Journal of Electronic Testing (2024) 40:469–485
https://doi.org/10.1007/s10836-024-06129-3

RESEARCH

Dynamic Smartcard Protection and SSELUR-GRU-Based Attack Stage Identification in Industrial IoT

S. K. Mouleeswaran[1] · K. Ramesh[2] · K. Manikandan[3] · VivekYoganand Anbalagan[4]

Received: 2 January 2024 / Accepted: 11 July 2024 / Published online: 30 July 2024

Abstract
In recent years, the Industrial Internet of Things (IoT) has grown significantly. Automation along with intelligence introduces a slew of cyber risks while implementing industrial digitalization. But, none of the prevailing work focused on provoking alerts to future attacks and protecting the dynamic smart card from malicious attacks.Therefore, a Smooth Scaled Exponential Linear Unit and Reinforcement Learning-based Gated Recurrent Unit (SSELUR-GRU)-based stage identification and dynamic smart card protection are proposed in this paper.Primarily, the data pre-processing is done, and the preprocessed data are balanced using the ADASYN technique. Then, the data is clustered using the CD-KM algorithm for the feasible training of the data. After that, the clustered data is normalized and the patterns of normalized data are analyzed. Further, the important features are chosen by employing the proposed LWSO algorithm for minimizing the processing time of the classifier. Both the obtained optimal features and the patterns are data trained using Log Mish-based Pyramid Net (LM-PN), for classifying the attacked and non-attacked data. In contrast, the input data features and the attacked data are trained by using the proposed SSELUR-GRU for identifying the attack stages.Thus, based on the attack stage, the dynamic card is protected by updating its number, or else the admin is alerted.The experimental outcome stated that when analogized to prevailing methodologies, the proposed method withstands a maximum accuracy of 98.71% and a higher identification rate of 98.21%.

Keywords Cayley Distance based K-Means (CD-KM) · Lehmer-based war strategy optimization algorithm (LWSO) · Log Mish based Pyramid Net (LM-PN) · Smooth Scaled Exponential Linear Unit and Reinforcement Learning based Gated Recurrent Unit (SSELUR-GRU) · Adaptive Synthetic (ADASYN)

Responsible Editor: C. A. Papachristou.

✉ VivekYoganand Anbalagan
anbuvivekram1@yahoo.com

S. K. Mouleeswaran
meetmoulee@gmail.com

K. Ramesh
nishusishu@gmail.com

K. Manikandan
kmanikandan@vit.ac.in

1 Department of CSE, Dayanandasagar university, Bengaluru, Karnataka 560078, India

2 Department of Electronics and Communication Engineering, P.S.V College of Engineering & Technology, Mittapalli, Balinayanapalli (PO), Elathagiri (SO), Krishnagiri District, Tamil Nadu 635 108, India

3 Department of Computer Science and Engineering, Vellore Institute of Technology (VIT), Vellore, Tamil Nadu 632014, India

4 Department of Cyber Security, REDHAT INDIA Pvt Limited, Doddanekkundi, Bengaluru, Karnataka 560037, India

Abbreviations

CD-KM	Cayley Distance based K-Means
LWSO	Lehmer-based war strategy optimization algorithm
LM-PN	Log Mish based Pyramid Net
SSELUR-GRU	Smooth Scaled Exponential Linear Unit and Reinforcement Learning based Gated Recurrent Unit
ADASYN	Adaptive Synthetic
LM-PN	Log Mish-based Pyramid Net
IoT	Internet of Things
IDS	Intrusion Detection System

1 Introduction

The utilization of the Internet of Things (IoT) technologies in the industrial sector are referred to as "IIoT". Its main goal is to combine the advanced IoT and industrial automation technologies. The integration of sensors, devices, and any instruments that are connected with computer-based industrial applications for optimizing the device process is defined as the Industrial IoT (IIoT) [1]. A previously unheard-of integration of production, monitoring, as well as management subsystems is made possible [2]. The unified management processes different industrial data more efficiently [3]. In other directions, the network's vulnerability increases with Catastrophic harm,which is quickly caused by any failure orirregularity in thesystem's component [4]. Thus, while transmitting, the data gets attacked by hackers. Network intrusions typically encompass several attack classes, and each class may have numerous attack types, which causes two main issues,namely 1) Out-of-distribution problem and 2) Multiclass classification problem on unbalanced data. Thus, the Intrusion Detection System (IDS) is developed for the efficient network reaction and security from intrusions [5–7].

Since IDS facilitates the protection of the confidentiality, integrity, and security of data, it plays a significant role in the IIoT [8]. The IDS is accountable for tasks, including preventing, detecting,and reporting harmful behavior as well as partially or completely protecting an IIoT network [9]. For secure IIoT, IDS should also identify the attacks when hackers start probing devices [10]. The design and implementation of appropriate security solutions can be accomplishedby feature extraction [11]. The '2' mutual levels for IDS prototypes are data preprocessing and identification, which arerequisite for enhanced accuracy detection approaches [12]. The identification procedure is eliminated after the removal of redundant features from the dataset [13]. This mitigated feature set may be then employed to forecast attack classes by employing the base classifier [14].

Deep Learning (DL)-based IDS, particularly the Deep Neural Network (DNN) model, Recurrent Neural Network (RNN), Long-Short Term Memory (LSTM), and Gated Recurrent Unit (GRU) models were the possible IDS techniques a few years ago [13]. Nevertheless, these DL techniques are not successful in identifying data attacks with an uneven distribution. Furthermore, the pre-existing models didn't focus on the classification of attack stages, which was also regarded as one of the factors of the advanced IDS system. Therefore, a novel SSELUR-GRU-based attack stage identification with attack classification is proposed in this work.

2 Research Motivation

Nowadays, smart card technology has evolved in every sector of the industry, including public services, education, healthcare, business, and so on. This smart card accommodates the user'ssensitive information along with the personal details. In the business sector, the individual information and their production details are updated in the smartcards.The corruption or tampering made in such smart cards collapses the user details and the industrial machine processes. Hence, it is necessary to analyze suspicious behaviour before protecting the dynamic smart cards. The data attack identification was also attempted in the existing works. However, the future alert about the influence of attack is not provided to the user. Therefore, this work is motivated inorder to fulfill the mentioned gap along with improving the attack detection performance for protecting the smartcard.

2.1 Problem Statement

The prevailing research methodologies' limitations are discussed as follows,

- Prevailing works only focused on detecting the intrusion in the transmitted data. But, no work is there to avert the network from future possibilities of attacks by identifying the stages of attacked data.
- Employing unwanted computing resources like data processing, memory usage, etc., by attackers causes high memory usage, network congestion, latency, etc.
- It is challenging and time-consuming to distinguish the features from the slightly attacked data.
- Bad packets produced by software errors, defective data, and locally escaping packets can cause a high falsealarm rate.

2.2 Major Contributions

To overcome the limitations in the existing works, the major contributions presented by the proposed work are listed as follows;

- To make the network more complex for the attackers, an SSELUR-GRU model is employed for identifying the stages of attacked data. So, the different stages of attack are classified by the processing of optimal hyperparameters with improved learning efficiency using the introduced model.
- To avoid unauthorized access and save computing resources, dynamic smart card-based protection is done based on the detected stages of attacks.
- A feature calculation method is introduced by extracting the features of input data, which aids in diminishing the processing time of the attack identification.
- An LWSO-centeredFeature Selection (FS) framework is wielded for the selection of the best features and mitigating the false alarm rate of the proposed scheme. This process also reduces the complexity of reaching an optimal solution with the help of the Lehmer approach.

The paper is structured as: Section 2 analyzes the associated work concerning the proposed work. Section 3 displays the proposed framework. Section 4 analyses the proposed system's performance. Finally, Section 5 concludes the paper along with the future work.

3 Related Literature Review

Aouedi et al. [2] presented a federated semi-supervised learning scheme for attack detection in IIoT. For learning the representative as well as low-dimensional features, an Auto Encoder (AE) was trained on each device. The experiential outcomes illustrated lower communication overhead. However, owing to the usage of a large amount of data, there was more possibility to elevate the communication overhead.

Wang et al. [15] introduced a stacked DL methodologyfor identifying cyber-attacks targeting Supervisory Control And Data Acquisition (SCADA) systems. Each densely connected layer in the model was followed by a batch normalization to prevent overfitting. The outcomes signified thesatisfactory detection performance by the layered DL technique. Nevertheless, the lack of SCADA data influenced the effective detection solutions.

Liang et al. [16] recommended an industrial network centered on multi-feature data clustering optimization. Data security coefficients and weighted distances were categorized centered on the data attribute feature's priority threshold. The detection rate was significantly enhanced by this technique. Nevertheless, the framework didn't focus on the behavior features of nodes, whichcaused the misclassification of data attacks.

Idrissi et al. [17] presented a Lightweight Optimized DL-centered Host-IDS deployed on the Edge for the IoT. Primarily, the raw data was pre-processed, and the intrusion was detected using a Convolution Neural Network. As the model was trained by utilizing a low amount of features, the system didn't offer satisfactory accuracy even though the training time for intrusion detection was low.

Vargas et al. [18] offered an integrated system for the detection as well as containment of intruders on the Edge. For classifying the attacks presented in the transmitted data, the K-Nearest Neighbors algorithm was utilized. The way for a positive trend in industrial manufacturing was opened by the findings. However, the approach didn't concentrate on more sophisticated attacks against IoT deployments.

Fatani et al. [19] illustrated an integrated detection systemalong with the containment of intruders on the Edge. The recommended system was split into '2' phases: a CNN-based feature extraction phase and anFS phase centered on the developed Aquila optimizer approach.The developed approach's higher performance was exhibited by the outcomes. However, the real-time feasibility of these models was still unknown.

Altan [20] developed a Deep Belief Network to detect invasion in IIoT sensing systems. A Secure DeepNet-IoT platform with numerous adaptive kernels was employed for sensing inputs. The IoT-centered invasion was recognized more accurately by deep AE with extreme learning machine kernels. However, the training time was increased by increasing the number of parameters in the developed model.

Aleesa et al. [21] employed a deepIDS. Primarily, preprocessing was executed over the raw input data. By using the min-max technique, the preprocessed data were normalized in the second phase. The evaluation findings demonstrated that the DL model produced better predictions. However, an uncertain detection rate was caused by the unequal distribution of classes in the data.

Patel et al. [22] exhibited a smart network IDS. Thedeveloped model's preprocessing stage included feature transformation and normalization. After that, by using an enhanced optimization technique, the preprocessed data features were selected. The experimental findings exhibited thatbetter performance was exhibited by the technique. However, the noisy and poor-quality data was not processed by the model.

Abd et al. [23] offered an enhancedIDS for binary classification. Extreme Learning Machine, Support Vector Machine (SVM), Rao Optimization Algorithm, and Logistic Regression were integrated with supervised Machine Learning (ML)approachesin the developed IDS. The experimental outcomesexhibited higher accuracy of Rao-SVM on the UNSW-NB15 dataset. The system's loss function was high owing to the improper updation of weight values.

Liu et al. [24] developed the anomaly detection model in industrial cyber-physical systems. Here, the model division technique was employed to design the detector. This framework analyzed the operating data in a closed loop and quantified the mode deviation. Further, the detection model classified the DoS, Stuxnet-like, and False Data Injection attacks. The performance of the model was improved regarding accuracy, precision, and detection rate. However, the stages of the attacks were not detected, which lacks the model's efficacy.

3.1 Improvements of the Proposed Methodology

As reviewed in the previous section, while detecting the attacks, the related works had some notable limitations. Some works including [16, 17] mainly influenced by the processing of insufficient or sub-optimal features, resulting in inaccurate classification of data or expanded processing time. Thus, the proposed work selected the optimal features of the data by using the LWSO algorithm, so that the attack is identified by learning only the significant features. This in turn lowers the model's complexity and time consumption. Also, the training time is increased in [20] due to the involvement of huge parameters. Hence, such hyperparameters are optimized with the help of Reinforcement Learning and SSELU activation function in the proposed approach to lower the training time of the classifier. Moreover, the data patterns and future awareness regarding the data attacks were not concentrated in any of the prevailing works, which limited the performance outcome and efficiency [25, 26]. Therefore, in the proposed work, data preprocessing, data balancing, data clustering, optimal features, and data patterns are carefully performed to produce the accurate identification of attacks to protect the dynamic smart card.

4 Proposed Approach For Attack Stage Classification With Intrusion Detection

This research model proposed an efficient classification of attack stages along with intrusion detection that aims to detect abnormal activities of intruders and facilitate preventive measures to avoid risk in IIoT. The proposed scheme'sblock diagram isrepresented in Fig. 1.

4.1 Training Phase

Here, to train the proposed system, the network packet data (P) from the "UNSW_NB15 dataset" is regarded as the input. To carry out the training, the preprocessing, feature calculation, data balancing, data sampling, data standardization, FS, pattern analysis, IDS classification, and stages identification steps are employed.

4.1.1 Data Preprocessing

In general, since the raw packet data contains some missing data and different data formats,dealing with the raw packet data (P) often has challenges.

(a) ***Missing Value Imputation:*** The loss of vital information and the incorrect classification of data may be caused by the missing values in P. Hence, the missing value is imputed with the mean of prevailing data of every single column. After that, the recovered data (P^*) can be expressed as,

$$P^* = \overset{imput(mean)}{P}\left(P_1^*, P_2^*, \ldots\ldots, P_z^*\right) \tag{1}$$

Here,the imputation function is denoted as $\overset{imput(mean)}{P}$ and P_z^* specifies the z^{th} number of data.

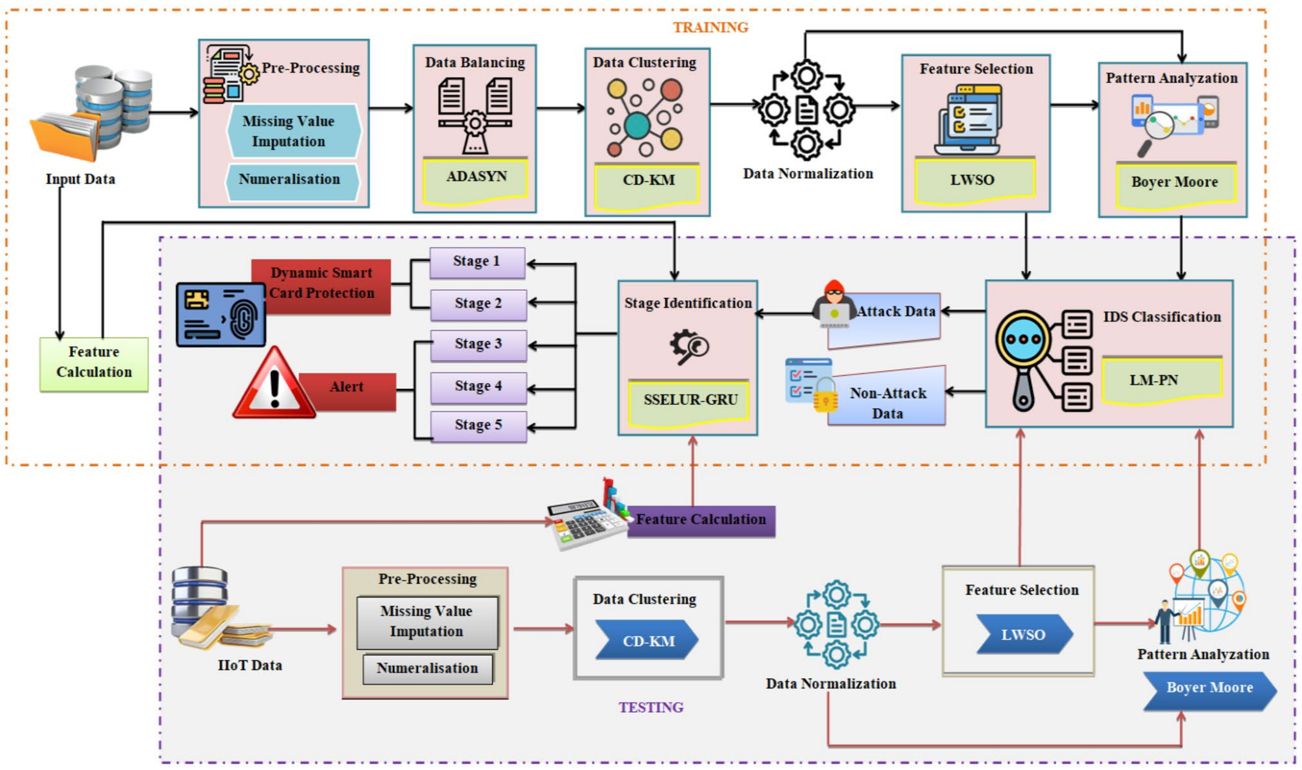

Fig. 1 Block Diagram of Proposed Methodology

(b) ***Numeralization:*** The packet data after imputing missing values (P^*) contains certain string data within it. Therefore, to make an efficient packet data transmission, numeralization ($\aleph$) is done to convert all those strings into numerical values. Also, the resultant numeralized data $(P^*)^{num}$ is,

$$(P^*)^{num} = \aleph\left(\sum_{a=1}^{z} P_a^*\right) \tag{2}$$

Here, P_a^* depicts the ath number of data.Later, the preprocessed data is signifiedas P^{pre}.

4.1.2 Data Balancing

In general, classifiers are biased with the majority of the classes in the data, which leads to poor classification of the minority class. Therefore, the data is balanced, which provides the data with an equal number of classes. ADASYN is used in this proposed paper for balancing the data with minority classes in P^{pre}. Primarily, the ratio of the minority to the majority of the classes $R \in P^{pre}$ is computed as,

$$R = \frac{(A)^{\min}}{(A)^{maj}} \tag{3}$$

Here, the minority and majority of the classes presented in P^{pre} are signified as $(A)^{\min}$ and $(A)^{maj}$. If the computed ratio is lower than the preset threshold for the maximally tolerated degree of class imbalance ratio, then the total number of synthetic minority data $\left(A_{SYN}^{\min}\right)$ is calculated as,

$$A_{SYN}^{\min} = \left((A)^{maj} - (A)^{\min}\right)R \tag{4}$$

Then, the KNNof every single minority data $\left(D_a\right)$ is computed as,

$$D_a = \frac{\left(A_a\right)^{maj}}{K} \tag{5}$$

Here, the number of the nearest neighbor is signified as K, and $\left(A_a\right)^{maj}$ denotes the ath majority sample. Following this, each minority data $\left(D_a\right)$ above obtained is normalized as,

$$D_a^* = \frac{D_a}{\sum D_a} \tag{6}$$

By utilizing these normalized nearest neighbors $\left(D_a^*\right)$, the number of synthetic examples D^{sam} is computed to generate neighborhood data as,

$$D^{sam} = A_{SYN}^{\min} D_a^* \tag{7}$$

where P^{bal} implies the balanced data after inducing the synthetic samples.

4.1.3 Data Clustering

Next, based on the balanced data's protocol type, the $\left(P^{bal}\right)$ is clustered. Sequential data under every protocol could be obtained via clustering that aids in diminishing the complexity during training by feeding input data sequentially. By utilizing CD-KM, the clustering is performed. The conventional K-Means (KM) algorithm is computationally efficient. Therefore, the prevailing algorithm doesn't perform very well in clustering the non-linearly separable data in each protocol. Cayley distance is used rather than Euclidean distance to solve this issue. The distance between the diverse varying data is efficiently identified by CD to avoid the outlier issue and make an accurate cluster.

- Initialize all the data P^{bal} and it is considered as the data points, which are signified as,

  $$P^{bal} = \left\{p^1, p^2, p^3, ...p^N\right\} \; where, b = 1, 2., 3, ...N \tag{8}$$

  Here, the number of datapoints isdisplayed as b and p^N denotes the Nth data point.
- Then, choose the number of centroids randomly C.

  $$C_k = \left\{C_1, C_2, C_3, ...C_\Lambda\right\} \quad k = 1, 2, ...\Lambda \tag{9}$$

  Here,the centroid number is represented as k, and C_Λ notates the Λ^{th} number of the centroid.
- Assign a data point to the closest centroid.
- Calculate the distance (d^{cly}) betwixt the data point and the centroid utilizinga CD. It can be expressed as,

  $$d^{cly} = \rho^{cycle}\left|P^b C_k - 1\right| \tag{10}$$

  Here, the number of permutation cycles is denoted as ρ^{cycle} and P^b denotes the bth data point.
- Choose a cluster for the data point in which the distance betwixt the data point and the centroid is minimal.

The obtained clustered data can be denoted as P^{clus} via this clustering.The proposed CD-KM's pseudo-code is given as,

```
Input: Balanced Data $\left(P^{bal}\right)$
Output: Clustered Data $\left(P^{clus}\right)$
Begin
    Initialize number of clusters, iteration $(iter)$, maximum iteration $(iter_{\max})$
    Perform clustering
    Select the number of centroids
    Set $iter = 1$
    While $iter \le iter_{\max}$
        For each data point, do
            Compute $\Gamma$
                $d^{cly} = \rho^{cycle}\left|P^{b}C_{k} - 1\right|$
        End for
        If ( $P^{b} == undercluster$ ) {
            Stop criteria
        } Else {
            Set $iter = iter + 1$
        } End if
    End While
    Return $P^{clus}$
End
```

4.1.4 Data Normalization

The biased outcome of predictions in classification may be caused by training the IDS model with a different range of data variables. Therefore, by employing min-max normalization, the data variables in the clustered data $\left(P^{clus}\right)$ are normalized within the range of [0, 1]. The normalized data $\left(P^{Nor}\right)$ is,

$$P^{Nor} = \frac{P^{clus} - \left(P^{clus}\right)^{\min}}{\left(P^{clus}\right)^{\max} - \left(P^{clus}\right)^{\min}} \tag{11}$$

Here, the maximum and minimum data variables in data are denoted as $\left(P^{clus}\right)^{\max}$ and $\left(P^{clus}\right)^{\min}$.

4.1.5 Feature Selection

Next, for reducing the computational time and training time of the classifier, the most interesting features are selected from P^{Nor}. By employing LWSO, the FS is done. War Strategy Optimization (WSO) has the capability of achieving a good balance of the exploration and exploitation stages. However, the random updation of a soldier rather than a weak soldier causes a higher computational burden for achieving global optimum value. The Lehmer technique, which significantly diminishes the problem caused by the random updation during exploration, is utilized to solve this issue. Therefore, by this, the number of iterations gets reduced with optimal features.

(a) ***Population Initialization:*** The population of the army troops (F) (Features presented in P^{Nor}) in a g-dimensional area are initialized as,

$$F = \left[F^1, F^2, F^3, ...F^N\right] \; where, i = 1, 2, 3, ...N \tag{12}$$

Here, the N^{th} number of features is notated as F^N, and the feature number is signified as i. Here, the person with the maximum attacking force is declared as the army chief.

(b) ***Evaluate Fitness Function:***Based on attaining the higher accuracy as $\max\limits_{acc}$ of the classifier, the fitness value is calculated for each soldier. The fitness function of each soldier ($\Im(F)$) is computed as,

$$\Im(F) = \max_{acc}\left[F^1, F^2, F^3, ...F^N\right] \tag{13}$$

(c) ***Charge strategy:***The king is accountable for commanding and guiding all other soldiers. Here, the soldier with the maximum fitness value is declared the king. The rank and weight of the soldier are also upgraded as the war progresses, and it is given as,

$$F^i(t+1) = F^i(t) + 2 \times \alpha(L^{com} - K) + \hbar \times \left(\omega^i \times K - F^i(t)\right) \tag{14}$$

Here, the current and previous location of the i^{th} soldierare symbolized as i^{th} and $F^i(t+1)$ and $F^i(t)$ the location of the king and commander location are notated as K and L^{com}, ω^i denotes weight, and random values are specified as α and $\hbar$.

(d) ***Update the rank and weight of the soldiers:*** Each soldier's rank is determined by their record of accomplishments on the battlefield. If the new position $\left(\Im\left(F_{new}\right)\right)$ is lower than the prior objective $\left(\Im\left(F_{old}\right)\right)$, the soldier considers the previous position as,

$$\begin{aligned} F^i(t+1) = {} & \left(F^i(t+1) \times \Im\left(F_{new}\right) \geq \Im\left(F_{old}\right)\right) \\ & + \left(F^i(t)\right) \times \Im\left(F_{new}\right) < \Im\left(F_{old}\right) \end{aligned} \tag{15}$$

If the soldier updates the location successfully, the soldier's rank $\left(\wp^i\right)$ is updated as,

$$\wp^i = \left(\wp^i + 1\right) \times \Im\left(F_{new}\right) \geq \Im\left(F_{old}\right) + \left(\wp^i \times \Im\left(F_{new}\right) < \Im\left(F_{old}\right)\right) \tag{16}$$

(e) ***Protection strategy:*** Here, the king is protected by the soldier from the opponent's battle. Hence, according to the neighborhood soldier and king, every soldier changes their location.

$$F^i(t+1) = F^i(t) + 2 \times \alpha\left(K - F^{rand}(t)\right) + \hbar \times \omega^i \times \left(L^{com} - F^i(t)\right) \tag{17}$$

where, a random neighborhood soldier is signified as $F^{rand}(t)$.

(f) ***Substitution of weak soldiers:*** Here, the weak soldiers are replaced by some soldiers who died in the war to enhance the outcome of the war. The replacement $F^{weak}(t+1)$ is articulated by,

$$F^{weak}(t+1) = l_b + \hbar\left(u_b - l_b\right) \tag{18}$$

Here, the battlefield's lower and upper boundariesare denoted as l_b and u_b. F^{sel} signifies the final outcome of the successful war (Selected Features).

4.1.6 Pattern Analyzation

To enhance the IDS classification's accuracy, the patterns of P^{Nor} are analyzed. The analysis is performed by employing Boyer Moore, which is incredibly quick. Each character of sub-data is compared by this algorithm to find the same characters in the data.When characters don't match, the search jumps to the subsequent matching position in the pattern by the value specified in the Bad Match Table. The analyzed patterns (δ) of normalized data are articulated as,

$$\delta = \left\{\delta^1, \delta^2, \delta^3, ..., \delta^B\right\} \tag{19}$$

where, the B^{th} analyzed pattern of P^{Nor} is denoted as δ^B.

4.1.7 IDS Classification

Then, for classifying the types of attacks, the LM-PN is trained using the inputs of F^{sel} and (δ). Pyramid Net (PN) has the potential to gradually increase the feature map dimensions. Therefore, it often suffers from a vanishing gradient issue, which causes slow convergence. The log mish activation function is used in PN to solve the abovementioned issue. This adjusts the range of convolution output and generates the linear output. Thus, the computational time is diminished abundantly. A number of pyramid layers with skip connections are encompassed in this proposed LM-PN. Each layer consists of convolution and log mish activation functions. Figure 2 signifies the proposed LM-PN's architecture.

Here, the selected features and analyzed patterns are commonly signified as Γ. The number of input features is articulated as,

$$\Gamma = \left[\Gamma^1, \Gamma^2, \Gamma^3, ...\Gamma^W\right] \tag{20}$$

Here, Γ^W depicts the W^{th} number of features.

Convolution Operation At each location of the tensor, an element-wise product betwixt each element of the kernel and the input array is computed and summed to obtain the output value in the corresponding position of the output array. Then, the convolution for Γ (Φ^{con}) is articulated as,

$$\Phi^{con} = \sum_b \sum_b (\Gamma)(q-b, Q-b) * w(b,b) \tag{21}$$

where, $w(b,b)$ represents the kernel in the dimension size $b \times b$, and the input matrix's dimension size is given as q and Q.

Nonlinear Activation Function The proposed network avoids learning the trivial linear combinations of the inputs by employing activation functions. Therefore, the paper employed the log mish activation function.

$$A^{\Phi^{con}} = \Phi^{con} * \log\left(1 + \tanh\left(e^{\Phi^{con}}\right)\right) \tag{22}$$

where, the log mish activation function's output is signified as $A^{\Phi^{con}}$. The convolution operation is performed again after the activation function. The obtained features are

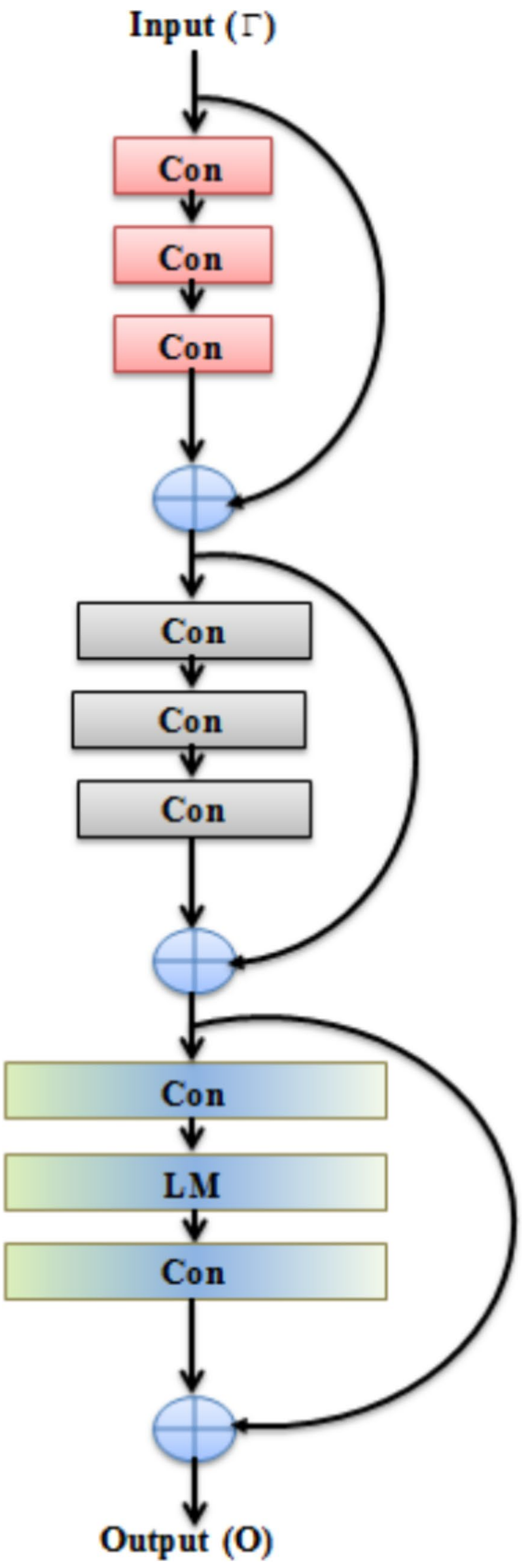

Fig. 2 Architecture of the proposed LM-PN

signified as the first pyramid layer's output. The obtained feature along with the input feature is given into the subsequent pyramid layer. In addition, the same procedure above explained is applied to the entire pyramid layer. The output as of the final pyramid layer is given into the output layer, which decides whether the data is non-attacked or attacked. Finally, by analyzing the target outcome $(O)^{tar}$ with the actual outcome $(O)^{act}$, the loss values of the output are computed. Therefore, the overall loss value is calculated as follows,

$$Loss = \left((O)^{tar} - (O)^{act}\right)^2 \tag{23}$$

If $Loss = 0$, then the model gives an accurate prediction. If $Loss \neq 0$, then the back-propagation occurs by updating the weight values.Lastly, the data is significantly classified as attacked or non-attacked by the classification technique. From this process,O notates the obtained output, where the attacked data and non-attacked data are denoted as O^{att} and $O^{non-att}$.

4.1.8 Feature Calculation

Meanwhile, for gathering more important information about data and improving the accuracy of stage identification operation, the features from the input data packet (P) are extracted.Therefore, the key features like skewness, kurtosis, mean, entropy, etc., are extracted, and it is articulated as,

$$F^{ex} = \left\{F_1^{ex}, F_2^{ex}, F_3^{ex}, ...F_Z^{ex}\right\} \tag{24}$$

here, the Z^{th} feature of input data is denoted as F_Z^{ex}, andthe number of extracted features is symbolized as F^{ex}.Thus, the processing time for detecting attacks is reduced and meets the proposed objective.

4.1.9 Stage Identification

Further, to prevent the data from future attacks, the SSELUR-GRU is trained to utilize O^{att} $_{\text{and}}$ F^{ex} for identifying the stages of the attack.The prevailing GRU has more expressiveness and versatility. Nevertheless, when GRU directly controls the behavior of the training process, it is difficult to select the most suitable hyperparameters and is more time-consuming. To solve these issues, Reinforcement Learning (RL) is used rather than the reset gate of GRU, and Smooth Scaled Exponential Linear Unit (SSELU) activation function is employed.The RL allows the optimization process to make larger updates to the model's parameters in the early stages of training, thereby promoting faster learning and convergence in the course of the initial exploration phase. And, utilizing SSELU assists in activating appropriate neurons. Thus, the proposed model's time consumption is diminished.The architecture of the proposed SSELUR-GRU is Fig. 3,

Initially, the regularized data of O^{att} and $F^{ex}\left((X)^t\right)$ are fed as the input into the input layer. Later, the data update is assessed at the t^{th} time and ϕ^{th} task $\left(u_\phi^t\right)^*$ over a regularized layer and is computed as,

$$\left(u_\phi^t\right)^* = \sigma\left(w(X)^t + U^t\left(\vartheta^t \otimes (X)_{\phi-1}^t\right)\right)^\phi \tag{25}$$

where, the undetermined parameter and vector at the time t is notated as w and U^t, $(X)_{\varphi-1}^t$ signifies the prior regularized data, which is stored in the memory at the time t and ϕ^{th} task, the SSELU activation function is denoted as σ, the element-wise multiplication operation is represented as $\otimes$, $(X)^t$ denotes regularized data at the time t and ϕ^{th} task, and the hyperparameter obtained from RL is denoted as ϑ^t, which controls how much the unit updates from all units.

Reinforcement Learning (RL) Based Parameter Selection RL is a subfield of MLin which an agent interacts with an environment to learn by trial and error. The agent takes actions in the environment and receives feedback in the form of rewards, which signifythe desirability of its actions. The hyperparameter is derived in this proposed work by using these characteristics.

Step 1: Define the Environment:
The updated data $\left(u_\phi^t\right)^*$ is inputted to the RL, and it is regarded as the agent. Then, the environment (τ^E) around the agent is defined by states (τ^s), actions (τ^a),

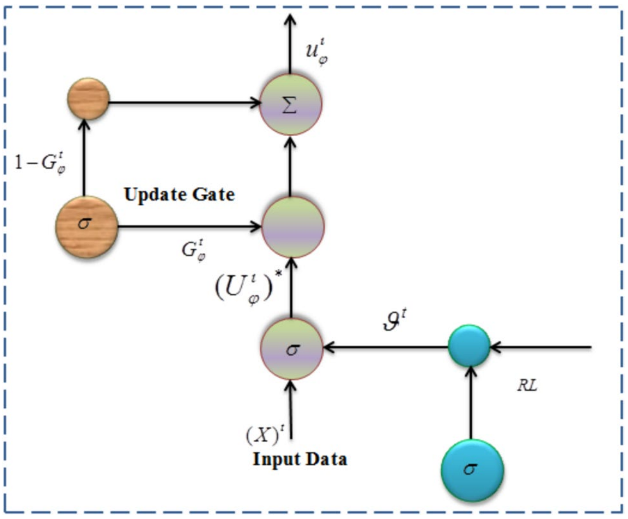

Fig. 3 Structure of proposed SSELUR-GRU

and rewards (τ^r). The current situation of the agent $\left(\left(u^t_\phi\right)^*\right)$ is represented by the state (τ^s). The decision made by the agent is represented by the action (τ^a). Based on the agent's action, the reward (τ^r) indicates the feedback from the environment.

$$\tau^E \leftarrow \{\tau^s, \tau^a, \tau^r\} \tag{26}$$

Step 2: Select the best agent
Here, the best and most efficient agents are chosen from $\left(u^t_\phi\right)^*$ to learn effective decision-making policies in complex as well as dynamic environments. Then, the optimized agent is notated as U^{opt}
Step 3: Initialize the Q-Network:
The Q-Network is a neural network that approximates the Q-values for State-Action Pairs (SAPs), which is symbolized as $\zeta(\tau^s, \tau^a)$. A state is taken as input by the Q-Network, which outputs Q-values for every single possible action.

$$\zeta(\tau^s, \tau^a) = \zeta_{Net}(\tau^s, \tau^a) \tag{27}$$

where, the Q-Network's output is depicted as $\zeta_{Net}(\tau^s, \tau^a)$.
Step 4: Exploration-Exploitation Trade-off:
The agent balances the exploration (trying out new actions) as well as exploitation (selecting actions centered on learned knowledge) and chooses the action with the highest Q-value, which is notated as,

$$\chi(\tau^a|\tau^s) = (1-\varepsilon) * Arg\max(\zeta(\tau^s, \tau^a)) + \varepsilon * \imath(\tau^a) \tag{28}$$

where, the likelihood of choosing an action (τ^a) in the state (τ^s) is represented as χ, the Q-value for SAP (τ^s, τ^a) is signified as $\zeta(\tau^s, \tau^a)$, the exploration rate is displayed as ε, and $\imath(\tau^a)$ refers to the uniform distribution over all possible actions.
Step 5: Agent-Environment Interaction:
By taking actions centered on the current state, the agent interacts with the environment. It receives a reward and transitions to the subsequent state. The state, action, reward, and next states ($(\tau^s)^*$) are stored in the replay memory (M_R) after each agent-environment interaction.
Step 6: Q-Learning Update:
Here, by using the Q-Learning equation, the Q-values are updated grounded on the agent's experiences, which are defined by,

$$\begin{aligned}\zeta(\tau^s, \tau^a) = \zeta(\tau^s, \tau^a) + B * (\ell(\tau^r) \\ + \varphi * Max(\zeta((\tau^s)^*, (\tau^a)^*)) - \zeta(\tau^{s^*}, \tau^a))\end{aligned} \tag{29}$$

where, the current Q-value for the SAP (τ^s, τ^a) issymbolized as $\zeta(\tau^s, \tau^a)$, the immediate reward received after taking action (τ^a) in the state (τ^s) isrepresented as $\ell(\tau^r)$, φ defines the discount factor that balances the significance of future rewards when analogized to immediate rewards, the maximum Q-value over all possible actions $\left((\tau^a)^*\right)$ in the next state $\left((\tau^s)^*\right)$ is notated as $Max(\zeta((\tau^s)^*, (\tau^a)^*))$, and B is the exponential learning rate that determines the influence of new information on the current Q-value. The highest action value with the agent is chosen as the hyperparameter from the updated Q-value, and it is symbolized as ϑ^t.

The SSELU nonlinear activation function executes the nonlinear transformation to the input data, thus making it capable to learn and execute more complicated tasks, and it is computed as,

$$\sigma = \lambda \begin{cases} \alpha\left(e^X - 1\right), for X < 0 \\ X, \qquad for X \geq 0 \end{cases} \tag{30}$$

where, the predefined SSELU parameters are denoted as a $_{\text{and}}$ λ. After that, the current candidate update is:

$$u^t_\phi = \left(1 - G^t_\phi\right)\left(u^t_{\phi-1}\right) + G^t_\phi\left(u^t_\phi\right)^* \tag{31}$$

here, the candidate update of the previous state is depicted as $u^t_{\phi-1}$, and the update gate that decides how much unit is updated from its activation is displayed as G^t_ϕ, and it is calculated as,

$$G^t_\phi = \sigma\left(w(X)^t + U^t(X)^t_{\phi-1}\right)^* \tag{32}$$

The loss function (γ) is estimated for determining the performance of training, and it is articulated as,

$$\gamma = \ell^{tar} - u^t_z \tag{33}$$

here, the targeted output is signified as ℓ^{tar}. If there is a higher loss function, then the maximum iteration is executed. Finally, the classification technique significantly classifies the stages of attacks as stage 1, stage 2, stage 3, and stage 4. The final classified attack stages are depicted as S $_,$ thus addressing the main objective of the proposed work. The proposed SSELUR-GRU's pseudocode is,

```
Input: Calculated Features $(F^{ex})$ and Attacked data $(O^{att})$
Output: Stages of Attacks $(S)$

Begin
        Initialize inputs $(X)^t$ ,          ,
        For all data, do
                Feed all the data
                While $\varphi^{th}$ features
                        Evaluate update gate {
                                $(u_\varphi^t)^* = \sigma\left(w(X)^t + U^t\left(\vartheta^t \otimes (X)_{\varphi-1}^t\right)\right)^\varphi$
                        }
                        Estimate RL
                        Activate adaptive glumber activation {
                                $r^t = \sigma\left(w(Y^{reg})^t + U^t(Y^{reg})_{z-1}^t\right)^z$
                        }
                        Evaluate update gate
                        Compute candidate update
                End while
        End for
        Summing all the candidate update
        Return $S$
End
```

4.2 Testing Phase

The real-time data implementation is executed here by handling some steps, which are explained briefly here.Here, the input data packet is preprocessed, where the missing value imputation and nominalization are performed. After that, by utilizing CD-KM, the preprocessed data is clustered centered on its protocol and then normalized to a certain range. Later, by employing LWSO, the most significant features are chosen from normalized data. In the meantime, the patterns of normalized data are analyzed. Then, for testing the real-time data whether it is attacked $\left(O_{real}^{att}\right)$ or non-attacked $\left(O_{real}^{non-att}\right)$, the selected features and analyzed patterns are fed into the trained LM-PN. The features from the input data packet are computed simultaneously. Lastly, the classified attacked data and calculated features are further carried into the stage identification step, which classifies the stages of attack in data. Further steps of smartcard protection (Ψ) are executed by meeting the following condition, which is expressed as,

$$\Psi = \begin{cases} if\ S = 1, 2, & perform\ \varsigma \\ if\ S = 3, 4, & Sent\ alert \end{cases} \tag{34}$$

If the attacks in the transmitted data are in stages 1 and 2, then the safety precautions are taken by performing dynamic smart card protection. Or else, the alert message is sent to the corresponding admin.

4.3 Dynamic Smart Card Protection

Here, the smart card is dynamically updated when the stage of attacked data is detected as stages 1 and 2. Generally, for the device in the IIoT environment, the smart card is allotted.The smartcard is any chip or integrated circuit card, which holds the user's personal information, account details or device' sensitive data.The data can be transmitted in the IIoT environment via authentication using a smart card. The tampered smart card performs all the malicious activities formed by the attacker in the data. Therefore, the updation of smartcards is prominent. It assists in preventing the data from the attacks in the future. The device information like IP address $\left(\eta^{IP}\right)$, MAC address $\left(\eta^{MAC}\right)$, and serial numbers (η^{sn}) are considered as the input $\left(\varsigma^{input}\right)$ for the smart card updation. They are initialized as,

$$\varsigma^{input} \leftarrow \left(\eta^{IP}, \eta^{MAC}, \eta^{sn}\right) \tag{35}$$

For updating the smartcard, the Frame Checksum Sequence (FCS) is created for the initialized input. Therefore, the initialized number format is split into '2' data formats. Later, FCS bits $\left(\Lambda^{FCS}\right)$ (random 16 or 32 bits) are added in between the divided data. The data after adding FCS is signified as,

$$\varsigma = \left(\eta^{IP}\eta^{MAC}\right)\Lambda^{FCS}\eta^{sn} \tag{36}$$

Here, the updated smartcard is denoted as ς. The obtained number employing FCS is taken as the updated smartcard number. In the meantime, the updated smartcard is distributed to the admin. Thus, only the authenticated admin can be executed in the IIoT environment by the updation of the smartcard. Therefore, the attacks could be diminished abundantly and addressed the research objective.

5 Results and Discussion

The proposed model's performance is analyzed in this section by comparing its results with other prevailing models. In the working platform of PYTHON, the proposed model is implemented.

5.1 Dataset Description

The proposed work utilized the UNSW-NB15 dataset'sraw network packets for the performance analysis. The dataset includes the combination of real modern normal activities as well as synthetic contemporary attack behaviors. This dataset has '9'sorts of attacks: Exploits, Fuzzers, Generic, Analysis, Reconnaissance, Backdoors, Shellcode, Worms, and Denial of Service (DoS). It contains 175,341 and 82,332 records for the training set as well as the testing set, respectively, from attacked and normal data.

5.2 Performance Analysis of Stage Identification

The proposed SSELUR-GRU's performance is analyzed and analogized with the prevailing models like GRU, LSTM, RNN, and CNN to prove the proposed scheme's superiority.

Regarding the identification rate, the proposed and prevailing models'performance evaluation is presented in Fig. 4. The proposed SSELUR-GRU gains an identification rate of 98.21%. However, when analogized to the proposed model, the existing models like GRU (95.23%), LSTM (91.9%), RNN (90.4%), and CNN (89.3%) attained a lower rate. As the complexity due to hyperparameters during data training was not resolved in existing works, the identification rate is sub-optimal.The developed model's higher performance is attained by the optimal selection of a hyperparameter utilizing RL and also the inducement of SSELU, which diminishes the developed model's complexity and increases the identification rate.

Concerning Mean Square Error (MSE), Mean Absolute Percentage Error (MAPE), and Root Mean Square Error

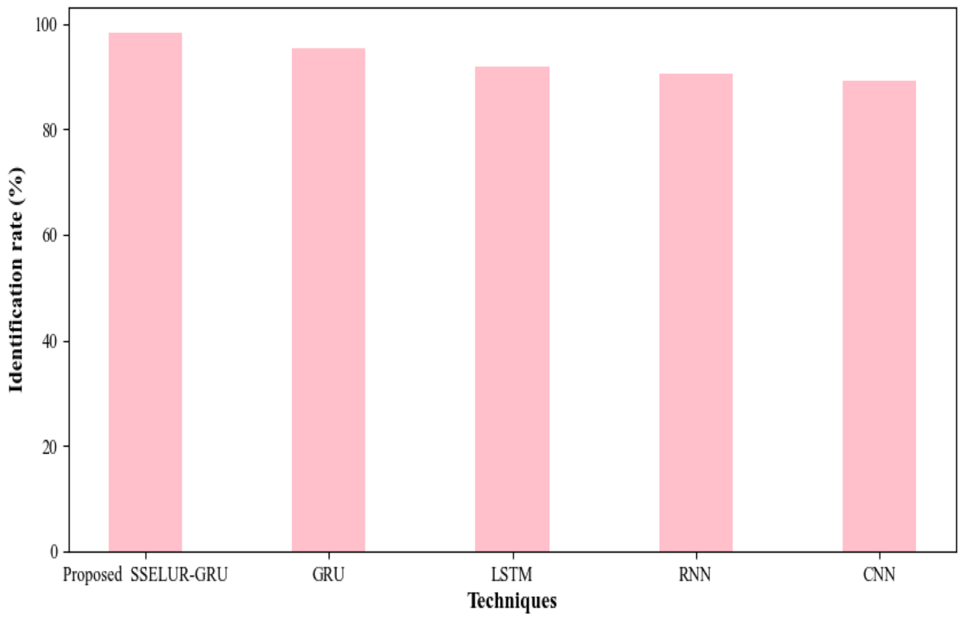

Fig. 4 Graphical representation of proposed SSELUR-GRU and existing models in terms of identification rate

(RMSE), the performance evaluation of proposed and prevailing models is depicted in Table 1. Here, the proposed SSELR-GRU attained MSE, MAPE, and RMSE of 0.0423, 0.21343, and 0.20566, correspondingly. However, lower performance was attained by the conventional models due to a lack of processing optimal parameters. The proposed scheme's higher performance is attained by the neurons that remain active for a maximum number of iterations by enhancing the prevailing GRU, thus learning the inputs significantly.Therefore, in the identification of stages, the proposed SSELUR-GRU was more robust with minimum error.

5.3 Performance Analysis of IDS Detection

Regarding accuracy, precision, recall, sensitivity, specificity, F-measure, False Positive Rate (FPR), False Negative Rate (FNR), Negative Predictive Value (NPV), and Positive Predictive Value (PPV), the proposed LM-PN's performance analysis is validated and analogized with PN, Visual Geometry Group 16 (VGG16), EfficientNet, and CNN.

Table 1 Performance evaluation of the proposed SSELR-GRU and existing models

Techniques	MSE	MAPE	RMSE
Proposed SSELR-GRU	0.0423	0.21343	0.20566
GRU	0.11433	0.7453	0.33812
LSTM	0.7134	1.08656	0.84463
RNN	1.4246	1.17565	1.193566
CNN	8.86443	2.1465	2.977319

The accuracy, precision, along with recall rate of the proposed LM-PN and the prevailingapproaches like PN, VGG16, EfficientNet, and CNN are compared in Fig. 5(a). The proposed LM-PN is modified with the efficient activation function, whichprevents the neurons from dying and the convergence of the network is improved. Therefore, 98.71% of accuracy, 98.01% of precision, and 98% of recall are attained by the proposed LM-PN, whereas the prevailing techniques obtain accuracy, precision, and recall at an average of 91.59%, 91.70%, and 91.28%, respectively. As the vanishing gradient issue that affects the learning process was not rectified by the prevalent methods, the attack detection performance is lower than the proposed technique. Figure 5(b) signifies that the proposed LM-PN achieves higher performance with a sensitivity of 98%, specificity of 97.93%, and F-measure of 98.13%.

Table 2 exhibits that the proposed model's FNR is enhanced by 62% more than PN, 79% more than VGG16, and 90% more than CNN. Similarly, the proposed model's PPV is improved by 42% and 60% than PN and VGG16, respectively.Likewise, the proposed model's FPR and NPV aredrastically augmented than the prevailing schemes. 37245.13ms is the proposed scheme'straining time, which is attained by employing an efficient activation function. The proposed method processed the data patterns and optimal features for efficient attack detection, which was not focused on in the traditional methods. Therefore, the data is classified efficiently by the proposed LM-PN over the existing methods.

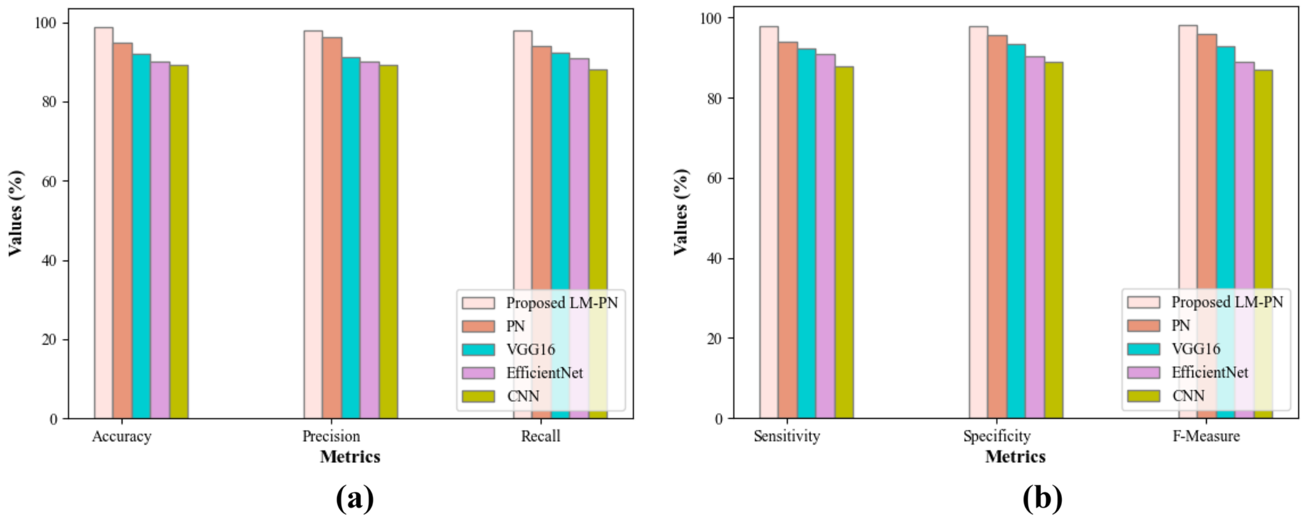

Fig. 5 Graphical representation of the proposed LM-PN based on **a** Accuracy, Precision, andRecall **b** Sensitivity, specificity, and F-measure

Table 2 Performance evaluation of proposed LM-PNand existing models

Techniques	FNR	PPV	FPR	NPV	Training Time(ms)
proposed LM-PN	0.0158	0.98	2.3767	98.023	37245.13
Pyramid Net	0.0418	0.948	5.379	95.356	44710.54
VGG16	0.0753	0.921	9.65	93.087	472154.5
EfficientNet	0.1624	0.886	11.64	90.875	51745.22
CNN	0.1908	0.8787	12.0896	89	59314.14

5.4 Performance Analysis of Feature Selection

Regarding fitness Vs iteration, the proposed LWSO's performance is analyzed, and its outcomes are analogized with the WSO, Particle Swarm Optimization (PSO), Harris Hawks Optimizer (HHO), and Cockroach Swarm Optimization (CSO).

The optimization ability of the proposed LWSO and existing algorithms is compared in Fig. 6. By employing the Lehmer technique, the proposed LWSO improved the position updation process. Therefore, the proposed LWSO renders the optimum outcome with a minimum number of iterations, and it converges at the iteration of 50; nevertheless, the conventional techniques require more iterations due to the random position updation during the exploration phase for attainingan optimal solution.

5.5 Performance Analysis of Data Clustering

CD-KM is analyzed and compared with the existing models like KM, Clustering Large Applications (CLARA), K-medoid, and Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) to prove the superiority of the proposed method's performance.

The proposed and the existing models' CT and CE are portrayed in Table 3. It is known from the comparison analysis that the proposed methodology remains more accurate and faster for clustering when analogized with the prevailing research methodologies. The varying data are identified efficiently in the proposed scheme. Therefore, the iteration for clustering gets minimal with very accurate clustering outcomes. The proposed methodology attains better efficiency (97.84%) and a clustering time of 22547ms. Meanwhile, the existing algorithms could not cluster the non-linearly distinct data, which lowers their clustering efficacy.Thus, the prevailing models do not provide higher performance than the proposed algorithm.

5.6 Comparative Analysis With Literature Papers

In this, the proposed scheme'scomparative analysis is performed with the prevailing works of [2, 15, 20], and [19] centered on their classification accuracy.

The proposed and the prevailing research methodologies' efficiency are compared in Fig. 7. Optimized algorithms like LWSO for FS and CD-KM for data clustering are utilized by the proposed method. These algorithms enhance efficiency by mitigating convergence time, augmenting

Fig. 6 Convergence analysis

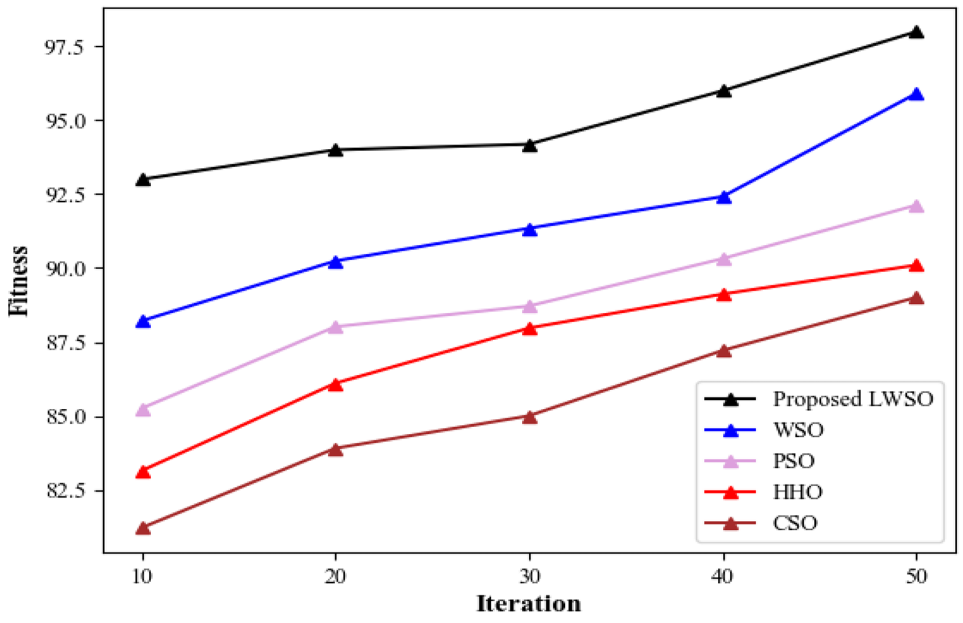

Table 3 Clustering analysis

Techniques	Clustering Time (CT) (ms)	Clustering Efficiency (CE) (%)
Proposed CD-KM	22547	97.84
KM	28756	94.16
CLARA	32478	93.53
K-medoid	37457	90.27
BIRCH	43258	89.26

exploration capability, and considering the relevant relationships between data features. Hence, a higher accuracy was attained by the proposed methodology. Therefore, the proposed scheme is superior to the prevailing works.

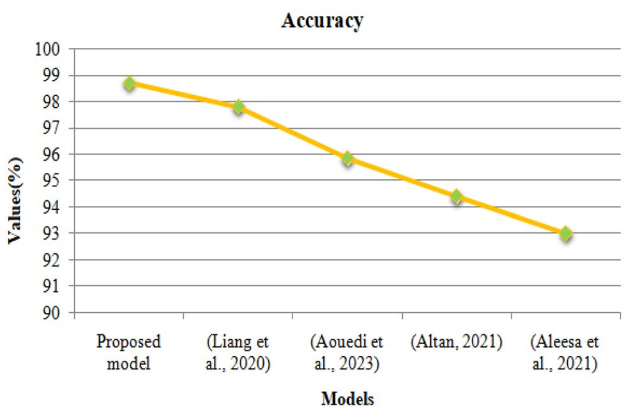

Fig. 7 Comparative analysis of the proposed system with existing works

6 Conclusion

An efficient stage identification and smartcard protection using SSELUR-GRU have been proposed in this work. The pre-processing and data balancing were executed. After this, the data were clustered within the time of 22547ms. After that, the data normalization was performed, and the features were chosen with low iterations. In the meantime, the patterns from normalized data were analyzed. Both selected features and analyzed patterns were fed into the trained LM-PN model. This model classified the data with an accuracy of 98.71%, precision of 98.01%, recall of 98%, along with a training time of 37245.13ms. Lastly, the attacked data and calculated features were given to the trained SSELUR-GRU, which identified the attack's stages with an identification rate of 98.21%. Therefore, it can be concluded from the overall analysis that the proposed model is highly efficient. The work concentrated only on detecting intrusions and preventing the data from future attacks. To improve the success rate of data transmission, the work will be extended in the future with some advanced techniques.

Acknowledgement We thank the anonymous referees for their useful suggestions.

Author's Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by [1]DrMouleeswaranSK ,[2] Dr. K. Ramesh, [3]DrManikandan K, [*4]DrVivekYoganand. The first draft of the manuscript was written by [*4]DrVivekYoganand and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding This work has no funding resource.

Availability of Data and Materials Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

Declarations

Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

Consent of Publication Not applicable.

Conflict of Interest The authors declare that they have no conflict of interest.

References

1. Ruzafa-Alcazar P, Fernandez-Saura P, Marmol-Campos E, Gonzalez-Vidal A, Hernandez-Ramos JL, Bernal-Bernabe J, Skarmeta AF (2023) Intrusion Detection Based on Privacy-Preserving Federated Learning for the Industrial IoT. IEEE Trans Industr Inf 19(2):1145–1154. https://doi.org/10.1109/TII.2021.3126728
2. Aouedi O, Piamrat K, Muller G, Singh K (2023) Federated Semisupervised Learning for Attack Detection in Industrial Internet of Things. IEEE Trans Industr Inf 19(1):286–295. https://doi.org/10.1109/TII.2022.3156642
3. Altuna HC, Albayrak Z (2023) Eng Sci Technol Int J 38:101322. https://doi.org/10.1016/j.jestch.2022.101322
4. Abdel-Basset M, Chang V, Hawash H, Chakrabortty RK, Ryan M (2021) Deep-IFS: Intrusion detection approach for industrial internet of things traffic in fog environment. IEEE Trans Industr Inf 17(11):7704–7715. https://doi.org/10.1109/TII.2020.3025755
5. Essop I, Ribeiro JC, Papaioannou M, Rodriguez J, Zachos G, Mantas G (2021) Generating datasets for anomaly-based intrusion detection systems in iot and industrial iot networks. Sensors 21(4):1–31. https://doi.org/10.3390/s21041528
6. Tharewal S, Ashfaque MW, Banu SS, Uma P, Hassen SM, Shabaz M (2022) Intrusion detection system for industrial internet of things based on deep reinforcement learning. Wirel Commun Mob Comput 2022:1–8. https://doi.org/10.1155/2022/9023719
7. Latif S, Idrees Z, Zou Z, Ahmad J (2020) Drann: A deep random neural network model for intrusion detection in industrial iot. International Conference on UK-China Emerging Technologies UCET 2020:1–4. https://doi.org/10.1109/UCET51115.2020.9205361
8. Liang W, Hu Y, Zhou X, Pan Y, Wang KIK (2022) variational few-shot learning for microservice-oriented intrusion detection in distributed industrial IoT. IEEE Trans Industr Inf 18(8):5087–5095. https://doi.org/10.1109/TII.2021.3116085
9. Arshad J, Azad MA, Abdeltaif MM, Salah K (2020) An intrusion detection framework for energy constrained IoT devices. Mech Syst Signal Process 136:1–12. https://doi.org/10.1016/j.ymssp.2019.106436
10. Adnan A, Muhammed A, Ghani AAA, Abdullah A, Hakim F (2021) An intrusion detection system for the internet of things based on machine learning: Review and challenges. Symmetry 13(6):1–13. https://doi.org/10.3390/sym13061011
11. Kasongo SM (2021) An advanced intrusion detection system for IIoT Based on GA and tree based algorithms. IEEE Access 9:113199–113212. https://doi.org/10.1109/ACCESS.2021.3104113
12. Khraisat A, Alazab A (2021) A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. Cybersecurity 4(1):1–27. https://doi.org/10.1186/s42400-021-00077-7
13. Liu J, Yang D, Lian M, Li M (2021) Research on intrusion detection based on particle swarm optimization in IoT. IEEE Access 9:38254–38268. https://doi.org/10.1109/ACCESS.2021.3063671
14. Le TTH, Oktian YE, Kim H (2022) XGBoost for imbalanced multiclass classification-based industrial internet of things intrusion detection systems. Sustainability (Switzerland) 14(14):1–21. https://doi.org/10.3390/su14148707
15. Wang W, Harrou F, Bouyeddou B, Senouci SM, Sun Y (2022) A stacked deep learning approach to cyber-attacks detection in industrial systems: application to power system and gas pipeline systems. Clust Comput 25(1):561–578. https://doi.org/10.1007/s10586-021-03426-w
16. Liang W, Li KC, Long J, Kui X, Zomaya AY (2020) An industrial network intrusion detection algorithm based on multifeature data clustering optimization model. IEEE Trans Industr Inf 16(3):2063–2071. https://doi.org/10.1109/TII.2019.2946791
17. Idrissi I, Azizi M, Moussaoui O (2022) A lightweight optimized deep learning-based host-intrusion detection system deployed on the edge for IoT. Int J Comput Digit Syst 11(1):209–216. https://doi.org/10.12785/ijcds/110117
18. Vargas H, Lozano-GarzonC MGA, Donoso Y (2021) Detection of security attacks in industrial iot networks: A blockchain and machine learning approach. Electronics (Switzerland) 10(21):1–18. https://doi.org/10.3390/electronics10212662
19. Fatani A, Dahou A, Al-Qaness MA, Lu S, Elaziz MA (2021) Advanced feature extraction and selection approach using deep learning and Aquila optimizer for IoT intrusion detection system. Sensors 22(1):1–20. https://doi.org/10.3390/s22010140
20. Altan G (2021) SecureDeepNet-IoT: A deep learning application for invasion detection in industrial Internet of Things sensing systems. Trans Emerg Telecommun Technol 32(4):1–13. https://doi.org/10.1002/ett.4228
21. Aleesa AM, Younis M, Mohammed AA, Sahar NM (2021) Deep-intrusion detection system with enhanced UNSW-NB15 dataset based on deep learning techniques. J Eng Sci Technol 16(1):711–727
22. Patel P, Gunja H, Ebrahim D (2022) Smart network intrusion detection system for cyber security of industrial IoT. TechRxiv. Preprin 1–16. https://doi.org/10.36227/techrxiv.21431889.v1
23. Abd SN, Alsajri M, Ibraheem HR (2020) Rao-SVM machine learning algorithm for intrusion detection system. Iraqi J Comput Sci Math 1(1):23–27. https://doi.org/10.52866/ijcsm.2019.01.01.004
24. Liu B, Chen J, Yong Hu (2022) Mode division-based anomaly detection against integrity and availability attacks in industrial cyber-physical systems. Comput Ind 137:103609
25. Awotunde JB, Chakraborty C, Adeniyi AE (2021) Intrusion detection in industrial internet of things network-based on deep learning model with rule-based feature selection. Wirel Commun Mob Com 1–17. https://doi.org/10.1155/2021/7154587
26. Lundberg H, Mowla NI, Abedin SF, Thar K, Mahmood A, Gidlund M, Raza S (2022) Experimental Analysis of Trustworthy In-Vehicle Intrusion Detection System Using eXplainable Artificial Intelligence (XAI). IEEE Access 10(August):102831–102841. https://doi.org/10.1109/ACCESS.2022.3208573

S. K. Mouleeswaran received his Ph.D. Degree from Faculty of Information and Communication Engineering, Anna University, Chennai, India, in 2017. He is currently an Associate Professor at Dayananda Sagar University, Bengaluru, India. His current research interest span Cloud Computing, Cyber Security, IoT, Software Engineering, Big Data analytic, Data Mining, and Autonomous Driving Assistance System.

K. Ramesh was born in Odugathur, Tamilnadu, India on 10th June 1966. He received his Bachelors of Electrical and Electronics Engineering from Anna University in the year 2007. Masters Degree in VLSI Design from Sathyabama University in the year 2011 and Ph. D from St. Peter's University, Chennai, in the year of 2017. Currently, he serves as a Professor at Department of Electronics and Communication Engineering, PSV College of Engineering and Technology, India. His current research focused on wireless sensor networks and his interested areas are Low-Power VLSI Design, WSN based Electrical Apparatus and Nano Electronics. He has ten research publications in National/International Journals and Conferences to his credit.

K. Manikandan earned his Ph.D. degree in Wireless Networks from VIT, Vellore, India, in 2015. Currently, he serves as a Professor at School of Computer science and Engineering, VIT, Vellore. His research interests IoT, Wireless Networks, Data science, Network Security and AI.

Vivek Yoganand Anbalagan earned his Ph.D. degree in Information Communication Engineering from Anna University, Chennai, India, in 2019. Currently, he serves as a Senior Technical Support Engineer and Open Shift Security Specialist at Red Hat India Pvt Limited. His research interests span Cloud Security, Cybersecurity, IoT, Video Analytics, Blockchain, Software Engineering, and Data Science.