ICT Express 11 (2025) 358–363

Contents lists available at ScienceDirect

ICT Express

journal homepage: www.elsevier.com/locate/icte

Robust hierarchical anomaly detection using feature impact in IoT networks

Joohong Rheey, Hyunggon Park *

Graduate Program in Smart Factory, Department of Electronic and Electrical Engineering, Ewha Womans University, Seoul, Republic of Korea

ARTICLE INFO

Keywords:
Adversarial attack
Anomaly detection
Feature impact
Intrusion detection system
Variational autoencoder

ABSTRACT

Security threats in Internet of Things (IoT) networks increased, but the lack of labeled data and limited resources hinder intrusion detection system design for IoT networks. We propose a robust hierarchical anomaly detection method based on a variational autoencoder for IoT networks. Our proposed approach includes a shallow detection stage for obvious outliers with an in-depth detection stage that explicitly measures the impact of individual features on latent representations using Shapley values, enhancing the ability to detect adversarial attacks without adversarial training. Simulations confirm the effectiveness against adversarial attacks, with almost 100% detection rates for NSL-KDD and CIC-IDS2017 datasets.

1. Introduction

Internet of Things (IoT) has become prominent in the design of 5G releases and futuristic 6G visions [1]. The exponential growth in the number of interconnected IoT devices over the globe has prompted a shift in focus from rate-centric use cases, towards ultra-reliable low-latency communications (URLLC) [2]. However, the vast number of IoT devices raises significant concerns about their vulnerability to attacks [3], as IoT devices are typically connected wirelessly through the IoT gateway, where data is either analyzed locally or sent to the cloud for analysis [4]. Concurrently, cyber-attacks have become more sophisticated, with adversarial attacks emerging as a pervasive threat, implying that small imperceivable perturbations of the input data can significantly increase the misclassification in deep learning (DL) models [5]. This can be a more significant threat because DL has emerged as a powerful tool, demonstrating superior performance in detecting anomalies compared to statistical techniques [6].

Furthermore, designing an intrusion detection system (IDS) for IoT networks faces significant challenges, including the difficulty of collecting labeled data [7,8] and the resource constraints of IoT devices [9]. Due to the deficiency of labeled data, variational autoencoder (VAE) [10], a representative unsupervised learning approach among DL models, has been adopted in IDS as it does not require labeled data. The training of VAE can be conducted exclusively on normal traffic data, thereby eliminating the need for labeled abnormal data, which is often scarce or unavailable in IoT networks due to the diversity of devices and traffic patterns.

While VAE is known to be more resistant to input perturbations caused by adversarial attacks than its deterministic counterparts, VAE still remains susceptible to adversarial attacks [11,12]. Moreover, the adoption of existing VAE against adversarial attacks in the network security domain may not be straightforward [13], as most of the studies have been developed for image data [14–16]. Unlike image data, the network traffic data consists of time-based records of network connection flows, generally stored in a structured format [17]. Additionally, while adversarial training may be an effective method against adversarial attacks in image data, it is significantly challenging to apply it in the network security domain due to the difficulty of obtaining labeled attack traffic and generating adversarial attacks against IDS for the defender [18].

To enhance the security of IoT networks, we propose a hierarchical network anomaly detection based on VAE. We use Shapley values [19] to explicitly quantify the feature impact on the latent representation creation of VAE and measure the difference in feature impact between normal and abnormal data. While the Shapley value for model interpretability is studied in [20], we extend their use to anomaly detection by analyzing feature impacts in the latent space, a novel approach for detecting adversarial data in the structured format. The proposed system has a multi-stage detection, i.e., a shallow detection stage and an in-depth detection stage. Since comprehensive detection of the entire data may require enormous computational complexity, the proposed solution selectively investigates the anomalies only for the data points that are not clearly identified in the shallow detection stage. The hierarchical structure enables the proposed IDS to improve the proportion of detected anomalies, especially adversarial traffic data. Furthermore, the proposed method does not require additional adversarial training, making it suitable for resource-constrained IoT environments. This approach distinguishes itself from existing studies [12,21] by specifically

* Correspondence to: Department of Electronic and Electrical Engineering, Ewha Womans University, Seoul 03760, Republic of Korea
E-mail addresses: joohong.rheey@ewha.ac.kr (J. Rheey), hyunggon.park@ewha.ac.kr (H. Park).

https://doi.org/10.1016/j.icte.2025.02.009
Received 4 November 2024; Received in revised form 18 December 2024; Accepted 19 February 2025
Available online 25 February 2025
2405-9595/© 2025 The Authors. Published by Elsevier B.V. on behalf of The Korean Institute of Communications and Information Sciences. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

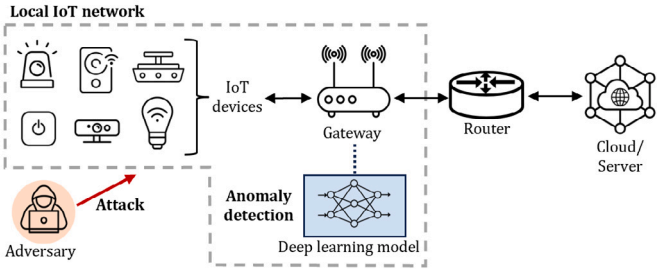


Fig. 1. System architecture including IDS in IoT network scenario.

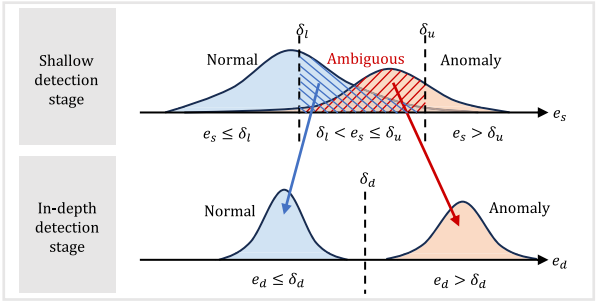


Fig. 2. Proposed multi-stage hierarchical anomaly detection process.

targeting network security domains, rather than focusing on the general robustness of VAEs or adversarial training methods.

2. Training and design of VAE for IoT anomaly detection

We consider IoT scenario consisting of IoT devices connected to a gateway, where traffic patterns of the local IoT network are continuously monitored for anomalies [22]. The proposed IDS includes a VAE as an anomaly detector, designed to analyze IoT traffic at the gateway and send alerts when detecting malicious patterns, as illustrated in Fig. 1.

The VAE consists of an encoder $\mathcal{E}$ and a decoder D. The encoder maps a known input space to the low-dimensional latent space by extracting the mean vector and variance vector of the input data $\mathbf{x} \in \mathbb{R}^K$. It generates the latent representation $\mathbf{z} \in \mathbb{R}^M$ by sampling from the multivariate normal distribution corresponding to the mean vector and variance vector, i.e., $\mathcal{E}(\mathbf{x}) = \mathbf{z}$. Since the latent representation $\mathbf{z}$ is a compressed version of the high-dimensional input $\mathbf{x}$, $M < K$. Consequently, the decoder D makes output $\hat{\mathbf{x}} \in \mathbb{R}^K$ that follows a distribution similar to the input $\mathbf{x}$ using $\mathbf{z}$, i.e., $D(\mathbf{z}) = \hat{\mathbf{x}}$.

Given encoder parameters θ and decoder parameters ϕ, the goal of the training process of the VAE is to determine the optimal parameters, θ^* and ϕ^*, that make the output as close to input as possible for normal traffic data $\mathbf{x}_{tr} \in \mathbb{R}^K$. Specifically, θ^* and ϕ^* can be found by maximizing the evidence lower bound (ELBO) [10], i.e.,

$$\{\theta^*, \phi^*\} = \underset{\theta,\ \phi}{\operatorname{argmax}} \sum_{j=1}^{N} \Big[E_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}_j)} \log p_\theta\left(\mathbf{x}_j \mid \mathbf{z}\right) - D_{KL}\left(q_\phi\left(\mathbf{z} \mid \mathbf{x}_j\right) \parallel p_\theta(\mathbf{z})\right) \Big], \tag{1}$$

where $\mathbf{x}_j$ denotes jth data in the train data batch, E is the operator for the expectation, and N is the number of data in a data batch. In (1), Kullback–Leibler (KL) divergence forces the posterior distribution $q_\phi\left(\mathbf{z} \mid \mathbf{x}_j\right)$ to the prior distribution of the latent representation $p_\theta(\mathbf{z})$, working as a regularization term. Note that the VAE for network anomaly detection is trained only on normal traffic data. Once the training of VAE is completed, the system can detect abnormal traffic data based on the difference from normal traffic data, which is discussed in Section 3.

Since we consider the scenario where the adversary hinders anomaly detection, the input $\mathbf{x}$ of VAE in the detection stages also contains adversarial data that are imperceivable perturbations to the input. In this study, we focus on white-box evasion attacks, where the adversary has access to the parameters, architecture, and gradients of the targeted model. This scenario is different from scenarios with black-box attacks which rely on input–output observations. The adversary crafts adversarial examples by making only minimal modifications to the normal traffic data, while perturbed data can still be classified as benign. An adversarial data $\mathbf{x}_{adv} \in \mathbb{R}^K$ is expressed as $\mathbf{x}_{adv} = \mathbf{x}_{tr} + \boldsymbol{\omega}$, where $\boldsymbol{\omega} \in \mathbb{R}^K$ is perturbation vector.

To attack the VAE, the adversary evaluates the sensitivity of the encoder to each input feature by identifying directions in the data manifold around the normal data in which the targeted model is most sensitive and likely to cause changes in the latent representation [23]. The adversary then exploits the knowledge of sensitive information to select the perturbation $\boldsymbol{\omega}$. Therefore, the perturbation vector $\boldsymbol{\omega}$ can be determined by solving the optimization problem

$$\begin{aligned} &\underset{\boldsymbol{\omega}}{\text{maximize}} \ \|\mathcal{E}(\mathbf{x}_{tr}) - \mathcal{E}(\mathbf{x}_{adv})\| \\ &\text{subject to} \ \|\boldsymbol{\omega}\|_p \le \epsilon \end{aligned} \tag{2}$$

where $\|\cdot\|$ is a norm to appropriately measure the differences between points in the latent space of VAE. Since the adversarial examples should be generated while minimizing perturbation to the original input features, $p = 2$ or $p = \infty$ in general.

Note that we compartmentalize adversarial data that modify the input with imperceivable perturbation and malicious data that is any unauthorized penetration of a network except adversarial attacks. For example, Distributed Denial of Service (DDoS), port scan, and Heartbleed belong to the malicious data. In addition, anomaly or abnormal data encompass both adversarial and malicious data.

3. Hierarchical detection framework for adversarial attacks

In this section, we propose a hierarchical detection method against adversarial data detection that operates on the IoT gateway. The proposed hierarchical solution has two stages, which are a shallow detection stage and an in-depth detection stage. In our hierarchical approach, the shallow detection stage is designed to minimize false negatives, while the in-depth detection stage focuses on reducing false positives. This design is to balance false negatives and false positives, ensuring robust and efficient anomaly detection [24]. In each stage, the abnormality is evaluated based on a reconstruction error of the VAE and a feature impact of the data. An illustrative explanation of the proposed hierarchical detection process is shown in Fig. 2.

3.1. Shallow detection

The shallow detection stage aims to detect distinguishable anomalies. For this, we use a simple but efficient anomaly score $e_s(\mathbf{x}_i)$ for the ith input data $\mathbf{x}_i = [x_i^1, x_i^2, \ldots, x_i^K]$ and the corresponding output data $\hat{\mathbf{x}}_i = [\hat{x}_i^1, \hat{x}_i^2, \ldots, \hat{x}_i^K]$ computed based on the Mean Absolute Error (MAE) [25], i.e.,

$$e_s(\mathbf{x}_i) = \frac{1}{K} \sum_{a=1}^{K} |x_i^a - \hat{x}_i^a|, \tag{3}$$

where $\hat{x}_i^a$ is feature a of the ith reconstructed output data.

Unlike general approaches with a single threshold, we propose to use an upper threshold δ_u and a lower threshold δ_l to explicitly identify the range of data ambiguity. Specifically, input data $\mathbf{x}_i$ is classified as normal if $e_s(\mathbf{x}_i) \le \delta_l$ or abnormal if $e_s(\mathbf{x}_i) > \delta_u$. Otherwise, i.e., $\delta_l < e_s(\mathbf{x}_i) \le \delta_u$, $\mathbf{x}_i$ is considered ambiguous and moved to the next in-depth detection stage. The thresholds in this stage can be determined to minimize the false alarms, aligning with the practical requirements of network anomaly detection.

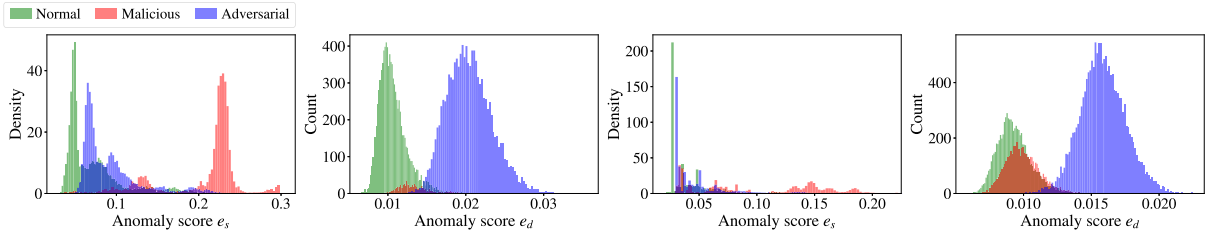


Fig. 3. Histograms of anomaly score at (a) shallow detection stage and (b) in-depth detection stage for NSL-KDD dataset. Histograms of anomaly score at (c) shallow detection stage (d) and in-depth detection stage for CIC-IDS2017 dataset.

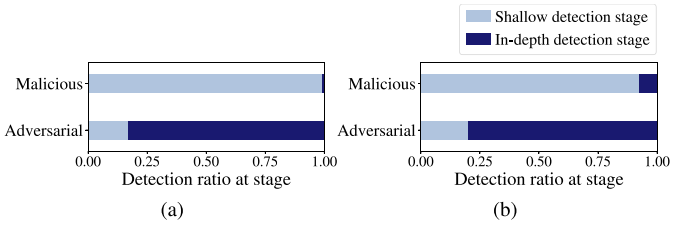


Fig. 4. Detection ratio of each stage for malicious and adversarial data in (a) NSL-KDD dataset and (b) CIC-IDS2017 dataset.

3.2. In-depth detection with feature impact

The in-depth detection stage is only performed for ambiguous data, which is not classified in the shallow detection stage. For the in-depth classification of ambiguous data, it is essential to capture the impact of individual features on generating the latent representation of trained VAE. This is because adversarial examples for VAE are designed to be perceptually similar to normal input data, while their latent representations are encoded to be as far as possible from those of normal input data [21]. To explicitly measure the feature impact on the latent representation creation as a criterion for detection, we adopt the concept of Shapley value, which can measure the average marginal contribution.

For the feature set F and its subset $S(\subseteq F \setminus \{a\})$ that excludes feature a, the impact of feature a in the ith input data is computed as

$$\psi_i^a = \sum_{S \subseteq F \setminus \{a\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} (\mathcal{V}(S \cup \{a\}) - \mathcal{V}(S)), \tag{4}$$

where $\mathcal{V}(\cdot)$ is the value function that maps coalition $|S|$ to a real value that represents the latent representation achieved by the encoder of the VAE with features in the subset S. $|\cdot|$ denotes the cardinality of a set. Therefore, $\mathcal{V}(S \cup \{a\}) - \mathcal{V}(S)$ can be interpreted as the difference between the latent representation with and without feature a. Hence, it can be interpreted that the larger the value, the greater the impact of the feature in creating the latent variable of the VAE.

To compare the impact of each feature between train data (i.e., normal data) and ambiguous data, we define *feature impact* of each feature on generating the latent representation in the train data. It is defined as the average impact of feature a of the train data based on (4), i.e.,

$$\psi_T^a = \frac{1}{L} \sum_{i=1}^{L} |\psi_i^a|, \tag{5}$$

where L denotes the number of instances of train data.

To determine the anomaly of the ambiguous data $\mathbf{x}_i$, we define the anomaly score $e_d(\mathbf{x}_i)$ at the in-depth detection stage by the average discrepancy of feature impacts, expressed as

$$e_d(\mathbf{x}_i) = \frac{1}{K} \sum_{a=1}^{K} |\psi_T^a - \psi_i^a|. \tag{6}$$

An ambiguous data $\mathbf{x}_i$ is classified as normal if $e_d(\mathbf{x}_i) \leq \delta_d$ or abnormal otherwise.

4. Simulation results

This section presents the performance evaluation of the proposed hierarchical detection. To verify the performance of the proposed method, we use popular and high-dimensional network traffic datasets, CIC-IDS2017 [26] and NSL-KDD [27]. Note that we remove the labels from the datasets during the training and detection phases, while using labels only for performance evaluation. Since the datasets consist of normal and malicious data but do not include adversarial attack data, we generate perturbed instances $\mathbf{x}_{adv}$ by (2) and add them to the datasets in the simulations. The model is tested against Fast Gradient Sign Method (FGSM) attack [28], which adds linear perturbation to the input dataset. We use the Kernel SHAP [20] that approximates the Shapley values to compute the value function.

4.1. Performance analysis

We investigate the performance improvement of hierarchical detection over stages. Fig. 3 shows the histograms of the anomaly score and detection ratio at each stage. In Figs. 3(a) and 3(c), which show the histogram of the anomaly score in the shallow detection stage e_s, the normal and adversarial data are overlapped. Consequently, most adversarial data are classified as ambiguous and moved to the in-depth detection stage. The histograms of the anomaly score in the in-depth detection stage e_d are shown in Figs. 3(b) and 3(d). The goal of the in-depth detection stage is to minimize the proportion of undetected anomalies (i.e., false negatives). The ambiguous adversarial data, which were moved from the shallow detection stage, are successfully classified. Only 1.01% of the ambiguous adversarial data in the NSL-KDD dataset and 0.13% of the ambiguous adversarial data in the CIC-IDS2017 dataset are falsely predicted as normal. Therefore, we confirm that the proposed solution is effective in detecting ambiguous adversarial data.

We also study the role of each stage by investigating the detection ratio of each stage. Fig. 4 shows the detection ratio of each stage for the detected malicious and adversarial data. The malicious data are predominantly detected in the shallow detection stage, while the adversarial data are mainly detected in the in-depth detection stage. Hence, each stage of the proposed hierarchical architecture functions well as designed.

4.2. Performance comparisons

We compare the performance of the proposed solution with that of commonly used unsupervised one-class classification approaches, which do not require additional adversarial training, such as One-Class Support Vector Machine (OC-SVM) [29], Isolation Forest (iForest) [30], Histogram-based Outlier Score (HBOS) [31], Deep Support Vector Data Description (DeepSVDD) [32], Copula-Based Outlier Detection (COPOD) [33], and Empirical-Cumulative-distribution-based Outlier Detection (ECOD) [34].

Table 1
Performance comparisons for NSL-KDD dataset.

	Recall	Accuracy	Precision	F1-score	AUC
Proposed	**0.9710**	**0.9375**	**0.9375**	**0.9540**	**0.9208**
OC-SVM	0.5293	0.6518	0.9113	0.5293	0.7131
iForest	0.8470	0.8321	0.8954	0.8705	0.8246
HBOS	0.7184	0.7468	0.8797	0.7184	0.7610
DeepSVDD	0.9058	0.8700	0.8998	0.9028	0.8520
COPOD	0.7268	0.7706	0.9111	0.8086	0.7924
ECOD	0.8841	0.8787	0.9305	0.9067	0.8760

Table 2
Performance comparisons for CIC-IDS2017 dataset.

	Recall	Accuracy	Precision	F1-score	AUC
Proposed	**0.9008**	**0.8695**	**0.9032**	**0.9020**	**0.8539**
OC-SVM	0.3326	0.5228	0.8731	0.4817	0.6180
iForest	0.7834	0.7880	0.8855	0.8313	0.7904
HBOS	0.5366	0.6236	0.8412	0.6552	0.6670
DeepSVDD	0.8916	0.8609	0.8989	0.8953	0.8455
COPOD	0.3581	0.5184	0.8163	0.4978	0.5985
ECOD	0.3897	0.5424	0.8368	0.5318	0.6188

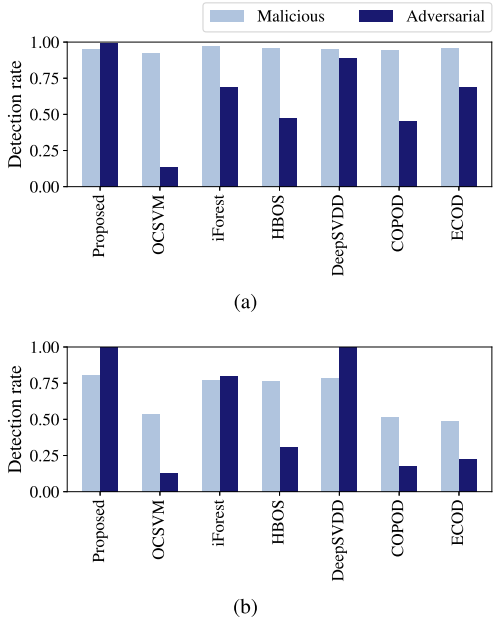


Fig. 5. Detection rate of malicious and adversarial data in (a) NSL-KDD dataset and (b) CIC-IDS2017 dataset.

The numerical results are shown in Tables 1 and 2. Among all metrics and datasets, the proposed solution outperforms all benchmarks. Specifically, the recall performance of the proposed solution is considerably higher than the other algorithms. This highlights the effectiveness of the proposed solution against adversarial attacks, as the goal of adversarial attacks is to deceive a target classifier, leading to a reduction in the proportion of detected anomalies (i.e., recall). This is also confirmed by Fig. 5, which shows the detection rate of each algorithm for adversarial and malicious data. The detection rate of the proposed solution for adversarial data approaches 1.0, which means almost 100% of adversarial data are detected. However, other algorithms struggle to detect adversarial data while they detect malicious data relatively well. Therefore, the proposed method is effective in identifying adversarial attacks, which are difficult to distinguish with existing methods.

5. Conclusion

This paper addresses adversarial attacks in IoT networks by deploying an unsupervised DL model for anomaly detection on an IoT gateway. We propose a robust hierarchical network anomaly detection against adversarial attacks. The simulation results show that the detection performance of the proposed solution for adversarial data is remarkably high. This confirms that the proposed in-depth detection based on feature impact is efficient for adversarial data, which does not require adversarial training. Furthermore, the hierarchical structure can be adapted to the system with distributed computing strategies, e.g., federated learning. Specifically, the shallow detection stage could be operated on resource-constrained edge devices, while computationally intensive tasks such as Shapley value-based feature impact analysis could be performed on centralized servers. Such a distributed approach would enhance scalability in large-scale IoT deployments, addressing concerns for resource-constrained IoT gateways. Future work will focus on integrating distributed computing strategies into the system to further improve scalability and efficiency.

CRediT authorship contribution statement

Joohong Rheey: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Hyunggon Park:** Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing 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 was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) funded by the Korean Government (MSIT) (No. RS-2021-II210739, Development of Distributed/Cooperative AI based 5G+ Network Data Analytics Functions and Control Technology and No. RS-2024-00344830, the Development of 6G Network Architecture and Industrial Convergence Standards Technologies and Standardization).

Appendix A. Computational complexity of in-depth detection stage

The proposed method uses the Shapley value in the in-depth detection stage. The computational complexity associated with feature impact computation using Shapley values is a critical consideration, particularly in resource-constrained IoT environments. This appendix presents a comprehensive examination of the complexity and the methodologies to alleviate the computational burden of the proposed system.

By considering the features as players in a coalition cooperating for accurate predictions, the Shapley value is used for feature selection [35] and is further extended to the decision of a local model feature importance [20]. The Shapley value importance of feature a in model g is computed by

$$\phi_i = \sum_{S \subseteq F \setminus \{a\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} \left[g(\mathbf{x}) - g_{-a}(\mathbf{x}_{-a}) \right], \tag{A.1}$$

where F is the feature set, $S(\subseteq F \setminus \{a\})$ is a subset that excludes feature a, and $|\cdot|$ denotes the cardinality of a set. To compute this, another model g_{-a} is trained with $\mathbf{x}_{-a}$, which represents the values of the features in S. The exact computation of Shapley values involves evaluating the value function for all possible subsets of features, both with and without the feature of interest. For a dataset with $|F|$ features, the complexity of computing the Shapley value for a feature is $O(2^{|F|})$ [36]. This exponential growth in complexity makes direct computation infeasible for high-dimensional datasets.

To address the computational challenges of exact Shapley value computation, [37] uses a sampling approximation of a permutation version of the (A.1) and calculates Shapley sampling values. The Shapley sampling method in [37] approximates the effect of removing a variable from the model by integrating over samples from the training dataset. This eliminates the need to retrain the model and allows fewer than $2^{|F|}$ differences to be computed.

While the Shapley sampling method requires separate sampling estimation for each feature attribution, a Kernel SHAP method [20] requires fewer evaluations of the original model to obtain similar approximation accuracy. The Kernel SHAP method merges concepts from linear regression of a LIME method [38] and Shapley values.

The LIME method interprets model predictions by locally approximating the model around a specific prediction using a set of linear surrogate models. In the LIME, the simplified input $\mathbf{x}'$ is considered as interpretable input that maps original input $\mathbf{x}$ through a mapping function h, i.e., $\mathbf{x} = h(\mathbf{x}')$. The explanation model h' should be described as additive feature attribution methods. The explanation model can be expressed as

$$h'(\mathbf{s}') = \phi_0 + \sum_{i=1}^{M} \phi_i s'_i, \tag{A.2}$$

where $\mathbf{s}' \in \{0,1\}^M$ and s'_i is ith element of $\mathbf{s}'$. M is the number of simplified input features. To find ϕ_a, LIME minimizes the objective function defined as

$$\xi = \underset{h' \in \mathcal{H}'}{\arg\min}\ \mathcal{L}\left(h, h', \pi_{x'}\right) + \Omega(h'), \tag{A.3}$$

where $\mathcal{L}$, $\pi_{x'}$ and Ω refer to the loss function, weighting kernel, and regularization term that penalizes the complexity of the explanation model h'. In the LIME method, the computational complexity is reduced by using a linear explanation model to locally approximate g, where local is measured in the simplified binary input space.

Since LIME is an additive feature attribution method, the Shapley values are the only possible solution to (A.2) that satisfies three properties which are local accuracy, missingness, and consistency. There are the specific forms of $\mathcal{L}$, $\pi_{x'}$ and Ω that make solutions of (A.3) to recover the Shapley value, i.e.,

$$\begin{aligned} &\Omega(h') = 0\ , \\ &\pi_{x'}\left(\mathbf{s}'\right) = \frac{(M-1)}{(M \text{ choose } |\mathbf{s}'|)\,|\mathbf{s}'|\,(M-|\mathbf{s}'|)}\ , \\ &\mathcal{L}\left(h, h', \pi_{x'}\right) = \sum_{\mathbf{s}' \in S} \left[h\left(b^{-1}\left(\mathbf{s}'\right)\right) - h'\left(\mathbf{s}'\right)\right]^2 \pi_{x'}\left(\mathbf{s}'\right). \end{aligned} \tag{A.4}$$

In the Kernel SHAP methods, the importance value for a particular prediction is assigned to each feature based on the LIME method. To assign an importance value to each feature for a particular prediction, the equation for the calculation of additive feature attribution can be defined as

$$\phi_n(h) = \sum_{\mathbf{s}' \subseteq \mathbf{x}'} \frac{|\mathbf{s}'|!\,(M - |\mathbf{s}'| - 1)!}{M!} \left[h\left(\mathbf{s}'\right) - h\left(\mathbf{s}'_{-a}\right)\right]. \tag{A.5}$$

$\mathbf{s}' \subseteq \mathbf{x}'$ represents all $\mathbf{s}'$ vectors for which the non-zero entries are a subset of the non-zero entries in $\mathbf{x}'$. $|\mathbf{s}'|$ denotes the number of non-zero entries in $\mathbf{s}'$. Computing (A.5) is much simpler than computing (A.1).

In the proposed hierarchical approach, Shapley value approximation is adopted exclusively for ambiguous data identified in the shallow detection stage, significantly reducing the overall computational burden. Alternative approaches for Shapley value approximation [39–41] may also be explored depending on factors such as dataset size or complexity.

Appendix B. Simulation setup

In the simulation, the same amount of normal, malicious, and adversarial data are included in the test. Table B.1 shows dataset statistics that are related to normal and malicious data. To generate adversarial examples, we employed the FGSM attack. The process involves perturbing normal data points by adding small, calculated changes to maximize the loss function. For each normal instance x in the dataset, the gradient of the loss function concerning the input, $\nabla_x L(\theta, x, y)$, is calculated. The perturbed instance is then generated as

$$x_{\text{adv}} = x + \epsilon \cdot \text{sign}(\nabla_x L(\theta, x, y)),$$

where ϵ is the perturbation magnitude. In our experiments, we set $\epsilon = 0.1$, as it was empirically determined to balance the generation of challenging adversarial examples while maintaining realistic network traffic characteristics.

Table B.2 summarizes the detailed model architecture and hyperparameter settings used in this study. The configurations are selected based on preliminary experiments to achieve optimal performance. Similarly, the hyperparameter settings for the comparison methods are also fine-tuned to ensure a fair performance evaluation. For the implementation of comparison methods, we use PyOD [42] which is a Python library for detecting anomalous objects.

Table B.1
Dataset statistics.

	NSL-KDD	CIC-IDS2017
Total instances	125,973	2,830,743
Normal traffic	67,343 (53.46%)	2,273,097 (80.30%)
Malicious traffic	58,630 (46.54%)	557,646 (19.70%)
Number of features	41	67

Table B.2
Summary of model architecture and training parameters.

Layered architecture	NSL-KDD (41,30,20,30,41)
	CIC-IDS2017 (67,60,50,40,33,40,50,60,67)
Learning rate	0.001
Batch size	128
Activation function	ReLU
Loss function	MSE
Early stopping	Patience = 20, $\delta_{min} = 0.001$

References

[1] E. Illi, M. Qaraqe, S. Althunibat, A. Alhasanat, M. Alsafasfeh, M. de Ree, G. Mantas, J. Rodriguez, W. Aman, S. Al-Kuwari, Physical layer security for authentication, confidentiality, and malicious node detection: A paradigm shift in securing IoT networks, IEEE Commun. Surv. Tuts. 26 (1) (2024) 347–388.

[2] W. Saad, M. Bennis, M. Chen, A vision of 6G wireless systems: Applications, trends, technologies, and open research problems, IEEE Netw. 34 (3) (2019) 134–142.

[3] V. Gugueoth, S. Safavat, S. Shetty, Security of Internet of Things (IoT) using federated learning and deep learning—Recent advancements, issues and prospects, ICT Express 9 (5) (2023) 941–960.

[4] B. Kang, H. Choo, An experimental study of a reliable IoT gateway, ICT Express 4 (3) (2018) 130–133.

[5] J. Seo, S. Park, J. Kang, Secure wireless communication via adversarial machine learning: A Priori vs. A Posteriori, ICT Express 8 (2) (2022) 220–224.

[6] M. Nasr, A. Bahramali, A. Houmansadr, Defeating DNN-based traffic analysis systems in real-time with blind adversarial perturbations, in: Proc. 30th USENIX Security Symposium, 2021, pp. 2705–2722.

[7] Z. Zhao, Y. Lai, Y. Wang, W. Jia, H. He, A few-shot learning based approach to IoT traffic classification, IEEE Commun. Lett. 26 (3) (2022) 537–541.

[8] H. Kye, M. Kim, M. Kwon, Hierarchical detection of network anomalies: A self-supervised learning approach, IEEE Signal Process. Lett. 29 (2022) 1908–1912.

[9] K. Le, K. Le-Minh, H. Thai, Brainyedge: An AI-enabled framework for IoT edge computing, ICT Express 9 (2) (2023) 211–221.

[10] D.P. Kingma, M. Welling, Auto-encoding variational Bayes, in: Proc. Int. Conf. on Learn. Represent, 2014.

[11] P. Ghosh, A. Losalka, M.J. Black, Resisting adversarial attacks using Gaussian mixture variational autoencoders, in: Proc. AAAI Conf. on AI, 2019, pp. 541–548.

[12] A. Camuto, M. Willetts, S. Roberts, C. Holmes, T. Rainforth, Towards a theoretical understanding of the robustness of variational autoencoders, in: Proc. Int. Conf. on AI and Statistics, 2021, pp. 3565–3573.
[13] H. Jmila, M.I. Khedher, Adversarial machine learning for network intrusion detection: A comparative study, Comput. Netw. 214 (2022) 109073.
[14] I. Higgins, L. Matthey, A. Pal, C.P. Burgess, X. Glorot, M.M. Botvinick, S. Mohamed, A. Lerchner, Beta-VAE: Learning basic visual concepts with a constrained variational framework, in: Proc. Int. Conf. on Learn. Represent, 2017.
[15] S.L. Yin, X.L. Zhang, L.Y. Zuo, Defending against adversarial attacks using spherical sampling-based variational auto-encoder, Neurocomputing 478 (2022) 1–10.
[16] S. Jalalinour, B. Rekabdar, Noisy-defense variational auto-encoder (ND-VAE): An adversarial defense framework to eliminate adversarial attacks, in: Proc. 5th Int. Conf. on Transdisciplinary AI, 2023, pp. 50–57.
[17] A. Azab, M. Khasawneh, S. Alrabaee, K.K.R. Choo, M. Sarsour, Network traffic classification: Techniques, datasets, and challenges, Digit. Commun. Netw. 10 (3) (2024) 676–692.
[18] K. He, D.D. Kim, M.R. Asghar, Adversarial machine learning for network intrusion detection systems: a comprehensive survey, IEEE Commun. Surv. Tuts. 25 (1) (2023) 538–566.
[19] L.S. Shapley, A value for n-person games, in: H.W. Kuhn, A.W. Tucker (Eds.), Contributions to the Theory of Games II, Princeton University Press, Princeton, 1953, pp. 307–317.
[20] S.M. Lundberg, S. Lee, A unified approach to interpreting model predictions, in: Proc. 31st Int. Conf. on Neural Inf. Process. Syst, 2017, pp. 4768–4777.
[21] M. Salehi, A. Arya, B. Pajoum, M. Otoofi, A. Shaeiri, M.H. Rohban, H.R. Rabiee, ARAE: Adversarially robust training of autoencoders improves novelty detection, Neural Netw. 144 (2021) 726–736.
[22] A.K. Pathak, S. Saguna, K. Mitra, C. Åhlund, Anomaly detection using machine learning to discover sensor tampering in IoT systems, in: Proc. IEEE Int. Conf. on Commun, 2021, pp. 1–6.
[23] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, D. Mukhopadhyay, A survey on adversarial attacks and defences, CAAI Trans. Intell. Technol. 6 (1) (2021) 25–45.
[24] H.L. Van Trees, Detection, Estimation, and Modulation Theory, Part I: Detection, Estimation, and Linear Modulation Theory, John Wiley & Sons, 2004.
[25] C.J. Willmott, K. Matsuura, Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance, Clim. Res. 30 (1) (2005) 79–82.
[26] I. Sharafaldin, A.H. Lashkari, A.A. Ghorbani, Toward generating a new intrusion detection dataset and intrusion traffic characterization, in: Proc. 4th Int. Conf. on Inf. Syst. Security and Privacy, 2018.
[27] M. Tavallaee, E. Bagheri, W. Lu, A.A. Ghorbani, A detailed analysis of the KDD CUP 99 data set, in: Proc. IEEE Symposium on Comput. Intell. for Security and Defense Appl, 2009, pp. 1–6.
[28] I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, 2014, arXiv preprint arXiv:1412.6572.
[29] B. Schölkopf, J.C. Platt, J. Shawe Taylor, A.J. Smola, R.C. Williamson, Estimating the support of a high-dimensional distribution, Neural Comput. 13 (7) (2001) 1443–1471.
[30] F.T. Liu, K.M. Ting, Z.H. Zhou, Isolation forest, in: Proc. IEEE Int. Conf. on Data Mining, 2008, pp. 413–422.
[31] M. Goldstein, A. Dengel, Histogram-based outlier score (HBOS): A fast unsupervised anomaly detection algorithm, in: Proc. 35th Annual German Conf. on Advances in AI, vol. 1, 2012, pp. 59–63.
[32] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S.A. Siddiqui, A. Binder, E. Müller, M. Kloft, Deep one-class classification, in: Proc. Int. Conf. on Mach. Learn, 2018, pp. 4393–4402.
[33] Z. Li, Y. Zhao, N. Botta, C. Ionescu, X. Hu, COPOD: Copula-based outlier detection, in: Proc. IEEE Int. Conf. on Data Mining, 2020, pp. 1118–1123.
[34] Z. Li, Y. Zhao, X. Hu, N. Botta, C. Ionescu, G.H. Chen, ECOD: Unsupervised outlier detection using empirical cumulative distribution functions, IEEE Trans. Knowl. Data Eng. 35 (12) (2022) 12181–12193.
[35] S. Cohen, G. Dror, E. Ruppin, Feature selection via coalitional game theory, Neural Comput. 19 (7) (2007) 1939–1961.
[36] T.P. Michalak, K.V. Aadithya, P.L. Szczepanski, B. Ravindran, N.R. Jennings, Efficient computation of the Shapley value for game-theoretic network centrality, J. AI Res. 46 (2013) 607–650.
[37] E. Štrumbelj, I. Kononenko, Explaining prediction models and individual predictions with feature contributions, Knowl. Info. Syst. 41 (2014) 647–665.
[38] M.T. Ribeiro, S. Singh, C. Guestrin, "Why should I trust you?" explaining the predictions of any classifier, in: Proc. 22nd ACM SIGKDD Int. Conf. on Knowl. Discovery and Data Mining, 2016, pp. 1135–1144.
[39] B. Williamson, J. Feng, Efficient nonparametric statistical inference on population feature importance using Shapley values, in: Proc. Int. Conf. on Mach. Learn, 2020, pp. 10282–10291.
[40] M. Sundararajan, A. Najmi, The many Shapley values for model explanation, in: Proc. Int. Conf. on Mach. Learn, 2020, pp. 9269–9278.
[41] K. Aas, M. Jullum, A. Løland, Explaining individual predictions when features are dependent: More accurate approximations to Shapley values, AI 298 (2021) 103502.
[42] Y. Zhao, Z. Nasrullah, Z. Li, PyOD: a python toolbox for scalable outlier detection, J. Mach. Learn. Res. 20 (96) (2019) 1–7.