Arabian Journal for Science and Engineering (2024) 49:12561–12577
https://doi.org/10.1007/s13369-024-08742-y

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

Robust Botnet Detection Approach for Known and Unknown Attacks in IoT Networks Using Stacked Multi-classifier and Adaptive Thresholding

Deepa Krishnan[1] · **Pravin Shrinath**[1]

Received: 16 September 2023 / Accepted: 8 January 2024 / Published online: 14 February 2024

Abstract
The detection of security attacks holds significant importance in IoT networks, primarily due to the escalating number of interconnected devices and the sensitive nature of the transmitted data. This paper introduces a novel methodology designed to identify both known and unknown attacks within IoT networks. For the identification of known attacks, our proposed approach employs a stacked multi-classifier trained with classwise features. To address the challenge of highly imbalanced classes without resorting to resampling, we utilize the Localized Generalized Matrix Learning Vector Quantization (LGM-LVQ) approach to select the most relevant features for each class. The efficacy of this model is evaluated using the widely recognized NF-BoT-IoT dataset, demonstrating an impressive accuracy score of 99.9952%.. The proposed study also focuses on detecting unseen attacks leveraging a shallow autoencoder, employing the technique of reconstruction error thresholding. The efficiency of this approach is evaluated using benchmark datasets. namely NF-ToN-IoT and NF-CSE-CIC-IDS 2018. The model's performance on previously unseen samples is noteworthy, with an average accuracy, precision, recall and F1-Score of 93.715%, 99.955%,90.865% and 95.145%, respectively. The proposed work presents significant contributions to IoT security by proposing a comprehensive solution with demonstrated performance in detecting both known and unknown attacks in the context of imbalanced data.

Keywords Autoencoder · Botnet · Detection · Imbalanced · Unseen attacks

1 Introduction

The term"Internet of Things"(IoT) refers to a network of physical objects that are equipped with sensors and technology enabling them to connect and interact with other devices or objects over the Internet. The Internet of Things (IoT) has become widely popular due to the growing utilization of cost-effective sensors, Cloud platforms, Big Data, and artificial intelligence (AI). Numerous household and industrial applications are built to capitalize on the tremendous business value they can offer. It is increasingly used in various applications like Smart Grids, Smart Health, Smart Cities, Logistics, and various consumer and industrial applications [1–3]. IoT's growing popularity and scope also lead us to recognize its security and privacy concerns. Many IoT installations are mission-critical and offer much strategic and business value; hence, continued undisrupted operation is critical [4].

In [5], the authors detailed the different categories of security attacks possible in IoT's edge, network, and application layers. Despite the similarity of IoT attacks with wired networks, the security solutions designed for IoT need to consider the resource-constrained nature of IoT. IoT device manufacturers often prioritize the functionality of devices over the security and privacy aspects. The inherent resource constraints like low memory, computation ability, and processing power demand greater care when designing security solutions for IoT. The considerations for the security framework for IoT infrastructure are illustrated in [6–8] and form a baseline for future IoT security designs.

Machine learning and deep learning algorithms have been used in recent years to detect security attacks in IoT [9–12]. A comprehensive review of machine learning-based IoT security solutions is done in [13, 14]. The authors have investigated the significant security challenges and attacks of IoT

✉ Deepa Krishnan
deepa.krishnan@nmims.edu

Pravin Shrinath
pravin.shrinath@nmims.edu

[1] Department of Computer Engineering, Mukesh Patel School of Technology, Management and Engineering, SVKM's Narsee Monjee Institute of Management Studies (NMIMS) Deemed-to-University, Mumbai, Maharashtra 400056, India

devices and popular machine learning and deep learning techniques researchers use. Many researchers undertaking machine learning-based models focus on effective feature selection before building the model. Most feature selection techniques fall under three categories: filter-based, wrapper-based, and embedded techniques. The embedded feature selection approach allows classifiers to remove attributes dynamically for simultaneous feature selection and classification [15, 16]. However, the current methodologies for feature selection take into account all of the target classes, making them insufficient for determining the most relevant features for distinguishing each class. The classwise feature selection techniques can provide significant benefits in supervised machine learning by reducing the overhead in training.

Most of the research works in detecting security attacks focus on supervised learning techniques, and they are proven successful in detecting known security attacks. The threat detection techniques focusing on signatures have proven ineffective for zero-day attacks. Although considerable work has been done in this area, detecting unknown security attacks is still challenging. An exhaustive review of important works related to zero-day attack detection is done in [17]. The most prominent approaches identified by authors are outlier-based, autoencoders, generative adversarial networks (GANs) and transfer learning-based techniques. In [18], the identification of zero-day threats is accomplished by the utilization of a transfer learning strategy that has its basis in deep transcoding and cluster correspondence. Transfer learning's applicability in detecting zero-day attacks is motivated by the assumption that zero-day attacks are versions of existing assaults and, hence, share the same characteristics as known attacks [19, 20]. Another approach for detecting zero-day malware is done in [21], where the authors have used a novel method termed transferred deep convolutional tDCGAN to generate fake malware images. Further, an autoencoder-based technique distinguishes fake malware images from real, hence detecting zero-day malware. The literature review conducted in this study highlights the potential of various approaches in identifying zero-day attacks. However, it is important to note that the detection quality of the proposed techniques does not exhibit promising results when applied to different types of attacks.

In response to IoT's security challenges, researchers are increasingly employing machine learning and deep learning techniques for attack detection. While supervised learning excels in known attack detection, the challenge of identifying zero-day attacks persists. The findings of our literature review suggest a growing trend in recent publications toward utilizing edge nodes to deploy security solutions for IoT nodes. The deployment of IoT security solutions on edge nodes is a promising strategy due to the computational limitations of IoT nodes. The proposed research aims to develop a comprehensive solution for IoT security, focusing on effective detection and classification of both known and unknown attacks on IoT nodes. The primary objective is to achieve a high accuracy rate, particularly when dealing with highly imbalanced datasets. Additionally, the solution seeks to be adaptable to evolving threat landscapes, emphasizing the identification of zero-day attacks. The effectiveness of the approach would involve demonstrating the capability to effectively identify and respond to new and previously unseen attack patterns. Additionally, the study seeks to minimize false positives and false negatives in attack detection and classification, aiming for a balanced approach between precision and recall. This involves reducing the chances of both false positives and false negatives. This research contributes significantly to the existing literature by addressing the dual challenges of known and unknown attack detection in IoT environments.

1.1 Major Contributions

The main contributions of this research are summarized as follows:

- A modified matrix learning vector quantization approach for selecting the most relevant features for detecting each type of security attack in the context of a highly imbalanced dataset. The proposed algorithm will derive the class-specific discriminative features for identifying the class and helps in detecting imbalanced classes with competitive performance scores.
- A stacked multi-classifier based on MLP trained on the selected relevant features capable of detecting known attacks with competitive performance scores. The proposed MLP model design ensures that the model has few parameters thus ensuring lesser size of model.
- Two shallow autoencoders have been trained on benign and attack samples, respectively, that can detect zero-days attacks based on adaptive re-construction error thresholding technique. The design of the shallow autoencoders with input layer, latent layer and output layer ensured minimum parameters and a compact model size of only 24KB.

2 Related Work

2.1 Review of Detection of Known Attacks in IoT

Over the years, a multitude of machine learning (ML) and deep learning (DL) security solutions have been proposed for detecting threats within Internet-of-Things (IoT) networks. This section provides a comprehensive review of frequently used ML and DL algorithms, the approaches to feature selection, and the notable challenges inherent in these studies. In

[22] introduced a Bijective soft function for picking relevant features and a novel measure, CorrAUC, for selecting the best features using the benchmark datasets BoT-IoT and UNSW-NB15. With the exception of the Normal and Data Theft categories, the findings are encouraging. However, the study utilized a wrapper-based feature selection technique that is computationally expensive by nature. In [23], authors proposed an optimized approach of the Whale Optimization Algorithm (WOA), a significant feature selection approach in the area of IoT attack detection that can manage the large dimensionality of the dataset. The researchers examined the V-shaped and S-shaped transfer functions in WOA and compared the optimal transfer function to other well-known evolutionary optimizers. Experiments are conducted using the well-known N-BaIoT dataset, and statistical evaluation is performed using the Wilcoxon test to determine the optimal classifier. The computational complexity of the proposed algorithm is not, however, evaluated.

In [16], the proposed random forest classifier used a wrapper-based feature selection based on Tabu search. Even though the model's performance ratings are adequate, the computational complexity of the feature selection method must be assessed. In a comparable study [24] focusing on the detection of botnet attacks within IoT, the authors employ a feature selection algorithm based on principal components using the BoT-IoT dataset. The Lock Edge model demonstrated a 0.90862 detection rate for DoS HTTP attacks but exhibited a lower rate of 0.56098 for Data Theft. Notably, the model's complexity is reduced compared to traditional neural networks. In another significant investigation utilizing the BoT-IoT dataset, particle swarm optimization and voting techniques are employed for feature identification [25]. The voting system is applied to classify botnet samples, utilizing a combination of deep neural network methods, support vector machines (SVM), and the decision tree C4.5. The primary contribution of this study lies in the combination of the particle swarm optimization (PSO) feature selection algorithm with a voting mechanism, leveraging deep learning techniques for effective botnet detection

2.2 Review of Detection of Unseen Attacks in IoT

Another key focus of the literature work is the review of methodologies for identifying previously unseen attacks. In the study by [26], a novel approach to detect unseen attacks is discussed. The authors assessed the effectiveness of trained models in categorizing new samples belonging to the same attack category but from a different dataset. The model, initially trained on the CICIDS 2017 dataset, was evaluated for its ability to detect new attack classes using the CICIDS 2018 dataset. Although it was anticipated that integrating trained models with closely related novel data would maintain exceptional classification scores, the evaluation on the unseen dataset revealed a decrease in accuracy, precision, and recall. Notably, the model's performance remained consistent only when evaluated on the dataset used for training.

Another approach for detecting unseen attacks involves the use of ensemble deep learning architectures, as discussed in [27]. In this study, if the model predicts an attack test case as belonging to one of its attack classes, the classification is considered correct, indicating that the model correctly identified it as an attack but was unsure about its specific class. On the other hand, misclassifying an attack test instance as benign is considered a classification error. The study achieved accurate classification when the model correctly identified a DDoS test case from the BoT-IoT dataset as DoS, Reconnaissance, or Theft. However, some of the proposed work exhibited detection accuracy close to zero for certain classes.

Similar work is done to detect unseen attacks using Correntropy with a Gaussian-mixture-based One-class ensemble model [28]. The authors focus on deploying the model developed on edge nodes however, the experimentation is performed on NSL-KDD and UNSW-NB15 datasets. Transfer Learning based approach for detect the zero-day attacks in IoT is utilized in [29]. The approach involves implementing the solution utilizing the BoT-IoT dataset as the source domain for knowledge acquisition and subsequently applying this knowledge to the UNSW-NB15 dataset as the target domain. Another significant work using a novel two-stage methodology that leverages deep neural networks and transfer learning is done in [30]. This intelligent and adaptable approach is designed for precise detection of zero-day malware by analyzing hardware events through image. However, one of the major drawback of transfer learning approach is that it relies on pre-trained models from a different dataset to enhance performance on a target domain. This results in model inadvertently capturing characteristics specific to the source domain that do not generalize well on unseen data. In [31], the authors have done a systematic literature review on diverse methodologies, techniques, and machine learning and deep learning algorithms used for the detection of zero-day attacks. Through an extensive review of recent publications, this study identifies emerging research trends and challenges in the field. Despite technological advancements, the abundance of large datasets, and the robust processing capabilities of machine learning and deep learning algorithms, the identification of entirely new or unknown attacks remains an ongoing research frontier.

2.3 Motivation and Problem Definition

The existing landscape of IoT security attack detection methods reveals notable gaps in addressing effective feature selection and detecting unseen attacks. While current research predominantly focuses on feature selection that considers the relationship between all features in predicting all

classes, there is a significant oversight in class-specific feature extraction, particularly in analyzing the statistical and predictive power of characteristics specific to each class. This limitation, compounded by the disregard for the distribution of each class, underscores the need for a feature selection technique capable of identifying discriminative features for each target class.

Additionally, the emergence of unseen attacks poses a significant challenge for ML-based security systems, as these attacks, whether entirely new or redesigned versions of older threats, aim to challenge established strategies. Several studies that analyze the capabilities of machine learning classifiers to identify zero-day attacks reveal a noticeable decline in performance when evaluated against new threats [14, 32]. Recent efforts include unsupervised learning [33, 34], autoencoder-based [35], and transfer learning [19] to detect zero-day attacks. However, quantitative comparisons between these schemes are challenging due to differences in evaluation datasets, metrics, and comparison schemes. A transfer-learning-based method may be ineffectual when there is a dissimilarity between unknown and known threats. Additionally, unsupervised methods might produce more false positives, and autoencoder-based methods must account for the model's high complexity and parameters.

Addressing these research gaps, our proposed work introduces a feature selection technique that identifies discriminative features for each target class which are further used to construct a stacked MLP classifier. Additionally, our work introduces a reconstruction error thresholding technique for detecting unseen attacks, leveraging two shallow autoencoders to reduce model size and complexity. This research aims to fill critical gaps in feature selection and unseen attack detection, offering a more nuanced and effective approach for attack detection in IoT networks.

3 Datasets and Evaluation Criteria Used

The proposed work has used the extended version of NF-BoT-IoT, NF-ToN-IoT and NF-CSE-CIC-IDS2018 datasets, which are developed by the authors to create benchmark datasets with common set of features [36]. This dataset provides a common ground feature set from multiple datasets and facilitates model's detection ability and generalizability across multiple datasets. These datasets are generated from the following existing benchmark NIDS datasets: BoT-IoT [37], ToN-IoT [38], and CSE-CIC-IDS2018 [39].

3.1 NF-BoT-IoT V2

The dataset utilized in this study is the NF-BoT-IoT V2 dataset, which is derived from the BoT-IoT dataset and is based on IoT NetFlow data. The features were derived from the pcap data that were openly accessible, and subsequently, the flows were assigned labels corresponding to their various attack categories. The original dataset consists of a total of 37,763,497 data flows comprising of four distinct attack categories including DDoS, DoS, Reconnaissance and Theft. We performed a random undersampling, reducing to 25% of the entire dataset while retaining the original imbalance ratio. Further we considered a subset of 10,48,248 data samples encompassing five categories which were utilized for class-wise feature selection and subsequent model building. Table 1 shows the distribution of NF-BoT-IoT flows utilized in our experimental analysis for the purpose of detecting known attacks.

Table 1 Number of samples of NF-BoT-IoT dataset

Class	Training	Testing	Validation
Benign	264532	66133	36740
DDoS	488922	122231	67906
DoS	343	86	48
Reconnaissance	884	221	123
Theft	57	14	8

3.2 NF-ToN-IoT V2

The NF-ToN-IoT V2 dataset, derived from the benchmark network dataset ToN-IoT, is utilized for conducting experimental investigations on the detection of previously unobserved attacks. The features were derived from the pcap files that were accessible to the public, and afterward, the flows were assigned labels corresponding to their various attack types.The original dataset comprises a total of 16,940,496 data flows, with 10,841,027 (63.99%) classified as instances of attacks and 6,099,469 (36.01%) categorized as benign samples. The dataset consists of nine distinct attack categories, namely Backdoor, Denial of Service (DoS), Distributed Denial of Service (DDoS), Injection, Man-in-the-Middle (MITM), Password, Reconnaissance, Scanning, and Cross-Site Scripting (XSS). We performed a random undersampling, reducing to 6.25% of the entire dataset while retaining the original imbalance ratio. In the experimental study conducted for the proposed work on the detection of unseen attacks, the Benign samples were utilized to create the Benign class, while samples from all nine attack classes were aggregated to form the attack class. Table 2 illustrates the distribution of NF-ToN-IoT samples included in the experimental analysis to evaluate the proposed approach.

Table 2 Number of samples of NF-ToN-IoT dataset

Class	Testing samples
Benign	330665
Attack	612758

Table 3 Number of samples of NF-CSE-CIC-2018 dataset

Class	Testing samples
Benign	20000
Attack	20000

3.3 NF-CSE-CIC-2018 V2

The dataset used in this study is NF-CSE-CIC-IDS2018-V2, which is derived from the CSE-CIC-IDS2018 dataset and focuses on NetFlow-based data in the context of the Internet of Things (IoT). The features were derived from the pcap files that are accessible to the public, and the flows were further categorized and labeled according to their corresponding attack types.The original dataset consists of a total of 18,893,708 flows, with 2,258,141 (11.95%) identified as instances of attacks and 16,635,567 (88.05%) characterized as benign. The dataset comprises a total of 14 attack categories, which include DDoS Attacks HOIC, DoS Attacks Hulk, DDoS Attacks LOIC HTTP, BoT, Infiltration, SSH Bruteforce, DoS Attacks Golden Eye, FTP BruteForce, DoS Attacks Slow HTTP Test, DoS Attacks Slowloris, Brute Force Web, DDoS Attack LOIC UDP, Bruteforce XSS, and SQL Injection. We also performed a random undersampling, reducing to 6.25% of the entire dataset while retaining the original imbalance ratio. In the course of our experimental investigation, we considered only 20,000 samples each from the benign and attack categories. It is essential to highlight that the 14 distinct attack categories were combined to create a unified attack class for analysis. The distribution of all flows utilized in the proposed method for detecting unseen attacks is presented in Table 3, which corresponds to the NF-CSE-CIC-IDS 2018 V2 dataset.

3.4 Evaluation Metrics

The performance metrics used to evaluate the techniques described in this paper are defined in this subsection.

- **Accuracy:** Accuracy quantifies the ratio of correctly predicted instances out of all total instances. The calculation involves dividing the number of correct predictions by the total number of predictions.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TN} + \text{TP} + \text{FN} + \text{FP}} \tag{1}$$

- **Precision:** Precision is the ratio of true-positive predictions out of all positive predictions. The calculation of precision involves dividing the number of true positives by the sum of true positives and false positives.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

- **Recall:** The recall, which is also referred to as sensitivity or true-positive rate, is a performance evaluation measure that represents the proportion of true positives (TPs) correctly identified by the model out of all actual positive instances.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

- **F1-Score:** It measures the harmonic mean of precision and recall score. This is useful when both precision and recall hold equal significance, and there is a need to balance between these metrics for assessing the model's overall efficacy.

$$\text{F1 Score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

4 Proposed Methodology

This section outlines the proposed solution developed in consideration of the research gaps presented in Sect. 2. The proposed technique is illustrated in Fig. 1, which provides a schematic overview of the system architecture.

The analysis and detection engine that could be installed in edge nodes consists of two functional components: the Known Attack Detection Module and the Unseen Attack Detection Module. The methodology of detecting the known attacks and unseen attacks is illustrated in the flow diagram in Fig. 2. The functionality of each of the modules is described in the following subsections.

4.1 Detection of Known Attacks

This section describes the steps of detection of known attacks with the proposed classwise feature selection method and the design of the stacked multi-classifier. The functional modules are described as follows:

4.1.1 Data Pre-processing and Feature Engineering

The NF-BoT-IoT dataset used in our proposed study has an initial 43 features. During the pre-processing, features deemed redundant or less important are excluded from further study. The features IPV4_SRC_ADDR, IPV4_DST_

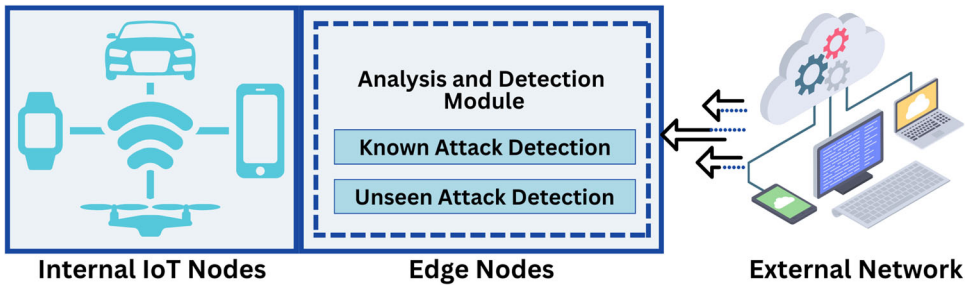


Fig. 1 Schematic overview of proposed method with the analysis and detection module

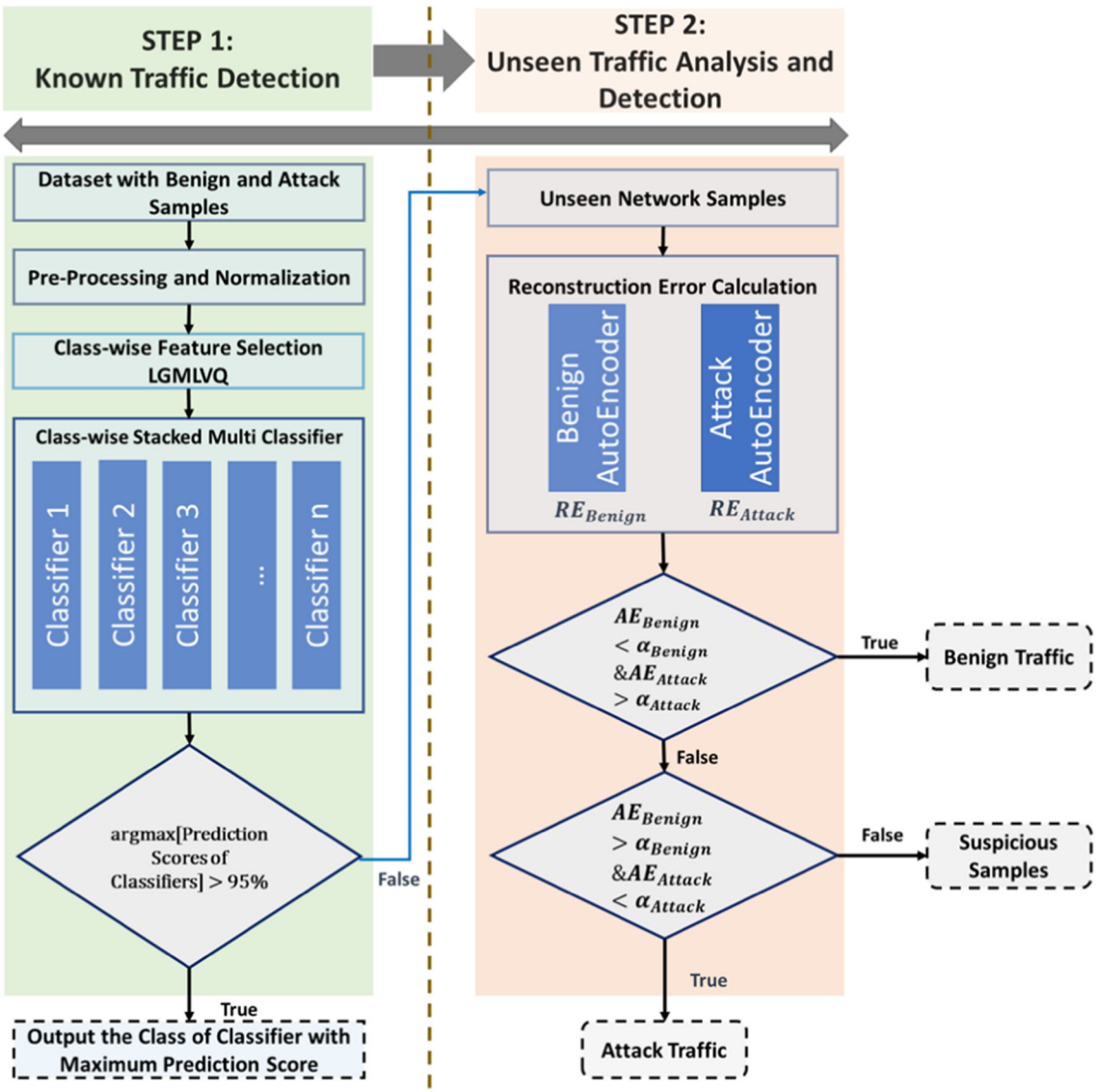


Fig. 2 Workflow of proposed method with the analysis and detection module

ADDR, L4_SRC_PORT, L4_DST_PORT are removed from the feature set as they do not contribute to the model's generalizability. The category feature that contributes to various kinds of attacks is label encoded. The same features are dropped from the NF-ToN-IoT dataset and NF-CSE-CICDS-2018 dataset also. The dataset is further examined for any instances of null values, which are then removed. We have also undertaken feature scaling as most of the features in the dataset show huge difference in their scale. Most of the machine learning algorithms are sensitive to this variation in scale, and we have used Standard Scaler. Standard scaler scales the data with zero mean and unit variance. Standard-

izing each feature vector involves subtracting the mean of the feature vector from each feature value and then dividing by the standard deviation of the feature vector. The resulting values will have a mean of zero and a standard deviation of one. The standard scaler scales the feature values as shown in Eq. 5

$$X_{\text{std}} = \frac{X - \mu}{\sigma} \tag{5}$$

where X is the initial value of the feature, μ is the mean value of the feature column, and σ is the standard deviation of the feature column.

4.1.2 Proposed Localized Generalized Matrix Learning Vector Quantization (LGLMVQ) Approach for Feature Selection

In the proposed work, the feature selection process is undertaken to identify the most relevant feature that can discriminate each class. The proposed feature selection strategy is based on learning vector quantization-based method [40] for identifying prototypes representing the classes in the data. Each sample is classified based on the distance from all prototypes and allocating it to the closest prototype's class. The NF-BoT-IoT dataset used for the experimental study can be considered as $D = \{(x_i, y_i) \mid x_i \in \mathbb{R}^N, y_i \in \{1, \ldots, C\}\}_{i=1}^{P}$ where P represents the number of samples of x_i with labels y_i of C classes. An LVQ model on D consists of various prototypes identified by their location on the weight space $w_i \in \mathbb{R}^N$ and their class label $c(w_i) \in \{1, \ldots, C\}$. A data point x_i is classified using similarity measure d^{λ}, where λ is the metric parameter that can be adapted during training. X is mapped to a label $c(x_i) = c(w_i)$ of the prototype i for which $d^{\lambda}(w_i, x_i) \leq d^{\lambda}(w_j, x_i)$ where $j \neq i$, thereby mapping it to the class of the closest prototype.

The dataset used in the experimental study is found to have nonlinear class boundaries and a modified version of LVQ known as Localized General Matrix Learning Vector Quantization[LGMLVQ] which can give better class separation is used [41, 42]. The LGMLVQ algorithm updates the prototype vectors using the LGMLVQ update rule, which combines the standard LVQ update rule with a neighborhood function that depends on the distances between the training data and the prototype vectors. The proposed feature selection technique using LGMLVQ is described in Algorithm 1 and is used to generate the relevance matrix for each class of the dataset. It focuses on selecting relevant features for each class by considering their contributions to class separation using Fisher's discriminant ratio.

The within-class covariance matrix measures the variability of feature values within a particular class. It's calculated by considering the differences between the feature values of samples within the same class. The within-class covariance matrix S_c for a class c is as follows:

$$S_c = \frac{1}{n_c - 1} \sum_{i=1}^{n_{\text{samples}}} (x_i - \mu_c)(x_i - \mu_c)^T$$

where

n_c is the number of samples in class c.
x_i represents the feature vector of the i-th sample in class c.
μ_c is the mean feature vector of all samples in class c.

The between-class covariance matrix quantifies the variability between different classes. It captures how the feature values of samples from different classes are spread out in relation to each other. The between-class covariance matrix $S_{\neg c}$ for class c is calculated as follows:

$$S_{\neg c} = \sum_{i=1}^{C} n_i (\mu_i - \mu)(\mu_i - \mu)^T$$

where

C is the total number of classes.
n_i is the number of samples in class i.
μ_i is the mean feature vector of all samples in class i.
μ is the overall mean feature vector calculated across all samples and classes.

The trace of the within-class covariance matrix S_c multiplied by the inverse of the sum of S_c and $S_{\neg c}$is calculated as J_c. Similarly, the trace of the between-class covariance matrix $S_{\neg c}$ multiplied by the inverse of the sum of S_c and $S_{\neg c}$, is calculated in $J_{\neg c}$. The Fisher's discriminant ratio is calculated as $\frac{J_c}{J_{\neg c}}$. is a statistical measure used to evaluate the separation between two or more classes in a feature space. It quantifies how well the classes are separated while considering the within-class variability and the between-class variability of the feature values. The most discriminative features for each dataset class are chosen from the relevance matrix created with Fischer's ratio. The selection is undertaken using the threshold for relevance parameter θ. The feature subset $\{F_i\}$ for each class C is created by choosing features from the relevance matrix created with Fisher's score $\theta > 0.5$. Fisher's discriminant ratio is a statistical measure used to evaluate the separation between classes in a feature space, and the relevance matrix is constructed based on the

Fisher's scores. The threshold serves as a criterion for selecting the most discriminative features. A Fisher's score greater than 0.5 indicates that the feature has a higher contribution to the between-class variability compared to the within-class variability, ensuring that only features with a strong contribution to class separation are selected. The threshold of 0.5 strikes a balance between being too permissive (including less discriminative features) and too restrictive (excluding potentially useful features). It helps in creating a feature subset that captures important discriminative information by filtering out less informative or less discriminatory features, reducing the chances of overfitting. The features selected from the NF-BoT-IOT dataset for Benign, DDoS, DoS, Reconnaissance, and Theft classes are listed in Table 4.

Algorithm 1 Localized General Matrix Learning Vector Quantization (LGMLVQ)

Require:
- Training Dataset X of size $(n_{\text{samples}}, n_{\text{features}})$
- Label Vector Y of size (n_{samples})
- Threshold for feature relevance θ

Ensure:
- Dictionary of Feature Vectors for each class F_{rC}

for each class c in unique(Y) **do**
  Compute the within-class covariance matrix S_c and between-class covariance matrix $S_{\neg c}$
  $J_c \leftarrow \text{tr}(S_c \cdot \text{inv}(S_c + S_{\neg c}))$
  $J_{\neg c} \leftarrow \text{tr}(S_{\neg c} \cdot \text{inv}(S_c + S_{\neg c}))$
  Compute the Fischer discriminant ratio $J_c / J_{\neg c}$
  $F_{rC}[c] \leftarrow \{\}$ ▷ Initialize an empty list to store relevant features for class c
  for each feature i from 1 to n_{features} **do**
    if $J_c/(J_c + J_{\neg c}) > \theta$ **then**
      Add feature i to $F_{rC}[c]$
Return F_{rC}

4.1.3 Design of Stacked Multi-classifier and Detection of Known Attacks

The pre-processed dataset is split into D' and D'' according to the 90:10 rule. D' is further split into five sub-datasets D'_{Benign}, D'_{DDoS}, D'_{DoS}, $D'_{\text{Reconnaissance}}$, and D'_{Theft} based on the five classes. The relevant features listed in Table 4 are selected from each of the five sub-datasets using the LGLMVQ algorithm. The classwise sub-datasets with only relevant features are split into training and testing sets according to the 80:20 rule. Algorithm 2 describes the proposed model building and classification on the dataset. The training and testing classwise sub-datasets are used to build the proposed multi-layer perceptron classifier MLP). $X^{\text{train}}_{\text{Benign}}$ and $Y^{\text{train}}_{\text{Benign}}$ are used to build the classifier $\text{MLP}_{\text{Benign}}$. The testing performance of the model is further evaluated using $X^{\text{test}}_{\text{Benign}}$ and $Y^{\text{test}}_{\text{Benign}}$. Similarly, MLP_{DDoS}, MLP_{DoS}, $\text{MLP}_{\text{Reconnaissance}}$, and $\text{MLP}_{\text{Theft}}$ classifiers are built and evaluated on the respective sub-datasets created for DDoS, DoS, Reconnaissance, and Theft classes. Furthermore, the D'' dataset, which has samples from all the classes, is used to evaluate the robustness in classifying the known attack classes. The batch samples consisting of all classes are fed to each individual MLP classifiers. The relevant features of Benign class are only selected while testing with the benign MLP classifier and the prediction score from each MLP_i for that batch is appended to prob$'$. Similarly the batch of network samples is evaluated using each of the stacked MLP classifier. The final prediction of the batch samples is obtained by choosing the prediction of the MLP classifier with the maximum prediction score.

The proposed study considers the attack classes as positive and benign classes as negative classes. The assessment of the classifier's efficacy is carried out to reduce the occurrence of false positives, which ideally should approach zero. Simultaneously, our objective is to lower the occurrence of false positives also, hence achieving a balance between high precision and recall is the optimal approach. The classification threshold has been determined by analyzing the RoC and precision–recall curve. The optimal threshold refers to the classification threshold that yields the upper-left corner of the curve, thereby minimizing the gap between the true-positive rate (TPR) and false-positive rate (FPR). The threshold value that yields the true-positive rate (TPR) at the maximum acceptable false-positive rate (FPR) and the highest precision–recall values is determined within threshold regions exceeding 0.95. When none of the stacked classifiers yielded a classification score exceeding 95%, the network samples will be evaluated as unknown samples.

4.2 Proposed Scheme for Detection of Unknown Attacks

This section outlines the proposed two-level shallow autoencoder's design and its mechanism for detecting unidentified attacks.

4.2.1 Design of Proposed Two-Level Shallow Autoencoder

The proposed design for the detection of unknown attacks involves two shallow autoencoders. The shallow autoencoder is a lightweight design comprising only one hidden layer or latent layer between the input and output layers. In the proposed approach, the benign samples of the BoT-IoT dataset is used to train the benign autoencoder represented as $\text{AE}_{\text{Benign}}$ and the attack samples from all the four classes DDoS, DoS, Reconnaissance, and Theft are combined to train the attack autoencoder $\text{AE}_{\text{Attack}}$. The $\text{AE}_{\text{Benign}}$ and $\text{AE}_{\text{Attack}}$ autoencoders are comprised of an encoder and a decoder. The encoder utilizes a deterministic nonlinear mapping function

Table 4 Features selected for different classes for NF-BoT-IoT dataset

Class	Features selected
Benign	PROTOCOL,CLIENT_TCP_FLAG,FLOW_DURATION_MILLISECONDS, DURATION_IN, MIN_TTL,MAX_TTL,SHORTEST_FLOW_PKT, RETRANSMITTED_OUT_PKTS, DNS_QUERY_ID, DNS_QUERY_TYPE
DDoS	PROTOCOL,L7_PROTO,FLOW_DURATION_MILLISECOND,DURATION_IN, DURATION_OUT,MIN_TTL,SHORTEST_FLOW_PKT,RETRANSMITTED_OUT_PKTS,SRC_TO_DST_AVG_THROUGHPUT
DoS	PROTOCOL,L7_PROTO,OUT_BYTES,OUT_PKTS,TCP_FLAGS, NUM_PKTS_512_TO_1024_BYTES,NUM_PKTS_1024_TO_512_BYTES, TCP_WIN_MAX_IN,FTP_COMMAND_RET_CODE
Reconnaissance	PROTOCOL,L7_PROTO,CLIENT_TCP_FLAGS,FLOW_DURATION _MILLISECOND,DURATION_IN,MIN_TTL,MAX_TTL,MIN_IP_PKT_LEN, RETRANSMITTED_IN_BYTES,RETRANSMITTED_OUT_BYTES, SRC_TO_DST_AVG_THROUGHPUT,TCP_WIN_MAX_IN,DNS_QUERY_ID, DNS_QUERY_TYPE,DNS_TTL_ANSWER
Theft	L7_PROTO,CLIENT_TCP_FLAGS,SERVER_TCP_FLAGS, FLOW_DURATION_MILLISECOND,DURATION_IN,MIN_TTL,MAX_TTL, SHORTEST_FLOW_PKT,RETRANSMITTED_OUT_BYTES, NUM_PKTS_UP_TO_128_BYTES,NUM_PKTS_128_TO_256_BYTES, TCP_WIN_MAX_IN,TCP_WIN_MAX_OUT

Algorithm 2 Proposed Stacked Multi-Classifier using MLP Building and Classification

Require:
- Classwise split datasets D'_{Benign}, D'_{DDoS}, D'_{DoS}, $D'_{\text{Reconnaissance}}$, and D'_{Theft}
- 10 percent of pre-processed dataset with instances of all classes D''

Ensure:
- Predicted Class Labels for each observation

Initialize $prob = []$
▷ To hold predicted probabilities of each class
for $i = 1$ to n **do**
▷ $n \in$ [Benign, DDoS, DoS, Reconnaissance, Theft]
  $X_i^{\text{train}}, X_i^{\text{test}}, Y_i^{\text{train}}, Y_i^{\text{test}} \leftarrow$ train_test_split(D'_i)
  ▷ Split classwise data into training and testing sets:
  $MLP \leftarrow$ MLPClassifier()
  ▷ Build an MLP Classifier on training data:
  $MLP.fit(X_i^{\text{train}}, Y_i^{\text{train}})$
  $prob.append(MLP.predict\_prob(X_i^{\text{test}}))$
  ▷ Predict probabilities on test data and append to vector *prob*:
$Predicted\_Label = \text{argmax}\left(\frac{1}{n}\sum_{i=1}^{n} \text{prob}\right)$
▷ Calculating Final Prediction Probability
Validating on dataset D''
Initialize $prob' = []$
▷ To hold predicted probabilities from each MLP model
Split D'' into X_{val} and Y_{val}
for $i = 1$ to n **do** ▷
$n \in$ [Benign, DDoS, DoS, Reconnaissance, Theft]
  $X_{\text{val}} \leftarrow X_{\text{val}}[f_1, f_2, f_3, \ldots, f_n]$
  ▷ With relevant features selected for MLP_i
  $prob'$.append(MLP_i.predict_proba(X_{val}))
$Predicted\_Label = \text{argmax}prob'$
▷ Final prediction for validation sample

to map input x to represent the hidden layer, H. The encoding procedure can be written as $H = F(Wx + b)$ where W is the weight between the input and the hidden layer representation H, and b is the bias. The decoder reconstructs the hidden representation of the hidden layer to output $\hat{x}$ and $\hat{x}$ is represented as $F'(W'H + b')$ where W' is the weight between the hidden layer representation and the output, b' is the deviation, and x' is the reconstruction output of Y. The shallow autoencoders $\text{AE}_{\text{Benign}}$ and $\text{AE}_{\text{Attack}}$ are trained by minimizing the reconstruction error. Consider the training data $D = \{x^i_{\text{Benign}}\}_{i=1}^{N}$ and $\{x^i_{\text{Attack}}\}_{i=1}^{N}$, the loss function of AE is $L(x, \hat{x}) = \frac{1}{N}\sum_{x \in D} \|x - \hat{x}\|^2$. The $\text{AE}_{\text{Benign}}$ trained with normal traffic data is capable of reconstructing any input as closely as feasible to the learned normal patterns. Similarly, the $\text{AE}_{\text{Attack}}$ trained with attack traffic will reconstruct any input as closely as possible to the learned attack patterns of the BoT-IoT dataset.

The hyper-parameters of the benign and attack autoencoder are outlined in Table 5. The selection of 39 input neurons aligns with the dimensionality of the input data features. Notably, the design opts for a straightforward architecture devoid of hidden layers, aiming to avoid introducing additional transformations to the input data prior to reaching the latent layers. This choice minimizes the number of parameters, thereby reducing overall model complexity. The number of neurons in the latent layer is chosen as 10 to create a low-dimensional latent space of the most essential features. It is observed during experimentation that the number of parameters and the size of the model increases with increase in number of neurons. The optimizer is chosen as Adam as it combines the benefit of both Adagrad and RMSprop with adaptive learning rate [43]. ReLU is chosen as the activation

Table 5 Hyper-parameters of benign and attack autoencoder

Model parameters	Values
Input neurons	39
Hidden layer	–
Number of neurons in hidden layer	–
Latent layer	1
Number of neurons in latent layer	10
Output neurons	39
Optimizer function	Adam
Activation function	ReLU
Dropout	–
Loss function	MSLE
Epoch	20
Batch size	128

function as it is a commonly used computationally efficient activation function to introduce non-linearity to the model. After 20 epochs, the MSLE loss stabilized for both Benign and Attack Autoencoder, suggesting that further epochs may not yield substantial improvements as evident in Fig. 3

The encoder compresses the training data D into a low-dimensional representation that captures the essential features of the data, and the decoder reconstructs the compressed representation back to the original input vector. It is expected that AE_{Benign} trained with only normal traffic data will recover any input as closely as feasible to the learned normal patterns. Similarly, AE_{Attack} trained with only Attack traffic data will recover any input as closely as feasible to the learned attack. The unknown sample batch is categorized as either benign or attack if the reconstruction error is below the threshold and the proposed algorithm is described in Algorithm 3.

4.2.2 Analysis and Detection of Unknown Network Traffic Using Proposed Scheme

The unknown network samples are prepared using the ToN-IoT dataset, which includes normal traffic samples and various attack categories, such as Backdoor, DoS, DDoS, Injection, MiTM, Password, Ransomware, Scanning, and XSS. The nine attack category samples are combined to create the attack class, while benign traffic samples are used to create the benign class. However, the labels of each class are removed to prepare the batch of unknown network samples. The detection of unknown attacks is preceded by training and optimizing the AE_{Benign} and AE_{Attack} autoencoders. The reconstruction errors of the two autoencoders are calculated as $\arg\min \sum_{i=1}^{d}(x_i - x_i')^2$, where x_i and x_i' are the input vector and reconstructed vector. The training procedure of both the autoencoders are optimized to minimize the reconstruction error as shown in Eq. 6

Algorithm 3 Proposed Algorithm for Detection of Unknown Attack

Require:
- D_{Benign} - Dataset of Known Benign Data (a d-dimensional vector)
- D_{Attack} - Dataset of Known Attack Data (a d'-dimensional vector)
- α_{benign} - Threshold of reconstruction error for benign
- α_{attack} - Threshold of reconstruction error for attack

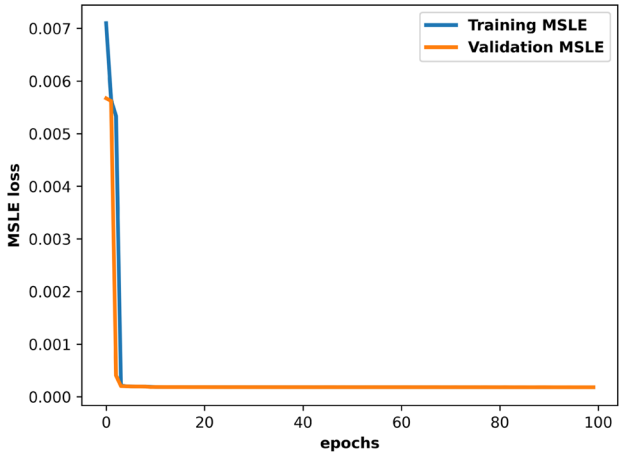
Ensure:
Predicted Class Labels for Unknown Network Samples

function BENIGNAUTOENCODER_TRAINING
  Initialize parameters $\theta = \{W^{\text{in}}, B^{\text{in}}\}, \theta' = \{W^{\text{hid}}, B^{\text{hid}}\}$
  for i in D_{Benign} **do**
    $f_\theta(x) \leftarrow s\left(\sum_{j=1}^{n} W_{ij}^{\text{in}} x_j + B_i^{\text{in}}\right)$
    $h_i \leftarrow f_\theta(x)$
    $g_{\theta'}(x) \leftarrow s\left(\sum_{j=1}^{n} W_{ij}^{\text{hid}} x_j + B_i^{\text{hid}}\right)$
    $x_i' \leftarrow g_{\theta'}(x)$
    $\text{RE}_{\text{Benign}} \leftarrow \arg\min \sum_{i=1}^{d}(x_i - x_i')^2$
    Train $\text{AE}_{\text{Benign}}$ $\arg\min_{\theta,\theta'} \frac{1}{d}\sum_{i=1}^{d} \text{RE}_{\text{Benign}}(x_i g_{\theta'}(f_\theta(x)))$
function ATTACKAUTOENCODER_TRAINING
  for i in D_{Attack} **do**
    Repeat steps 2 to 8 for training $\text{AE}_{\text{Attack}}$
function UNKNOWNATTACKDETECTION
  for i in D' **do**
    Calculate $\text{RE}_{\text{benign}}^{i}$ using $\text{AE}_{\text{Benign}}$
    Calculate $\text{RE}_{\text{Attack}}^{i}$ using $\text{AE}_{\text{Attack}}$
  $\mu_{\text{Benign\_AE}} \leftarrow \frac{1}{N_{\text{Benign}}}\sum_{i=1}^{N_{\text{Benign}}} \text{RE}_{\text{Benign},i}$
  ▷ Calculate mean of reconstruction errors of all samples with benign autoencoder
  $\mu_{\text{Attack\_AE}} \leftarrow \frac{1}{N_{\text{Attack}}}\sum_{i=1}^{N_{\text{Attack}}} \text{RE}_{\text{Attack},i}$
  ▷ Calculate mean of Reconstruction errors of all samples with Attack Autoencoder
  $\alpha_{\text{Benign}} \leftarrow \mu_{\text{Benign\_AE}} \pm 2 \cdot \sigma_{\text{Benign\_AE}}$
  $\alpha_{\text{Attack}} \leftarrow \mu_{\text{Atatck\_AE}} \pm 2 \cdot \sigma_{\text{Attack\_AE}}$
  for i in D' **do**
    if $\text{RE}_{\text{benign}}^{i} < \alpha_{\text{benign}}$ and $\text{RE}_{\text{Attack}}^{i} > \alpha_{\text{attack}}$ **then**
      Label i as Benign
    else if $\text{RE}_{\text{benign}}^{i} > \alpha_{\text{benign}}$ and $\text{RE}_{\text{Attack}}^{i} < \alpha_{\text{attack}}$ **then**
      Label i as Attack
    else
      Label i as Suspicious for further analysis

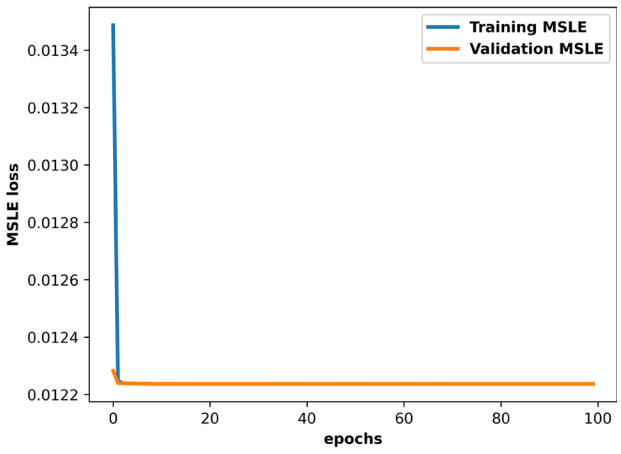
$$\arg\min_{\theta,\theta'} \frac{1}{d}\sum_{i=1}^{d} \text{RE}_{\text{Benign}}(x_i g_{\theta'}(f_\theta(x))) \tag{6}$$

Here, θ and θ' are the weight and bias parameters of the input layer and hidden layer, respectively.

The process of classifying the unknown network samples is done as described in Algorithm 3. The process involves utilizing the optimized and trained AE_{Benign} and AE_{Attack} to compute the reconstruction error of individual samples. The reconstruction error of a sample that a benign autoencoder has analyzed is represented as $\text{RE}_{\text{Benign}}$, while the reconstruction error of the same sample analyzed by an attack autoencoder is represented as $\text{RE}_{\text{Attack}}$. The classification of an unknown network sample to benign or attack involves



(a) MSLE Loss Training and Validation of Benign Autoencoder



(b) MSLE Loss Training and Validation of Attack Autoencoder

Fig. 3 Loss versus epochs

usage of a reconstruction error threshold. There are two thresholds; one for Benign which is referred to as α_{benign} and one for attack which is referred to as α_{attack}.

The calculation of the threshold requires two parameters: the mean of the reconstruction error of the samples, the standard deviation of the reconstruction error of all samples. The benign threshold α_{benign} is calculated using the mean of the reconstruction errors obtained for each sample with the benign autoencoder and the standard deviation of the reconstruction error of each sample. Thus, the benign threshold α_{Benign} is calculated as $\mu_{\text{Benign\_AE}} \pm 2 \cdot \sigma_{\text{Benign\_AE}}$. Similarly Attack Threshold is calculated as α_{Attack} is calculated as $\mu_{\text{Attack\_AE}} \pm 2 \cdot \sigma_{\text{Attack\_AE}}$.

A sample can be classified as benign if the computed value of $\text{RE}_{\text{Benign}}$ is less than the threshold value of α_{benign}, and the computed value of $\text{RE}_{\text{Attack}}$ is greater than the threshold value of α_{attack}. In the scenario where the value of $\text{RE}_{\text{Attack}}$ is lower than that of α_{attack}, and the value of $\text{RE}_{\text{Benign}}$ is higher than that of α_{benign}, the sample will be categorized as an attack. In instances where the value of $\text{RE}_{\text{Attack}}$ is lower than α_{attack} and the value of $\text{RE}_{\text{Benign}}$ is also lower than α_{benign}, the sample will be categorized as suspicious and subjected to further analysis. When the value of $\text{RE}_{\text{Attack}}$ exceeds α_{attack} and the value of $\text{RE}_{\text{Benign}}$ exceeds α_{benign}, the network sample is designated as suspicious for further evaluation. The adaptation of the threshold to the mean and standard deviation ensures that the threshold dynamically adjusts based on the characteristics of the reconstruction errors. This adaptivity allows the system to accommodate changes in the data distribution and maintain a balance between false positives and false negatives.

5 Results and Analysis

The experiments for the proposed work were conducted on a Dell G3 3500 laptop with Microsoft Windows 11 Pro 64-bit (22000.1574 build) operating system, Intel(R) Core (TM) i7-10750H processor at 2.50 GHz, and 16 GB Memory. The machine learning models were implemented using the Jupyter Notebook 6.4.5 application and Python 3.9.7 as the programming language. For data cleaning, extraction, and feature selection, the Pandas 1.3.4 and NumPy 1.22.4 frameworks were utilized, the Matplotlib 3.4.3 and Seaborn 0.11.2 frameworks for data visualization, and the Scikit-learn's 1.0.2 package for data analysis.

5.1 Performance Analysis of Proposed Known Attack Detection Methodology

The proposed classifier is tested on the hold-out dataset D", which contains samples of all classes and includes 36740,67906,48,123 and 08 observations for Benign, DDoS, DoS, Reconnaissance, and Theft. Figure 4 depicts the model's confusion matrix after performing 10-fold cross-validation using the D"NF-BoT-IoT dataset. The diagonal elements of the confusion matrix reflect the number of examples from each class that our method accurately classified. According to the confusion matrix, the false-positive rates for each class in the BoT-IoT dataset are near zero, indicating that the classifier is promising. The proposed stacked classifier has zero misclassifications in the Benign, DDoS, DoS, and Theft categories and two in the category of Reconnaissance. The high detection rate by the proposed classifier on various attack categories is evident from the confusion matrix. It is worth noting that all 8 samples of the Theft class are correctly classified with no resampling. Benign and DDoS have a higher

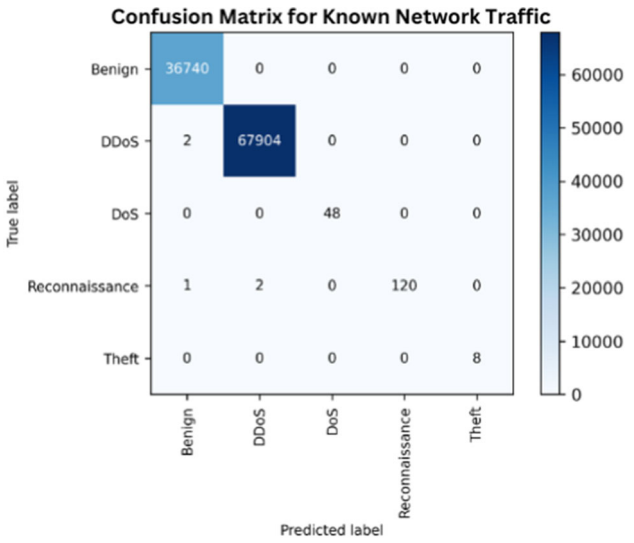
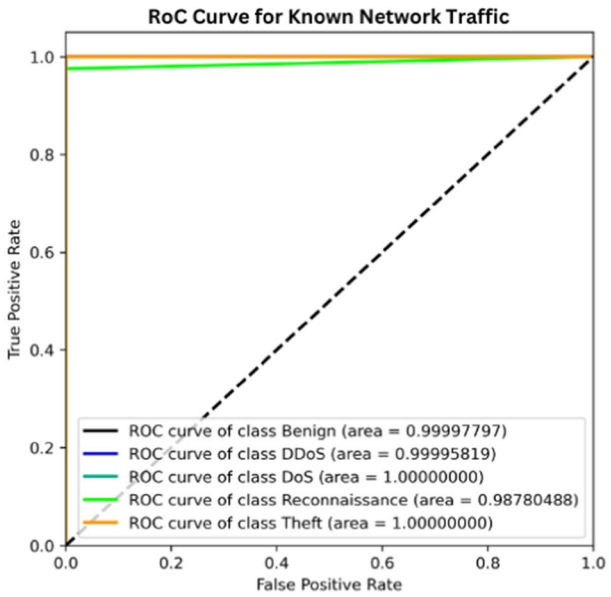
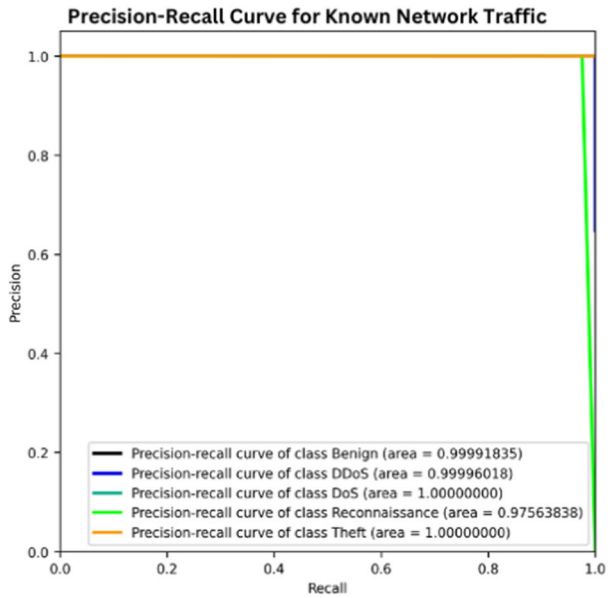


Fig. 4 Confusion matrix for stacked multi-classifer for known attack detection with BoT-IoT dataset



(a) RoC Curve for Stacked Multi-Classifier



(b) Precision-Recall Curve for Stacked Multi-Classifer

Fig. 5 Performance analysis of known attack detection with proposed classifier

number of samples in comparison to DoS, Reconnaissance, and Theft.

The Stacked MLP classifiers designed for each class with selected features of each class using proposed LGLMVQ have given accuracy, precision, recall and F1 score values of 0.999−0.9989. The 0.9989 value of precision indicates very few false-positive predictions. An exceptionally high RoC-AUC score of 0.9975 indicates that the proposed model is very effective in distinguishing and classifying all majority and minority classes.

The ROC curve and precision–recall curve of the detection model are drawn in Fig. 5a, b. ROC curves depict the trade-off between TPR and FPR for a prediction model with various probability thresholds, whereas precision–recall curves describe the trade-off between the TPR and the positive predictive value for a predictive model with varying probability thresholds. As seen in Fig. 5a, b, both curves distinguish all classes while covering 100 percent or close to 100 percent of the region. We evaluated the performance of our model individually for each class, and the micro-average value of our model's RoC and precision–recall curves indicates outstanding performance. The exceptionally high ROC scores for both minority and majority classes suggest that the model has likely learned distinct and well-separated features for these classes, making their identification easier. The precision–recall curves are excellent metrics for evaluating the performance of the model under imbalanced scenario. The exceptionally high scores for all classes except Reconnaissance indicate very high precision and recall scores. The negligible misclassifications of the Reconnaissance class to Benign class have resulted in a score of 0.9756. The slightly lower PR curve for Reconnaissance"may be due to the more complex and subtle nature of reconnaissance activities, which can be challenging to distinguish from Benign traffic.

5.2 Comparative Analysis of Proposed Known Attack Detection with Existing Works

The exceptional performance of proposed stacked MLP multi-classifiers as shown in Table 6 with nearly perfect accuracy (0.9999), precision (0.9989), recall (0.9921), F1

Table 6 Comparative analysis of proposed algorithm for known attack detection with existing works

Research work	Technique used	Accuracy	Precision	Recall	F1 score	ROC-AUC score
[44] 2023	Multi Graph Neural Network	0.9891	0.98896	0.9874	0.9891	0.75
[45] 2023	DNN Federated	0.9308	0.931	0.9293	0.9301	0.9595
[45] 2023	LSTM Federated	0.9257	0.9257	0.9248	0.9252	0.9518
[46] 2023	Deep Neural Network	0.8691	0.7982	0.8521	0.8242	0.8215
[47] 2023	Extra Trees	0.9982	0.9912	0.9891	0.9901	0.9734
Proposed work	Stacked MLP	0.9999	0.9989	0.9921	0.9955	0.9975

score (0.9955), and ROC-AUC score (0.9975) outperforms existing works in known attack detection across all metrics. The comparative analysis is undertaken with only research works using the NF-BoT-IOT dataset. The research work [44] using concatenated multi-graph gave high accuracy, precision, recall and F1 score; however, the ROC-AUC score is very low of 0.75. The federated approach using DNN and LSTM [45] has performed better than [44]; however, its ROC score is lower than that of the proposed approach. Deep neural network approach proposed in [46] has a lower score among all the compared works. The Extra Trees approach described in [47] gave a decent classification score, however, the proposed approach gave outstanding performance with the stacked classwise MLP Classifier in distinguishing even the most minor classes with only 8 samples.

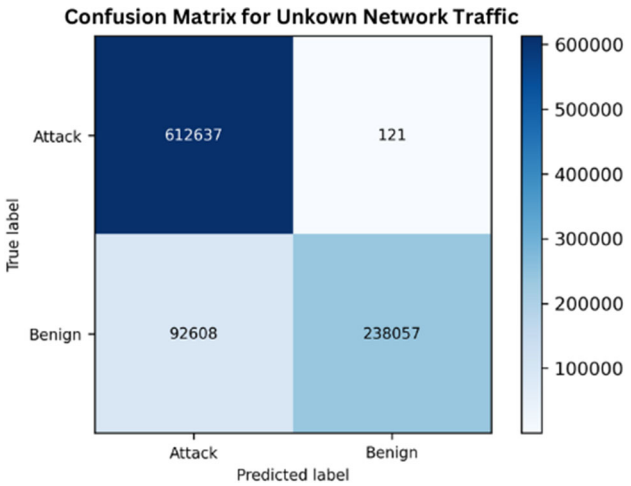


Fig. 6 Confusion matrix for unknown attack detection with ToN-IoT dataset

5.3 Performance Analysis of Proposed Unseen Attack Detection Methodology

5.3.1 NF-ToN-IoT Dataset

The ToN-IoT dataset is used to evaluate the performance of the proposed two-level shallow autoencoder for detecting unknown attacks. The labels of 612,758 Attack samples and 330,665 Benign samples are removed and batches of unknown network samples are prepared. The methodology for differentiating between benign and attack samples involves a pair of benign and attack threshold values. The calculation of the benign threshold and attack threshold is explained in the subsection 4.2.2. Upon establishing the aforementioned thresholds, any subsequent sample exhibiting a reconstruction error less than attack threshold and more than the benign threshold shall be categorized as an attack, whereas any sample exhibiting a reconstruction error below the benign threshold and greater than the attack threshold shall be classified as benign. The instances exhibiting reconstruction errors lesser than both the attack threshold and benign threshold and more than both the attack threshold and benign threshold are categorized as suspicious and necessitate additional scrutiny.

The Benign autoencoder is trained on Benign data and Attack autoencoder is trained on attack samples from the BoT-IoT dataset. To evaluate the performance of the proposed method, a test set consisting of benign and attack samples from the ToN-IoT is used. The results given in Table 9 show that the proposed method can distinguish between unseen benign and attack samples with high accuracy, precision, recall and F1 Score. The misclassifications illustrated in Fig. 6 validate the efficiency of the proposed approach in detecting unknown samples. This suggests that the proposed method is effective in distinguishing between benign, attack, and unknown samples, and has potential for use in real-world scenarios for detecting unknown attacks.

In conclusion, the proposed method effectively detects unknown network samples by thresholding the benign and attack autoencoder. The results presented in Table 9 demonstrate the method's ability to accurately classify unknown samples and detect previously unseen attack types, suggesting its potential as a practical approach for network intrusion detection.

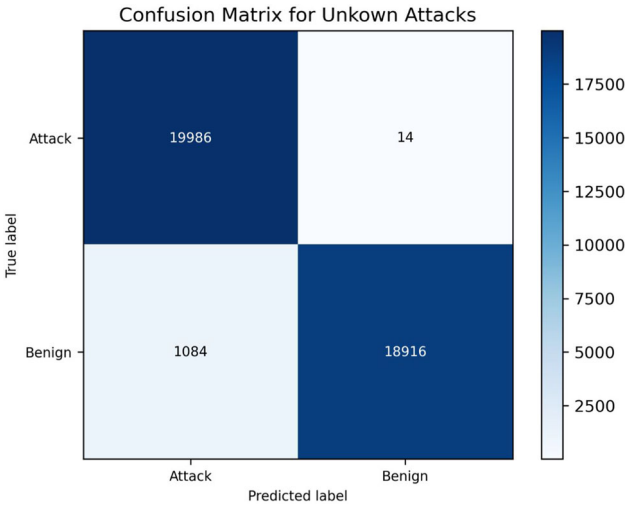


Fig. 7 Confusion matrix for unknown attack detection with NF-CIC-IDS-2018

5.3.2 NF-CIC-IDS 2018 Dataset

The generalizability of the proposed method for detection of unseen attacks is evaluated on NF-CIC-IDS 2018 dataset. The NF-CIC-IDS 2018 dataset consists of samples of benign traffic and attack traffic belonging to Bruteforce, BoT, DoS, DDoS, Infiltration and Web Attacks category. The Benign and Attack Autoencoder trained using NF-BoT-IoT are evaluated using the samples of unlabeled benign and attack traffic of NF-CIC-IDS 2018. The evaluation results of benign and attack autoencoder for detecting unseen samples are summarized in the confusion matrix and classification report shown in Fig. 7 and Table 8. The number of misclassifications out of 20,000 attack samples is only 14. Similarly, the number of Benign samples wrongly classified as attack samples is 1084. The evaluation results add value as very few attack samples are misclassified as Benign. The classwise performance results illustrated in Table 8 show that the proposed technique for unseen attack detection performs well achieving high accuracy, precision, recall, and F1 score. Further in Table 7, the average classification scores of each dataset is illustrated. The proposed technique sufficiently high classification scores and negligible false positive and false negative rates.

There are state of works done in area of unknown attack detection using the same datasets, and one of the most comprehensive performance analysis of deep learning, supervised and unsupervised machine learning classifiers in detecting unknown attacks is done in [14]. The dataset employed for evaluating the efficacy of the classifiers encompasses a collection of attack samples with an unknown distribution, spanning from 0.5% to 0.60%. The remaining samples consist exclusively of the pre-existing known samples. In the context of supervised learning classifiers, it was observed that the Decision Tree model exhibited superior performance. Specifically, the aforementioned model achieved an accuracy rate of 0.935% and a Recall value of 0.73%. The recall rate for unidentified samples, as determined through empirical analysis, is a mere 0.40%. Similarly, the best possible recall achieved through the utilization of unsupervised learning techniques for unknown samples of unknown is quantified at 0.58%. The meta-supervised and meta-unsupervised approaches exhibited superior performance in terms of recall for unknown samples, achieving a range of 0.38% to 0.65%.

The proposed technique using adaptive thresholding for unknown attack detection gave comparatively superior performance. The experimental results obtained from the proposed methodology on the NF-ToN-IoT dataset are reported herein. The accuracy achieved by the proposed approach is 0.9017, indicating a high level of correctness in identifying unknown attacks. Furthermore, the precision of the proposed approach is measured at 0.9998, signifying a minimal number of false positives in the detection process. The recall, which measures the ability to identify all instances of attacks, is reported to be 0.8687, indicating a satisfactory level in the detection process. Lastly, the F1 score, which combines the precision and recall metrics, is calculated to be 0.9296, indicating a balanced performance in terms of both precision and recall. These results demonstrate the effectiveness of the proposed methodology in accurately detecting unknown attacks on the NF-ToN-IoT dataset. The proposed technique exhibited superior performance on the NF-CIC-IDS 2018 dataset as well, achieving notable accuracy, precision, recall, and F1 Score values of 0.9726, 0.9996, 0.9486, and 0.97333, respectively. It is of utmost importance to acknowledge that the test samples employed in this study are derived from two separate datasets, wherein the attack types included in these samples differ entirely from the attack classes used for training.

Table 7 Performance metrics on unseen network samples

Dataset	Accuracy	Precision	Recall	F1 score	False positive rate	False negative rate
NF-ToN-IoT	0.9017	0.9998	0.8687	0.9296	0.0005	0.1313
NF-CIC-IDS-2018	0.9726	0.9993	0.9486	0.9733	0.0007	0.0514

Table 8 Classwise performance metrics for NF-CIC-IDS 2018

Class	Accuracy	Precision	Recall	F1 score
Attack	0.972	1.000	0.950	0.970
Benign	0.972	0.950	1.000	0.970

Table 9 Classwise performance metrics for NF-ToN-IoT

Class	Accuracy	Precision	Recall	F1 score
Attack	0.9017	1.000	0.870	0.930
Benign	0.9017	0.720	1.000	0.840

6 Discussion

The primary objective of this research is to develop and evaluate a machine learning classifier to accurately identify both known and unknown attacks. The evaluation of the classifier focused on its ability to classify various attack categories with high levels of accuracy, precision, recall, and F1 score. Additionally, it was important to ensure that the classifier maintained a low false-positive rate throughout the classification process. The findings of our research showcase the efficacy of the proposed stacked multilayer perceptron (MLP) classifier in accurately categorizing attack and benign traffic. The model demonstrated remarkable performance by achieving exceptionally high accuracy, precision, recall, and F1 score values across a wide range of attack categories. The proposed methodology for the stacked MLP classifier involves the utilization of class-specific MLP classifiers. These classifiers are trained using class-specific features that have been selected through the implementation of proposed LGLMVQ algorithm. The results indicate that the chosen features and the learning algorithm effectively captured unique attributes associated with each class.

The present study also outlines a novel approach for detecting unknown attacks by utilizing shallow benign and attack autoencoders, coupled with adaptive reconstruction error thresholding. The experimental results demonstrate that this proposed methodology yields competitive classification scores for both the bench mark datasets in comparison with existing state-of-the-art techniques. In the context of the present study, a comprehensive evaluation of our proposed approach was conducted, with a particular focus on comparing its performance against existing research. The evaluation encompassed various metrics, including accuracy, precision, recall, and F1 score. Through this comparative analysis, it was observed that our proposed approach exhibited superior performance in all aforementioned metrics when compared to previous methods. The ROC-AUC score exhibits a noteworthy increase, thereby signifying the model's proficiency in accurately discerning between attack and benign traffic. The superior performance exhibited by our model can be attributed to the implementation of class-specific Multi-Layer Perceptron (MLP) classifiers designed for each attack category, coupled with the incorporation of selected features obtained through the novel LGLMVQ technique. The aforementioned methodology exhibits a high degree of efficacy in discerning and differentiating various attack patterns. Also, the competitive performance of the benign and attack autoencoder in classifying unknown samples is due to the adaptive reconstruction error thresholding.

In light of the encouraging outcomes obtained, it is imperative to duly recognize and address specific constraints that may impact the validity and generalizability of our findings. The evaluation of performance is contingent upon the utilization of precise datasets with netflow-based features, while the inclusion of real-world scenarios may introduce additional complexities. Furthermore, it is worth noting that the suggested methodology may necessitate additional enhancements to ensure its scalability when applied to larger and more intricate networks. The efficacy of the adaptive reconstruction error thresholding technique for unknown attack detection may be compromised when applied to non-Gaussian distributions.

The present study yields noteworthy practical ramifications in the domain of network security for Internet-of-Things (IoT) systems. The proposed technique exhibits a commendable level of precision and an impressively low rate of false positives. As a result, it holds significant potential as a robust technique for identifying both known and unknown network attacks. Additionally, the reduction of false alarms can alleviate the burden on security personnel, allowing them to focus their attention on genuine attacks and streamline incident response processes. Consequently, organizations stand to gain enhanced threat detection capabilities and improved operational efficiency through the implementation of advanced intrusion detection techniques.

7 Conclusion and Future Scope

This research paper introduces a novel methodology for effectively detecting both known and unknown attacks within IoT networks. The experimental findings underscore the remarkable performance of the proposed stacked multilayer perceptron (MLP) classifier in identifying known attacks, achieving exceptional accuracy, precision, recall, and F1 score metrics within imbalanced class contexts. The utilization of classwise MLP classifiers, combined with innovative feature selection through the Localized Generalized Matrix Learning Vector Quantization (LGMLVQ) technique, showcases efficacy in precisely detecting and distinguishing both minority and majority classes. Furthermore, the proposed methodology extends its capabilities to the detection of

unknown attacks on benchmark datasets, yielding competitive classification scores.

While the results are promising, it is crucial to acknowledge certain limitations and identify avenues for future research. Rigorous optimization and scalability testing are imperative to validate the model's applicability in larger and more intricate network environments. Ongoing research efforts should focus on seamlessly integrating the proposed model into real-time network monitoring systems. In summary, this research study marks a substantial advancement in identifying both known and unknown attacks in IoT networks. The introduced contributions encompass a refined matrix learning vector quantization approach for feature selection in imbalanced datasets, a stacked multi-classifier leveraging MLP for known attack detection, and a shallow autoencoder designed for zero-day attack detection. These contributions collectively form a comprehensive solution to enhance IoT security, demonstrating competitive performance across diverse metrics and datasets.

Author Contributions Deepa Krishnan contributed to conceptualization of methodology, resources, and writing—original draft preparation; Pravin Shrinath contributed to writing—review and editing. All authors have read and agreed to the published version of the manuscript

Funding Not applicable.

Data Availability All data generated or analyzed during this study are included in this published article. The datasets used in this study are publicly available and links to them can be found in the references section of this article.

Declarations

Conflict of interest The authors declare no conflict of interest.

Consent to Participate Not applicable.

Consent for Publication Not applicable.

References

1. Maraveas, C.; Piromalis, D.; Arvanitis, K.G.; Bartzanas, T.; Loukatos, D.: Applications of Iot for optimized greenhouse environment and resources management. Comput. Electron. Agric. **198**, 106993 (2022)
2. Verma, D.; Singh, K.R.B.; Yadav, A.K.; Nayak, V.; Singh, J.; Solanki, P.R.; Singh, R.P.: Internet of things (Iot) in nano-integrated wearable biosensor devices for healthcare applications. Biosensors Bioelectron. X **11**, 100153 (2022)
3. Sha, K.; Wei, W.; Yang, T.A.; Wang, Z.; Shi, W.: On security challenges and open issues in internet of things. Future Gener. Comput. Syst. **83**, 326–337 (2018)
4. Rekeraho, A.; Cotfas, D.T.; Cotfas, P.A.; Balan, T.C.; Tuyishime, E.; Acheampong, R.: Cybersecurity challenges in Iot-based smart renewable energy. Int. J. Inf. Secur. (2023).
5. Haddad Pajouh, H.; Dehghantanha, A.; Parizi, R.M.; Aledhari, M.; Karimipour, H.: A survey on internet of things security: requirements, challenges, and solutions. Internet of Things **14**, 100129 (2021)
6. Ammar, M.; Russello, G.; Crispo, B.: Internet of things: a survey on the security of Iot frameworks. J. Inf. Secur. Appl. **38**, 8–27 (2018)
7. Karale, A.: The challenges of iot addressing security, ethics, privacy, and laws. Internet of Things **15**, 100420 (2021)
8. Conti, M.; Dehghantanha, A.; Franke, K.; Watson, S.: Challenges and opportunities. In: Internet of Things Security and Forensics (2018)
9. Bhaskara, S.; Rathore, S.S.: Causal effect analysis-based intrusion detection system for Iot applications. Int. J. Inf. Secur. 1–16 (2023).
10. Zeghida, H.; Boulaiche, M.; Chikh, R.: Securing MQTT protocol for Iot environment using ids based on ensemble learning. Int. J. Inf. Secur. 1–12 (2023)
11. Alkasassbeh, M.; Al-Haj, B.S.: Intrusion detection systems: a state-of-the-art taxonomy and survey. Arab. J. Sci. Eng. **48**(8), 10021–10064 (2023)
12. Lee, S.; Abdullah, A.; Jhanjhi, N.; Kok, S.: Classification of botnet attacks in iot smart factory using honeypot combined with machine learning. PeerJ Comput. Sci. **7**, e350 (2021)
13. Ahmad, R.; Alsmadi, I.: Machine learning approaches to iot security: a systematic literature review. Internet of Things **14**, 100365 (2021)
14. Zoppi, T.; Ceccarelli, A.; Puccetti, T.; Bondavalli, A.: Which algorithm can detect unknown attacks? Comparison of supervised, unsupervised and meta-learning algorithms for intrusion detection. Comput. Secur. 103107 (2023).
15. Talukder, M.A.; Hasan, K.F.; Islam, M.M.; Uddin, M.A.; Akhter, A.; Yousuf, M.A.; Alharbi, F.; Moni, M.A.: A dependable hybrid machine learning model for network intrusion detection. J. Inf. Secur. Appl. **72**, 103405 (2023)
16. Nazir, A.; Khan, R.A.: A novel combinatorial optimization based feature selection method for network intrusion detection. Comput. Secur. **102**, 102164 (2021)
17. Guo, Y.: A review of machine learning-based zero-day attack detection: challenges and future directions. Comput. Commun. (2022).
18. Sameera, N.; Shashi, M.: Deep transductive transfer learning framework for zero-day attack detection. ICT Exp. **6**(4), 361–367 (2020)
19. Zhao, J.; Shetty, S.; Pan, J.W.; Kamhoua, C.; Kwiat, K.: Transfer learning for detecting unknown network attacks. EURASIP J. Inf. Secur. **2019**, 1–13 (2019)
20. Çavuşoğlu, Ü.; Akgun, D.; Hizal, S.: A novel cyber security model using deep transfer learning. Arab. J. Sci. Eng. 1–10 (2023).
21. Kim, J.-Y.; Seok-Jun, B.; Cho, S.-B.: Zero-day malware detection using transferred generative adversarial networks based on deep autoencoders. Inf. Sci. **460**, 83–102 (2018)
22. Shafiq, M.; Tian, Z.; Bashir, A.K.; Du, X.; Guizani, M.: Iot malicious traffic identification using wrapper-based feature selection mechanisms. Comput. Secur. **94**, 101863 (2020)
23. Mafarja, M.; Heidari, A.A.; Habib, M.; Faris, H.; Thaher, T.; Aljarah, I.: Augmented whale feature selection for Iot attacks: structure, analysis and applications. Future Gener. Comput. Syst. **112**, 18–40 (2020)
24. Huong, T.T.; Bac, T.P.; Long, D.M.; Thang, B.D.; Binh, N.T.; Luong, T.D.; Phuc, T.K.: Lockedge: low-complexity cyberattack detection in Iot edge computing. IEEE Access **9**, 29696–29710 (2021)
25. Asadi, M.; Jamali, M.A.J.; Parsa, S.; Majidnezhad, V.: Detecting botnet by using particle swarm optimization algorithm based on voting system. Futur. Gener. Comput. Syst. **107**, 95–111 (2020)
26. D'hooge, L.; Wauters, T.; Volckaert, B.; De Turck, F.: Inter-dataset generalization strength of supervised machine learning methods for intrusion detection. J. Inf. Secur. Appl. **54**, 102564 (2020)

27. Ahmad, R.; Alsmadi, I.; Alhamdani, W.; Tawalbeh, L.: A deep learning ensemble approach to detecting unknown network attacks. J. Inf. Secur. Appl. **67**, 103196 (2022)
28. Moustafa, N.; Keshk, M.; Choo, K.-K.R.; Lynar, T.; Camtepe, S.; Whitty, M.: Dad: a distributed anomaly detection system using ensemble one-class statistical learning in edge networks. Future Gener. Comput. Syst. **118**, 240–251 (2021)
29. Rodríguez, E.; Valls, P.; Otero, B.; Costa, J.J.; Verdú, J.; Pajuelo, M.A.; Canal, R.: Transfer-learning-based intrusion detection framework in Iot networks. Sensors **22**(15), 5621 (2022)
30. He, Z.; Rezaei, A.; Homayoun, H.; Sayadi, H.: Deep neural network and transfer learning for accurate hardware-based zero-day malware detection. In: Proceedings of the Great Lakes Symposium on VLSI, vol. 2022, pp. 27–32 (2022)
31. Ahmad, R.; Alsmadi, I.; Alhamdani, W.; Tawalbeh, L.: Zero-day attack detection: a systematic literature review. Artif. Intell. Rev. 1–79 (2023).
32. Zoppi, T.; Gharib, M.; Atif, M.; Bondavalli, A.: Meta-learning to improve unsupervised intrusion detection in cyber-physical systems. ACM Trans. Cyber-Phys. Syst. (TCPS) **5**(4), 1–27 (2021)
33. Blaise, A.; Bouet, M.; Conan, V.; Secci, S.: Detection of zero-day attacks: an unsupervised port-based approach. Comput. Netw. **180**, 107391 (2020)
34. Zoppi, T.; Ceccarelli, A.; Bondavalli, A.: Unsupervised algorithms to detect zero-day attacks: strategy and application. IEEE Access **9**, 90603–90615 (2021)
35. Hindy, H.; Atkinson, R.; Tachtatzis, C.; Colin, J.-N.; Bayne, E.; Bellekens, X.: Utilising deep learning techniques for effective zero-day attack detection. Electronics **9**(10), 1684 (2020)
36. Sarhan, M.; Layeghy, S.; Moustafa, N.; Portmann, M.: Netflow datasets for machine learning-based network intrusion detection systems. In: Big Data Technologies and Applications: 10th EAI International Conference, BDTA 2020, and 13th EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, December 11, 2020, Proceedings 10, pp. 117–135. Springer (2021).
37. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B.: Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: bot-iot dataset. Futur. Gener. Comput. Syst. **100**, 779–796 (2019)
38. Moustafa, N.: A new distributed architecture for evaluating AI-based security systems at the edge: network ton_iot datasets. Sustain. Cities Soc. **72**, 102994 (2021)
39. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp **1**, 108–116 (2018)
40. Kohonen, T.: The self-organizing map. Proc. IEEE **78**(9), 1464–1480 (1990)
41. Sato, A.; Yamada, K.: Generalized learning vector quantization. In: Advances in Neural Information Processing Systems, vol. 8 (1995).
42. Schneider, P.; Biehl, M.; Hammer, B.: Adaptive relevance matrices in learning vector quantization. Neural Comput. **21**(12), 3532–3561 (2009)
43. Hassan, E.; Shams, M.Y.; Hikal, N.A.; Elmougy, S.: The effect of choosing optimizer algorithms to improve computer vision tasks: a comparative study. Multimedia Tools Appl. **82**(11), 16591–16633 (2023)
44. Altaf, T.; Wang, X.; Ni, W.; Yu, G.; Liu, R.P.; Braun, R.: A new concatenated multigraph neural network for iot intrusion detection. Internet of Things **22**, 100818 (2023)
45. Sarhan, M.; Layeghy, S.; Moustafa, N.; Portmann, M.: Cyber threat intelligence sharing scheme based on federated learning for network intrusion detection. J. Netw. Syst. Manag. **31**(1), 3 (2023)
46. Vishwakarma, M.; Kesswani, N.: Dids: a deep neural network based real-time intrusion detection system for Iot. Decision Anal. J. **5**, 100142 (2022)
47. Fraihat, S.; Makhadmeh, S.; Awad, M.; Al-Betar, M.A.; Al-Redhaei, A.: Intrusion detection system for large-scale Iot netflow networks using machine learning with modified arithmetic optimization algorithm. Internet of Things 100819 (2023).