Adaptive Approach to Anomaly Detection in Internet of Things using Autoencoders and Dynamic Thresholds

[1]Nayer Tumi Figueroa E, [2]Vishnu Priya A, [3]Selvanayaki Kolandapalayam Shanmugam, [4]Kiran Kumar V, [5]Sudhakar Sengan and [6]Alexandra Melgarejo Bolivar C
[1]Universidad Nacional del Altiplano de Puno, P.O. Box 291, Puno – Peru.
[2]Department of Computational Intelligence, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore 632014, TamilNadu, India.
[3]Department of Computer Science, Ashland University, Ashland, OH, USA.
[4]Department of Computer Science, Dravidian University, Andhra Pradesh 517426, India.
[5]Department of Computer Science and Engineering, PSN College of Engineering and Technology, Tirunelveli 627451, TamilNadu, India.
[6]Universidad Nacional del Altiplano de Puno, P.O. Box 291, Puno – Peru.
[1]nayer.tumi@unap.edu.pe, [2]a.vishnupriya@vit.ac.in, [3]skolanda@ashland.edu, [4]kirankumar.v@rediffmail.com, [5]sudhasengan@gmail.com, [6]cmelgarejo@est.unap.edu.pe

Correspondence should be addressed to Sudhakar Sengan : sudhasengan@gmail.com.

Article Info
Journal of Machine and Computing (http://anapub.co.ke/journals/jmc/jmc.html)
Doi: https://doi.org/10.53759/7669/jmc202404001
Received 10 May 2023; Revised from 20 August 2023; Accepted 24 September 2023.
Available online 05 January 2024.

Abstract – The Internet of Things (IoT) represents a vast network of interconnected devices, from simple sensors to intricate machines, which collect and share data across sectors like healthcare, agriculture, and home automation. This interconnectivity has brought convenience and efficiency but also introduced significant security concerns. Many IoT devices, built for specific functions, may lack robust security, making them vulnerable to cyberattacks, especially during device-to-device communications. Traditional security approaches often fall short in the vast and varied IoT landscape, underscoring the need for advanced Anomaly Detection (AD), which identifies unusual data patterns to warn against potential threats. Recently, a range of methods, from statistical to Deep Learning (DL), have been employed for AD. However, they face challenges in the unique IoT environment due to the massive volume of data, its evolving nature, and the limitations of some IoT devices. Addressing these challenges, the proposed research recommends using autoencoders with a dynamic threshold mechanism. This adaptive method continuously recalibrates, ensuring relevant and precise AD. Through extensive testing and comparisons, the study seeks to demonstrate the efficiency and adaptability of this approach in ensuring secure IoT communications.

Keywords – Internet of Things, Anomaly Detection, Cyber Attacks, Autoencoders, Security, Accuracy.

I. INTRODUCTION

The Internet of Things (IoT) has shifted the technology landscape, resulting in a broad interconnectedness. An array of devices, from sensors to more advanced machines, collaborates across domains such as healthcare, agriculture, urban planning, and home automation [1]. These devices' capacity to gather, share, and analyze data has positioned IoT as an influential part of the evolving digital realm. However, the adoption of IoT is not without challenges. The wide range of IoT devices presents potential vulnerabilities, with challenges in ensuring consistent security and privacy. Notably, many IoT devices, tailored for specific roles and sometimes having limited computational capabilities, might not have the necessary security mechanisms, exposing them to potential threats [2].

Communication between devices is a notable issue within the IoT realm. While these communications involve routine data exchanges, they are essential to the IoT framework. However, they can be targets for cyber-attacks, including man-in-themiddle, denial-of-service, and eavesdropping [3]. Given the expansive nature of the IoT network, traditional security methods, which often depend on recognized threat signatures, might fall short. This highlights the importance of advanced Anomaly Detection (AD) in IoT. Such detection can pinpoint unusual patterns, acting as an alert system against emerging or unfamiliar threats. As cyber threats change and grow, there is an apparent demand for flexible and expandable security measures [4].

In recent years, the domain of AD has witnessed a plethora of approaches, ranging from traditional statistical methods to Machine Learning (ML) techniques and further to advanced Deep Learning (DL) methodologies [5]. These methods, although effective in specific scenarios, often grapple with challenges introduced by the unique landscape of the IoT. The sheer scale of data, its high dimensionality, the evolving nature of IoT data patterns, the scarcity of labelled anomalies for supervised techniques, and the high false alarm rates stand as significant obstacles. Moreover, deploying sophisticated models on IoT devices, which are typically resource-constrained, adds another layer of complexity [6].

Recognizing these challenges, the proposed work is anchored in the strengths of autoencoders. The work seeks to augment them with a dynamic threshold mechanism, striving to offer a solution that is not only scalable and adaptive to the fluid nature of IoT data but also astute in reducing false alarms and ensuring ease of deployment. The proposed work recognizes the dynamic nature of IoT data and proposes a novel threshold determination method that is both adaptive and effective [7]. Going beyond static thresholds, which may not capture the evolving nuances of IoT data, the proposed method continuously recalibrates itself, ensuring that the AD remains relevant and precise. Through rigorous experimentation using benchmark datasets and comparison with existing models, the work aims to display the proposed approach's efficacy, resilience, and adaptability, offering a comprehensive solution for securing IoT communications in this ever-connected world.

The paper is organized as follows: Section 2 presents the literature review, Section 3 presents the methods, Section 4 presents the proposed model, Section 5 presents the experimental analysis, and Section 6 concludes with future work.

II. LITERATURE REVIEW

In the realm of IoT security, considerable emphasis has been placed on the utilization of ML and DL methods for AD. [8-10] researched device type identification using network traffic analysis. They adopted supervised learning to understand characteristics from network packets and then delved into unsupervised learning methods, such as One-class SVM, Isolation Forest, and autoencoders, for dimensionality reduction. Their application of autoencoders with the Modbus TCP protocol yielded an impressive F1 score of 98.36%. Similarly, [11-13] introduced an unsupervised Attention-based ConvLSTM Autoencoder with a Dynamic Thresholding (ACLAE-DT) framework for handling multivariate time series anomalies. By processing the data and creating feature images, they utilized an attention-based ConvLSTM autoencoder to discern temporal behaviors, achieving AD through dynamic thresholding of reconstruction errors [14-15].

Diverging slightly, [16-18] focused on the data reconstruction error of autoencoders. Their innovative approach perceived the reconstruction error as a vector rather than a singular value, enabling a granular AD method. [19-20] centered their research around autoencoders, aiming to detect malicious nodes in IoT. Their methodology involved dimensional reduction through autoencoders and subsequent clustering in low-dimensional space using the Bayesian Gaussian mixture model, resulting in a 99% accuracy rate in distinguishing traffic types [21-22].

Furthermore, [23-25] developed the Autoencoder Deep Neural Network (AENN) that classifies transmission outcomes, predicting intermediate attacks and ensuring a 99.02% detection rate, emphasizing the potential of autoencoders in tackling IoT security challenges.

III. METHODS

Autoencoder Architecture

Autoencoders represent a specialized category within neural networks designed to capture compact representations of input data. These networks are structured around two central components: the Encoder and the Decoder. The Encoder transforms the input data from its original space into a concise, lower-dimensional representation. Mathematically, this transformation can be denoted as EQU (1)

$$f: \mathbb{R}^{d_{in}} \rightarrow \mathbb{R}^{d_{code}} \tag{1}$$

where d_{in} is the dimensionality of the input data and d_{code} signifies the dimensionality of the latent or compressed space. Conversely, the Decoder operates to revert this transformation, aiming to reproduce the input data from its compressed form. This reverse transformation is expressed as EQU (2)

$$g: \mathbb{R}^{d_{code}} \rightarrow \mathbb{R}^{d_{in}} \tag{2}$$

The input layer of the autoencoder handles data with a dimensionality of d_{in}. As data advances through the architecture, it encounters hidden layers. For instance, the first hidden layer could be composed of n_1 neurons, applying an activation function $\sigma_1(x)$. Subsequent layers continue similarly, each potentially employing different neuron counts and activation functions.

Activation functions such as the Rectified Linear Unit (ReLU), Sigmoid, and Tanh are crucial, as they introduce non-linear transformations, allowing the network to capture intricate patterns within the data. A significant aspect of the autoencoder's design is the latent space, a condensed input representation. The goal during training is to minimize the discrepancy between the original input and its reconstruction by the autoencoder. This discrepancy or reconstruction error is quantified through metrics like the Mean Squared Error (MSE): EQU (3)

$$L = \frac{1}{N}\sum_{i=1}^{N} (x_i - \hat{x}_i)^2 \tag{3}$$

For binary data, the Binary Cross-Entropy metric is suitable: EQU (4)

$$L = -\frac{1}{N}\sum_{i=1}^{N} [x_i \log(\hat{x}_i) + (1 - x_i)\log(1 - \hat{x}_i)] \tag{4}$$

Training involves refining network weights using optimization algorithms like Adam, SGD, or RMSprop. The algorithm choice often hinges on data properties and encounters training challenges. Given the propensity of neural networks to overfit, regularization strategies, including dropout, weight decay, and early stopping, are integrated to ensure model robustness. In **Fig 1**.

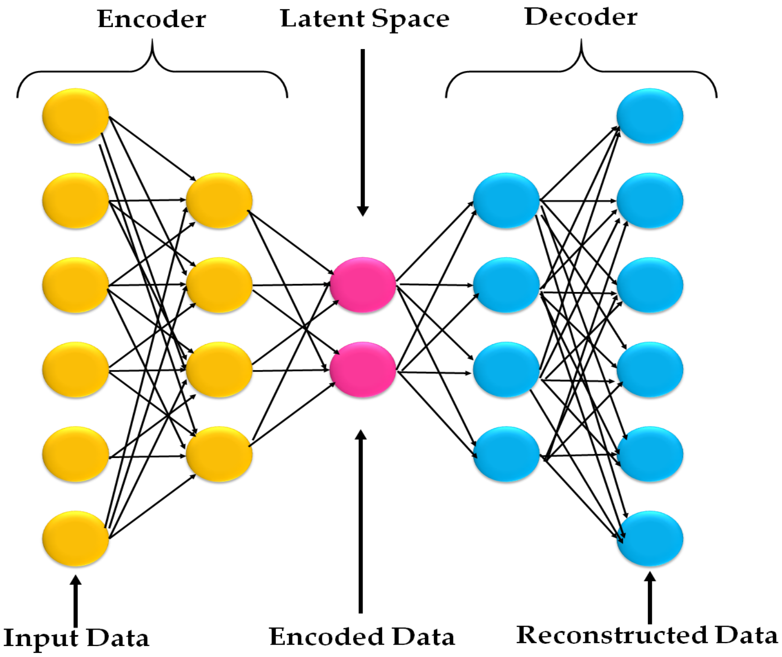


Fig 1. Autoencoder Structure

AE for Anomaly Detection

AD in loT device communication involves identifying sequences or patterns significantly diverge from expected behavior. Such anomalies can originate from various sources, including malicious attacks, equipment malfunctions, or unpredictable data transmissions. Autoencoders (AE), a specific type of neural network, have emerged as a potent tool due to their inherent ability to learn, compress, and reconstruct data patterns.

Given an input sequence represented as $x = \{x_1, x_2, \ldots, x_t\}$, where each x_i denotes a data point (possibly multi-dimensional) encapsulating features of loT device communications, the AE processes this input to produce a reconstructed sequence $\hat{x} = \{\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_t\}$. The fidelity of this reconstruction is pivotal. If the AE is proficiently trained on 'normal' data, it should ideally reproduce such sequences with minimal discrepancies. However, for anomalous sequences, the difference or reconstruction error for each point is given by $e_i = (x_i - \hat{x}_i)^2$. The MSE across the entire sequence is then calculated as EQU (5)

$$MSE = \frac{1}{t}\sum_{i=1}^{t} e_i \tag{5}$$

To distinguish between minor deviations and genuine anomalies, we introduce an anomaly threshold, denoted as θ. If the computed MSE for a sequence exceeds θ, that sequence is classified as anomalous. The process of determining an optimal θ is crucial, ensuring that the system is neither too lenient nor overly strict. At the core of the AE's operation lies the latent representation, a compressed form of the input sequence realized within the AE's encoder. While this representation is adept at encapsulating key data features, its direct interpretability remains challenging. This lack of transparency can sometimes complicate understanding anomalies based purely on the AE's outputs.

IV. PROPOSED MODEL

Novel Detection Threshold Determination

Establishing an optimal threshold for AD is critical in deploying autoencoder-based models. While many traditional methods opt for static thresholds or derive from domain-specific insights, the dynamic nature of loT data necessitates a more adaptive technique. Central to this proposed method is the acknowledgement that, despite the evolving nature of loT data, the statistical properties of the reconstruction errors remain invaluable for threshold determination. Consider the set of all reconstruction errors observed during the validation phase, denoted by M, where $\mathrm{m_i}$ represents the reconstruction error for the ith sample: EQU (6)

$$\mathrm{M} = \{\mathrm{m_1}, \mathrm{m_2}, \dots, \mathrm{m_n}\} \tag{6}$$

From this set, the mean of the errors, represented as μ, is calculated as EQU (7)

$$\mu = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{m_i} \tag{7}$$

Simultaneously, gauging the spread and variability of these errors provides insights into their dispersion. This variability is encapsulated using the standard deviation: EQU (8)

$$\sigma = \sqrt{\frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} (\mathrm{m_i} - \mu)^2} \tag{8}$$

From these statistical properties, a dynamic threshold, θ, is determined by EQU (9)

$$\theta = \mu + \alpha \cdot \sigma \tag{9}$$

Here, α functions as a sensitivity parameter. A higher α results in a more conservative threshold, leading to fewer ADs, while a lower value amplifies the system's sensitivity. Recognizing the time-sensitive nature of loT data, periodic recalibration of the threshold becomes imperative. If t signifies the current time and T represents a predetermined recalibration period, the threshold undergoes re-evaluation each time $\mathrm{tmodT} = 0$. This involves re-computing both μ and σ based on recent data, followed by an update to θ using the dynamic threshold formula.

The advantages of this novel threshold determination are multifold. It offers adaptability in response to changing data patterns, ensuring continuous relevance, and the flexibility afforded by the sensitivity parameter α permits system-specific fine-tuning. Furthermore, the periodic recalibrations underpin a system that is consistently updated, reducing the likelihood of both false negatives and positives. This dynamic and adaptive threshold determination approach enhances the efficacy of autoencoders in AD for loT device communication, promoting a more resilient and reactive system.

AE-Based Anomaly Detection Using the Proposed Detection Threshold

Before employing the proposed AE learning model, the input data are preprocessed. This entails the handling of sporadic transmissions by using domain-informed imputation, normalizing data using Min-Max scaling, transforming the time-series data into overlapping sequences, reducing high-frequency noise through smoothing functions, and ensuring dimensionality compatibility with the autoencoder's input layer. Once the data is preprocessed, the detection model comes into action. Upon receiving the preprocessed input sequence $\mathrm{X} = \{\mathrm{x_1}, \mathrm{x_2}, \dots, \mathrm{x_t}\}$ that captures various loT data points, the encoder component of the AE processes this dataset. This phase is pivotal as it transforms the raw data, yielding a latent representation denoted by EQU (10)

$$\mathrm{Z} = \mathrm{f(X)}, \tag{10}$$

where f represents the encoding function mapping the input sequence X into its latent counterpart Z. Following this transformation, the decoder comes into play. Its primary role is to reverse-engineer the encoding process, striving to recreate the original data from the condensed latent information. This reconstruction phase is captured by EQU (11)

$$\hat{\mathrm{X}} = \mathrm{g(Z)}, \tag{11}$$

where g stands as the decoding function, orchestrating the conversion of the latent sequence Z back into the reconstructed sequence $\hat{X}$.

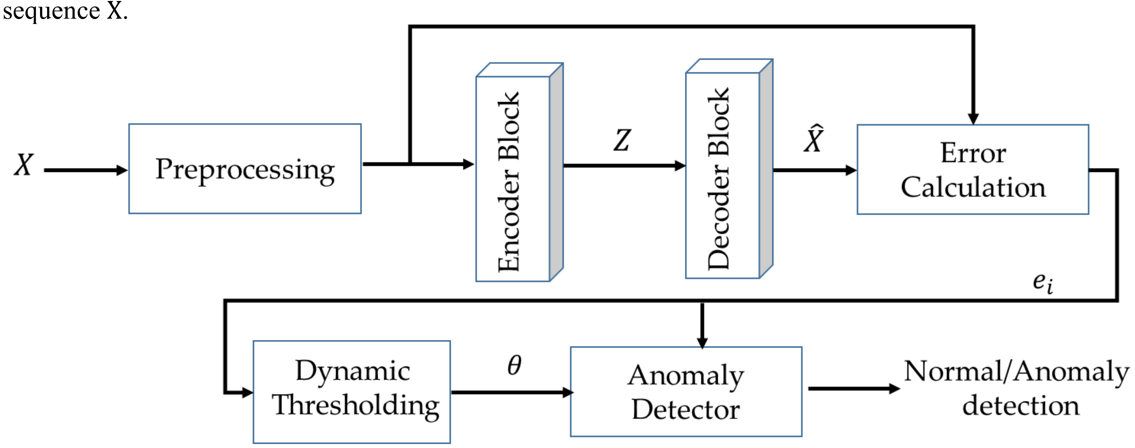


Fig 2. Proposed AD architecture

The crux of AD rests on assessing the fidelity of the reconstruction in **Fig 2**. For every original data point x_i in X and its corresponding reconstructed counterpart $\hat{x}_i$ in $\hat{X}$, a reconstruction error is computed. This error measures the disparity between the original and the reconstructed data and is typically expressed as EQU (12)

$$e_i = (x_i - \hat{x}_i)^2 \tag{12}$$

The culmination of these individual errors aids in identifying anomalies. If the MSE of a sequence surpasses the previously derived dynamic threshold θ, the sequence is flagged as anomalous. The strength of this method lies in its adaptability; the threshold is continuously coordinated with the recent patterns in IoT data, courtesy of its dynamic nature. By harmonizing the AE's reconstruction faculties with this nimble thresholding technique, the system maintains precision and is agile, positioning it as a vital instrument in IoT AD.

```
Algorithm: AE-Based AD with Dynamic Threshold
Input:
    • Data sequence $X = \{x_1, x_2, \dots, x_t\}$
    • Autoencoder (AE) with encoder function f and decoder function g
    • Dynamic threshold formula $\theta = \mu + \alpha \cdot \sigma$
Output:
    • AD result for each data point in X
Steps:
    Initialization:
        Load the pre-trained AE model.
        Set sensitivity parameter α based on desired system responsiveness.
    Data Encoding:
        For each data point $x_i$ in X, obtain the latent representation: $z_i = f(x_i)$
    Data Decoding:
        For each latent representation $z_i$, reconstruct the original data point: $\hat{x}_i = g(z_i)$
    Compute Reconstruction Error:
        For each pair $x_i$ and $\hat{x}_i$, calculate the squared reconstruction error: $e_i = (x_i - \hat{x}_i)^2$
    Calculate MSE:
        Obtain the mean error across the sequence: $MSE = \frac{1}{t}\sum_{i=1}^{t} e_i$
    Determine Dynamic Threshold:
        Calculate the mean μ and standard deviation σ of the reconstruction errors for recent data points.
        Update the threshold θ using: $\theta = \mu + \alpha \cdot \sigma$
    Anomaly Detection
        For Each $e_i$ in the sequence:
```

```
        If $e_i > \theta$, mark $x_i$ as an anomaly.
        Else, mark $x_i$ as normal.
  **Return Results**
        Provide a list indicating AD results for each data point in X.
```

V. EXPERIMENTAL ANALYSIS

Dataset Used

The experiments primarily utilized the N-BaIoT dataset [7], a widely recognized dataset in the field of IoT security research. This dataset from Ben-Gurion University encompasses network traffic from various smart devices, thus representing typical household IoT communication patterns. It contains traces of benign and malicious activities, making it suitable for AD tasks. The dataset consists of features like packet size, transmission rate, source and destination IP addresses, and protocol type, among others.

System Configuration

The experiments were conducted on a workstation with an Intel Core i9-10900K CPU, 64 GB RAM, and an NVIDIA RTX 3090 GPU. The software environment was set up with Python 3.8, TensorFlow 2.5, and Scikit-learn 0.24.

Metrics Employed

To evaluate the efficacy of the proposed AD model, the following metrics were employed:

- **Accuracy:** Proportion of correctly predicted instances over total instances.
- **Precision:** Proportion of correctly predicted anomalies over total predicted anomalies.
- **Recall (Sensitivity):** Proportion of correctly predicted anomalies over actual anomalies in the dataset.
- **F1-Score:** Harmonic meaning of Precision and Recall, giving a balanced measure.

Baseline Models for Comparison

The proposed AE-based model was compared against the following baseline models to ascertain its relative performance:

- **Isolation Forest:** An ensemble-based method effective for detecting outliers.
- **One-Class SVM:** A kernel-based method tailored for novelty detection.
- **Local Outlier Factor (LOF):** Measures the local deviation of the density of a given sample concerning its neighbors.
- **DL-based Classifier:** A simple feed-forward neural network trained explicitly for the AD task.

By comparing the proposed model's performance against these baseline models, a comprehensive understanding of its effectiveness and potential advantages in IoT-AD was ascertained.

Training The Proposed Model

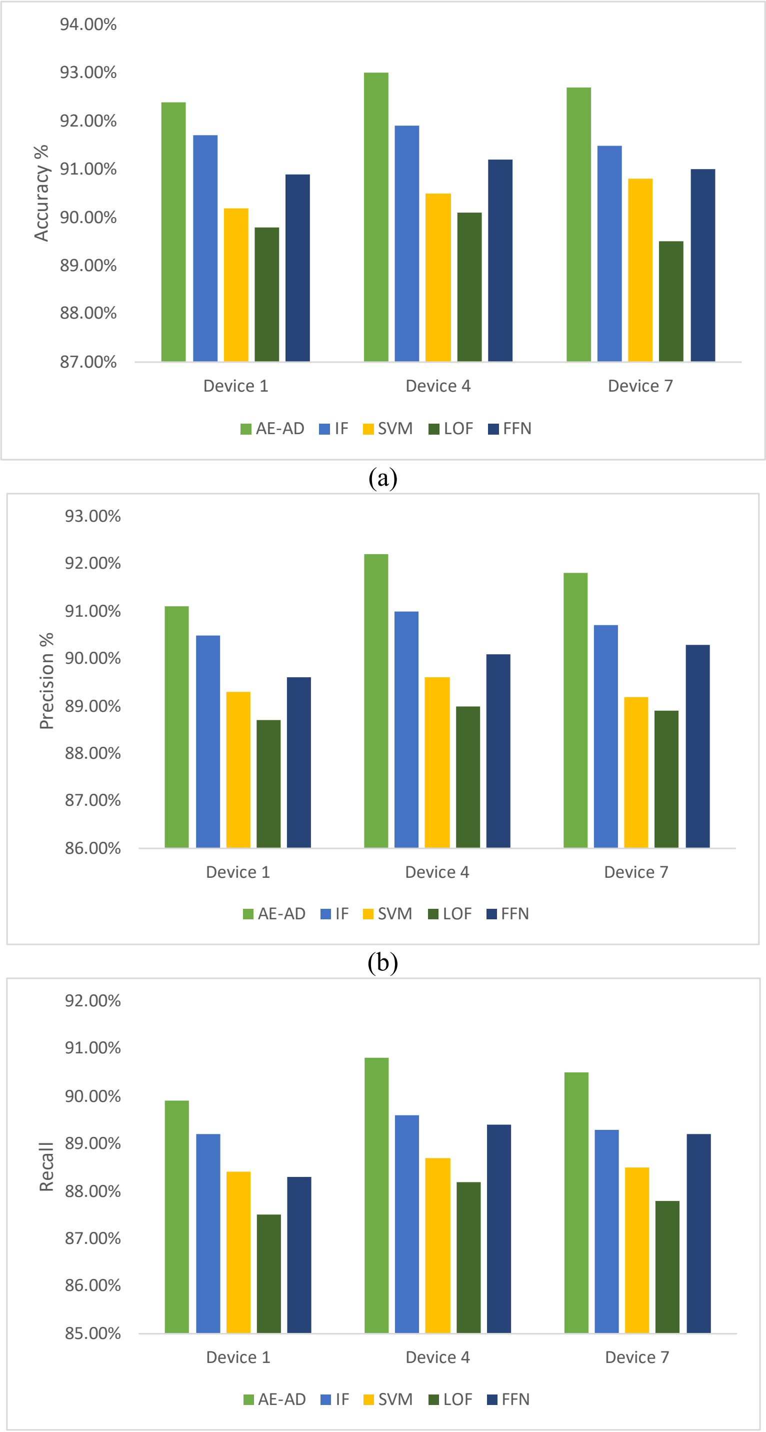
The proposed AE-based AD model was trained using the N-BaIoT dataset. The dataset was split such that 80%, equivalent to 1.6 million instances, was used for training, containing 'normal' IoT communication patterns to ensure that the autoencoder learns a proper representation. The remaining 20% (400,000 instances) was set aside for testing and included normal and anomalous patterns to evaluate the model's ability to detect anomalies.

For training, a set of hyperparameters was chosen to optimize performance. A learning rate of 0.001 was used to ensure a balanced convergence rate. The model was trained with a batch size of 256 for computational efficiency and to capture the nuances in the data. The training was performed over 50 epochs, which was determined to be where the validation loss began to plateau. The Adam optimizer was employed for its adaptability, and the primary loss function used was the MSE to measure the difference between the original and reconstructed data. To prevent overfitting and enhance the model's generalization capability, dropout regularization was applied with a rate of 0.5. See the **Table 1.**

Table 1. Comparative Performance Metrics across IoT Devices for Various Models

Metrics / Device	AE-AD	IF	SVM	LOF	FFN
			Device 1		
Accuracy	92.4%	91.7%	90.2%	89.8%	90.9%
Precision	91.1%	90.5%	89.3%	88.7%	89.6%
Recall	89.9%	89.2%	88.4%	87.5%	88.3%
F1-score	90.5%	89.8%	88.8%	88.1%	89.0%
			Device 4		
Accuracy	93.0%	91.9%	90.5%	90.1%	91.2%
Precision	92.2%	91.0%	89.6%	89.0%	90.1%

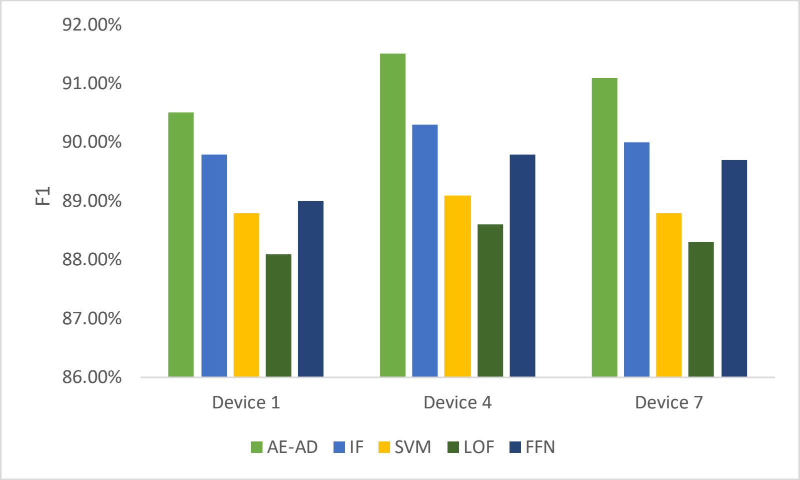
Recall	90.8%	89.6%	88.7%	88.2%	89.4%
F1-Score	91.5%	90.3%	89.1%	88.6%	89.8%
Device 7					
Accuracy	92.7%	91.5%	90.8%	89.5%	91.0%
Precision	91.8%	90.7%	89.2%	88.9%	90.3%
Recall	90.5%	89.3%	88.5%	87.8%	89.2%
F1-score	91.1%	90.0%	88.8%	88.3%	89.7%



(a)

(b)

(c)



(d)

Fig 3. (a) Accuracy, (b) Precision, (c) Recall and (d) F1-score

The analysis for three devices among nine from the N-BaIoT dataset highlights how each AD model performs across different metrics **Fig 3**. For Device 1, the proposed AE-AD model conspicuously distinguishes itself, achieving an accuracy of 92.4%. This superior performance can be attributed to the model's adeptness in effectively leveraging compact representations of the data and its dynamic thresholding method, which constantly adapts to the evolving nature of IoT data. The Isolation Forest follows closely with an accuracy of 91.7%, suggesting that ensemble-based methods also hold merit in this domain. However, SVM, FFN, and LOF yield lower accuracies of 90.2%, 90.9%, and 89.8%, respectively. With respect to precision, the AE-AD model continues to shine, notching a precision of 91.1% for Device 1.

This indicates the model's consistent ability to identify anomalies while minimizing false alarms correctly—a crucial aspect in real-world IoT applications where erroneous anomaly flags can be costly. While FFN and SVM also highlight commendable precision values around the 89-90% range, the intricate design of the AE-AD model places it a step ahead. The recall and F1-score metrics resonate with these findings. The performance hierarchy remains broadly similar for Device 4. Once again, AE-AD takes the lead, securing an accuracy of 93.0%. Its superiority is also evident in other performance measures, with precision, recall, and F1-score hovering above the 90% mark. FFN appears to be the second-best performer for this device, consistently outpacing SVM, LOF, and IF, albeit by a small margin.

For Device 7, the patterns persist. The AE-AD model reigns supreme with an accuracy of 92.7%. Its precision is 91.8%, achieving recall and F1-Score values of 90.5% and 91.1%, respectively. As with the other devices, the FFN comes second, showing its ability to perform consistently across various devices.

All models manifest robust results, but the AE-AD model remains at the forefront, underlining its balanced and efficient approach in AD. Its novel encoding-decoding mechanism, coupled with adaptive thresholding, makes it especially apt for pinpointing genuine anomalies in IoT device communications, yielding accurate and consistent results across varied data scenarios.

Training & Testing Loss Across Models

The provided **Fig 4** displays the average MSE over 50 epochs for various AD models. At the forefront is the AE-AD, the proposed model, recording a training loss of 0.03 and a testing loss of 0.05. These values highlight its efficacy in capturing intricate data patterns during training and generalizing effectively on unseen data. In stark contrast, models such as the Isolation Forest and LOF seem to lag, with the former registering MSE values of 0.09 and 0.11 for training and testing, respectively, and the latter posting similar figures of 0.10 and 0.12. Such elevated errors indicate potential challenges these models might face in grasping the nuances of the data across the epochs. The SVM model demonstrates a marginal improvement, its average MSE values being 0.08 and 0.10 for training and testing.

Nevertheless, it does not reach the proficiency of the AE-AD model. Meanwhile, the Feed Forward Neural Network (FFN) displays a middling performance, with its training and testing MSE at 0.06 and 0.08, respectively. Though it surpasses the Isolation Forest, SVM, and LOF in efficiency, it remains overshadowed by the exemplary performance of the AE-AD model. Conclusively, when considering the average MSE across 50 epochs, the AE-AD proposed model emerges as the most potent, signaling its robustness and reliability for AD compared to its counterparts.

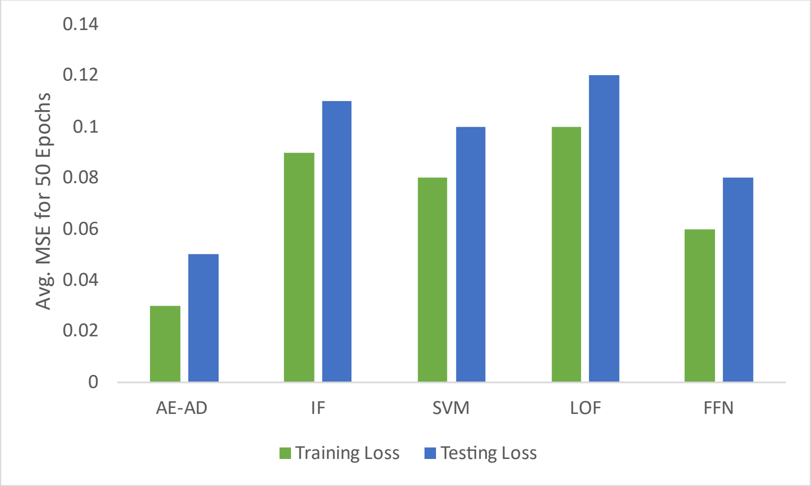


Fig 4. MSE for Training and Testing

VI. CONCLUSION AND FUTURE WORK

The IoT has rapidly grown, integrating various sectors like healthcare, agriculture, and home automation. With this integration, security has emerged as a significant concern due to the vastness and diversity of IoT devices. Many of these devices, designed for specific roles, may lack robust security features, rendering them vulnerable to cyberattacks. There is a pressing need for effective Anomaly Detection (AD), a method that identifies unusual data patterns, warning against potential threats. Research efforts have been dedicated to enhancing AD techniques suitable for the IoT landscape. ML and DL have been prominently featured in recent studies. Autoencoders have gained attention for their ability to process vast amounts of data and adapt to its evolving nature. Their application, when combined with dynamic thresholds, has shown promising results in identifying and addressing anomalies in real-time. In conclusion, as the IoT ecosystem grows, its security must evolve. Recent advances with autoencoders provide a durable foundation.

Future work, it would be beneficial to delve deeper into integrating autoencoders with other emerging technologies and algorithms. For instance, exploring the constructive collaboration between blockchain technology and autoencoders might offer enhanced data integrity and security in IoT networks.

Data Availability

Data sharing is not applicable to this article as no new data were created or analysed in this study.

Conflicts of Interests

The author(s) declare(s) that they have no conflicts of interest.

Funding

No funding agency is associated with this research.

Competing Interests

There are no competing interests.

References

[1]. S. Sengan, O. I. Khalaf, Vidya Sagar P., D. K. Sharma, Arokia Jesu Prabhu L., and A. A. Hamad, “Secured and Privacy-Based IDS for Healthcare Systems on E-Medical Data Using Machine Learning Approach,” International Journal of Reliable and Quality E-Healthcare, vol. 11, no. 3, pp. 1–11, Oct. 2021, doi: 10.4018/ijrqeh.289175.

[2]. P. Sharma, N. R. Moparthi, S. Namasudra, V. Shanmuganathan, and C. Hsu, “Blockchain-based <scp>IoT</scp> architecture to secure healthcare system using identity-based encryption,” Expert Systems, vol. 39, no. 10, Dec. 2021, doi: 10.1111/exsy.12915.

[3]. S. Rajasoundaran et al., “Secure routing with multi-watchdog construction using deep particle convolutional model for IoT based 5G wireless sensor networks,” Computer Communications, vol. 187, pp. 71–82, Apr. 2022, doi: 10.1016/j.comcom.2022.02.004.

[4]. S. Gali and V. Nidumolu, “An intelligent trust sensing scheme with metaheuristic based secure routing protocol for Internet of Things,” Cluster Computing, vol. 25, no. 3, pp. 1779–1789, Nov. 2021, doi: 10.1007/s10586-021-03473-3.

[5]. A. D. Jadhav and V. Pellakuri, “Highly accurate and efficient two phase-intrusion detection system (TP-IDS) using distributed processing of HADOOP and machine learning techniques,” Journal of Big Data, vol. 8, no. 1, Oct. 2021, doi: 10.1186/s40537-021-00521-y.

[6]. “Optimal Ensemble Learning Based on Distinctive Feature Selection by Univariate ANOVA-F Statistics for IDS,” International Journal of Electronics and Telecommunications, Jul. 2023, doi: 10.24425/ijet.2021.135975.

[7]. N. Satheesh et al., “Flow-based anomaly intrusion detection using machine learning model with software defined networking for OpenFlow network,” Microprocessors and Microsystems, vol. 79, p. 103285, Nov. 2020, doi: 10.1016/j.micpro.2020.103285.

[8]. G. Rekha, S. Malik, A. K. Tyagi, and M. M. Nair, “Intrusion Detection in Cyber Security: Role of Machine Learning and Data Mining in Cyber Security,” Advances in Science, Technology and Engineering Systems Journal, vol. 5, no. 3, pp. 72–81, 2020, doi: 10.25046/aj050310.

[9]. A. Madhuri, V. E. Jyothi, S. P. Praveen, S. Sindhura, V. S. Srinivas, and D. L. S. Kumar, “A New Multi-Level Semi-Supervised Learning Approach for Network Intrusion Detection System Based on the ‘GOA,’” Journal of Interconnection Networks, Jan. 2022, doi: 10.1142/s0219265921430477.

[10]. R. Ganeshan and P. Rodrigues, “Crow-AFL: Crow Based Adaptive Fractional Lion Optimization Approach for the Intrusion Detection,” Wireless Personal Communications, vol. 111, no. 4, pp. 2065–2089, Nov. 2019, doi: 10.1007/s11277-019-06972-0.

[11]. D. B. Dasari, G. Edamadaka, Ch. S. Chowdary, and M. Sobhana, “Anomaly-based network intrusion detection with ensemble classifiers and meta-heuristic scale (ECMHS) in traffic flow streams,” Journal of Ambient Intelligence and Humanized Computing, vol. 12, no. 10, pp. 9241–9268, Nov. 2020, doi: 10.1007/s12652-020-02628-1.

[12]. P. G. Om Prakash, B. Maram, G. Nalinipriya, and R. Cristin, “Harmony search Hawks optimization-based Deep reinforcement learning for intrusion detection in IoT using nonnegative matrix factorization,” International Journal of Wavelets, Multiresolution and Information Processing, vol. 19, no. 04, p. 2050093, Jan. 2021, doi: 10.1142/s0219691320500939.

[13]. R. Srilakshmi and J. Muthukuru, “Intrusion detection in mobile ad-hoc network using Hybrid Reactive Search and Bat algorithm,” International Journal of Intelligent Unmanned Systems, vol. 10, no. 1, pp. 65–85, Feb. 2021, doi: 10.1108/ijius-09-2020-0049.

[14]. M. M. V., B. RM, H. K Mewada, R. BR, and B. D, “Hybrid machine learning approach-based intrusion detection in cloud: A metaheuristic assisted model,” Multiagent and Grid Systems, vol. 18, no. 1, pp. 21–43, May 2022, doi: 10.3233/mgs-220360.

[15]. K. Singamaneni, K. N. Reddy, N. Yamsani, K. Sarada, and K. Saikumar, “Exploration of Convolutional Neural Network With Node - Centred Intrusion Detection Structure Plan for Green Cloud,” Journal of Green Engineering, vol. 10, no. 11, pp. 10781–10792, 2020.

[16]. R. Gangula, M. M. V, and R. K. M, “Network intrusion detection system for Internet of Things based on enhanced flower pollination algorithm and ensemble classifier,” Concurrency and Computation: Practice and Experience, vol. 34, no. 21, Jul. 2022, doi: 10.1002/cpe.7103.

[17]. M. K. Chandol and M. K. Rao, “Border Collie Cat Optimization for Intrusion Detection System in Healthcare IoT Network Using Deep Recurrent Neural Network,” The Computer Journal, vol. 65, no. 12, pp. 3181–3198, Nov. 2021, doi: 10.1093/comjnl/bxab136.

[18]. B. Samatha, T. Syamsundararao, and N. Karyemsetty, “Deep Learning Based Intrusion Prevention System in Vehicular Network,” Review of Computer Engineering Research, vol. 9, no. 3, pp. 169–180, Sep. 2022, doi: 10.18488/76.v9i3.3145.

[19]. N. Narisetty, G. R. Kancherla, B. Bobba, and K. Swathi, “Hybrid Intrusion Detection Method Based on Constraints Optimized SAE and Grid Search Based SVM-RBF on Cloud,” International Journal of Computer Networks and Applications, vol. 8, no. 6, p. 776, Dec. 2021, doi: 10.22247/ijcna/2021/210725.

[20]. A. D. Jadhav and V. Pellakuri, “Efficient Intrusion Detection Systems Using Machine Learning Approach for Sustainable IT Development,” Journal of Green Engineering, vol. 10, no. 10, pp. 8298–8310, 2020.

[21]. Y. P. Kumar and B. V. Babu, “Stabbing of Intrusion with Learning Framework Using Auto Encoder Based Intellectual Enhanced Linear Support Vector Machine for Feature Dimensionality Reduction,” Revue d’Intelligence Artificielle, vol. 36, no. 5, pp. 737–743, Dec. 2022, doi: 10.18280/ria.360511.

[22]. P. Chellammal, S. K. Malarchelvi, K. Reka, and G. Raja, “Fast and Effective Intrusion Detection Using Multi-Layered Deep Learning Networks,” International Journal of Web Services Research, vol. 19, no. 1, pp. 1–16, Nov. 2022, doi: 10.4018/ijwsr.310057.

[23]. K. Vamsi Krishna, K. Swathi, P. Rama Koteswara Rao, and B. Basaveswara Rao, ‘TSC: A Two-Stage Classifier for Network Intrusion Detection System on Green Cloud’, Journal of Green Engineering, vol. 11, no. 2, pp. 1500–1510, 2021.

[24]. M. B. Tamboli and Dr. N. R. Moparthi, “Deep Learning Model for Intrusion Identification,” Journal of Advanced Research in Dynamical and Control Systems, vol. 12, no. 5, pp. 388–395, May 2020, doi: 10.5373/jardcs/v12i5/20201726.

[25]. S. Kumar, A. Jain, S. Rani, H. Alshazly, S. Ahmed Idris, and S. Bourouis, “Deep Neural Network Based Vehicle Detection and Classification of Aerial Images,” Intelligent Automation &amp; Soft Computing, vol. 34, no. 1, pp. 119–131, 2022, doi: 10.32604/iasc.2022.024812.