1-1-2022

A Novel Tunicate Swarm Algorithm with Hybrid Deep Learning Enabled Attack Detection for Secure IoT Environment

Fatma Taher
Zayed University

Mohamed Elhoseny
University of Sharjah

Mohammed k. Hassan

Ibrahim M. El-Hasnony
Mansoura University

IEEE Access

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS. 2017.Doi Number

A Novel Tunicate Swarm Algorithm with Hybrid Deep Learning Enabled Attack Detection for Secure IoT Environment

Fatma Taher[1], Mohamed Elhoseny[2,3], Mohammed k. Hassan[4], and Ibrahim M. El-Hasnony[3]

[1] College of Technological Innovation, Zayed University, Dubai, UAE
[2] College of Computing and Informatics, University of Sharjah, UAE
[3] Faculty of Computers and Information, Mansoura University, Egypt
[4] Mechatronics department, Faculty of Engineering, Horus university in Egypt (HUE)

Corresponding author: Ibrahim M. El-Hasnony (ibrahimhesin2005@mans.edu.eg).

ABSTRACT The Internet of Things (IoT) paradigm has matured and expanded rapidly across many disciplines. Despite these advancements, IoT networks continue to face an increasing security threat as a result of the constant and rapid changes in the network environment. In order to address these vulnerabilities, the Fog system is equipped with a robust environment that provides additional tools to beef up data security. However, numerous attacks are persistently evolving in IoT and fog environments as a result of the development of several breaches. To improve the efficiency of intrusion detection in the Internet of Things (IoT), this research introduced a novel tunicate swarm algorithm that combines a long-short-term memory-recurrent neural network. The presented model accomplishes this goal by first undergoing data pre-processing to transform the input data into a usable format. Additionally, attacks in the IoT ecosystem can be identified using a model built on long-short-term memory recurrent neural networks. There is a strong correlation between the number of parameters and the model's capability and complexity in ANN models. It is critical to keep track of the number of parameters in each model layer to avoid over- or under-fitting. One way to prevent this from happening is to modify the number of layers in your data structure. The tunicate swarm algorithm is used to fine-tune the hyper-parameter values in the Long Short-Term Memory-Recurrent Neural Network model to improve how well it can find things. TSA was used to solve several problems that couldn't be solved with traditional optimization methods. It also improved performance and shortened the time it took for the algorithm to converge. A series of tests were done on benchmark datasets. Compared to related models, the proposed TSA-LSTMRNN model achieved 92.67, 87.11, and 98.73 for accuracy, recall, and precision, respectively, which indicate the superiority of the proposed model.

INDEX TERMS Data security, Deep learning, Fog computing, Internet of Things, Intrusion detection, Tunicate swarm algorithm.

I. INTRODUCTION

Exponential growth in the standard utilization of electronic services and applications has prompted gigantic advances in broadcast communications organizations and the development of the idea of the Internet of Things (IoT). An IoT is arising in an interchangeable worldview in which gadgets fill in as items or "things" that can detect their current circumstances, interface with one another, and trade information on the Internet [1]. The Internet of Things (IoT) worldview has recently been used in establishing smart conditions, for example, brilliant urban communities and smart homes with various application spaces and related administrations [2]. The objective of creating such brilliant conditions is to make human existence more useful and agreeable by tackling difficulties connected with the living climate, energy utilization, and modern necessities [3]. It can be seen objectively in the significant growth of available IoT-based administrations and applications across various organizations. As security will be a fundamental supporting component of most IoT applications, IoT intrusion identification frameworks will also be created to support the interchanges powered by such IoT advances. Fig. 1 demonstrates the types of IoT security attacks.

Considering the security estimates, it is fundamental to foster a security instrument for an IoT environment. An information-arrangement security system should be engaged to forestall unapproved access to information sources from malignant clients. It is fundamental to center on information secrecy and honesty, which significantly decreases the genuine security dangers in an IoT environment. Ordinary security instruments are created in light of cryptographic strategies, and they aren't broadly embraced for the IoT climate because of the enormous volume of information. The dangers should be related to in a base time frame that will decrease the issues in the organization, and traditional security models require more opportunity to handle such an enormous measure of information to recognize the dangers [4]. Unapproved admittance to information for a brief period is adequate for a malignant client to acquire secret information, and the change of such information has an enormous effect on the client. So it is fundamental to recognize an interloper in an IoT organization. Implementing IDS is important [2]. Intrusion detection frameworks identify gatecrashers and secure the organization and information by preventing unapproved clients from entering. With the restriction of fuel sources, carrying out IDS is a complicated cycle [5]. If this is too complicated, a central intrusion detection system could be used. This system, which monitors the organization as well as other hubs far away, looks for intrusions and tells you about them [6].

In recent years, advancements in artificial intelligence (AI) have been used to further develop IoT IDS. For example, AI and deep learning strategies have been used to further develop IoT IDS. The current necessity is to do a state-of-the-art, careful scientific classification and a basic survey of this new work [7]–[12]. Various related investigations applied different AI and profound learning procedures through different datasets to approve the improvement of IoT IDS. Yet, it's as yet not satisfactory which dataset, AI, or profound learning methods are more successful for building a proficient IoT IDS. In addition, some IDS strategies don't take into account how long it takes to build and test IoT IDSs, even though this is a key factor in the viability of "online" IDSs [13].

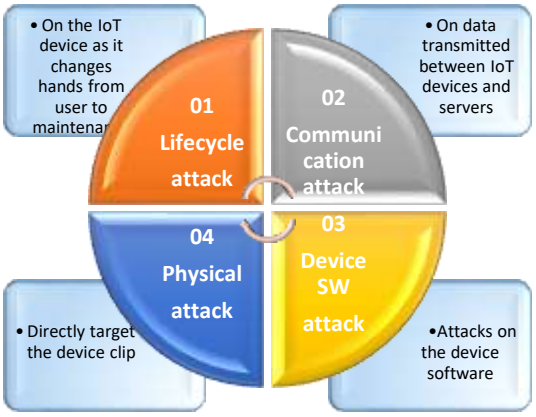


Figure 1. Types of IoT security attacks

The objective of this paper is to develop a novel tunicate swarm algorithm with a Long Short-Term Memory-Recurrent Neural Network for intrusion detection in the IoT environment. As a result of this research, we have developed a unique tunicate swarm algorithm combined with a Long Short-Term Memory-Recurrent Neural Network (TSA-LSTMRNN) for intrusion detection on the Internet of Things (IoT). The TSA-LSTMRNN model that has been given is primarily intended to detect the presence of assaults. As a result, the data preparation performed by the presented model is necessary to transform the incoming data into a format that can be used. In addition, the LSTMRNN model is used for the identification and classification of threats in the Internet of Things environment, among other things. The TSA is used to properly fine-tune the hyper-parameter values involved in the LSTMRNN model to improve the detection outcomes of the model, and this is done in order to improve the detection outcomes of the model.

Therefore, the main contributions of this paper are as follows.

- For threat detection in the IoT environment, a TSA-LSTMRNN enabled deep-learning-driven solution is proposed that is highly cost-effective and scalable.
- In IoTs, the LSTMRNN classifier is used for effective threat detection;
- For enhancing the LSTMRNN model, the TSA algorithms is used for adapting its hyper-parameters;
- The proposed mechanism is compared to existing literature works for a better performance evaluation under the used data set for verification purposes;
- Finally, 10 fold cross-validation was used in this study to demonstrate the unbiasedness of our findings;
- The results of the evaluation show that the proposed mechanism is capable of multiclass detection and

outperforms in terms of detection accuracy and computational complexity.

The rest of this paper is arranged as follows. The related work and recent literature reviews are discussed in Section 2. The LSTM and TSA algorithms are introduced in Section 3 as preliminary. In section 4, the proposed model is discussed in detail. The simulation results based on the TSA-LSTMRNN algorithm and the comparison with the other algorithms are analyzed in Section 5. Finally, the research and future work of this paper are summarized in Section 6.

II. RELATED WORK

This section introduces the recent literature on anomaly detection. Roy et al. [14] presented an IDS which employs ML to efficiently identify anomalies and cyberattacks in resource-limited IoT networks. Utilizing a set of optimizations that include dimensionality reduction, extracting multi-collinearity, and sampling, their method could recognize the vital features to identify intrusion with minimum training time and data. In [15], they proposed an IDS based on the concept of the Exact Greedy Boosting ensemble model for device application in the fog node due to precise recognition of malicious activity and protection of crucial structures. Liu et al. [16] presented a particle swarm optimization-based gradient descent (PSO-LightGBM) for the IDS. In their study, PSO-LightGBM was utilized for extracting the characteristics of the information and inputting them to a one-class SVM (OCSVM) for discovering and identifying malicious information. The UNSW-NB15 data set is employed for verifying the IDS. Tharewal et al. [17] introduced a near-end optimization technique for the IIoT-IDS-based This approach integrates DL observation ability with reinforcement learning (RL) decision-making ability to enable effective detection of various types of cyber-assault on the IIoT. De Souza et al. [18] introduced a two-step technique for identifying and detecting intrusions. The initial phase implements a traffic analysis with Extra Tree binary classification. The events recognized as intrusive are analyzed in the next phase by an ensemble method comprised of RF, DNN, and Extra Tree.

As a result, a number of forensic frameworks for the Internet of Things (IoT) have been developed [19]–[22]. ProbeIoT and FIF-IoT, two models that handle the acquisition of evidence from IoT devices in a forensically sound fashion to ensure integrity and chain of custody without infringing on users' privacy, were proposed by Hossain et al. [21] and Hossain et al. [20]. FSAIoT is a framework for collecting state data from IoT devices introduced by Meffret et al. [19]. Following that, Cebe et al. [22] built a Block4Forensic acquisition model that was specifically tailored for the collection of vehicle data. Probe-IoT, FIF-IoT, and Block4Forensic are all based on the blockchain scheme. An open, distributed public ledger was set up for Probe and FIF-IoT, while Block4Forensic relied on a fragmented ledger to decrease storage requirements for its ledger. Information regarding IoT devices and their interactions with other network entities is kept in the blockchain in all of the aforementioned cases. FSAIoT, on the other hand, makes use of local networks with centralized controllers to keep track of device states and data flows.

There is evidence that many traditional machine learning algorithms do not separate the difficult work of intrusion detection; identifying anomalous data and preprocessing the data set are both complex and time-consuming processes [21]–[23]. When used with traditional machine-learning classification models to speed up preprocessing, deep learning [24] can map features to a more multidimensional and easily distinguishable feature space because it has already learned the non-linear mixture of features from the original data set, so it can do this better.

Meanwhile, some researchers have tested their methodology using the botnet viral data set. Using feature selection in [26], it is able to reduce the number of features needed to detect IoT bots with more accuracy while still allowing for interpretable findings to be produced by a decision tree for modern intrusion detection. It is possible to identify botnet-related distributed denial of service (DDoS) attacks using a detection method based on anomalous traffic and a deep automated encoder [27], [28]. Anomaly detection can considerably improve the detection accuracy of aberrant traffic, according to experiments. According to [29], and improved deep learning method, the cloud-based LSTM model makes use of more powerful computing resources to execute tasks like anomaly detection. Each assault is treated as aberrant traffic to be separated from regular traffic and implemented several binary classification tasks, even though they take numerous network attacks into account.

As discussed, although much effort has been dedicated to solve anomaly detection in the IoT environment, the noted algorithms suffer from the following limitations and challenges.

- Because of the advancement of sensor monitoring technologies, low-cost solutions, and high impact in a variety of application domains, anomaly detection has gotten a lot of attention from the research community in the last few years.
- While monitoring physical spaces and objects, sensors generate a large amount of data. These massive data streams can be analyzed to uncover unhealthy habits. It has the potential to reduce functional risks, avoid problems that aren't visible, and prevent system downtime.
- Defining the boundaries between normal and abnormal behavior is difficult. A few anomalies are available to train models. Real-life abnormal behavior is rare compared to normal behavior. In avionics systems, we have a lot of normal flight data, but unusual data is rare. In many applications, it's difficult or impossible to generate anomalous behavior due to resource waste or system constraints.
- The majority of models used in anomaly detection are based on time series patterns observed in the wild. An anomaly is defined as anything that doesn't fit the mound. Detecting anomalies in historical data, performing real-time analyses, and forecasting

unusual behavior in an IoT environment have all been accomplished by researchers. As part of our research, it's important to look at how the Internet of Things (IoT) is creating new ways to find, analyze, and predict anomalies.

- To meet these challenges, it is important to give clear information about the methods that have been developed and how they should be used in different situations.

In this study, a novel tunicate swarm algorithm with long-term short-term memory-recurrent neural network for ID in the IoT environment was developed. The presented TSA-LSTMRNN model majorly intends to identify the presence of attacks. In order to achieve this, the presented model initially undergoes data preprocessing to transform the input data into a useful format. Besides, the LSTMRNN model is utilized for the identification and classification of attacks in the IoT environment. The TSA is applied to properly adjust the hyper-parameter values involved in the LSTMRNN model for improving the detection outcomes. A series of experimental analyses are performed on benchmark datasets.

III. PRELIMENARIES

To train with sequence data, the most popular model is the Recurrent Neural Network (RNN). When trained with a large step size, the standard RNN has issues. The vanishing problem and RNN's formalization are briefly discussed in this section. To remedy this, we'll go over something called "Long-Short Term Memory."

A. RECURRENT NEURAL NETWORK

A feed-forward neural network has been extended to include recurrent neural networks (RNNs). An RNN's cyclic connections allow it to simulate complex sequences more effectively than feedforward networks. X, H, and Y are used to signify a series of inputs, a hidden vector, and an output vector. $X = (x_1, x_2, \dots, x_T)$ provides the input sequence. Hidden vector sequence $\left(H = (h_1, h_2, \dots, h_T)\right)$ and output vector sequence $\left(Y = (y_1, y_2, \dots, y_T)\right)$ with $t = 1$ to T are calculated in a standard RNN as follows:

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \quad (1)$$

$$y_t = W_{hy}h_t + b_y \quad (2)$$

where W is a weight matrix and b is a bias component, and function σ is a nonlinearity function.

The RNN protocol for dealing with a variable-length sequence input is Back Propagation Training Time (BPTT) [30]. Models are built using data initially in BPTT. Then, each time step's output error gradient is recorded. The RNN is difficult to train, yet when using the BPTT algorithm, it causes the gradient to explode or vanish. This issue was brought up and resolved by Bengio et al. in [31].

B. LONG SHORT TERM MEMORY

A recurrent neural network classifier known as Long Short-Term Memory (LSTM) LSTM cells is shown in Fig. 2 [32]. We also go over the equations for calculating the three gates' values and the current state of the cell.

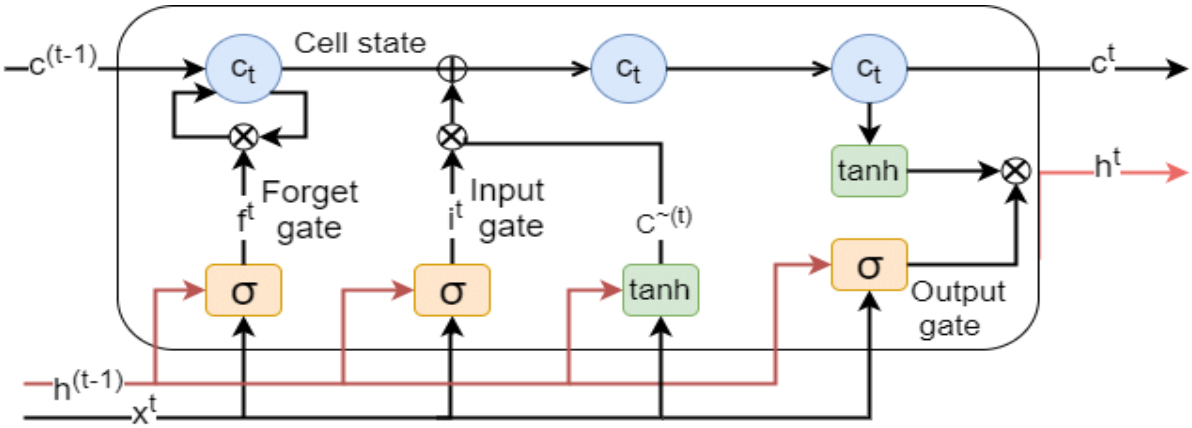


Figure 2. Long Short Term Memory Cell

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \quad (3)$$

$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right) \quad (4)$$

$$c_t = f_t c_{t-1} + i_c tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (5)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \quad (6)$$

$$h_t = o_t tanh(c_t) \quad (7)$$

f, i, c, o are the forget gate, input gate, cell state and output gate and σ refer to the logistic sigmoid function. Weight matrices for peephole interconnections are designated as W_{ci}, W_{co} and W_{cf}. The information flow in LSTM is controlled by three gates (i, f, o). The ratio of input is determined by the input gate. This ratio affects the equation used to calculate cell state (5). The prior memory h_{t-1} is either passed through the forget gate or it isn't. It is determined in the equation (3) and utilized in the equation (4) to determine the ratio of the prior memory. Passing or rejecting data from the memory cell is determined by an input gate. Eq. (6) demonstrates how this procedure works. The three gates of the LSTM allow us to address the vanishing and bursting gradient problems. The recurrent hidden layer in LSTM-RNN architecture is replaced by an LSTM cell.

C. TUNICATE SWARM ALGORITHM

Tunicates are cylindrical-shaped animals that have one of their two ends open, and they travel through the water with a

jet-like velocity [33]. It is possible for them to look for food in the sea, even if they have no prior knowledge of where to find it. The tunicates' jet-like propulsion and swarm intelligence are the basis for the TSA optimization technique. When it comes to TSA's optimization dilemma, the food supply serves as the best answer. Some conditions must be met in order to accurately depict TSA jet propulsion movements. Two things must happen before anything else can happen: First of all, the tunicates need to avoid confrontation, and second of all, they need to keep moving towards their best search agent. Finally, they need to remain near that agent. The swarm intelligence of the other tunicates in the mathematical model is used to update their positions in relation to the optimal solution. According to [29], the mathematical system is specified as follows [33]:

Condition 1: There must be no conflicts among search agents

No conflict between search agents can be avoided by employing the following vector to compute their respective positions [33][34].

$$\vec{A} = \frac{\vec{G}}{\vec{M}} \tag{8}$$

$$\vec{G} = c_2 + c_3 - \vec{F} \tag{9}$$

$$\vec{F} = 2^{*}c_1 \tag{10}$$

where the gravitational force is denoted by $\vec{G}$ and $\vec{F}$ is the change in temperature of the water flow in the deep ocean. To calculate the social forces, represented by vector $\vec{M}$, between tunicates, we use the following formula: c_3, c_2 and c_1 are random variables whose values range from 0 to 1.

$$\vec{M} = \lfloor P_{min} + c_1 * P_{max} - P_{min} \rfloor \tag{11}$$

$P_{\min}$ and $P_{\max}$ represent the initial and subordinate speeds, respectively, of social interaction. In most cases, 1 and 4 are the default numbers for them.

Condition 2: Orientation towards optimal search agent

According to [33], this phase of optimization requires ensuring that the tunicate is moved in a specific direction.

$$\overrightarrow{\boldsymbol{PD}} = \left|\overrightarrow{\boldsymbol{FS}} - \boldsymbol{r_{and}} * \overrightarrow{\boldsymbol{P_p(x)}}\right| \tag{12}$$

Where the current iteration is denoted by x, the distance between the food source and search agent is denoted by the vector $\overrightarrow{PD}$, location of search agents is represented by $\overrightarrow{P_p(x)}$, the position of food source is referred by $\overrightarrow{FS}$ and in a range from zero to one, the value of a random variable r_{and} is used.

Condition 3: moving in the direction of the best search agent:

In order to accomplish this, search agents are reordered as follows [34]:

$$\overrightarrow{P_p(x')} = \begin{cases} \overrightarrow{FS} + \vec{A} * \overrightarrow{PD}, \text{if } r_{\text{and}} \geq 0.5 \\ \overrightarrow{FS} - \vec{A} * \overrightarrow{PD}, \text{if } r_{\text{and}} \leq 0.5 \end{cases} \tag{13}$$

$\overrightarrow{P_p(x')}$ represents the search agent's current location in relation to the food supply. The first best two solutions are stored and used to change the placements of other tunicates in order to replicate swarm behavior. Mathematically, this is what a swarm looks like:

$$\overrightarrow{\boldsymbol{P_p(x+1)}} = \frac{\overrightarrow{\boldsymbol{P_p(x)}} + \overrightarrow{\boldsymbol{P_p(x+1)}}}{\boldsymbol{2+c_1}} \tag{14}$$

The important steps to demonstrate the flow of the original TSO are presented below for clarification of the TSO. The TSO algorithm's flowchart is shown in Fig. 3 [34].

1 Set the initial population of tunicates $\vec{P}_p$ to its default value.
2 Set the parameter's initial value and the maximum number of iterations.
3 Every exploration agent's fitness value should be calculated.
4 Finally, the best-fitting agent is inspected in the search space provided after evaluating its fitness.
5 Explore agents should be upgraded. It is time to put the newly enhanced agent back in his or her place of origin.
6 Calculate the fitness cost of a more advanced search agent.
7 The best answer X_{best} is stored and $\vec{P}_p$ is upgraded when the prior solution is no longer optimal.

IV. DESIGN OF TSA-LSTMRNN MODEL

In this study, a novel tunicate swarm algorithm with LSTMRNN for ID in the IoT environment is proposed. The presented TSA-LSTMRNN model applies the LSTMRNN model for the identification and classification of attacks in the IoT environment. To improve the detection outcomes of the LSTMRNN model, the TSA is applied to properly adjust the hyper-parameter values involved in it. Fig. 4 showcases the overall process of the TSA-LSTMRNN technique.

A. PREPROCESSING PHASE

The quality of the data used in data mining operations must be high in order to achieve a high level of performance at a cheap cost. Anomaly type characteristics will be converted to numeric in the preprocessing step.

1) Missing values:

Many of the variables in most datasets are missing, necessitating the handling of missing values in order to improve accuracy. The mode method is used to replace the empty value with the attribute's maximum frequency when a value is lacking. Attributes can be univariate, monotonous in their missing values, or arbitrary. If at least three attributes have missing values, the model is said to be monotonous. If the missing values are of random characteristics, then it is arbitrary [45].

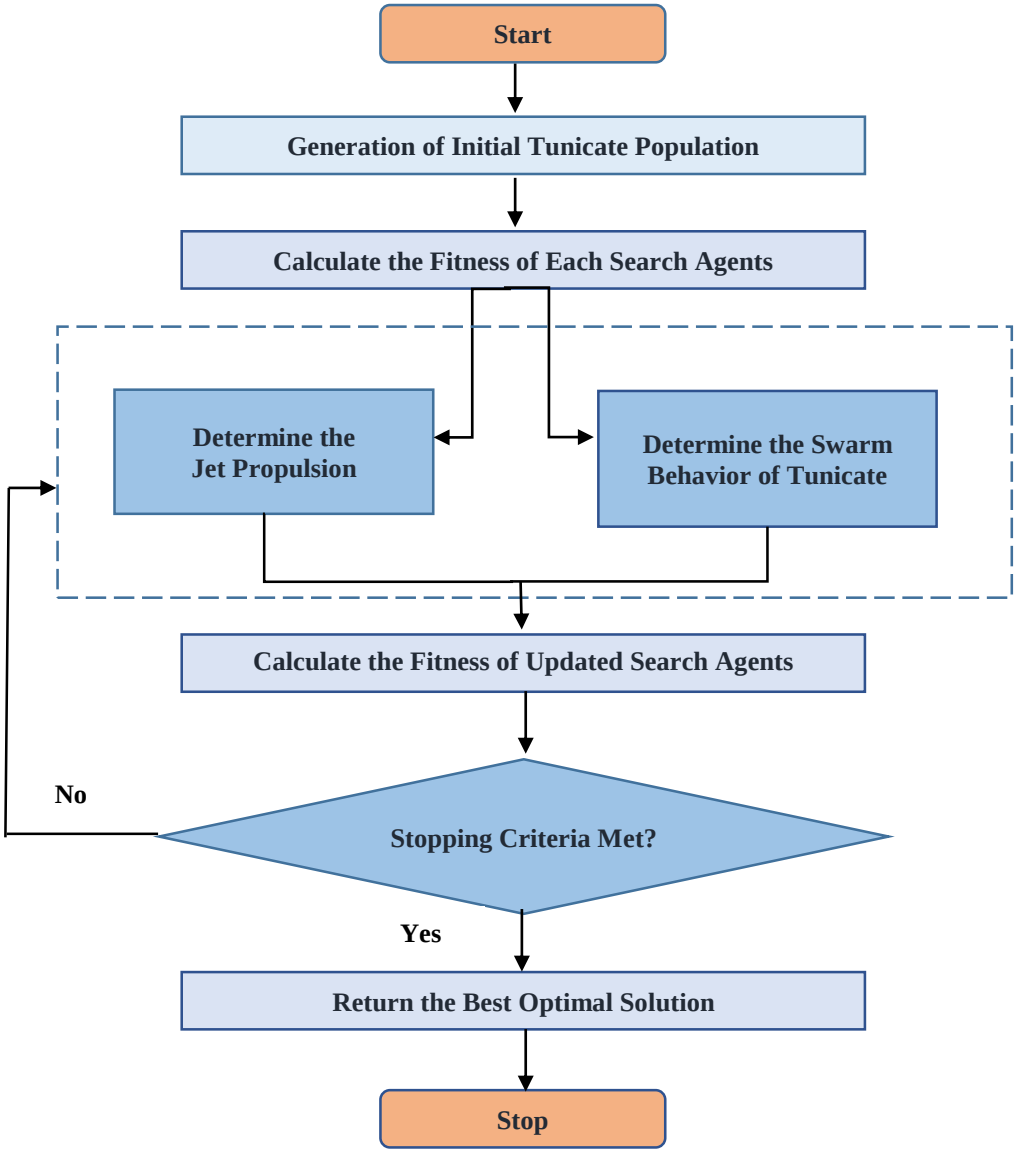


Figure 3. Flowchart of TSO algorithm

2) Data normalization:

There are numerous approaches to data normalization. Keep the data in a range for each input feature in order to reduce the neural network's preference for one feature over another. Training time can be reduced by normalizing data such that all features are trained at once. It is particularly beneficial for modelling applications when the inputs are often on a wide range of scales. The features or outputs are rescaled using the Min-Max normalizing method from one range of values to another. Most of the time, the features are rescaled to fall between 0 and 1 or -1 and 1. It is common to perform the rescaling by applying a linear interpretation formula like:

$$x_i' = ((\boldsymbol{max}_{\boldsymbol{target}} - \boldsymbol{min}_{\boldsymbol{target}}) x \frac{(x_i - \boldsymbol{min}_{\boldsymbol{value}})}{(\boldsymbol{max}_{\boldsymbol{value}} - \boldsymbol{min}_{\boldsymbol{value}})} + \boldsymbol{min}_{\boldsymbol{targe}}) \quad (15)$$

where $(\max_{\text{value}} - \min_{\text{value}}) = 0$. when $(\max_{\text{value}} - \min_{\text{value}}) = 0$ for a feature, it shows that that feature in the data has a constant value. Feature values having a constant value should be deleted from the data set because they do not contribute any useful information to the neural network. Min-max normalization maintains the same range of values for each feature when it is applied. The advantage of using min-max normalization is that it keeps all of the data's relationships intact.

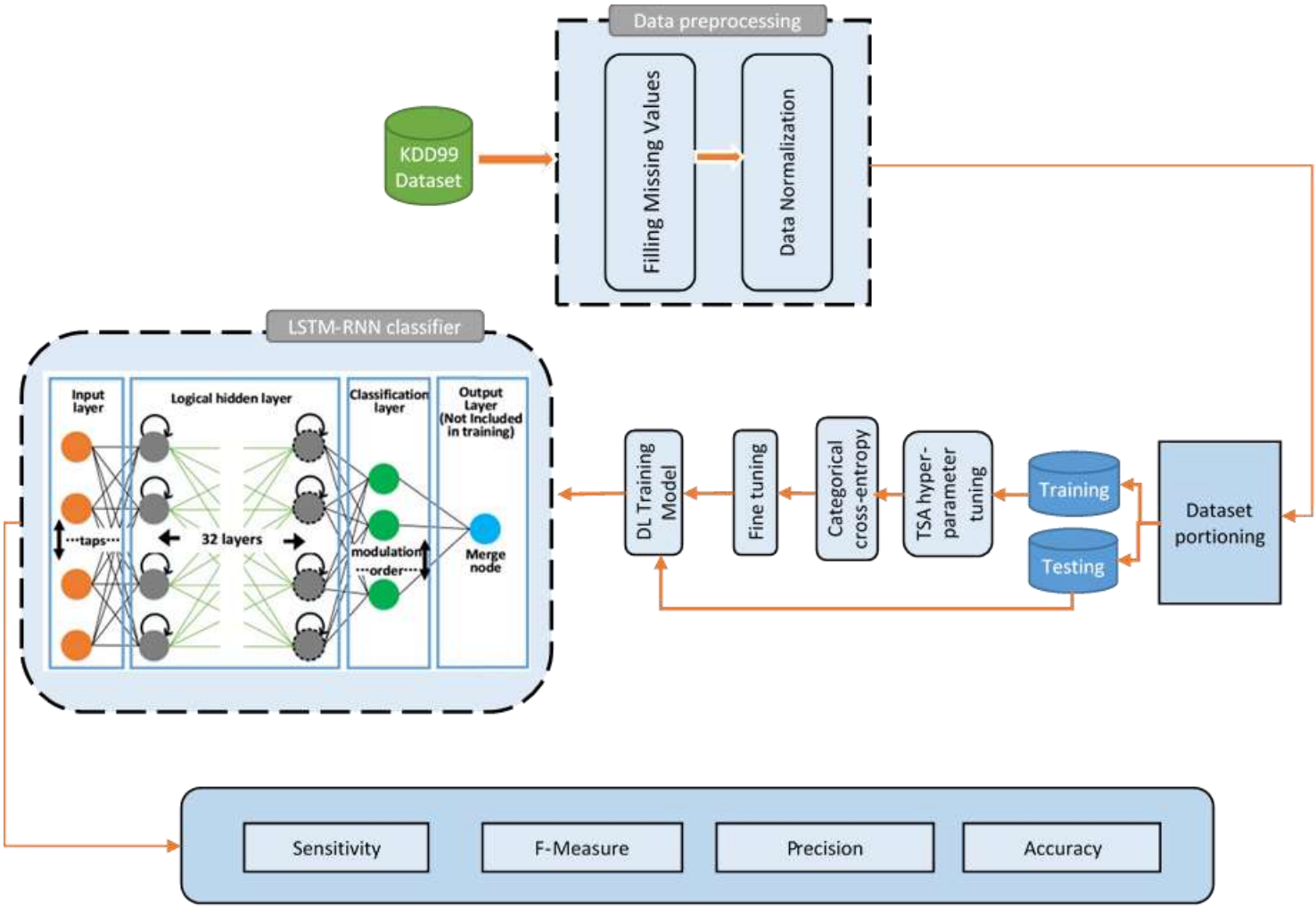


Figure 4. **The proposed TSA-LSTMRNN model**

B. PROCESS INVOLVED IN LSTMRNN MODEL

The presented TSA-LSTMRNN model applied the LSTMRNN model for the identification and classification of attacks in the IoT environment. LSTM is introduced for solving the issue of gradient vanishing by making novel path wherein the gradient flow is implemented for a protracted duration. Firstly, input, output, and forget gates are the main key utilized for managing data flows. Once there is an input, a forget gate chooses long-term memory data for removing the cell, and retrieving the input is noted in protracted memory by allotting a weight for each individual. In forward propagation, the input gate is utilized for estimating to allow the received memory unit. The output gate suggests the activation time. In the event of backpropagation, the output gate defines during input gate and error flow measurement whereas assigning the flow out of memory unit. Now, $o_t, f_t, i_t,$ and c_t determines the cell vector, output, forget, and input gates similarly. An LSTM is defined below [35], [36]:

$$i_t = \sigma(W_i x_t + U_i c_{t-1} + b_i) \quad (16)$$

$$f_t = \sigma\left(W_f x_t + U_f c_{t-1} + b_f\right) \quad (17)$$

$$o_t = \sigma(W_o x_t + U_o c_{t-1} + b_o) \quad (18)$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \ tanh\ (W_c x_t + U_c c_{t-1} + b_c) \quad (19)$$

$$h_t = o_t \cdot \ tanh\ (c_t) \quad (20)$$

Whereas $W_i, W_f, W_o,$ and $W_c$shows the input connection along with recurrent connection of the input, forget, output gates, and cell, next $U_i, U_f, U_o,$ and U_c are determined by peephole connection. $b_i, b_f, b_o,$ and $b_c$indicates the bias weight. $\sigma(\cdot)$ represent a sigmoid function. $Tanh(\cdot)$ and $\sigma(\cdot)$ indicate gate activation function.

Here, LSTM-RNN was employed to classify the financial position of the information. The LSTM-RNN using LSTM hidden layer was designed. The count of units hidden for each hidden layer is carefully chosen on a trial-and-error basis. The resulting layer is comprised of 5 neurons to categorize 4 kinds of sound and defects regions. The sample information is distributed into validation and training sets for encompassing an LSTM-RNN. Henceforth, the trained performance is minimal that denotes under-fitting. It turns out to be very complex while the sum total of hidden layers

is improved. Maximal training performance with minimum validation performance characterizes that a system is overfitting. Consequently, the LSTM-RNN method experiences training with huge areas. The training loss is minimized.

C. HYBER-PARAMETERS TUNING USING TSA

For improvising the detection outcomes of the LSTMRNN model, the TSA is applied to properly adjust the hyper parameter values involved in it. The TSO was encouraged by the abnormal actions of tunicates in ocean, particularly, the SI and jet drive of foraging process. An arithmetical technique of jet propulsion is innovative in 3 limits [34]: follows the location of maximal qualified agent, remaining nearby the optimum agent, and avoids conflicts among the exploration agents. For avoiding inter agent conflicts while looking for an optimum position, the novel agent position is assessed as follows:

$$\vec{A} = \frac{\vec{G}}{\vec{M}} \quad (21)$$

$$\vec{G} = c_2 + c_3 - \vec{F} \quad (22)$$

$$\vec{F} = c_1 \cdot \vec{F}. \quad (23)$$

Here $\vec{G}$ specifies the gravity force, $\vec{A}$ represents a vector of agent location, c_1, c_2 and c_3 characterizes 3 arbitrary amounts and $\vec{F}$ indicates the water flow in the deep ocean. The social forces amongst the agents are stored in a vector $\vec{M}$, as:

$$\vec{M} = [P_{min} + c_1 \cdot P_{max} - P_{min}]. \quad (24)$$

In which $P_{\min} = 1$ and $P_{\max} = 4$ defines the 1st and 2nd subordinates, representative of the speed of emerging social connection. Consequently, make certain that no conflict occurs amongst adjacent agents, the optimum position of optimum agent is estimated as:

$$\overrightarrow{PD} = \left| X_{best} - r_{rand} \cdot \overrightarrow{P_p(x)} \right| \quad (25)$$

In the equation, the vector $\overrightarrow{P_p(x)}$ has the location of the tunicate in iteration x. $\overrightarrow{PD}$ save the length amongst the optimal agent and food origin, X_{best} specifies optimum position, and $r_{rand,}$ characterizes a stochastic value within [0,1] For guarantying that searching agent is close to the optimum agent, its positions are estimated as follows:

$$\overrightarrow{P_p(x)} = \begin{cases} X_{best} + A \cdot \vec{P}D, if\ r_{rand} \geq 0.5 \\ X_{best} - A \cdot \vec{P}D, if\ r_{rand} < 0.5 \end{cases} \quad (26)$$

Now, $\overrightarrow{P_p(x)}$ represents the upgraded location in iteration x relative to the optimum scored position X_{best}. For modeling the swarming behaviors of tunicate, the location of the existing agent is upgraded according to the location of 2 agents:

$$P_p(\vec{x} + 1) = \frac{\overrightarrow{P_p(x)} + P_p(\vec{x}+1)}{2+c1} \quad (27)$$

Algorithm 1 shows the overall TSA process.

For better exploration and exploitation [37], TSA requires time complexity for jet propulsion and swarm behavior. As a result, the TSA algorithm's overall time complexity is defined as:

$$\text{TSA} = \text{O (Max iterations * n* d * N)} \quad (28)$$

To find the most optimal food source for tunicate, n, d, and N are used to define population size, jet propulsion, and swarm behavior, respectively.

V. EXPERIMENTAL RESULTS

A. DATASET DESCRIPTION

The TSA-LSTMRNN model's performance is tested using the KDD Cup99 Dataset [38], which contains data in five categories: DoS, R2L, normal, U2R, and Probe.

Many studies have used the KDD Cup99 dataset to evaluate the performance of IDS. Even though the dataset is outdated, it is still helpful to examine the IDS models. Because the same dataset yields a plethora of performance measurement findings. We choose the KDD Cup99 dataset primarily for this reason.

The dataset contains 4,898,431 network traffics, each with 41 unique attributes. In addition, there are 22 distinct types of attacks. Table I categorizes the assaults. When a DoS attack is launched, the target servers' resources are depleted, preventing any service from being provided. It is possible to gain remote access to a computer without authorization when using an R2L attack. An attack known as U2R aims to gain control of the system's superuser privileges. The purpose of a probing attack is to determine whether the targeted server is vulnerable.

We utilize KDD Cup9910 percentage data for testing and training because the original dataset has an excessive number of records. The data proportion of the attacks, on the other hand, leans heavily toward denial-of-service attacks. Only 1% of the population is made up of the rest. Figure 2 depicts the situation. IDS will be trained unjustly as a result. Thus, DoS assaults are easy to identify, but other attacks remain undetected.

B. PERFORMANCE METRICS

One-half of the data is used to train the model, and the other half is used to test the model's predictions on the data. For testing, the data out from the second portion is used (30 percent of the time). Six performance metrics are used to analyze and validate the proposed model. Sensitivity, Precision, Accuracy, and F-Measure [34], [39] is an examples of one of these characteristics. Contrast matrices quantify the performance of classification algorithms by assessing performance metrics. Methodological performance was evaluated using the following metrics.

True Positive (TP): This denotes instances of correctly classified positive outputs.

True Negatives (TN): These denote instance of correctly classified negative output.

Algorithm# 1 Tunicate Swarm Algorithm (TSA)

Input ← Tunicate population $\vec{P}_p$

Output ← Optimal fitness value $\overrightarrow{FS}$

procedure TSA

Initialize the parameters $\vec{A}, \vec{G}, \vec{F}, \vec{M}$, and $\text{Max}_{\text{iterations}}$

Set $P_{\min} \leftarrow 1$

Set $P_{\max} \leftarrow 4$

Set Swarm ← 0

while $(x < \text{Max}\, x_{\text{iterations}})$ do

for i ← 1 to 2 do

$\overrightarrow{FS} \leftarrow$ ComputeFitness $(\vec{P}_p)$

$c_1, c_2, c_3, r_{\text{and}} \leftarrow$ Rand ()

$\vec{M} \leftarrow \lfloor P_{\min} + c_1 \times P_{\max} - P_{\min} \rfloor$

$\vec{F} \leftarrow 2 \times c_1$

$\vec{G} \leftarrow c_2 + c_3 - \vec{F}$

$\vec{A} \leftarrow \vec{G}/\vec{M}$

$\overrightarrow{PD} \leftarrow \text{ABS}(\overrightarrow{FS} - r_{\text{and}} \times P_p(x))$

if $(r_{\text{and}} \leq 0.5)$ then

Swarm ← S warm $+\overrightarrow{FS} + \vec{A} \times \overrightarrow{PD}$

else

Swarm ← S warm $+\overrightarrow{FS} - \vec{A} \times \overrightarrow{PD}$

end if

end for

$\vec{P}_p(x) \leftarrow S$ warm $/(2 + c_1)$

Swarm ← 0

Update the parameters $\vec{A}, \vec{G}, \vec{F}$, and $\vec{M}$

$x \leftarrow x + 1$

end while

return $\overrightarrow{FS}$

End procedure

Procedure ComputeFitness $(\vec{P}_p)$

for i ← 1 to n do

$FIT[i] \leftarrow$ Fitness Function $(P_p(\vec{i}, :)$

end for

$FIT_{p_{\text{best}}} \leftarrow \text{BEST}(FIT_p[\ ])$

return $FIT_{p_{\text{best}}}$

End procedure

Procedure *BEST* (FIT_p)

Best ← $FIT_p[0]$

for i ← 1 to n do

if $(FIT_p[i] < Best)$ then

Best ← $FIT_p[i]$

end if

end for

return Best

End procedure

TABLE I

CATEGORY OF THE ATTACKS

Category	Attacks
DoS	back, land, neptune, pod, smurf, teardrop
R2L	ftp-write, guess-passwd, imap, multihop, phf, spy, warezclient, warezmaster
U2R	buffer-overflow, loadmodule, perl, rootkit
Probe	ipsweep, nmap, portsweep, satan

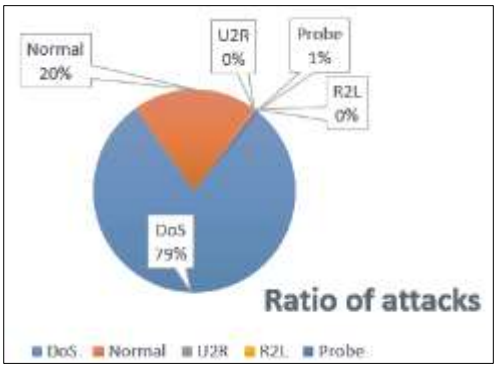


Figure 5. KDD Cup9910 dataset categories of attacks

We use the term "*False positive*" (FP) for negative outcomes that are wrongly deemed positive. Good events that were incorrectly labelled as negative in the report are known as *false negatives* (FN). For an image to be considered accurate in a database, it must have coordinates that are very similar to the database's actual value.

Accuracy measures are very near to the true value and are processed as a give perfect of the outcomes.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{29}$$

To be sensitive, a person is said to be sensitive, as in the adjective "sensitive." Sensitivity or recall is the ability to correctly recognize people who are suffering from a specific disease (True Positive Rate). As a result, the following can be said:

$$Sensitivity = \frac{TP}{TP+FN} \tag{30}$$

People who don't have the condition can be correctly identified with a test that has excellent specificity (True Negative Rate). The following is what it means by that definition:

$$Specificity = \frac{TN}{TN+FP} \tag{31}$$

Accuracy is measured by the Predictive Value (PPV) or Precision. We arrived to this conclusion using the following formula:

$$Precision = \frac{TP}{TP+FP} \tag{32}$$

It is possible to determine the harmonic mean of recall and precision using the following formula:

$$F - \text{Measure} = 2 \times \frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \tag{33}$$

C. PERFORMANCE COMPARISON

Fig. 6 illustrates a set of three confusion matrices generated by the TSA-LSTMRNN model on applied dataset. On entire dataset, the TSA-LSTMRNN model has recognized 9451 samples under normal class, 7380 samples under DoS, 2497 samples under Probe, 2241 samples under R2L class, and 169 samples under U2R class. Meanwhile, on 70% of training dataset, the TSA-LSTMRNN technique has recognized 6618 samples under normal class, 5174 samples under DoS, 1731 samples under Probe, 1551 samples under R2L class, and 114 samples under U2R class. Eventually, on 30% of testing dataset, the TSA-LSTMRNN method has recognized 2833 samples under normal class, 2206 samples under DoS, 766 samples under Probe, 690 samples under R2L class, and 55 samples under U2R class.

Table II and Fig. 7 investigate brief classification outcomes of the TSA-LSTMRNN model on test dataset. The experimental outcomes indicated that the TSA-LSTMRNN model has resulted in maximum performance. For instance, with entire dataset, the TSA-LSTMRNN model has attained average $accu_y$ of 98.57%, $prec_n$ of 85.59%, $reca_l$ of 92.80%, and F_{score} of 87.73%. Concurrently, with 70% of training dataset, the TSA-LSTMRNN technique has reached average $accu_y$ of 98.50%, $prec_n$ of 84.97%, $reca_l$ of 92.92%, and F_{score} of 87.17%. Simultaneously, with 30% of testing dataset, the TSA-LSTMRNN approach has reached average $accu_y$ of 98.73%, $prec_n$ of 87.11%, $reca_l$ of 92.67%, and F_{score} of 89.01%.

Fig. 8 reports the precision-recall curve analysis of the TSA-LSTMRNN model under entire dataset. The figures indicated that the TSA-LSTMRNN model has resulted in effectual outcomes under all five classes.

TABLE II
TSA-LSTMRNN RESULTS ANALYSIS

Class Labels	Accuracy	Precision	Recall	F-Score
Entire Dataset				
Normal	98.19	98.46	97.32	97.89
Dos	98.90	97.76	98.95	98.35
Probe	98.09	93.49	90.67	92.06
R2L	98.86	96.64	92.57	94.56
U2R	98.81	41.63	84.50	55.78
Average	98.57	85.59	92.80	87.73
Training (70%)				
Normal	98.11	98.47	97.14	97.80
Dos	98.87	97.62	98.99	98.30
Probe	97.92	93.01	89.69	91.32
R2L	98.83	96.46	92.43	94.40
U2R	98.77	39.31	86.36	54.03
Average	98.50	84.97	92.92	87.17
Testing (30%)				
Normal	98.37	98.44	97.76	98.10
Dos	98.99	98.09	98.88	98.48
Probe	98.49	94.57	92.96	93.76
R2L	98.91	97.05	92.87	94.91
U2R	98.91	47.41	80.88	59.78
Average	98.73	87.11	92.67	89.01

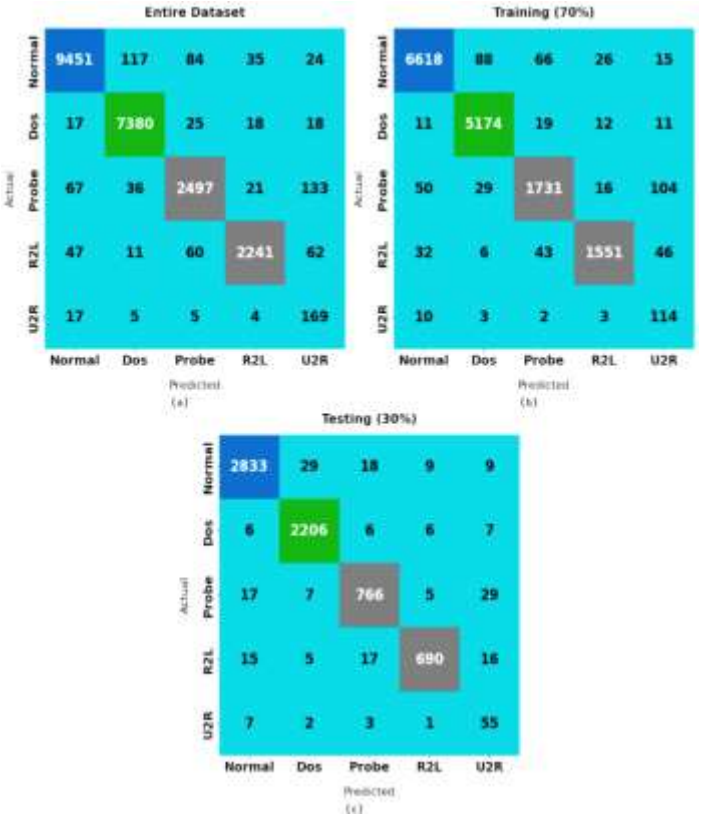


Figure 6. Confusion matrix of TSA-LSTMRNN technique with different three datasets

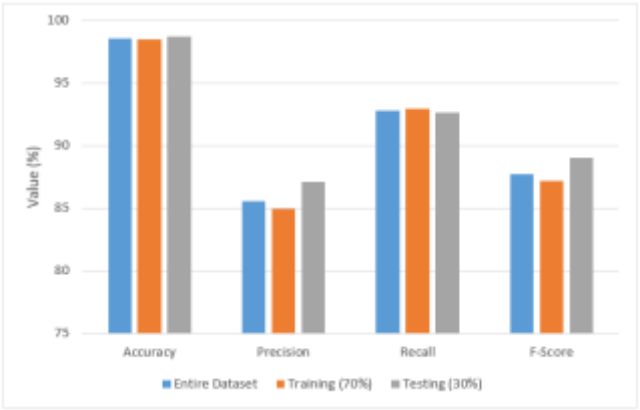


Figure 7. Result analysis of TSA-LSTMRNN technique with distinct measures and datasets

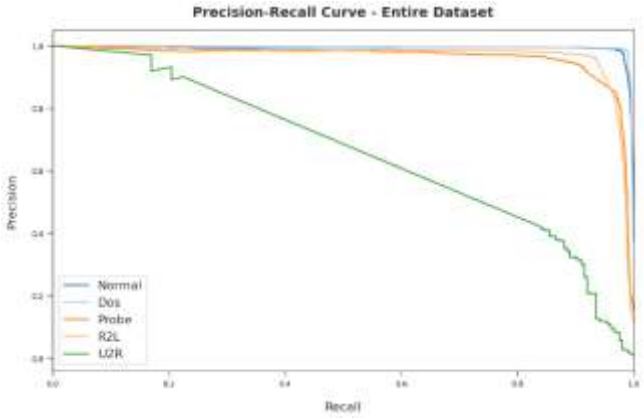


FIGURE 8. Precision-recall analysis of TSA-LSTMRNN technique under entire dataset

Fig. 9 showcases the precision-recall curve analysis of the TSA-LSTMRNN approach under 70% of training dataset. The figures exposed that the TSA-LSTMRNN technique has resulted in effectual outcomes under all five classes.

Fig. 10 defines the precision-recall curve analysis of the TSA-LSTMRNN method under 30% of testing dataset. The figures indicated that the TSA-LSTMRNN model has resulted in effectual outcomes under all five classes.

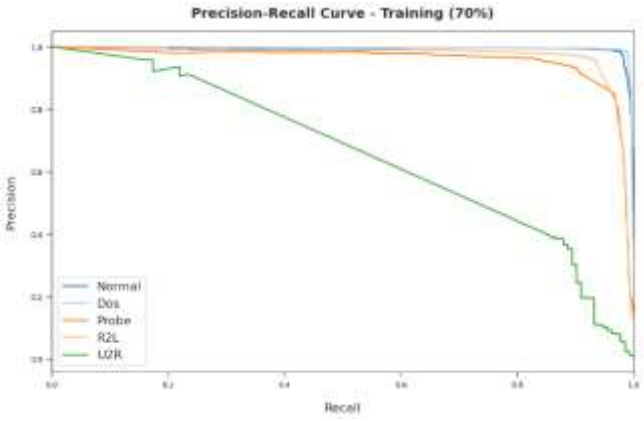


Figure 9. Precision-recall analysis of TSA-LSTMRNN technique under 70% of training dataset

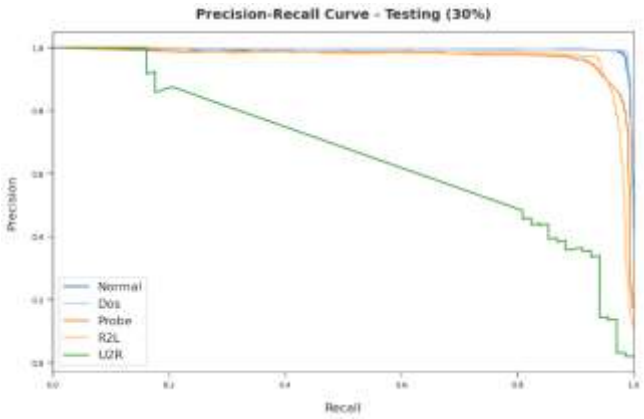


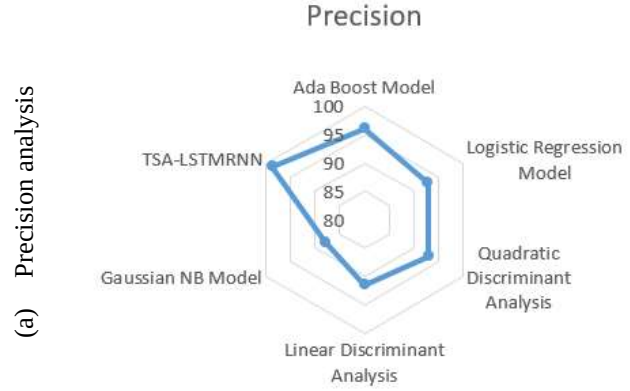
Figure 10. Precision-recall analysis of TSA-LSTMRNN technique under 30% of testing dataset

TABLE III
COMPARATIVE ANALYSIS OF TSA-LSTMRNN WITH LITERATURE

Methods	Precision	Recall	Accuracy
Ada Boost Model	96.10	85.55	91.07
Logistic Regression Model	93.01	83.09	90.93
Quadratic Discriminant Analysis	93.04	83.20	91.75
Linear Discriminant Analysis	91.56	82.13	92.54
Gaussian NB Model	87.94	87.09	88.11
TSA-LSTMRNN	98.73	87.11	92.67

For ensuring the supremacy of the TSA-LSTMRNN model, a comparative study with existing methods is made in Table III [40]. Fig. 11 (a) investigates the comparison study of the TSA-LSTMRNN model with recent models. The figure indicated that the Gaussian NB model has resulted in lower precision of 87.94%. In line with, the LR, QDA, and LDA models have accomplished moderately improved precision of 93.01%, 93.04%, and 91.56% respectively. Though the Adaboost model has resulted in reasonable precision of 96.10%, the presented TSA-LSTMRNN model has outperformed the other methods with maximum precision of 98.73%.

Fig. 11 (b) examines the comparison study of the TSA-LSTMRNN model with recent models. The figure referred that the Gaussian NB model has resulted in lower recall of 87.09%. Along with that, the LR, QDA, and LDA models have accomplished moderately improved recall of 83.09%, 83.20%, and 82.13% respectively. In addition, the Adaboost model has resulted in reasonable recall of 85.55%, the presented TSA-LSTMRNN model has outperformed the other methods with maximal recall of 87.11%.



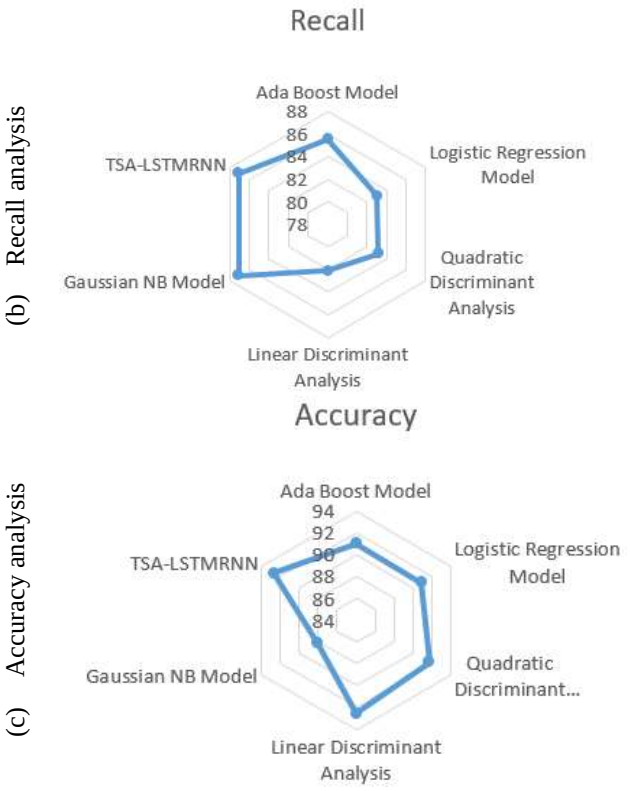


Figure 11. Analysis of TSA-LSTMRNN approach with recent algorithms

Fig. 11 (c) inspects the comparison study of the TSA-LSTMRNN model with recent models. The figure indicated that the Gaussian NB model has resulted in lower accuracy of 88.11%. Also, the LR, QDA, and LDA models have accomplished moderately improved accuracy of 90.93%, 91.75%, and 92.54% correspondingly. But, the Adaboost model has resulted in reasonable accuracy of 91.07%, the presented TSA-LSTMRNN model has outperformed the other methods with maximum accuracy of 92.67%.

After observing the above mentioned tables and discussion, it can be ensured that the TSA-LSTMRNN technique has been able reasonable performance over the other methods.

As can be seen in the table, the proposed model is evaluated in comparison to a number of other models, some of which are the Ada Boost Model, the Logistic Regression Model, the Quadratic Discriminant Analysis, the Linear Discriminant Analysis, and the Gaussian NB Model, respectively that can be applied in our practical section. The TSA-LSTMRNN model outperforms every other model that was investigated and evaluated, making it the clear winner of this round of comparisons. The TSA-LSTMRNN model is put through a series of experimental evaluations using benchmark datasets. The results of these analyses show that the TSA-LSTMRNN model possesses superior properties. When compared to previous models, the TSA-LSTMRNN model that was proposed had superior results in terms of accuracy (92.67%), recall (87.11%), and precision (98.73%). One of the shortcomings of the model that was proposed is that it can only be utilized with datasets that contain a non-uniform distribution of class labels. This is one of the limitations of the model. In addition to that, there is a rule that states that no metaheuristic algorithm can solve all problems. Because of this rule, the TSA is unable to consistently guarantee the best performance. This rule was made to take into account the fact that no metaheuristic algorithm can solve all of the problems in the world.

VI. CONCLUSION

As a result of this research, a novel TSA-LSTMRNN model for detecting the presence of attacks in the IoT environment has been developed. As a result, the data preprocessing performed by the presented model is used to convert the input data into a format that can be used. In addition, the LSTMRNN model is used for the identification and classification of attacks in the Internet of Things environment, as previously stated. The TSA is used to properly adjust the hyper-parameter values involved in the LSTMRNN model to improve the detection outcomes of the model. A series of experimental analyses are carried out on benchmark datasets, with the results demonstrating that the TSA-LSTMRNN model has superior characteristics. In terms of accuracy (92.67%), recall (87.11%), and precision (98.73%), the proposed TSA-LSTMRNN model did better than other models. In the future, feature selection models can be used to improve the performance of the proposed model. Moreover, new metaheuristic algorithms can be used for better performance with new datasets. For anomaly detection, a new hybrid algorithm may be useful for algorithm exploration and exploitation.

REFERENCES

[1] J. C. S. Sicato, S. K. Singh, S. Rathore, and J. H. Park, "A comprehensive analyses of intrusion detection system for IoT environment," *J. Inf. Process. Syst.*, vol. 16, no. 4, pp. 975–990, 2020.

[2] V. Kumar, A. K. Das, and D. Sinha, "UIDS: a unified intrusion detection system for IoT environment," *Evol. Intell.*, vol. 14, no. 1, pp. 47–59, 2021.

[3] M. F. Elrawy, A. I. Awad, and H. F. A. Hamed, "Intrusion detection systems for IoT-based smart environments: a survey," *J. Cloud Comput.*, vol. 7, no. 1, pp. 1–20, 2018.

[4] M. A. Alsoufi *et al.*, "Anomaly-based intrusion detection systems in iot using deep learning: A systematic literature review," *Appl. Sci.*, vol. 11, no. 18, p. 8383, 2021.

[5] H. Alkahtani and T. H. H. Aldhyani, "Intrusion detection system to advance internet of things infrastructure-based deep learning algorithms," *Complexity*, vol. 2021, 2021.

[6] P. K. Keserwani, M. C. Govil, E. S. Pilli, and P. Govil, "A smart anomaly-based intrusion detection system for the Internet of Things (IoT) network using GWO--PSO--RF model," *J. Reliab. Intell. Environ.*, vol. 7, no. 1, pp. 3–21, 2021.

[7] J. Xing and Z. Zhang, "Hierarchical Network Security Measurement and Optimal Proactive Defense in Cloud Computing Environments," *Secur. Commun. Networks*, vol.

2022, 2022.

[8] Y. N. Soe, P. I. Santosa, and R. Hartanto, "Ddos attack detection based on simple ann with smote for iot environment," in *2019 fourth international conference on informatics and computing (ICIC)*, 2019, pp. 1–5.

[9] S. V Simpson and G. Nagarajan, "A fuzzy based Co-Operative Blackmailing Attack detection scheme for Edge Computing nodes in MANET-IOT environment," *Futur. Gener. Comput. Syst.*, vol. 125, pp. 544–563, 2021.

[10] A. Y. Khan, R. Latif, S. Latif, S. Tahir, G. Batool, and T. Saba, "Malicious insider attack detection in IoTs using data analytics," *IEEE Access*, vol. 8, pp. 11743–11753, 2019.

[11] M. Waqas *et al.*, "Botnet attack detection in Internet of Things devices over cloud environment via machine learning," *Concurr. Comput. Pract. Exp.*, vol. 34, no. 4, p. e6662, 2022.

[12] Q. Jiang *et al.*, "Intelligent monitoring for infectious diseases with fuzzy systems and edge computing: A survey," *Appl. Soft Comput.*, p. 108835, 2022.

[13] M. S. A. Muthanna, R. Alkanhel, A. Muthanna, A. Rafiq, and W. A. M. Abdullah, "Towards SDN-Enabled, Intelligent Intrusion Detection System for Internet of Things (IoT)," *IEEE Access*, vol. 10, pp. 22756–22768, 2022.

[14] S. Roy, J. Li, B.-J. Choi, and Y. Bai, "A lightweight supervised intrusion detection mechanism for IoT networks," *Futur. Gener. Comput. Syst.*, vol. 127, pp. 276–285, 2022.

[15] D. K. K. Reddy, H. S. Behera, J. Nayak, B. Naik, U. Ghosh, and P. K. Sharma, "Exact greedy algorithm based split finding approach for intrusion detection in fog-enabled IoT environment," *J. Inf. Secur. Appl.*, vol. 60, p. 102866, 2021.

[16] J. Liu, D. Yang, M. Lian, and M. Li, "Research on intrusion detection based on particle swarm optimization in IoT," *IEEE Access*, vol. 9, pp. 38254–38268, 2021.

[17] S. Tharewal, M. W. Ashfaque, S. S. Banu, P. Uma, S. M. Hassen, and M. Shabaz, "Intrusion Detection System for Industrial Internet of Things Based on Deep Reinforcement Learning," *Wirel. Commun. Mob. Comput.*, vol. 2022, 2022.

[18] C. A. de Souza, C. B. Westphall, and R. B. Machado, "Two-step ensemble approach for intrusion detection and identification in IoT and fog computing environments," *Comput. Electr. Eng.*, vol. 98, p. 107694, 2022.

[19] C. Meffert, D. Clark, I. Baggili, and F. Breitinger, "Forensic State Acquisition from Internet of Things (FSAIoT) A general framework and practical approach for IoT forensics through IoT device state acquisition," in *Proceedings of the 12th International Conference on Availability, Reliability and Security*, 2017, pp. 1–11.

[20] M. Hossain, Y. Karim, and R. Hasan, "FIF-IoT: A forensic investigation framework for IoT using a public digital ledger," in *2018 IEEE International Congress on Internet of Things (ICIOT)*, 2018, pp. 33–40.

[21] M. M. Hossain, R. Hasan, and S. Zawoad, "Probe-IoT: A public digital ledger based forensic investigation framework for IoT.," in *INFOCOM workshops*, 2018, pp. 1–2.

[22] M. Cebe, E. Erdin, K. Akkaya, H. Aksu, and S. Uluagac, "Block4forensic: An integrated lightweight blockchain framework for forensics applications of connected vehicles," *IEEE Commun. Mag.*, vol. 56, no. 10, pp. 50–57, 2018.

[23] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 2, no. 1, pp. 41–50, 2018.

[24] S. Prabakaran and S. Mitra, "Survey of analysis of crime detection techniques using data mining and machine learning," in *Journal of Physics: Conference Series*, 2018, vol. 1000, no. 1, p. 12046.

[25] Z. Wang, "The applications of deep learning on traffic identification," *BlackHat USA*, vol. 24, no. 11, pp. 1–10, 2015.

[26] G. Zhao, C. Zhang, and L. Zheng, "Intrusion detection using deep belief network and probabilistic neural network," in *2017 IEEE international conference on computational science and engineering (CSE) and IEEE international conference on embedded and ubiquitous computing (EUC)*, 2017, vol. 1, pp. 639–642.

[27] A. Zela, A. Klein, S. Falkner, and F. Hutter, "Towards automated deep learning: Efficient joint neural architecture and hyperparameter search," *arXiv Prepr. arXiv1807.06906*, 2018.

[28] B. Wang and N. Z. Gong, "Stealing hyperparameters in machine learning," in *2018 IEEE Symposium on Security and Privacy (SP)*, 2018, pp. 36–52.

[29] T. Chen *et al.*, "Learning to optimize tensor programs," *Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[30] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[31] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[32] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short term memory recurrent neural network classifier for intrusion detection," in *2016 international conference on platform technology and service (PlatCon)*, 2016, pp. 1–5.

[33] Q. Zhai, H. Rahardjo, A. Satyanaga, Y. Zhu, G. Dai, and X. Zhao, "Estimation of wetting hydraulic conductivity function for unsaturated sandy soil," *Eng. Geol.*, vol. 285, p. 106034, 2021.

[34] S. Kaur, L. K. Awasthi, A. L. Sangal, and G. Dhiman, "Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization," *Eng. Appl. Artif. Intell.*, vol. 90, p. 103541, 2020.

[35] M. Abdel-Nasser and K. Mahmoud, "Accurate photovoltaic power forecasting models using deep LSTM-RNN," *Neural Comput. Appl.*, vol. 31, no. 7, pp. 2727–2740, 2019.

[36] J. Kumar, R. Goomer, and A. K. Singh, "Long short term memory recurrent neural network (LSTM-RNN) based workload forecasting model for cloud datacenters," *Procedia Comput. Sci.*, vol. 125, pp. 676–682, 2018.

[37] E. H. Houssein, B. E.-D. Helmy, A. A. Elngar, D. S. Abdelminaam, and H. Shaban, "An improved tunicate swarm algorithm for global optimization and image segmentation," *IEEE Access*, vol. 9, pp. 56066–56092, 2021.

[38] K. "@" Ics.uci.edu, "KDD Cup 1999 Data." [Online]. Available: https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. [Accessed: 06-May-2022].

[39] S. K. Baliarsingh, W. Ding, S. Vipsita, and S. Bakshi, "A memetic algorithm using emperor penguin and social engineering optimization for medical data classification," *Appl. Soft Comput.*, vol. 85, p. 105773, 2019.

[40] W. Ben Daoud and S. Mahfoudhi, "SIMAD: Secure Intelligent Method for IoT-Fog Environments Attacks Detection," *C. Mater. Contin.*, vol. 70, no. 2, pp. 2727–2742, 2022.

Fatma Taher (Member, IEEE) received the Ph.D. degree from the Khalifa University of Science, Technology and Research, United Arab Emirates, in 2014. She is currently the Assistant Dean of the College of Technological Innovation, Zayed University, Dubai, United Arab Emirates. She has published more than 40 articles in international journals and conferences. Her research interests are in the areas of signal and image processing, pattern recognition, deep learning, machine learning, artificial intelligence, medical image analysis, especially in detecting of the cancerous cells, kidney transplant, and autism. In addition to that, her researches are watermarking, remote sensing, and satellite images. She served as a member of the steering, organizing, and technical program committees of many international conferences.

Dr. Mohamed Elhoseny is an Associate Professor at the University of Sharjah, UAE. Dr. Elhoseny is an ACM Distinguished Speaker and IEEE Senior Member. His research interests include Smart Cities, Network Security, Artificial Intelligence, Internet of Things, and Intelligent Systems. Dr. Elhoseny is the founder and the Editor-in-Chief of IJSSTA journal published by IGI Global. Also, he is an Associate Editor at several Q1 journals such as IEEE Journal of Biomedical and Health Informatics, IEEE Access, Scientific Reports, IEEE Future Directions, Remote Sensing, International Journal of E-services and Mobile Applications and Human-centric Computing and Information Sciences. Moreover, he served as the co-chair, the publication chair, the program chair, and a track chair for several international conferences published by recognized publishers such as IEEE and Springer. Dr. Elhoseny is the Editor-in-Chief of The Sensors Communication for Urban Intelligence CRC Press-Taylor& Francis Book Series, and the Editor-in-Chief of The Distributed Sensing and Intelligent Systems CRC Press-Taylor& Francis Book Series.

Mohammed K. Hassan received his B.Sc., M.Sc., and Ph.D. degrees in Computer engineering from the Faculty of Engineering, Mansoura University, Egypt. His research interests include (but are not limited to) data mining, classification, big data, cloud computing and Big Data analysis.

Dr. **Ibrahim M. EL-Hasnony** received his B.Sc., M.Sc., Ph.D degrees in information systems from Mansoura University, Egypt in 2010, 2016 and 2021, respectively. His research interests include cloud computing, big data, data analysis, smart city, the Internet of Things, neural networks, artificial intelligence, web service composition, blockchain, and evolutionary algorithms. He has published several papers in international and local conferences, journals and proceedings in different fields of information technology.