion detection, an ensemble of ML techniques was employed such as the SVR model, LSSVM method, and MELM technique.

i) SVR Classifier

SVM was initially presented to solve classification problems. The aim of SVM in classification is to find a hyperplane in an N-dimensional space that mostly classifies the points of data [20]. The hyperplane has been selected to maximize the margin, which is the space between the hyperplane and the adjacent points of data from both class labels. These adjacent points are denoted as support vectors on the margin boundaries. These support vectors are important because they decide the orientation and position of the hyperplane. After the success of SVM in classification tasks, SVR has been presented for regression problems. SVR works on principles equivalent to SVM but is improved for constant objective variables. SVR intends to discover a function that diverges from the target values by a value not larger than a pre-defined margin (ε). During SVR, the aim is to fit the error in a particular threshold, i. Assuming the size or complexity of the dataset, it is frequently not possible to fit each data point completely inside the margin. As a result, a penalty parameter (C) has been presented to permit some adaptability by allowing data points to lie outside the margin when they are penalized. These outcomes are in an exchange between the flatness of the regression function and the quantity by which deviations greater than ε are accepted. Data points are permitted to lie inside a distance ξ, named the slack variable, from the real margin limit. A kernel function has been applied to convert the data into a higher-dimensional space. This aids the process of non-linear relationships by making the data linearly independent in the transformed space. Some general kinds of kernel functions namely Linear, Sigmoid, Radial Basis Function (RBF), and Polynomial Kernels. SVR aims to discover a function $f(x)$ that contains at most ε deviation from the real targets y for each training data but concurrently maximizes the margin and reduces the weights. This is mathematically stated by Eq. (10):

$$f(x) = w.\phi(x) + b \tag{10}$$

Whereas w refers to the weight vector, $\phi(x)$ means a change of the input data (x) into a higher dimensional area utilizing the kernel function, and b denotes the bias term. The optimization problem in SVR aims to minimize the objective function in Eq. (11)

$$\min \frac{1}{2} w^T w + C \sum_{i=1}^{n} (\xi_i + \xi_i^*) \tag{11}$$

dependent on the limitations Eq. (12):

$$\begin{cases} y_i - w^T \phi(x_i) - b \le \varepsilon + \xi_i \\ w^T \phi(x_i) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_j, \xi_i^* \ge 0 \end{cases} \tag{12}$$

Here C refers to the penalty parameter, n_i and ξ_i^* are slack variables that permit various prediction errors, producing flexibility for data points that drop outside the margin. The term $\frac{1}{2} w^T w$ targets to reduce the weights, which indirectly maximizes the margin. The main hyperparameter in SVR contains C, ε, and the type of kernel.

ii) LSSVM Classifier

The LSSVM model, in particular, the key features that permit representation of the sought-after analogy [21]. This model, in its primary space expression, searches to discover a model as

$$\mathcal{Y} \approx \hat{y} = \mathcal{M}_{LSSVM}(x) = \langle w, \quad \varphi(x) \rangle + b = \sum_{k=1}^{K} w_k \varphi_k(x) + b, \tag{13}$$

Whereas $\varphi(x) = \left(\varphi_1(x), \dots, \varphi_K(x)\right)$ means vector of basic functions (not exactly orthogonal). This method is a parametrical approach using attribute space$\varphi: \mathbb{R}^d \to \mathbb{R}^K$ and is particularly related to, using the term of zero-order signified as b. The primary space method is equally casting in the dual space expression, which states

$$\mathcal{M}_{LSSVM}(x) = \sum_{l=1}^{L} a_l k(x, x_l) + b, \tag{14}$$

whereas $\{x_l\}_{l=1}^{L}$ represents collection of training instances of the input parameter x and

$$k(x, x') = \langle \varphi(x), \varphi(x') \rangle = \sum_{k=1}^{K} \varphi_k(x)\, \varphi(x') \tag{15}$$

means a function of the kernel. Certainly, the dual space method is a linear pattern of function of the kernel positioned at the training instances. The coefficients α and b are calculated by resolving the linear method

$$\begin{pmatrix} \Omega + I_L/\gamma & 1_L \\ 1_L^T & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ b \end{pmatrix} = \begin{pmatrix} y \\ 0 \end{pmatrix} \tag{16}$$

Whereas, $y = (y_1, \dots, y_L)^T$ denotes observations sector assessed at the training samples, for example, $y_l = \mathcal{M}(x_l)$ for $l = 1, \dots, L$; I_L means LxL identity matrix and $1_L = (1, \dots, 1)^T \in \mathbb{R}^L$ represents one column vector; γ refers to

regularization hyperparameter, which can be enhanced by minimizing some error metric, for example; an error of cross-validation; Ω stands for LxL Gram matrix with entries

$$\Omega_{lm} = k(x_l, x_m) = \sum_{k=1}^{K} \varphi_k\,(x_l)\varphi_k(x_m) \quad (17)$$

Furthermore, the coefficients α fulfill

$$\sum_{l=1}^{L} \alpha_l = 0 \quad (18)$$

When the dare space coefficients α are calculated, the original space coefficients w are gained as

$$w_k = \sum_{l=1}^{L} \alpha_l\, \varphi_k(x_l) \quad (19)$$

For example: as a linear arrangement of the base operations assessed toward the training instances. There dual characteristics that make the LSSVM method attractive. Initially, the dual space expression includes $L + 1$ terms, which are gained by resolving the $(L + 1)$x$(L + 1)$ linear method, rather than $K + 1$ terms as in the primary area or PCE methods. Therefore, in complexity which dual space method can be established by the accessible training data, which generally we wanted to be lower than the dimensions K of the space of the feature. Most significantly, the internal products within the kernel explanation do not require to be calculated clearly, then implicit (some) function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ that fulfils Mercer's condition, for example;

$$\iint_{\mathbb{R}^d \times \mathbb{R}^d} g\,(x)k(x, x')g(x')dxdx' \geq 0, \forall g(x) \in L^2, \quad (20)$$

represents valid kernel for the LSSVM. Therefore, the kernel matrix Ω in and the predictions of the model are calculated by simple function assessments, which is called the Kernel Trick.

$$k(x, x') = (c + x^T x')^p \quad (21)$$

and the RBF (or squared exponential) kernel

$$k(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right), \quad (22)$$

Here c, p, and σ are hyperparameters that are naturally enhanced within the training procedure. Conversely, a kernel is continuously built particularly, more commonly, as

$$k(x, x') = \sum_{k=1}^{K} \lambda_k\, \varphi_k(x)\varphi_k(x'), \quad (23)$$

which fulfills offered that $\lambda_k > 0, \forall k$.

iii) MELM Classifier

The last stage in the classification of HSI comprises determining a classification approach over an ML method. The generally well-organized model suggested for a Single Hidden Layer Feedforward Network (SLFN) is ELM [22]. Nevertheless, at particular periods, the ELM can verify insufficient due to the bias and weight random parameters choice inside the Hidden Layer (HL) and Input Layer (IL). To deal with these challenges, a Modified Extreme Learning Machine (MELM) has been introduced to tackle the tasks of ELM. The MELM extracts the requirement for weight initialization in the HL and IL. Fig. 2 represents the architecture of MELM Classifier

The train data (y_q, z_q) contains $l-$instances, followed by the HL within the MELM can be provided as h. The activation term gy of the (y_q, z_q) MELM is provided as:

$$\sum_{q=1}^{h} \beta_q\, g(y_q) = \sum_{q=1}^{h} \beta_q\, g(w_q y_q + b_q) = z_p \quad (24)$$

Whereas $p = 1,2,3 \ldots k$. The result of MELM can be provided as:

$$N_h(y) = \sum_{q=1}^{h} \beta_q\, g(w_q y_q + b_q) = \gamma h(y) \quad (25)$$

Here w_q, β_q carry the output, weight vector, and bias values of q^{th}hidden neurons. $h(y)$ denotes HL; Nevertheless, there are samples where the total hidden neuron counts are significantly smaller than the complete training examples. These situations result in the $J\ th$ variable establishing a non-square matrix. Subsequently, a matrix is non-square, and the survival of β has been committed. Nevertheless, it is necessary to derive β to evaluate the weight of the output neurons.

$$\beta^f = J^* U \quad (26)$$

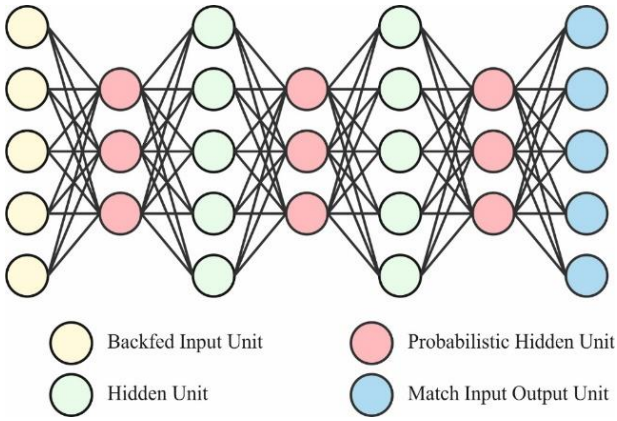

Figure 2. Structure of MELM Classifier

Here J^* means the output of the MELM. The kernel term can be combined with the MELM using $\frac{1}{u}$ and it is signified as:

$$\beta = J^T \left(\frac{1}{u} + JJ^T\right)^{-1} U \tag{27}$$

The feature mapping for $h(y)$ can be unidentified, and afterward kernel term has been presented to the ELM. The ELM's kernel term is ψ.

$$\psi_{ELM} = h(y_q)h(y_p) = K(y_q, y_p) \tag{28}$$

Here $K\ (y_q, y_p)$ represents term kernel. The kernel using the output can be provided as:

$$\begin{bmatrix} K(y, y_1) \\ \vdots \\ K(y, y_k) \end{bmatrix}^T \left(\frac{1}{u} + \psi_{ELM}\right)^{-1} U \tag{29}$$

The selection of activation function significantly influences the precision functioning in MELM training. The traditional function of sigmoid helps as a strong threshold function and is normally used as the function of the activation in MELM. It can be identified as:

$$f(x) = \frac{1}{1+e^{-(x)}} \tag{30}$$

Softplus presents a systematic smooth calculation for Rectified Linear Units (ReLU). Described as shown: The function of Softplus is non-linear, differentiable, and continuous like the activation method performed in environmental science more accurately than the function of Sigmoid. In this work, for improved generalizability, this paper includes Softplus into the ELM structure, substituting sigmoid. This variation can be denoted as Softplus-modified ELM.

$$f(x) = \ln\left(1 + e^{-(x)}\right) \tag{31}$$

D. COA-based Hyperparameter Tuning Process

Utimately, the hyperparameter optimization process is executed by using the **COA.** The COA is mainly classified into dual phases: development and exploration. Initially, begin by arbitrarily allocating the coati count and their positions inside the identified variety of boundaries [23]. In the exploratory stage, it can be assumed that the most beneficial location for the present population is the position of iguanas, with a partial of the coati species climbing trees whereas the rest of them stay stationary, waiting for food to decline. The coati's position on the tree can be:

$$x_i^{t+1}(j) = x_i^t(j) + r\left(x_{best}^t(j) - Ix_i^t(j)\right) \tag{32}$$

Whereas i ranges between 1 to $N/2$. The coati's earth location is adapted to:

$$Iguana_{ground}^t(j) = lb_j + r\left(ub_j - lb_j\right) \tag{33}$$

$$x_i^{t+1}(j) = \begin{cases} x_i^t(j) + r\left(Iguana_{ground}^t(j) - Ix_i^t(j)\right), & if\ fitness\left(Iguana_{ground}^t\right) < fitness(x_i^t) \\ x_i^t(j) + r\left(x_i^t(j) - Iguana_{ground}^t(j)\right), & else \end{cases} \tag{34}$$

Here $i = N/2 + 1,\ N/2 + 2, \ldots,\ N; N$ represents size of the population; t characterizes the current number of iterations; lb_j and Ub_j are lower and upper limits on the jth variable dimension; r denotes randomly generated numbers among (0,1); I signifies an arbitrary integer of a set (1,2); $Igunaua_{ground}^{t}$ signifies the novel position of the iguana once landed; $x_i^{t+1}(j)$ refers to jth parameter dimension for the ith individual in the present iteration. When the novel position of every individual improves the objective function values, the location is upgraded; or else, the location has remained unchanged:

$$x_i^{t+1}(j) = \begin{cases} x_i^{t+1}, & if\ fitness\left(x_i^{t+1}\right) < fitness(x_i^t) \\ x_i^t, & else \end{cases} \tag{35}$$

In the development stage, an arbitrary location has been made near every individual location to duplicate the behavior of coatis feeding predators.

$$lb_j^{local} = \frac{lb_j}{t} \tag{36}$$

$$ub_j^{local} = \frac{ub_j}{t} \tag{37}$$

$$x_i^{t+1}(j) = x_i^t(j) - (1 - 2r) \cdot \left(lb_j^{local} + r \cdot \left(ub_j^{local} - lb_j^{local}\right)\right) \tag{38}$$

Here t range between (1-T); i range between (1-N); r represents randomly generated number among (0,1). T symbolizes the higher boundary for the iteration counts. lb_j^{local} and ub_j^{local} signify the minimum and maximum bounds of the jth parameter dimension, which alteration by every iteration.

The fitness selection is the extensive feature prompting the performance in the COA. The hyperparameter choice procedure contains the solution encoding approach to evaluate the efficacy of the candidate solutions. In this paper, the COA considered accuracy as the major condition to propose the FF, which is expressed as shown.

$$Fitness\ =\ \max(P) \tag{39}$$

$$P = \frac{TP}{TP + FP} \tag{40}$$

Here, TP symbolizes the true positive and FP designates the false positive value.

4. Performance Analysis

The stimulated analysis of the TSOFSEML-CTDC approach is examined under dual datasets such as CICIDS-2017 [24] and ToN-loT [25]. The CICIDS-2017 dataset contains 10973 records under five classes as illustrated in Table 1. The total feature counts is 78 but only 42 features are selected.

Table 1: Details of CICIDS-2017 Dataset

CICIDS-2017 Dataset	
Classes	No. of Records
Benign	2500
DDoS	2500
DoS	2500
Bot	1966
Web Attack	1507
Total Records	10973

Fig. 3 offers the classifier outcomes of the TSOFSEML-CTDC methodology under the CICIDS-2017 dataset. Figs. 3a-3b presents the confusion matrix with accurate recognition and classification of all classes under 70%TRPH and 30%TSPH. Fig. 3c establishes the PR investigation, representing superior performance through all class labels.

At the same time, Fig. 3d exemplifies the ROC inspection, signifying capable results with better values of ROC for different classes.

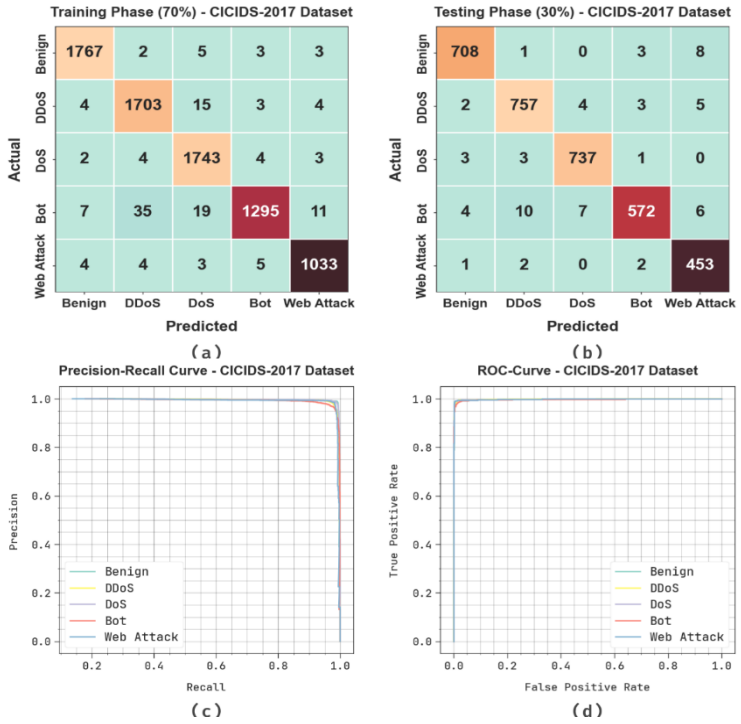

Figure 3. CICIDS-2017 dataset (a-b) confusion matrix, (c) curve of PR, and (d) curve of ROC

Table 2 represents the intrusion detection of TSOFSEML-CTDC algorithm under CICIDS-2017 dataset. The outcomes imply that the TSOFSEML-CTDC technique correctly recognized the samples. With 70%TRPH, the TSOFSEML-CTDC system offers average $accu_y$, $prec_n$, $reca_l$, $F1_{score}$ and AUC_{score} of 99.27%, 98.20%, 98.05%, 98.11%, and 98.79%, respectively. Besides, with 30%TSPH, the TSOFSEML-CTDC methodology offers average $accu_y$, $prec_n$, $reca_l$, $F1_{score}$ and AUC_{score} of 99.21%, 97.90%, 98.00%, 97.94%, and 98.75%, correspondingly.

Table 2: Intrusion detection of TSOFSEML-CTDC method under CICIDS-2017 dataset

Class Labels	$Accu_y$	$Prec_n$	$Reca_l$	$F1_{score}$	AUC_{score}
TRPH (70%)					
Benign	99.61	99.05	99.27	99.16	99.49
DDoS	99.08	97.43	98.50	97.96	98.87
DoS	99.28	97.65	99.26	98.45	99.28
Bot	98.87	98.85	94.73	96.75	97.25
Web Attack	99.52	98.01	98.47	98.24	99.08
Average	99.27	98.20	98.05	98.11	98.79
TSPH (30%)					
Benign	99.33	98.61	98.33	98.47	98.97
DDoS	99.09	97.93	98.18	98.06	98.77
DoS	99.45	98.53	99.06	98.79	99.31
Bot	98.91	98.45	95.49	96.95	97.58
Web Attack	99.27	95.97	98.91	97.42	99.12
Average	99.21	97.90	98.00	97.94	98.75

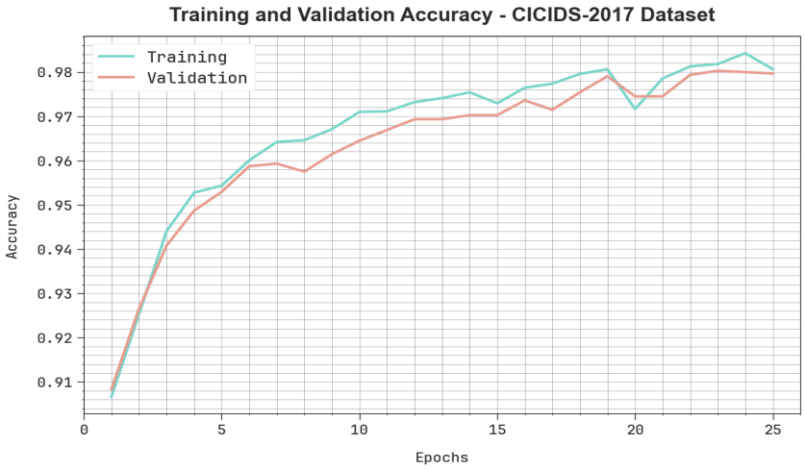

Figure 4. $Accu_y$ Curve of TSOFSEML-CTDC method under CICIDS-2017 dataset

In Fig. 4, the training (TRA) $accu_y$ and validation (VAL) $accu_y$ results of the TSOFSEML-CTDC approach under CICIDS-2017 dataset are illustrated. The $accu_y$ values are computed for 0-25 epoch counts. The figure highlights that the TRA and VAL $accu_y$ values demonstrate growing tendencies, which informed the capacity of the TSOFSEML-CTDC algorithm with maximum outcomes across various iterations. Followed by, the TRA and VAL $accu_y$ remainders closer across the epoch counts, which identifies inferior overfitting and presents greater performance of the TSOFSEML-CTDC methodology, pledging dependable prediction on undetected instances.

In Fig. 5, the TRA loss (TRALOS) and VAL loss (VALLOS) curve of the TSOFSEML-CTDC algorithm under CICIDS-2017 dataset is demonstrated. The values of loss are computed throughout 0-25 epochs. It is denoted that the TRALOS and VALLOS values exemplify declining tendencies, indicating the ability of the TSOFSEML-CTDC algorithm to balance an exchange between generality and data fitting. The continuous reduction in values of loss also assurances the superior performance of the TSOFSEML-CTDC methodology and tunes the prediction outcomes gradually.

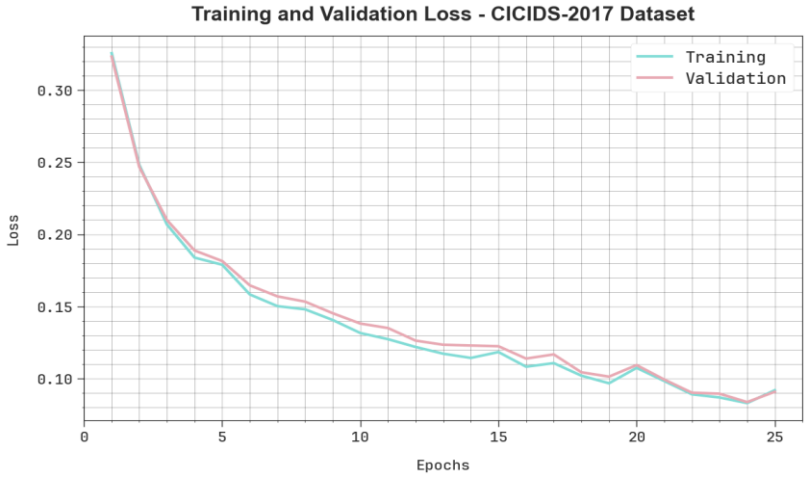

Figure 5. Loss graph of TSOFSEML-CTDC method under CICIDS-2017 dataset

Table 3 inspects the comparison results of the TSOFSEML-CTDC system under CICIDS-2017 dataset with the existing methods [26,27, and 28]. The results emphasized that the GCNN-AE, Kernel SVM, AE-GAN, DT, KNN, and RF algorithms have reported worse performance. Simultaneously, POADEL-ID model has gained closer outcomes. Likewise, the TSOFSEML-CTDC approach reported maximal performance with minimal $prec_n, reca_l, accu_y, and\ F1_{score}$ of 98.20%, 98.05%, 99.27%, and 98.11%, respectively.

Table 3: Comparative analysis of TSOFSEML-CTDC technique under CICIDS-2017 dataset

CICIDS-2017 Dataset				
Methodology	$Accu_y$	$Prec_n$	$Reca_l$	$F1_{score}$
TSOFSEML-CTDC	99.27	98.20	98.05	98.11
POADEL-ID	99.16	97.85	97.76	97.47
AE-GAN Method	94.04	94.17	96.42	93.96
GCNN-AE Model	90.35	97.26	85.37	90.98
Kernel SVM	93.64	95.76	89.80	90.82
Decision Tree	94.04	92.10	93.84	92.85
KNN Algorithm	99.06	96.07	96.08	98.78
RF Methodology	98.791	95.799	94.858	97.29

The ToN-loT dataset covers 119957 instances under nine classes as shown in Table 4. The total number of features is 75 but only 51 features are selected.

Table 4: Details of ToN-loT Dataset

ToN-loT Dataset	
Class	No. of Instances
Normal	78369
MiTM	336
DoS	5440
DDoS	5987
Password	6016
Injection	5867
xss	5951
Ransomware	5976
Backdoor	6015
Total Instances	119957

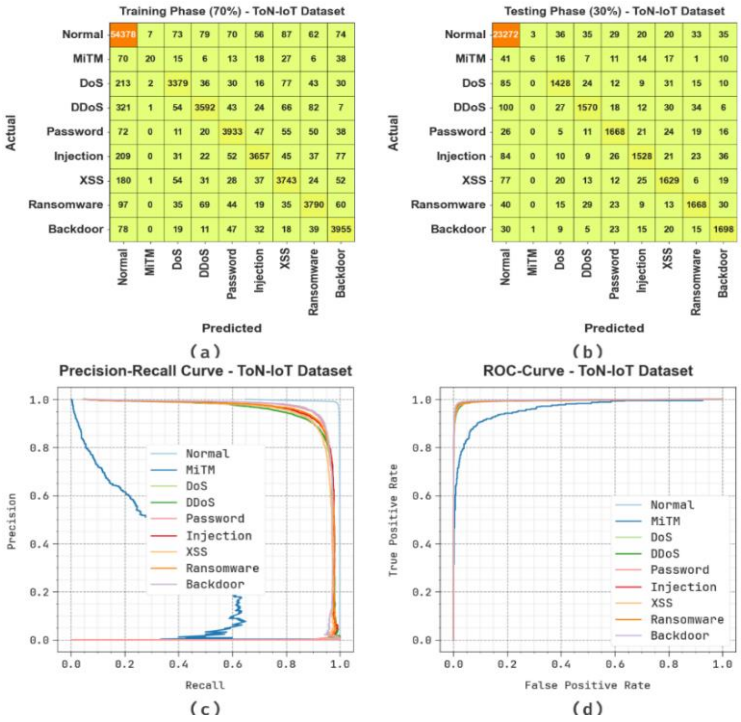

Figure 6. ToN-loT dataset (a-b) confusion matrix, (c) curve of PR, and (d) curve of ROC

Fig. 6 offers the classifier outcomes of the TSOFSEML-CTDC technique under ToN-loT dataset. Figs. 6a-6b demonstrates the confusion matrices with accurate recognition and classification of all classes under 70%TRPH and 30%TSPH. Fig. 6c establishes the PR investigation, signifying better performance through all class labels. In the meantime, Fig. 6d exemplifies the ROC study, signifying proficient results with great ROC values for various classes.

Table 5 presents the intrusion detection of TSOFSEML-CTDC algorithm under ToN-loT dataset. The outcomes imply that the TSOFSEML-CTDC methodology correctly identified the samples. With 70%TRPH, the TSOFSEML-CTDC system offers average $accu_y$, $prec_n$, $reca_l$, $F1_{score}$ and AUC_{score} of 99.07%, 89.59%, 82.21%, 83.58%, and 90.71%, respectively. Furthermore, with 30%TSPH, the TSOFSEML-CTDC approach offers average $accu_y$, $prec_n$, $reca_l$, $F1_{score}$ and AUC_{score} of 99.06%, 88.76%, 81.80%, 82.66%, and 90.52%, correspondingly.

Table 5: Intrusion detection of TSOFSEML-CTDC method under ToN-loT dataset

Class Labels	$Accu_y$	$Prec_n$	$Reca_l$	$F1_{score}$	AUC_{score}
TRPH (70%)					
Normal	97.92	97.77	99.07	98.42	97.41
MiTM	99.76	64.52	09.39	16.39	54.69
DoS	99.12	92.05	88.32	90.14	93.98
DDoS	98.96	92.91	85.73	89.18	92.69
Password	99.26	92.32	93.07	92.69	96.33
Injection	99.14	93.63	88.55	91.02	94.12
XSS	99.03	90.13	90.19	90.16	94.84
Ransomware	99.16	91.70	91.35	91.52	95.46
Backdoor	99.26	91.32	94.19	92.73	96.86
Average	99.07	89.59	82.21	83.58	90.71
TSPH (30%)					
Normal	98.07	97.97	99.10	98.53	97.62
MiTM	99.66	60.00	04.88	09.02	52.43
DoS	99.10	91.19	88.48	89.81	94.04
DDoS	99.00	92.19	87.37	89.71	93.49
Password	99.23	91.55	93.18	92.36	96.37
Injection	99.07	92.44	87.97	90.15	93.80
XSS	99.03	90.25	90.45	90.35	94.97
Ransomware	99.15	91.95	91.30	91.62	95.43
Backdoor	99.22	91.29	93.50	92.38	96.51
Average	99.06	88.76	81.80	82.66	90.52

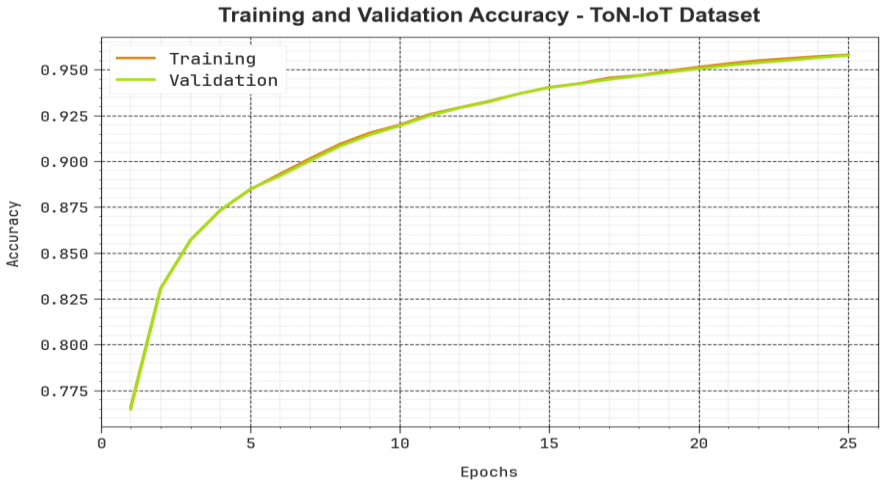

Figure 7. $Accu_y$ Curve of TSOFSEML-CTDC method under ToN-loT dataset

In Fig. 7, the TRA $accu_y$ and VAL $accu_y$ analysis of the TSOFSEML-CTDC technique under ToN-loT dataset is displayed. The $accu_y$ values are computed throughout 0-25 epoch counts. The figure highlights that the TRA and VAL $accu_y$ values demonstrate increasing tendencies that informed the capacity of the TSOFSEML-CTDC method with superior outcomes over various iterations. Meanwhile, the TRA and VAL $accu_y$ stay closer across the epoch counts, which indicates inferior overfitting and displays better performance of the TSOFSEML-CTDC model, pledging continual prediction on unidentified samples.

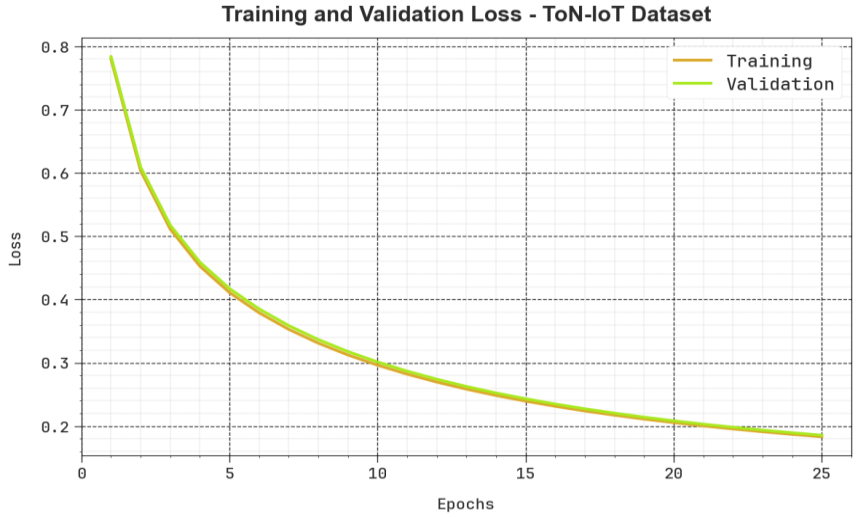

Figure 8. Loss analysis of TSOFSEML-CTDC method under ToN-loT dataset

In Fig. 8, the TRALOS and VALLOS curve of the TSOFSEML-CTDC algorithm under ToN-loT dataset is demonstrated. The values of loss are calculated throughout 0-25 epochs. It is signified that the TRALOS and VALLOS values show-decreasing tendencies, indicating the capacity of the TSOFSEML-CTDC technique to balance a trade-off between data fitting and generality. The constant fall in values of loss furthermore assurances the improved performance of the TSOFSEML-CTDC technique and tunes the prediction results gradually.

Table 6 studies the comparison results of the TSOFSEML-CTDC algorithm under ToN-loT dataset with the existing methods. The results emphasized that the IDS-EESAEE, RF, GDNN-AE, LSTM, GRU and GLSTM approaches have reported inferior performance. Meanwhile, EBWO-HDLID system have achieved closer outcomes. In addition, the TSOFSEML-CTDC algorithm reported enhanced performance with better $prec_n$, $reca_l$, $accu_y$, $and\ F1_{score}$ of 89.59%, 82.21%, 99.07%, and 83.58%, respectively.

Table 6: Comparative analysis of TSOFSEML-CTDC technique under ToN-loT dataset

ToN-loT Dataset				
Method	$Accu_y$	$Prec_n$	$Reca_l$	$F1_{score}$
TSOFSEML-CTDC	99.07	89.59	82.21	83.58
EBWO-HDLID	98.88	89.26	79.03	79.57
IDS-EESAEE	96.07	89.17	76.30	75.88
Random Forest	97.09	86.17	77.92	77.99
GDNN-AE Method	89.80	82.89	77.56	79.11
LSTM Approach	97.03	88.04	76.76	77.97
GRU Model	96.55	89.76	75.79	75.85
GLSTM Model	85.50	78.39	76.88	75.11

5. Conclusion

This study develops a TSOFSEML-CTDC technique. The proposed TSOFSEML-CTDC model concentrates on detecting and classifying intrusions on the network. Initially, the TSOFSEML-CTDC algorithm performs data preprocessing using min-max normalization to convert an input data into a beneficial format. Next, the feature selection process is performed using TSO algorithm. For the classification of intrusion detection, ensemble of ML techniques was employed such as SVR model, LSSVM method, and MELM technique. At last, the hyperparameter optimization process is executed by using the **COA.** The experimental evaluation of the TSOFSEML-CTDC method occurs using a benchmark dataset. The stimulated results emphasized the enhanced performance of the TSOFSEML-CTDC method compared to existing approaches.

Funding: "This research received no external funding"

Conflicts of Interest: "The authors declare no conflict of interest."

References

[1] O. Y. Al-Jarrah, C. Maple, M. Dianati, D. Oxtoby, and A. Mouzakitis, "Intrusion detection systems for intra-vehicle networks: A review," *IEEE Access*, vol. 7, pp. 21266–21289, 2019.

[2] L. Yang, A. Moubayed, and A. Shami, "MTH-IDS: A multitiered hybrid intrusion detection system for Internet of Vehicles," *IEEE Internet of Things Journal*, vol. 9, pp. 616–632, 2021.

[3] W. Wu, R. Li, G. Xie, J. An, Y. Bai, J. Zhou, and K. Li, "A survey of intrusion detection for in-vehicle networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, pp. 919–933, 2019.

[4] K. Agrawal, T. Alladi, A. Agrawal, V. Chamola, and A. Benslimane, "NovelADS: A novel anomaly detection system for intra-vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, pp. 22596–22606, 2022.

[5] A. Goyal, S. Mishra, and V. K. Chaurasiya, "Intrusion detection in wireless sensor networks using deep learning," in *Proc. 4th Int. Conf. Emerging Technologies (INCET)*, Belgaum, India, May 2023, pp. 1–13.

[6] P. Muragod and V. Reddy, "Animal intrusion detection using various deep learning models," in *Proc. IEEE North Karnataka Subsection Flagship Int. Conf. (NKCon)*, Vijaypur, India, Nov. 2022, pp. 1–12.

[7] A. Alotaibi and M. A. Rassam, "Enhancing the sustainability of deep learning-based network intrusion detection classifiers against adversarial attacks," *Sustainability*, 2022.

[8] J. Khan, D.-W. Lim, and Y.-S. Kim, "Intrusion detection system CANBus in-vehicle networks based on the statistical characteristics of attacks," *Sensors*, vol. 23, no. 8, p. 3554, 2023.

[9] A. A. Alsulami, Q. Abu Al-Haija, A. Alqahtani, and R. Alsini, "Symmetrical simulation scheme for anomaly detection in autonomous vehicles based on LSTM model," *Symmetry*, vol. 14, no. 8, p. 1450, 2022.

[10] A. Maseleno, "Design of optimal machine learning-based cybersecurity intrusion detection systems," *Journal of Cybersecurity and Information Management*, vol. 4, pp. 32–43, 2019.

[11] A. Tedyyana, O. Ghazali, and O. Purbo, "Model design of intrusion detection system on web server using machine learning based," in *Proc. 11th Int. Applied Business and Engineering Conf. (ABEC)*, Bengkalis, Indonesia, Sep. 2023.

[12] Y. Han, Y. Wang, Y. Cao, Z. Geng, and Q. Zhu, "A novel wrapped feature selection framework for developing power system intrusion detection based on machine learning methods," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2023.

[13] A. V. Turukmane and R. Devendiran, "M-MultiSVM: An efficient feature selection assisted network intrusion detection system using machine learning," *Computers & Security*, vol. 137, p. 103587, 2024.

[14] B. R. Chirra, "Advancing cyber defense: Machine learning techniques for next-generation intrusion detection," *International Journal of Machine Learning Research in Cybersecurity and Artificial Intelligence*, vol. 14, no. 1, pp. 550–573, 2023.

[15] M. E. Manaa, S. M. Hussain, S. A. Alasadi, and H. A. Al-Khamees, "DDoS attacks detection based on machine learning algorithms in IoT environments," *Inteligencia Artificial*, vol. 27, no. 74, pp. 152–165, 2024.

[16] H. Attou, A. Guezzaz, S. Benkirane, M. Azrour, and Y. Farhaoui, "Cloud-based intrusion detection approach using machine learning techniques," *Big Data Mining and Analytics*, vol. 6, no. 3, pp. 311–320, 2023.

[17] Y. Brinkley, D. Thompson, and N. Simmons, "Machine learning-based intrusion detection for zero-day ransomware in unseen data," 2024.

[18] M. S. Yadav and R. Kalpana, "Data preprocessing for intrusion detection system using encoding and normalization approaches," in *Proc. 11th Int. Conf. Advanced Computing (ICoAC)*, Dec. 2019, pp. 265–269.

[19] W. Li and R. K. Pandit, "Data-centric predictive control with tuna swarm optimization-backpropagation neural networks for enhanced wind turbine performance," *Renewable Energy*, vol. 215, p. 121821, 2024.

[20] N. Alidadi and S. Pezeshk, "State of the art: Application of machine learning in ground motion modeling," *SSRN*, 2024. [Online]. Available: https://ssrn.com/abstract=5002073.

[21] P. Manfredi and R. Trinchero, "Nonparametric formulation of polynomial chaos expansion based on least-square support-vector machines," *Engineering Applications of Artificial Intelligence*, vol. 133, p. 108182, 2024.

[22] V. A. A. Daniel, K. Vijayalakshmi, P. P. Pawar, D. Kumar, A. Bhuvanesh, and A. J. Christilda, "Enhanced affinity propagation clustering with a modified extreme learning machine for segmentation and classification of hyperspectral imaging," *e-Prime–Advances in Electrical Engineering, Electronics and Energy*, vol. 9, p. 100704, 2024.

[23] Y. Lv et al., "Rock dynamic strength prediction in cold regions using optimized hybrid algorithmic models," *Geomechanics and Geophysics for Geo-Energy and Geo-Resources*, vol. 10, no. 1, pp. 1–29, 2024.

[24] Kaggle, "Network intrusion dataset." [Online]. Available: https://www.kaggle.com/datasets/ chethuhn/ network-intrusion-dataset.

[25] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustainable Cities and Society*, vol. 72, Sep. 2021, Art. no. 102994.

[26] C. Park, J. Lee, Y. Kim, J. G. Park, H. Kim, and D. Hong, "An enhanced AI-based network intrusion detection system using generative adversarial networks," *IEEE Internet of Things Journal*, vol. 10, no. 3, pp. 2330–2345, 2022.

[27] F. S. Alrayes et al., "Optimizing security protocol: A synergy of bio-inspired planet optimization algorithm with ensemble learning-based attack detection for connected and autonomous vehicles," *IEEE Access*, 2024.

[28] R. Y. Aburasain, "Enhanced Black Widow Optimization with hybrid deep learning-enabled intrusion detection in Internet of Things-based smart farming," *IEEE Access*, 2024.