PAPER

Network Traffic Anomaly Detection: A Revisiting to Gaussian Process and Sparse Representation*

Yitu WANG[†a)], ***Nonmember and*** **Takayuki NAKACHI**[††], ***Senior Member***

SUMMARY Seen from the Internet Service Provider (ISP) side, network traffic monitoring is an indispensable part during network service provisioning, which facilitates maintaining the security and reliability of the communication networks. Among the numerous traffic conditions, we should pay extra attention to traffic anomaly, which significantly affects the network performance. With the advancement of Machine Learning (ML), data-driven traffic anomaly detection algorithms have established high reputation due to the high accuracy and generality. However, they are faced with challenges on inefficient traffic feature extraction and high computational complexity, especially when taking the evolving property of traffic process into consideration. In this paper, we proposed an online learning framework for traffic anomaly detection by embracing Gaussian Process (GP) and Sparse Representation (SR) in two steps: 1). To extract traffic features from past records, and better understand these features, we adopt GP with a special kernel, i.e., mixture of Gaussian in the spectral domain, which makes it possible to more accurately model the network traffic for improving the performance of traffic anomaly detection. 2). To combat noise and modeling error, observing the inherent self-similarity and periodicity properties of network traffic, we manually design a feature vector, based on which SR is adopted to perform robust binary classification. Finally, we demonstrate the superiority of the proposed framework in terms of detection accuracy through simulation.

key words: *network traffic, anomaly detection, Gaussian process, sparse representation*

1. Introduction

Network monitoring provides a view of the current network conditions, which is essential for further network management to maintain secured and high quality transmission. Especially, network anomalies consume additional network resources, and pose significant influence on performance of the communication networks. According to [2], network anomalies can be roughly classified into the following categories,

- *Alpha anomaly* brings unusual high point-to-point traffic, which emits on one dominant source-destination stream, and brings a jump of traffic volume with short duration.
- *Overload* indicates unusual high demand for one network resource or service, which brings a jump of traffic volume in short term.
- *Network/port scanning* means scanning the network for specified open ports or scan a single host for all ports to look for vulnerabilities, which usually brings a jump of traffic volume in short term.
- *Disconnection* indicates network problems that cause a drop in traffic between one source-destination pair, which causes a drop in traffic volume in short/long term.
- *Flow switching* means unusual switching of traffic flows from one inbound router to another, which causes a drop in one traffic stream and release in another.
- *Worm activity* is a malicious program that spreads itself over a network and exploits Operating System (OS) vulnerabilities, which discharges in traffic without a dominant destination address.
- *DoS-/DDoS-attack* represents distributed denial-of-service attack per victim, which emits from multiple sources to a single destination.
- *Point to Multi-point* distributes contents from one server to many users, which emits from the dominant source to several destinations.

Based on the above network anomaly categories, it is observed that most network anomalies cause a sudden drop/jump of network traffic volume, which significantly affects the Quality of Service (QoS) of data transmission, as the bandwidth is constrained and network anomalies often have higher transmission priority. Therefore, the recognition and prediction of future traffic anomaly is extremely important, which could provide the ISP with extra degree of freedom to reconfigure the communication network in a proactive manner, so as to provide reliable and stable network services.

1.1 Literature Survey

Due to its importance, extensive research have been done to understand network traffic and detect traffic anomalies. The existing approaches can be roughly partitioned into two categories, i.e., statistics-based approaches and data-

Manuscript received December 14, 2022.
Manuscript revised May 5, 2023.
Manuscript publicized June 27, 2023.

[†]Tha author is both with the School of Electrical and Information Engineering, Microelectronics and Solid-state Electronics Device Research Center, and the Intelligent Equipment and Precision Measurement Technology R&D Group (2022BSB03104), North Minzu University, Ningxia, Yinchuan, 750021, China.

[††]The author is with the Information Technology Center, University of the Ryukyus, Nishihara-cho, Okinawa, 903-0213 Japan.

*This work is supported in part by Fundamental Research Funds for Central Universities, North Minzu University (No. 2022QNPY07), National Natural Science Foundation of China (No. 62301007), and JSPS Grant-in-Aid for Scientific Research (22K04089). Part of this work has been accepted by IEEE WCNC 2023 [1].

a) E-mail: yituwang@nmu.edu.cn (Corresponding author)

DOI: 10.1587/transfun.2022EAP1161

driven learning-based approaches. In [3], the statistic of network traffic is fitted using the alpha-stable model, such that it becomes possible to classify traffic patterns through a thresholding approach. In [4], a flow-based anomaly detection algorithm is proposed with cross entropy method based on the statistical modeling of network traffic. Even though statistics-based approaches could achieve excellent performance, the network characteristics are actually evolving over time, making it difficult to select the best model and set a suitable threshold, as inappropriate selections may significantly jeopardize the detecting performance. With the advancement of machine learning, data-driven learning-based approaches are gaining increasing popularity. In [5], an auto-encoder has been proposed based on Long Short Term Memory (LSTM) to detect malicious traffic. In [6], Deep Reinforcement Learning (DRL) is utilized to adaptively learn the anomaly features. However, big data is a prerequisite for the well training of Neural Networks (NN), and the learned features are generally packed in a black box, which makes it difficult to understand the network traffic. When labeled traffic data is available, classification-based algorithms, such as Support Vector Machine (SVM) and Bayesian network, have been adopted to learn a classifier for anomaly detection, and achieve high classification accuracy [7], [8]. These classification-based approaches bring considerable flexibility and high accuracy, but collecting labeled training data requires substantial efforts, especially in the context of evolving traffic. Without labeled data, clustering-based algorithms, such as K-means, could work, but with limited performance, since the available information is rather limited [9], [10].

To exploit the past traffic records and obtain insight to facilitate traffic anomaly detection, we consider a light-weight and non-parametric learning technique, i.e., Gaussian Process (GP), which proves to be empirically effective in statistical modeling [11]. In a nutshell, with GP, it becomes possible to encode the discovered domain/expert knowledge into the kernel function of GP, and explicitly optimize the hyper-parameters based on the Bayes theorems to generate explainable results. When handling traffic anomaly detection problems, by encoding the discovered characteristics of network traffic into the kernel function, it becomes possible to accurately model future traffic for anomaly detection. In addition, as a measure of uncertainty over the predicted traffic, GP could analytically infer a posterior distribution of the prediction, which is of vital importance for anomaly detection. In [12], if the predicted traffic exceeds a likelihood threshold based on the inferred distribution, it is assumed that traffic anomaly occurs. However, such method is vulnerable to noise and modeling error. In [13], GP is utilized to model the traffic time series, while the adopted Squared Exponential (SE) kernel is not competent enough to capture much of the variations of network traffic, and the fixed threshold-based classification may suffer from performance degradation with evolving traffic.

1.2 Contributions

In this work, we propose an online learning framework for traffic anomaly detection with GP and SR. Specifically,

1. We reveal that normal and abnormal traffic are composed of not only different patterns, but also more patterns than intuition, which necessitates constructing a more flexible kernel to better model the evolving traffic. For this purpose, we utilize Spectrum Mixture (SM) function, i.e., the mixture of Gaussian in the spectral domain, to dynamically encode the traffic patterns into the kernel of GP, where the mixture components are adjusted according to the current traffic.
2. To maintain high anomaly traffic recognition accuracy with evolving traffic characteristics, we proceed in three steps:
   a). We utilize Wasserstein distance [14] to measure the difference between the distributions of normal and abnormal traffic, such that the evolving traffic statistic is transformed into non-evolving feature.
   b). To combat the noise and modeling error, we manually design a feature vector by involving more past traffic records based on two special properties of network traffic, i.e., self-similarity and periodicity.
   c). To alleviate the difficulty in collecting labeled training data, an ensemble learning framework with SR and Q-learning is adopted to perform robust traffic classification.
3. Through the simulation based on the *GÈANT* dataset [15], it is demonstrated that accurate traffic anomaly detection can be accomplished, and the proposed framework achieves superior performance compared with several baselines.

The rest of this article is organized as follows. Section 2 states the problem. In Sect. 3, we propose the traffic anomaly detection framework. Section 4 compares the performance with several baselines. Finally, Sect. 5 concludes this article, and Sect. 6 demonstrates the future work.

2. Problem Statement

To detect traffic anomalies, we should collect the traffic volumes first. In this work, the time is slotted and there are V time slots in a day. Traffic volume at time t is denoted as $y(t)$, which can be measured at an interface or collected as a flow [16], [17]. In this case, $y(t)$ represents the aggregated amount of network traffic during the length of a time slot [18]. Note that in practice, routers and switches always count the number of packets or bytes, sent or received at each interface. In this regard, $y(t)$ can be collected by periodically reading the counter values [16]. If there is a sudden jump/drop of traffic volume at time slot t, possibly caused by Alpha anomaly, Overload, Network/port scanning, Disconnection and Flow switching, then $y(t)$ is defined as anomaly traffic, as shown in Fig. 1, which occurs at time slot 2700

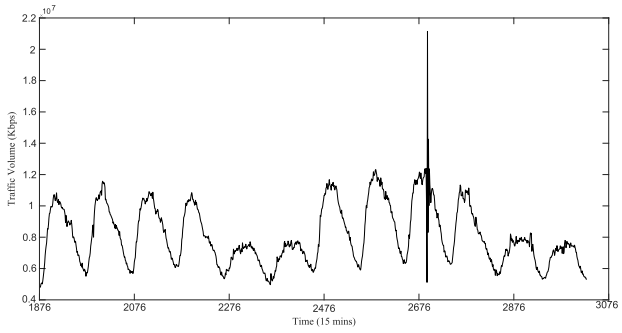


Fig. 1 An example of network traffic.

and ends by time slot 2709.

In this work, the objective is to determine whether traffic anomaly occurs at time slot t, where the decision should be made at the beginning of time slot t. For this purpose, the past traffic volumes are cast into a time series $\{y(t-N), y(t-N+1), \cdots, y(t-1)\}$, where we try to statistically model network traffic based on the past traffic volumes, and estimate the statistic of $y(t)$, which is compared with the past traffic statistics for anomaly detection. The challenge of accurate anomaly detection is pronounced by the intricate evolving characteristics. We try to address this problem head-on by embracing GP and SR to establish an online learning framework for dynamic traffic modeling and classification.

3. Online Learning Framework for Traffic Anomaly Detection

In this section, we propose the online learning framework for traffic anomaly detection in the following steps: First, to more accurately model the traffic based on past traffic records, we adopt Spectral Mixture (SM) function, and encode up to Q patterns into the kernel of GP, such that it becomes possible to more accurately predict the statistic of $y(t)$. Second, Wasserstein distance is utilized to transform the evolving traffic statistic into non-evolving feature, based on which a feature vector is formulated to combat the potential modeling error. Finally, we exploit the inherent sparsity in network traffic to alleviate the difficulty in collecting labeled training data, and adopt SR and ensemble learning to perform robust binary classification.

3.1 Dynamic Traffic Modeling

1) Gaussian Process:

GP is a powerful statistical modeling tool, whose performance is highly controlled by the kernel [20]. In this work, GP models the network traffic by exploring the interconnection among time slots $\mathbf{T}(t) = \{t-N, \cdots, t-1\}$ and traffic records $\mathbf{Y}(t) = \{y(t-N), \cdots, y(t-1)\}$ from a Bayesian perspective. The kernel determines the correlation between any two points, i.e., learning the temporal correlation of network traffic.

First of all, we briefly introduce GP. Given $\mathbf{T}(t)$ and noisy observations $\mathbf{Y}(t)$

$$\mathbf{Y}(t) = f(\mathbf{T}(t)) + \epsilon, \tag{1}$$

where ϵ is an i.i.d. Gaussian noise with zero mean and σ^2 variance, which is caused by system errors, such as measurement and modeling inaccuracy. Under the framework of GP, the mapping function $f(\cdot)$ is approximated according to a probability distribution as

$$f(\mathbf{T}(t)) \sim \mathcal{GP}(m(\mathbf{T}(t)), K(\mathbf{T}(t), \mathbf{T}(t))). \tag{2}$$

It is seen that the approximation accuracy is entirely controlled by the mean function $m(\mathbf{T}(t))$, w.l.o.g. set to zero, and the covariance function $K(\mathbf{T}(t), \mathbf{T}(t))$, which is called kernel of GP.

When applying GP to predict the statistic of network traffic, i.e., inferring the distribution of $y(t^*)$ given a new input $t^* \notin \mathbf{T}(t)$, we first derive the joint prior distribution of $\mathbf{T}(t)$ together with $f(t^*)$ as $\begin{bmatrix} \mathbf{Y}(t) \\ f(t^*) \end{bmatrix}$, which follows

$$\mathcal{N}\Big(\begin{bmatrix} \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{K}(\mathbf{T}(t), \mathbf{T}(t)) + \sigma^2\mathbf{I} & \mathbf{K}(\mathbf{T}(t), t^*) \\ \mathbf{K}(t^*, \mathbf{T}(t)) & K(t^*, t^*) \end{bmatrix}\Big), \tag{3}$$

where the kernel $\mathbf{K}(\mathbf{T}(t), \mathbf{T}(t))$ is generally chosen as a Positive and Symmetric semi-Definite (PSD) function, and the entries of $\mathbf{K}(\cdot)$ are given as $\mathbf{K}_{ij} = K(t-i, t-j)$. Then, by conditioning the joint Gaussian prior distribution on $\mathbf{T}(t)$, the posterior distribution of $f(t^*)$ can be analytically derived as

$$p(f(t^*)|(\mathbf{T}(t), \mathbf{Y}(t), t^*)) \sim \mathcal{N}(\hat{f}(t^*), \sigma^2(t^*)), \tag{4}$$

where the prediction mean and variance are

$$\begin{aligned} \hat{f}(t^*) &= \mathbf{K}_*^T(\mathbf{K}(\mathbf{T}(t), \mathbf{T}(t)) + \sigma^2\mathbf{I})^{-1}\mathbf{Y}(t) \\ \sigma^2(t^*) &= K(t^*, t^*) - \mathbf{K}_*^T(\mathbf{K}(\mathbf{T}(t), \mathbf{T}(t)) + \sigma^2\mathbf{I})^{-1}\mathbf{K}(\mathbf{T}(t), t^*). \end{aligned} \tag{5}$$

The hyper-parameters in $\mathbf{K}$ and σ are trained according to

$$\begin{aligned} \min_{\mathbf{K},\sigma} & \mathbf{Y}^T(t)(\mathbf{K}(\mathbf{T}(t), \mathbf{T}(t)) + \sigma^2\mathbf{I})^{-1}\mathbf{Y}(t) \\ & + \log_2 |\mathbf{K}(\mathbf{T}(t), \mathbf{T}(t)) + \sigma^2\mathbf{I}|, \end{aligned} \tag{6}$$

which can be solved by gradient algorithms, such as Adaptive Moment Estimation (Adam) [20].

2) Kernel Design:

There exists several special properties of network traffic. Especially, self-similarity and periodicity are two major properties, as traffic pattern and human activity are profoundly coupled [19]. In this regard, the existing GP-based traffic modeling algorithms try to combine SE function, Rational Quadratic (RQ) function, and periodic function to mimic these two properties [12], [13]. However, based on the analysis of the *GÈANT* dataset†, we find that both normal and anomaly traffic contains more patterns than intuition, which is also evolving over time. Specifically, to obtain the upper figure in Fig. 2, we manually select a normal time series, i.e., $y(500), y(500), \cdots, y(1459)$, which is

†Please refer to Sect. 4 for the details of the *GÈANT* network.

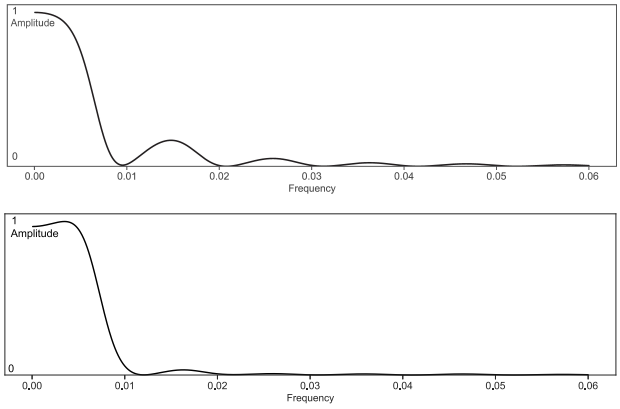


Fig. 2 Spectral graph.

then transformed into the spectral domain based on Fourier transform. The x-axis represents the frequency, and the y-axis denotes the amplitude. Similarly, to obtain the bottom figure in Fig. 2, we manually select an anomaly time series, i.e., $y(2600), y(500), \cdots, y(3559)$, which is then transformed into the spectral domain. We find that besides the major pattern (the peak with the largest amplitude in the spectral domain), minor patterns also exists, which are not only related to anomalies, but also temporally evolving. Since the conventional kernels, such as squared exponential kernel, can only capture the major pattern located at the origin, they are faced with large modeling error when handling such evolving traffic.

In this work, we try to improve kernel flexibility to capture the minor patterns, so as to reduce the traffic modeling error. According to Bochner's theorem [21], it is found that we can formulate a viable kernel through the engineering in spectral domain. Since a mixture of Gaussian is dense [22], through constructing a kernel using a Gaussian mixture in spectral domain, it becomes possible to capture much of the variations of network traffic. Based on the above discussion, the SM kernel is as follows,

$$K(t-i,t-j) = \sum_{q=1}^{Q} \omega_q e^{-2\pi\tau_p^2 v_q} cos(2\pi\tau_p \mu_q), \tag{7}$$

where $\tau_p = j - i$, Q represents the number of mixture components that controls the expressivity of the kernel, μ_q indicates component period of the q-th component, and v_q represents the length-scale of the q-th component [23]. By adjusting each component to different spectral peaks, it becomes possible to dynamically include Q different patterns into the kernel function for more accurate traffic modeling.

3.2 Proposed Online Learning Framework

1) Feature Formulation:

Given the past M days traffic records, i.e., MV traffic samples, we model the traffic based on GP using Algorithm 1. Specifically, Line 1 sets the time index τ to the earliest traffic sample. Line 3 specifies the training set for GP.

Algorithm 1 Traffic Modeling based on GP

1: Initialize $\tau = t - (MV - N)$.
2: **repeat**
3: Select $N < MV$ consecutive traffic samples, i.e, $\mathbf{T}(\tau) = \{\tau - N, \cdots, \tau - 1\}$ and $\mathbf{Y}(\tau) = \{y(\tau - N), \cdots, y(\tau - 1)\}$, as the input of GP.
4: Train the hyperparameters of GP with kernel Eq. (7) according to Eq. (6), to dynamically incorporate Q peaks with larger amplitude.
5: Predict the mean and variance of $y(\tau)$ according to Eq. (5).
6: $\tau = \tau + 1$.
7: **until** $\tau = t$.

Lines 4-5 model the involved traffic time series based on GP, and make one-slot-ahead prediction for traffic anomaly detection. Line 6 updates the time index τ for traffic modeling for the subsequent time slot. In this way, the statistics, i.e., the mean and variance, of $y(t - MV + N), \cdots, y(t - 1)$ can be obtained.

Self-similarity and periodicity are widely acknowledged in traffic time series [19]. As shown in Fig. 1, the fluctuation of 12 days traffic from the *GÈANT* dataset is demonstrated, whose starting time is $t = 1876$ and the ending time is $t = 3028$. Each peak (e.g., from $t = 1876$ to $t = 1876+96$) represents the network traffic for a day. It is observed that the traffic volume is larger in the daytime while becomes smaller at night, and such pattern repeats everyday, which verifies the self-similarity property of network traffic. In addition, the largest traffic volume for the first four days is around 1.1×10^7 Kbps, which is much larger than that for the 5th and the 6th days, which is around 0.75×10^7 Kbps. In the following 6 days, such pattern repeats, i.e., the largest traffic volume for the 7th-10th days is similar, which is much larger than that for the 11th-12th days. As a result, both small-scale periodicity (a day) and large-scale periodicity (every 6 days) exist. Based on the above discussion, it is reasonable to assume that *the statistics of* $y(t)$ *should be similar to those of* $y(t-iV), i \in \{1, 2, \cdots, M\}$*, for otherwise anomaly occurs.*

To measure the difference between two probability distributions $\mathcal{N}(\hat{f}(t), \sigma^2(t))$ and $\mathcal{N}(\hat{f}(t - iV), \sigma^2(t - iV))$, some widely utilized methods in statistical analysis and pattern recognition include Kullback-Leibler (KL) divergence, Jensen-Shanno (JS) divergence [24], and Wasserstein distance [14]. Among these three methods, KL divergence produces a boundless and unbalanced result, and JS divergence is reduced to a fixed constant if the overlap between the support sets of two distributions is small, and thus, we adopt Wasserstein distance in this paper. Since the involved probability distribution is Gaussian, we can calculate the Wasserstein distance between $\mathcal{N}(\hat{f}(t), \sigma^2(t))$ and $\mathcal{N}(\hat{f}(t - iV), \sigma^2(t - iV))$ in closed form as

$$\begin{aligned} S_i(t) &= W_2(\mathcal{N}(\hat{f}(t), \sigma^2(t)), \mathcal{N}(\hat{f}(t - iV), \sigma^2(t - iV))) \\ &= \|\hat{f}(t) - \hat{f}(t - iV)\|_2^2 + \|\sigma(t) - \sigma(t - iV)\|_{\text{Frobenius}}^2. \end{aligned} \tag{8}$$

In this way, $S_i(t)$ should be small if both $y(t)$ and $y(t-iV)$ are normal traffic, while becomes much larger if abrupt anomaly

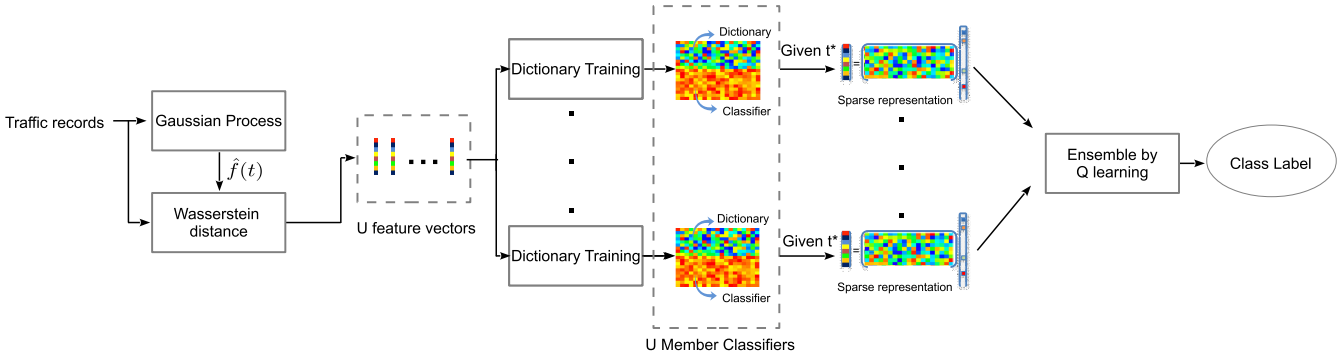


Fig. 3 Structure of the proposed online learning framework.

occurs. However, since the characteristics of the traffic process is evolving, selecting a fixed threshold for anomaly detection is not reliable. In addition, when judging whether anomaly occurs using only two points, such as considering $S_1(t)$ only, the algorithm will be extremely sensitive to noise and modeling error.

Alternatively, we consider to utilize classification algorithms for anomaly detection, which brings considerable flexibility and accuracy [25]. In a nutshell, classification-based algorithms utilize labeled training data to learn a classifier, which is used to classify a new instance. Especially, in the context of traffic anomaly detection, the problem is further simplified to binary classification, i.e., normal and anomaly. To this end, to better exploit the past data for anomaly detection in a holistic manner, we formulate a feature vector based on periodicity and self-similarity of network traffic as

$$\boldsymbol{Z}(t) = (S_{M-1}(t), S_{M-2}(t), \cdots, S_1(t))^T, \tag{9}$$

where the j-th column can be interpreted as the degree of support given by $y(t - jV)$ to $y(t)$. Intuitively, the larger the support, the more likely to identify $y(t)$ to the class which $y(t - jV)$ belongs to. By stacking the support $S_i(t), \forall i \in \{1, 2, \cdots, M-1\}$ into a vector, the influence of few modeling error of $y(t - jV)$ could be reduced.

To improve the robustness of classification, and further combat noise and modeling error, we make use of the idea of ensemble learning, whose basic concept is to create several member classifiers with similar bias and then combine their outputs to reduce the variance, as shown in Fig. 3. For this purpose, we partition the support $S_i(t), \forall i \in \{1, 2, \cdots, M-1\}$ into U groups, e.g., the j-th group contains $\{S_j(t), S_{j+U}(t), \cdots, S_{j+(W-1)U}(t)\}$, where $W = (M-1)/U$ is assumed to be an integer, which are further stacked into U vectors according to Eq. (9), respectively. In this way, we can obtain the feature vectors $\boldsymbol{Z}_j(t), \forall j \in \{1, 2, \cdots, U\}$ according to Algorithm 2.

2) Training:

Training the member classifiers requires labeled traffic, which is nontrivial, especially when utilizing the feature vectors $\boldsymbol{Z}_j(t), \forall j \in \{1, 2, \cdots, U\}$. Consider the following example: $y(t)$ belongs to normal traffic, and there exists a index $k^* \in \{1, 2, \cdots, W-1\}$ such that $y(t - (j + k^*U)V)$ belongs to

Algorithm 2 Feature Vector Formulation

1: Initialize $\tau = t - N$.
2: **repeat**
3: Given $\hat{f}(\tau - iV), \forall i \in \{1, 2, \cdots, M-1\}$ and $\sigma(\tau - iV)), \forall i \in \{1, 2, \cdots, M-1\}$, calculate $S_i(\tau), \forall i \in \{1, 2, \cdots, M-1\}$ according to Eq. (8).
4: $j = 1$.
5: **for** $j \le U$ **do**
6: Formulate feature vector $\boldsymbol{Z}_j(\tau)$ by stacking $\{S_j(t), S_{j+U}(t), \cdots, S_{j+(W-1)U}(t)\}$ according to Eq. (9).
7: $j = j + 1$.
8: **end for**
9: $\tau = \tau + 1$.
10: **until** $\tau = t$.

anomaly traffic. Different index k^* results in a different class of the feature vector $\boldsymbol{Z}_j(t)$. In addition, if there exists more than one index where the associated traffic is anomaly, then any different combination of the positions of anomaly traffic corresponds to a distinguished class of $\boldsymbol{Z}_j(t)$, which further pronounces the challenge of training data collection. To address this issue, we utilize SR [26], which has been successful in computer vision. The reason to adopt this technique is two-fold,

- It is robust to a few variations of $y(t - (j + kU)V), \forall k \in \{1, 2, \cdots, W-1\}$ in terms of normal/anomaly due to the sparsity consideration. In this sense, it becomes possible to partition $\boldsymbol{Z}_j(t)$ into much fewer number of classes, which significantly alleviates the difficulty in training data collection.
- The algorithm requires few training data, and is with low computational complexity, which makes it possible to frequently update the hyper-parameters in evolving environment.

Observing that most traffic anomalies affect the traffic volume in short term, i.e., only a small number of $y(t - (j + kU)V)$ is anomaly, we partition the formulated feature vectors into two categories as follows,

- Normal feature vector: If $y(t)$ belongs to normal traffic, then $\boldsymbol{Z}_j(t)$ is a normal feature vector.
- Anomaly feature vector: If $y(t)$ belongs to anomaly traffic, then $\boldsymbol{Z}_j(t)$ is an anomaly feature vector.

In order to train U member classifiers, it is necessary to obtain U training sets. As for the j-th member classifier, since the associated feature vector $\boldsymbol{Z}_j$ is W-dimensional, and thus, the corresponding training set is denoted as $\boldsymbol{B}_j \in \mathbb{R}^{W\times|\boldsymbol{B}_j|}$. To generate $\boldsymbol{B}_j$, we should first manually gather $|\boldsymbol{B}_j|$ normal/anomaly traffic samples, and calculate the corresponding feature vectors according to Eq. (8) and Algorithm 2. Then, we stack the feature vectors into a matrix $\boldsymbol{B}_j$, and store the corresponding class labels in $\boldsymbol{H}_j$. Next, we jointly train a dictionary with K atoms for sparse representation, and classifier parameters for anomaly detection by adopting Label Consistent K-Singular Value Decomposition (LC K-SVD) [28] based on the following optimization problem,

$$\begin{aligned}&\arg\min_{\boldsymbol{X},\boldsymbol{D}_j,\boldsymbol{W}_j,\boldsymbol{A}_j} \|\boldsymbol{G}_j - \boldsymbol{E}_j\boldsymbol{X}\|_2^2\\ &s.t. \|\boldsymbol{x}_i\|_0 \le \epsilon, \forall i \in \{1,2,\cdots,|\boldsymbol{B}_j|\},\end{aligned} \tag{10}$$

where

$$\boldsymbol{G}_j = \begin{bmatrix}\boldsymbol{B}_j\\ \sqrt{\alpha}\boldsymbol{C}_j\\ \sqrt{\beta}\boldsymbol{H}_j\end{bmatrix}, \boldsymbol{E}_j = \begin{bmatrix}\boldsymbol{D}_j\\ \sqrt{\alpha}\boldsymbol{A}_j\\ \sqrt{\beta}\boldsymbol{W}_j\end{bmatrix}, \tag{11}$$

in which $\boldsymbol{D}_j \in \mathbb{R}^{M\times K}$ represents the dictionary, which expresses the training set $\boldsymbol{B}_j$ in the most compact manner, i.e., we can recover any feature vector in $\boldsymbol{B}_j$ using the fewest number of atoms (or columns) in $\boldsymbol{D}_j$. It is demonstrated that by mimicking the sparsity mechanism of human vision system, the inherent sparsity of SR also brings significant discriminativity [27]. To enhance such discriminativity for improving the performance of classification, a classifier $\boldsymbol{W}_j$ can be trained jointly with $\boldsymbol{D}_j$ [29]. In the above equation, $\boldsymbol{X} \in \mathbb{R}^{K\times|\boldsymbol{B}_j|}$ denotes the sparse representation based on $\boldsymbol{D}_j$, α and β are the weights for the label consistent term and the reconstruction error term, controlling the balance between discriminativity and representability, $\boldsymbol{C}_j$ is the discriminative sparse code, $\boldsymbol{A}_j$ is the linear transformation matrix, and ϵ is the sparsity constraint. This problem can be solved efficiently using K-SVD algorithm [28].

3) Classification:

Given a testing sample $\boldsymbol{Z}^*(t)$, its sparse representation based on $\boldsymbol{D}_j$ can be calculated according to

$$\arg\min_{\boldsymbol{x}^*(t)} \|\boldsymbol{Z}^*(t) - \boldsymbol{D}_j\boldsymbol{x}^k(t)\|_2^2 \ s.t. \ \|\boldsymbol{x}^k(t)\|_0 \le \epsilon, \tag{12}$$

which can be solved efficiently using Orthogonal Matching Pursuit (OMP) algorithm [29]. Then, the degree of support given by the j-th member classifier that $\boldsymbol{Z}^*$ belongs to each class can be estimated according to

$$\boldsymbol{L}_j(t) = \boldsymbol{W}_j\boldsymbol{x}^*(t), \tag{13}$$

which is a two dimensional vector.

In this work, we use the Q-learning algorithm to ensemble the outputs of the member classifiers. The agent keeps trying all actions in all states with non-zero probability and must sometimes explore by choosing at each step a random action with probability $1-\beta$. This is referred to as β-greedy approximation [30]. The agent, state, and perceived reward associated to the Q-learning framework are defined as follows,

- Agent: The ensembler as demonstrated in Fig. 3.
- State: $\chi(t) = (\boldsymbol{L}_1(t), \boldsymbol{L}_2(t), \cdots, \boldsymbol{L}_U(t))$.
- Action: Member classifier selection ($\boldsymbol{L}_l(t)$).
- Reward: $f_{ik}(\chi(t), \boldsymbol{L}_l(t)) = 1$ if $\arg\max \boldsymbol{L}_l(t)$ is the correct label of $\boldsymbol{Z}^*(t)$. Otherwise, $f_{ik}(\chi(t), \boldsymbol{L}_l(t)) = 0$.

Q-factors are updated according to the following dynamics,

$$\begin{aligned}&Q_{ik}^{t+1}(\chi(t), \boldsymbol{L}_l(t)) =\\ &(1-\alpha)Q_{ik}^t(\chi(t), \boldsymbol{L}_l(t)) + \gamma(f_{ik}(\chi(t), \boldsymbol{L}_l(t))),\end{aligned} \tag{14}$$

where $Q_{ik}^t(\chi(t), \boldsymbol{L}_l(t))$ represents the current value in the Q-table, γ is the learning rate. If the algorithm chooses to explore, we select each member classifier with equal probability. Then, the Q-table is updated according to (14). If the algorithm chooses to exploit, we will use the current Q-table for classifier selection, and update the Q-table according to the classification result.

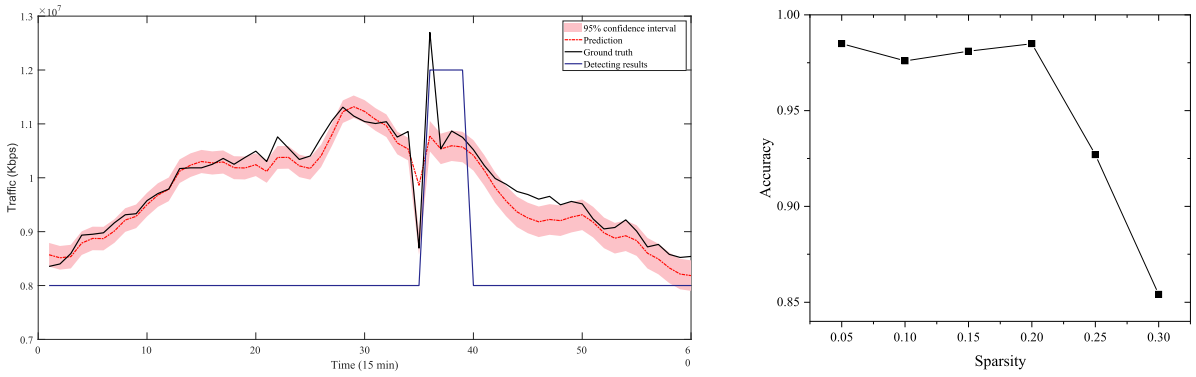
Figure 3 demonstrates the structure of the proposed learning framework. Given the past traffic samples, network traffic is analyzed and modeled using GP based on Algorithm 1. Then, based on self-similarity and periodicity of network traffic, we assume that the statistics of $y(t)$ should be similar to those of $y(t-iV), i \in \{1,2,\cdots,M\}$, for otherwise anomaly occurs. Next, to quantify the difference between two probability distributions, Wasserstein distance is adopted to formulate the feature vector $\boldsymbol{Z}(t)$ for anomaly detection based on Algorithm 2. After that, Sparse Representation (SR) is invoked for binary classification. To promote the robustness, we incorporate ensemble learning to SR, and train U member classifiers based on SR. Given a new traffic sample, each of the U member classifiers generates a classification result, which are ensembled by Q-learning and produces the class label.

Remark 1. *Regarding the computational complexity of the proposed online learning framework,*

- *In the dynamic traffic modeling algorithm, the most time-consuming part lies in calculating the inverse of* $\mathbf{K}$*, which needs* $O(N\log(N))$ *operations [31]. If* $\mathbf{K}$ *is a positive semi-definite matrix, which generally holds, then low-rank approximations can be applied, and the computational complexity can be further reduced to* $O(N)$ *[32].*
- *In the classification algorithm, the most time-consuming part of classification lies in jointly training the dictionary and classifier using K-SVD, in which the running time follows the order* $O(|\boldsymbol{B}_j|^2MK)$ *[33]. Note that since SR only requires few training data,* $|\boldsymbol{B}|_j$ *can be chosen small in practice.*

4. Simulation Results

In this section, the performance is evaluated through simula-



(a) Detecting results (b) Influence of sparsity

Fig. 4 Performance evaluation.

tion. After briefly introducing the adopted publicly available dataset, we compare the performance with three baselines,

- Baseline 1 (GP-based approach [12]): GP is adopted to calculate the mean and variance of the upcoming time slot, if the actual traffic is larger than $\hat{f}(t) + \delta\sigma^2(t)$, where δ controls the confidence region, then it is believed to be anomaly.
- Baseline 2 (Threshold-based approach) [13]: If Wasserstein distance $W_2(\mathcal{N}(\hat{f}(t), \sigma^2(t)), \mathcal{N}(\hat{f}(t - iV)$ is larger than a fixed threshold, if $y(t - V)$ is known normal, then $y(t)$ is anomaly, and wise versa.
- Baseline 3 (Proposed without ensemble learning) [1]: The entire $(S_{M-1}(t), S_{M-2}(t), \cdots, S_1(t))^T$ is served as the feature vector, based on which one classifier is trained to perform binary classification.

4.1 Dataset Description

We adopt *GÈANT* [15] for performance evaluation. In this paper, the traffic samples are collected from a link from *GÈANT*. Specifically, flow duration conforms to $\mathcal{N}(60, 12)$, flow arrival conforms to uniform distribution, and traffic volume conforms to Gaussian distribution [34]. Every 15 minutes, the accumulated amount of traffic is recorded during a 4-month period, i.e., 10772 traffic data points from 2015-01-01 00:00AM to 2015-04-30 00:00AM are considered.

4.2 Results and Analysis

In this simulation, regarding the parameters adopted,

- As for dynamic traffic modeling, SM function with $Q = 8$ mixture component is utilized as the kernel of GP, $N = 960$ traffic sample points, corresponding to one day traffic, are used for traffic modeling. Adam with learning rate 0.5 and iteration number 150 is adopted to train the hyper-parameters of GP.
- As for binary classification, the total number of training vectors is set to 20, half of which belongs to normal class, and the other belongs to anomaly class. We adopt two member classifiers for ensemble learning, where 20 training vectors are equally divided into $\boldsymbol{B_1}$ and $\boldsymbol{B_2}$ for training the member classifiers. Through one-dimensional search, we have $\alpha = 1$ and $\beta = 1$ to achieve a nice balance between discriminativity and representability, so as to achieve the best performance. The discriminative dictionary is with 10 atoms, and the total number of testing vectors is 8000.

The simulation is executed on a personal computer with GPU Nvdia GeForce 3060 Laptop (65 W), which is equivalent to conventional Nvdia GeForce 2060 Super. Upon observing a new traffic sample, it approximately takes 13.8 seconds to execute the proposed algorithm, where the training of the two member classifiers is in parallel. Compared with the slot length of the *GÈANT* dataset, which is 15 minutes, the computation can be finished in realtime. Therefore, the proposed framework can be applied to scenarios with a wide range of time scales.

In Fig. 4(a), the detecting results are presented. It is demonstrated that there will be a one-slot-delay, i.e., anomaly can be detected right after it happened. In addition, the anomaly traffic poses an exponentially decreasing impact on traffic modeling, as shown in Eq. (7), the weight ω_q is exponentially decreasing with τ_p. Therefore, in a few subsequent time slots, traffic is still alarmed. Figure 4(b) demonstrates the influence of sparsity on performance. It is observed that when a small fraction of atoms in the dictionary is used (less than 20 percent), the classification accuracy is higher, which verifies the robustness of the proposed framework to a few variations of $y(t-(j+kU)V), \forall k \in \{1, 2, \cdots, W-1\}$ in terms of normal/anomaly when enforcing a tight sparsity constraint.

Figure 5 compares the performance of the proposed framework and those of the three baselines in a holistic manner, i.e., all the anomaly points in the *GÈANT* dataset are considered. To quantify the performance, we first partition the dataset into two parts, one for training and the other for testing, where the total number of testing vectors is 8000. Then define *accuracy* as the number of correct classification, i.e., classify a normal traffic into normal class or classify an anomaly traffic into anomaly class, divide the total number

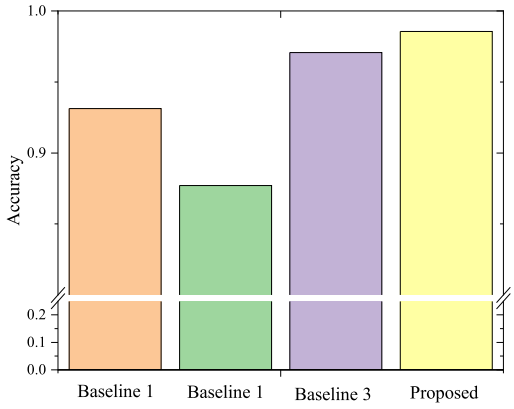


Fig. 5 Performance comparison.

of testing points, i.e., 8000. The baseline 1 and baseline 2 are sensitive to noise and modeling error, as a fixed threshold is not reliable enough. The baseline 3 relies on a single classifier, since the optimal trade-off between bias and variance is difficult to find, the proposed framework with ensemble learning performs better.

5. Conclusions

In this paper, we established an online learning framework for network traffic anomaly detection. 1). We utilize the SM kernel to more accurately model the network traffic, in which the position of mixture components are dynamically adjusted to follow the evolving traffic characteristics. 2). Wasserstein distance is utilized to transform the evolving traffic statistic into a non-evolving feature for traffic classification. 3). To alleviate the difficulty in data collection and perform robust classification, an ensemble learning framework is proposed based on SR and Q-learning. Finally, the proposed framework performs better in terms of detection accuracy compared with several baseline algorithms through simulation.

6. Future Work

The growing number of anomaly traffic behaviors calls for new technologies to detect anomaly in real-time and with high accuracy. In cases with big data, deep learning-based algorithms have been proven to show higher accuracy and robustness [35], [36]. In cases with small data, although the proposed framework works well, it encounters additional challenge, i.e., the difficulty of collecting labeled traffic samples is alleviated by leveraging several inherent properties of network traffic and sparsity, but not completely eliminated, which reduces the timeliness of the proposed framework. In the future, we are willing to address this issue by incorporating unsupervised machine learning, so as to facilitate real-time detection of network traffic anomaly.

References

[1] Y. Wang, R. Dong, T. Nakachi, and W. Wang, "A light-weight online learning framework for network traffic abnormality detection," Proc. IEEE WCNC 2023, 2023.

[2] D. Ageyev, T. Radivilova, O. Mulesa, O. Bondarenko, and O. Mohammed, "Traffic monitoring and abnormality detection methods for decentralized distributed networks," Springer Inf. Security Technol. Decentralized Distrib. Netw., vol.115, pp.287–305, April 2022.

[3] F. Simmross-Wattenberg, J.I. Asensio-Perez, P. Casaseca-de-la-Higuera, M. Martin-Fernandez, I.A. Dimitriadis, and C. Alberola-Lopez, "Anomaly detection in network traffic based on statistical inference and α-stable modeling," IEEE Trans. Depend. Sec. Comput., vol.8, no.4, pp.494–509, Aug. 2011.

[4] I. Nevat, D. Divakaran, S. Nagarajan, P. Zhang, L. Su, L. Ko, and V. Thing, "Anomaly detection and attribution in networks with temporally correlated traffic," IEEE/ACM Trans. Netw., vol.26, no.1, pp.131–144, Feb. 2018.

[5] M. Elsayed, N. Le-Khac, S. Dev, and A. Jurcut, "Detecting abnormal traffic in large-scale networks," Proc. IEEE ISNCC 2020, pp.1–7, Oct. 2020.

[6] S. Dong, Y. Xia, and T. Peng, "Network abnormal traffic detection model based on semi-supervised deep reinforcement learning," IEEE Trans. Netw. Service Manag., vol.18, no.4, pp.4197–4212. Dec. 2021.

[7] J. Yang, J. Deng, S. Li, and Y. Hao, "Improved traffic detection with support vector machine based on restricted Boltzmann machine," Soft Comput., vol.21, no.11, pp.3101–3112, June 2017.

[8] I. Karatepe and E. Zeydan, "Anomaly detection in cellular network data using big data analytics," Proc. EWC 2014, pp.1–5, May 2014.

[9] M. Parwez, D. Rawat, and M. Garuba, "Big data analytics for user-activity analysis and user-anomaly detection in mobile wireless network," IEEE Trans. Ind. Informat., vol.13, no.4, pp.2058–2065, Aug. 2017.

[10] D. Naboulsi, R. Stanica, and M. Fiore, "Classifying call properties in large-scale mobile traffic datasets," Proc. IEEE INFOCOM 2014, pp.1806–1814, May 2014.

[11] H. Liu, Y.S. Ong, X. Shen, and J. Cai, "When Gaussian process meets big data: A review of scalable GPs," IEEE Trans. Neural Netw. Learn. Syst., vol.31, no.1, pp.4405–4423, Jan. 2020.

[12] C. Bock, F. Aubet, J. Gasthaus, A. Kan, M. Chen, and L. Callot, "Online time series anomaly detection with state space Gaussian processes," arXiv preprint, arXiv:2201.06763, 2022.

[13] J. Pang, D. Liu, H. Liao, Y. Peng, and X. Peng, "Anomaly detection based on data stream monitoring and prediction with improved Gaussian process regression algorithm," Proc. IEEE ICPHM 2014, pp.1–7, June 2014.

[14] L. Chizat, P. Roussillon, F. Léger, F. Vialard, and G. Peyre, "Faster Wasserstein distance estimation with the Sinkhorn divergence," Proc. NeurIPS 2020, no.33, pp.2257–2269, Nov. 2020.

[15] S. Uhlig, B. Quoitin, J. Lepropre, and S. Balon, "Providing public intradomain traffic matrices to the research community," SIGCOMM Comput. Commun. Rev., vol.36, no.1, pp.83–86, Jan. 2006.

[16] T. Li, S. Chen, and Y. Ling, "Fast and compact per-flow traffic measurement through randomized counter sharing," IEEE Proc. IEEE INFOCOM 2011, pp.1799–1807, April 2011.

[17] A. Svigelj, R. Sernec, and K. Alic, "Network traffic modeling for load prediction: A user-centric approach," IEEE Netw., vol.29, no.4, pp.88–96, Aug. 2015.

[18] L. Nie, Z. Ning, M. Obaidat, B. Sadoun, H. Wang, S. Li, L. Guo, and G. Wang, "A reinforcement learning-based network traffic prediction mechanism in intelligent internet of things," IEEE Trans. Ind. Informat., vol.17, no.3, pp.2169–2180, June 2020.

[19] T. Akgul, S. Baykut, M. Erol-Kantarci, and S.F. Oktug, "Periodicity-based anomalies in self-similar network traffic flow measurements," IEEE Trans. Instrum. Meas., vol.60, no.4, pp.1358–1366, Oct. 2010.

[20] C. Rasmussen and C. Williams, Gaussian Processes for Machine Learning, MIT Press, 2006.

[21] M. Stein, Interpolation of Spatial Data: Some Theory for Kriging, Springer Verlag, 1999.

[22] A. Wilson and R. Adams, "Gaussian process kernels for pattern discovery and extrapolation," Proc. ICML 2013, pp.1067–1075, Feb. 2013.
[23] Y. Wang, T. Nakachi, and W. Wang, "Pattern discovery and multi-slot-ahead forecast of network traffic: A revisiting to Gaussian process," IEEE Trans. Netw. Service Manag., vol.20, no.2, pp.1691–1706, 2023.
[24] J. Hershey and P. Olsen, "Approximating the Kullback Leibler divergence between Gaussian mixture models," Proc. ICASSP 2007, pp.317–320, April 2007.
[25] Q. Zhu and L. Sun, "Big data driven anomaly detection for cellular networks," IEEE Access, vol.8, pp.31398–31408, Feb. 2020.
[26] Y. Wang and T. Nakachi, "A privacy-preserving learning framework for face recognition in edge and cloud networks," IEEE Access, vol.8, pp.136056–136070, July 2020.
[27] Y. Sun, Q. Liu, J. Tang, and D. Tao, "Learning discriminative dictionary for group sparse representation," IEEE Trans. Image Process., vol.23, no.9, pp.3816–3828, June 2014.
[28] Z. Jiang, Z. Lin, and L. Davis, "Learning a discriminative dictionary for sparse coding via label consistent K-SVD," Proc. IEEE CVPR 2011, pp.1697–1704, June 2011.
[29] Q. Zhang and B. Li, "Discriminative K-SVD for dictionary learning in face recognition," Proc. IEEE CVPR 2010, pp.2691–2698, June 2010.
[30] M. Bennis and D. Niyato, "A Q-learning based approach to interference avoidance in self-organized femtocell networks," Proc. IEEE GLOBECOM 2010, pp.706–710, Dec. 2010.
[31] V. Raykar and R. Duraiswami, "Fast large scale Gaussian process regression using approximate matrix-vector products," Proc. Learning Workshop 2017, pp.1–8, March 2007.
[32] H. Harbrecht, M. Peters, and R. Schneider, "On the low-rank approximation by the pivoted Cholesky decomposition," Applied Numerical Mathematics, vol.62, no.4, pp.428–440, 2012.
[33] S. Hsieh, C. Lu, and S. Pei, "Fast OMP: Reformulating OMP via iteratively refining $\ell 2$-norm solutions," Proc. IEEE SSPW 2012, pp.189–192, Aug. 2012.
[34] J. Reis, M. Rocha, T.K. Phan, D. Griffin, F. Le, and M. Rio, "Deep neural networks for network routing," Proc. IEEE IJCNN 2019, pp.1–8, July 2019.
[35] R. Hwang, M. Peng, C. Huang, P. Lin, and V.L. Nguyen, "An unsupervised deep learning model for early network traffic anomaly detection," IEEE Access, vol.8, pp.30387–30399, Feb. 2020.
[36] G. Wei and Z. Wang, "Adoption and realization of deep learning in network traffic anomaly detection device design," Soft Comput., vol.25, pp.1147–1158, Aug. 2020.

Yitu Wang received the Ph.D. degree from Zhejiang University, Hangzhou, China, in 2018. From August to November 2014, he was a visiting scholar with the University of Paris-Sud, Orsay, France. From January 2019 to March 2022, he was a researcher with NTT Innovation Laboratories, NTT Corporation, Japan. He is currently a Lecturer with the School of Electrical and Information Engineering, North Minzu University, China.

Takayuki Nakachi received the Ph.D. degree in electrical engineering from Keio University, Tokyo, Japan, in 1997. From 1997 to 2021, he was a senior researcher with Nippon Telegraph and Telephone Corporation (NTT), Japan. From 2006 to 2007, he was a visiting scientist at Stanford University. He is currently a Professor with the Information Technology Center, University of the Ryukyus, Japan.