Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 8614903, 11 pages
https://doi.org/10.1155/2022/8614903

Research Article

An Intrusion Detection System for the Internet of Things Based on the Ensemble of Unsupervised Techniques

Yao Wang[1,2] **Guozi Sun**[1,2] **Xiaochun Cao**[3] **and Jiale Yang**[1,2]

[1]*School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China*
[2]*Key Laboratory of Urban Land Resources Monitoring and Simulation, MNR, Shenzhen 518000, China*
[3]*State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China*

Correspondence should be addressed to Guozi Sun; sun@njupt.edu.cn

Received 4 November 2021; Revised 18 May 2022; Accepted 21 May 2022; Published 28 July 2022

Academic Editor: Mingqian Liu

Recently, machine learning techniques, especially supervised learning techniques, have been adopted in the Intrusion Detection System (IDS). Due to the limit of supervised learning, most state-of-the-art IDSs do not perform well on unknown attacks and incur high computational overhead in the Internet of Things (IoT). To overcome these challenges, we propose a novel IDS based on unsupervised techniques, namely, UTEN-IDS. UTEN-IDS uses the ensemble of autoencoders to handle the network data and performs the anomaly detection by an Isolation Forest algorithm. The effectiveness of the proposed method is verified using two benchmark datasets. The results show that our approach has significant advantages in classification performance and proves its utility in the IoT network when compared to other approaches.

1. Introduction

Network security has become an essential challenge in information systems, especially in the Internet of Things (IoT). IoT is a network of devices such as computers and sensors. The devices in IoT are likely to be vulnerable to various attacks [1]. According to the report [2], two-thirds of enterprises have already experienced a cybersecurity incident linked with their IoT devices. Some common cyberattacks, such as Distributed Denial of Service (DDoS), call for further research [3] and pose serious threats to IoT. The Intrusion Detection System (IDS) has been created to detect attacks in modern networks, including IoT networks. By analyzing the traces of network intrusion, IDS can detect attacks and raise alarms in real time.

However, traditional IDSs suffer from weaknesses such as a low detection rate and high false alarm rate [4]. Nowadays, Machine Learning (ML) techniques are used in IDSs to overcome weaknesses. ML is a type of interdisciplinary field that emulate human intelligence. Supervised learning and unsupervised learning are two types of ML [4]. The difference between them lies in the use of labeled data. The state-of-the-art intrusion detection methods are almost supervised learning methods, which means these methods need both attack and normal data for model training. They have achieved a high detection rate for some well-known attacks. However, the labeling process is completed manually, and the samples may be mislabeled. Moreover, the labels need to be updated regularly. Because of the lack of relevant labels, these methods fail to detect increasing novel attacks in the current IoT network.

Deep learning (DL) is one branch of ML, while the neural network is the key component of DL. Compared to ML models, DL models deal with big data effectively and gain more and more attention. Some state-of-the-art neural networks, such as Reservoir Computing Network (RCN), are applied to target detection [5], text classification, cybersecurity, and so on. In intrusion detection, DL models with complex structures still require an amount of labeled data for training and cost many computational resources. However, the high computational overhead makes these models challenging to be used for IoT devices with few resources [6].

In general, the existing solutions are in the following difficulties:

(a) Over-reliance on supervised techniques.

(b) High computational overhead.

(c) Poor detection performance on unseen attacks.

We propose an Unsupervised Technique ENsemble-based IDS (UTEN-IDS) to solve the above challenges. UTEN-IDS combines unsupervised technologies in an ensemble way. For example, an ensemble of lightweight autoencoders (AEs) in UTEN-IDS is used to reconstruct the input data and calculate the Root Mean Square Errors (RMSEs). After the reconstruction, the Isolation Forest (IF) [7] uses the RMSEs for final classification. Excellent detection performance is the most fundamental requirement for an IDS [8]. We use the CES-CIC-IDS 2018 dataset [9] and the MQTT-IOT-IDS2020 dataset [10] to verify the performance. The results show that UTEN-IDS has a better detection performance. Because UTEN-IDS uses lightweight models and generates low overhead, it can be applied to IoT network.

The contributions of this paper are as follows:

(a) We propose UTEN-IDS, an unsupervised technique-based IDS. The IDS is composed of unsupervised techniques, such as AE and IF. AEs learn the input data without the label, and IF is used to make the final decision. AE can extract crucial features from data, and the IDS takes full advantage of AE in intrusion detection to improve the detection rate.

(b) We propose a feature clustering method, which uses the Mean Shift algorithm to cluster the features based on the correlation of features. Without any predefined parameters, this method divides the closely related features into the same cluster for the training of AE.

(c) We evaluate the performance of the proposed method. The experimental results suggest that the proposed UTEN-IDS is superior to the state-of-the-art approaches.

The rest of the paper is organized as follows: Section 2 discusses the related research; Section 3 introduces the theoretical knowledge; Section 4 describes the proposed method; Section 5 presents the experiment setup and results; Section 6 concludes the paper.

2. Related Work

The ML classifiers used in IDSs, such as Decision Tree (DT) [11], Random Forest (RF) [12], Support Vector Machine (SVM) [13], and neural network [14], can analyze the features of network traffic to distinguish malicious activities from network traffic. Ingre et al. [15] proposed a detection method based on the DT classifier, and the NSL-KDD [16] dataset was used to test the performance. The experimental results showed that the proposed method could achieve high detection rates. Zhang et al. [17] presented an approach based on the Convolutional Neural Network (CNN) and sampling technique, with an accuracy of 98.82% on the UNSW-NB15 dataset [18] for binary classification. In [19], the authors designed a feedback mechanism to detect errors based on the recent detection results. They used Multilayer Perceptron (MLP) as the classifier, with an accuracy of 97.66% on the NSL-KDD. As one paradigm of ML, the ensemble method constructs the ensemble with base classifiers to improve accuracy [20]. Voting is the simplest way to implement the ensemble method. Gu et al. [21] proposed a method based on SVM ensemble and feature augmentation, with an accuracy of 99.41% on the NSL-KDD dataset. In their research, the quality-improved technique [22] provided high-quality training data, and an SVM ensemble was applied for classification. In [23], the authors proposed a Voting-based Neural Network (VNN). The method creates various neural network models and picks the best models from them. The chosen models are used to perform the detection by majority voting. Although the above methods achieve high accuracy, they need labeled data for training and are insufficient to provide security against unknown attacks.

Supervised learning is more common than unsupervised learning for intrusion detection. However, the unlabeled traffic data generated in the network is suitable for unsupervised learning, such as Unsupervised Feature Selection (UFS) [24]. In [25], the authors used one UFS technique to select features from intrusion detection datasets, and Redundancy Penalization (RP) technique based on mutual information was applied to filter the features further. Carrasco and Sicilia [26] proposed an unsupervised neural network model based on Natural Language Processing (NLP), tested against the UNSW-NB15 dataset, and it achieved 99.20% precision and 82.07% recall. Bohara et al. [27] used two unsupervised clustering algorithms for intrusion detection. But this method needs a combination of host and logs to achieve a good detection performance. Mingqiang et al. [28] used a graph-based algorithm to cluster the data and an outlier detection method to decide which cluster was malicious. This method remains computationally expensive due to the complexity of the algorithm.

In [29], the authors proposed Kitsune, an online intrusion detection method based on AE. Kitsune reconstructed the input data through AEs and used RMSE to record the reconstruction error. The maximum RMSE in the training phase was stored as a classification threshold. Kitsune performed anomaly detection tasks in a semi-supervised way. However, the threshold obtained in this way may be inaccurate. In [30], the authors proposed AE-IDS, used the RF algorithm to select features, and grouped the features by affinity propagation clustering [31], and AEs reconstructed the feature groups. AE-IDS selects features in a supervised way, and selected features are grouped based on the

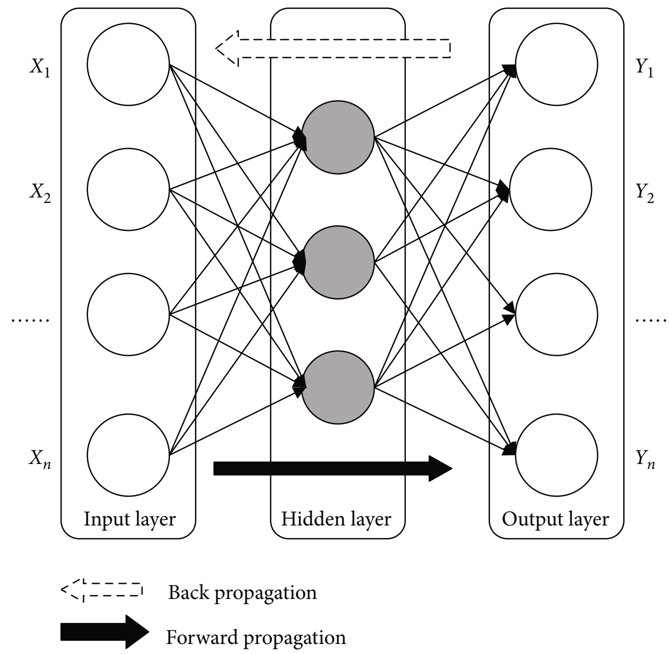

Figure 1: AE topological structure.

average, ignoring the correlation between the features. Zavrak and İskefiyeli [32] used Variational Autoencoder (VAE) for intrusion detection and calculated Reconstruction Probability (RP). The RP value is used for classification, but the proper threshold value was still hard to determine. In [33], the authors used AE to classify the intrusion behaviours, and IF is applied twice for reclassification. However, this method did not consider the effect of redundant features on detection performance and the shortcoming of IF when tackling high-dimensional data. The detection performance also depends on the predefined value of the threshold. Unlike [33], we use IF to perform the final prediction based on the output of AEs, and our solution does not require a predefined threshold.

Although some achievements have been made in AE-based intrusion detection, few researchers focus on the choice of threshold for classification. It is critical to find one proper threshold which directly affects the performance of the classifier. In this paper, the proposed method can address this issue. UTEN-IDS is inspired by the existing research, but it is pretty different from the above methods. It is composed of two layers. The first layer consists of AE, and IF is in the second layer. IF plays the role of the threshold by detecting unknown attacks adaptively.

3. Background

3.1. Autoencoder. AE is one kind of neural network. It is used to output an accurate data representation by learning the the low-dimensional features [34]. AE can transform the data into a lower-dimensional space [35], and we can perform the classification with the transformation differences.

Figure 1 shows the structure of the AE with three layers. Suppose that the input sample X has n features, and X_i denotes the ith feature where $1 \leq i \leq n$. Each feature corresponds to a neuron in the input layer separately. Encoding and decoding are two main working phasesof AE. The data is transformed from the input layer to the hidden layer in the encoding phase, while the transformation from the hidden layer to the output layer is described as decoding. X is reconstructed after the two phases to obtain the output Y, and Y is close to X. It is considered that an AE learns the function Q: $Q(X) \approx X$.

The execution process of AE is not complex. Let L_j denotes the jth layer in AE, and $\|L_j\|$ denotes the total number of neurons in L_j. Thus, the weights which connect L_j to L_{j+1} can be described as the matrix W_j with $\|L_j\|$ rows and $\|L_{j+1}\|$ columns. The bias of connection is denoted as the vector b_j with $\|L_{j+1}\|$ dimensions. Let τ record the total parameters for all layers, and $\tau = (W, b)$. τ is randomly generated when AE is initialized. Forward propagation is used to activate the neural network layer by layer. L_2 is activated by W_1, and L_3 is activated by $W_2 \ldots$, the output layer is also activated by doing so. Let Z_j be $\|L_j\|$-dimension vector generated by neurons in L_j, the calculation of Z_{j+1} can be defined as follows:

$$Z_{j+1} = F\left(W_j{}^T Z_j + b_j\right), \tag{1}$$

where F is called the activation function.

To make the propagation process more effective, sigmoid is usually applied as F, and it is is given by:

$$F(X) = \frac{1}{1 + e^{-X}}, \tag{2}$$

Y is calculated in the output layer when forward propagation is finished. AE tries to make the output value Y equal to the actual value X. We denoted the above process as g; then, we

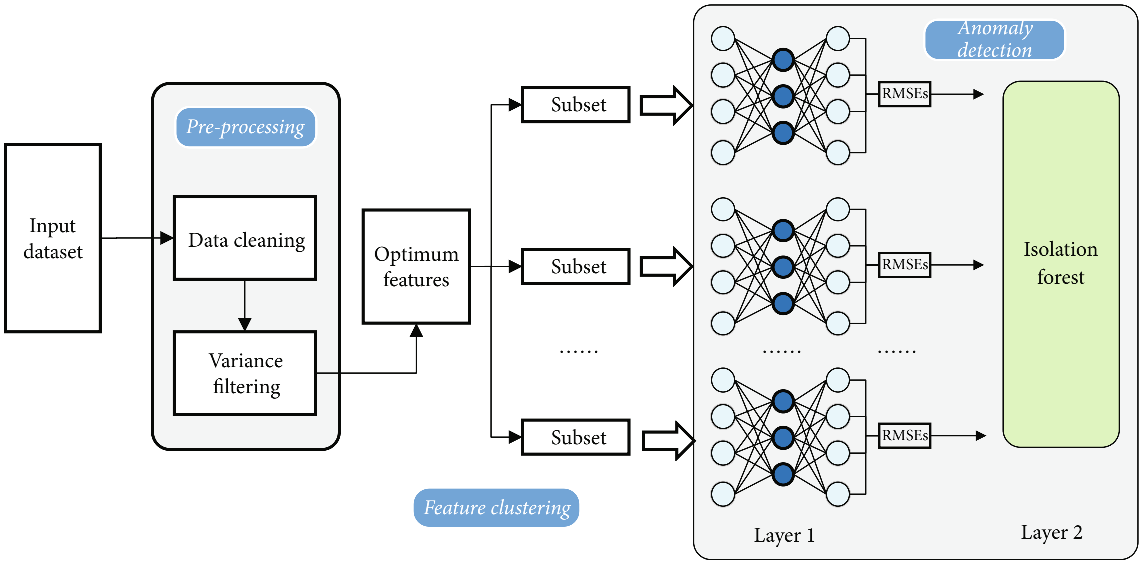

Figure 2: The framework of UTEN-IDS.

have $g_\tau(X) = Y$. After that, the Back Propagation (BP) algorithm is usually used to reduce the losses during reconstruction.

Finally, the pattern of data is learned by AE. If the input sample X is different from the samples that AE has learned, there will be a significant error between X and Y. In this work, the AE is only trained using normal data and learns the concepts of legitimate behaviours in the network. After that, the AE is tested with mixed data containing abnormal cases, leading to a high reconstruction error for the anomaly. Let n denotes the dimensionality of X. We can compute the errors of reconstruction by RMSE:

$$\mathrm{RMSE}(X, Y) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - Y_i)^2}. \quad (3)$$

3.2. Isolation Forest. IF is an unsupervised learning-based anomaly detection algorithm, which focuses on the isolation of few outliers [7]. IF could be seen as the ensemble of isolation trees, which does not require the whole input data for training and captures the character of outliers through the samples of data. These trees work by splitting the data with the randomly selected feature value. The path lengths of anomalies are usually shorter than the normal ones in these trees. Based on the path length of trees, the anomaly score is calculated to identify the outliers. Suppose that the input dataset has N samples, the anomaly score of sample X is defined as follows:

$$\mathrm{Score}(X) = 2^{-E(l(X))/C(N)}, \quad (4)$$

where $l(X)$ denotes the path length in one isolation tree for X, $E(l(X))$ denotes the expected path length of these trees, and $C(N)$ is a constant associated with the dataset. The outliers usually have high scores. IF has been applied in different fields due to its low memory consumption and high detection precision.

4. Methodology

In this work, we have proposed UTEN-IDS to identify cyberattack, especially unknown attacks. We aim to design a lightweight IDS for IoT networks, providing accurate detection performance. The framework of the UTEN-IDS is shown in Figure 2. UTEN-IDS works through 3 main phases: preprocessing, feature clustering, and anomaly detection.

We clean the input data and select the best features during the preprocessing phase. In the feature clustering phase, the features are grouped into several subsets according to the correlation between these features. After that, each sample in the dataset is divided into several sub-samples, which are distributed in different feature subsets. In the anomaly detection phase, the sub-samples in the same feature subset are processed separately by AE. All the AEs are considered an ensemble, and the number of AEs is also the number of subsets. When AE completes the reconstruction of the samples in the subset, we have the collection of RMSEs. We use the IF algorithm to make the final classification based on all the collections.

The current phase serves the next stage. As shown in Figure 3, only normal traffic data is collected for training, so there is no need to collect and label attack instances. The mixed data consisting of both normal and attack instances are used for the testing. In the following subsections, we will describe the proposed method in detail.

4.1. Preprocessing. The network traffic data with null values or infinite values, in many cases, cannot be used as the input of ML algorithms. Therefore, we process the input data in the first phase. We replace the infinite value and null value with zero for input data. Additionally, the features with low variance are removed because the information provided

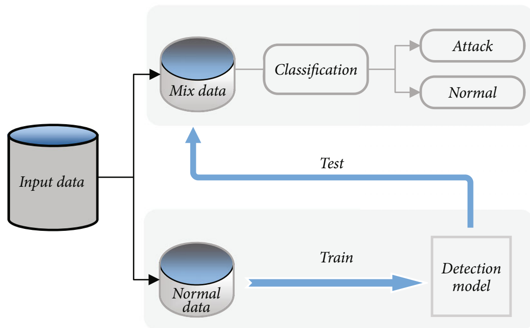

Figure 3: Training and testing processes.

by these features is minimal, and they are not of benefit to the training of AE.

4.2. Feature Clustering. Feature clustering is the process that merges all features with high correlation into the same feature cluster. The purpose of feature clustering used in UTEN-IDS is to train the ensemble of AEs better. By doing so, redundant features are merged, and their impact on final classification is reduced. On the other hand, the features with the strongest correlation should be grouped into the same subset. Different subsets represent different characteristics of input data. An AE is trained with only one subset, which helps the AE learn the data pattern deeply. We will show the performance difference between the ensemble of AEs and the single AE in the experiment.

In this process, we first evaluate the correlation between the input features. The correlation coefficient between the vectors α and β is computed as follows:

$$\mathrm{Corr}(\alpha, \beta) = \frac{\mathrm{Cov}(\alpha, \beta)}{\sigma_\alpha \alpha_\beta}, \tag{5}$$

where σ_α, σ_β, $\mathrm{Cov}(\alpha, \beta)$, and Corr are the standard deviation of α, the standard deviation of β, the covariation of α and β, and the correlation coefficient, respectively.

Secondly, we define the distance between two features f_i and f_j based on the correlation coefficient:

$$\mathrm{Distance}\left(f_i, f_j\right) = 1 - \left|\mathrm{corr}\left(f_i, f_j\right)\right|, \tag{6}$$

where $1 < i, j < T$ and T is the number of features.

We believe that features are either relevant or irrelevant, and both positive and negative correlations show a link between the features. Therefore, the absolute value of the correlation coefficient is used in the equation. By doing so, the distance of features is limited to the range $[0, 1]$. If f_i is close to f_j, the two features have a strong correlation with each other. Then, we have the T-by-T distance matrix A, where A_{ij} denotes the distance between f_i and f_j.

Finally, we use an unsupervised clustering algorithm to cluster the features. Specifically, we use the Mean Shift algorithm [36] to cluster A. Mean Shift is an iterative algorithm for clustering. There are two main reasons for using Mean Shift:

(1) Mean Shift does not require predefined parameters, such as cluster numbers and initial cluster centers.

(2) The performance of Mean Shift is robust with acceptable algorithm complexity.

After that, the features in the same cluster are highly correlated, and the features in different clusters are almost irrelevant. If the input samples are all used for clustering, there will be a significant increase in the computational complexity. Therefore, we use 25% of the training set for clustering. Let $\mathbf{C}$ represent the feature clustering result, a list of clusters $\mathbf{C} = [C_1, C_2, \cdots, C_h]$, where $\sum_{i=1}^{h} |C_i| = T$, h is the number of clusters, and $|C_i|$ is the number of features in C_i. $\mathbf{C}$ is seen as the feature mapping function. According to $\mathbf{C}$, the features of all training samples are grouped into h subsets, which are used for the next phase. These subsets composes the training set. It should be noted that feature clustering is only performed in the training phase. For the testing process, UTEN-IDS maps the input samples to $h$subsets according to $\mathbf{C}$.

4.3. Anomaly Detection. A two-layer unsupervised ensemble model is used for the anomaly detection phase. The ensemble of AEs is implemented in the first layer, and IF is in the second layer.

4.3.1. The Ensemble of Autoencoders. The standard DL models (e.g., MLP) are trained in a supervised manner and consume many computing resources. But AE is trained in an unsupervised fashion, and it does not require labeled samples. AE is also applied to the anomaly detection domain [37]. For the abnormal sample, the reconstruction error calculated by AE is different from the error of normal ones. The IDS can classify the samples correctly with the errors. Therefore, we choose AE as the core of UTEN-IDS and use AE to capture the changes in the network behaviours from the input.

As mentioned before, the AE is used to reconstruct the input sample X. To make the reconstruction more efficient, we apply the following settings to all the AEs:

(1) The number of layers is set to 3. AE with a three-layer structure can reconstruct X well. On the other hand, larger layers will increase the computational overhead and require more time for training.

(2) The input and output layers have t neurons, where t is the number of input features. The hidden layer has $75\% * t$ neurons because too many neurons in this layer may lead to overfitting.

(3) The weights W_{D} in the decoding phase is the matrix transpose of the weights W_{E} in the encoding, where $W_{\mathrm{D}} = {W_{\mathrm{E}}}^T$. It is known as Tied Weights [38]. Only a set of weights is adjusted by the AE, which speeds up the training process and enables AEs to capture more information from the data.

Table 1: Summary of CES-CIC-IDS 2018 dataset used in the experiments.

Subdataset	Size	Distribution
D_1	1048574	Benign: 446772 DoS attacks-SlowHTTPTest: 139890 DoS attacks-Hulk: 461912
D_2	1048575	Benign: 996077 DoS attacks-GoldenEye: 41508 DoS attacks-Slowloris: 10990
D_3	1048575	Benign: 360833 DDoS attack-LOIC-UDP: 1730 DDoS attack-HOIC: 686012
D_4	1048571	Benign: 472380 DDoS attacks-LOIC-HTTP: 576191
D_5	1048522	Benign: 1048009 Brute force-web: 362 Brute force-XSS: 151
D_6	1048575	Benign: 667626 FTP-brute force: 193360 SSH-brute force: 187589

Table 2: Summary of MQTT-IOT-IDS2020 dataset used in the experiments.

Traffic type	Size	Distribution (%)
Benign	165362	85.23
MQTT brute force	14544	7.50
Sparta SSH brute force	14116	7.27

Suppose that there are h feature subsets, they are $S_1, S_2, \cdots, S_h$, where S_i represents the feature subset i and $1 \leq i \leq h$. According to $\mathbf{C}$, sample X is mapped to h subsamples; then, we have $x_1, x_2, \cdots, x_h$, and x_i is in S_i. All the subsamples in S_i are used to train individual AE Ω_i separately, and the number of AEs is h. We take x_i as an example to explain this process.

For Ω_i, the weights are initialized randomly before training. Firstly, the input x_i is 0-1 normalized to get $x_i{}'$. Secondly, based on $x_i{}'$ and the weights, forward propagation using the sigmoid function is completed through the entire network; then, we have y_i in the output layer. Thirdly, the BP algorithm is used to propagate the errors during the process. Based on the errors, we use Stochastic Gradient Descent (SGD) to tune the weights of Ω_i. By doing so, every sample is learned by AE only once and the weights are updated gradually. SGD has the advantage of being able to train a detection model online [29]. After basic training of UTEN-IDS locally, we can deploy it on the network for detections or continue training online. Finally, the RMSE between $x_i{}'$ and y_i is calculated and returned as the output for Ω_i. We repeat the process until all the AEs are trained. After that, the ensemble of AEs is generated.

After training, AE can execute the prediction on unknownsamples. The input sample is also mapped into h subsamples according to $\mathbf{C}$. The sub-samples are used as the input of $\Omega_1, \Omega_2, \cdots, \Omega_h$, respectively. More specifically, the weights of Ω_i are not updated anymore, only forward propagation is performed and RMSE is returned as a prediction score. If one attack instance is processed by AE, we will have R_{attack}, and $R_{\text{attack}} \gg R_{\text{normal}}$, where R denotes the RMSE.

Table 3: The proportion of the dataset.

Division	Distribution
Training	80% normal
Validation	10%normal + 50%attack
Testing	10%normal + 50%attack

Table 4: Comparison results between anomaly detection methods.

Attack type	Metric (%)	LOF	EE	IF
MQTT brute force	Recall	98.93	97.21	99.46
	F1	94.98	98.09	99.44
	AA	97.40	98.39	99.60
Sparta SSH brute force	Recall	100.00	100.00	100.00
	F1	95.43	99.49	99.63
	AA	97.96	99.78	99.84

4.3.2. Detection Using Isolation Forest. The ensemble of AEs complete reconstruction for input subsets; then, we have the RMSEs $[R_1, R_2, \cdots, R_h]$. However, it is insufficient to make accurate decisions using these RMSEs simply.

In [29], the maximum RMSE in the training phase is used as the classification threshold. Let Φ denote the threshold. For a given instance, if the value of reconstruction error is higher than Φ, this instance will be considered an anomaly. Furthermore, the larger values indicate more significant anomalies. The threshold affects the classification performance, and the proper value of the threshold is usually tuned by experiments. It is not easy to find the optimal Φ. In other words, the existing solutions do not have the self-learning ability, and they can not detect attacks adaptively.

Based on the RMSEs, we use the IF algorithm to solve the threshold problem. The IF model brought great performance in the area of anomaly detection. It also works well in situations where the training set does not contain anomalies [7].

One advantage of our method is its self-learning ability. The IF algorithm distinguishes anomalies from normal activities, making UTEN-IDS detect different kinds of attacks in an adaptive manner. Specifically, the trained AEs are used to predict the normal samples, and the obtained RMSEs are used to train the IF. Then, AE and IF process the test set which contains attack samples. The workflow of anomaly detection can be summarized as follows:

Step 1. Split the input training set into two datasets, namely, D_{AE} and D_{IF}.

Step 2. AEs are trained on D_{AE}.

Table 5: Performance comparison between single AE and multiple AEs.

Attack type	Metric (%)	Single AE	Multiple AEs
MQTT brute force	Recall	97.15	99.46
	F1	97.03	99.44
	AA	97.89	99.60
Sparta SSH brute force	Recall	100.00	100.00
	F1	98.10	99.63
	AA	99.17	99.84

Table 6: Performance comparison between IF and UTEN-IDS.

Method	MQTT brute force			Sparta SSH brute force		
	AA (%)	F1 (%)	Recall (%)	AA (%)	F1 (%)	Recall (%)
IF	99.33	98.84	99.44	99.60	99.07	100
UTEN-UDS	99.60	99.44	99.46	99.84	99.63	100

Table 7: A comparison of average accuracy (%) on CES-CIC-IDS 2018 dataset.

Attack type	UTEN-IDS	AE-IDS	KitNET	AE
DoS attacks-SlowHTTPTest	94.77	99.82	94.04	85.97
DoS attacks-Hulk	48.65	63.20	50.15	41.63
DoS attacks-GoldenEye	94.92	49.44	82.95	87.93
DoS attacks-Slowloris	94.73	71.22	82.53	87.88
DDoS attack-LOIC-UDP	91.69	98.55	89.04	86.83
DDoS attack-HOIC	92.50	52.50	88.86	87.71
DDoS attacks-LOIC-HTTP	88.98	54.44	85.04	80.58
Brute force-XSS	66.85	44.91	56.88	55.83
Brute force-web	72.31	58.81	63.00	69.57
SSH-brute force	93.28	57.58	82.74	80.46
FTP-brute force	95.19	79.46	82.79	80.47

Step 3. AEs predict $D_{\mathrm{IF}}$and obtain the collections of RMSEs, R_{IF}.

Step 4. IF is trained on R_{IF}.

Step 5. AEs predict the testing set and obtain the RMSEs, then IF classifies the RMSEs.

D_{IF} is set to 25% of the training dataset, and the samples ofD_{IF} is also used in the feature clustering phase to obtain **C**. The remaining 75% of samples are used to train AEs. The samples of D_{IF} are unknown to AE, and the RMSEs calculated by AE will be close to the ones in the realistic scenario. This allows UTEN-IDS to take full advantage of the training set. If the whole training set or D_{AE} is used as D_{IF}, we find that it takes more time to create the model, but the detection rate is not improved.

Table 8: A comparison of F1 scores (%) on CES-CIC-IDS 2018 dataset.

Attack type	UTEN-IDS	AE-IDS	KitNET	AE
DoS attacks-SlowHTTPTest	96.77	99.89	93.67	91.78
DoS attacks-Hulk	14.65	41.76	0.59	18.76
DoS attacks-GoldenEye	80.74	26.24	54.99	63.33
DoS attacks-Slowloris	52.68	19.87	24.27	31.33
DDoS attack-LOIC-UDP	22.39	62.28	17.95	15.40
DDoS attack-HOIC	99.22	9.51	98.84	98.72
DDoS attacks-LOIC-HTTP	93.02	63.61	95.90	95.41
Brute force-XSS	0.72	0.10	0.22	0.20
Brute force-web	2.11	0.48	0.67	0.78
SSH-brute force	94.75	76.75	89.06	87.79
FTP-brute force	96.79	87.58	89.38	88.12

If the RMSE of one instance is considered an outlier by IF, this instance will be classified as an attack. IF suffers from a "curse of dimensionality" [7], but the critical low-dimensional features (RMSEs) will help IF achieve excellent detection performance.

5. Experiments and Analysis

5.1. Dataset. The latest intrusion detection dataset represents the modern malicious behaviours in the current network. For this reason, the CSE-CIC-IDS 2018 dataset and the MQTT-IOT-IDS2020 dataset are selected to demonstrate the effectiveness of the proposed method.

The CSE-CIC-IDS 2018 dataset was published by the Canadian Institute for Cybersecurity (CIC) in 2018, which collected a variety of modern attack behaviours. This dataset includes the experimental machine's network traffic and system logs, along with more than 80 features extracted from the source pcap files by CICFlowMeter, a network traffic flow generator and analyzer [39]. The dataset consists of different attack scenarios, and these scenarios are stored in subdatasets. However, not all the attack samples in the dataset are suitable for testing models. Some attacks like "SQL injection" and "Infiltration" are insufficient to detect with network traffic [30]. In this work, we pay close attention to common attacks, such as DoS, DDoS, and brute force attack. Therefore, as shown in Table 1, 6 subdatasets are used in our experiments, involving eleven types of attacks. To test the performance of UTEN-IDS properly, the features such as source and destination IP are removed.

In this paper, we use the MQTT-IOT-IDS2020 dataset to test the performance of UTEN-IDS in the IoT network. The Message Queuing Telemetry Transport (MQTT) protocol is one of the most standard communication protocols used in IoT. The dataset consists of several kinds of IoT traffic, which is generated under the simulated environment of the MQTT-based IoT network. For this dataset, we are still concerned about the attacks such as brute force. We extract two kinds of brute force attacks from the original bi-directional

Table 9: Performance (%) of the four intrusion detection methods on the IoT dataset.

Attack type	Metric	UTEN-IDS	AE-IDS	KitNET	AE
MQTT brute force	Recall (%)	99.45	100	100	100
	F1 (%)	99.48	99.07	85.14	67.24
	AA (%)	99.62	99.59	92.33	78.57
Sparta SSH brute force	Recall (%)	100	99.31	100	100
	F1 (%)	99.73	99.65	84.79	66.61
	AA (%)	99.89	99.65	92.34	78.60

Table 10: A comparison of running time (S).

Attack type	UTEN-IDS	AE-IDS	KitNET	AE
MQTT brute force	46.96	98.06	82.59	17.99
Sparta SSH brute force	47.25	73.81	83.41	18.07

flow-based dataset. The data record the characteristics of the flows in the IoT. The final dataset used in our experiment is summarized in Table 2.

5.2. Evaluation Metrics. To evaluate the performance of UTEN-IDS, we use the following evaluation metrics: recall, F1 score, and Average Accuracy (AA). Recall reveals the detection rate, and the F1 score comprehensively evaluates the detection ability. It is the harmonic mean of the precision and recall, which can be formulated as:

$$\text{F1 score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{7}$$

The average accuracy evaluates the generalization ability of the classifier. In the binary classification task, it is the mean value of specificity and recall. F1 score, recall, and AA are all positively correlated with the detection performance.

5.3. Experimental Setup. We implement the UTEN-IDS in Python. The experiments are conducted on the machine with Linux operating system, 32 GB RAM, Tesla V100 GPU, and Xeon GOLD 6148 CPU.

UTEN-IDS is only trained with normal traffic data in specific network scenarios. In Table 3, for both the subdatasets of CES-CIC-IDS 2018 and the IoT dataset, 80% of normal samples are used in the training process, 10% of normal samples and 50% of attack samples are used as the test set, and the remaining samples are for validation. The normal samples are far more than the attack samples, and AEs need many samples for the training process. Therefore, 80% of the normal data is selected for training. The proportion of each class in the test and validation set is the same. To evaluate the performance of UTEN-IDS, different types of attacks are combined with normal data as the test set, respectively.

5.4. Comparative Experiments. "Contamination" is a key parameter for IF, which means the amount of contamination of the dataset. This parameter determines the cutoff value of anomaly score for IF. To obtain a robust model, we choose a range of "contamination" values and perform the comparison on the validation dataset. Through experiment, we observe that the values of 0.1 and 0.02 offer optimum performance for CES-CIC-IDS 2018 and MQTT-IOT-IDS2020, respectively.

Our method completes the detection through the anomaly detection algorithm, and the selection of anomaly detection algorithms affects the final classification result. Only those state-of-the-art algorithms in anomaly detection are worth considering. For this reason, we select Elliptic Envelope (EE) [40] and Local Outlier Factor (LOF) [41] as competitors of IF. They are all robust and explicable anomaly detection algorithms. Table 4 shows that the UTEN-IDS using IF achieves the best performance on all the metrics on the validation set of the IoT dataset.

Table 5 shows the effect of feature clustering on the final performance. We conduct a comparison experiment based on the two schemes: single AE and multiple AEs. Single AE represents only one AE used in UTEN-IDS, which skips the feature clustering phase; multiple AEs represent the UTEN-IDS with the ensemble of AEs. It can be inferred that multiple AEs outperform single AE from Table 5.

UTEN-IDS solves the issue of intrusion detection by using the IF, and IF is trained based on the RMSEs. To prove that the AE helps improve the performance of the anomaly detection algorithm, we compare the performance of UTEN-IDS with the IF. Here, IF is trained based on the raw data. In Table 6, the obtained results of UTEN-IDS are significantly better than IF model, which proves that IF with the RMSE attains a higher performance compared to IF with the raw data.

The above experiments explain the necessity of some steps or hyperparameters in our method. We conduct comparisons with several detection methods in the following subsections.

5.4.1. Performance Analysis on CES-CIC-IDS 2018 Dataset. In the selection of competitors, we consider the intrusion detection methods belonging to the state-of-the-art. Therefore, we select AE-IDS, KitNET [29], and AE. They are all advanced detection methods. Here, AE is an unsupervised neural network, which network structure is the same as UTEN-IDS. We use one statistical approach used in [42] to set the threshold of AE reconstruction loss, and the threshold is used for classification. As mentioned earlier, Kitsune is an online method and KitNET is the core detection algorithm of Kitsune. The anomaly threshold is a

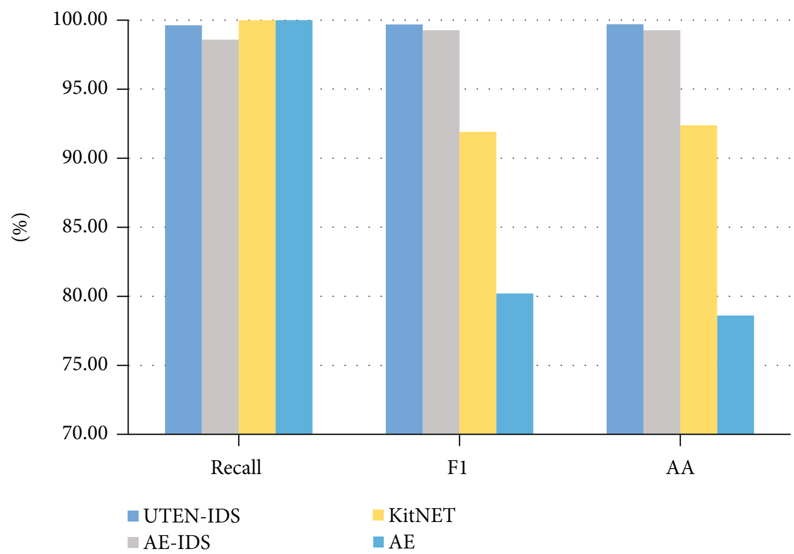

Figure 4: A comparison (%) of binary classification.

crucial parameter for this method. We tune the parameter value by experiments.

To prove the advantages of our proposed method against new attacks, we compare the performances of UTEN-IDS with the competitors on the CES-CIC-IDS 2018 dataset. The four approaches use the same samples fortraining. Then, the performance is measured by the testing set.

Table 7 shows the performance comparison of the four methods on the different attack classes of the CES-CIC-IDS 2018 dataset. For DoS attacks-GoldenEye, SSH-brute force, and DoS attacks-HOIC, the accuracy values of UTEN-IDS are significantly higher than that of others.

Table 8 shows the comparison results of the F1 score on the dataset. F1 scores take into account both the detection rate and classification performance. For most subdatasets, the performance of UTEN-IDS is better than that of the other methods.

In general, UTEN-IDS achieves the best results on most subdatasets. The result shows that UTEN-IDS has a strong generalization and detection ability against DoS, DDoS, and brute force attacks. However, the proposed method does not perform well in detecting DoS attacks-Hulk, DoS attacks-SlowHTTPTest, and DDoS attack-LOIC-UDP.

For SlowHTTPTest attacks and LOIC-UDP attacks, we notice that UTEN-IDS outperforms KitNET and AE, but it is not as good as AE-IDS. One reason is the redundant features of the subdatasets. We observe that AE-IDS uses the RF algorithm to select less than 20 features and achieves the best records, which indicates that many features can be removed. On the contrary, UTEN-IDS uses more than 60 features, including many noisy features. These features have a negative effect on the result. On the other hand, we find that the reconstruction errors of the two types of attacks are both great through the validation set. We take 0.001 as the "contamination" value of IF and implement the experiment again. For both attacks, the accuracy of UTEN-IDS reached 99.98% and obtain a better performance.

In detecting "DoS attacks-Hulk," the F1 score of our method does not reach 20%. The reason for this is that there is some similarity between the features of the subdataset. We calculate the mean RMSE values of normal samples and the attack samples, respectively. We find that their difference was less than 0.01, which shows that these two kinds of samples are similar. Therefore, the IF algorithm cannot make accurate decisions based on the RMSEs, which leads to poor performance.

5.4.2. Performance Analysis on the IoT Dataset. To evaluate the performance of our method in the IoT environment, we compare UTEN-IDS with other detection methods using the extracted samples of MQTT-IOT-IDS2020.

Table 9 shows the detailed performance comparison of the four methods on the test set. The recall of UTEN-IDS (99.45%) is lower than the best record (100%) in detecting MQTT brute force attacks. However, UTEN-IDS achieves the best accuracy and F1 score records of each class. The results indicate that UTEN-IDS cope effectively with the brute force attacks in the IoT network.

To further interpret the effectiveness of UTEN-IDS, we conduct running time comparisons on the four detection methods. Table 10 shows the total running time of these approaches based on the extracted samples of MQTT-IOT-IDS2020. It can be seen that the time cost of UTEN-IDS is lower than that of AE-IDS and KitNET. Although AE takes the shortest time, the classification performance is not as good as UTEN-IDS.

Figure 4 shows the results for binary classification performance based on the IoT dataset, which are measured by recall, F1 score, and AA. Both kinds of brute force attack samples are used to test the performance of different detection methods. We notice that both the F1 score and accuracy of UTEN-IDS are higher than other methods. The comparison results suggest that the proposed method is superior to other intrusion detection methods.

The results of the above experiments have demonstrated the superiority of UTEN-IDS. The proposed method has not only a robust detection performance but also a low time complexity.

6. Conclusion

The threats of IoT intrusion are increasing day by day. In our opinion, the solution is IDS. In this work, we proposed UTEN-IDS, a lightweight IDS based on unsupervised techniques, to tackle IoT security threats. It was divided into preprocessing, feature clustering, and anomaly detection phases. A variance filtering method was used to select features in the preprocessing stage. The feature clustering phase was used to obtain the feature subsets. The anomaly detection module of the proposed method was a two-level ensemble model. AEs were used in the first level, and IF was in the second level. The AEs reconstructed the input feature subset and calculated the RMSEs. The IF performed the classification based on all the RMSEs.

Two public datasets, CES-CIC-IDS 2018 and MQTT-IOT-IDS2020, were used to verify the performance of the proposed method. The results showed that our method is superior to the other methods. However, our approach did not perform well in detecting some attack types, such as "DoS attacks-Hulk." In our future studies, we plan to optimize our method to improve the detection rate of attacks that are hard to classify.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China (grant number 61906099) and the Open Fund of Key Laboratory of Urban Land Resources Monitoring and Simulation, Ministry of Natural Resources (grant number KF-2019-04-065).

References

[1] D. K. Sharma, T. Dhankhar, G. Agrawal et al., "Anomaly detection framework to prevent DDoS attack in fog empowered IoT networks," *Ad Hoc Networks*, vol. 121, article 102603, 2021.

[2] S. Fadilpašić, "IoT Boom Raises Important Security Questions," 2021, https://www.itproportal.com/news/iot-boom-raises-important-security-questions/.

[3] A. Wang, W. Chang, S. Chen, and A. Mohaisen, "Delving into internet DDoS attacks by botnets: characterization and analysis," *IEEE/ACM Transactions on Networking*, vol. 26, no. 6, pp. 2843–2855, 2018.

[4] H. Liu and B. Lang, "Machine learning and deep learning methods for intrusion detection systems: a survey," *Applied Sciences*, vol. 9, no. 20, p. 4396, 2019.

[5] M. Liu, C. Liu, M. Li, Y. Chen, S. Zheng, and N. Zhao, "Intelligent passive detection of aerial target in space-air-ground integrated networks," *China Communications*, vol. 19, no. 1, pp. 52–63, 2022.

[6] A. J. Siddiqui and A. Boukerche, "Adaptive ensembles of autoencoders for unsupervised IoT network intrusion detection," *Computing*, vol. 103, no. 6, pp. 1209–1232, 2021.

[7] F. T. Liu, K. M. Ting, and Z. H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining*, pp. 413–422, Pisa, Italy, December 2008.

[8] D. Jin, Y. Lu, J. Qin, Z. Cheng, and Z. Mao, "SwiftIDS: realtime intrusion detection system based on LightGBM and parallel intrusion detection mechanism," *Computers & Security*, vol. 97, article 101984, 2020.

[9] UNB CSE-CIC-IDS2018 on AWS dataset, Canadian Institute for Cybersecurity (CIC), Univ. New Brunswick, 2018, https://www.unb.ca/cic/datasets/ids-2018.html.

[10] H. Hindy, E. Bayne, M. Bures, R. Atkinson, C. Tachtatzis, and X. Bellekens, "Machine learning based IoT intrusion detection system: an MQTT case study," 2020, https://arxiv.org/abs/2006.15340.

[11] A. Khraisat, I. Gondal, and P. Vamplew, "An anomaly intrusion detection system using C5 decision tree classifier," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 149–155, Springer, Cham, 2018.

[12] E. Min, J. Long, Q. Liu, J. Cui, and W. Chen, "TR-IDS: anomaly-based intrusion detection through text-convolutional neural network and random forest," *Networks*, vol. 2018, article 4943509, 9 pages, 2018.

[13] E. Mugabo and Q. Y. Zhang, "Intrusion detection method based on support vector machine and information gain for mobile cloud computing," *IJ Network Security*, vol. 22, no. 2, pp. 231–241, 2020.

[14] S. Sarvari, N. F. M. Sani, Z. M. Hanapi, and M. T. Abdullah, "An efficient anomaly intrusion detection method with feature selection and evolutionary neural network," *IEEE Access*, vol. 8, pp. 70651–70663, 2020.

[15] B. Ingre, A. Yadav, and A. K. Soni, "Decision tree based intrusion detection system for NSL-KDD dataset," in *International Conference on Information and Communication Technology for Intelligent Systems (ICTIS 2017)*, Springer, Cham, 2017.

[16] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE symposium on computational intelligence for security and defense applications*, pp. 1–6, Ottawa, Canada, 2009.

[17] H. Zhang, L. Huang, C. Q. Wu, and Z. Li, "An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset," *Computer Networks*, vol. 177, article 107315, 2020.

[18] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 military communications and information systems conference (MilCIS)*, pp. 1–6, Canberra, Australia, 2015.

[19] M. Wang, Y. Lu, and J. Qin, "A dynamic MLP-based DDoS attack detection method using feature selection and feedback," *Computers & Security*, vol. 88, article 101645, 2020.

[20] A. A. Aburomman and M. B. I. Reaz, "A survey of intrusion detection systems based on ensemble and hybrid classifiers," *Computers & Security*, vol. 65, pp. 135–152, 2017.

[21] J. Gu, L. Wang, H. Wang, and S. Wang, "A novel approach to intrusion detection using SVM ensemble with feature augmentation," *Computers & Security*, vol. 86, pp. 53–62, 2019.

[22] H. Wang, J. Gu, and S. Wang, "An effective intrusion detection framework based on SVM with feature augmentation," *Knowledge-Based Systems*, vol. 136, pp. 130–139, 2017.

[23] M. H. Haghighat and J. Li, "Intrusion detection system using voting-based neural network," *Tsinghua Science and Technology*, vol. 26, no. 4, pp. 484–495, 2021.

[24] M. Prasad, S. Tripathi, and K. Dahal, "Unsupervised feature selection and cluster center initialization based arbitrary shaped clusters for intrusion detection," *Computers & Security*, vol. 99, article 102062, 2020.

[25] M. A. Ambusaidi, X. He, and P. Nanda, "Unsupervised feature selection method for intrusion detection system," in *2015 IEEE Trustcom/BigDataSE/ISPA*, vol. 1, pp. 295–301, Helsinki, Finland, 2015.

[26] R. S. M. Carrasco and M. A. Sicilia, "Unsupervised intrusion detection through skip-gram models of network behavior," *Computers & Security*, vol. 78, pp. 187–197, 2018.

[27] A. Bohara, U. Thakore, and W. H. Sanders, "Intrusion detection in enterprise systems by combining and clustering diverse monitor data," in *Proceedings of the Symposium and Bootcamp on the Science of Security*, pp. 7–16, Pittsburgh, Pennsylvania, 2016.

[28] Z. Mingqiang, H. Hui, and W. Qian, "A graph-based clustering algorithm for anomaly intrusion detection," in *2012 7th International Conference on Computer Science & Education (ICCSE)*, pp. 1311–1314, Melbourne, Australia, 2012.

[29] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," *Machine Learning*, vol. 5, p. 2, 2018.

[30] X. K. Li, W. Chen, Q. Zhang, and L. Wu, "Building autoencoder intrusion detection system based on random forest feature selection," *Computers & Security*, vol. 95, article 101851, 2020.

[31] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[32] S. Zavrak and M. İskefiyeli, "Anomaly-based intrusion detection from network flow features using variational autoencoder," *IEEE Access*, vol. 8, pp. 108346–108358, 2020.

[33] K. Sadaf and J. Sultana, "Intrusion detection based on autoencoder and isolation forest in fog computing," *IEEE Access*, vol. 8, pp. 167059–167068, 2020.

[34] N. Lannan and G. Fan, "Filter guided manifold optimization in the autoencoder latent space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, Long Beach, CA, USA, 2019.

[35] R. McConville, R. Santos-Rodriguez, R. J. Piechocki, and I. Craddock, "N2d:(not too) deep clustering via clustering the local manifold of an autoencoded embedding," 2019, https://arxiv.org/abs/1908.05968.

[36] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[37] H. Choi, M. Kim, G. Lee, and W. Kim, "Unsupervised learning approach for network intrusion detection system using autoencoders," *The Journal of Supercomputing*, vol. 75, no. 9, pp. 5597–5621, 2019.

[38] O. E. David and I. Greental, "Genetic algorithms for evolving deep neural networks," in *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pp. 1451-1452, Vancouver, BC, Canada, 2014.

[39] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related," in *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*, pp. 407–414, Rome, Italy, 2016.

[40] P. J. Rousseeuw, "Least median of squares regression," *Journal of the American Statistical Association*, vol. 79, no. 388, pp. 871–880, 1984.

[41] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander, "LOF," *ACM SIGMOD Record*, vol. 29, no. 2, pp. 93–104, 2000.

[42] G. Dlamini, R. Galieva, and M. Fahim, "A lightweight deep autoencoder-based approach for unsupervised anomaly detection," in *2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA)*, pp. 1–5, Abu Dhabi, United Arab Emirates, November 2019.