SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

July 2024, Vol. 67, Iss. 7, 170305:1–170305:16
https://doi.org/10.1007/s11432-023-3978-y

Special Topic: XAI-Enabled Network Automation for 5G and Beyond: Move Closer to Zero-Touch

Path signature-based XAI-enabled network time series classification

Le SUN[1], Yueyuan WANG[1], Yongjun REN[1*] & Feng XIA[2]

[1]*Department of Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET), Nanjing University of Information Science and Technology, Nanjing* 210044*, China;*
[2]*School of Computing Technologies, Royal Melbourne Institute of Technology, Melbourne* VIC 3000*, Australia*

Received 19 June 2023/Revised 24 January 2024/Accepted 12 March 2024/Published online 27 June 2024

Abstract Classifying network time series (NTS) is crucial for automating network administration and ensuring cyberspace security. It enables the detection of anomalies, the identification of network attacks, and the monitoring of performance issues, thereby providing valuable support for network protection and optimization. However, modern communication networks pose challenges for NTS classification methods. These include handling large-scale and complex NTS data, extracting features from intricate datasets, and addressing explainability requirements. These challenges are particularly pronounced for complex 5G networks. Notably, explainability has become crucial for the widespread deployment of network automation for 5G networks and beyond. To tackle these challenges, we propose a path-signature-based NTS classification model called recurrent signature (RecurSig). This innovative model is designed to overcome the time-consuming feature selection problem by utilizing deep-learning (DL) techniques. Additionally, it provides a solution for addressing the black-box issue associated with DL models in network automation systems (NAS) by incorporating an explainable classification approach. Extensive experimentation on various public datasets demonstrates that RecurSig outperforms existing models in accuracy and explainability. The results indicate its potential for application in cyberspace security and automated network management, offering an explainable solution for network protection and optimization.

Keywords network time series classification, explainable artificial intelligence, path signature, automated network management, recurrent neural network

1 Introduction

Classifying network time series (NTS) is pivotal for cybersecurity and network performance management [1,2]. Analyzing and classifying NTS data aid in the detection of abnormal behavior, identification of network attacks, and monitoring of performance issues. This process provides essential data support for allocating automated network resources and security protection. Automation boosts network efficiency and performance, reducing the reliance on manual intervention.

The 5G era has spurred rapid Internet growth, expanding the scale and complexity of NTS [3,4]. This trend is expected to continue into the upcoming 6G era, envisioning ubiquitous connectivity [5]. Traditional methods for classifying NTS, such as port- and payload-based techniques, are losing effectiveness due to increased encryption in the 5G era. Researchers have proposed combining statistical or time series features with machine learning (ML). Nevertheless, these methods rely heavily on expert knowledge and feature engineering, which increase costs and limit their universality. Deep learning (DL) has improved performance in encrypted NTS classifications due to its end-to-end design and robust learning capabilities [6,7]. Nonetheless, the lack of transparent model structures and training processes in DL presents 'black box' issues, including unclear classification mechanisms and missing decision criteria [8]. These challenges hinder users from establishing trust in the models.

A network automation system (NAS) requires explainability to support accurate decision-making in NTS classification problems [9]. A significant amount of research has been conducted on explainability in computer vision [10] and natural language processing [11]. However, further effort is required to develop

* Corresponding author (email: renyj100@126.com)

methods suitable for time-series data. Recent studies introduced explainable artificial intelligence (XAI) techniques for NTS classification [12]. For instance, Zebin et al. [13] proposed a balanced and stacked random forest (RF) classifier to maintain transparency in decision-making. This is effective only for specific NTS types and involves the challenge of feature selection. Similarly, Cui et al. [14] suggested a flow prototype network (FP) that provided flow-level traceability and packet-level annotation. However, limitations persist in model training, tuning, and computational efficiency.

To enhance deep neural network (DNN)-based NTS classification, path signatures are employed for both explainability and accuracy. This method effectively represents time-series data and transforms them into continuous paths [15]. By capturing dynamic time-series patterns, it generates high-dimensional, interpretable features, simplifying key feature extraction. Its geometric interpretation also enhances the explainability [16]. In [16], a practical implementation of path signatures, including embedding methods and truncation orders, was presented, with an emphasis on statistical and ML applications. Our research centers on reusing path signatures and integrating them into DL techniques. Furthermore, it addresses data imbalance issues by capturing underlying structural patterns, resulting in improved accuracy, particularly for categories with fewer samples.

Accurate NTS classification methods play a crucial role in network automation [17]. NAS aims for unsupervised operation, and misclassifying NTS can lead to data breaches and system disruptions. Conversely, an accurate NTS classification aids in automation by offering insight into the distribution of temporal patterns in the network. These insights enable the evaluation of network loads, resource optimization, and performance enhancement. Therefore, improving NTS classification accuracy is essential for NAS reliability, performance, and security [18], providing robust support for network management and optimization.

To address the issues above, we propose an NTS classification model based on path signatures that support XAI called recurrent signature (RecurSig). RecurSig includes two main components: Verbesserung (Verbe) and signature recurrent neural networks (SigRNN). Verbe is a data augmentation module utilizing one-dimensional convolutional neural networks (1D-CNN). It enhances the original data while retaining its intrinsic flow properties. SigRNN is a pioneering feature-extraction module equipped with XAI-enabled properties. It converts time series into multidimensional paths and computes interpretable signature features. Extensive experiments on six public datasets validate the superiority of RecurSig by comparing it with state-of-the-art methods. Our distinctive contributions are as follows.

• We integrate explainable feature extraction techniques into DNN to build an NTS classification model called RecurSig. This model autonomously extracts interpretable spatiotemporal features from NTS data, tackling both the time-consuming feature extraction problem and the black box issues associated with DL.

• We propose the Verbe and SigRNN modules for extracting explainable features. Verbe helps alleviate information loss caused by truncating signatures. SigRNN addresses the issue of path signatures disrupting the streamlike nature of the data. The explainable spatiotemporal features are thoroughly captured through the application of multiple signature extraction layers.

• We conduct extensive experiments on six public datasets to validate the effectiveness of RecurSig. RecurSig achieves higher accuracy than non-XAI DNN models for NTS classification and general time-series classification. Furthermore, we effectively transfer it to two general time-series classifications, achieving promising classification accuracy.

The paper is structured as follows. Section 2 provides a brief review of related studies. Section 3 provides a detailed description of RecurSig. Experimental settings and results are outlined in Section 4. Lastly, Section 5 concludes the paper and suggests future research directions.

2 Related work

2.1 XAI technologies

The XAI research focuses on developing systems for explaining decisions and reasoning. Gu et al. [19] proposed a comprehensive attention-based CNN (CA-Net) for interpretable medical image segmentation. Tan et al. [20] proposed an uncertainty-aware convolutional RNN (UA-CRNN) with an uncertainty-aware attention module. These studies employed attention mechanisms for interpretability but faced challenges in capturing dynamic changes in time-series data. The path signature method addresses this issue through

Table 1 Comparison of different explainable models

Model	Explainable method	Application field	Time series characteristics	Data imbalance
CA-Net [19]	Attention	Image segmentation	✗	✓
UA-CRNN [20]	Attention	Time series forecasting	✓	✗
Grad-CAM [21]	Heatmap	Image classification	✗	✗
CAVs [22]	Concept	Image classification	✗	✗
FP [14]	Prototype	NTS classification	✓	✗
ROULETTE [9]	Attention	NTS classification	✓	✗
RecurSig	Path signature	NTS classification	✓	✓

its intuitive representation of data dynamics and versatile applications. Additionally, techniques such as gradient-weighted class activation mapping (Grad-CAM) [21] explain network attention areas using heat maps. Kim et al. [22] presented concept activation vectors (CAVs) to explain internal model states in image classification and medical applications. However, the reliance of the CAVs on linear separability is limited to complex data contexts. To address this limitation, the optimization algorithm concept activation regions (CARs) has been proposed [10]. Although effective for image and language processing, their direct applicability to NTS classification is limited. NAS requires tailored approaches to interpret the complex and dynamic nature of NTS.

2.2 5G NTS classification

Various methods have been employed for NTS classification [1]. Taylor et al. [23] utilized the statistical features of packet size to train an RF classifier for fingerprint recognition. In the 5G era, the dynamic and complex nature of NTS fuels the need for NAS. Consequently, numerous studies have turned to DNNs for NTS classification. Hassan et al. [24] proposed a weight-dropped long short-term memory network for detecting network intrusions. Lotfollahi et al. [6] introduced a deep packet approach utilizing CNN and stacked autoencoder frameworks for classifying encrypted packet payloads. Various approaches have emerged to address the black-box nature of DNNs in NTS classification. Zebin et al. [13] introduced a balanced and stacked RF model for accurate detection of domain-name service traffic. Cui et al. [14] proposed an FP, which is an explainable encrypted traffic classification network based on a flow picture scheme. Andresini et al. [9] introduced ROULETTE, a novel neural model focusing on precise and explainable NTS classification. Additional discussions on achieving powerful and interpretable AI in 6G networks are presented in [25]. Table 1 [9,14,19–22] compares RecurSig and the aforementioned explainable models. This underscores RecurSig's innovation, employing path signatures for NTS classification interpretability. RecurSig uniquely addresses data imbalance and time-series analysis, challenges not concurrently managed by other models.

2.3 General time series classification

Most cutting-edge time-series classification methods employ DL approaches [26]. The rapid development of explainable methods has spurred interest in enhancing the explainability of time-series analysis models [20]. Be et al. [27] introduced an explainable time-frequency CNN to discriminate the microseismic waveforms of rock fracturing. Hsieh et al. [28] proposed a modular CNN-based feature extraction and attention mechanism for explainable multivariate time series classification. Kim et al. [29] introduced a DL model based on an autoencoder for predicting energy demand. For time-series classification in 6G-enabled intelligent transportation systems, our previous study [30] proposed a meta-transfer metric learning method. This study proposes RecurSig, an XAI-enabled DNN based on path signatures.

3 Recurrent signature

In this section, we introduce the proposed RecurSig model for explainable feature extraction and NTS classification. Figure 1 illustrates the RecurSig framework for network automation in a 5G environment, outlining the process from data collection (see [31]) to application.

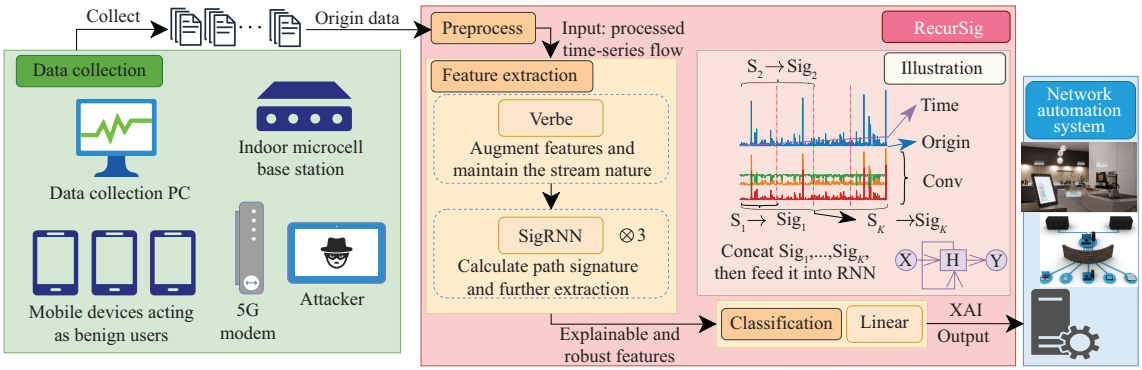

Figure 1 (Color online) Overview of RecurSig: an XAI-enabled NTS classification and automation system for 5G and beyond. The green-framed left section depicts the architecture for 5G network-based NTS collection, which is not the focus of this study. NTS data is fed into RecurSig. The outcome is then integrated into the NAS. The 'Illustration' provides a detailed view of RecurSig's feature extraction process.

Table 2 Meaning of the main notations

Notation	Meaning
$p_k = \{x_{k1}, x_{k2}, \ldots, x_{kl}\}$	The k_{th} path; l is the length of a path
$t_k = \{t_{k1}, t_{k1}, \ldots, t_{k(lt)}\}$	A time sequence normalized in [0,1]
$V_k = \{v_k^1, v_k^2, \ldots, v_k^m\}$	V_k is the collection of feature sequences of p_k; $v_k^i (i \in [0, 1, \ldots, m])$ is the i_{th} channel of V_k; m is the channel size of V_k
$a_k^{\theta_1} = \{\{p_k\}, \{t_k\}, \{V_k = \{v_k^1, v_k^2, \ldots, v_k^m\}\}\}$	$a_k^{\theta_1}$ is the collection of augment sequences after Verbe; θ_1 is the parameter of 1D-CNN in Verbe
$R_k = \{r_k^1, r_k^2, \ldots, r_k^{\text{oc}}\}$	R_k is the collection of final feature sequences of p_k; oc is the out channel sizes of RNN in SigRNN
$s_k = \{s_{k,1}, s_{k,2}, \ldots, s_{k,K}\}$	s_k is the collection of slices yielded by moving extended window; K is the number of slices
$\text{Sig}_k^N = \{\text{Sig}_{k,1}^N, \text{Sig}_{k,2}^N, \ldots, \text{Sig}_{k,K}^N\}$	Sig_k^N is the N truncated path signature of s_k; N is the truncation order

3.1 RecurSig

Path P in $\mathbb{R}^d$ is a continuous mapping from the interval $[0,T]$ to $\mathbb{R}^d$, which is written as $P : [0,T] \to \mathbb{R}^d$. Using NTS signals as an example, the temporal variation in each dimensional feature (such as the mean time for two data packets to be sent forward) can be regarded as a one-dimensional path P_t of length l. Specifically, $P_t = P_t^1 = \text{mean}_t$ and t assume values from a set of discrete time steps $\{t_1, t_2, \ldots, t_l\} \in [0,T]$. NTS flows with multiple features can be considered multi-dimensional paths, represented as $P_t = \{P_t^1, P_t^2, \ldots, P_t^d\}, t \in [0,T]$. Each $P_t^k = \{x_{t_1}^k, x_{t_2}^k, \ldots, x_{t_l}^k\}$ represents a series of observations for the kth feature, where $k \in \{1, 2, \ldots, d\}$.

Table 2 summarizes the main symbols used in RecurSig. RecurSig is the XAI-based NTS classification model implemented in Algorithm 1. Firstly, we apply min-max normalization (line 1 in Algorithm 1) to the numerical feature vectors to ensure stable and faster model convergence by balancing the feature scales. Due to the data imbalance in real-world temporal datasets, we use one-sided selection and the synthetic minority oversampling technique (SMOTE) [6] for data resampling (line 2). One-sided selection cleans the data and reduces overfitting, whereas SMOTE adds synthetic minority samples to improve model generalization. In the second step, we enhance the data by concatenating the time-series signals and features extracted from the 1D-CNN model, referred to as Verbe (line 3). Subsequently, SigRNN is used to iteratively extract more explainable features (lines 4–6). A critical step involves constructing stream of streams through the extended windows. Finally, feature classification is performed using a fully connected layer (line 7).

3.1.1 *Verbe in RecurSig*

The steps of Verbe are illustrated in Algorithm 2. In practical applications, truncating the path signature to the Nth-order commonly occurs, leading to the loss of higher-order information. Therefore, enhancing the data before computing the path signature is necessary. To mitigate this issue and preserve the streamlike nature of the data, pointwise augmentation is employed. This enables an Nth-order truncated

Algorithm 1 RecurSig(p_k)

Input: A path p_k; signature depth N; number of categories n;
Output: c = the class of p_k;
1: Scale and normalize p_k using min-max normalization: $p_k^{\text{norm}} = (p_k - \min)/(\max - \min)$;
2: Create a balanced subset of training data using one-sided selection and SMOTE;
3: Augment p_k: $a_k^{\theta_1} = \{\{p_k\}, \{t_k\}, \{V_k = \{v_k^1, v_k^2, \ldots, v_k^m\}\}\} = \text{Verbe}(p_k)$;
4: **for** $i = 1$ to 3 **do**
5: $R_k = \{r_k^1, r_k^2, \ldots, r_k^{\text{oc}}\} = \text{SigRNN}(a_k^{\theta_i}, N)$;
6: **end for**
7: XAI output: $u = P_1, P_2, \ldots, P_n = \text{Linear}(R_k)$;
8: Return $c = \text{argmax}(u)$.

Algorithm 2 Verbe(p_k)

Input: A path p_k; hyperparameter of 1D-CNN;
Output: $a_k^{\theta_1}$ = the augment features of p_k;
1: Preserve the stream-like nature of the data: $a_k^{\theta_{1,1}} = f_{\text{conv}}(p_k, \theta_{1,1})$;
2: **for** $i = 1$ to m **do**
3: Augment: $a_k^{\theta_{1,i+1}} = f_{\text{conv}}(a_k^{\theta_{1,i}}, \theta_{1,i+1})$;
4: **end for**
5: Truncated p_k: $p_k = \{x_{k1}, x_{k2}, \ldots, x_{k(lt)}\}$;
6: Include time t_k: $t_k = \{t_{k1}, t_{k2}, \ldots, t_{k(lt)}\}$;
7: Concatenate p_k, t_k and $a_k^{\theta_{1,m+1}}$: $a_k^{\theta_1} = \text{concat}(p_k,\ t_k\ , a_k^{\theta_{1,m+1}})$;
8: Return $a_k^{\theta_1}$.

signature to capture relevant information from higher- through lower-order terms [32]. Intuitively, let $F(x) = (x, f(x))$ and f be fixed feature-mapping functions, for example $f(x) = x^2$. Assuming $x \in \mathbb{R}$ with a length of l, the shape of the data before augmentation is $l \times 1$. After pointwise augmentation, the shape becomes $l \times 2$. This augmentation preserves the flow property of the data, allowing the computation of the path signature of $F(x)$ instead of x, thereby extracting more meaningful features. However, each dataset has unique characteristics necessitating a dataset-specific feature-mapping function f. Selecting the optimal value of f is time-consuming. Instead, we prefer f to be data-dependent rather than pre-fixed. Therefore, we employ a DNN approach to train f, making it learnable and denoting it as f^θ.

Let $p_k = \{x_{k1}, x_{k2}, \ldots, x_{kl}\}$, where $x_{ki} \in \mathbb{R}^d (i \in [1, l])$. First, pointwise augmentation is considered. Let the feature mapping be denoted by $f^\theta : \mathbb{R}^d \to \mathbb{R}^m$. The extracted features are given by

$$a_k^\theta = \left(f\left(x_{k1}, \theta\right), f\left(x_{k2}, \theta\right), \ldots, f\left(x_{kl}, \theta\right)\right), \tag{1}$$

where $f\left(x_{ki}, \theta\right) \in \mathbb{R}^m\ (i \in [1, l])$. 1D-CNNs frequently replace FC layers for pointwise augmentation to reduce parameters and enhance nonlinearity. By employing a kernel size of one, FC layers can be replaced with 1D-CNNs. Regardless of the kernel size, the stream-like nature of the data remains unchanged as the data is sequentially scanned along the time axis. Intuitively, for a convolutional kernel of size kernal $\in N$ and feature mapping $f^\theta : \mathbb{R}^{d \times \text{kernal}} \to \mathbb{R}^m$, the extracted features are expressed by

$$\begin{aligned} a_k^\theta = \big(&f\left(x_{k1}, x_{k2}, \ldots, x_{k(\text{kernalt})}, \theta\right), f\left(x_{k2}, x_{k3}, \ldots, x_{k(\text{kernal}+1)}, \theta\right), \\ &\ldots, f\left(x_{k(\text{l}-\text{kernal}+1)}, x_{k(\text{l}-\text{kernal}+2)}, \ldots, x_{kl}, \theta\right)\big) = \left(f_{\text{conv}}\left(p_k, \theta\right)\right). \end{aligned} \tag{2}$$

In this case, $f\left(x_{ki}, x_{ki+1}, \ldots, x_{k(i+\text{kernal}-1)}, \theta\right) \in \mathbb{R}^m$ (where $i \in [1, \text{l} - \text{kernal} + 1]$). As depicted in (2), the sequence length shifts from l to $\text{l} - \text{kernal} + 1$. This enables us to extract informative features while reducing the data flow, resulting in faster model training.

Thus, we initially employ a 1D-CNN with a kernel size greater than one to traverse the entire feedforward network along the stream (see (3)).

$$a_k^{\theta_{1,1}} = f_{\text{conv}}\left(p_k, \theta_{1,1}\right) = \sigma\left(W^1 * p_k + b^1\right). \tag{3}$$

Parameter θ_1 represents the weight parameters that the first group aims to learn. Specifically, $\theta_{1,1} = \left\{W^1, b^1\right\}$ refers to the parameters of the first hidden layer in the network and W^1 denotes the convolutional kernel utilized in the CNN. The symbol $*$ represents the convolution operation and $\sigma()$ represents the activation function ReLU(). In addition, an m-layer 1D-CNN network is added to perform pointwise

enhancement and obtain the extracted feature $a_k^{\theta_1}$, as shown in

$$\begin{aligned} a_k^{\theta_{1,2}} &= f_{\text{conv}}\left(a_k^{\theta_{1,1}}, \theta_{1,2}\right) = \sigma\left(W^2 * a_k^{\theta_{1,1}} + b^2\right), \ldots, \\ a_k^{\theta_{1,m+1}} &= f_{\text{conv}}\left(a_k^{\theta_{1,m}}, \theta_{1,m+1}\right) = \sigma\left(W^{m+1} * a_k^{\theta_{1,m}} + b^{m+1}\right). \end{aligned} \tag{4}$$

Due to the reduced length of feature $a_k^{\theta_1}$, we truncate the original path p_k to align it with $a_k^{\theta_1}$(line 5 in Algorithm 2). Additionally, we treat time $t_k = \{t_{k1}, t_{k2}, \ldots, t_{klt}\}$ as a one-dimensional feature (line 6), where t_{ki}(where $i \in [1, lt]$) is in the range of [0,1], defined as $t_{ki} = \frac{1}{lt} \times i$. Finally, we concatenate p_k, t_k, and $a_k^{\theta_1}$ along the channel dimension (line 7) to obtain the features extracted using Verbe. After all processes, the input data shape becomes $lt \times (m+2)$, incorporating one original path, one time-information path, and m feature paths captured by the CNN. Thus, Verbe supplements information to the original path, thereby alleviating data imbalance.

3.1.2 *SigRNN in RecurSig*

The structure of SigRNN is depicted in Figure 2, with the steps outlined in Algorithm 3. Path signatures are typically applied in the final stage of feature extraction due to data stream disruptions. However, considering their powerful feature extraction capabilities, it is natural to consider reusing path signatures for improved feature extraction. Therefore, we propose an extended-window approach to construct a stream of streams for the data flow (line 2). Preserving the inherent temporal and sequential properties of NTS is crucial. The concept of stream of streams involves sequentially arranging multiple streams to form a higher-level stream. For instance, 100 packets over one period can be divided into five segments of 20 packets each. These segments are sorted and aggregated into a higher-order stream with a length of five, where each stream contains 20 packets. Unlike the average distribution method, our extended-window approach maintains a constant starting position while adjusting its size. We expand the intervals, starting with a window size (w) and incrementing it with a fixed length adjustlength (al) for each iteration. For the Verbe-enhanced feature stream, denoted as $a_k^{\theta_1}$ and simplified to $a_k^{\theta_1} = \left\{a_{k1}, a_{k2}, \ldots, a_{k(lt)}\right\}$, where a_{ki} $(i \in [1, lt]) \in \mathbb{R}^d$, we construct a stream of flows using

$$\begin{aligned} s_k &= \left\{s_{k,1}, s_{k,2}, \ldots, s_{k,K}\right\} \\ &= \left\{\left\{a_{k1}, a_{k2}, \ldots, a_{kw}\right\}, \left\{a_{k1}, a_{k2}, \ldots, a_{kw+\text{al}}\right\}, \ldots, \left\{a_{k1}, a_{k2}, \ldots, a_{k(lt)}\right\}\right\}. \end{aligned} \tag{5}$$

Despite the increase in data volume resulting from this expansion method, the efficiency of the path signature remains high according to Chen's identity. Specifically, we consider two d-dimensional path segments: $P_1 : [0, T_1] \to \mathbb{R}^d$ and $P_2 : [T_1, T] \to \mathbb{R}^d$. A natural approach is to compute the signature of the complete path formed by concatenating the two segments. Chen's identity demonstrates that a signature can be efficiently computed using

$$S(X * Y)_{0,T} = S(X)_{0,T_1} \otimes S(Y)_{T_1,T}, \tag{6}$$

where $*$ denotes the operation of the concatenating paths.

Next, we compute the nth-order truncated signature of each path in s_k (line 3). Let $P : [0, T] \to \mathbb{R}^d$ be a d-dimensional path with $T > 0$. For any positive integer $k \geqslant 1$ and $i_1, i_2, \ldots, i_k \in \{1, 2, \ldots, d\}$, the kth-order feature coefficient of path P is defined by

$$S(P)_{0,T}^{i_1,\ldots,i_k} = \int_{0<t_k<T} \cdots \int_{0<t_1<t_2} \mathrm{d}P_{t_1}^{i_1} \ldots \mathrm{d}P_k^{i_k}. \tag{7}$$

The Nth-order truncated path signature of s_k, denoted by $\text{Sig}_k^N = \left\{\text{Sig}_{k,1}^N, \text{Sig}_{k,2}^N, \ldots, \text{Sig}_{k,K}^N\right\}$ is derived using

$$\text{Sig}_{k,i}^N = (1, S(P_i)_{0,T}^1, \ldots, S(P_i)_{0,T}^d, \ldots, S(P_i)_{0,T}^{1,\ldots,1}, \ldots, S(P_i)_{0,T}^{i_1,\ldots,i_N}, \ldots, S(P_i)_{0,T}^{\overbrace{d,..,d}^{N}}), \ i \in [1, K]. \tag{8}$$

If $d > 1$, then $\text{Sig}_{k,i}^N (i \in [1, K]) \in \mathbb{R}^{(d^{N+1}-1)/(N-1)}$, and $\text{Sig}_k^N \in \mathbb{R}^{K \times (d^{N+1}-1)/(N-1)}$. If $d = 1$, $\text{Sig}_{k,i}^N \in \mathbb{R}^k$, and $\text{Sig}_k^N \in \mathbb{R}^{K \times k}$. It can be observed that for the original data stream, Sig_k^N is a vector of length 1 with

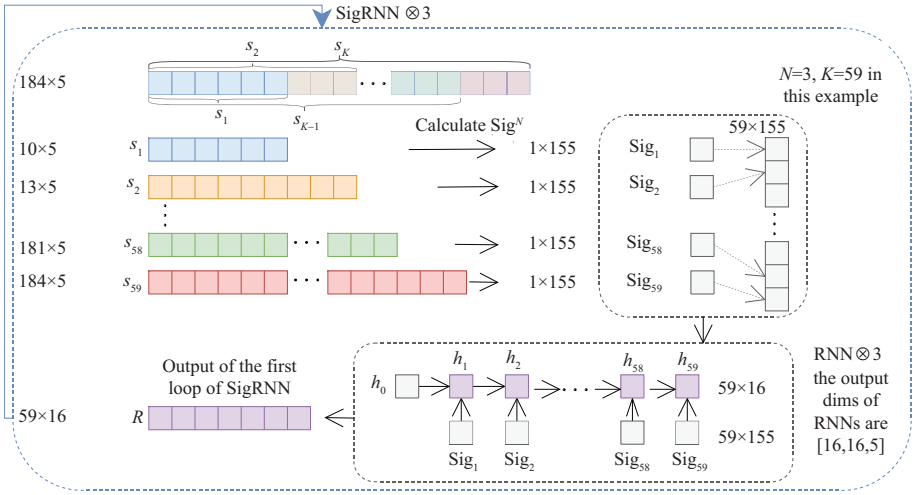

Figure 2 (Color online) Structure of SigRNN.

Algorithm 3 SigRNN($a_k^{\theta_1}$)

Input: A path $a_k^{\theta_1}$; signature depth N; hyperparameter of RNN;
Output: $R_k^{\theta_h}$ = the features of $a_k^{\theta_1}$;
1: **for** $i = 1$ to 3 **do**
2: Build stream of stream $s_k : s_k = \{s_{k,1}, s_{k,2}, \ldots, s_{k,K}\} = \text{Window}(a_k^{\theta_1})$;
3: Calculate N truncated path signature Sig_k^N of $s_k : \text{Sig}_k^N = \left\{\text{Sig}_{k,1}^N, \text{Sig}_{k,2}^N, \ldots, \text{Sig}_{k,K}^N\right\}$;
4: **for** $j = 1$ to 3 **do**
5: Extract features $R_k^{\theta_{i+1,j}} = f_{\text{rnn}}(\text{Sig}_k^N, \theta_{i+1,j})$;
6: **end for**
7: **end for**
8: Return $R_k^{\theta_4}$.

a channel count of $(d^{N+1}-1)/(N-1)$. The data in each channel lacks temporal relationships, indicating that Sig_k^N is no longer a time series. However, for the stream of streams, Sig_k^N maintains its stream-like nature, enabling the utilization of path signatures to extract features once more.

Finally, RNNs outperform CNNs in capturing temporal dependencies due to their recurrent connections, which maintain information across time steps. In contrast, CNNs lack sequential memory and are primarily designed for spatial analyses. Because an NTS typically lacks long-term dependency, a standard RNN integrated with path signatures is sufficient for feature extraction. Hence, we employ a three-layer RNN to further extract temporal features (refer to (9) and (10)).

$$h_{k,t} = \sigma\left(U^1 a_k^{\theta_1} + W^1 h_{k,t-1}\right), \tag{9}$$

$$R_k^{\theta_{2,1}} = f_{\text{rnn}}\left(a_k^{\theta_1}, \theta_{2,1}\right) = \sigma\left(V^1 h_{k,t}\right), \tag{10}$$

where $\theta_{2,1} = \left\{U^1, W^1, V^1\right\}$ represents the network parameters of the first hidden layer of the weight parameters θ_2 to be learned in the second group of RecurSig. The superscript denotes the network layer, and $\sigma()$ represents the activation function. By repeating (9) and (10) three times (lines 4 and 5 in Alogrithm 3), we acquire the features $R_k^{\theta_2}$ extracted from the first layer of SigRNN. We then feed the obtained feature sequence as a new path into SigRNN, repeating the above process three times, resulting in the final feature sequence $R_k^{\theta_4}$.

3.1.3 *Classification*

The final features extracted by SigRNN are passed to an FC layer to obtain the predicted values for different classes, as shown in

$$\text{out} = f_{\text{dense}}\left(R_k^{\theta_4}, \theta_5\right) = W R_k^{\theta_4} + b, \tag{11}$$

where $\theta_5 = W, b$ represents the parameters of the FC layer in the RecurSig network.

Let F^{θ_1} represent the process of Verbe, W denote the construction of the stream of streams, R^{θ_i} (for $i \in [2,4]$) represent the RNN layers, and f represents the fully connected layers. The entire process of RecuSig is represented by

$$[P_1, P_2, \ldots, P_n] = p_k \left(F^{\theta_1} \otimes W \otimes \mathrm{Sig}^N \otimes R^{\theta_2} \otimes \cdots \otimes W \otimes \mathrm{Sig}^N \otimes R^{\theta_4} \otimes f \right). \tag{12}$$

The utilized loss function is a multi-class cross-entropy loss function, accompanied by L2 regularization to prevent overfitting. Eq. (13) illustrates the multi-classification loss function.

$$L = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_i^k \log \left(\widehat{y}_i{}^k \right) + \frac{\lambda}{2N} \sum_{i=1}^{N} {w_i}^2, \tag{13}$$

where N is the total sample count, and K is the class count. The term y_i^k denotes the actual label of the ith sample and $\widehat{y}_i{}^k$ represents the predicted probability of the ith sample being classified as class k. Additionally, λ is the L2 regularization penalty coefficient, and w denotes the weight parameters.

3.2 Path signature-based explainable RecurSig

We integrate the path signature into DNN to construct an explainable NTS classification model called RecurSig. Insights are derived by analyzing the signature features extracted by SigRNN, thereby enhancing the understanding of the model's decision-making process. RecurSig's explainability can be summarized as follows.

Intuitiveness of path representation. We employ piecewise linear interpolation to transform discrete data streams into continuous paths. SigRNN transforms $a_k^{\theta_1}$ into paths that visually depict the evolutionary process of a time series. An intuitive explanation of the data is obtained by analyzing the geometric shapes (e.g., peaks and periodicity) and dynamic variations (e.g., slopes and inflection points) of the paths.

Explainability of feature vectors. SigRNN produces feature vectors that encapsulate the diverse characteristics of a time series. Each dimension corresponds to a specific path feature, such as statistical measures (e.g., mean and standard deviation) or descriptive features (e.g., trends and periodicity). Analyzing these vectors aids in identifying influential features and deepens our understanding of time-series classification.

Geometric explanation. For features $s_{k,1} = \{a_{k1}, a_{k2}, \ldots, a_{kw}\}$ in (5), we construct path A. Assume that path A is two-dimensional, denoted as $A : [T_1, T_w] \to \mathbb{R}^2$. According to (8), the 1st-order truncated signature of path A can be expressed as $[S(A)^1_{T_1,T_w}, S\left(A\right)^2_{T_1,T_w}] = \left[A^1_{T_w} - A^1_{T_1}, A^2_{T_w} - A^2_{T_1}\right]$. This implies that the 1st-order path signature of A represents the increments in A along the corresponding dimensions over interval $[T_1, T_w]$. This concept can also be extended to multi-dimensional paths. Building on (8), the 2nd-order truncated signature of A is defined in (14)–(17):

$$S(A)^{1,1}_{T_1,T_w} = \frac{1}{2!} \left(A^1_{T_w} - A^1_{T_1} \right)^2, \tag{14}$$

$$S(A)^{2,2}_{T_1,T_w} = \frac{1}{2!} \left(A^1_{T_w} - A^1_{T_1} \right)^2, \tag{15}$$

$$S(A)^{1,2}_{T_1,T_w} = \int_{T_1<t_2<T_w} \int_{T_1<t_1<T_w} \mathrm{d}A^{i_1}_{t_1}\, \mathrm{d}A^{i_2}_{t_2}, \tag{16}$$

$$S(A)^{2,1}_{T_1,T_w} = \int_{T_1<t_1<T_w} \int_{T_1<t_2<T_w} \mathrm{d}A^{i_2}_{t_1}\, \mathrm{d}A^{i_1}_{t_2}. \tag{17}$$

Referring to (14) and (15), the 2nd-order path signature of A along a specific dimension is proportional to the square of the increment in the dimension over the interval $[T_1, T_w]$. By referring to Figure 3, geometric understanding is enhanced. The solid black line represents path A, while the orange and blue shaded areas correspond to $S(A)^{1,2}_{T_1,T_w}$ and $S(A)^{2,1}_{T_1,T_w}$, respectively. The geometric explanation of (16) and (17) is further elaborated in [16].

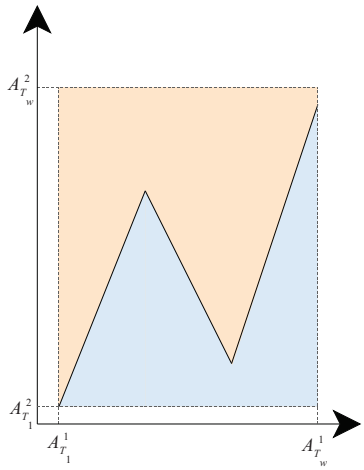

Figure 3 (Color online) Geometric meaning.

4 Experiment

The experiments are conducted on a computer with an Intel Core i9-9900K CPU, 32.00 GB of memory, and an Nvidia RTX 2080 GPU.

4.1 Datasets and experimental setup

Our experiments utilize six datasets: (1) 5G NTS: a comprehensive network intrusion detection dataset generated over a 5G wireless network (5G-NIDD) [31]. We conduct multiclass classification to evaluate RecurSig's efficiency in detecting the attack types. (2) Nonencrypted NTS: IP network traffic flows labeled with 75 applications (INTF) [33]. This dataset exhibits significant data imbalance issues. We select the top 53 applications based on the number of samples, in descending order, for the application identification task. (3), (4) Encrypted NTS: ISCX-VPN-NonVPN2016 (ISCX) [34]. Based on the labels of this dataset, we construct two sub-datasets. (a) The ISCX_application (ISCX_app) includes 15 categories and is used for application identification. (b) ISCX_traffic (ISCX_tra) consists of 10 categories and is used for traffic characterization. (5), (6) Two general time-series datasets: TSD1 [35] and TSD2 [36]. TSD1 comprises five class labels with a sample length of 187, enabling the evaluation of RecurSig's efficacy with shorter-length data samples. TSD2 contains six class labels with a dataset size of 10299 and nine channels. This enables us to evaluate RecurSig's performance with small-scale multichannel time series data. These datasets facilitate the evaluation of RecurSig across different data distributions, sizes, and feature dimensions.

The following metrics are used to evaluate the performance of RecurSig: per-class precision (Pre), per-class F1-score recall (Rec), per-class F1-score (F1), overall accuracy (ACC), macro-averaged precision (MP), macro-averaged recall (MR), and macro-averaged F1-score (MF1) [17].

Based on previous benchmarks, we choose 16 as the number of hidden units in RNN layers. A SigRNN block comprises three layers, with the first two having multiples of 16 hidden units, and the last aligned with the Verbe-enhanced channel count. This setup ensures standardized input dimensions across SigRNN blocks for uniform processing. For example, the hyperparameters are set to $((16, 16, 5) \otimes 3)$ for TSD1. The configuration is refined through iterative experiments. It optimally aligns the model performance with computational resources. Other crucial hyperparameters for TSD1 include Verbe's CNN hidden units set at $(16, 16, 3)$ and SigRNN's w and al configured to 2 and 1, respectively.

4.2 Experiment 1: classification performance comparison of RecurSig with six baseline models

RecurSig is compared with six baseline models, including three DL models: extended byte segment neural network (EBSNN) [7], deep packet based on CNN (DP-CNN) [6], 1D-CNN [37] and three ML models: K-nearest neighbor (KNN) [38], C4.5 decision tree (C4.5) [39], and RF [40].

The datasets are split into 80% training and 20% testing sets, with 20% of the training set allocated for validation. The batch size is 256 for TSD1 and 64 for the other datasets. We use the Adam optimizer with

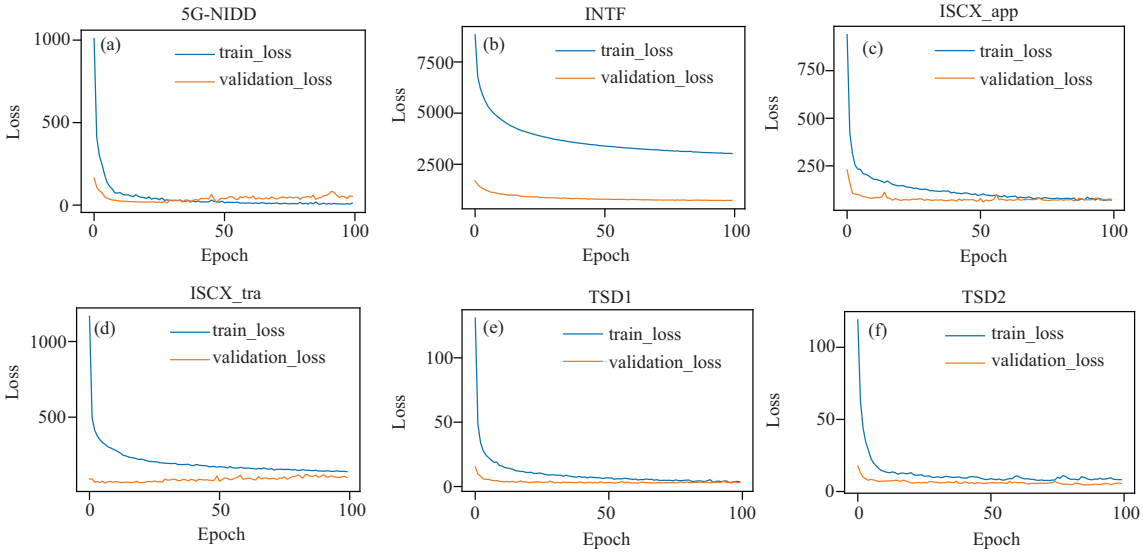

Figure 4 (Color online) Display the training loss and validation loss of RecurSig on (a) 5G-NIDD, (b) INTF, (c) ISCX_app, (d) ISCX_tra, (e) TSD1, and (f) TSD2, respectively.

Table 3 Comparison of the performance of seven models across six datasets

Model	5G-NIDD		INTF		ISCX_app	
	MP/MR/MF1	ACC	MP/MR/MF1	ACC	MP/MR/MF1	ACC
EBSNN [7]	99.79/99.78/99.78	99.78	72.09/66.53/67.40	71.00	66.72/89.95/68.61	90.35
DP-CNN [6]	99.72/99.72/99.72	99.72	67.13/64.10/65.54	72.70	**67.73**/84.22/68.85	89.38
1D-CNN [37]	96.27/95.46/95.52	95.53	72.03/65.86/67.58	71.91	67.35/88.92/68.67	89.36
KNN [38]	98.97/99.11/98.97	99.48	75.42/60.84/65.55	71.51	39.12/47.13/24.22	35.20
C4.5 [39]	97.75/98.39/98.01	99.17	**84.59**/56.99/63.31	63.32	62.51/75.33/60.79	72.64
RF [40]	99.20/98.92/99.16	99.48	70.18/57.06/56.03	56.69	66.91/92.06/67.12	87.37
RecurSig	**99.86**/**99.86**/**99.86**	**99.86**	71.64/**66.86**/**68.28**	**74.20**	67.10/**92.32**/**70.86**	**92.28**

Model	ISCX_tra		TSD1		TSD2	
	MP/MR/MF1	ACC	MP/MR/MF1	ACC	MP/MR/MF1	ACC
EBSNN [7]	69.30/**94.92**/71.27	91.61	78.32/92.41/83.65	96.39	90.89/90.77/90.77	90.86
DP-CNN [6]	74.18/92.30/77.68	92.85	86.08/91.34/88.47	98.03	91.98/91.99/91.93	91.78
1D-CNN [37]	70.38/91.48/74.80	92.88	88.92/92.89/90.75	98.44	89.98/89.65/89.71	89.23
KNN [38]	39.82/54.16/30.36	59.82	93.51/79.53/85.33	97.20	80.86/63.12/63.74	65.05
C4.5 [39]	69.06/85.29/68.57	86.15	78.54/78.03/78.28	95.23	71.57/71.51/71.50	72.38
RF [40]	64.67/87.54/67.39	81.11	**96.36**/81.33/87.44	97.56	84.45/84.48/84.41	84.59
RecurSig	**77.63**/93.53/**80.28**	**93.53**	92.30/**93.64**/**92.96**	**98.85**	**93.58**/**93.33**/**93.37**	**93.48**

a learning rate of and 0.001 a weight decay of 0.001. RecurSig's performance stabilized before reaching 100 epochs; hence, we train all models for 100 epochs over five rounds to ensure fair performance comparison. The best model is selected based on MF1. Figures 4(a)–(f) illustrate the training and validation losses of RecurSig across six datasets, respectively. RecurSig exhibits stable and rapid convergence, typically within 20 epochs. However, there is a slight upward trend in the validation loss for 5G-NIDD (see Figure 4(a)) and ISCX_app (see Figure 4(c)), possibly due to the dataset's more complex features, leading to decreased generalization. To mitigate this, a straightforward strategy is to implement early termination during training. Future work will explore advanced data augmentation and ensemble methods to further enhance generalization. Overall, our model exhibits strong and consistent performance across various datasets, indicating its robustness against overfitting.

Table 3 [6,7,37–40] presents a comparison of RecurSig with the six baseline models for the six datasets. The best results are highlighted in bold. RecurSig exhibits stable and excellent classification performance owing to its powerful feature-extraction module. It achieves the highest accuracy across all datasets, demonstrating its effectiveness. RecurSig outperforms the other models in terms of MP, MR, and MF1 values, except for slight differences in a few datasets. Notably, RecurSig's MF1 value surpasses those

Table 4 Comparison results of four DL models on the ISCX_app dataset

Lable	Support	EBSNN [7] Pre/Rec/F1	DP-CNN [6] Pre/Rec/F1	1D-CNN [37] Pre/Rec/F1	RecurSig Pre/Rec/F1
AIM Chat	2	2.20/100.00/4.30	1.79/50.00/3.45	1.45/50.00/2.82	**3.70**/**100.00**/**7.14**
Email	69	7.25/78.26/13.27	5.15/76.81/9.65	5.17/79.71/9.71	**14.86**/**79.71**/**25.06**
Facebook	2715	**91.54**/83.24/87.19	97.17/77.27/86.09	88.63/82.73/85.58	85.41/**90.79**/**88.02**
FTPS	4065	99.95/99.61/99.78	99.93/99.66/99.79	99.95/99.83/99.89	**99.98**/**99.85**/**99.91**
Gmail	12	9.01/83.33/16.26	13.51/83.33/23.26	15.63/83.33/26.32	**17.14**/**100.00**/**29.27**
Hangouts	3891	98.63/87.25/92.60	**99.47**/86.35/92.45	99.35/86.33/92.38	98.04/**87.61**/**92.54**
ICQ	5	2.83/60.00/5.41	1.26/40.00/2.44	4.81/**100**/9.17	**5.77**/60.00/**10.53**
Netflix	159	**100.00**/98.73/**99.37**	96.89/98.11/97.50	96.88/97.48/97.18	97.52/**98.74**/98.13
SCP	410	85.78/92.68/89.10	97.42/91.95/94.60	**98.18**/92.20/**95.09**	73.94/**93.41**/82.54
SFTP	662	99.09/98.64/98.86	98.64/98.94/98.79	97.91/99.09/98.50	**98.25**/**99.70**/**99.47**
Skype	3772	93.45/85.07/89.06	90.36/86.00/88.13	96.37/81.73/88.45	**96.65**/**87.12**/**91.63**
Spotify	22	35.71/90.91/51.28	40.91/81.92/54.55	25.97/90.91/40.40	**38.89**/**95.45**/**55.26**
Vimeo	97	94.94/97.93/97.44	97.93/97.93/97.93	**100**/96.91/**98.43**	91.26/**96.91**/94.00
Voipbuster	860	97.53/96.40/96.96	97.53/96.51/97.02	96.62/96.28/96.45	**98.00**/**96.86**/**97.43**
Youtube	144	80.93/97.22/88.33	78.02/98.61/87.12	83.33/97.22/89.74	**86.06**/**98.61**/**91.91**
Overall metric	ACC	90.35	89.38	89.36	**92.28**
	MP/MR/MF1	66.72/89.95/68.61	**67.73**/84.22/68.85	67.35/88.92/68.67	67.10/**92.32**/**70.86**

of the best baseline models by 0.08%, 0.7%, 2.01%, 2.6%, 2.21%, and 1.44% for the six datasets. This indicates that RecurSig strikes a favorable balance between precision and recall, resulting in the best overall performance.

Table 4 [6, 7, 37] evaluates four DL models (EBSNN, DP-CNN, 1D-CNN, and RecurSig) on ISCX_app for each class, comparing their handling of class imbalance. RecurSig outperforms most categories, even in sparsely sampled classes, such as AIM Chat, Gmail, and ICQ, with F1 score improvements of 2.84%, 2.95%, and 1.36%, respectively. This indicates its feature extraction strength from limited data, thus mitigating class imbalance issues. The unique nature of the path signatures contributes to this capability. Nonetheless, RecurSig's F1 scores in certain categories remain suboptimal, which could lead to limitations in practical applications. This suggests there is room for further improvement. However, RecurSig shows the highest overall accuracy and MF1 score, underscoring its robustness across diverse classes.

Figures 5(a)–(f) present the ACC and MF1 scores of the seven models for the six datasets. RecurSig achieves the highest score in both ACC and MF1. It demonstrates smaller gaps in ACC and MF1 compared with the other models, indicating its strong performance in handling imbalanced datasets. Three ML models (KNN, C4.5, and RandomForest) perform well on the INTF dataset (see Figure 5(b)) but falter on others. This may be because non-encrypted NTS data have more distinct features on which these models rely. In contrast, RecurSig excels at handling high-dimensional complex data and capturing nonlinear relationships. Consequently, RecurSig exhibits stable classification performance across all datasets.

Figures 6(a)–(c) depict the training loss reduction for four DL models on various datasets. All models converge within 20 epochs, with RecurSig achieving the lowest convergence loss on ISCX_app (see Figure 6(b)) and TSD1 (see Figure 6(c)). This indicates its strong convergence and fitting capabilities, providing support for handling complex NTS. Moreover, we assess the computational complexity of the four models by measuring GPU memory usage per training epoch, as detailed in Table 5 [6, 7, 37]. The memory usage is measured in megabytes (MB). RecurSig demonstrates the lowest memory consumption on the 5G-NIDD and INTF datasets at 3.96 and 4.53 MB, respectively. Although not equal to EBSNN, RecurSig is markedly more efficient than the DP-CNN and 1D-CNN, particularly on larger datasets such as ISCX_tra and ISCX_app. These results highlight RecurSig's memory efficiency, making it suitable for resource-constrained environments and scalable for practical applications.

4.3 Experiment 2: ablation experiment

Ablation studies assess the Verbe module and path signatures. We compare four models: (1) multilayer perceptron (MLP) as the baseline; (2) SigMLP with path signature input into MLP; (3) VerMLP with Verbe-enhanced data input into MLP; and (4) VerSigMLP with path signatures from Verbe-enhanced

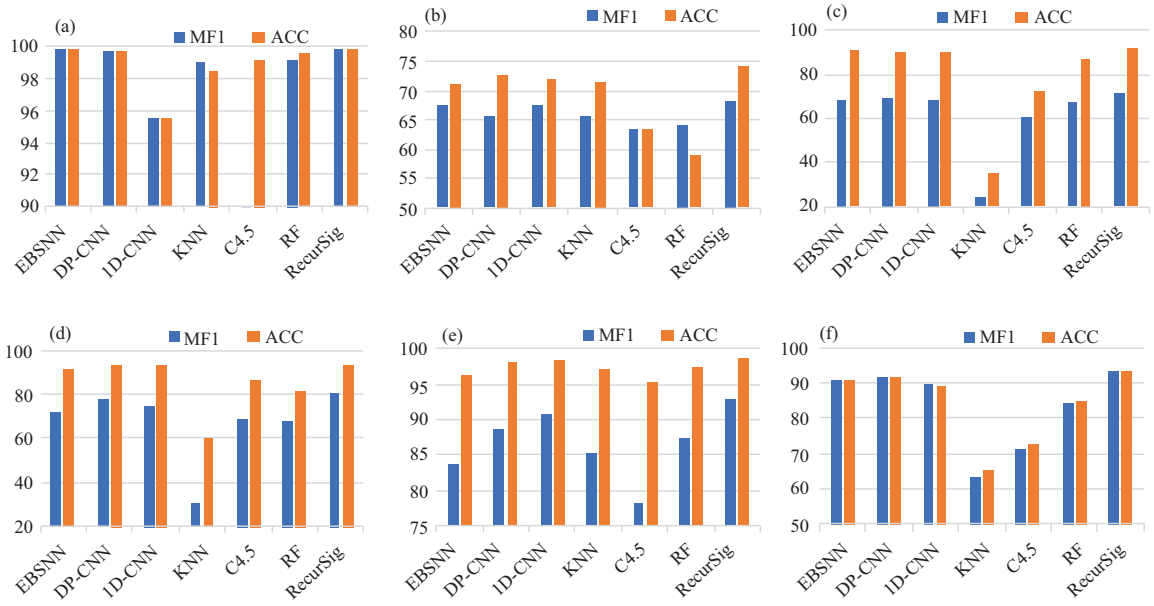

Figure 5 (Color online) Overall ACC and MF1 of the seven models across six datasets. (a) 5G-NIDD; (b) INTF; (c) ISCX_app; (d) ISCX_tra; (e) TSD1; (f) TSD2.

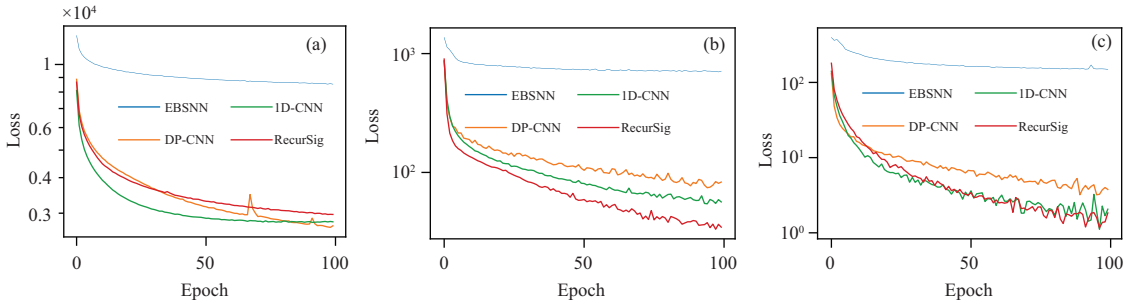

Figure 6 (Color online) Training loss reduction process of the four DL models on three datasets. (a) INTF (non-encrypted NTS); (b) ISCX_app (encrypted NTS); (c) TSD1 (non-encrypted general time series).

Table 5 GPU memory usage per epoch for four models across six datasets during training

	5G-NIDD	INTF	ISCX_app	ISCX_tra	TSD1	TSD2
EBSNN [7]	7.63	**12.79**	**9.24**	**7.63**	8.02	**11.81**
DP-CNN [6]	6.96	167.05	16.99	6.96	11.54	174.92
1D-CNN [37]	6.26	174.87	15.98	25.26	11.15	180.84
RecurSig	**3.96**	32.92	24.70	22.87	**4.53**	19.63

data input into MLP. Figures 7(a) and (b) show ACC and MF1 values across the five datasets. SigMLP underperformed, indicating that the direct use of path signatures as inputs disrupts the temporal nature of the data, leading to inferior feature extraction compared with using the original signals. VerMLP outperforms MLP in both ACC and MF1 on INTF, TSD1, and TSD2, thus demonstrating Verbe's enhancement capabilities. However, its performance decreases for ISCX_app and ISCX_tra. The longer sample lengths of these datasets may allow FC layers to extract more comprehensive features, thereby diminishing Verbe's relative advantages. Conversely, VerSigMLP achieves the best performance, confirming the effectiveness of Verbe for data augmentation and path-signature-based feature extraction.

Experiments are conducted to evaluate the effectiveness of different RNN architectures in the SigRNN module: CNN (SigCNN), gate-recurrent unit (SigGRU), and long short-term memory (SigLSTM). Figures 8(a) and (b) show the ACC and MF1 values for the various datasets. All RNN variants surpass the CNN, demonstrating the RNN's superior time-series feature extraction. SigRNN leads in performance on INTF, TSD1, and TSD2. SigGRU slightly outperforms SigRNN on datasets with longer sample lengths

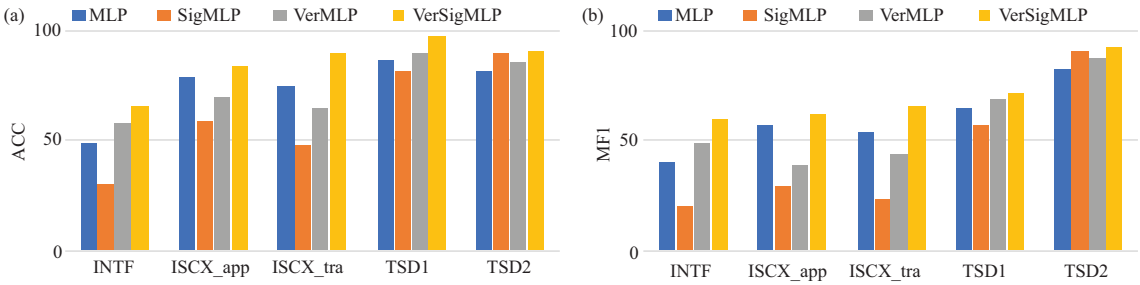

Figure 7 (Color online) Ablation experiments are conducted on Verbe and path signatures in RecurSig. The pictures depict the ACC and MF1 of the four models across five datasets respectively. (a) ACC; (b) MF1.

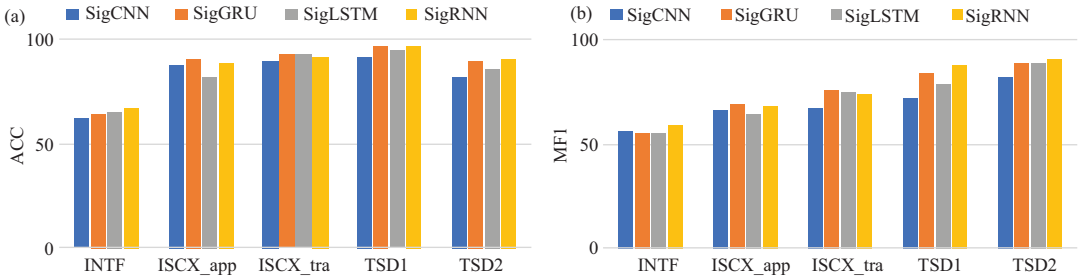

Figure 8 (Color online) Comparative results of different RNNs in the SigRNN module. The pictures depict the ACC and MF1 of the four models across five datasets respectively. (a) ACC; (b) MF1.

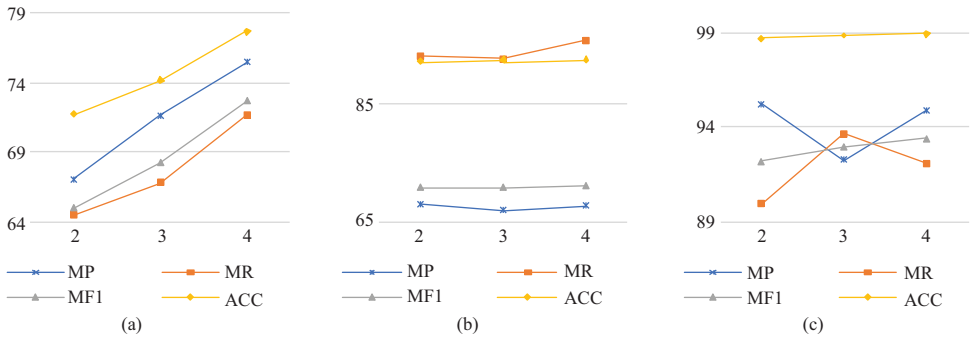

Figure 9 (Color online) Effect of truncated order on RecurSig. (a) INTF; (b) ISCX_app; (c) TSD1.

(ISCX_app and ISCX_tra), hinting at GRU's advantage in long-term dependency capture. Therefore, replacing RNN with GRU may yield improved results for certain datasets.

4.4 Experiment 3: performance comparison of Recursig within various truncated orders

The truncated order in RecurSig influences the captured information. We test orders 2–4 on six datasets, and Figures 9(a)–(c) show the results for INTF, ISCX_app, and TSD1. Increasing the truncated order enhanced ACC and MF1, suggesting that higher truncation orders provide richer feature information. However, it also increases the computational complexity. Table 6 lists the computational complexities of the models with different truncated orders for ISCX_tra and TSD2. The evaluation metrics include the average training time and memory usage per epoch, as well as parameter numbers. All increased exponentially with a truncated order. High computational complexity may affect practical applications and generalization. Thus, it is crucial to balance performance and efficiency. A suitable strategy involves setting a threshold for minimal accuracy gains to determine the optimal tradeoff.

4.5 Experiment 4: explainability analysis

We apply XAI and the Shapley additive explanations (SHAP) to visualize RecurSig's decision-making process. Figure 10 highlights the eight key features of INTF. Destination.IP, Flow.Duration, and PSH. Flag.Count exhibits larger widths and longer horizontal lengths, indicating their positive contribution

Table 6 Computational complexity of RecurSig with various truncated orders

	ISCX_tra			TSD2		
Truncated orders	2	3	4	2	3	4
Training time (s)	473.81	691.84	1169.26	11.98	15.45	59.08
Number of parameters	304497	731121	4012401	13603	60711	623187
Memory usage (MB)	12.1	32.92	191.21	3.93	24.7	260.62

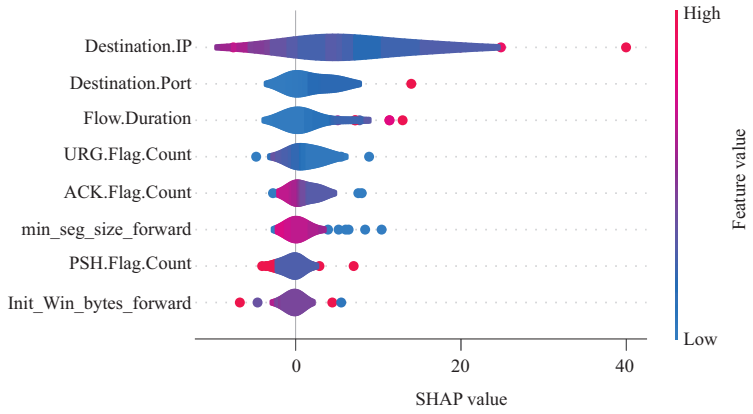

Figure 10 (Color online) SHAP summary plot of RecurSig on the INTF dataset.

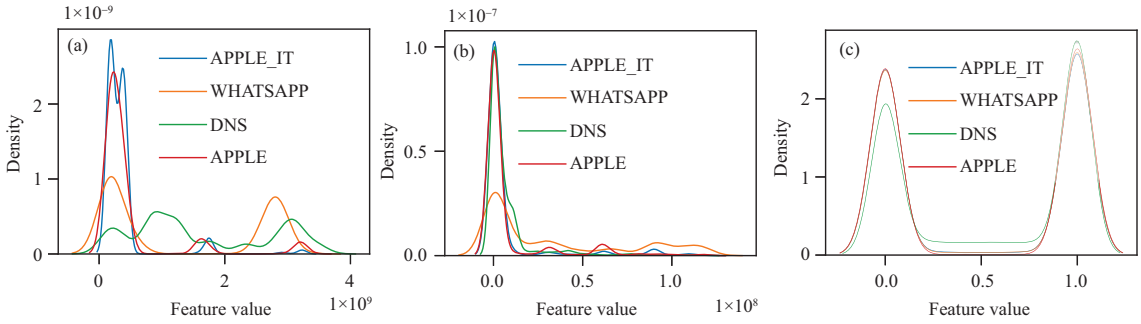

Figure 11 (Color online) Distribution of different categories on (a) Destination.IP, (b) Flow.Duration, and (c) PSH.Flag.Count.

to the classification. PSH.Flag.Count, which represents the number of packets utilizing the PSH flag, negatively impacts the results. These observations match real-world NTS classification insights.

To understand this phenomenon, we analyze the distributions of Destination.IP, Flow.Duration, and PSH.Flag.Count the across four categories, as shown in Figures 11(a)–(c). The distributions of Destination.IP (see Figure 11(a)) and Flow.Duration (see Figure 11(b)) exhibit clear distinctions among labels. This indicates their contribution to label differentiation. This aligns with the human understanding of NTS classification, as the destination address and flow duration are crucial factors for identifying different applications. The similar distribution of PSH.Flag.Count (see Figure 11(c)) explains its negative impact on the classification results. Our analysis confirms that RecurSig offers robust interpretability. The alignment of model behavior with domain knowledge not only validates its practical utility but also enhances users' trust in the decision-making processes.

4.6 Case study: applications of RecurSig in 5G and beyond networks

RecurSig enhances security in NASs for 5G and beyond, as illustrated in Figure 12. It conducts intrusion detection for the input NTS and activates the NAS firewall to filter out malicious traffic and block attacks. RecurSig offers explainable outputs with high accuracy, bolstering network operator trust in NAS. Understanding its classification and response mechanisms will encourage broader NAS adoption, accelerating telecom network advancements. RecurSig can also enable real-time remote healthcare through 5G's speed and low latency, as shown in Figure 13. It provides initial diagnoses with high accuracy and explainability, boosting physicians' confidence and enabling personalized diagnosis.

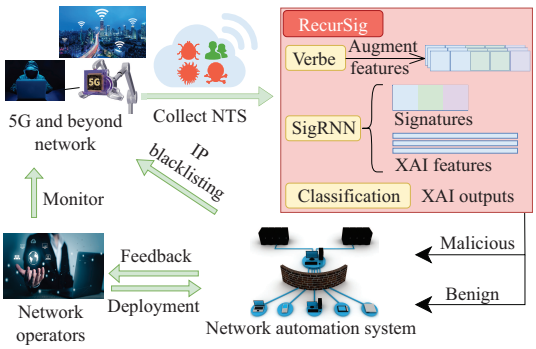

Figure 12 (Color online) Automatic network security management.

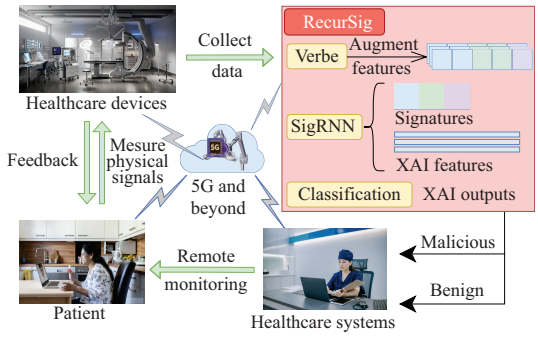

Figure 13 (Color online) Telemedicine diagnosis.

5 Conclusion

This study advances cybersecurity and network automation management for 5G networks and beyond by introducing RecurSig, a path-signature-based XAI-enabled network time-series classification model. It merges DL with path-signature techniques to enhance identification efficiency and explainability. This addresses the information loss due to truncated signatures. Additionally, it captures explainable spatiotemporal features and enhances the model accuracy. Our experiments confirm RecurSig's competitive edge, addressing the societal demand for transparent and reliable AI solutions in a rapidly evolving network landscape. This represents a step forward in the application of XAI to critical domains. Although RecurSig mitigates class imbalance to some extent, its classification accuracy has not yet reached an ideal level. Looking ahead, we plan to enhance RecurSig by implementing more robust data augmentation techniques and integrating ensemble learning methods, such as boosting or bagging. Additionally, considering the real-time requirements of NAS in the 5G era and beyond, we aim to focus on improving the computational efficiency of path signatures, paving the way for more advanced network management applications.

Acknowledgements This work was partially supported by National Natural Science Foundation of China (Grant Nos. 61702274, 62072249).

References

1 Rodriguez E, Otero B, Gutierrez N, et al. A survey of deep learning techniques for cybersecurity in mobile networks. IEEE Commun Surv Tut, 2021, 23: 1920–1955
2 Zhao Y, Xu K, Li Q, et al. Intelligent networking in adversarial environment: challenges and opportunities. Sci China Inf Sci, 2022, 65: 170301
3 Lacava A, Polese M, Sivaraj R, et al. Programmable and customized intelligence for traffic steering in 5G networks using open RAN architectures. IEEE Trans Mobile Comput, 2024, 23: 2882–2897
4 Zhao J Q, Zhu H, Wang F W, et al. ACCEL: an efficient and privacy-preserving federated logistic regression scheme over vertically partitioned data. Sci China Inf Sci, 2022, 65: 170307
5 Wang Z, Hu J, Min G, et al. Spatial-temporal cellular traffic prediction for 5G and beyond: a graph neural networks-based approach. IEEE Trans Ind Inf, 2023, 19: 5722–5731
6 Lotfollahi M, Siavoshani M J, Hossein Zade R S, et al. Deep packet: a novel approach for encrypted traffic classification using deep learning. Soft Comput, 2020, 24: 1999–2012
7 Xiao X, Xiao W, Li R, et al. EBSNN: extended byte segment neural network for network traffic classification. IEEE Trans Dependable Secure Comput, 2021, 19: 3521–3538
8 Setzu M, Guidotti R, Monreale A, et al. GLocalX — from local to global explanations of black box AI models. Artif Intell, 2021, 294: 103457
9 Andresini G, Appice A, Caforio F P, et al. ROULETTE: a neural attention multi-output model for explainable network intrusion detection. Expert Syst Appl, 2022, 201: 117144
10 Crabbé J, van der Schaar M. Concept activation regions: a generalized framework for concept-based explanations. In: Proceedings of the 36th International Conference on Neural Information Processing Systems, 2022. 2590–2607
11 Li L, Zhang Y, Chen L. Personalized prompt learning for explainable recommendation. ACM Trans Inf Syst, 2023, 41: 1–26
12 Han D, Wang Z, Zhong Y, et al. Evaluating and improving adversarial robustness of machine learning-based network intrusion detectors. IEEE J Sel Areas Commun, 2021, 39: 2632–2647
13 Zebin T, Rezvy S, Luo Y. An explainable AI-based intrusion detection system for DNS over HTTPS (DoH) attacks. IEEE Trans Inform Forensic Secur, 2022, 17: 2339–2349
14 Cui J, Ma K, Sun Y, et al. Deep explainable method for encrypted traffic classification. J Comput Appl, 2023, 43: 1151
15 Morrill J, Salvi C, Kidger P, et al. Neural rough differential equations for long time series. In: Proceedings of the 38th International Conference on Machine Learning, 2021. 7829–7838
16 Fermanian A. Learning time-dependent data with the signature transform. Dissertation for Ph.D. Degree. Paris: Sorbonne Université, 2021

17 Lin X, Xiong G, Gou G, et al. Et-bert: a contextualized datagram representation with pre-training transformers for encrypted traffic classification. In: Proceedings of the ACM Web Conference, 2022. 633–642
18 Towhid M S, Shahriar N. Encrypted network traffic classification using self-supervised learning. In: Proceedings of the 8th International Conference on Network Softwarization, Piscataway, 2022. 366–374
19 Gu R, Wang G, Song T, et al. CA-Net: comprehensive attention convolutional neural networks for explainable medical image segmentation. IEEE Trans Med Imag, 2020, 40: 699–711
20 Tan Q, Ye M, Ma A J, et al. Explainable uncertainty-aware convolutional recurrent neural network for irregular medical time series. IEEE Trans Neural Netw Learn Syst, 2020, 32: 4665–4679
21 Selvaraju R R, Cogswell M, Das A, et al. Grad-CAM: visual explanations from deep networks via gradient-based localization. In: International Conference on Computer Vision, Venice, 2017. 618–626
22 Kim B, Wattenberg M, Gilmer J, et al. Interpretability beyond feature attribution: quantitative testing with concept activation vectors (TCAV). In: Proceedings of the 35th International Conference on Machine Learning, Cambridge, 2018. 2668–2677
23 Taylor V F, Spolaor R, Conti M, et al. Robust smartphone APP identification via encrypted network traffic analysis. IEEE Trans Inform Forensic Secur, 2017, 13: 63–78
24 Hassan M M, Gumaei A, Alsanad A, et al. A hybrid deep learning model for efficient intrusion detection in big data environment. Inf Sci, 2020, 513: 386–396
25 Fiandrino C, Attanasio G, Fiore M, et al. Toward native explainable and robust AI in 6G networks: current state, challenges and road ahead. Comput Commun, 2022, 193: 47–52
26 Sun L, Li C, Liu B, et al. Class-driven graph attention network for multi-label time series classification in mobile health digital twins. IEEE J Sel Areas Commun, 2023, 41: 3267–3278
27 Bi X, Zhang C, He Y, et al. Explainable time-frequency convolutional neural network for microseismic waveform classification. Inf Sci, 2021, 546: 883–896
28 Hsieh T Y, Wang S, Sun Y, et al. Explainable multivariate time-series classification: a deep neural network which learns to attend to important variables as well as time intervals. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining, New York, 2021. 607–615
29 Kim J Y, Cho S B. Electric energy consumption prediction by deep learning with state explainable autoencoder. Energies, 2019, 12: 739
30 Sun L, Liang J, Zhang C, et al. Meta-transfer metric learning for time series classification in 6G-supported intelligent transportation systems. IEEE Trans Intell Transp Syst, 2024, 25: 2757–2767
31 Samarakoon S, Siriwardhana Y, Porambage P, et al. 5G-NIDD: a comprehensive network intrusion detection dataset generated over 5G wireless network. 2022. ArXiv:221201298
32 Mekruksavanich S, Jitpattanakul A. A multichannel CNN-LSTM network for daily activity recognition using smartwatch sensor data. In: Proceedings of Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunication Engineering, Cha-am, 2021. 277–280
33 Rojas J S, Gallón á R, Corrales J C. Personalized service degradation policies on OTT applications based on the consumption behavior of users. In: Proceedings of Computational Science and its Applications-ICCSA, Berlin, 2018. 543–557
34 Draper-Gil G, Lashkari A H, Mamun M S I, et al. Characterization of encrypted and VPN traffic using time-related. In: Proceedings of the 2nd International Conference on Information Systems Security and Privacy, Portugal, 2016. 407–414
35 Goldberger A L, Amaral L A N, Glass L, et al. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. Circulation, 2000, 101: E215–E220
36 Anguita D, Ghio A, Oneto L, et al. A public domain dataset for human activity recognition using smartphones. In: Proceedings of the 21st European Symposium on Artificial Neural Networks, Bruges, 2013
37 Wang W, Zhu M, Wang J, et al. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In: Proceedings of IEEE International Conference on Intelligence and Security Informatics, Piscataway, 2017. 43–48
38 Zhang S, Li J, Li Y. Reachable distance function for KNN classification. IEEE Trans Knowl Data Eng, 2022, 35: 1–15
39 Moral-García S, Mantas C J, Castellano J G, et al. Using credal C4.5 for calibrated label ranking in multi-label classification. Int J Approx Reason, 2022, 147: 60–77
40 Chen H, Wu L, Chen J, et al. A comparative study of automated legal text classification using random forests and deep learning. Inf Process Manage, 2022, 59: 102798