GMR
The Original

Updating A Heuristic And Optimizer Learning Model For Intrusion Detection In The Internet Of Things Environment

Bino Thomas[1], Dr S.P.Sasirekha[2]

[1]Research Scholar, Department of Computer Science & Engineering Karpagam Academy of Higher Education Coimbatore, India. Corresponding Author:binot608@gmail.com
[2]Assistant Professor Department of Computer Science Engineering Karpagam Academy of Higher Education Coimbatore, India. Email:csrd411@gmail.com

Abstract- Web technologies in the modern period have produced an enormous amount of data. The relationship between various gadgets and services has also been investigated to make effective and extensive use of modern technologies. Because of these restrictions, the probability of a security breach is significantly rising on devices with limited resources. Improved scalability and dependability of public services can be achieved by integrating an IoT backend with multi-cloud architecture. Managing user requests for IoT services may include several users accessing multi-cloud resources, posing a data security risk. Coming up with new functional components and security plans becomes more difficult. This study presents an advanced Intrusion Detection Model (IDM) designed to detect threats in both network and application contexts. The framework is organized into three essential phases: data preprocessing, feature selection, and classification. In the initial phase, the gathered datasets are subjected to a transformation process employing the Improved Rat Optimizer (IRO) method, which aims to achieve a balanced and standardized data representation. Following this, the feature selection process utilizes the Opposition Heuristic Learning (OHL), which mimics the adaptive behaviors exhibited by rats to systematically determine the most significant features for classification purposes. The stability of the selected features is ensured by evaluating their fittest value. In conclusion, the suggested model for binary classification is a Long Short Term Convolutional Network Model (LSTCNM), which utilizes a two-dimensional array structure to retain input features and handle intricate layer processing efficiently. The purpose of this model is to distinguish between typical and unusual network traffic. The suggested framework achieves 2.5% false positive rate, 97.24% detection rate and 95.20% accuracy when trained and assessed using Netflow-based datasets.

Keywords- prediction, intrusion detection, deep learning, accuracy, optimizer

1. INTRODUCTION

Recent developments in web technologies have significantly impacted the modern age. In recent times, integrating the Internet of Things (IoT) has become essential. According to Ericsson's research [1], many devices and machines are anticipated to be interconnected, facilitating seamless communication and automation. There will be more than 75 trillion interconnected devices, according to research released by [2], which might cause a major financial disaster due to the IoT. These reports show the IoT's reach in a practical setting. Multiple PCs with different channels for sharing data and resources make up a PC. Hubs

are the devices found within PCs [3]. An anomaly is a significant problem in secured networks and other interconnected devices. Hackers develop new ways to steal private data or carry out other illegal activities that can impact the lives of IoT users. Security is essential for IoT users; security safeguards connected devices while efficiently detecting and identifying unauthorized users attempting to access the network. Secure communication with Internet technologies has been implemented in several IDFs. To stop malicious behavior, the performance of the related devices is closely watched. Users are alerted via a warning from the linked devices if any harmful activity is detected [4]. IoT technological breakthroughs have undergone various iterations to make them more portable and smaller. It has recently gained popularity in isolated areas. Despite the IoT environment's constraints in computing power, including limited size and battery capacity, its adoption remains in the early stages of development and continues to expand. To safeguard the communication channel, a number of low-power protocols have been created [5]. Because network traffic fluctuates so much in real-time networks, it is more challenging to design for intruder detection and prediction. Numerous research have illustrated the ID paradigm by identifying the particular type of data and presenting different strategies to improve system efficiency and detection accuracy. The difference between observed and actual network traffic makes deploying generated IDF in real-time difficult, even when the results are improved on the designated dataset [6]. Real-time observation of similar characteristics poses a greater challenge due to the specific data requirements necessary for effective model training. Only the restricted attacks it has been taught to identify are detectable by previous IDFs. The author in [7] claims that the restricted processing power will lead to high communication and computation time requirements. It is no longer feasible to deploy IDFs in IoT devices with limited resources. It is quite difficult to build security technologies that strike a balance between performance and security.

Cloud computing is a highly efficient technology for users dependent on on-demand services. It has transformed IT infrastructure by providing cutting-edge solutions such as servers, storage, databases, networking, applications, and a wide range of computing resources. The idea of many clouds, which depend on computational resources, is modified to satisfy the client's needs [8]. The concept of multi-cloud is extensively utilized in IoT environments to tackle security and performance challenges. The advantages of improved service quality and efficient resource allocation have propelled its implementation. As a bridge between interconnected IoT devices, multi-cloud emphasizes the importance of security. This study highlights the necessity of focusing on security measures, with IDFs being designed to intelligently identify intruders within security-aware networks, leveraging the effectiveness of deep learning (DL) algorithms [9]. DL is gaining widespread recognition for addressing a diverse range of challenges. As an advanced subset of machine learning (ML), it effectively tackles the complexities of analyzing vast amounts of data, overcoming limitations associated with traditional computational methods. Designing a better IDS requires an efficient testing dataset. The non-uniform set of data attacks differs from the accessible datasets in several ways. This difficult problem motivates us to create scalable and intelligent IDS to predict normal and anomalous traffic in real-time [10].

The key findings of the study are summarized as follows:

- This study emphasizes the significance of the IDF in a multi-cloud IoT system. The most recent research on DL and ML methods is reviewed.
- Provide a straightforward preprocessing method called Improved Rat Optimizer (IRO) to use the gathered data in a special grading format. This guarantees that the data kept for various purposes is fair.
- A new feature selection method called Opposition Heuristic Learning (OHL) explores and uses the local searching process to extract important features. The rat-inspired optimizer design examines all the data to choose the best characteristics.
- Attack classes are classified using a Long Short Term Convolutional Network Model (LSTCNM). Filtering layers for regularization are incorporated into the suggested model to address the over-fitting problem.

- In comparison to the previous method, the created framework yielded the highest detection accuracy when evaluated and applied to the NF-UQ-NIDS combined dataset.

The structure of the paper is as follows: The literature is reviewed in Section 2, the suggested IRO-LSTCNM is explained in Section 3, and the findings are examined in Section 4. The overview of the work is given in section 5.

2. Related works

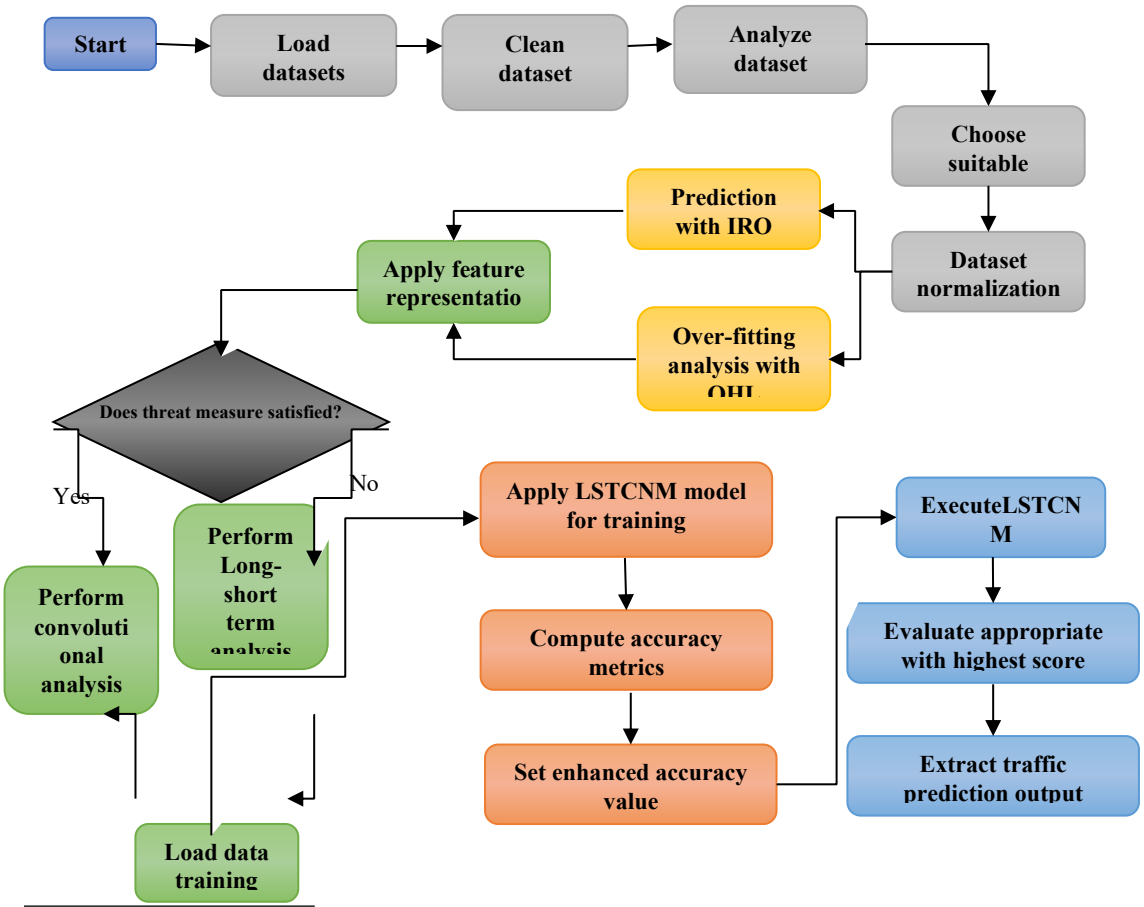
The ML and DL techniques examined in the IDF are covered in this section in order to identify the research gaps. The author of [11] talked about the issues with IDS, looking at risks such targeted attacks, identity theft, and data exfiltration. To examine the characteristics of the attacks, a comparison study is performed. However, the latest attacks were not looked into. The methods and solutions for security in IoT networks were thoroughly examined [12]. To meet the security requirements, the shortcomings of the ML and DL algorithms were investigated. The scope of IoT security research has been highlighted as well. Comprehensive privacy and security solutions for creating heterogeneous devices centered on the Edge Computing (EC) idea have been described by the author of [13]. It was demonstrated how to use ML and DL approaches to applications with an emphasis on EC. In [14], the author looked at the expansion of intrusions and their extent in fog computing using IoT. Various approaches to addressing intrusions in IoT architecture were examined [15]. Multiple designs, detection models, and preventive models have been introduced to enhance detection efficiency. The types of architectural prototypes associated with IoT security were the main topic of the review study.

In order to precisely detect Distributed Denial of Service (DDoS) attacks, the author of [16] investigated the application of distributed IDFs in fog computing. AI and fog nodes were combined to detect the attacks. IoT data has been balanced in the architecture of the IPFS-based data storage paradigm. Relying on the computation of reciprocal information gain yielded an impressive outcome. To detect routing attacks in Iot networks, key-based and clustering-based techniques were created [17]. The detection performance of such attacks has been greatly enhanced by the use of a cluster-based approach [18]. The k-mean technique and naïve Bayes classification were demonstrated in [19] using a two-stage IDS model. During training, these methods have increased the detection accuracy of low-scale computing demands. The Sybil attack was detected using a lightweight IDS model [20]. An advanced three-layer IoT security framework [21] that employs attack classification determined by observed outcomes, activity monitoring module assignment, and threat detection based on connected device kinds and their behaviors. Recently, a software-defined AI-assisted two-phase IDS have been investigated to boost the throughput rate. First, bat methods were used to construct the common traits. To determine the classes, the weights of the data samples have been estimated using random forest classifiers. The characteristics of attackers were also carefully examined in [22]. To avoid real-time IDS, honeypot modules were examined in [23]. To identify malicious activity early, the author has created a simulation tool. It cannot, however, be used to detect heterogeneous devices. On CIC IDS 2018 datasets, statistical and random sampling methods [24] were used to solve the class imbalance issue. For IIoT unbalanced datasets, the XGBoost model was applied in a similar manner. The author has set up a testbed to record data packets in order to reduce the dangers in smart homes [25]. The power consumption of the data packets was minimized. An ensemble classifier-based ID model was introduced to examine the effectiveness of voting [26], logistic regression [27], decision trees [28], and naïve Bayes [29]. The new occurrences were not correctly detected despite improving detection accuracy.

3. Methodology

The feature selection and categorization stages are improvised in this study to produce a creative and perceptive Intrusion Detection Method (IDM). Multiple cloud deployment can be done in two ways: multi-cloud and federated. Although it establishes an agreement between the providers, the federated cloud does not ensure that customers can access multiple clouds. Whatever the case, multi-cloud fosters dynamic cooperation between providers based on user needs. Additionally, users are conscious of how resources are being used. The utilization of several clouds for IoT security services that users and companies can use at any time is the subject of this study. The communication status between the IoT services and multi-cloud users is shown in Figure 1. Conventional cloud architecture may have an impact on IoT service consumers in a multi-cloud environment. Equitable industry and user collaboration is the aim of a multi-cloud system. The necessary services' administration is still being developed. Using several clouds raises security concerns in addition to service management. IoT users and services now require dynamic synchronization, which is impacted by traditional network security solutions. It causes issues like data unavailability, transmission overhead, and illegal access. Suppose a cloud outside the nation is used to instantiate a service. Since cloud-IoT consumers are unaware of the kind of network service they have been assigned, they will depend on the provider to check the service modeling. Without consumers' authorization, sharing information about dangerous or faulty services will negatively impact security features like data modification, privacy, and authentication.

Fig 1: Proposed flow diagram



3.1. Pre-processing

The first step in preparing the gathered records for use in subsequent data analysis is preprocessing. Some records, such as IP addresses and flow duration, contain inaccurate information that is eliminated immediately. A new method called grading normalization is used to normalize each feature. The proposed normalization like earlier normalizing techniques like min-max and z-score, yields well-structured data with a 0–1 range. The following is how it is expressed:

$$\text{Normalization} = \frac{(|A|)-\left(10^{d-1}\right)*(|P|)}{10^{d-1}} \quad (1)$$

Here, P is the first digit in element A; d is the number of digits in element A; A is the selected data element, and GN is a normalized value between 0 and 1. Given the enormous volume of collected information, the suggested preprocessing method will use all data items, regardless of size or volume.

3.2. Feature selection

The second step in choosing the optimal characteristics to create an effective classifier is to create feature assortments. The idea is to carefully examine every feature to use the extractor module to produce the best features. To handle complicated design problems, the Improved Rat Optimizer (IRO) has gained popularity recently. The IRO algorithm is enhanced to create an adaptive feature selection method for intrusion detection. The Opposition Heuristic Learning (OHL) concept is integrated with the IRO algorithm to improve the feature selection module's performance and efficacy. Generally speaking, Starting with a random beginning data set, IRO is a population-oriented optimization technique that modifies the answers to optimize them. Eq. (2) to Eq. (4) expresses the rats' starting position.

$$R_i = r_{i\text{-}min} + rand*(r_{i\text{-}max} - r_{i\text{-}min});\ \ i=1,2,\ldots,N \quad (2)$$

$$\overrightarrow{pos}\ (r+1) = |\overrightarrow{pos}\ (r) - \overrightarrow{pos}| \quad (3)$$

$$\overrightarrow{pos} = M*\overrightarrow{pos}\ (r) + N\ (\overrightarrow{pos}\ (r) - \overrightarrow{pos}\ (r)) \quad (4)$$

The lower limits indicate the lowest set of iterations, while the higher limits indicate the maximum set. The top search agents know the bait's location, and rats typically follow its instructions. The local search agent is used; however, it has the disadvantage of causing complicated issues and getting stuck in local optima. Certain agents become stagnant throughout the local search phase for several iterations because they depend on the local minimum. Based on its top search agents, the search agent modifies its position. Eq. (3) is used to estimate the rat's updated position. The following is the expression for the parameters M and N about the above. Here, $\overrightarrow{pos}\ (r+1)\rightarrow$ updated the position of i^{th} rat; $\overrightarrow{pos}\ (r)\rightarrow$ evaluated the optimal solution.

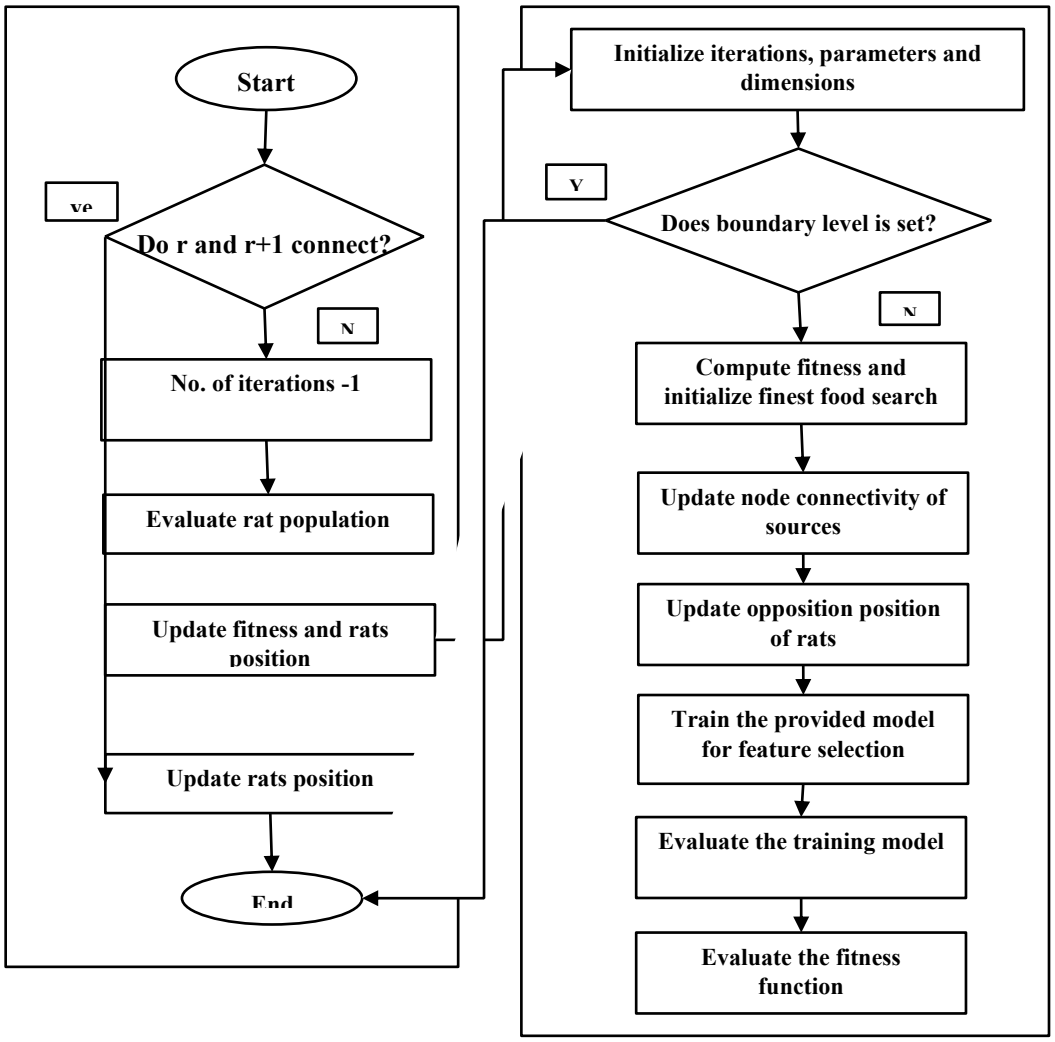
$$M = rand - j*\left(\frac{rand}{\quad}\right)\ j=1,2,3,\ldots, \quad (5)$$

$$N = 2*rand \quad (6)$$

Here, N is a random number between 0 and 2. j→ Present iteration usage; rand→ Considering a random number between 1 and 5; → Use as many iterations as possible to complete the assignment. An objective function evaluates the set of random functions. Based on rat behavior, it has been enhanced. A global optimizer is developed for the best local search process in order to estimate global features and enhance convergence speed performance. The OHL method is also used to determine the rat's opposite location, which is represented as follows:

$$\overrightarrow{pos_i}=\left(r^{imax}-r^{imin}\right)-r^i i=1,2,\ldots,n\text{-dimensional} \quad (7)$$

Fig 2. Flow diagram



The rat's locations at $\overrightarrow{pos}_i$ and pos_i are estimated using the objective function f(.). The final solution that takes into account the subset of features is evaluated using the fitness function. Feature count, False Positive Rate (FPR) and True Positive Rate (TPR) are taken into account. The number of characteristics shows the caliber of the TPR and FPR solutions. The expression for the fitness function is:

$$Fitness_r = w_1 * \frac{\text{chosen fea FPR}}{\text{No. offeat}} + w_2 * FPR + w_3 * \frac{1}{TPR} \quad (8)$$

Where, $w_1 = 0.1$, w_2 and $w_3 = 0.45$, and weight W = 1. If the $f(pos_i)$ is greater than the $f(pos_i)$, the $\overrightarrow{pos}_i$ will replace the search agent at pos_i. In certain situations, applying Eq. (9) to replace the worst answer with the new one yields the biggest optimal value.

$$f(r_{worst}) = \{rand_1 * \overrightarrow{pos}_i (r) \text{ if } rand_3 < 0.5 \; (r^{i_{max}} - r^{i_{min}}) \text{ if } rand_3 > 0.5 \quad (9)$$

The objective function's maximum value is stored in r $r_{worst}\rightarrow$; the random numbers from 0 to 1 are $rand_1$, $rand_2$, and $rand_3$.

3.3. Prediction

DL algorithms have become increasingly common in industrial systems in the past few years because of an increasing amount of data. The DNN's multiple abstraction layers send the input data to the output layer. The feature selection engineering skill-learning process is unaffected. Considering the probability of inaccuracy based on many parameters, several statistical techniques can be employed to assess the accuracy of a categorization. DL is concerned with DNs that use multi-layer unsupervised learning to complete the classification goal in hierarchical networks. DL networks can differ based on the DL approach, ID technique, and technology. The CNNs and LSTMs have been utilized as DL models in this research. Each DL technique employed accuracy and loss metrics to compute these models' outputs. Furthermore, the performance evaluation measures employed were precision, F1-score, recall, and accuracy. Figure 2 shows a flowchart of the proposed DL models for the binary and multi-class classification procedures. The UNSW-NB15 and X-IIoTID datasets were cleaned as the initial step in the data preprocessing layer.

The min-max approach was then used to normalize the datasets. The datasets were then separated into sets for testing, validation, and training. Both CNN and LSTM classifiers were built utilizing the normalized dataset in the DL classifier step. Both binary and multi-class classifiers were trained using DL models. Recall, precision, accuracy, F1-Score, and loss for multi-class and binary classifications corresponding to every model generated were used to evaluate the suggested IDS based on DL techniques as the evaluation process ended. The CNN is a widely used multi-layer ANN technology for ID. CNNs employ convolution, a mathematical technique. Convolution is one particular kind of linear operation. Neural networks that use convolution on at least one layer rather than traditional matrix multiplication are known as convolutional networks. CNNs are composed mostly of three layers: convolution, pooling, and fully connected. Complex attributes can be automatically extracted by convolutional neural networks. A more complex attribute representation is produced by the convolution layer. Eq. (10) illustrates how the convolution operation is presented. The convolution layer a's attribute 'i' map is represented by the x_i^a that i showed here. The activation function is denoted by the Ø, K_i represents layer (a-1)'s input attribute set. Layer (a-1)'s characteristics i and j and the convolution layer a connection weight are defined by W_{ij}^a. Here, b_j^a denotes the relevant layer's deviation.

$$x_i^a = \emptyset \left[\sum_{i \in k_i} x_j^{a-1} * w_{ji}^a + b_j^a \right] \quad (10)$$

The pooling layer comes after the convolution layer. The pooling layer's goal is to make the attribute map smaller. This process guarantees that important features are identified, lowers the complexity of the data, and strengthens the network's resistance to changes in the environment. Eq. (11) illustrates how to present the pooling layer.

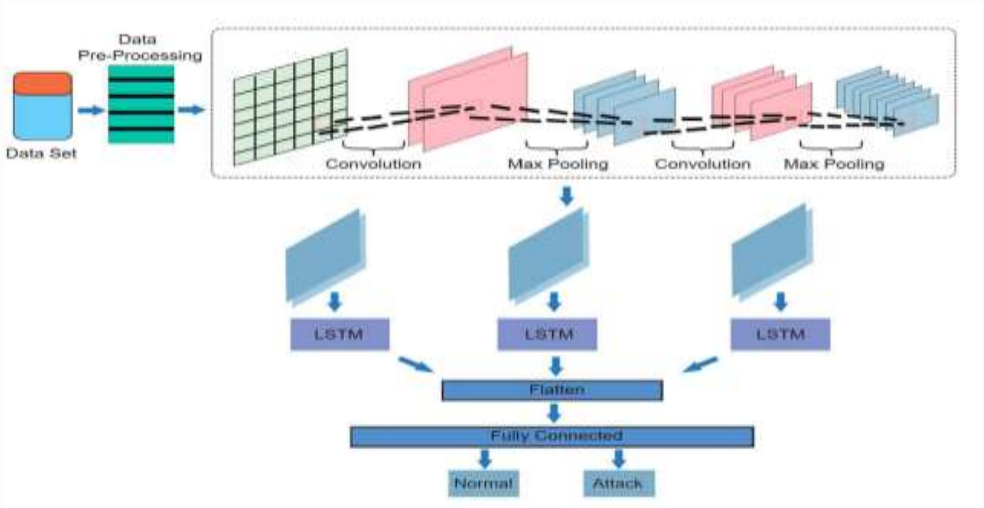
$$x_i^a = \emptyset \left[\beta_i^a c(x_i^{a-1} + b_i^a \right] \quad (11)$$

Here, b represents the weighting matrix, and c represents the sub-sampling function. After the pooling and convolution layers, the fully connected layer completes the classification process. The output function of the completely connected layer is shown in Eq. (12).

$$y^m = \emptyset \left[w^m x^{m-1} + b^m \right. \quad (12)$$

This example uses m to represent the layer index, y^m to represent the fully connected layer's output, x^{m-1} to represent its input, b^m deviation, and w^m to represent the weighting coefficient. With recurrent connections, LSTM cells replace the recurrent networks' hidden units. In the LSTM block, x^t in time step t represents the input vector, h_{t-1} in time step (t-1), and c_{t-1} in time step (t-1) represents the state of the memory cell. These make up the block's inputs. Forget output and Input gates are all part of the LSTM. The following equations represent the calculations for the LSTM's cell state, output, forget, and input gates. The forget gate $f_i^{(t)}$ uses a sigmoid activation function (r) in Eqs. (13) to (17) in determining which information can pass through or not for cell i at time step t.

Fig 3. Proposed network architecture



$$f_i^{(t)} = \sigma\left(b_i^f + \sum \; jZ_{i,j}^f x_j^{(t)} + \sum \; jD_{i,j}^f h_j^{(t-1)} \right) \quad (13)$$

$$n_i^{(t)} = f_i^{(t)} n_i^{(t-1)} + p_i^{(t)} \sigma \left(jZ_{i,j} x_j^{(t)} + \sum \; jD_{i,j} h_j^{(t-1)} \right)_{(t-1)} \quad (14)$$

$$p_i^{(t)} = \sigma(b_i^p + \sum \; jZ_{i,j}^p x_j^{(t)} + \sum \; jD_{i,j}^p h_j \;) \quad (15)$$

$$h_i^t = \tanh \tanh \left(n_i^{(t)} \right) s_i^{(t)} \quad (16)$$

$$s_i^{(t)} = \sigma(b_i^0 + \sum \; jZ_{i,j}^0 x_j^{(t)} + \sum \; jD_{i,j}^0 h_j^{t-1}) \quad (17)$$

The forget gates' respective input weight, recurrent weight, and deviation are denoted by b^f, Z^f, and D^f. Eq. (14) displays the current state of cell $n_i^{(t)}$, while b, Z, and D stand for the recurrent weights, the deviation, and the input weight moving into the LSTM cell, respectively. The input gate calculation for the cell $p_i^{(t)}$ is shown in Eq. (15) and is comparable to the forget gate calculation. The hidden state is denoted by $h_i^{(t)}$ in Eq. (16), and $s_i^{(t)}$ denotes the output gate. While b^0, Z^0, and D^0 stand for the deviation, input weight, and recurring weights, respectively, Eq. (17) illustrates the output gate equation.

4. Numerical results

The NF-UQ-NIDS data sets, which comprise four distinct databases, are subjected to the suggested IDF. NF-CSE CI-CIDS2018, NF-ToNIoT, NF-BoTIoT and NF-UNSWNB15 are particularly among them. The obtained raw packet data was transformed into the NetFlow format by earlier researchers. The issues brought on by different features on different data kinds have been fixed by the NetFlow format by combining a similar feature set. The datasets have higher scalable qualities since they are practically appropriate in various contexts. Table 1 displays the categories of all attacks, and the various dataset versions have been appropriately covered. By removing characteristics like IP addresses and the initial flow time, the biased section of the record is removed in the current study.

Table 1. Dataset characteristics

Datasets	Data volume	Training data samples	Testing data samples
BoT-IoT	600100	480080	120020
ToNIoT	1379274	1103419	275855
UNSWNB15	1623118	1298494	324624
CICIDS 2018	8392401	6713920	1678481

MATLAB 2020a is used for the implementation and testing of the suggested work. Math libraries make up this collection. It is also employed in classification. The performance of the suggested IRO with the LSTCNM method is seen in Table 2. Utilizing the collected data, an optimized preprocessing technique is

applied. The suggested preprocessing method can be used for a lot of data to grade individual elements. The preprocessing procedures are applied to each piece of data, and each data point is given a grade between 0 and 1 to ensure fairness. It has been found that the time required to conclude increases with the percentage of unstructured data. For ease of interpretation, the suggested IRO now uses conventional round-of values. To our knowledge, an optimization problem is used to simulate the Netfow datasets. However, the parameters are displayed in Table 3, and a comparison is made between standard RSO and the suggested IRO with the LSTCNM approaches.

Table 2. Pre-processing with dataset samples

Original data samples	BoTIoT	ToNIoT	UNSWNB15	CICIDS2018
1	0.799	0.631	0.630	0.510
2	0.490	0.575	0.323	0.445
3	0.346	0.575	0.211	0.122
4	0.799	0.139	0.239	0.611
5	0.799	0.520	0.645	0.445
6	0	0.540	0.574	0.550
7	0.366	0.605	0.415	0.445
8	0.799	0.500	0.293	0.630
9	0.799	0.360	0.180	0.490
10	0.509	0	0.200	0.510

Table 3. Feature selection

Parameters	Description
No. of rats	Count available solutions
Rats' position	Selected features
Finest rat	Optimal value
Opposition position	Best rat based change
Fitness function	FPR and TPR features
	Iterations

Fig 4. Pre-processing with dataset samples

The suggested characteristics of their assessed fitness solutions are shown in Table 3 and Figures 4 and 5. The findings indicate that the suggested IRO with the LSTCNM has produced the best outcomes when compared to the standard IRO. After 30 to 40 iterations, the approach aims to reduce the solution's goals, producing a high-quality response. Predicting the opposite location improves the replies, and the unreliability in local search is eliminated. The DL model and the selection of features have been examined and learned from many correlations between the normalized data. The chosen features significantly advance our understanding of ID capabilities. This study's shortcoming is that multi-class systems take more work despite IRO with the LSTCNM superior performance in producing the most significant features. Since not all features are required to construct an IDF, the IRO with the LSTCNM approach chooses the significant features. This procedure is crucial for removing features requiring much computing time and effort. The selected features are then sent into the 2D-CNN classifier when the feature selection procedure is finished. Predicting both normal and abnormal traffic kinds is the goal of this classifier. The suggested classifiers are validated using several measures. Here, the experimental procedure is carried out using a straightforward confusion matrix. Four labels make up the confusion matrix:

a) False Positive (FP): When a typical traffic count is seen as abnormal.
b) False Negative (FN): It is assumed that the unusual traffic volume is typical.

c) True Negative (TN): the number of common traffic classifications that were accurately identified. In regard to it, we examine the following performance metrics:

d) True Positive (TP): the quantity of unusual traffic classes correctly identified.

a) Detection rate or recall: It outlines which strategies for attack are optimal for each attack class. The following is how it is stated:

$$\text{TPR (detection rate)} = \frac{TP}{TP+FN} \tag{18}$$

b) FPR (false positive rate): Attack classes' suitable countermeasures are determined by the false positive rate (FPR). This is how it is stated:

$$FPR = \frac{FP}{TN+FP} \tag{19}$$

c) Classification accuracy: It determines how many attack classes are accurately classified based on the total number of classes. It is said as follows:

$$\text{Accuracy} = \frac{TP+TN}{TN+TN+FP+FN} \tag{20}$$

d) F-measure: This determines the model's classification accuracy, with particular attention paid to parameter recall and precision. This is how it is expressed:

$$\text{F1-measure} = \frac{2*TP}{2*TN+FP+FN} \tag{21}$$

Table 4. Dataset-based TPR, FPR, accuracy, and F1-score comparison

Datasets	TPR	FPR	Accuracy	F1-score
BoT-IoT	80.12	55	95.9	94.5
ToNIoT	85.6	84	99.3	98.4
UNSWNB15	70.7	56	89.2	89.5
CICIDS2018	61.3	38	91.5	86.1

Table 5. Performance analysis with existing approaches

Performance (%)	IRO with the LSTCNM	LSTM	2D-CNN	CNN
Accuracy	96	93	89	75
FPR	1.5	2.6	5.6	8.9
Detection rate	98.5	90	88	76

Fig 4. Performance analysis with existing approaches

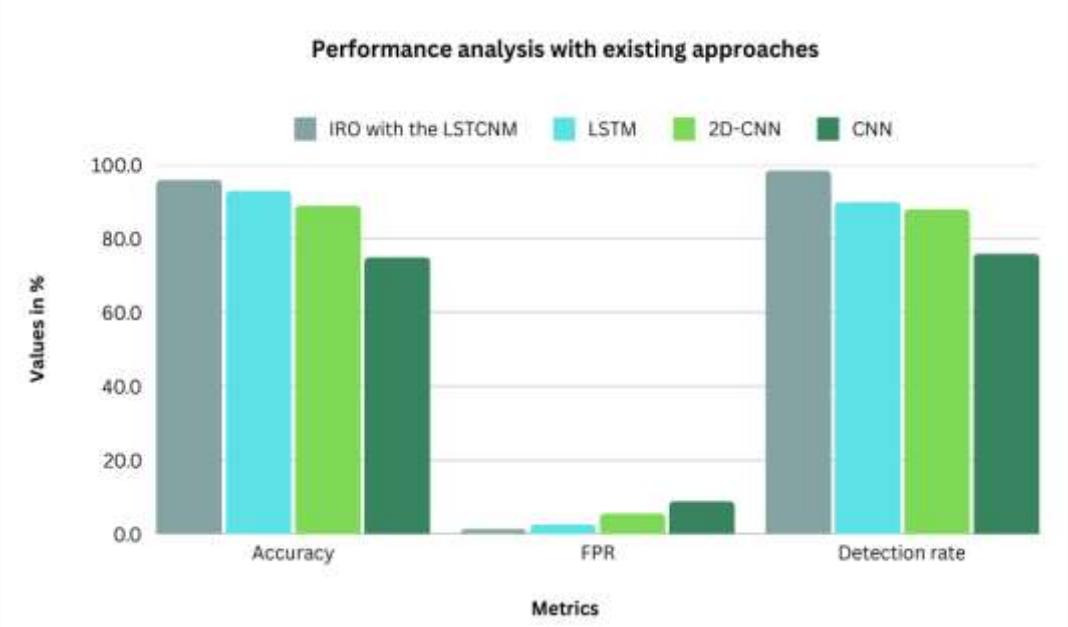


Fig 5. TPR, FPR, F1-score and accuracy comparison with various benchmark datasets

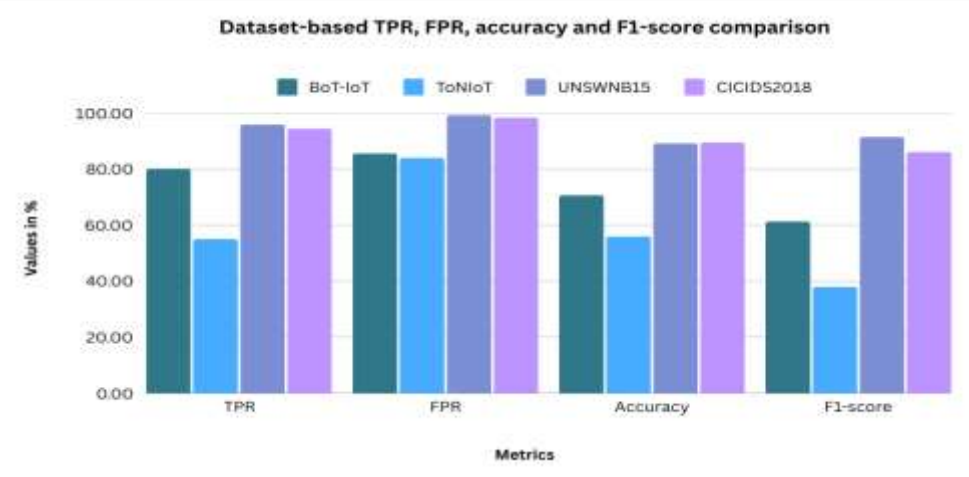


Table 4 and Table 5 show the suggested classifier's performance analysis. The suggested IRO with the LSTCNM design is to provide a reliable IDM without compromising the FPR, detection rate, or accuracy. The initial difficulty is to develop a set of attack signatures for the best feature selection process. While the 2D-CNN technique lacks attack data, it is beneficial to employ IRO with the LSTCNM to generate a significant amount of training data with its attack kinds. Even when the ensuing attacks are not real attacks, heterogeneity stops complicated attacks from emerging. Multi-cloud-IOT is even more diverse due to its complicated topology expansion and strong network interaction across traffic components. The suggested outcome has improved TPR and accuracy in a short time.

5. Conclusion

This project is an interesting and innovative attempt to use a swarm-based DL classifier to improve the IDM. The suggested IRO with the LSTCNM is divided into three stages to identify risks centered on the network and application layers. The IRO with the LSTCNM technique scales the data easily and is used to preprocess the collected data to guarantee fairness. The OBL-RIO feature selection approach investigates the local search of all features to extract the most important characteristics. In conclusion, an IRO with the LSTCNM is the recommended binary classifier. The input features are preserved by the IRO with the LSTCNM to work on the complex layers. Regular traffic and intruders can be distinguished by it. The proposed methodology is trained on and evaluated using NetFlow-based datasets. The proposed framework has a detection rate of 97.5% percent, an FPR of 2.5 percent, and an accuracy of 96%.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, ‘‘Deep learning,’’ Nature, vol. 521, no. 7553, pp. 436–444, May 2015.
[2] Reddy, Y. Ramadevi, and K. V. N. Sunitha, ‘‘Effective discriminant function for intrusion detection using SVM,’’ in Proc. Int. Conf. Adv. Comput., Commun. Inform. (ICAC), Sep. 2016, pp. 1148–1153.
[3] Ingre and A. Yadav, ‘‘Performance analysis of NSL-KDD dataset using ANN,’’ in Proc. Int. Conf. Signal Process. Commun. Eng. Syst., Jan. 2015, pp. 92–96.
[4] Farnaaz and M. A. Jabbar, ‘‘Random forest modelling for network intrusion detection system,’’ Procedia Comput. Sci., vol. 89, pp. 213–217, Jan. 2016.

[5] Khan and N. Jain, ‘‘A survey on intrusion detection systems and classification techniques,’’ Int. J. Sci. Res. Sci., Eng. Technol., vol. 2, no. 5, pp. 202–208, 2016
[6] Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software-defined networking," Proc. Int. Conf. Wireless Netw. Mobile Commun. (WINCOM), Oct. 2016, pp. 258–263.
[7] Ashfaq, X.-Z. Wang, J. Z. Huang, H. Abbas, and Y.-L. He, "Fuzziness based semi-supervised learning approach for an intrusion detection system," Inf. Sci., vol. 378, pp. 484–497, Feb. 2017.
[8] Ashfaq, X.-Z. Wang, J. Z. Huang, H. Abbas, and Y.-L. He, "Fuzziness based semi-supervised learning approach for an intrusion detection system," Inf. Sci., vol. 378, pp. 484–497, Feb. 20
[9] Chang, W. Li, and Z. Yang, "Network intrusion detection based on random forest and support vector machine," Proc. IEEE Int. Conf. Comput. Sci. Eng./IEEE Int. Conf. Embedded Ubiquitous Comput., Jul. 2017, pp. 635–638
[10] Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, “Deep learning and its applications to machine health monitoring: A survey,” Submitted to IEEE Trans. Neural Netw. Learn. Syst., 2016. [Online]. Available: http://arxiv.org/abs/1612.07640
[11] Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” J. Mach. Learn. Res., vol. 11, pp. 3371–3408, 2010.
[12] In Proc, you, Y. Li, Y. Wang, J. Zhang, and Y. Yang, "A deep learning-based RNNs model for automatic security audit of short messages," in Proc. 16th Int. Symp. Commun. Inf. Technol., Qingdao, China, Sep. 2016, pp. 225–229.
[13] Alrawashdeh and C. Purdy, “Toward an online anomaly intrusion detection system based on deep learning,” in Proc. 15th IEEE Int. Conf. Mach. Learn. Appl., Anaheim, CA, USA, Dec. 2016, pp. 195–200
[14] Potluri and C. Diedrich, "Accelerated deep neural networks for an enhanced intrusion detection system," Proc. IEEE 21st Int. Conf. Emerg. Technol. Factory Autom., Berlin, Germany, Sep. 2016, pp. 1–8.
[15] Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software-defined networking," Proc. Int. Conf. Wireless Netw. Mobile Commun., Oct. 2016, pp. 258–263
[16] Hodo, X. J. A. Bellekens, A. Hamilton, C. Tachtatzis, and R. C. Atkinson, Shallow and deep networks intrusion detection system: A taxonomy and survey, Submitted to ACM Survey, 2017, [Online]. Available: http://arxiv.org/abs/1701.02145
[17] Kim, G., Yi, H., Lee, J., Paek, Y., Yoon, S.: Lstm-based system-call language modelling and robust ensemble method for designing host-based intrusion detection systems. arXiv preprint arXiv:1611.01726 (2016)
[18] Buczak, A.L., Guven, E.: A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications Surveys & Tutorials 18(2) (2016) 1153–1176
[19] Javaid, A., Niyaz, Q., Sun, W., Alam, M.: A deep learning approach for network intrusion detection system. In: Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), New York, NY, USA. Volume 35. (2015) 2126
[17] Safa Otoum, Burak Kantarci, and Hussein T. Mouftah, “Adaptively supervised and intrusion-aware data aggregation for wireless sensor clusters in critical infrastructures,” in 2018 IEEE International Conference on Communications (ICC), May 2018, pp. 1–6
[18] Arnaldo Gouveia and Miguel Correia, A Systematic Approach for the Application of Restricted Boltzmann Machines in Network Intrusion Detection, vol. 10305, 05 2017.
[19] Beigh and M. A. Peer, “Performance evaluation of different intrusion detection system: An empirical approach,” in Intl Conf. on Computer Communication and Informatics, Jan 2014, pp. 1–7.

[20] Zhou W., Wen J., Koh Y., Xiong Q., Gao M., Dobbie G., Alam S. (2015), “Shilling Attacks Detection in Recommender Systems Based on Target Item Analysis” PLoS One, July, 10(7), p.e0130968
[21] Kumari T., Bedi P. (2017), “A Comprehensive Study of Shilling Attacks in Recommender Systems”, IJCSI International Journal of Computer Science Issues, 14(4), 44-50
[22] Cao, J., Wu, Z., Mao, B. & Zhang, Y (2013). “Shilling attack detection utilizing semi-supervised learning method for attack detection utilizing semi-supervised learning method for collaborative recommender system”. World Wide Web Jornal, 16(5-6): 729-748.
[23] Yu H, Gao R, Wang K, Zhang F (2017), “A novel robust recommendation method based on kernel matrix factorization”. J Intell Fuzzy Syst 32(3):2101–2109
[24] Yang Z., Cai Z. (2017), “Detecting abnormal profiles in collaborative filtering recommender systems”. Journal of Intelligent Information Systems, 48(3), 499-518
[25] Zhou W., Wen J., Qu Q., Zeng J., Cheng T. (2018), “Shilling attack detection for recommender systems based on credibility of group users and rating time series”, PLoS One, May, 13(5), p.e0196533
[26] Turk A., Bilge A., (2019). “Robustness analysis of multi-criteria collaborative filtering algorithms against shilling attacks”, Expert Systems with Applications, 115, p.386-402
[27] Moradi P., Ahmadian S., (2015), “A reliability-based recommendation method to improve trust-aware recommender systems”, Expert Systems with Applications, 42, 7386-7389.
[28] Paradarami, N.D. Bastian, J.L. Wightman, “A Hybrid recommender system using artificial neural networks”, Expert Systems with Applications, Vol. 83, (2017), 300-313.
[29] Agar ap, "A neural network architecture combines gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data," Proc. 10th Int. Conf. Mach. Learn. Comput., Feb. 2018, pp. 26–30.
[30] Around, M.-A. El Hussaini, A. El Hore, and J. Ben-Othman, "Real-time detection of MAC layer misbehaviour in mobile ad hoc networks,’’ Appl. Comput. Information., vol. 13, no. 1, pp. 1–9, 2017.