CESSO-HCRNN: A Hybrid CRNN With Chaotic Enriched SSO-based Improved Information Gain to Detect Zero-Day Attacks

Dharani Kanta Roy
Department of Computer Science and Engineering,
Central Institute of Technology Kokrajhar
Kokrajhar, India

Ripon Patgiri
Department of Computer Science and Engineering
National Institute of Technology Silchar
Silchar, India

Abstract—**Hackers use the vulnerability before programmers have a chance to fix it, which is known as a zero-day attack. Zero-day attackers have a variety of abilities, including the ability to alter files, control machines, steal data, and install malware or adware. When a series of complex assaults uses one or more zero-day exploits, the result is a zero-day attack path. Timely assessment of zero-day threats might be enabled by early detection of zero-day attack pathways. To detect this zero-day attack, this paper introduced a Chaotic Enriched Salp Swarm Optimization (CESSO) with the help of a hybrid Convolutional Recursive Neural Network (HCRNN) is implemented. The input data is retrieved from two datasets called IDS 2018 Intrusion CSVs (CSE-CIC-IDS2018) and NSL-KDD. The data is pre-processed with the help of data cleaning and normalization. A unique hybrid feature selection method that is based on the CESSO and Information Gain(IG) is introduced. The CESSO is also used to improve the Recursive Neural Network (RNN) performance to produce an optimized RNN. The selected features are classified, and prediction is performed using the hybrid Convolutional Neural Network (CNN) with RNN called HCRNN. The implementation of the zero-day attack is performed using MATLAB software. The accuracy achieved for dataset 1 is 98.36%, and for dataset 2 is 97.14%.**

Keywords—Hackers; vulnerability; zero-day attack; chaotic enriched salp swarm optimization; data cleaning; normalization; and MATLAB software.

I. Introduction

An online assault that targets a software vulnerability that neither the programmed developer nor antivirus vendors are aware of is known as a zero-day (0-day) exploit. Before anybody else interested in resolving the issue can identify the software vulnerability, the attacker does, writes an exploit rapidly and then utilizes it to launch an attack [1], [2]. Particularly in today's environment, a zero-day vulnerability poses a major and possible threat to many enterprises. When a software flaw is a zero-day vulnerability, it goes beyond the expected immediate or almost daily instantaneous detection and remains unnoticed for a considerable amount of time [3]. Before the manufacturer even notices the issue or rushes to patch it, this security flaw is frequently exploited by hackers [4], [5]. This exploit is, hence, often referred to as a zero-day assault. Zero-day attacks might evade detection by traditional defenses for a considerable amount of time since network managers' primary focus is to stop assaults before they happen [6]. This makes dangerous agents more likely to penetrate the networks and makes the administrator's job more difficult. Preventing zero-day attacks is one of the most challenging aspects of risk management.

Risk management requires knowledge of the potential threats to be faced as well as the strength of the attack surface [7]. Zero-day attacks make use of vulnerabilities that have already been identified but of which the vendor or developer of the application is ignorant. The vulnerability's recent discovery may take weeks for identification and patching, leaving the program vulnerable to exploits. All kinds of intrusion detection systems are thought to face their greatest threat from zero-day (unknown) assaults [8], [9]. Various studies have dealt with the challenge of identifying unidentified assaults. Applying unsupervised anomaly detection algorithms is one way to find new attack kinds. Despite how difficult this issue is, resolving it would significantly improve the security of our computer system [10]. When additional safeguards are put in place to identify zero-day attacks, the difficulties listed below must be addressed. Not all zero-day vulnerabilities result in zero-day attacks [11].

The players without adequate security measures to thwart all potential attacks lose and are taken advantage of [12]. These weaknesses in security mechanism implementation by an organization or a person allow hackers an exploitation window that might result in zero-day attacks. Zero-day vulnerabilities can be challenging to identify since they can manifest in a variety of ways, including a lack of data encryption, a lack of authorizations, a weakness in an algorithm, a vulnerability in the password security system, etc. [13], [14]. Due to the nature of these security flaws, detailed information on zero-day exploits may only be accessed after the exploit has been discovered [15]. An enterprise may notice suspicious scanning activity or unexpected traffic originating from a customer or service due to a zero-day vulnerability.

The foremost contribution of the paper is as follows-

- The filter technique IG is used before initializing the population in the SSO model. Then, the CESSO approach is used to choose the optimum feature subset.
- Salp Swarm Optimization is improved by incorporating the Chimp Optimization algorithm's strategy,

i.e., its Chaotic Map, named Chaotic Enriched SSO (CESSO).

- This CESSO model is used to improve the performance of the RNN to produce an optimized RNN.
- The optimized RNN is a hybrid with the CNN called HCRNN, which will enhance the prediction accuracy of the zero-day attack.

The organization of the paper is as follows: Section II explains the literature review of the paper, Section III describes the problem statement, Section IV is the detailed description of the proposed methodology, Section V discusses the result and discussion, and finally, Section VI concludes the paper with a detailed conclusion.

II. Literature Review

In this section, the recent papers related to the zero-day attack are discussed, and their drawbacks are also discussed.

In 2016, Zhang et al. [16] suggested a security metric for assessing network resilience to Zero-Day Attacks. By creating and assessing several diversity measures, this research takes the first step toward formally characterizing network diversity as a security parameter.

In 2018, Sun et al. [17] proposed the probabilistic identification of zero-day attack paths, and Bayesian networks were used. This research suggested a probabilistic method and developed a prototype system called ZePro for identifying zero-day attack paths.

In 2019, Afek et al. [18] suggested the extraction of zero-day signatures for high-volume attacks. The key contributions of this research are the method created to extract the necessary signatures, together with the concept of the string-heavy hitters' issue and the technique for solving it.

In 2021, Zoppi et al. [19] presented the strategy and application of unsupervised algorithms for detecting zero-day attacks. The article uses a question-and-answer format to highlight common problems encountered when performing quantitative studies for zero-day detection, and it demonstrates how to build up and test unsupervised algorithms using the proper equipment.

In 2022, Mbona and Eloff [20] recommended machine learning approaches for the semi-supervised detection of zero-day intrusion attacks. The method suggested in this paper shows that Benford's rule, the law of anomalous numbers, is a workable technique that may successfully discover major network aspects that are suggestive of abnormal behavior and can be utilized for identifying zero-day assaults.

In 2022, Popoola et al. [21] examined federated deep learning for IoT-edge devices to detect zero-day botnet attacks. To protect IoT edge devices from leaking personal data, this study presented the federated DL (FDL) approach for zero-day botnet attack detection. This approach uses an ideal deep neural network (DNN) model to classify network traffic.

In 2022, Bar and Hajaj [22] proposed the SimCSE for the prediction of encrypted traffic and zero-day attacks. The simple contrastive learning of sentence embedding (SimCSE) proposed in this study serves as the embedding model for a novel method for traffic identification at the packet level.

In 2021, Kumar and Sinha [23] proposed an effective method for detecting zero-day cyberattacks. This study suggests an innovative, robust, and intelligent cyber-attack detection model that makes use of the heavy-hitter idea and a graphical approach to identify zero-day assaults.

III. Problem Statement

Traditional intrusion detection approaches are no longer enough for identifying zero-day attacks, which may exploit security vulnerabilities that may exist in a network due to the fast expansion of network-based cyberattacks. With the advancement and new development of systems, cyber security vulnerabilities, and technology, existing approaches in intrusion detection have not been sufficient to defend systems from cyber security vulnerabilities. There is a need for more proactive ways to identify known and new vulnerabilities to stop zero-day attacks from taking advantage of such network security weaknesses. If a hacker is successful in exploiting a vulnerability in software before the developers of the affected system can find a solution, this is known as a zero-day attack [24]. Zero-day vulnerabilities can take on practically any form since they can be disguised as any type of more generic software flaw.

IV. Proposed Methodology

Detecting zero-day threats is one of the most challenging responsibilities of an IDS. An attack that makes use of a network vulnerability that has not yet been found is known as a zero-day attack. The first step in the life of a zero-day assault is the identification of a software vulnerability. Once an exploit has been made, the vulnerability is utilized to attack the targets. After the initial assaults are carried out, the vendors distributing the weak software learn about it and produce a patch to fix it. This project will design a fresh, optimized deep learning system to address the difficulties of detecting zero-day ransomware. Fig. 1 shows the proposed approach for detecting zero-day attacks.

From Fig. 1, the input data is received from the two datasets called CSE-CIC-IDS2018 and NSL-KDD. Then, the input data is cleaned and normalized in the pre-processing stage. The statistical features and higher-order features are extracted in the feature extraction stage; the required features are selected using the improved IG using the CESSO model. Then, the same CESSO will enhance the features of the RNN, and it is hybrid with the CNN to produce HCRNN, which is used for zero-day attack detection.

A. Dataset Description

In the Zero-day attack model, two datasets are used.

1) IDS 2018 Intrusion CSVs (CSE-CIC-IDS2018): This dataset was created by the University of New Brunswick initially for research on DDoS data. This dataset won't be modified in the future because it was totally compiled in 2018. The dataset was indeed based on logfiles kept by the institution, which revealed several DoS assaults during the course of the period that were made accessible to the public. Because

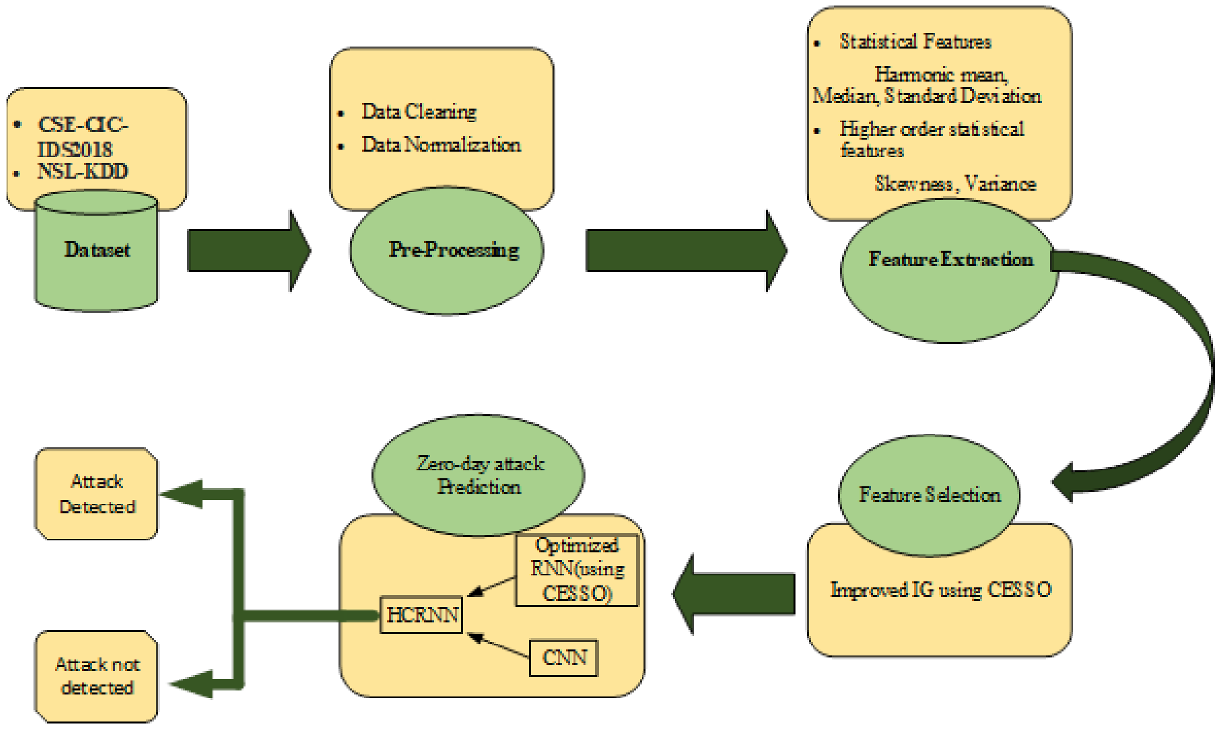

Fig. 1. Block diagram of the Zero attack detection mechanism.

it specifies whether or not the sent packets are malicious, the Label column is possibly the essential information for building machine learning algorithms for this dataset.

Based on date, data is separated into several files. Each file is imbalanced; the notebook creator is responsible for partitioning the dataset into a balanced shape for more accurate predictions. This dataset contains a total of 80 columns, each of which corresponds to a record in the IDS logging system used by the University of New Brunswick. Both columns for forward and backward traffic are present because their system distinguishes between the two. The following are the columns in this dataset that are most important: Label, Destination port, Protocol, Flow Duration, Total forward packets, Total backward packets, and Total forward packets [25].

2) NSL-KDD: As compared to the original KDD data set, the NSL-KDD data set is better in the following aspects:

The classifiers won't be skewed in favor of a more prominent impact since similar data are excluded from the train set. Due to the lack of duplicate data in the proposed test sets, methods with greater diagnostic accuracy for frequent records have no impact on the learners' outcomes. The proportion of records selected from each class of difficulty levels in the original KDD data set is inversely proportional to the total number of entries in each class.

As a result, there is a wider range of variance in the classification rates of different machine learning techniques, making it simpler to assess different learning approaches correctly. With a manageable number of records in both the train and test sets, it is possible to run the tests on the complete set without having to choose a subset at random. Therefore, the assessment results from different research initiatives will be comparable and consistent [26].

B. Pre-processing

The raw dataset gathered during the prior step must be cleaned and normalized during this stage. It is possible to access the dataset's raw log data set. To assist in identifying online assaults, the data is normalized, encrypted, and stripped of duplicate and missing information during this phase of the evaluation process.

1) Data Cleaning: Finding duplicate data is a common need of data cleaning operations. Duplicates should be eliminated since deep learning produces the same patterns when non-duplicate variables are used. The cleaned dataset has been obtained after some columns have been cleared of duplicate and missing values. This dataset contains no duplicates.

2) Normalization: By transforming them into unique patterns, data are normalized. The process of converting raw data into a format that can be handled effectively is known as data transformation. Normalization focuses mostly on reducing or even getting rid of redundant data. Because of this crucial problem, managing data in relational databases that store similar data in several places is getting harder.

C. Feature Extraction

The features such as statistical features (Harmonic mean, median, standard deviation), higher order statistical features (skewness, Variance), Improved Information gain (Proposed), and based features will then be retrieved from the preprocessed data.

1) Statistical Features:

a) Harmonic Mean: Numerical averages include the harmonic mean. It is determined by multiplying the number of observations, or series elements, by the inverse of each integer in the series. The harmonic mean is, therefore, the inverse of the arithmetic mean of the reciprocals, as shown in Eq. (1)

$$\mu_{\mathcal{H}} = \frac{T}{\sum_{i=1}^{n} \frac{1}{\alpha_i}} \tag{1}$$

where, α_i a series with T numbers.

b) Median: The median, which is the midpoint in a list of numbers arranged either ascending or descending, may be more representative of the set of data than the average.

$$Median = \begin{cases} \frac{l+1}{2}, & for\ odd\ numbers \\ \frac{l}{2}, & for\ even\ numbers \end{cases} \tag{2}$$

where, l is the last number.

c) Standard deviation: The term "standard deviation" (σ) refers to a measurement of the data's dispersion from the mean. The values are centered all around the mean whenever the standard deviation is small and widely scattered when it is high.

$$SD(\sigma) = \sqrt{\frac{\sum_{i=1}^{n}(\alpha_i - \mu)^2}{N}} \tag{3}$$

where α_i is the input value, μ is the mean and N is the total number of elements.

2) Higher-order statistical features:

a) Skewness: A distribution's skewness can be measured to see how asymmetrical it is. A distribution is asymmetric whenever its left and right sides do not reflect each other in the same way. A distribution may have right (or positive), left (or negative), or zero skewness.

$$Skewness = \frac{3(\mu - Median)}{\sigma} \tag{4}$$

b) Variance: Variance is the analytical evaluation of the numerical variation inside a data set. More specifically, variance determines how far apart every integer in the set is from the mean and, thus, from the other numbers in the set.

$$Variance = \frac{\sum_{i=1}^{n}(\alpha_i - \mu)^2}{N} \tag{5}$$

D. Feature Selection

This section introduces a unique hybrid feature selection method that is based on the CESSO algorithm and IG. Let's start by outlining the IG algorithm and then describe how the CESSO technique was implemented in detail.

E. Improved Information Gain

Information gain uses correlations between features and classifications to distribute feature weights statistically. Let $F = \{f_1,\ f_2,\ f_3, \ldots, f_n\}$ be the group of n data points in a dataset, $X = \{x_1,\ x_2,\ x_3, \ldots, x_p\}$ be the collection of p features, and $Y = \{y_1,\ y_2,\ y_3, \ldots, y_m\}$ be a group of m type label. The number $P(Cl_i)$ denotes the percentage of classes in F with the label P_i where $i = 1,\ 2,\ 3, ..., m$. Eq. (6) provides the dataset's entropy.

$$H(Y) = -\sum_{i=1}^{m} P(Cl_i)\ log_2\ P(Cl_i) \tag{6}$$

Each feature in the data classification system has an information gain (IG). Where $Q = \{Q_j^1,\ Q_j^2, , \ldots, Q_j^p\}$ represents the $p^{th}(p = 1, 2, \ldots, n)$ a data set feature. Regarding the characteristic, $Q = \{Q_j^1,\ Q_j^2, , \ldots, Q_j^p\}$ the corresponding conditional entropy is:

$$H(Y|Q_j) = -\sum_{p=1}^{k} P(Q_j^p) \sum_{i=1}^{m} P(Cl_i|Q_j^p) log_2 P(Cl_i|Q_j^p) \tag{7}$$

where, $P(Q_j^p)$ denotes the categorical variable Cl's prior probability and Q_j^p is the value of Q_j, an attribute with k different kinds of values. The $P(Cl_i|Q_j^p)$ represents the conditional probability of the variable Cl after the attribute Q_j is fixed. Consequently, the formula below states that the value of the information gained from the attribute Q_j is provided by the difference between $H(Y), H(Y|Q_j)$, which is given by Eq. (8).

$$IG(Q_j) = H(Y) - H(Y|Q_j) \tag{8}$$

An increased IG often denotes the importance of the feature for the categorization.

F. IG-CESSO for Optimal Feature Subset Selection

The CESSO algorithm finds the optimal feature subset with the fewest features and highest classification accuracy, which is what feature selection seeks to do for a given dataset. It is utilized for giving each component a binary string to represent whether an attribute is picked; for example, the feature is chosen when the binary value is 1, whereas a binary 0 denotes that it is not.

Each of these two indications uniquely impacts the classifier's classification performance. In this instance, just merge them into a single weighted indicator and use the same fitness function as in Eq. (9).

$$F_i = \delta_1 * A_{clasifier} + \delta_2 * (1 - \frac{N}{T}) \tag{9}$$

where T stands for the total number of characteristics and N for the number of attributes that were selected; here, δ_1 and δ_2 have values of 1 and 0.001, respectively[27]. Eq. (10) may be used to determine the classification accuracy derived from the HCRNN classifier.

$$A_{classifier} = \frac{K_c}{K_c + K_i} \tag{10}$$

In this example, the numbers K_i and K_c stand for the cases that were classified wrongly and properly, respectively. With the help of the fitness value, it is possible to ensure that the chosen features have a limited amount of features while yet providing the highest possible classification accuracy[27]. The following section provides a detailed description of the CESSO algorithm with basic Salp Swarm Optimization (SSO)

1) Salp Swarm Optimization: The Salp Swarm Optimization (SSO) imitates the behavior of salps, a kind of planktonic tunicate belonging to the Salpidae family with a barrel-like structure. Additionally, their tissues resemble those of jellyfish, and they move similarly to jellyfish, and a large portion of their weight is made up of water. Pumping water through their jello-like bodies causes them to move by contracting, which alters their locations. Salps in the seas engage in a swarm activity known as the salp chain, which may aid salps in better mobility by employing rapid, coordinated shifts and feeding. Based on this behavior, they created a mathematical model of the salp chains and evaluated it using optimization issues. The leader and the followers are the two groups that are initially separated out in SSO [28]. The leading Salp in a chain is referred to as the leader, while the trailing Salps are referred to as the followers. In n-dimensions, where n stands for the variables in the issue, and n denotes the search space, the salps' location is established. These salps look for food sources since they can identify the swarm's goal. The position should be updated often. Thus, the salp leader updates the position using Eq. (11) below-

$$a_j^1 = \begin{cases} S_j + K_1((U_j - L_j) * K_2 + L_j), & K_3 <= 0 \\ S_j - K_1((U_j - L_j) * K_2 + L_j), & K_3 > 0 \end{cases} \quad (11)$$

where, a_j^1 is the position of the leader within j^{th} dimension, where the food source in this dimension is S_j, the upper and the lower bounds are U_j and L_j , respectively. The K_2 and K_3 are generated randomly in the range [0, 1] to maintain the search space. Additionally, the parameter K_1 is a crucial coefficient in this method since it helps to balance the exploration and exploitation phases [29]. It is derived as follows:

$$K_1 = 2e^{-(\frac{v}{v_{max}})^2} \quad (12)$$

where, the letters v and v_{max} stand for the current iteration and the maximum number of iterations, respectively. Using Eq. (13), the SSA begins updating the followers' position after changing the leader's position.:

$$a_j^i = \frac{1}{2}(a_j^i + a_j^{i-1}) \quad (13)$$

where a_j^i is the i^{th} position of the follower within j^{th} dimension and $i > 1$.[28]

However, the SSO has the limitation of falling into the local optima problem. Therefore, the proposed work has introduced a new SSO variant based on the Chimp Optimization Algorithm, i.e., the Chaotic Map. Therefore, chaotic enriched SSO provides high-level efficiency in optimization performance. The following shows a detailed description of the CESSO algorithm.

2) CESSO-based on COA Algorithm: Salp Swarm Optimization is improved by incorporating the strategy of the COA. The COA enhances the performance of the SSO due to the incorporation of the chaotic map strategy. The COA is mainly focusing on hunting its prey based on determining the prey and the chimp. In the CESSO algorithm, this step is reformulated by considering the current best solution, and the next solution is respectively represented as a_j^i and $a_j^(i-1)$. The mathematical derivation of the Chaotic map based on the COA [30] is shown in Eq. (14) and Eq. (15).

$$d = |K.a_j^i - C_v.a_j^{i-1}| \quad (14)$$

$$a_j^i(t+1) = a_j^i - p.d \quad (15)$$

where t denotes the current iteration, p, C_v, and c are the coefficient vectors, $a_j^{i-1}$is the position vector of the targeted solution, and a_j^i is the position vector of a current best solution. The coefficient vectors p, C_v, and K are determined using Eq. (16).

$$\begin{cases} p = 2.f.r_1 - f \\ K = 2.r_2 \\ C_v = Chaoticvalue \end{cases} \quad (16)$$

In the aforementioned equation, the variable f has a specific range, i.e., drastically minimizing from 2.5 to 0 in every iteration process [30]. Therefore, exploitation and exploration have maintained the equilibrium. On the other hand, the random vectors are set between the range of 0 and 1. At last, the chaotic vector is illustrated as m, i.e., a strategy of the COA. This strategy is executed based on executing several chaotic map processes. This strategy can enrich the optimization performance, and the chaotic enriched chimp-based salp optimization can provide higher performance than the traditional SSO and the COA. This hybrid model, Chaotic Enriched SSO (CESSO), can provide better performance.

The proposed work uses the CESSO algorithm to improve the feature selection, especially on Information Gain (IG). The IG-CESSO provides a high-level feature that helps to improve the classification performance. Apart from feature selection, the CESSO algorithm is utilized for improving the prediction performance; therefore, the parameters of the RNN are optimized using CESSO, which is discussed in section IV.G.2.

G. Prediction of Zero-day Attack

The optimized hybrid classifier that makes the zero-day attack detection will be trained using the selected optimal features. The hybrid classifier will be a combination of optimized RNN and CNN. To enhance the prediction accuracy of the zero-day attack detection model, the activation function of RNN is fine-tuned via a new CESSO.

1) CNN: The suggested model's structure is discussed in this part, along with specifics on its underlying motives. The proposed model has developed based on a set of layers, including the Input layer, Convolutional Layer, Max Pooling, and Recurrent Layer or Output Layer. Here, the CNN architecture has the fully connected layer that is replaced by Recursive Neural Networks (RNN); therefore, the proposed model has concatenated the Convolutional and recurrent layers for detecting zero-day attacks. These layers are discussed in their respective sections as follows.

In the architecture of the HCRNN for Zero Day attack detection, the CNN executes the layer-wise feature extraction using convolution layers and pooling layers [31]. The shortcomings of conventional machine learning methods, which need to extract data features manually, are overcome by CNN because it does not call for preprocessing or reconstruction of the original data. Furthermore, the model's complexity is substantially reduced by its weight-sharing characteristics.

The CNN's structure is depicted in Fig. 2, and the kernel function transforms the input data, which must be in two-dimensional form. The hidden layer comprises several convolution layers and pooling layers, where the input of one layer becomes the output of the following layer. There are not many restrictions on the input data using this structure, and it can accurately retrieve the hierarchical expression of the input data.

a. Convolutional Layer The convolution layer—which includes several feature matrices—is used to extract the local characteristics of the input data. Each feature matrix is capable of parallel processing since it can be thought of as a plane, which can significantly reduce the number of free parameters. Additionally, because convolution kernels vary depending on the plane, it is possible to display the extracted features completely. The following Eq. (17) and Eq. (18) determine the size of the output feature map following the convolution procedure:

$$h_o = [\frac{h_i - f_i + p_c}{s_i} + 1] \tag{17}$$

$$w_o = [\frac{w_i - f_i + p_c}{s_i} + 1] \tag{18}$$

Where the resulting feature map's height is h_o, w_o is the feature map's width at the output side, the input image's height is represented as h_i, w_i is the representation of input image's width, the size of the filter is denoted as f_i, the convolution operation's padding is represented as p_c, and the convolution' stride is denoted as s_i.

b. Max Pooling

The typical pooling technique known as max pooling (MP) operates on the feature map side and picks the largest value from a small n×n patch of the input feature map. The output feature map is then created by MP by combining the chosen values. The following Eq. (19) and Eq. (20) may be used to determine the output feature map size following the pooling procedure:

$$h_o = [\frac{h_i - f_i}{s_i}] \tag{19}$$

$$w_o = [\frac{w_i - f_i}{s_i}] \tag{20}$$

where, f_i is the size of the pooling area, the output feature map's height is indicated by the symbol h_o, The output feature map's width is indicated by the prefixw_o, the width of the input feature map is indicated by the letter w, the height of the input feature map's is indicated by the symbol h_i, and the width of the input feature map is denoted as w_i.

2) Recurrent Layer For Prediction: In the HCRNN architecture, the RNN layer joints the CNN by replacing the fully connected layer for prediction. The reason behind the selection of the RNN layer is for selecting temporal features, whereas CNN is for dealing with spatial features. However, the RNN has a drawback in that phrase parsing can be complex and slow. Interestingly, different parse trees may exist for the same text. Additionally, labeling the training data for recursive neural networks takes more time and effort than building recurrent neural networks. It takes more time and effort to manually break down a statement into smaller parts than it does to give it a label. To overcome this drawback, the parameters of the RNN are optimized using the CESSO algorithm.

The trees are symbolized by the letter T, and N samples are indicating the leaves of a tree. The neural network receives concatenated data containing a tree's possible child nodes. A combined score and a parent feature are produced by the neural network, which are the two features combined after being learned by a nonlinear mapping. A parent is created mathematically by applying a non-linear projection on the concatenated of the child characteristics, that is

$$PA_{x,y} = \rho[W_i(ch_x + ch_y) + B_p] \tag{21}$$

Where $(ch_x + ch_y)$ is the result of concatenating the child features x and y, W_i is the neural network's weight matrix, B_p is its bias parameter, ρ is a function that may be continually changed, and $PA_(x, y)$ is the last parent attribute [32]. The sign for this work was the logistic sigmoid since its derivative may be stated in terms of the primary equation. The neural network's learned merging score is used to determine whether samples are suitable for merging on the tree, as well as if they are neighboring when appropriate. To produce a binary tree, combining just the samples with the top score is permitted. The parent feature's score is calculated using Eq. (22)

$$PA_s = W^k_{score} PA_{x,y} \tag{22}$$

While the tree's score can be increased K, it is possible to determine the parameters W, B_p, and W^K_{score} of the neural network,

$$\operatorname*{argmax}_{k \in N} PA_s(RNN(\theta, x, k)) \tag{23}$$

where θ is the representation of neural network parameters to calculate a score PA_s, and T are any potential trees produced by the aforementioned merging procedure. Finally, the RNN provides the outcome with respect to zero-day attacks. Moreover, the prediction performance is further improved by using the CESSO algorithm. The CESSO algorithm utilizes the

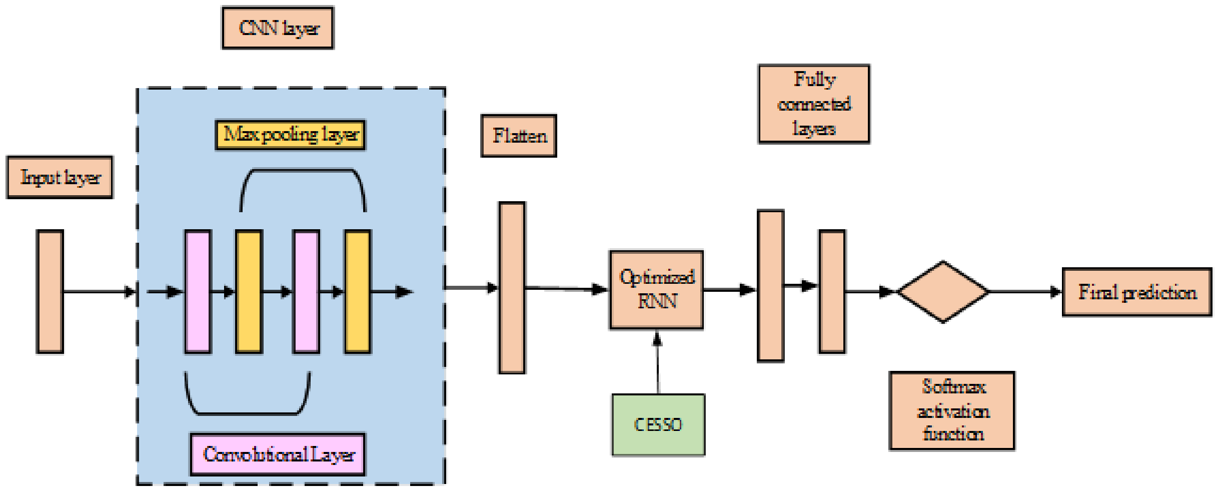

Fig. 2. Hybrid CNN with RNN architecture.

parameters of the RNN such as BATCH SIZE, MINI-BATCH, MOMENTUM, and LEARNING RATE. The prediction performance is further improved by this strategy. In this process, the CESSO algorithm evaluated the prediction performance of the RNN using the RMSE algorithm Eq. (24)

$$RMSE = \frac{1}{n}\sum_{i=1}^{n} w_i(y_i - \hat{y}_i)^2 \tag{24}$$

The n Number of samples, y_i Original value, y_i cap: Predicted outcome.

In the process, the RNN generates the value that is evaluated through the RMSE value. The minimal RMSE is considered the best solution. In each outcome, the parameter values are changed and updated with new solutions. Finally, the prediction performance of the RNN will be reached higher results. The overall flow of the proposed model is shown in Fig. 3.

V. Result and Discussion

The performance of the proposed strategy is compared to that of well-known methods such as CESSO-RNN, RNN, CNN, Bidirectional Long Short-Term Memory (Bi-LSTM), and Long Short-Term Memory (LSTM). The proposed CESSO-HCRNN model is compared with all existing models and performance for this is tabulated. The performance of the proposed is better than the other existing models because of improved IG using CESSO used in feature selection. Optimal RNN combined with CNN resultant the HCRNN also produces more accurate results. In this part, the performance measurements are also covered.

A. Performance Metrics

A number of matrices are used to measure the performance, including sensitivity, specificity, accuracy, recall, F-score, NPV, MCC, FPR, and FNR.

Sensitivity

The sensitivity value is obtained by just dividing the total positives by the proportion of true positive predictions

$$Sensitivity = \frac{TP}{TP + FN} \tag{25}$$

Specificity

Specificity is calculated by dividing the number of accurately anticipated negative outcomes by the total number of negatives

$$Specificity = \frac{TN}{TN + FP} \tag{26}$$

Accuracy The accuracy is the ratio of correctly classified data to all of the data in the log. The precision is described below-

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{27}$$

Precision By employing the entire number of samples used in the classification process, precision is the representation of the total number of genuine samples that are appropriately taken into consideration during the classification process.

$$Precision = \frac{TP}{TP + FP} \tag{28}$$

Recall

Recall rate is a measure of how many genuine samples overall are considered when categorizing data using all samples from the same categories from the training data.

$$Recall = \frac{TP}{TP + FN} \tag{29}$$

F- Measure

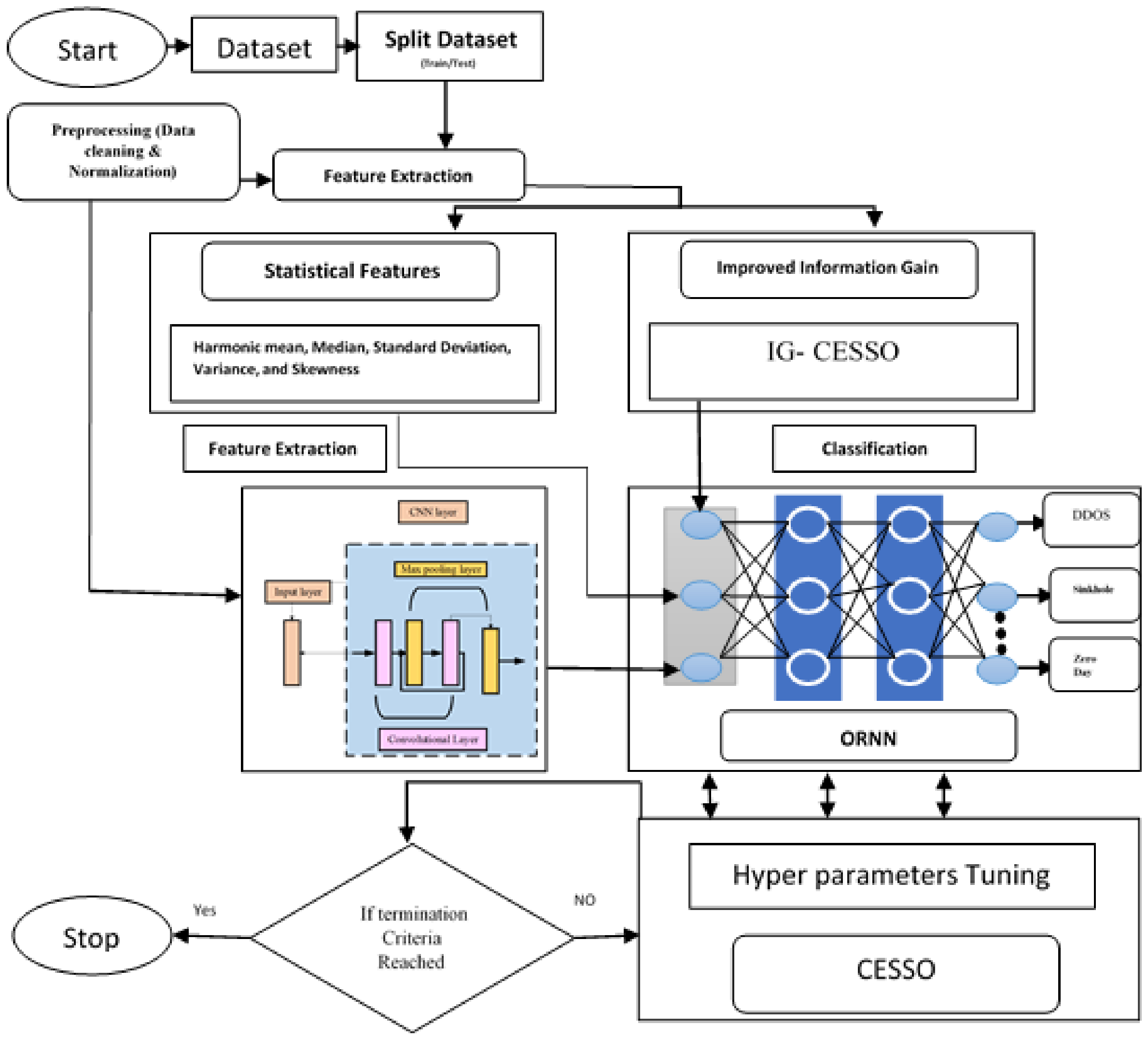

Fig. 3. Overall architecture of the proposed CESSO-HCRNN model.

The definition of the F-score is the harmonic mean of recall rate and accuracy.

$$F_{measure} = \frac{2precision * Recall}{Precision + Recall} \tag{30}$$

Negative Prediction Value (NPV)

NPV describes the effectiveness of a diagnostic test or other quantitative metrics.

$$NPV = \frac{TN}{TN + FN} \tag{31}$$

Matthews correlation coefficient (MCC) Below is a representation of the two-by-two binary variable association measure known as MCC.

$$MCC = \frac{(TP * TN - FP * FN)}{\sqrt{(TP + FN)(TN + FP)(TN + FN)(TP + FP)}} \tag{32}$$

False Positive Ratio (FPR) The number of negative occurrences divided by the number of negative events that were incorrectly classified as positive yields the false positive rate (false positives).

$$FPR = \frac{FP}{FP + TN} \tag{33}$$

False Negative Ratio (FNR) The likelihood that an actual positive may be overlooked by the test is known as the false-negative rate, sometimes referred to as the “miss rate”.

$$FNR = \frac{FN}{FN + TP} \tag{34}$$

B. *Comparative analysis for the performance metrics*

1) CSE-CIC-IDS2018 Dataset: Using the CSE-CIC-IDS2018 Dataset, the performance metrics for the proposed CESSO-HCRNN approach were computed and contrasted with

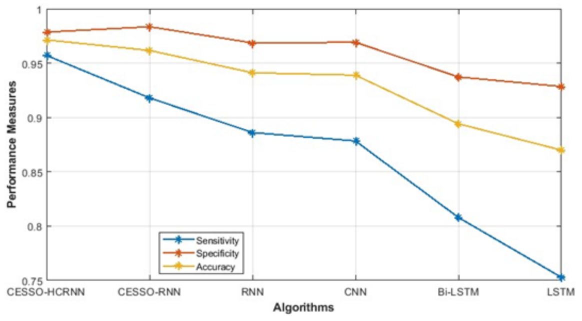

Fig. 4. Comparison of sensitivity, specificity, and accuracy of CSE-CIC-IDS2018 dataset for the proposed CESSO-HCRNN and existing techniques.

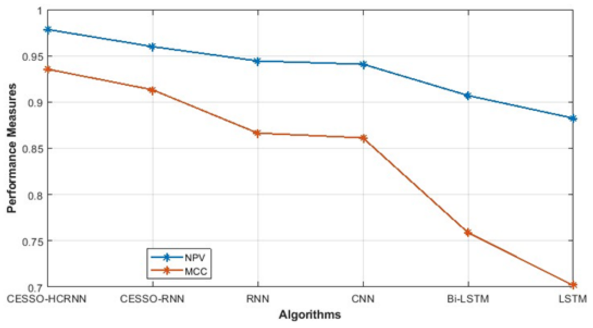

Fig. 6. Comparison of the NPV and MCC of CSE-CIC-IDS2018 dataset for the proposed CESSO-HCRNN and existing techniques.

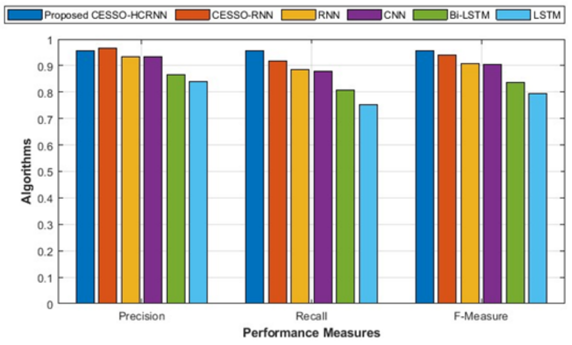

Fig. 5. Comparison of the precision, recall, and F- Measure of CSE-CIC-IDS2018 dataset for the proposed CESSO-HCRNN and existing techniques.

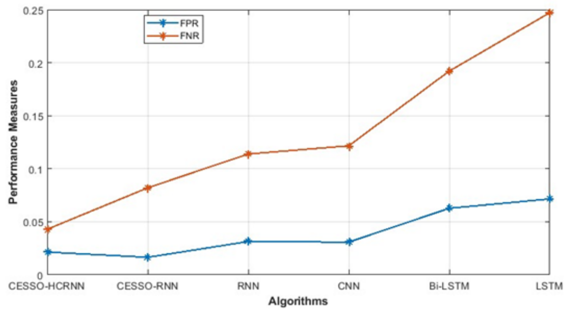

Fig. 7. Comparison of FPR and FNR of CSE-CIC-IDS2018 dataset for the proposed CESSO-HCRNN and existing techniques.

those of CESSO-RNN, RNN, CNN, Bi-LSTM, and LSTM. The suggested CESSO-HCRNN strategy's efficacy is evaluated here using the other metrics, whilst the proposed model's efficacy is evaluated here using measures like accuracy, sensitivity, precision, recall, and F-Measure. Table I compares the CSE-CIC-IDS2018 Dataset's performance metrics.

The metrics in the table are analyzed and their values are combined. Comparing the suggested CESSO-HCRNN model against older methods reveals that it is quite accurate. Graphs are used to show the difference in performance. The suggested and current models are contrasted using the graph in Fig. 4 in terms of metrics like sensitivity, specificity, and accuracy.

Fig. 4 displays performance metrics, such as sensitivity, specificity, and accuracy, for both the proposed and current approaches. Sensitivity values are 0.9571, 0.9180, 0.8861, 0.8784, 0.8079, and 0.7529 for the proposed CESSO-RNN, RNN, CNN, Bi-LSTM, and LSTM. The accuracy values are 0.9714, 0.9617, 0.9411, 0.9389, 0.8941, and 0.8699, respectively, while the specificity values are 0.9785, 0.9835, 0.9686, 0.9692, 0.9373, and 0.9285. The proposed model performs better in terms of sensitivity, specificity, and accuracy than the widely used methods.

Precision, recall, and F-Measure performance characteristics for the proposed CESSO-HCRNN and current techniques are shown in Fig. 5. Precision values for the proposed CESSO-RNN, RNN, CNN, Bi-LSTM, and LSTM are 0.9571, 0.9653, 0.9339, 0.9344, 0.8656, and 0.8403 respectively. Recall values for the proposed CESSO-HCRNN and current approaches are 0.9571, 0.9180, 0.8861, 0.8784, 0.8079, and 0.7529, respectively. The F-measure values are 0.9571, 0.9410, 0.9093, 0.9055, 0.8358, and 0.7942. The suggested CESSO-HCRNN model provides higher precision, recall, and F- Measure than the current approaches.

Fig. 6 illustrates the performance metrics, including NPV and MCC, for the proposed and existing techniques. The proposed CESSO-HCRNN and current models like CESSO-RNN, RNN, CNN, Bi-LSTM, and LSTM have NPV values of 0.9785, 0.9600, 0.9445, 0.9071, and 0.8826, respectively. The MCC values are 0.9356, 0.9133, 0.8664, 0.8614, 0.7588, and 0.7018 as well.

Fig. 7 visually represents the FPR and FNR performance indicators for both the proposed CESSO-HCRNN and the existing methods. The CESSO-HCRNN model has a lower FPR and FNR when compared to the approaches currently in use. The suggested CESSO- HCRNN and current approaches like CESSO-RNN, RNN, CNN, Bi-LSTM, and LSTM have the FPR and FNR are 0.0215, 0.0165, 0.0314, 0.0308, 0.0627, and 0.0715; and 0.0429, 0.0820, 0.1139, 0.1216, 0.1921, and 0.2471.

2) NSL- KDD Dataset: Performance metrics for the proposed CESSO-HCRNN approach are compared to those for

TABLE I. COMPARISON OF PERFORMANCE METRICS OF CSE-CIC-IDS2018 DATASET FOR THE PROPOSED CESSO-HCRNN AND EXISTING TECHNIQUES

Techniques	Sensitivity	Specificity	Accuracy	Precision	Recall	F-Measure	NPV	MCC	FPR	FNR
Proposed CESSO - HCRNN	0.9571	0.9785	0.9714	0.9571	0.9571	0.9571	0.9785	0.9356	0.0215	0.0429
CESSO-RNN	0.9180	0.9835	0.9617	0.9653	0.9180	0.9410	0.9600	0.9133	0.0165	0.0820
RNN	0.8861	0.9686	0.9411	0.9339	0.8861	0.9093	0.9445	0.8664	0.0314	0.1139
CNN	0.8784	0.9692	0.9389	0.9344	0.8784	0.9055	0.9410	0.8614	0.0308	0.1216
Bi-LSTM	0.8079	0.9373	0.8941	0.8656	0.8079	0.8358	0.9071	0.7588	0.0627	0.1921
LSTM	0.7529	0.9285	0.8699	0.8403	0.7529	0.7942	0.8826	0.7018	0.0715	0.2471

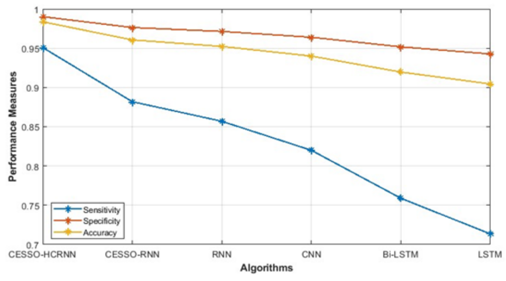

Fig. 8. Comparison of the sensitivity, specificity, and accuracy of NSL-KDD dataset for the proposed CESSO-HCRNN and existing techniques.

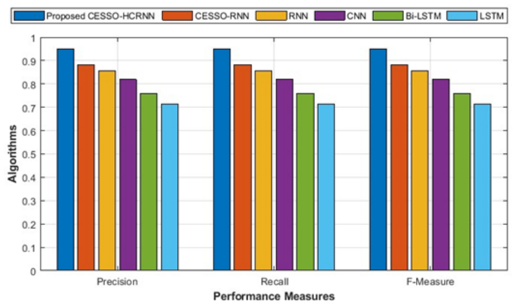

Fig. 9. Comparison of precision, recall, and F- Measure of NSL-KDD dataset for the proposed CESSO-HCRNN and existing techniques.

existing techniques like CESSO-RNN, RNN, CNN, Bi-LSTM, and LSTM. In this case, the loss values are evaluated using the FNR and FPR error measures, and the effectiveness of the suggested strategy and the chosen dataset for zero-day attack prediction are determined using the other metrics. A comparison of the performance metrics is shown in Table II.

From the table, the metrics values are calculated and contrasted. The proposed model is quite accurate when compared to previous approaches. To show the performance contrast, graphs are utilized. Utilizing the NSL-KDD dataset, the CESSO method is used to improve the prediction of zero-day attacks.

For both the proposed and current methodologies, Fig. 8 shows the graphical depiction of performance parameters including sensitivity, specificity, and accuracy. In comparison to previous methods, the CESSO-HCRNN model has good accuracy, sensitivity, and specificity.

Fig. 9 displays performance characteristics such as precision, recall, and F-measure graphically for both the proposed and existing techniques. When compared to earlier approaches, the CESSO-HCRNN model offers high precision, recall, and F- Measure.

Fig. 10 illustrates the performance metrics, including NPV and MCC, for the proposed and existing techniques. The CESSO-HCRNN model has a higher NPV and MCC when compared to the methods currently in use.

Fig. 11 visually represents the FPR and FNR performance metrics for the proposed and current techniques. The CESSO-HCRNN model exhibits lower FPR and FNR when compared to currently employed approaches, according to the comparison.

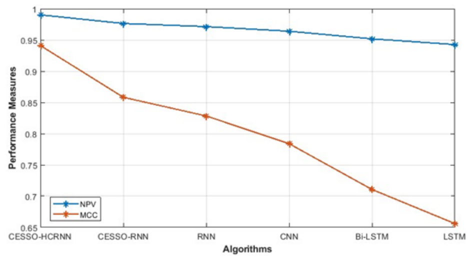

Fig. 10. Comparison of the NPV and MCC of NSL-KDD dataset for the proposed CESSO-HCRNN and existing techniques.

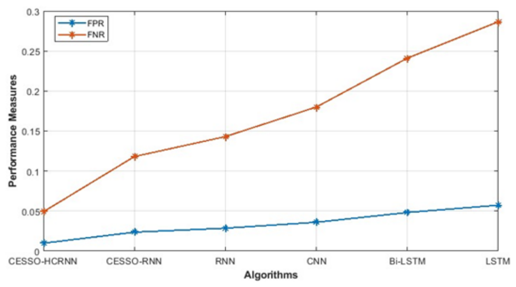

Fig. 11. Comparison of FPR and FNR of NSL-KDD dataset for the proposed CESSO-HCRNN and existing techniques.

TABLE II. Comparison of Performance Metrics of NSL-KDD Dataset for the Proposed CESSO-HCRNN and Existing Techniques

Techniques	Sensitivity	Specificity	Accuracy	Precision	Recall	F-Measure	NPV	MCC	FPR	FNR
Proposed CESSO - HCRNN	0.9509	0.9902	0.9836	0.9509	0.9509	0.9509	0.9902	0.9410	0.0098	0.0491
CESSO-RNN	0.8819	0.9764	0.9606	0.8819	0.8819	0.8819	0.9764	0.8582	0.0236	0.1181
RNN	0.8569	0.9714	0.9523	0.8569	0.8569	0.8569	0.9714	0.8283	0.0286	0.1431
CNN	0.8200	0.9640	0.9400	0.8200	0.8200	0.8200	0.9640	0.7840	0.0360	0.1800
Bi-LSTM	0.7589	0.9518	0.9196	0.7589	0.7589	0.7589	0.9518	0.7107	0.0482	0.2411
LSTM	0.7134	0.9427	0.9045	0.7134	0.7134	0.7134	0.9427	0.6560	0.0573	0.2866

VI. Conclusion

As a zero-day attack is a random assault that cannot be anticipated, the zero-day attack has recently taken on highly dangerous consequences. The zero-day threat takes the use of a software vulnerability to access the system or do significant harm, and system developers have no time to fix this vulnerability to reduce the threat. The input data is received from the two datasets, the missing values are removed, and the data normalization is performed in the pre-processed section. A unique hybrid feature selection method that is based on the CESSO and Information Gain(IG) are implemented. The CESSO is also used to improve the Recursive Neural Network (RNN) performance to produce an optimized RNN. The improved RNN is hybrid with the CNN to produce the HCRNN, which is used to predict the zero-day attack. The results of the proposed CESSO-HCRNN are compared with the existing models with reduced error values.

In two test cases, the performance of the proposed methodology is examined. The NSL-KDD dataset is used as the initial test case for the proposed system. The DoS module of the CSE-CIC-IDS2018 is utilized as the source dataset in the second test scenario, where the framework is employed to identify zero-day attacks on the NSL-KDD dataset. In this case, both domains have unique feature spaces and probability distributions. Although there is significant variation between the source and target domains, the experimental findings show that the suggested strategy is effective in identifying zero-day attacks with no tagged occurrences.

References

[1] P. Anand, Y. Singh, and A. Selwal, “Learning-based techniques for assessing zero-day attacks and vulnerabilities in iot,” *Recent Innovations in Computing: Proceedings of ICRIC 2021, Volume 1*, pp. 497–504, 2022.

[2] M. Nkongolo, J. P. Van Deventer, S. M. Kasongo, S. R. Zahra, and J. Kipongo, “A cloud based optimization method for zero-day threats detection using genetic algorithm and ensemble learning,” *Electronics*, vol. 11, no. 11, p. 1749, 2022.

[3] R. Kumar and G. Subbiah, “Zero-day malware detection and effective malware analysis using shapley ensemble boosting and bagging approach,” *Sensors*, vol. 22, no. 7, p. 2798, 2022.

[4] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, and S. Venkatraman, “Robust intelligent malware detection using deep learning,” *IEEE Access*, vol. 7, pp. 46 717–46 738, 2019.

[5] A. Mihoub, O. B. Fredj, O. Cheikhrouhou, A. Derhab, and M. Krichen, “Denial of service attack detection and mitigation for internet of things using looking-back-enabled machine learning techniques,” *Computers & Electrical Engineering*, vol. 98, p. 107716, 2022.

[6] B. I. Hairab, M. S. Elsayed, A. D. Jurcut, and M. A. Azer, “Anomaly detection based on cnn and regularization techniques against zero-day attacks in iot networks,” *IEEE Access*, vol. 10, pp. 98 427–98 440, 2022.

[7] A. Abeshu and N. Chilamkurti, “Deep learning: The frontier for distributed attack detection in fog-to-things computing,” *IEEE Communications Magazine*, vol. 56, no. 2, pp. 169–175, 2018.

[8] A. Fatima, S. Kumar, and M. K. Dutta, “Host-server-based malware detection system for android platforms using machine learning,” in *Advances in Computational Intelligence and Communication Technology: Proceedings of CICT 2019*. Springer, 2021, pp. 195–205.

[9] I. A. Khan, N. Moustafa, D. Pi, W. Haider, B. Li, and A. Jolfaei, “An enhanced multi-stage deep learning framework for detecting malicious activities from autonomous vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 25 469–25 478, 2021.

[10] M. Ali, A. Siddique, A. Hussain, F. Hassan, A. Ijaz, and A. Mehmood, “A sustainable framework for preventing iot systems from zero day ddos attacks by machine learning,” *Int. J. Emerg. Technol*, vol. 12, pp. 116–121, 2021.

[11] W. Haider, N. Moustafa, M. Keshk, A. Fernandez, K.-K. R. Choo, and A. Wahab, “Fgmc-hads: Fuzzy gaussian mixture-based correntropy models for detecting zero-day attacks from linux systems,” *Computers & Security*, vol. 96, p. 101906, 2020.

[12] V. Sharma, K. Lee, S. Kwon, J. Kim, H. Park, K. Yim, and S.-Y. Lee, “A consensus framework for reliability and mitigation of zero-day attacks in iot,” *Security and Communication Networks*, vol. 2017, 2017.

[13] S. Venkatraman and M. Alazab, “Use of data visualisation for zero-day malware detection,” *Security and Communication Networks*, vol. 2018, pp. 1–13, 2018.

[14] F. Alhaidari, N. A. Shaib, M. Alsafi, H. Alharbi, M. Alawami, R. Aljindan, A.-u. Rahman, and R. Zagrouba, “Zevigilante: Detecting zero-day malware using machine learning and sandboxing analysis techniques,” *Computational Intelligence and Neuroscience*, vol. 2022, 2022.

[15] W.-S. Choi, S.-Y. Lee, and S.-G. Choi, “Implementation and design of a zero-day intrusion detection and response system for responding to network security blind spots,” *Mobile Information Systems*, vol. 2022, 2022.

[16] X. Sun, J. Dai, P. Liu, A. Singhal, and J. Yen, “Using bayesian networks for probabilistic identification of zero-day attack paths,” *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2506–2521, 2018.

[17] Y. Afek, A. Bremler-Barr, and S. L. Feibish, “Zero-day signature extraction for high-volume attacks,” *IEEE/ACM Transactions on Networking*, vol. 27, no. 2, pp. 691–706, 2019.

[18] T. Zoppi, A. Ceccarelli, and A. Bondavalli, “Unsupervised algorithms to detect zero-day attacks: Strategy and application,” *Ieee Access*, vol. 9, pp. 90 603–90 615, 2021.

[19] I. Mbona and J. H. Eloff, “Detecting zero-day intrusion attacks using semi-supervised machine learning approaches,” *IEEE Access*, vol. 10, pp. 69 822–69 838, 2022.

[20] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, “Federated deep learning for zero-day botnet attack detection in iot-edge devices,” *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3930–3944, 2021.

[21] R. Bar and C. Hajaj, “Simcse for encrypted traffic detection and zero-day attack detection,” *IEEE Access*, vol. 10, pp. 56 952–56 960, 2022.

[22] M. Zhang, L. Wang, S. Jajodia, A. Singhal, and M. Albanese, “Network diversity: a security metric for evaluating the resilience of networks against zero-day attacks,” *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 5, pp. 1071–1086, 2016.

[23] V. Kumar and D. Sinha, “A robust intelligent zero-day cyber-attack detection technique,” *Complex & Intelligent Systems*, vol. 7, no. 5, pp. 2211–2234, 2021.

[24] A. Blaise, M. Bouet, V. Conan, and S. Secci, “Detection of zero-day attacks: An unsupervised port-based approach,” *Computer Networks*, vol. 180, p. 107391, 2020.

[25] “Data set1 collected from:,” https://www.kaggle.com/datasets/solarmainframe/ids-intrusion-csv” , dated on 29/11/2022.

[26] “Data set2 collected from:,” “https://www.kaggle.com/datasets/hassan06/nslkdd” , dated on 29/11/2022.

[27] G. Zhang, J. Hou, J. Wang, C. Yan, and J. Luo, “Feature selection for microarray data classification using hybrid information gain and a modified binary krill herd algorithm,” *Interdisciplinary Sciences: Computational Life Sciences*, vol. 12, pp. 288–301, 2020.

[28] S. Kassaymeh, S. Abdullah, M. A. Al-Betar, and M. Alweshah, “Salp swarm optimizer for modeling the software fault prediction problem,” *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 6, pp. 3365–3378, 2022.

[29] N. Singh, S. Singh, and E. H. Houssein, “Hybridizing salp swarm algorithm with particle swarm optimization algorithm for recent optimization functions,” *Evolutionary Intelligence*, pp. 1–34, 2022.

[30] M. Khishe and M. R. Mosavi, “Chimp optimization algorithm,” *Expert systems with applications*, vol. 149, p. 113338, 2020.

[31] S. U. Amin, M. Alsulaiman, G. Muhammad, M. A. Bencherif, and M. S. Hossain, “Multilevel weighted feature fusion using convolutional neural networks for eeg motor imagery classification,” *Ieee Access*, vol. 7, pp. 18 940–18 950, 2019.

[32] J. Ma, W. Gao, S. Joty, and K.-F. Wong, “An attention-based rumor detection model with tree-structured recursive neural networks,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, no. 4, pp. 1–28, 2020.