SECURING IOT DEVICES FROM BOTNETS USING LSTM AUTOENCODER

L.Sumathi
Department of Computer Science and Engineering
Government College of Technology,
Coimbatore,India

T.Suguna
Department of Information Technology
Government College of Technology,
Coimbatore,India

Abstract: *The devices present in the IoT environment has high vulnerability since it is on online for 24X7 hours. This scenario makes the attacker to establish plenty of security breaches towards on the side of IoT devices or its associated networks. In order to sustain in this situation without any huge loss of information or devices, an efficient design of security solution is an essential part. This can be achieved by the installation of intrusion detection system (IDS) either on the location of host or network of the IoT devices and its associated networks. The main purpose of IDS is to identify the incoming malicious traffic mingled with legitimate traffic. Recently the performance of deep learning has been dominated by the machine learning techniques. The types of deep learning techniques can be split into supervised and unsupervised. In supervised techniques, the labelled dataset is given as input for the deep learning model whilst unsupervised techniques utilized unlabeled dataset as input. In this way, unsupervised based deep learning model is efficient in detecting new incoming malicious traffic to or from the IoT device. Hence in this work, an unsupervised deep learning technique has been used here namely LSTM autoencoder to discriminate normal traffic from legitimate. To make the model more effective in terms of accuracy, LSTM autoencoder has been implemented on top of mayfly optimization algorithm. The performance of the model has been compared with the recent state-of-the approach.*

Keywords: *: LSTM Autoencoder, mayfly optimization, IoT Botnet, IoT, intrusion detection, classification*

1. Introduction

The Internet of Things (IoT) has gained popularity in recent years due to its new uses and support for a wide range of domains such as industrial processes, health care, automation, smart environments, and so on. Despite the fact that IoT offers a wide range of services and applications, it is vulnerable to cyber attacks. Because IoT is a diverse circumstances, and its compatibility process is incompatible with traditional security methods. Nevertheless, IoT security is improved in other ways, such as data authentication, confidentiality, and access controls. These safety precautions are developed in collaboration with the user, but they still face security issues. As a result, it is critical to provide a separate module to ensure IoT network security. Intrusion detection systems (IDS) are examples of such concepts that are already in use in wireless networks. Improving wireless network IDS features will aid IoT in protecting the network from attacks and other weaknesses. IoT made use of the internet and real-time applications to provide a more efficient and convenient environment for users. IoT is not limited to a single goal; it supports multiple goals and must meet security requirements for large-scale attacks. Prior knowledge of the IoT environment and security issues is required before developing an intrusion detection system.

A data-oriented security mechanism is required to prevent malicious users from gaining illicit access to data sources. It is critical to prioritise data confidentiality and integrity in order to reduce serious security threats in an IoT environment. Traditional security mechanisms are based on cryptography techniques and are not widely used in the IoT environment due to the large volume of data. Risks must be identified in a short period of time to reduce network issues, and traditional security models require more time to process such large amounts of data to identify threats. Unauthorised data access for a short period of time is sufficient for an unauthorised user to obtain confidential data, and modification of such data has a significant impact on the user. In order to detect an intruder in an IoT network, IDS must be implemented. Intrusion detection systems detect attackers and safeguard the network and data by avoiding unauthorized users from accessing it. However, due to limited energy resources, implementing IDS is a complex process. To reduce such complexity, a central intrusion detection system that tracks the network and distant nodes to detect intrusions could be used. As a result, the

network administrator receives an alert to respond to the security issues.

The intrusion detection system's operation is divided into three phases. Monitoring is the first stage of IDS and is based on network or host sensors. The second phase of IDSs is analysis, which is based on feature extraction and pattern identification. The detection phase of IDSs detects anomalies or intrusions in a network. IDS aids in the monitoring and analysis of information, services, and networks, as well as traffic analysis and vulnerability identification in a short period of time. It protects the network from attacks and improves data, network confidentiality, and network integrity. IDS summaries and analyses system data traffic to detect harmful or malicious activities. In recent decades, most of the analyzing process of traffic data has been carried out using deep learning techniques and achieves higher detection rate. Hence in this work, the hybrid unsupervised deep learning technique has been proposed by combining deep autoencoder with mayfly optimization technique. This approach enhances the higher detection rate in an unsupervised manner.

2. Related works

Fahimeh et al[1] proposed an intrusion detection system using deep auto encoder for classifying the network traffic as normal and malicious. The designed is trained using greedy layer-wise fashion in such a way that to avoid overfitting as well as local optima. This model is tested using NSL-KDD dataset. Hongpo Zhang et al [2] presented a security framework based on deep learning techniques. This model make use of denoising autoencoder along with weighted loss function used for feature selection to retrieve the significant features alone for multilayer perceptron classifier model. This model is evaluated using UNSW-NB dataset to prove its efficiency. Can Aygun et al[3] focused on the detection of zero-day attack with high accuracy using deep learning model. They are autoencoder connected with denoising autoencoder layers. In order to achieve more detection accuracy, stochastic approach is employed. Here the designed model is tested using KDD test+ dataset which is present in NSL KDD dataset. Giampaolo et al[4] proposed an intrusion detection system for IoT environment using lightweight proposed solution so called MultiModal Deep AutoEncoder. Further the given malicious attacks were classified using soft -output function. The proposed model is evaluated using the real time IoT dataset namely BoTIoT. Wei et al [5] proposed intrusion detection system for android based devices with two deep learning architectures such as CNN(convolutional neural network) and DAE(deep autoencoder).Here the proposed model is compared against with CNN-S(two CNN connected in series) and a machine learning technique namely SVM(Support Vector Machine). SHERAZ et al[6] proposed deep intrusion detection system using three conventional deep learning techniques such as Convolutional neural network, LSTM(Long short term memory) and deep autoencoders. The performance of these deep learning techniques is compared and analyzed using traditional machine learning techniques such as nearest neighbor, decision-tree, random-forest, support vector machine, naive-bayes, and quadratic discriminant analysis.Muna et al [7] proposed an intrusion detection framework for Internet Industrial Control Systems (IICSs) using deep learning techniques such as deep auto-encoder and deep feed forward neural network by analysing TCP/IP packets. The motive of this design is to detect new kind of attack and it is tested using UNSW-NB15 and NSL KDD datasets. XuKuiLi et al [8] suggested an autoencoder based intrusion detection system by using random forest algorithm. Here the feature selection and feature grouping is done implementing DAE algorithm. Mohammed et al [9] proposed an model called AutoIDS , it can classify the packet flows as normal and malicious packets. This proposed model comprises of two encoder and two decoder so that totally 2 detectors (DAE) is implemented here. If the malicious flow is missed in first detector, it will be captured at second detector. This model is trained and tested using NSL KDD dataset. Aqsa et al[10] presented an deep intrusion detection framework using deep sparse autoencoder along with adaptive transfer learning method. Here pretrained model designed for regression task is used to extract features from NSL KDD dataset and it is given as input to sparse autoencoder for further classification.Chawla et al [11] proposed an intrusion detection system using bidirectional LSTM by analyzing the system call traces. Here the Bidirectional Encoder and a unidirectional Decoder is trained on normal instances so when the abnormal instances were given as input, it shows high reconstruction error. But the normal instances tend to give low reconstruction error. Ashraf et al [12] suggested an intrusion detection framework for Internet of Vehicles environment. It is accomplished using LSTM autoencoder and this model is trained and tested using two datasets such as car hacking and UNSW-NB15 dataset. Mahmoud et al [13] presented an intrusion detection system using LSTM with autoencoder whereas

this model has 6-layers and this model has been trained and tested using NSL-KDD dataset. This model has been capable of performing both binary as well as multiple classification. Ketepalli et al [14] proposed an intrusion detection system using LSTM autoencoder and a Random Forest. Author has utilized autoencoder for feature extraction to obtain the significant attributes for classification. This suggested model has been trained and tested using NSLKDD dataset. Sumathi et al [15] proposed a one class SVM with grey wolf optimizer for IoT environment. Author has trained this model with normal traffic instances and the model achieves good detection performance without false prediction of abnormal as normal and negligible misprediction of normal as abnormal. Sumathi et al [16] presented an generative model , Apposite Adversarial Autoencoder for anomaly detection in IoT environment and the model used weighted R-score as an alternate to reconstruction based metric to generate anomaly score.

3. PROPOSED MODEL

This section involves about the detailed explanation of components involved in the architecture of the proposed model. In figure 4.1 the overall internal architecture of the proposed model has been illustrated in a detailed manner.

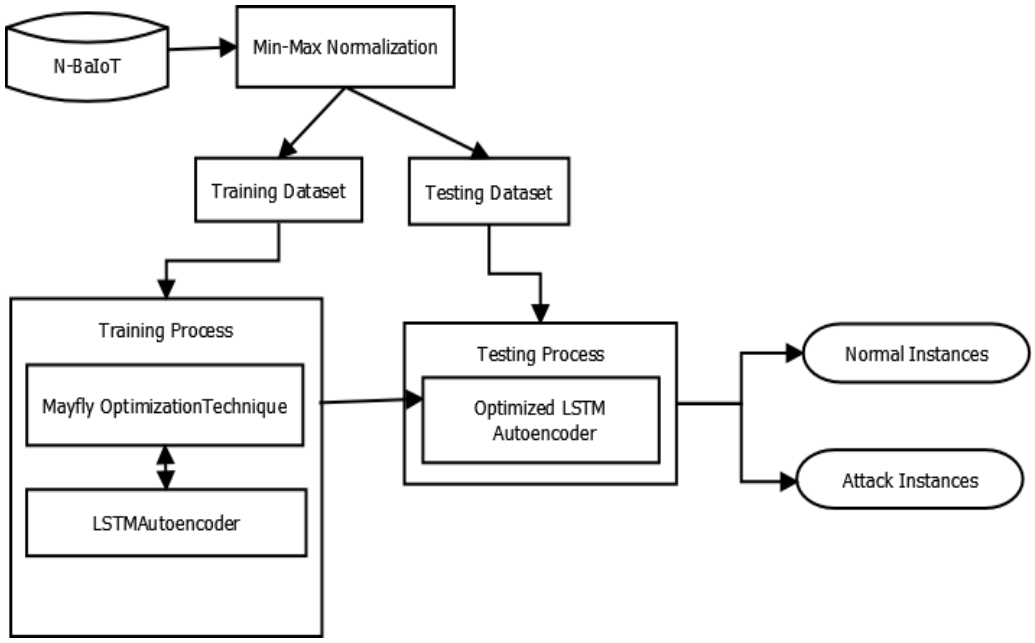

Figure:1 Overall system overview

3.1 Data preprocessing

The data preprocessing step is used to manipulate the input data in such a way to enhance the performance classification of the network traffic done by the deep learning technique. In this work, min-max normalization technique has been utilized here to normalize the values of the N-BaIoT dataset. In min-max normalization, the value is normalized from the minimum value to the maximum value in the range of 0 to 1. The min-max normalization can be calculated using the following equation

$$N' = \frac{N-\min}{\max-\min}(\max' - \min') + \min' \quad (1)$$

Here the variable max and min can be considered as the minimum and maximum values of particular feature correspondingly and the N can be considered as the number of samples present in that particular feature.

3.2 Classification Stage

After performing the data preprocessing process, these instances were classified into normal instances and malicious instances correspondingly. In this work, binary classification was done by implementing LSTM autoencoder. LSTM autoencoder is the type of unsupervised technique which is the combination of LSTM and deep autoencoder algorithms. In this algorithm the architecture of LSTM is designed with an encoder and a decoder.In the encoder part,the input attributes are compress to some extend without losing its quality whereas in the decoder part, the reconstructed input data

restored to the original data with less reconstruction loss.

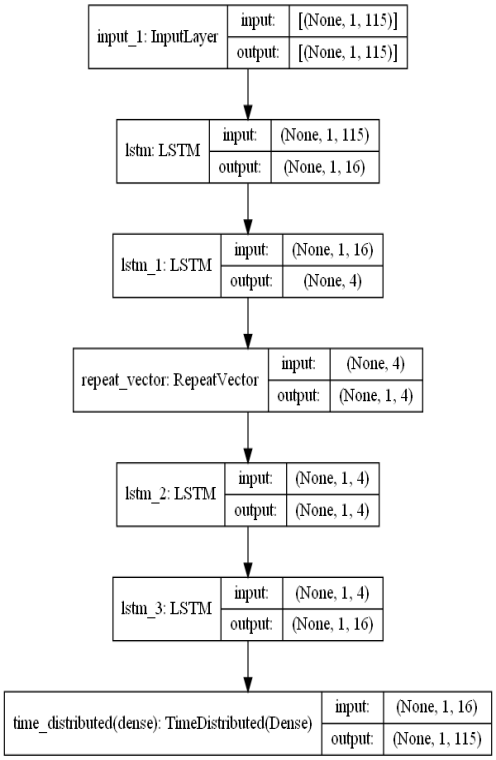

Figure 2 Architecture of LSTM autoencoder

2) In the training stage, the normal data alone is given for training the LSTM autoencoder model.As a result, at the testing stage, while giving both malicious traffic as well as normal traffic as an input to the LSTM autoencoder, the reconstruction error will be high for malicious traffic but the error will be low for normal traffic. The threshold value has to be fixed for reconstruction error to distinguish between malicious and normal traffic. The threshold value can be determined by taking mean value of reconstruction errors occurred during training phase.

3.3 Hyperparameter Tuning

The Mayfly Optimization Technique is a nature-inspired optimization algorithm based on the behavior of mayflies. Mayflies are insects that have a short lifespan and exhibit unique mating and migration behavior. The Mayfly Optimization Technique mimics these characteristics to solve optimization problems.

This optimization technique comprises of many steps they are Initialize Population, Evaluate Fitness, Mating, Migration, Evaluate Migrating Mayflies, Local Search, Update Population, Evaluate Updated Population, Termination Condition and Return Solution. The Mayfly Optimization Technique combines exploration through mating and migration with exploitation through local search to efficiently search the solution space. By iteratively updating the population based on fitness evaluation, it aims to find an optimal or near-optimal solution to the optimization problem. Furthermore, explanation of the mayfly optimization algorithm has been elaborated in algorithm 1.

Algorithm 1 Mayfly Optimization Technique for Hyperparameter Tuning

```
Initialize population of mayflies with random hyperparameters
Evaluate performance of each mayfly
while termination condition not met do
    Sort mayflies based on performance
    Perform mating between top mayflies
    Evaluate performance of offspring
    Select mayflies for migration
    Evaluate performance of migrating mayflies
    Perform local search on selected mayflies' hyperparameters
    Update population by replacing worst mayflies
    Evaluate performance of updated population
end while
return hyperparameters of best mayfly
```

The hyperparameter tuning has been carried out using mayfly optimization technique to fetch the best value for classification using LSTM Autoencoder. The hyperparameter value involved in the autoencoder are number of layers, number of epochs, number of neurons etc., The finalized value has been fixed using mayfly optimization technique has been elucidated in the table 1

Table 1 : Hyper parameter Values

Hyperparameters	**Optimized Values**
Number of Hidden layers	1-input layer,1-output layer and 5-hidden layers
Number of neurons in each layer	16,4,4,4,16,1
Number of epochs	10
Activation Function	relu
Batch Size	60
Loss	Mean absolute error

4. RESULTS AND DISCUSSION

The proposed model was implemented in this study using the Python 3.7 version on a Windows 10 64-bit operating system. The laptop used in this design has 4GB of RAM and a 2.30GHz Intel Core i3 processor. In this section, the evaluation of unsupervised deep learning technique has been used for binary classification using various performance metrics such as accuracy, precision, recall, F1-measure, Training and prediction time, specificity, false negative rate and false positive rate. ROC and reconstruction loss for two class are illustrated visually from figures 3 to 8 for all stages in Mirai attack traffic given for all five IoT devices.In this work, real-time IoT botnet datasets N-BaIoT with normal traffic and affected traffic of 9 IoT devices connected in a network. These datasets capture 23 traffic statistics over five temporal windows 100ms, 500ms, 1.5sec, 10sec, and 1 min) forming 115features .

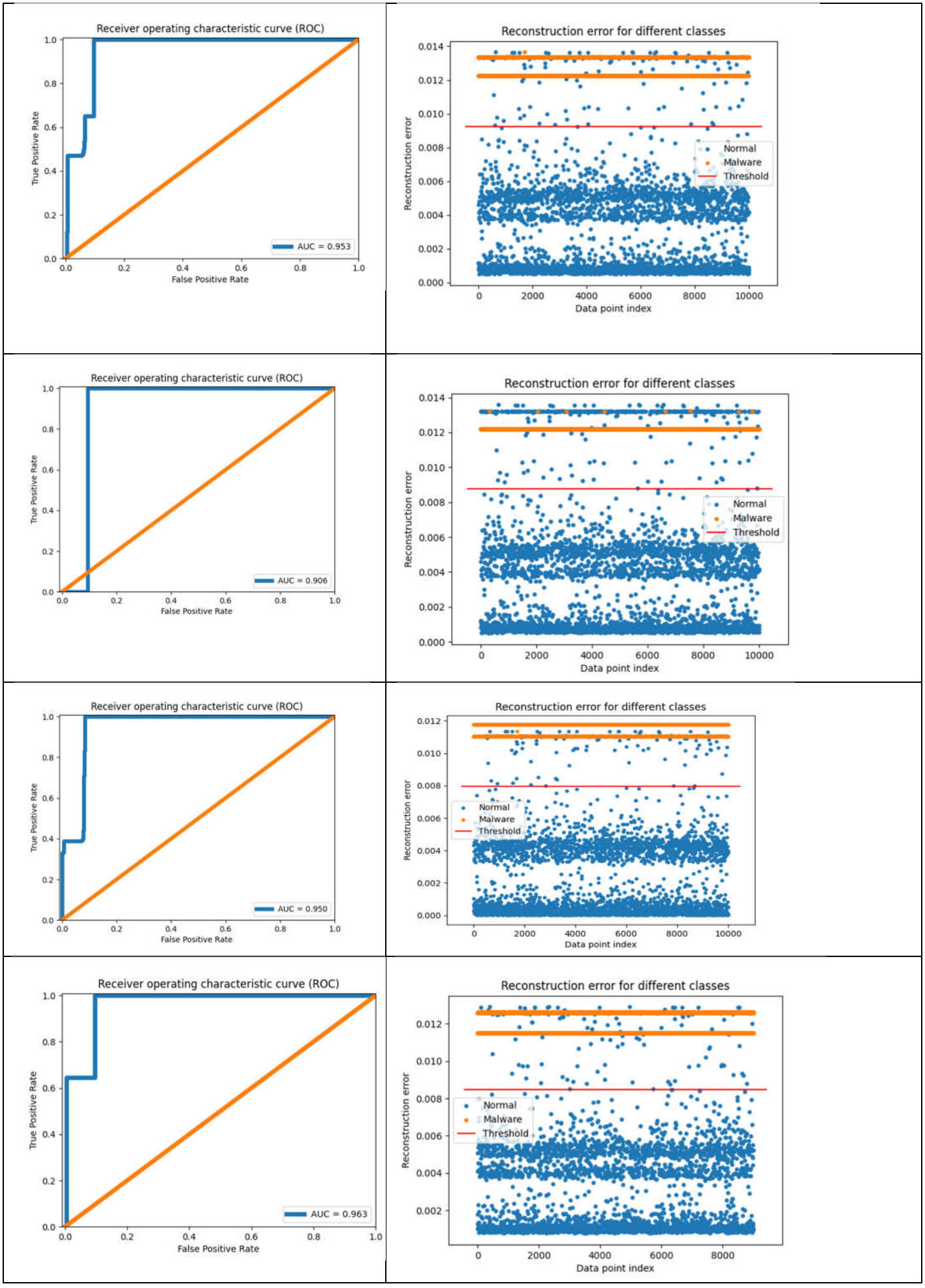

Figure 3 : ROC and Reconstruction error for Danmini doorbell

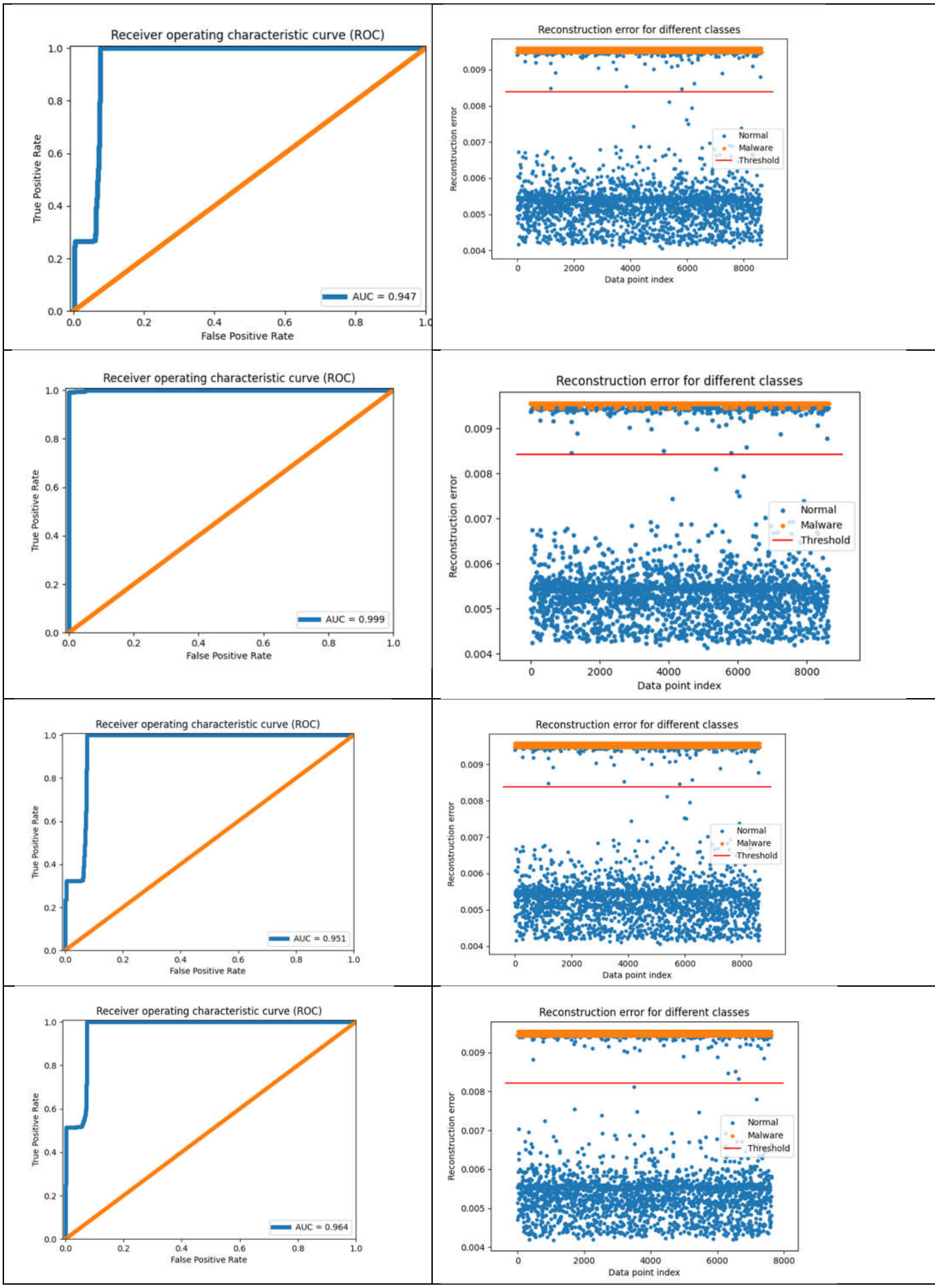

Figure 4 : ROC and Reconstruction error for Ecobee thermostat

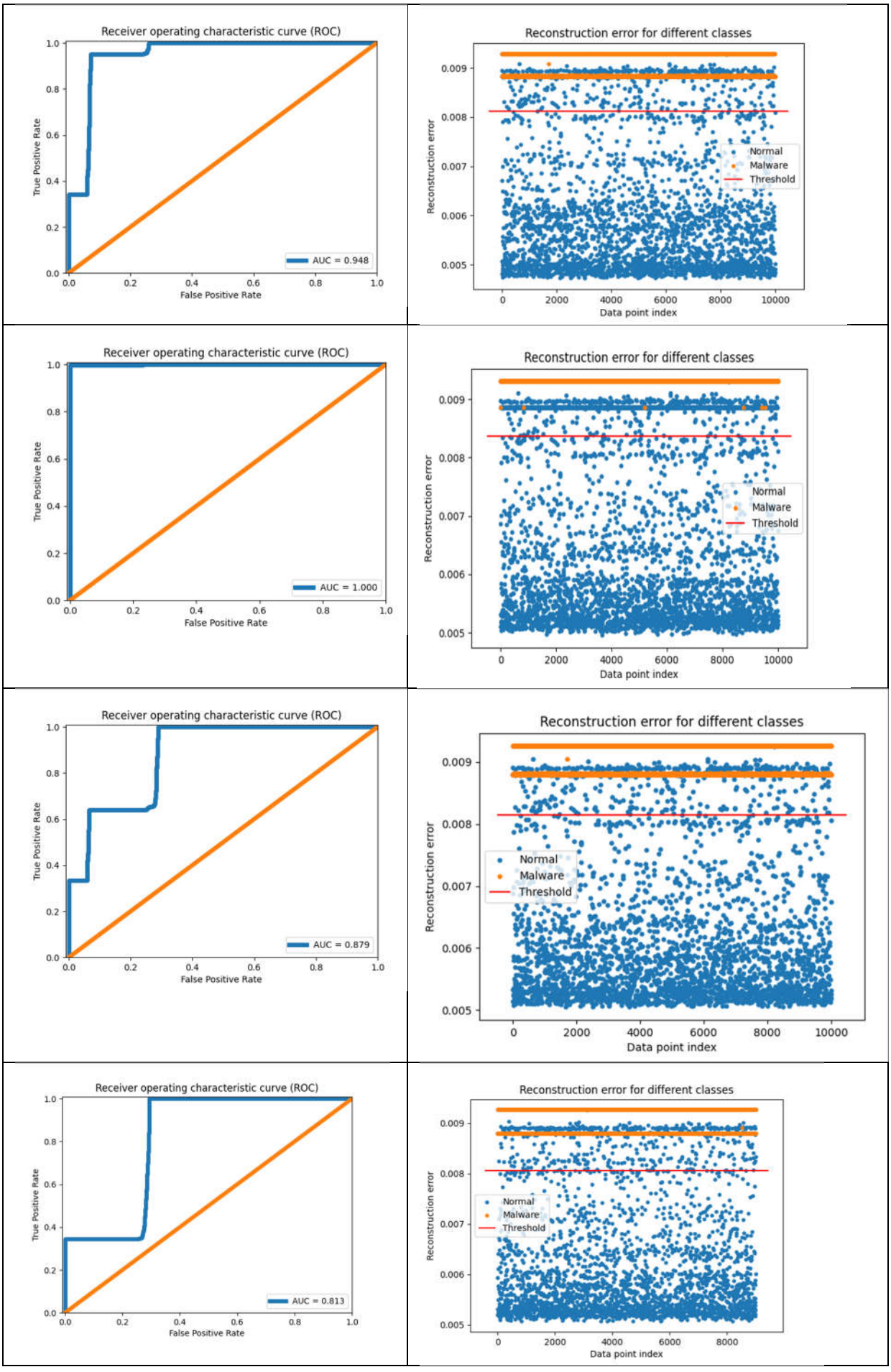

Figure 5: ROC and Reconstruction error for Philips_B120N10_Baby_Monitor

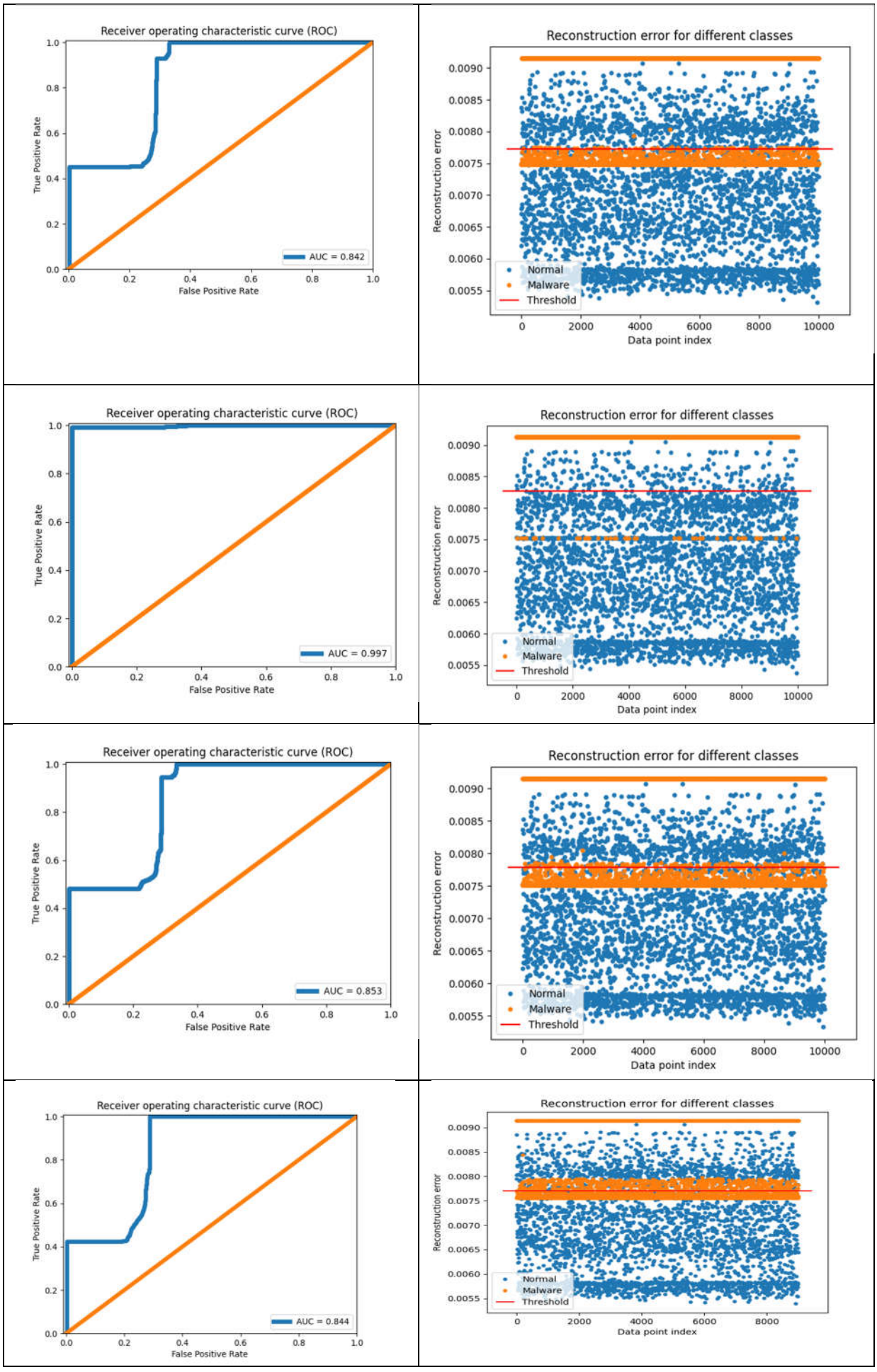

Figure 6: ROC and Reconstruction error for Provision_PT_737E_Security_Camera

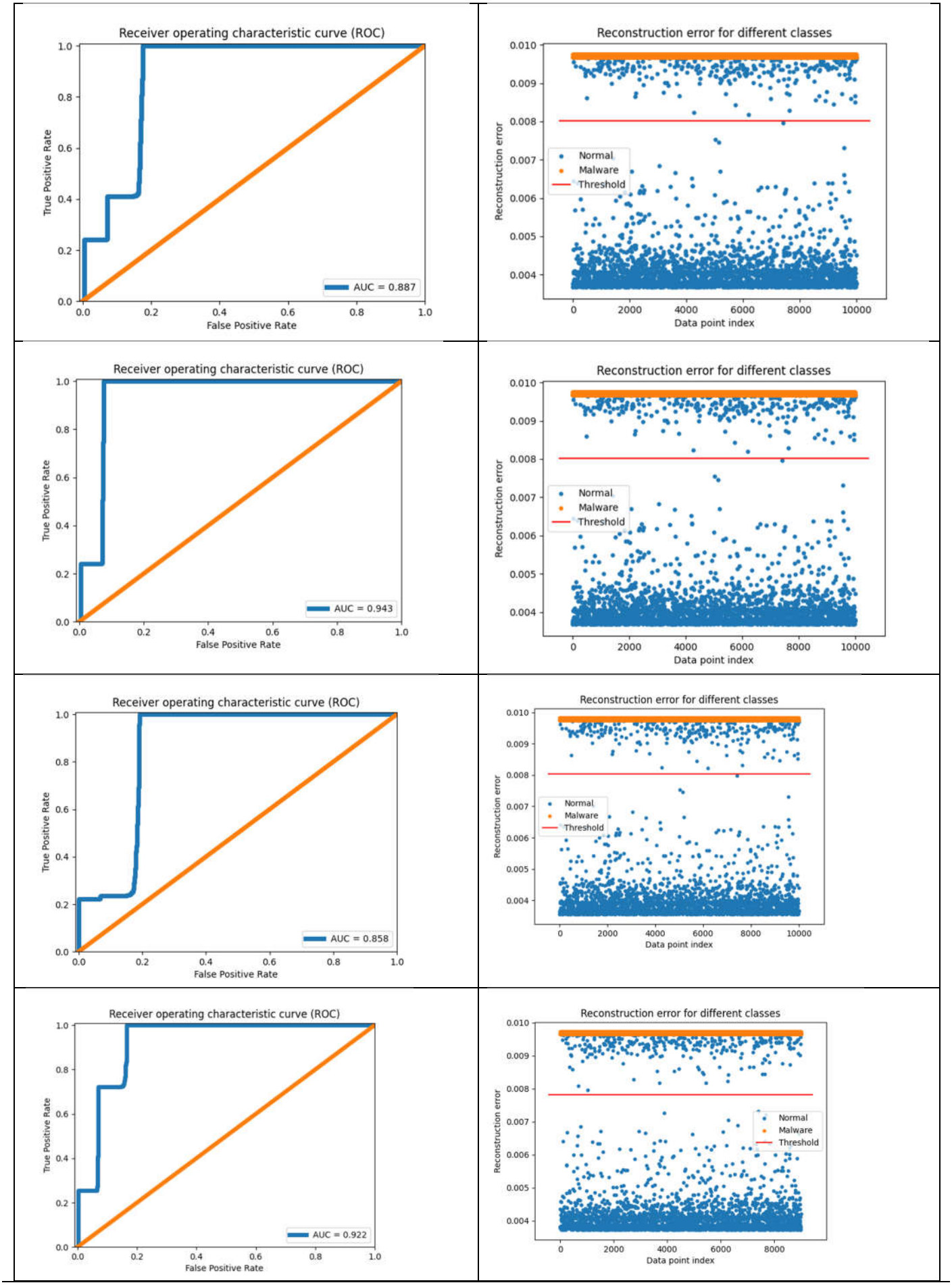

Figure 7: ROC and Reconstruction error for SimpleHome_XCS7_1002_WHT_Security_Camera

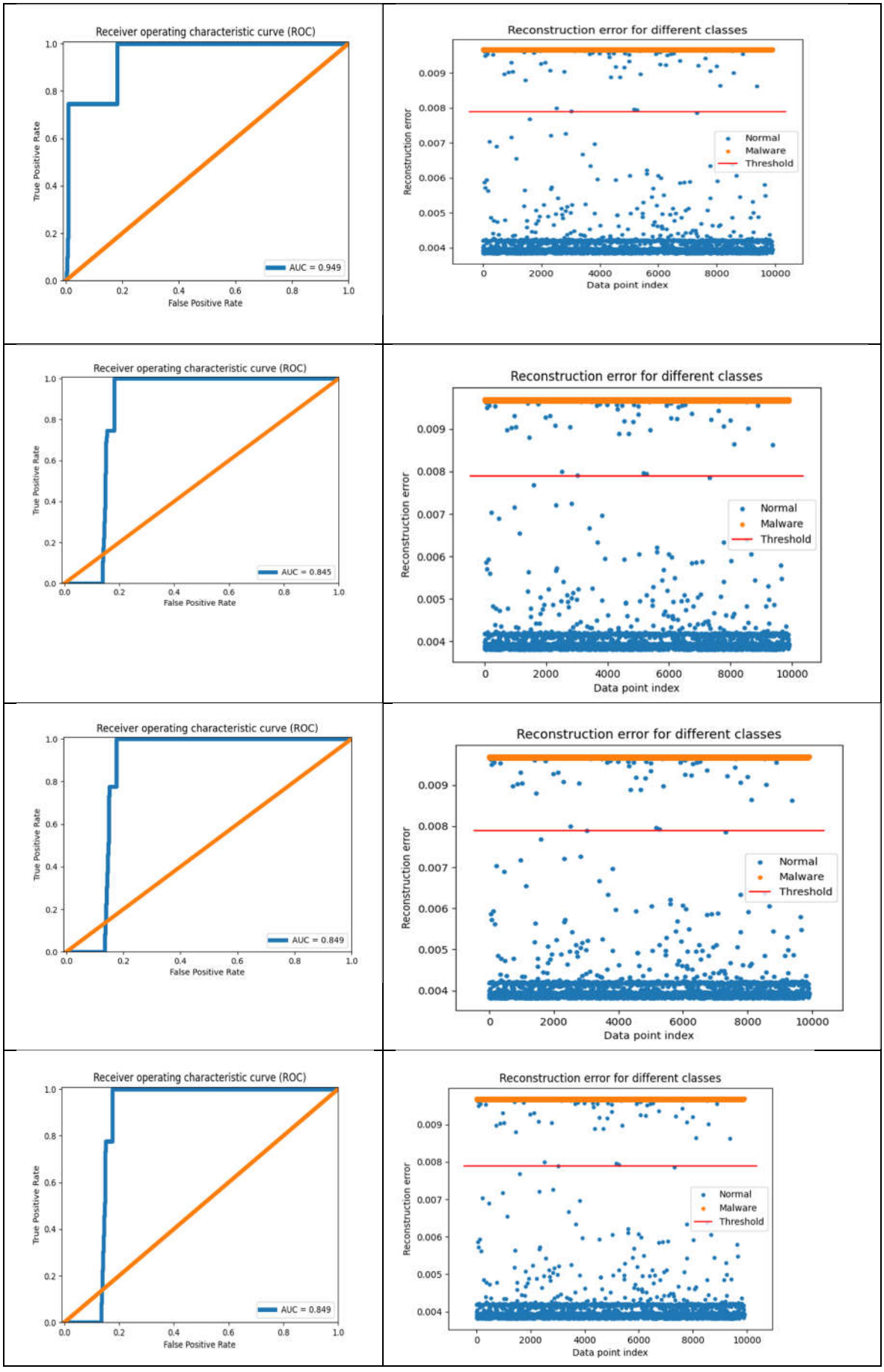

Figure 8 : ROC and Reconstruction error for SimpleHome_XCS7_1003_WHT_Security_Camera

Table 2: Accuracy for detection of Mirai attack at all stages

Device name	Accuracy			
	Ack	Scan	Syn	Udp
Danmini Doorbell	**95.750**	**95.690**	**95.790**	**95.177**
Ecobee Thermostat	95.511	95.511	95.511	95.027
Philips_B120N10_Baby_Monitor	85.890	86.700	86.010	83.977
Provision_PT_737E_Security_Camera	58.290	96.780	61.510	63.611
Provision_PT_838_Security_Camera	56.740	95.320	58.240	63.655
SimpleHome_XCS7_1002_WHT_Security_Camera	89	89	89.030	88.011
SimpleHome_XCS7_1003_WHT_Security_Camera	92.317	92.317	92.317	90.826

Table 3: F1-measure for detection of Mirai attack at all stages

Device name	F1-measure			
	Ack	Scan	Syn	Udp
Danmini Doorbell	0.965	0.965	0.965	0.958
Ecobee Thermostat	**0.968**	**0.968**	**0.968**	**0.963**
Philips_B120N10_Baby_Monitor	0.894	0.899	0.895	0.874
Provision_PT_737E_Security_Camera	0.571	0.973	0.613	0.629
Provision_PT_838_Security_Camera	0.557	0.962	0.575	0.637
SimpleHome_XCS7_1002_WHT_Security_Camera	0.915	0.915	0.915	0.902
SimpleHome_XCS7_1003_WHT_Security_Camera	0.940	0.915	0.915	0.924

Table 4: Precision for detection of Mirai attack at all stages

Device name	Precision			
	Ack	Scan	Syn	Udp
Danmini Doorbell	0.933	0.932	0.934	0.920
Ecobee Thermostat	**0.938**	**0.938**	**0.938**	**0.929**
Philips_B120N10_Baby_Monitor	0.808	0.817	0.809	0.776
Provision_PT_737E_Security_Camera	0.662	0.952	0.682	0.651
Provision_PT_838_Security_Camera	0.650	0.929	0.660	0.6483
SimpleHome_XCS7_1002_WHT_Security_Camera	0.844	0.844	0.844	0.822
SimpleHome_XCS7_1003_WHT_Security_Camera	0.887	0.887	0.887	0.859

Table 5: Recall for detection of Mirai attack at all stages

Device name	Recall			
	Ack	Scan	Syn	Udp
Danmini Doorbell	0.998	0.998	0.998	0.998
Ecobee Thermostat	1.000	0.998	0.998	0.998
Philips_B120N10_Baby_Monitor	**1.000**	**1.000**	**1.000**	**1.000**
Provision_PT_737E_Security_Camera	0.466	0.992	0.511	0.554
Provision_PT_838_Security_Camera	0.457	0.994	0.475	0.574
SimpleHome_XCS7_1002_WHT_Security_Camera	0.978	0.978	0.978	0.978
SimpleHome_XCS7_1003_WHT_Security_Camera	0.978	0.978	0.978	0.978

Table 6: Prediction time for detection of Mirai attack at all stages

Device name	Prediction time			
	Ack	Scan	Syn	Udp
Danmini Doorbell	1.476	1.525	1.399	1.578

Ecobee Thermostat	**0.471**	**0.495**	**0.489**	**0.399**
Philips_B120N10_Baby_Monitor	1.237	1.301	1.288	1.199
Provision_PT_737E_Security_Camera	1.124	1.167	1.171	1.106
Provision_PT_838_Security_Camera	1.136	1.319	1.157	1.596
SimpleHome_XCS7_1002_WHT_Security_Camera	1.499	1.158	1.106	2.916
SimpleHome_XCS7_1003_WHT_Security_Camera	0.721	0.448	0.426	0.397

Table 7: Specifivity time for detection of Mirai attack at all stages

Device name	**Specifivity time**			
	Ack	**Scan**	**Syn**	**Udp**
Danmini Doorbell	**0.894**	**0.893**	**0.895**	**0.891**
Ecobee Thermostat	0.855	0.855	0.855	0.854
Philips_B120N10_Baby_Monitor	0.650	0.670	0.653	0.638
Provision_PT_737E_Security_Camera	0.754	0.992	0.768	0.738
Provision_PT_838_Security_Camera	0.730	0.994	0.740	0.714
SimpleHome_XCS7_1002_WHT_Security_Camera	0.728	0.728	0.728	0.729
SimpleHome_XCS7_1003_WHT_Security_Camera	0.803	0.196	0.196	0.791

Table 8: False negative rate for detection of Mirai attack at all stages

Device name	**False negative rate**			
	Ack	**Scan**	**Syn**	**Udp**
Danmini Doorbell	**0.105**	**0.106**	**0.104**	**0.108**
Ecobee Thermostat	0.144	0.144	0.144	0.145
Philips_B120N10_Baby_Monitor	0.349	0.329	0.346	0.361
Provision_PT_737E_Security_Camera	0.533	0.068	0.488	0.445
Provision_PT_838_Security_Camera	0.542	0.108	0.524	0.425
SimpleHome_XCS7_1002_WHT_Security_Camera	0.271	0.271	0.271	0.270
SimpleHome_XCS7_1003_WHT_Security_Camera	0.196	0.196	0.196	0.208

The above given tables from table 2 to 8 demonstrates about the various performance metrics for each stage occurs during the attack of mirai malware. In table 2 Danmini doorbell achieves higher accuracy for all the stages of Mirai attack. In table 3 and 4 Ecobee thermostat secures higher value for F1-measure and precision respectively. In table 5 Provision_PT_737E_Security_Camera, Provision_PT_838_Security_Camera alone secures very less recall values except this remaining devices holds the higher recall value for classification..In table 7and 8 Danmini doorbell device achieves very high specificity and low false negative rate than the other devices.

CONCLUSION

In conclusion, combination of two powerful techniques has been implemented to improve our intrusion detection system (IDS). The first technique is called the Mayfly optimization, which helps us fine-tune the system for better performance. It explores different settings to find the best ones, making our IDS smarter and more effective at detecting attacks. The second technique is the LSTM autoencoder, which is like a smart filter for network data. It learns and remembers normal network patterns, so it can spot abnormal activities easily. This helps us detect new and unknown attacks, providing better security. By combining these two techniques, the proposed IDS becomes more accurate and efficient. It can detect even subtle anomalies and reduce mistakes in identifying threats. This makes our system reliable and adaptive to new types of attacks. However, this IDS still face some challenges, like the computational cost of the Mayfly optimization. Hence it is necessary to find ways to make it faster for larger networks. Also, it must keep updating the IDS with the latest information about new threats to stay effective. In conclusion, integrated intrusion detection system with Mayfly optimization and LSTM autoencoder is a significant step forward in network security. It can detect and respond to cyber threats effectively, making the digital world safer for everyone. With further improvements and research, it can make our system even better at protecting networks and data from

harmful intrusions.

References

[1] Farahnakian, F., & Heikkonen, J. (2018, February). A deep auto-encoder based approach for intrusion detection system. In *2018 20th International Conference on Advanced Communication Technology (ICACT)* (pp. 178-183). IEEE.

[2] Zhang, H., Wu, C. Q., Gao, S., Wang, Z., Xu, Y., & Liu, Y. (2018, August). An effective deep learning based scheme for network intrusion detection. In *2018 24th International Conference on Pattern Recognition (ICPR)* (pp. 682-687). IEEE.

[3] Aygun, R. C., & Yavuz, A. G. (2017, June). Network anomaly detection with stochastically improved autoencoder based models. In *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)* (pp. 193-198). IEEE.

[4] Bovenzi, G., Aceto, G., Ciuonzo, D., Persico, V., & Pescapé, A. (2020). A hierarchical hybrid intrusion detection approach in iot scenarios.

[5] Wang, W., Zhao, M., & Wang, J. (2019). Effective android malware detection with a hybrid model based on deep autoencoder and convolutional neural network. *Journal of Ambient Intelligence and Humanized Computing*, *10*(8), 3035-3043.

[6] Naseer, S., Saleem, Y., Khalid, S., Bashir, M. K., Han, J., Iqbal, M. M., & Han, K. (2018). Enhanced network anomaly detection based on deep neural networks. *IEEE access*, *6*, 48231-48246.

[7] Muna, A. H., Moustafa, N., & Sitnikova, E. (2018). Identification of malicious activities in industrial internet of things based on deep learning models. *Journal of information security and applications*, *41*, 1-11.

[8] Li, X., Chen, W., Zhang, Q., & Wu, L. (2020). Building auto-encoder intrusion detection system based on random forest feature selection. *Computers & Security*, *95*, 101851.

[9] Gharib, M., Mohammadi, B., Dastgerdi, S. H., & Sabokrou, M. (2019). Autoids: auto-encoder based method for intrusion detection system. *arXiv preprint arXiv:1911.03306*.

[10] Qureshi, A. S., Khan, A., Shamim, N., & Durad, M. H. (2019). Intrusion detection using deep sparse auto-encoder and self-taught learning. *Neural Computing and Applications*, 1-13.

[11] Chawla, A., Jacob, P., Lee, B., & Fallon, S. (2019). Bidirectional LSTM autoencoder for sequence based anomaly detection in cyber security. *International Journal of Simulation--Systems, Science & Technology*.

[12] Ashraf, J., Bakhshi, A. D., Moustafa, N., Khurshid, H., Javed, A., & Beheshti, A. (2020). Novel deep learning-enabled LSTM autoencoder architecture for discovering anomalous events from intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, *22*(7), 4507-4518.

[13] Mahmoud, M., Kasem, M., Abdallah, A., & Kang, H. S. (2022, July). AE-LSTM: Autoencoder with LSTM-Based Intrusion Detection in IoT. In *2022 International Telecommunications Conference (ITC-Egypt)* (pp. 1-6). IEEE.

[14] Ketepalli, G., & Bulla, P. (2022, June). Feature Extraction using LSTM Autoencoder in Network Intrusion Detection System. In *2022 7th International Conference on Communication and Electronics Systems (ICCES)* (pp. 744-749). IEEE

[15] Sumathi L , Valarmathi D.L (2024 , January) . Network based Anomaly detection using Self-defined One Class Support Vector Machine with Grey Wolf Optimizer for Internet of Things Environment. In Research Square on Jan 2024 . Available online : https://doi.org/10.21203/rs.3.rs-3844635/v1

[16] Sumathi L , Valarmathi D.L (2024 , January) . Towards securing the edge devices using Lightweight A 3E : Apposite Adversarial Autoencoder on Jan 2024 . Available online : https://doi.org/10.21203/rs.3.rs-3844635/v1