Learning Latent Representation with Limited Labels for IoT Anomaly Detection

Nguyen Huu Noi, Tran Nguyen Ngoc

***Abstract*— Malware detection is a critical challenge in the current era, especially for IoT devices. Previous studies have applied analytic techniques to reduce data size and extract valuable information. However, most of these studies count on a considerable quantity of outliers to perform anomaly detection. In this paper, we propose an enhanced method (named FeaWAD*) that improves the data encoding strategy based on the FeaWAD network** [1]**. These models require only a small fraction of anomalies for training. We evaluate the FeaWAD* method on the N-BaIoT dataset with various test scenarios for detecting known attacks as well as unknown future attacks. The experimental results demonstrate that the FeaWAD* method outperforms the original model FeaWAD and other popular anomaly detection methods such as Isolation Forest, Local Outlier Factor, and One-class Support Vector Machine. We also measure the time-based effectiveness of the model to assess its practical applicability.**

***Tóm tắt*— Phát hiện mã độc là một thách thức lớn trong thời đại hiện nay, đặc biệt là đối với các thiết bị IoT. Các nghiên cứu trước đây của nhóm tác giả đã áp dụng các kỹ thuật phân tích để giảm kích thước dữ liệu và trích xuất thông tin có giá trị. Tuy nhiên, hầu hết các nghiên cứu đó dựa vào một lượng lớn các điểm ngoại lai để thực hiện phát hiện bất thường. Trong bài báo này, nhóm tác giả đề xuất một phương pháp cải tiến dựa trên mạng FeaWAD (FeaWAD*) để cải thiện phương pháp biểu diễn dữ liệu. Mô hình này chỉ yêu cầu một phần nhỏ các bất thường để huấn luyện. Nhóm tác giả đánh giá phương pháp FeaWAD trên tập dữ liệu N-BaIoT với nhiều kịch bản kiểm tra khác nhau để phát hiện các cuộc tấn công đã biết cũng như các cuộc tấn công chưa biết. Kết quả thực nghiệm cho thấy phương pháp FeaWAD* cho kết quả tốt hơn mô hình gốc FeaWAD và các phương pháp phát hiện bất thường phổ biến khác như Isolation Forest, Local Outlier Factor và One-class Support Vector Machine. Đồng thời, nhóm tác giả cũng đánh giá hiệu quả của mô hình dựa trên thời gian để đánh giá khả năng áp dụng vào thực tế.**

***Keywords*— AutoEncoder; semi-supervised; latent representation; IoT; malware detection.**

***Từ khóa*— Bộ tự mã hóa; học bán giám sát; biểu diễn ẩn; IoT; phát hiện mã độc.**

I. INTRODUCTION

The task of anomaly detection is to spot data points that are different from the normal patterns in a specific domain. This task has many applications in fields such as networking, machine learning, and computer vision. Researchers have developed various algorithms for anomaly detection and tested them on public datasets. A common challenge in this task is the lack of labeled anomaly samples, which are often rare and difficult to obtain, while normal data is abundant and easily accessible [2].

Machine learning (ML), especially deep learning (DL), is a widely used technique in various security domains, such as classifying and detecting malicious codes [3], identifying network attacks [4], and forecasting anomalous behavior of network data [1,5,6]. These techniques can be classified into supervised learning [5], unsupervised learning [7,8], and semi-supervised [9,10] learning methods. Supervised learning methods use labeled data (both normal and abnormal) for training. The algorithms can operate on two or more classes. Unsupervised learning methods usually use only normal data for training, assuming that only benign data is collected in practice. If the collected data is different from the training data, it will be classified as abnormal data. Semi-supervised methods are used in cases where it is assumed that there is a certain amount of abnormal data included along with normal data

This manuscript is received on September 21, 2023. It is commented on September 26, 2023 and is accepted on November 10, 2023 by the first reviewer. It is commented on September 25, 2023 and is accepted on November 27, 2023 by the second reviewer.

for training. The percentage of abnormal data is typically much lower than that of normal data.

One of the outstanding and good results in anomaly detection based on semi-supervised learning is FeaWAD [1]. The authors have proposed a method to extract features for input data as well as identify anomalies through point identification. In this paper, we propose an improved model for the FeaWAD model for IoT anomaly detection. More specifically, we propose to improve the feature extraction method that aggregates data from the input, thereby improving the efficiency of the anomaly detection model, applied to IoT network anomaly detection.

Our main contributions are presented as follows:

- We propose an improved method of FeaWAD, named FeaWAD* to use for IoT network anomaly detection. It is a feature weighting mechanism that can characterize the data and enhance the discriminative power of the features.

- We perform experiments on various IoT datasets to assess the effectiveness of the proposed model. The obtained results show that the improved algorithm FeaWAD* gives better results than the original algorithm as well as other anomaly detection algorithms.

II. RELATED WORKS

This section reviews recent works on detecting anomalies, focusing on unsupervised and semi-supervised learning methods, particularly autoencoder-based algorithms. We also discuss studies that aim to handle classification tasks using semi-supervised techniques.

Currently, the research results in deep representation learning [1,6,11,12] have shown that deep neural models can effectively extract features for clustering and anomaly detection tasks. The essential factor for outlier detection is to use self-supervised techniques to learn representations. Linear models and probabilistic models are among the main techniques used, each based on specific assumptions.

Unsupervised anomaly methods for anomaly detection have been widely investigated because they can capture complex inner relations by learning features from unlabeled data [7,13,15]. Pei et al. [16] introduced a method called Donut that used a Variational AutoEncoder (VAE) for detecting anomalies in time series data. Donut performed much better than VAE-based and other supervised methods, thanks to the effective representation of the deep generative model. An unsupervised anomaly detection framework that integrated the anomaly detection task and the representation learning process was introduced by Pang et al [14]. They learned effective low-dimensional representations for the target detector from ultrahigh-dimensional data through this integration. Based on Self-organizing Maps, Nguyen et al [8] introduced a hybrid model of AE and SOM to detect IoT malware. The original model AESOM and its improved version DAESOM work well with original IoT data, mapping the inputs to groups and detecting anomalies.

In a semi-supervised manner, some recent works have explored anomaly detection with extremely unbalanced data, using the scarce but useful anomalous samples [9,10]. A feedback-guided anomaly detection framework was proposed by Siddiqui et al [17]. The framework could improve the unsupervised anomaly detection model by using the analyst's prior knowledge to assign higher anomaly scores to instances with higher chances of being abnormal. Ruff et al [10] introduced a deep semi-supervised method called Deep SAD, which extended a recent unsupervised anomaly detection method, namely Deep SVDD [18], to utilize labeled anomalies. Deep SAD introduced a novel loss to pull the normal data towards a fixed centroid and push the anomalies away. In [9], Pang et al. propose an anomaly detection model with a scenario for limited labeled anomaly data and unlabeled data. The model transforms the anomaly detection task into an ordinal regression problem of pairwise relationships, using the full number of labeled anomaly samples to form sample pairs for the next anomaly detection process.

The authors propose an AE-based method to extract three components including hidden representation, reconstruction error, and reconstructed residual vector to characterize each input data. These features are then used to generate the anomaly generator, which is used to classify the data as normal or out of the ordinary [1]. Our research is based on this research idea, with improvements in specific components to improve the efficiency of anomaly detection for IoT networks.

III. Feature encoding with autoencoder

A. The FeaWAD algorithm

In this section, we briefly describe the contributions of the research [1]. The paper proposes a feature extraction learning process and a method to generate a score for anomaly determination. The architecture overview of the study is shown in Figure 1. We refer to this algorithm as FeaWAD.

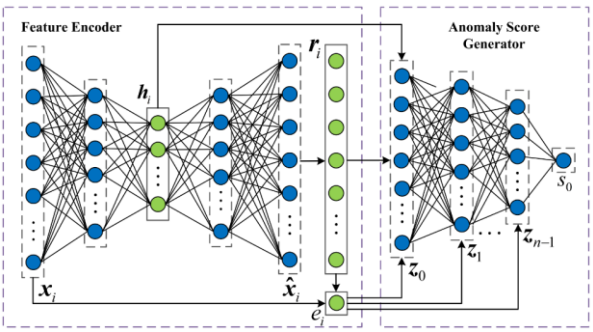

Figure 1. FeaWAD Anomaly Detection Model [1]

First, from the Autoencoder network, the model will separate three typical components representing the input data, including hidden representation h, reconstruction error e, and residual vector r.

The calculation formulas are respectively:

$$h = f_e(x_i; W_e) \quad (1)$$

$$e = |\hat{x}_i - x_i|_2 \quad (2)$$

$$r = \frac{\hat{x}_i - x_i}{|\hat{x}_i - x_i|_2} \quad (3)$$

In Eq. (*1*(1), W_e is the encoder's parameters, and $f(\cdot, W_e)$ is the mapping formulation for the encoding process. In Eq. (2) $\hat{x}_i$ is the reconstructed value of x_i through the decoding process and calculated by $\hat{x}_i = f_d(h; W_d)$ with W_d is the decoder's parameters, $f(\cdot, W_d)$ is the decoding formulation process, and $|\cdot|$ is the Euclidean norm. Then FeaWAD proposed an anomaly score generator (ASG). The network used in ASG is MLP. For each layer z_k is calculated by the formula:

$$z_k = f(W_m^k z_{k-1} + b_k + w_e^k e) \quad (4)$$

In Eq. (4), W_m^k is weight, b_k is the bias parameters for the k-layer.

At the first layer $(k-1)$, z_{k-1} is assigned by $[r, h]$, while the k-layer z_k determines the score s_0 for anomaly decision.

The FeaWAD network is composed of two components: a feature encoding network (FEN) $\psi(\cdot, \Theta_e)$ and anomaly score generator (ASG) $\varphi(\cdot, \Theta_g)$. The Θ_e and Θ_g are parameters of models FEN and ASG, respectively. The entire network is represented as $\phi(x_i; \Theta_e; \Theta_g)$.

The following formula shows how the loss function of the FeaWAD is calculated:

$$L(\Theta_e, \Theta_g) = L_d(\Theta_e, \Theta_g) + \lambda L_e(\Theta_e) \quad (5)$$

where $L_e = \sum_i (1 - y_i) e_i + y_i \max(0, a_0 - e_i)$ and $L_d = \sum_i (1 - y_i) |\phi(x_i)| + y_i \max(0, a_0 - \phi(x_i))$ and λ is the hyperparameter to balance the two components of the total loss function.

B. An improved version of FeaWAD

In this research, we propose an improved version of FeaWAD, which we call FeaWAD*. Our improvement goal focuses on the process of forming the values of h, e, and r.

First, we use the absolute value $|\hat{x}_i - x_i|$ instead of $(\hat{x}_i - x_i)$ in the r calculation formula to make sure all values are non-negative. Next, since the loss function is quite large, we multiply a hyperparameter, called σ, to the value of r (we set the value of σ to 10e-5 in experiments).

Then the formula of r is rewritten as follows:

$$r = \frac{\sigma \cdot |\widehat{x_i} - x_i|}{|\widehat{x_i} - x_i|_2} \quad (6)$$

Using the absolute value of $|\widehat{x_i} - x_i|$ in the reconstruction error in an autoencoder with ReLU activation function preserves more negative data after transforming to r, which improves the learning performance of our model. For the values h and e, we keep the same as in the original paper, with h being the hidden representation and e being the reconstruction error.

The use of ASG and Objective Function is kept as the original idea. The training process of the algorithm is also done using gradient descent and is described in more detail in the next subsection.

C. Traning Process

The network is trained using the gradient descent algorithm. To balance the normal and abnormal samples, a mini-batch is formed by sampling the same number of normal and anomalous samples. This means that the anomalous samples are taken more frequently.

The model is trained using a two-stage training strategy. The feature generator network is first trained using only reconstruction loss in the pre-training stage:

$$L_{ae}(\Theta_e) = \sum_i e_i \quad (7)$$

In this phase, the FEN $\psi(\cdot;\Theta_e)$ is pre-trained on all the unlabeled samples in the training dataset to learn a basic encoding strategy. The trained model is then later used for training the whole network. The loss function is the root mean squared error. Next, the entire network $\phi(\cdot;\Theta_e;\Theta_g)$ is trained with the FEN parameters initialized from the previous pre-training. In this phase, we use all the unlabeled data and some labeled anomalies to fine-tune the entire trained model based on the feature encoding strategy learned earlier.

IV. EXPERIMENTS

In this section, we present the description of datasets, experimental settings, and evaluation metrics for FeaWAD* model in detecting IoT attacks. The IoT dataset with nine device types was used, as described in the subsection below.

A. Datasets

The dataset N-BaIoT (Table 1) was first presented by Y. Meidan et al. [19]. The dataset has data samples from nine different IoT devices. These devices belong to four categories: doorbell, thermostat, monitor, and camera/webcam. Each sample in the dataset has 115 features.

TABLE 1. THE DESCRIPTION OF NBAIOT DATASET

ID	Device Name	Benign	Gafgyt	Mirai
D1	Danmini Doorbell	49548	652100	316650
D2	Ecobee Thermostat	13113	512133	310630
D3	Ennio Doorbell	39100	316400	
D4	Philips B120N10 Baby Monitor	175240	312273	610714
D5	Provision PT 737E Camera	62154	330096	436010
D6	Provision PT 838 Camera	98514	309040	429337
D7	Samsung SNH 1011 N Webcam	52150	323072	
D8	SimpleHome XCS7 1002 WHT Camera	46585	303223	513248
D9	SimpleHome XCS7 1003 WHT Camera	19528	316438	514860

The experiments consisted of two main scenarios, detecting known attacks on the trained dataset and detecting unknown attacks on unseen data. In the first scenario, we evaluated the performance of our model on both Gafgyt and Mirai attacks across all devices. In the second scenario, we excluded devices D3-D7 from the evaluation because they lacked data on Mirai attacks.

B. Parameters setting

First, we set the AE in FeaWAD and FeaWAD* with two layers as in the original research with the sizes of hidden layers being {500,100}, respectively. The activation function used is ReLU. The margin a_0 is set to 5, the hyperparameter λ is set to 1, σ is set to 10e-5. The number of epochs is 100, the batch size is 64 and the learning rate is 1e-3.

Through training and testing, the original dataset is split into two parts: one for training

and one for testing with 80% samples used for training, 20% samples used for testing.

C. Evaluation metrics

In experiments, we use Area Under the Curve (AUC), $F1$-score, FAR and MDR to assess the effectiveness of the FeaWAD* and related models.

1) AUC: First, we calculate the True Positive Rate (TPR) and False Positive Rate (FPR) using the following formulas.

$$TPR = \frac{TP}{TP + FN};\ FPR = \frac{FP}{FP + TN}$$

where TP: True Positives, TN: True Negatives, FP: False Positives and FN: False Negatives respectively.

The Receiver Operating Characteristic curve shows the TPR versus FPR for different thresholds. The AUC is the entire area below the ROC curve. AUC measures the performance for all possible thresholds.

2) $F1$-score: The $F1$-score is a metric of a test's accuracy that is used for binary classification problems. The $F1$-score can be calculated as:

$$F_1 = \frac{2}{recall^{-1} + precision^{-1}}$$

The $F1$-score is a harmonic mean of the precision and recall, where it reaches the optimal value of 1 and the worst value of 0. The $F1$-score gives equal weight to both precision and recall.

3) FAR and MDR

The two metrics are False Alarm Rate (FAR) and Miss Detection Rate (MDR) are also used to evaluate the effectiveness of the proposed model.

$$FAR = \frac{FP}{FP + TN};\ MDR = \frac{FN}{FN + TP}$$

FAR measures the proportion of negative samples that are incorrectly classified as positive. MDR measures the proportion of positive samples that are incorrectly classified as negative.

IV. Results and discussion

In this section, we present and discuss the results of the experiment process. We collected the results from the experiments as explained in section IV. Experiments

Experiments were implemented in Python using the Keras, Scikit-learn, and Tensorflow frameworks. We ran the experiments on a computer with Ubuntu 22.04 LTS, an Intel® Core i5 11400H CPU, 24 GB of RAM and a Geforce 3050 GPU.

A. Anomaly detection

We report the experimental results for our proposed model FeaWAD* and four other competing algorithms (Isolation Forest – IF, Local Outlier Factor – LOF, One-class SVM – OCSVM, and original FeaWAD). The evaluation results are based on two criterias, AUC and $F1$-score, as described in Section IV.C.

In this scenario, the experiments were conducted as follows: we train and test on the same type of attack. table 2 presents the test results for nine datasets. The ratios of outlier (attack) data to total data are described in the column *Outlier Perc*. The experiments were performed with different ratios, the smallest being 3.85% and the largest being 18.03%. The diverse data ratio helps us to have accurate assessments of the test results.

The comparison of the proposed method with the four other anomaly detection methods is presented in table 2. As shown, compared to the methods IF, LOF and OCSVM the proposed method performs better than them on all datasets in terms of both AUC and $F1$-score.

Compared to the FeaWAD algorithm, the proposed method achieved the best results on three datasets based on AUC and five datasets based on $F1$-score metric. Particularly on the datasets D2, D3, D5, D6, D7, D9, FeaWAD* outperformed FeaWAD. With the datasets D1 and D8, FeaWAD performed better. Only with the dataset D4, FeaWAD and FeaWAD* had similar results.

In general, the proposed method FeaWAD* gives the best average results among all tested methods.

B. Unknown anomaly detection

To test the performance of the proposed model as well as evaluate the training process, we also perform cross-validation on datasets containing unknown attack data types. Specifically, we perform training on data containing the Gafgyt attack, testing the type of data containing the Mirai attack, and vice versa. The results obtained are presented in table 3. The results are also evaluated according to two criteria, AUC and F1 scores.

TABLE 2. OUTLIERS DETECTION IN COLUMN DATA, G-GAFGYT, M-MIRAI

ID	Data	Outlier Perc	AUC					F1-score				
			IF	LOF	OCSVM	FeaWAD	FeaWAD*	IF	LOF	OCSVM	FeaWAD	FeaWAD*
D1	G	18.03	0.968	0.365	0.980	**0.997**	0.957	0.785	0.132	0.859	**0.981**	0.906
	M	18.03	0.989	0.426	0.991	**0.997**	0.995	0.854	0.099	0.925	**0.962**	0.939
D2	G	14.53	0.973	0.340	0.986	**0.998**	0.997	0.661	0.101	0.871	0.978	**0.983**
	M	14.53	0.995	0.346	0.995	0.363	**0.999**	0.910	0.051	0.944	0.275	**0.989**
D3	G	2.91	0.937	0.537	0.977	0.972	**0.996**	0.133	0.100	0.533	0.600	**0.700**
D4	G	7.41	0.981	0.446	0.990	0.997	**0.998**	0.653	0.139	0.847	**0.958**	0.917
	M	7.41	0.997	0.338	0.994	**0.999**	0.998	0.889	0.000	0.875	0.958	**0.972**
D5	G	6.54	0.987	0.349	0.991	0.999	**1.000**	0.634	0.082	0.836	0.973	**0.986**
	M	6.54	0.998	0.462	0.995	0.999	**1.000**	0.945	0.014	0.877	0.973	**0.986**
D6	G	15.97	0.960	0.345	0.965	0.999	**1.000**	0.728	0.103	0.745	0.984	**0.995**
	M	15.97	0.992	0.350	0.992	**1.000**	0.997	0.864	0.054	0.913	**0.973**	**0.973**
D7	G	15.25	0.898	0.364	0.910	0.992	**0.999**	0.545	0.073	0.534	0.916	**0.978**
D8	G	18.70	0.965	0.365	0.970	**0.998**	0.719	0.767	0.124	0.839	**0.940**	0.535
	M	18.70	0.987	0.394	0.987	**0.996**	0.926	0.834	0.106	0.899	**0.963**	0.654
D9	G	3.85	0.996	0.552	0.995	**0.999**	0.998	0.822	0.081	**0.865**	0.781	**0.865**
	M	3.85	0.998	0.417	0.996	**0.999**	**0.999**	0.892	0.000	0.865	0.946	**0.946**

TABLE 3. UNKNOWN OUTLIERS DETECTION
IN COLUMN DATA, G/M MEANS TRAIN ON GAFGYT, TEST ON MIRAI AND VICE VERSA

ID	Data	Outlier Perc	AUC					F1-score				
			IF	LOF	OCSVM	FeaWAD	FeaWAD*	IF	LOF	OCSVM	FeaWAD	FeaWAD*
D1	G/M	18.03	0.995	0.476	0.990	**0.996**	0.977	0.942	0.384	0.900	0.957	**0.966**
	M/G	18.03	0.813	0.162	0.982	**0.994**	0.832	0.542	0.000	0.883	**0.960**	0.467
D2	G/M	14.53	0.997	0.473	0.992	0.949	**0.998**	0.963	0.397	0.900	0.901	**0.979**
	M/G	14.53	0.877	0.136	0.972	0.902	**0.982**	0.560	0.000	0.846	0.446	**0.945**
D4	G/M	7.41	**0.997**	0.695	0.995	0.636	0.934	**0.916**	0.635	0.878	0.605	0.630
	M/G	7.41	0.976	0.070	0.991	**0.997**	**0.997**	0.568	0.000	0.883	**0.940**	0.925
D5	G/M	6.54	0.997	0.610	0.996	**0.998**	0.993	0.920	0.557	0.866	**0.923**	0.891
	M/G	6.54	0.993	0.055	0.991	0.994	**0.997**	0.661	0.000	0.863	0.623	**0.931**
D6	G/M	15.97	0.997	0.462	0.991	**0.999**	0.917	**0.949**	0.392	0.907	0.981	0.906
	M/G	15.97	0.526	0.162	0.981	0.646	**0.998**	0.506	0.000	0.859	0.424	**0.973**
D8	G/M	18.70	0.985	0.480	**0.989**	0.983	0.516	0.885	0.382	0.902	**0.944**	0.897
	M/G	18.70	0.916	0.170	**0.982**	0.663	0.802	0.596	0.000	0.888	0.860	**0.894**
D9	G/M	3.85	**0.998**	0.686	0.997	**0.998**	**0.998**	0.883	0.625	0.835	0.895	**0.915**
	M/G	3.85	0.980	0.034	0.993	**0.994**	0.992	0.561	0.000	0.800	0.841	**0.885**

Consider the case when training on Gafgyt and testing on Mirai, the IF algorithm gives better results on data D4, D5 (in term of AUC) and D4, D6 (in term of $F1$-score) while FeaWAD and FeaWAD* archive better results on the remaining datasets. According to AUC criteria, FeaWAD is better on D1, D5, and D8, while FeaWAD* is better on D2, D4, D6 and D9. According to the F1-score, FeaWAD is better on D5, D8, while FeaWAD* is better on D1, D2, D9 devices. In the case of training on Mirai and testing on Gafgyt, FeaWAD* is better in most cases (D2, D4, D5, D6 in term of AUC, and D2, D5, D6, D8, D9 in term of $F1$-score). FeaWAD shows better results on D1, D4, D9 (according to AUC) and D1, D4 (according to $F1$-score). Meanwhile, only OCSVM gives better results in one case, D8 (according to AUC).

In previous studies [8], we performed data representation, benign data, and Gafgyt data are next to each other, while Mirai is further away. Thus, if training is performed on benign and Gafgyt, it will be easier to detect Mirai. Our previous experimental results also showed limited results when training on Mirai. But obviously by using an improved version of FeaWAD* the results are giving better results. This will enhance anomaly detection in real-world deployments, where the model is trained on a limited amount of labeled data as well as a small number of known attack types.

C. Cross-validation Evaluation

The Gafgyt and Mirai botnets are two types of malware that infect IoT devices and launch DDoS attacks. Gafgyt is a simpler version of the IRC model, and it mainly uses SYN, UDP, and ACK Flooding attacks. Mirai is a more advanced and dangerous malware, as it can target devices with various architectures and perform different kinds of DDoS attacks based on TCP, UDP, or HTTP protocols. Both Gafgyt and Mirai can generate multiple DDoS attacks, which create different network traffic patterns and feature values for the infected devices. This experiment aims to test how consistent the latent representation produced by FeaWAD and FeaWAD* is when they are trained and tested on one botnet or the other. We consider two scenarios: (1) training and testing on the same attack data, and (2) training on one attack data (such as Gafgyt) and testing on another (such as Mirai).

In this experiment, the metrics AUC, FAR, MDR are used to evaluate the effectiveness of the models. The device used for evaluation is D1. Algorithms used also include: IF, LOF, OCSVM, FeaWAD, FeaWAD*. The experimental results are presented in the results tables table 4 and table 5.

TABLE 4. CROSS VALIDATION CHECKS ON THE SAME ATTACK EXPERIMENTS

Model	Gafgyt			Mirai		
	AUC	FAR	MDR	AUC	FAR	MDR
IF	0.973	0.008	0.298	0.995	0.004	0.085
LOF	0.340	0.041	0.872	0.346	0.047	0.900
OCSVM	0.986	0.007	0.149	0.995	0.006	0.128
FeaWAD	**0.998**	0.002	0.383	0.363	0.003	0.064
FeaWAD*	0.997	**0.001**	**0.021**	**0.999**	**0.001**	**0.001**

The table 4 presents the results for training and evaluation on the same attack type. On the Gafgyt attack, our FeaWAD* tuning model gives an AUC close to FeaWAD, while the FAR and MDR values both give better results. Similar results were obtained on the Mirai dataset.

The table 5 presents the results for cross evaluation. The model is trained on one type of attack and tested on the other type of data. When training on Gafgyt and testing on Mirai, FeaWAD* showed better AUC and MDR results than FeaWAD as well as the remaining models, while FeaWAD gives better results on FAR. When the model was trained on Mirai and evaluated on Gafgyt, FeaWAD* outperformed the remaining models on all evaluation metrics.

Overall assessment, the improved model FeaWAD* gives higher anomaly detection results and lower false positive results than the original FeaWAD model, as well as other competing models.

TABLE 5. CROSS VALIDATION CHECKS ON UNKNOWN ATTACK EXPERIMENTS

Model	Gafgyt/Mirai			Mirai/Gafgyt		
	AUC	FAR	MDR	AUC	FAR	MDR
IF	0.997	0.006	0.116	0.877	0.019	0.428
LOF	0.473	0.019	0.376	0.136	0.050	0.900
OCSVM	0.992	0.008	0.152	0.972	0.008	0.164
FeaWAD	0.949	**0.002**	0.040	0.902	0.006	0.528
FeaWAD*	**0.998**	0.005	**0.016**	**0.982**	**0.004**	**0.076**

D. Limitations

Our proposed models have many benefits for learning a new representation of the IoT datasets, but they also have some limitations. One limitation is that we need enough data to train the models effectively. Another limitation is that the models take more time to run than simpler methods.

From experimental results, the LOF algorithm is faster than all other algorithms (but it also gives the lowest detection results). The training and testing time of FeaWAD and FeaWAD* are slower than LOF and are approximately the same. FeaWAD is slightly faster than FeaWAD*, but not significantly.

IV. CONCLUSION

In this paper, we propose an improved method FeaWAD* for semi-supervised learning, used for IoT network anomaly detection. We've introduced an enhancement to a new encryption strategy that increases IoT malware detection. The proposed model is then tested on different devices, with different proportions of outliers. Experimental results show that our improved method archives better results than the original method as well as the previously available popular methods for anomaly detection.

For future work, we plan to use regularization and bagging methods to reveal the useful latent features of data and improve the anomaly score generator. Moreover, we will test our model on new datasets and domains to evaluate its performance and stability.

REFERENCES

[1] Y. Zhou, X. Song, Y. Zhang, F. Liu, C. Zhu, and L. Liu, "Feature Encoding with AutoEncoders for Weakly-supervised Anomaly Detection," *IEEE Trans Neural Netw Learn Syst*, vol. 33, no. 6, pp. 2454–2465, May 2021, doi: 10.1109/TNNLS.2021.3086137.

[2] A. E. Omolara *et al.*, "The internet of things security: A survey encompassing unexplored areas and new insights," *Comput Secur*, vol. 112, p. 102494, Jan. 2022, doi: 10.1016/J.COSE.2021.102494.

[3] D. T. Son, N. T. K. Tram, and P. M. Hieu, "Deep Learning Techniques to Detect Botnet," *Journal of Science and Technology on Information security*, vol. 1, no. 15, pp. 85–91, Jun. 2022, doi: 10.54654/ISJ.V1I15.846.

[4] N. Hung, Đ. Mai, N. T.-J. of S. and T. on, and undefined 2023, "Network attack classification framework based on Autoencoder model and online stream analysis technology," *isj.vn*, Accessed: Sep. 26, 2023. [Online]. Available: https://isj.vn/index.php/journal_STIS/article/view/938

[5] J. Liu *et al.*, "Deep anomaly detection in packet payload," *Neurocomputing*, vol. 485, pp. 205–218, May 2022, doi: 10.1016/J.NEUCOM.2021.01.146.

[6] C. Qiu, T. Pfrommer, M. Kloft, S. Mandt, and M. Rudolph, "Neural Transformation Learning for Deep Anomaly Detection Beyond Images." PMLR, pp. 8703–8714, Jul. 01, 2021. Accessed: Sep. 14, 2023. [Online]. Available: https://proceedings.mlr.press/v139/qiu21a.html

[7] V. L. Cao, M. Nicolau, and J. McDermott, "Learning Neural Representations for Network Anomaly Detection," *IEEE Trans Cybern*, vol. 49, no. 8, pp. 3074–3087, Aug. 2019, doi: 10.1109/TCYB.2018.2838668.

[8] H. N. Nguyen, N. N. Tran, T. H. Hoang, and V. L. Cao, "Denoising Latent Representation with SOMs for Unsupervised IoT Malware Detection," *SN Computer Science 2022 3:6*, vol. 3, no. 6, pp. 1–15, Sep. 2022, doi: 10.1007/S42979-022-01344-1.

[9] G. Pang, C. Shen, and A. Van Den Hengel, "Deep anomaly detection with deviation networks," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 353–362, Jul. 2019, doi: 10.1145/3292500.3330871.

[10] L. Ruff *et al.*, "DEEP SEMI-SUPERVISED ANOMALY DETECTION," in *8th International Conference on Learning Representations, ICLR 2020*, 2020.

[11] T. Shenkar and L. Wolf, "ANOMALY DETECTION FOR TABULAR DATA WITH INTERNAL CONTRASTIVE LEARNING," in

ICLR 2022 - 10th International Conference on Learning Representations, 2022.

[12] N. T. Dung, N. V. Quân, and N. V. Hùng, "Application of deep learning model in network reconnaissance attack detection," *Journal of Science and Technology on Information security*, vol. 2, no. 16, pp. 60–72, Feb. 2022, doi: 10.54654/ISJ.V1I16.922.

[13] B. Zong *et al.*, "Deep autoencoding Gaussian mixture model for unsupervised anomaly detection," in *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018.

[14] G. Pang, L. Chen, L. Cao, and H. Liu, "Learning representations of ultrahigh-dimensional data for random distance-based outlier detection," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2041–2050, Jul. 2018, doi: 10.1145/3219819.3220042.

[15] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, vol. Part F129685, pp. 665–674, Aug. 2017, doi: 10.1145/3097983.3098052.

[16] H. Xu *et al.*, "Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications," *The Web Conference 2018 - Proceedings of the World Wide Web Conference, WWW 2018*, pp. 187–196, Apr. 2018, doi: 10.1145/3178876.3185996.

[17] M. A. Siddiqui, R. Wright, A. Fern, A. Theriault, T. G. Dietterich, and D. W. Archer, "Feedback-guided anomaly discovery via online optimization," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2200–2209, Jul. 2018, doi: 10.1145/3219819.3220083.

[18] L. Ruff *et al.*, "Deep one-class classification," in *35th International Conference on Machine Learning, ICML 2018*, 2018.

[19] Y. Meidan *et al.*, "N-baiot—network-based detection of iot botnet attacks using deep autoencoders," *IEEE Pervasive Comput*, vol. 17, no. 3, pp. 12–22, 2018.

ABOUT THE AUTHORS

Nguyen Huu Noi

Workplace: Le Quy Don Technical University.

Email: noi.nguyen@lqdtu.edu.vn

Education: He received the BSc degree in applied mathematics and informatics from Lipetsk State University, Lipetsk, Russia. He is currently studying the PhD program in Computer Science at Le Quy Don Technical University.

Recent research interests: Machine learning; Anomaly detection; IoT; Information security.

Cơ quan làm việc: Đại học Kỹ thuật Lê Quý Đôn.

Email: noi.nguyen@lqdtu.edu.vn

Quá trình đào tạo: Cử nhân Toán ứng dụng và tin học tại Đại học bang Lipetsk, Nga. Hiện đang là Nghiên cứu sinh ngành Khoa học máy tính tại Đại học Kỹ thuật Lê Quý Đôn.

Hướng nghiên cứu hiện nay: Học máy; Phát hiện bất thường; IoT; Bảo mật thông tin.

Tran Nguyen Ngoc

Workplace: Le Quy Don Technical University.

Email: ngoctn@lqtdu.edu.vn

Education: He is an Associate Professor and the Head of Cyber Security Group at Le Quy Don Technical University. He received PhD in System analysis, control and information processing from Don State Technical University, Russia.

Recent research interests: Pattern recognition; Cyber security; Artificial intelligence.

Cơ quan làm việc: Đại học Kỹ thuật Lê Quý Đôn.

Email: ngoctn@lqtdu.edu.vn

Quá trình đào tạo: Ông là Phó Giáo sư và Trưởng nhóm An ninh mạng tại Đại học Kỹ thuật Lê Quý Đôn. Ông nhận bằng Tiến sĩ về Phân tích hệ thống, điều khiển và xử lý thông tin tại Đại học Kỹ thuật Don State, Nga.

Hướng nghiên cứu hiện nay: Nhận dạng mẫu; An ninh mạng; Trí tuệ nhân tạo.