IEEE ComSoc | IEEE COMPUTER SOCIETY | IEEE Signal Processing Society | VTS | IEEE Transactions on Machine Learning in Communications and Networking

Received 15 December 2022; revised 20 May 2023; accepted 8 October 2023. Date of publication 11 October 2023; date of current version 2 November 2023.

The associate editor coordinating the review of this article and approving it for publication was E. Bou-Harb.

Digital Object Identifier 10.1109/TMLCN.2023.3323915

WFF-EGNN: Encrypted Traffic Classification Based on Weaved Flow Fragment via Ensemble Graph Neural Networks

ZIHAN CHEN[1,2,3] (Member, IEEE), GUANG CHENG[1,2,3] (Member, IEEE), DANDAN NIU[1,2] (Student Member, IEEE), XING QIU[1,2,3] (Student Member, IEEE), YUYU ZHAO[1,2,3] (Member, IEEE), AND YUYANG ZHOU[1,2,3] (Member, IEEE)

[1]School of Cyber Science and Engineering, Southeast University, Nanjing 210096, China
[2]Jiangsu Province Engineering Research Center of Security for Ubiquitous Network, Nanjing 211189, China
[3]Purple Mountain Laboratories, Nanjing 211111, China

CORRESPONDING AUTHOR: G. CHENG (chengguang@seu.edu.cn)

This work was supported in part by the General Program of the National Natural Science Foundation of China under Grant 62172093 and in part by the National Natural Science Foundation of China under Grant 62202097.

ABSTRACT Traffic analysis plays an essential role in network management and security protection under the premise of fully protecting user privacy. Unfortunately, encryption dramatically reduces the disclosure of traffic information, making encrypted traffic analysis more challenging than plaintext traffic analysis, especially in the environment of new encryption protocols (e.g., TLS-1.3, QUIC). The existing tensor-based methods mainly focus on optimizing packet length sequence features and introducing the latest deep learning model. However, the tensor-based features cannot sufficiently express the structured non-Euclidean Markov properties inside the encrypted traffic. This paper proposes a novel traffic graphical expression model named Weaved Flow Fragment (WFF) to transform a packet sequence into a graph, which better represents the packet sequence's inner relationship than the tensor. WFF also considers the co-evolution relationship and the cross-direction change relationship in the bidirectional flow, breaking through the limitation that the tensor-like length sequence only considers the adjacent Markov properties. Then, we use the latest graph convolutional networks, gated graph neuron networks, and capsule graph neural networks to implement classification based on WFF, respectively. Further, to give full play to the advantages of different graph neural network classifiers to improve classification effect in large-scale data scenarios, we proposed the ensemble graph neural network architecture with several ensemble mechanisms to reduce the possibility of classification error caused by overfitting and model concerns. Experiments show that our classification effect is much better than the state-of-the-art methods (achieved 99.25% F1-score) in an open-world environment, and the model size is reduced by 99.1%.

INDEX TERMS Encrypted traffic classification, open-world environment, graph neural networks, ensemble graph deep learning, weaved flow fragment.

I. INTRODUCTION

CURRENTLY, most network applications are Over-The-Top (OTT) applications developed by various companies. For the third party, mastering the end data or server logs of such applications is impossible. However, as an essential medium for data transmission, network traffic can be legally and transparently accessed with privacy protection. Therefore, traffic analysis has become important in network management and security protection. Internet Service Providers (ISPs) can achieve effective network management on backbone networks through traffic analysis, which is representative of large-scale high-definition live streaming and video traffic Quality of Experience (QoE) guarantee. At the same time, many enterprises have complex network structures, and some have servers all over the world. Their Network Security Departments (NSDs) need to classify

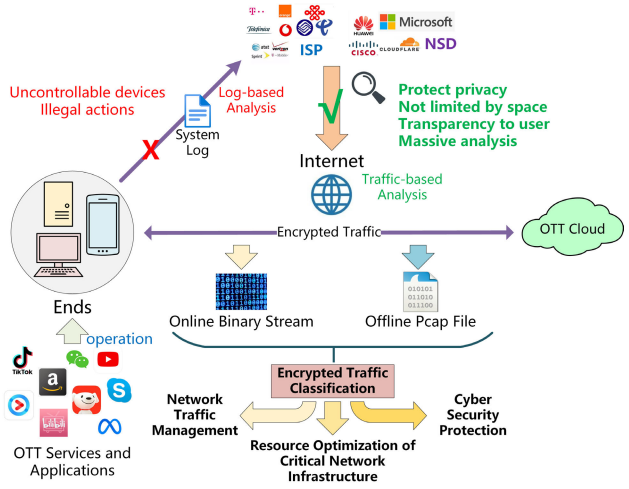


FIGURE 1. **Encrypted network traffic classification overview.**

network traffic to block it before or when a threat occurs or to check the compliance of network traffic. However, according to Google's Transparency Report [1], for November 2022, almost all traffic in Google apps and services is encrypted. Furthermore, Chrome's web traffic encryption ratio is also as high as 97%. It is challenging to analyze the encrypted traffic by traditional plaintext matching-based methods such as Deep Packet Inspection (DPI), so encrypted traffic classification has become an important research field. An overview of the meaning and process of encrypted traffic classification is shown in Figure 1.

Initially, traditional machine learning methods were used in encrypted traffic classification based on packet interval and flow statistics features [2]. However, its accuracy is not satisfactory. Subsequently, ensemble learning led by Random Forest (RF) [3] is also used, continuing the use of statistical features. However, the classification accuracy of these methods relies heavily on feature engineering, which requires much prior expert knowledge. Moreover, the huge size of the model and the potential possibility of easy overfitting make it challenging to apply the ensemble learning method to the open-world actual network environment. Deep learning is widely used because of its strong feature expression ability and end-to-end learning specialty [4]. However, some studies [5] show that the features obtained by automatic feature selection are not optimal in classification because the model does not effectively mine many hidden high-dimensional features. Therefore, some researchers tend to further use feature engineering methods for feature optimization within the framework of deep learning [6]. The sequential features, especially the length sequence features, based on the packet as the basic unit in encrypted traffic, are deeply explored.

Existing encryption traffic classification methods based on the features of length sequence are mainly expressed in the form of floating point number or integer tensor [7], or by embedding [6] the length sequence is converted into a larger scale $0-1$ tensor to explore its expressiveness. However, in the case of large-scale data, the effect of the method is still not satisfactory, and there are some confusion problems among similar categories. The main reason is that the packet sequence or the higher protocol level Protocol Data Unit (PDU) sequence [5] is a structured but non-European space data sequence [8], and the relationship between each unit is not a simple preorder or cross-direction relationship. Even in the packet length sequence, the length values are rich in various Markov properties. In order to deal with this problem, the current research [9] began to use the form of a graph to describe the complex relationships in the packet length sequence and use the Graph Neural Network (GNN) for classification. However, the lack of in-depth study on the Markov property of the packet length sequence in the existing research makes the graphical expression method (the extraction of features to form a graph) uninterpretable, and the classification effect, to some extent, is not even as satisfying as that of some tensor-based methods.

In order to solve the above problems, starting from the transmission mode of encrypted traffic, we deeply study the relationship between each packet and the feature of the encrypted traffic sequence, especially the packet length sequence, and summarize it into several representative Markov relationship paradigms. On this basis, we propose Weaved Flow Fragment (WFF), a novel flow graphical expression model aiming at the graphic of length sequence features. It consists of a set of interpretable graphical expression paths based on the Markov relationship paradigms of encrypted traffic packet sequence, which forms a graph of several packet length sequences in an encrypted flow. Then, we analyze the advantages of basic graph neural network models in expressing encrypted traffic sequence features. Three heterogeneous GNN models, WFF-based Graph Convolutional Networks (WFF-GCN), WFF-based Gated GNN (WFF-GGNN), and WFF-based Capsule Graph Neural Networks (WFF-CapsGNN), are designed to classify encrypted traffic. Further, to exploit the advantages of different graph neural network models, we propose an Ensemble GNN (EGNN) architecture. In EGNN, we used two ensemble mechanisms: voting and stacking. In addition, president mode and Category Cross-Correlation Weighted Voting (C3WV) mode are proposed. GNN participants in EGNN also used the traditional hard and soft modes. EGNN effectively handles the misclassification problem and realizes high-precision encrypted traffic classification.

It is worth noting that using graph neural networks to classify encrypted traffic is essentially the same as using tensor neural networks. Therefore, encrypted traffic classification based on graph neural network is typical inductive learning [10]. Because different traffic forms different graphs and we form the graph by partial data (not using the whole flow, but using the limited and fixed packet sequence), each graph is a sample to be classified.

The main contributions of our work are summarized as follows:

- We studied encrypted traffic transmission mode and interaction characteristics in-depth, and based on that, Markov relationship paradigms of encrypted traffic packet sequence are mined. Thus, we innovatively proposed a graphical expression model, WFF, to transform the packet sequence into a graph.
- We studied the advantages of different GNN-based models in feature expression and combined them with the characteristics of encrypted traffic. We proposed three different WFF-based graph neural network models: WFF-GCN, WFF-GGNN, and WFF-CapsGNN. They have different feature expression advantages and data affinity.
- We designed the EGNN architecture, the first attempt to combine ensemble learning with heterogeneous GNNs in encrypted traffic classification. In addition, We applied the existing ensemble mechanism of voting and stacking, and C3WV mode in voting is proposed, which effectively reduces the probability of misclassification among similar classes. We also introduced a variety of meta-models for stacking mode to explore classification capability boundaries.
- We refined the dataset from the previous study by providing a new CERNET-2022-Service dataset and conducting experiments in an open-world environment. The new dataset is published on **https://data.iptas.edu.cn/web/tbps**. The experimental results show that our method can significantly improve the classification effect, up to 99.25% F1-score, which outperforms various state-of-the-art encrypted traffic classification methods. At the same time, we also achieved 99.1% model size reduction so that EGNN can be effectively applied to the actual network environment.

The rest of the paper is arranged below. Section II describes the current research status of encrypted traffic classification, encrypted traffic feature engineering, and graph neural networks. Section III introduces the transmission mode and interaction characteristics of encrypted traffic and illustrates the problems to be solved in this paper. On this basis, the advantages of the three basic GNN models used in this paper in the expression of encrypted traffic features are introduced. Section IV starts with multiple Markov properties in encrypted traffic packet length sequences and introduces this paper's core graphical expression model, WFF. Section V focuses on EGNN in this paper and mainly shows the design of the classification model of the graph neural networks and the various ensemble schemas proposed in this paper. Section VI provides an in-depth experiment and analysis of the proposed method. Finally, we summarize the whole paper and prospect the future research. All the acronyms used in this paper are shown in Table 1.

TABLE 1. All the acronyms used in this paper.

Acronyms	Full name
OTT	Over-The-Top
WFF	Weaved Flow Fragment
ISP	Internet Service Providers
QoE	Quality of Experience
GNN	Graph Neural Network
NSDs	Network Security Departments
DPI	Deep Packet Inspection
WFF-GCN	Graph Convolutional Networks
WFF-GGNN	WFF-based Gated GNN
WFF-CapsGNN	WFF-based Capsule Graph Neural Networks
EGNN	Ensemble GNN
C3WV	Category Cross-Correlation Weighted Voting
SVM	Support Vector Machine
RF	Random Forest
CNN	Convolutional Neural Networks
LSTM	Long Short Term Memory
CapsNet	Capsule Neural Network
SAE	Stacked AutoEncoder
IAT	Inter Arrival Time
GRU	Gate Recurrent Unit
PDU	Protocol Data Unit
GAT	Graph Attention neTworks
kNN	k-Nearest Neighbor
HTTP	Hyper Text Transfer Protocol
TCP	Transmission Control Protocol
TLS	Transport Layer Security
PU	Protocol Unit
IP	Internet Protocol
RNN	Recurrent Neural Network
MTU	Maximum Transmission Unit
AR-D	Adjacent Relevance in one Direction
RS-D	Relevance with a Span in one Direction
AR-OD	Adjacent Relevance in Opposite Directions
UDP	User Datagram Protocol
LT	Link Type
SELU	Scaled Exponential Linear Units

II. RELATED WORK

The two core stages in the life cycle of encrypted traffic classification are classifier optimization and feature selection. Since basic features, especially statistical features, are easy to obtain from traffic, research on encrypted traffic classification begins with classifier optimization. Therefore, we first summarize and analyze the development of classifiers in encrypted traffic classification, then introduce the work related to feature engineering in another stage, and finally pay attention to the graph neural network and its application in encrypted traffic classification.

A. CLASSIFIER EVOLVING IN ENCRYPTED TRAFFIC CLASSIFICATION

In 2016, the ISCX VPN-nonVPN [2] dataset was published, supporting much current research on encrypted traffic classification. In the earliest research, traditional machine learning classifiers such as decision tree C4.5 and Support Vector Machine (SVM) [11] were used for classification. The

traditional machine learning method is straightforward, but the classification effect of the model is not good, and a lot of statistical features need to be used, which brings difficulties to the classification in the actual network.

Therefore, ensemble learning is further used in encrypted traffic classification to improve machine learning methods' classification capabilities. The representative classifiers are the gradient boosting tree [12], and the Random Forest (RF) [3]. However, ensemble learning methods still lack experimental verification in an open-world environment and have preconditions of feature engineering. At the same time, a method that can effectively prevent the random forest from classifying traffic was proposed [13]. It further proves the inapplicability of ensemble learning methods and traditional machine learning methods.

With the popularity of deep learning, some researchers have started using deep learning methods for encrypted traffic classification. The most significant advantage of deep learning is that it does not rely on prior expert knowledge of feature engineering. The end-to-end learning specialty of deep learning (automatic feature selection) can directly input the raw encrypted traffic into the neural network for training and classification. As one of the most commonly used neural networks, 1D-Convolutional Neural Networks (CNN) [4] was first applied to encrypt traffic service classification. Ordinary CNN will ignore its hierarchical structure given the sequential nature of encrypted traffic, so the Text-CNN [14] was introduced. The attention-based Long Short Term Memory (LSTM) model [15] was also proposed based on the LSTM model and attention mechanism. In order to further mine the Markov properties between packets, a Capsule Neural Network (CapsNet) based on vector neurons [16] had also been applied.

With the development of semi-supervised learning, Deep-Full-Range [17] framework was proposed to classify encrypted traffic using juxtaposed 1D-CNN, LSTM, and Stacked AutoEncoder (SAE). Then, a Deep Packet method [18] that integrates CNN and SAE models was proposed. Then, STNN [19] integrating LSTM and 3D-CNN, and the CENTIME [20] combining the advantages of ResNet and AE are further proposed.

B. ENCRYPTED TRAFFIC CLASSIFICATION WITH FEATURE ENGINEERING

Although the automatic feature selection of deep learning saves the trouble of feature engineering to some extent, the structural characteristics of the packet and the potential Markov properties make the feature automatically obtained by it not optimal, and this feature is not interpretable. It is the reason for the rise of encrypted traffic feature engineering.

Many studies show that encrypted packets have the characteristics of sequential state transition. The flow's statistical features vary with the input size (number and range of packets). Even for the same flow, the statistical features of different sub-sequences of packets may have different distributions without a strong correlation. In addition, the expression ability of surface-oriented statistical features is limited, so it is difficult to describe the characteristics of sequential state transition, and the classification effect is not adequate.

Therefore, many studies start with sequence features. The most representative ones are the time interval sequence and length sequence. Among the time-related features, the representative ones are FlowPic [21] based on the packet length and standardized Inter Arrival Time (IAT) distribution in the general domain and time-dependent features [22] under the transform domain.

Moreover, in the length sequence feature, based on the deep learning model, some researchers further focus on length features strongly correlated with transmission data to improve accuracy. The FS-Net [7], based on the multi-layer bidirectional GRU model, is characterized by a full-flow packet length sequence combined with representation learning. The LS-CapsNet [5] composes Gate Recurrent Unit (GRU) for feature extraction and CapsNet for classification. The TFSN model [23] based on LSTM is characterized by the length of bidirectional application data of TLS flow. Then, based on LS-CapsNet, LSCDL [6], a composite deep learning model architecture with Protocol Data Unit (PDU) length sequence as input and N-gram length sequence as a feature was proposed. It considers both emergent and persistent classification requirements.

In multiple-feature cooperation, multimodal deep learning was first introduced in mobile encrypted traffic scenarios, and the MIMETIC framework [24] was proposed. After extracting different input features using CNN and GRU, the dense layer combines and classifies their outputs. To utilize both the packet load sequence-structure features and statistical features, ResPacket [25], a variant of ResNet, was proposed. Then, based on the MIMETIC framework, MIMETIC-ALL [26] is proposed. This method uses contextual counters and multimodal deep learning to classify activity-level traffic in mobile communication applications, which can take advantage of context input as additional modals. However, although complex features improve the classification accuracy, they also increase the cost of training and feature calculation and increase the complexity of the classification model itself, making it difficult to apply such models to the actual high-speed network environment.

Although the above methods express the Markov properties, the sequence input can only be limited to the Markov properties of low-order continuous units (e.g., packets). In particular, packet length is not a feature expressed in Euclidean space. So, a graph would be a more appropriate data structure.

C. GRAPH NEURAL NETWORKS IN ENCRYPTED TRAFFIC CLASSIFICATION

GNNs shone in the computer vision field because of their powerful graph structure expression ability. In addition to

the most basic GNN, many new GNN architectures are also produced, which respectively draw on the advantages of various deep neural network structures in tensor space. For example, there are GCN [27] using convolution operation on a graph, spatial domain information transfer model GGNN [28] based on GRU, Graph Attention neTworks (GAT) [29] using attention mechanism, CapsGNN [30], which uses the dynamic routing mechanism of CapsNet to better fuse node features.

At present, there is some research introducing GNN into traffic analysis. First, GCN and AE are mixed with classifying encrypted traffic [31]. It first uses kNN's flow-based regularized 900-byte primitive features for clustering to form the graph input. After that, AE was used for feature extraction, and GCN was used for classification.

Then, Ji and Meng [32] proposed an attempt to classify traffic using GCN. However, in its graph composition, nodes represent flows, and edges represent the relationship between flows (whether two flows are connected to a public IP station). Zhao et al. [33] propose a method using a Residual GCN to discover anonymous services. They also construct several continuous flows into flow sequences and then convert them into graphs for classification. However, these two attempts are not directly related to encrypted traffic.

Considering the particularity of the graph that can be formed among packets, a basic GNN classification method [8], which takes packets as nodes and spatio-temporal relations of packets as edges, was proposed. Then, a GNN model named GraphDApp [9] based on MultiLayer Perceptron (MLP) was presented, which abstracted the graph structure for the upstream and downstream burst traffic and realized the classification of decentralized applications. Furthermore, based on the previous work, Huoh et al. [34] further introduced multimodal learning to add the second data modality to the relationship between statistical and timing features. At the same time, an end-to-end GNN architecture is proposed using ''encoder-decoder'' architecture, which is superior to GraphDApp.

Although they all transform the sequence of packets in the flow into graphs, they do not study the different Markov properties in the encrypted traffic, so there are specific problems in constructing their graphs, which cannot effectively express these features. Moreover, they only use the most basic GNN and fail to fully play the unique advantages of all kinds of new GNN models.

III. PRELIMINARIES

A. ENCRYPTED TRAFFIC TRANSMISSION MODE AND INTERACTION CHARACTERISTICS

Protocol engineering is the cornerstone of network protocol and network transmission. The transmission of encrypted traffic also follows this feature because the encryption behavior only adds a layer of encryption protocol based on network transmission. For example, the difference between HTTPS and HTTP is that the TLS encryption protocol is used on top of the TCP layer to encrypt the entire HTTP layer. Therefore, the essence of encrypted traffic transmission mode remains in the layer-by-layer segmentation and communication of the two sides of the interaction.

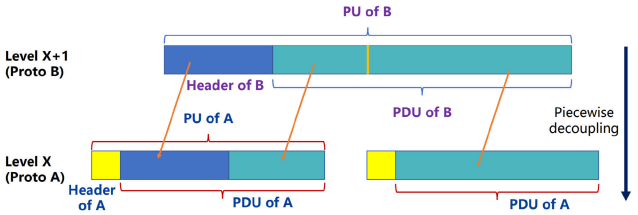


FIGURE 2. Piecewise decoupling of network protocol stack.

In our previous work, we defined what a Protocol Unit (PU) and a Protocol Data Unit (PDU) are. A PU is the smallest unit in the current network protocol stack layer and is protocol-specific. The PU contains the protocol header and data of the current protocol, and the PDU is the data part of the PU and the smallest independent unit for data transmission by the current protocol.

Each protocol layer and protocol has its own unique PU and PDU. Therefore, when the upper-layer PU is nested into the lower-layer protocol, it follows the segmentation mode of the lower-layer protocol to transmit effectively. This process is called piecewise decoupling and is shown in Figure 2.

Each packet in a flow [6] is in sequential order, determined by the timestamp at which the packet arrives. Therefore, a flow is a typical time series. In a bidirectional flow, the interaction characteristic of encrypted traffic is essentially derived from the data exchange between the two parties. With the help of piecewise decoupling, the ping-pong data exchange evolved into a packet sequence, which is also the origin of the Markov nature of the packet sequence, as shown in Figure 3.

The Markov properties mining in the following paper is also conducted based on the above.

B. PROBLEM DEFINITION

Encrypted traffic classification is a typical supervised learning problem. Without considering the unknown class classification, encrypted traffic classification can be regarded as a process of classifying encrypted traffic to a specific service or application under the condition of determining the number and label of categories. This paper takes the packet length sequence as the original input. Assuming there are N samples to be classified and C different categories, then the i-th sample (assuming the sequence size is m) $x_i = [l_1^{(i)}, l_2^{(i)}, \cdots, l_m^{(i)}]$, where $l_j^{(i)}$ refers to the length of the j-th packet in the sequence. Taking the service classification as an example, if the real category of x_i is S_i, the goal of the encrypted traffic service classification is to build a model $\phi(x_i)$ to get a predicted label $\hat{S}_i$ which is expected to be the real label S_i.

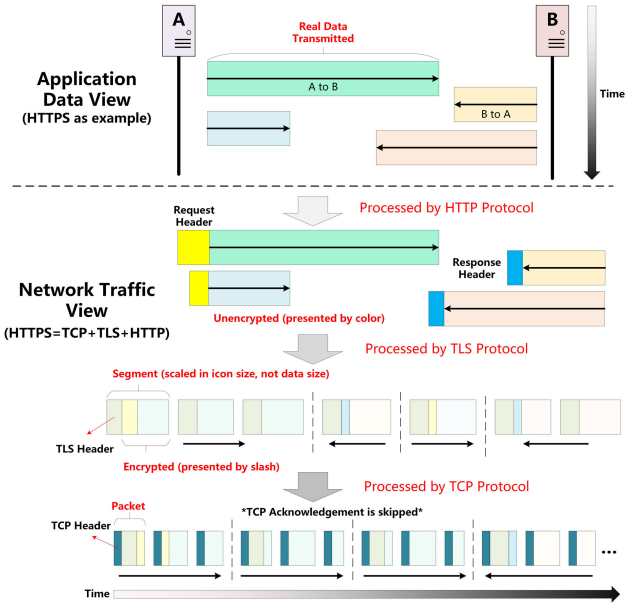


FIGURE 3. Encrypted traffic interaction characteristics and generation of packet sequence.

It is worth noting that since our input is bidirectional, the packet length also needs to reflect the direction. Packets are transmitted in different directions and represent different application data. As a description of the packet data size, the length value can only reflect the data transmission. Furthermore, the direction can reflect the interaction in the data transmission process. Therefore, when the packet length sequence is used as a feature, the direction is also considered. There are two typical approaches. The first is to separate the packet length value with the direction and then input them into the neural network respectively. The second is to use the signed length value to include the direction directly. The advantage of separate input is that it can be combined with multimodal learning, using different neural networks for feature extraction, and in most cases, neural networks are suitable to proceed with positive attention. However, it will consume more memory space, and it is not easy to distinguish between the homogeneity and heterogeneity of packet length sequences. Signed length can deal with this problem well. Therefore, we use a signed length sequence as input, and the input can be better for constructing the graph. We set the upstream flow as positive and the downstream flow as negative. That is:

$$l_j^{(i)} = \begin{cases} \mid l_j^{(i)} \mid, & direction = uplink \\ -1 * \mid l_j^{(i)} \mid, & direction = downlink \end{cases} \quad (1)$$

Our optimization goal is to maximize the effect of classification with a minor possible m.

In the actual network environment, at least one flow must be generated for each access to an application or service. In general, multiple flows are generated. Again, it is worth noting that the classification of encrypted traffic studied in this paper is for a flow.

A network access of network services/applications will generate many flows. Flows of different services/applications have different network ports and target IP addresses and do not conflict with each other, which is coordinated by the local operating system and the server. Therefore, a flow is classified as a certain application or service without considering the relationship between multiple flows. There is also no need to consider the relation between packets of different services/applications, as these are divided into different flows.

If multiple flows are generated in a single access of an application or service, the encrypted traffic classification in this paper requires the classification of each flow to obtain results. Even if this may cause efficiency reduction or confusing results, this method can ensure accurate classification results when obtaining any flow.

In the encrypted traffic classification using graph neural network, the input sample x_i is not directly entered into the classifier but first needs to be converted into a graph $G = < V, E >$, where V refers to all the nodes of the graph, in this paper is the input packet and its features, namely $V = < i, x_i > | i < m$. E refers to the edge of the graph. An edge indicates the relationship between nodes, that is, the relationship between packets. The relationship between packets is derived from the Markov paradigm within the packet sequence features, which is analyzed in Section IV-A. It is worth noting that edges have weights, but in this paper, we do not consider the weights of edges, only their directionality, which is introduced in Section IV-A. E is usually represented by the adjacency matrix of nodes, and can be expressed by the formula 2.

$$E = (x, y) | (x, y) \in V^2, x \neq y, x \; is \; connected \; to \; y \quad (2)$$

In this graph, due to the diversity of packet length sequences, the relationship between node values and edges will change, and the change will become more obvious with the passage of time. This phenomenon is a common problem in encrypted traffic classification: concept drift. Therefore, the graph is also dynamic, but its dynamics are reflected in node values in the short range and edges in the long range, which is different from the dynamic graph in the traditional sense. The features of the graph change, but the graph has a fixed size (m).

Therefore, in general, the graph-based encrypted traffic classification to be solved in this paper is a typical heterogeneous inductive GNN supervised learning problem, which uses node and edge features to classify the entire graph into different services or applications.

C. SPECIALTIES OF REPRESENTATIVE GRAPH NEURON NETWORKS

1) GRAPH CONVOLUTIONAL NETWORKS

GCN [27] is a neural network that takes graph structure as input and performs convolution operations on a graph.

The function of convolution is to extract the features of the input data, which is oriented to the regional features of the input data. Convolution is done using the convolution kernel by scanning the input to multiply and sum the receptive field features and superposition the bias. In encrypted traffic classification, the function of convolution is equivalent to feature extraction of the region composed of some features in the input, that is, focus on the domain of convolution kernel and carry out feature combination calculation.

Studies have shown that convolution is more suitable for processing regularly structured data, such as pictures, in computer vision. At the same time, it performs poorly in serialized data [15]. The main reason is that when serialized data is piled up into tensors (generally matrix in encrypted traffic classification), data adjacency relations that do not actually exist are added, leading to the extraction of non-existent advantage features by convolution. (Take 100 packets as an example. If the features of each packet are converted into a 10 * 10 matrix, a meaningless association is added between the features of the first packet and the 11th packet.)

Fortunately, we construct the packet length sequence into a graph in this paper. Nodes represent the signed length sequence features of packets, while edges represent the correlation between packets. In the whole part of the graph, a subgraph represents the relationship between several adjacent packets, and such a local Markov relationship exists in each subgraph. Graph convolution can effectively traverse each subgraph in the graph with the size of the kernel to extract its local dominant features. Therefore, we choose to use GCN as one of the base models of encrypted traffic classification.

2) GATED GRAPH NEURAL NETWORKS

GGNN [28] is a message transfer model introducing the GRU model, a variant of Recurrent Neural Network (RNN), into the graph space domain.

In the context of tensor input, the advantage of RNN, especially the LSTM and GRU models, is that it can effectively learn the input features of each time step. Since a conventional neural network can only handle one input at a time, it is inconsistent with the current situation that the previous input is related to the following input in the sequence data. Therefore, RNN class methods can effectively understand Markov relationships in sequences. In encrypted traffic classification, especially in the classification scenario with packet sequence as the input, the RNN class method can effectively extract the relationship between each packet feature to dig more in-depth features.

The graphical packet sequence is no longer the direction of Markov conduction from the front to the back but is represented as adjacent. As a graphical representation of the GRU model, GGNN still lies in iteration and output. For GGNN, if the packet sequence is graphed, the short-range Markov between the length features of adjacent packet nodes on the graph and the long-range Markov within the connected component are considered.

GGNN can also depict long and short term relationships, which is reflected in the propagation of relations between adjacent nodes and more distant connected nodes on the graph. GGNN can better learn different relationship influences centered on the communicator or the communicated because it separates the out-degree matrix from the in-degree matrix. After the packet length sequence of encrypted traffic is converted into a graph, there is a certain correlation between the length values. Since the carrier of the packet sequence (flow) is chronological data from the beginning to the end, the directed graph can better describe the relationship between each packet.

GGNN's advantages in digraphs make it more suitable for graphing packet length sequences. GGNN can better describe such properties because encrypted traffic transmission modes and interaction characteristics bring short-range and long-range Markov properties to packet length sequences [34]. So we choose to use GGNN as one of the base models.

3) CAPSULE GRAPH NEURAL NETWORKS

CapsGNN [30] refers to the GNN of CapsNet.

The most important element in CapsNet is the capsule, which in the CV field is a carrier containing multiple neurons. Each neuron represents various attributes of a specific entity in the image, attributes contain many different types of instantiation parameters, and values are instances of a certain category. In NLP, similar to encrypted traffic classification in sequence input, capsules correspond to word vectors. It uses a vector to replace the scalar in node embedding because the representation of the scalar node is not enough to effectively retain the complete attribute of a node or the whole graph. In encrypted traffic classification based on packet length sequence, a capsule is a packet length sequence feature vector generated after a series of processing of the packet length sequence.

The advantage of CapsNet is that capsule, as a vector, can more richly express extracted features. It is because for the same feature, multiple nodes may be required for recognition by scalar neurons, but the CapsNet only needs one vector. It can better identify the equivariance between labeled samples of the same class than the base model CNN, which only learns the invariance, improving the robustness of the neural network.

In addition, CapsNet uses a dynamic routing mechanism to update some parameters according to the capsule's characteristics to replace the gradient descent partially. Its essence is that the output capsule through the dynamic routing algorithm is some clustering result of the input capsule. It can get the significance of some features, mine the more significant Markov feature subset among the packet length sequence features, and rely on the reconstruction loss function to exclude some features that have little impact on the classification.

The most significant advantage of CapsGNN, like CapsNet, is that it can use equivariance to learn the directivity of features in the sample (it is worth noting that the directivity refers to the position relationship between features). Unlike

images, for encrypted traffic packet length sequence features, directivity refers to the specific sequence relationship of packet length values. However, the process is not without sacrifice, and the normalization of weights can lead to a decline in learning at invariance [16]. In the graphic packet length sequence, CapsGNN can better describe the direction of the edge, which is more explicit than that in the original packet length sequence, and has certain interpretability. Therefore, we also chose CapsGNN as one of the base models, complementing GCN.

IV. PACKET SEQUENCE MARKOV PROPERTIES AND WEAVED FLOW FRAGMENT

WFF is a new model that converts bidirectional flow into a graph based on Markov properties inside the flow. Before introducing the model, we describe the Markov properties found in the study of actual network traffic.

A. MARKOV RELATIONSHIP PARADIGMS OF ENCRYPTED TRAFFIC PACKET SEQUENCE

Markov relationship paradigms of encrypted traffic packet sequence are based on encrypted traffic transmission mode and interaction characteristics. Since the flow is bidirectional, the packet sequence is also bidirectional. Although there may be a sequence of packets in only one direction, its abstract nature is also a bidirectional sequence with no packets in the other direction. Therefore, the same and the opposite directions should be considered when mining the relationship between the packets.

1) PACKET RELEVANCE IN ONE DIRECTION

In Markov properties in the same direction, due to piecewise decoupling, data larger than MTU will be divided into multiple packets that belong to the same data block. We call this Adjacent Relevance in one Direction (AR-D). In addition, if one side of the communication sends data multiple times in a row and the other side does not respond, even packets of different data blocks may constitute continuous packets, which is also a kind of AR-D.

AR-D refers to the Markov property between each continuous packet in the same direction in a bidirectional flow. The status of the latter packet is related to the previous one, which conforms to the first-order Markov chain relationship. AD-R can be expressed by the following Formula 3.

$$\begin{aligned} &\forall p_i \in \text{flow}, 1 \le i \le N_{\text{flow}}, \\ &\text{direction}\,(p_i) = \text{direction}(p_{i+1}) \\ &\text{relevence}(p_i, p_{i+1}) = 1 \end{aligned} \tag{3}$$

where p_i refers to the i-th packet, N_{flow} refers to the number of packets in the flow.

In the actual network environment, a whole huge data file or multiple files of the sender will be transmitted multiple times to complete the delivery of the receiver(e.g., online video and web browsing). In addition, in some scenarios, the sender needs to continuously send data to the receiver (such as VoIP and live network broadcasting). In these scenarios, one party sends data continuously while the other party also sends data, so the packets in the same direction are separated.

In addition, after a certain amount of data has been transmitted in one direction, the other side may also return an acknowledgment to inform the sender that it has successfully received the data. The best example is the TCP protocol. In TCP transmission, after the sender transmits data, the receiver sends an acknowledgment (ACK) packet that the currently transmitted data is accepted successfully.

This is the core reason for the correlation between the two directions of the subflows in a bidirectional flow, and it is also why packets in the two directions appear alternately. As a result, continuous packets in the same direction are separated. Although the two separated subsequences may not necessarily belong to the same block, the last packet of the previous subsequence is related to the first packet of the following subsequence due to the persistence of data fragmentation and behavior logic.

We call this Markov relationship paradigm Relevance with a Span in one Direction (RS-D). RS-D refers to the Markov properties between the discontinuous nearest packet clusters in the same direction in a bidirectional flow. It can be expressed by the following Formula 4.

$$\begin{aligned} &\forall p_i \in \text{flow}, 1 \le i \le N_{\text{flow}}, \\ &\text{direction}\,(p_i) = -\,\text{direction}\,(p_{i+1}) \\ &j = \text{argmin}\left(\text{direction}\,(p_i) = \text{direction}\left(p_{\text{j}}\right), j > i\right) \\ &\text{relevence}\left(p_i, p_{\text{j}}\right) = 1 \end{aligned} \tag{4}$$

2) PACKET RELEVANCE IN OPPOSITE DIRECTIONS

When the interaction causes the RS-D, it also associates the packets of opposite directions. This association is manifested in two adjacent subsequences in opposite directions. Whether data packets or acknowledgment packets, adjacent heterogeneous packets are related without considering packet disorder due to the continuity of interaction. The acknowledgment packet is related to the previous data because the acknowledgment packet is defined as the acknowledgment. For data with opposite directions, there are two cases. The first is that in the request-response mode, the data is requested by the other party through the previous subsequence of the packet, and the two heterogeneous packet subsequences are related. The second type of data is actively pushed in the subscription-publishing or peer-to-peer network model, which is relevant at the behavioral level, such as the data generated by the behavior of the two participants in an online game.

Hence, this kind of packet correlation is named Adjacent Relevance in Opposite Directions (AR-OD). AR-OD refers to the Markov property between adjacent continuous packet subsequences of opposite directions in a bidirectional flow. In particular, in two adjacent packet subsequences of opposite directions, the status of the first packet in the latter subsequence is related to the last packet in the former subsequence.

AR-OD can be expressed by the following Formula 5.

$$\begin{aligned} &\forall p_i \in \text{flow}, 1 \le i \le N_{\text{flow}}, \\ &\text{direction}\,(p_i) = -\,\text{direction}(p_{i+1}) \\ &\text{relevence}(p_i, p_{i+1}) = 1 \end{aligned} \tag{5}$$

B. GRAPHICAL PACKET SEQUENCE

Based on the three different Markov relationship paradigms, we propose WFF, a graphical expression model suitable for both online and offline encrypted traffic classification. It is worth noting that there are noticeable differences between online classification scenarios and offline classification scenarios. The most obvious gap lies in data integrity. In online scenarios, packets continue to arrive, and the size of the target flow to be classified is unknown. Therefore, an input threshold (number of packets) is required. Offline scenarios have even more freedom, with all the complete data that can be extracted and analyzed at will. However, in terms of real-time requirements, offline classification cannot be guaranteed because it needs to persist the entire flow first. However, when the flow is completed, the behavior behind the stream may also end.

Therefore, the bottleneck of offline classified deployment is collectors and storage. In online classification deployment, traffic at the current collection point is directly transferred to the classification device for analysis through an optical splitter or port mapping. In rare cases, serial deployment is used. To satisfy both online and offline scenarios, we need first to limit the size of the input, that is, the number of nodes in the graph, and fix it.

To satisfy the input of GNN further (transforming the input into a graph), we proposed the Link Type (LT) as the detailed graphical expression paths to express all the relevance, respectively. Noteworthily, each packet appears as a node during graph construction, and each node v has a feature of packet length value.

We proposed four different LTs, corresponding to the three different Markov relationship paradigms mentioned above and a need for graphic integrity.

LT1 stands for AR-D. LT2 stands for RS-D. For graph construction, they sequentially connect all packets in the same direction. LT1 and LT2 are equivalent to an expression of unidirectional flow in each direction of bidirectional flow. However, it is worth noting that the expression differs from that of ordinary flow. The main reason is that LT2 connects the packets that should be connected in the middle of the flow but are separated by different packets, ensuring that their Markov properties can be extracted.

LT3 represents the AR-OD. Due to the sequential relationship between packets in the flow, if two adjacency packets are in opposite directions, they need to be connected. If the direction of the next packet and the current packet is opposite, the two nodes are connected with an edge to show that the next opposite packet is related to the current adjacent packet, as LT3 stands for. It is worth noting that in the actual network, if the two communication parties transmit data to each other, and the data is irrelevant, it will be realized through two flows, which are determined by the network transmission mechanism. Therefore, LT3 can fully express the AR-OD.

Although LT3 appears to be a time-consuming function for connecting the cross-packet node because it appears to merge the traffic from two directions, However, practically implementing does not implement these concerns because traffic is first collected at a collection point, two-way traffic is collected at the same time in the network adapter, and the flow forming is the first thing in traffic collection. The flow forming can be completed in $O(n)$ time complexity, which is not very time-consuming. Moreover, we only need a fixed number of packets in a flow to form our packet length sequence to build WFF. Therefore, we can judge each flow in real time and only retain its length sequence features. When the size of the length sequence reaches the predetermined threshold, the following operation can be carried out. The weaving process can even be implemented during a flow forming if the device supports it.

For LT3, there is another case that needs to be discussed in the real Internet: asymmetric traffic. In a narrow sense, asymmetric traffic refers to two-way traffic not routed in the same transmission path but transmitted in two different paths. As a result, only one-way traffic can be collected at a certain collection point in the path, resulting in inaccurate WFF generation. Studies show that asymmetric traffic accounts for a large proportion of the Internet, and the main reason lies in ISPs' efficient traffic management. In a broad sense, asymmetric traffic contains a special case in which, within the threshold value of the packet length sequence selected by us, one of the two communicating parties continuously sends data to the other without receiving any feedback packet (which may appear in UDP), resulting in only packets in one direction. WFF degrades to a unidirectional packet length sequence. We call this phenomenon oppressive transmission.

In order to solve asymmetric traffic, we need to collect traffic as close as possible to both communication parties or at the unique boundary that has been determined, which is a problem to be considered in engineering implementation. For oppressive transmission, the sequence itself is a special case of a mathematical graph, so it will not affect the method, and the experiment also shows that when oppressive transmission appears, it can still achieve effective classification.

Since the input sequence is finite, the LT4 represents the integrity of the graph structure, i.e., the self-consistency of the sequence. If the first or last packets in opposite directions are not connected, they will be connected to form a integrated connected component.

Figure 4 shows the weaving process of the construction of the graphic packet sequence, which is combined by four different LTs.

In actual network environments, in addition to oppressive transmission, there may also be unseen packets, a phenomenon commonly seen over UDP packet loss. When packets are missing, the structure of the graph changes. The biggest change that may happen to the graph structure is

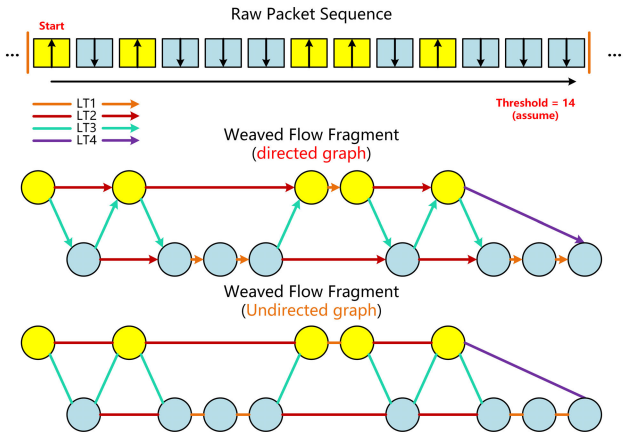


FIGURE 4. Weaving process to construct a graphic packet sequence.

the loss of a single packet in the opposite direction, which degrades RS-D and AR-OD to AR-D. However, the total size of a flow is generally much larger than the size required by the input. Therefore, the loss of some packets (nodes) will only affect a small part of the features of the graph but will not change the number of nodes in the graph, so it will not affect the classification process. If the model has strong robustness or the samples take this phenomenon into account, the classification accuracy can still be guaranteed, which is also a factor considered when constructing the classifier.

It is worth noting that WFF can be either directed graph mode or undirected graph mode. Different GNN models have different expressive abilities for directed or undirected graphs due to the gap between neurons or layers. We have conducted an in-depth experiment on whether a model should use the WFF of a directed or undirected graph. For both directed and undirected graphs, we set upstream traffic's packet length value as positive and downstream traffic as negative.

V. ENSEMBLE GRAPH NEURAL NETWORKS

After we construct the flow into WFF, we need to build the GNN model for classification. Since GCN, GGNN, and CapsGNN all have their own advantages in the classification of encrypted traffic theoretically, we first design three corresponding WFF-oriented classifiers based on these three basic models. Then we propose various ensemble mechanisms of GNNs according to the ensemble mechanisms of traditional ensemble learning and the characteristics of GNNs. The goal of EGNN is to balance the effect and efficiency of classification in resource-constrained multi-category scenarios and avoid misclassification among similar categories.

A. STRUCTURAL DESIGN OF THREE DIFFERENT GRAPH NEURAL NETWORKS

We proposed three models, WFF-GCN, WFF-GGNN, and WFF-CapsGNN, respectively, using the advantages of their base models. These models can classify encrypted traffic individually or as participants in EGNN.

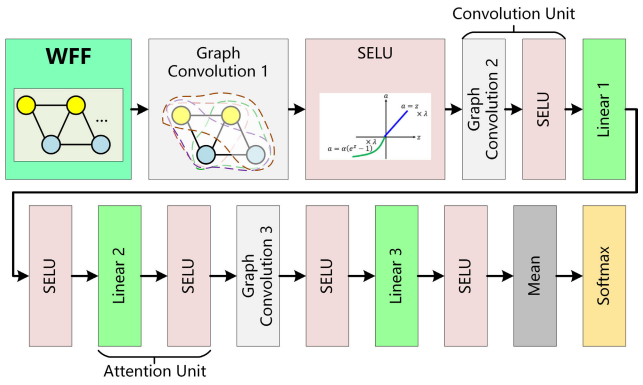


FIGURE 5. Network structure of WFF-GCN.

1) WFF-GCN

In WFF-GCN, the core calculation is to extract the features on the graph directly through graph convolution. Therefore, in the structure of traditional GCN, we use the two-layer iterative "convolution-activation" to mine the significant features between adjacent nodes in WFF. We use Scaled Exponential Linear Units (SELU) as the activation function to solve the problem of eliminating negative. Because in a bidirectional flow, the downstream packet length value is negative. If traditional activation functions such as ReLU are used, the negative feature will be lost. The specific network structure of WFF-GCN is shown in Figure 5.

2) WFF-GGNN

In WFF-GGNN, since GGNN is a model in which the input and output edges transmit messages respectively, we first need to convert the edge set of WFF into an adjacency matrix of in-degree and out-degree and then connect the edges after linear transformation of alignment. If the directed graph WFF is taken as the input, the in-degree and out-degree matrices are symmetric matrices. The two matrices corresponding to the undirected graph WFF are the same. The core of GGNN is the propagator that iterates continuously over the input, containing the reset and update gates, just like the GRU model. The update gate controls the extent to which state information from the previous state is carried into the current state, while the reset gate controls how much of the previous state is written into the candidate set. Both gates are transformed by a linear layer (i.e., a fully connected layer), and the sigmoid function is used as the activation function. On this basis, we used the transform component to transform the candidate state data obtained by the reset gate, which connected to the current state (using the linear layer and Tanh activation function) and propagated the information based on the weight of the update gate. The specific network structure of WFF-GGNN is shown in Figure 6.

3) WFF-CapsGNN

Since the CapsGNN model uses a capsule to solve the problem that the global characteristic of the graph will be lost, we divide WFF-CapsGNN into five phases based on the basic

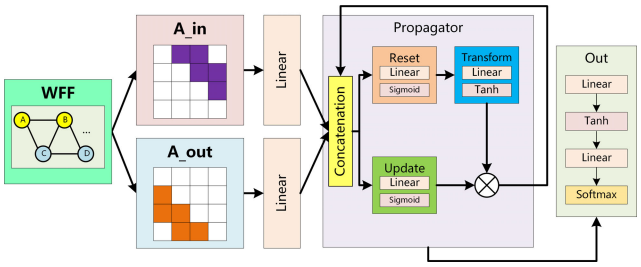


FIGURE 6. Network structure of WFF-GGNN.

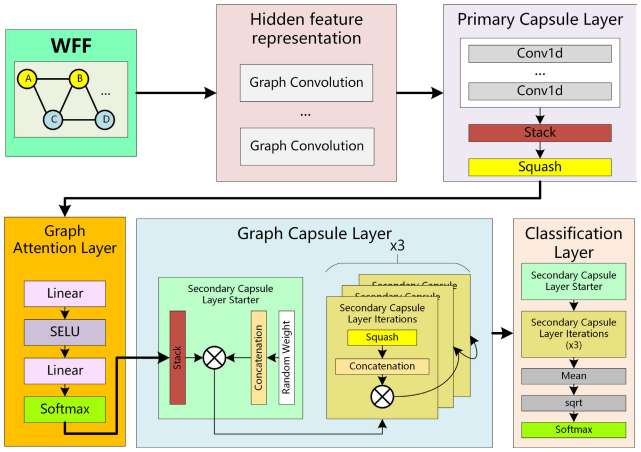


FIGURE 7. Network structure of WFF-CapsGNN.

CapsGNN. The first is the hidden feature representation, in which graph convolution is used to extract features. Then, the primary capsule layer generates the input capsule, and the graph attention layer is applied for key feature mining. Afterward, the graph capsule layer uses a dynamic routing algorithm to obtain the classification capsules through the secondary capsule layer. Finally, the secondary capsule in the classification layer employs a dynamic routing algorithm again to obtain the output capsule. In addition, to ensure the consistency of model output, we add a large product factor and the Softmax function to make it applicable to classification problems.

Due to the transmission pattern of encrypted traffic, some packets may be out of order in the actual process, which may be caused by network fluctuations or the randomness of the transmission order of non-synchronous resources. Therefore, we also redesigned the model's loss function and gave up the reconstructed loss function of the CapsGNN model to avoid the problem of decreasing the classification accuracy of similar samples caused by the packet disorder. The specific network structure of WFF-CapsGNN is shown in Figure 7.

The three classification models all adopt the **cross-entropy loss** as the loss function.

B. ENSEMBLE MECHANISM OF GRAPH NEURAL NETWORKS

In encrypted traffic classification, encrypted traffic in different categories may have relatively similar features due to network services' similarity and network applications' homogeneity.

The four most common ensemble methods in traditional ensemble learning are boosting, bagging, stacking, and voting [35]. The first two are dataset-oriented, while the last two are model-oriented.

Boosting methods can repeatedly improve the attention of the misclassified samples in the previous training process and train the next base classifier based on the adjusted sample distribution. However, boosting methods are used for repeated training of weak classifiers. GNN, as a strong classifier, has a better basic classification effect and a relatively long training time. Boosting methods would lead to higher training costs for the GNN model in repeated training than traditional machine learning classifiers, but the effect would not necessarily be improved sufficiently. Furthermore, boosting methods work on a single model (isomorphism algorithm). Using the misclassification results of one GNN to train the other GNNs would not necessarily get a boost since the other GNNs would not necessarily misclassify these samples. Boosting would only be meaningful if the samples were misclassified by all the GNN participants.

As a method of aggregating multiple models and aggregating the results together for classification, bagging is trained based on a random sampling of datasets (the most typical model is RF). The bagging methods can use different features in addition to sample extraction. However, in general, the same model will be applied to obtain the feature mining of datasets from different angles by one model. It is not easy to apply to the collaborative training between heterogeneous GNNs.

Stacking is an ensemble learning mechanism based on a meta-model. For EGNN, different GNNs are used as first-layer classifiers, and their classification results are used as the input of the second-layer meta-model to obtain the final prediction results. Stacking can be effectively used in EGNN, but the selection and training of the meta-model would require additional costs.

Voting is the most straightforward but practical ensemble mechanism. It directly combines the classification results of multiple models, and the result with the most votes will be selected as the final result. Voting consists of two modes, namely, hard voting and soft voting.

Therefore, we selected the stacking and voting modes, respectively, to construct the ensemble graph neural network.

1) STACKING MODE EGNN

In stacking, we also use the hard and soft modes of voting. That is, the output of a layer of classifiers is the result of a category or the output vector of the last layer Softmax function, which contains the confidence of each category.

Therefore, we adopted a relatively simple machine learning model for the meta-model because the output of each GNN in the first layer is just a value result or a Softmax vector,

and there is no need to use a complex model to increase the performance overhead of the overall model.

It is important to note that because stacking is a two-layer model, the output of classifiers at one layer is not necessarily the output of the results at the more intermediate layers, as this will affect the structure of each GNN itself.

2) VOTING MODE EGNN

In voting, if it is a simple hard mode, the results of the three GNNs are put to the vote directly. However, since encrypted traffic classification is a multi-classification task, three GNNs can give completely different results. In this case, we choose the random or president mode to make the final decision. The president mode refers to the prior selection of a classifier as the president. When there is a conflict between the results, the result of the president is taken as the final result.

In soft mode, we can weigh the results of the classification or the results of the Softmax vector. In the case of weighted voting of classification results, it needs to be based on historical data. Therefore, we proposed the Category Cross-Correlation Weighted Voting (C3WV) mechanism. In C3WV, by mining the categories that are prone to mutual confusion in the training process, the correlation number μ is set for the results of similar categories to correct the categories with ambiguity in the final voting through weighted calculation.

Assuming a total of n different categories and m classifiers, C3WV can be describe as:

$$\mathbf{CMatrix}^{(k)} = \begin{bmatrix} E^{(k)}_{1\to 1} & \cdots & E^{(k)}_{1\to n} \\ \vdots & \ddots & \vdots \\ E^{(k)}_{n\to 1} & \cdots & E^{(k)}_{n\to n} \end{bmatrix}, \quad \forall 1 \le i,j \le n, 0 \le E^{(k)}_{i\to j} \le 1, \sum_{g=1}^{n} E^{(k)}_{i\to g} = 1 \tag{6}$$

$$\mu_i^{(k)} = \mathbf{CMatrix}^{(k)^T}(i) = [E^{(k)}_{1\to i}, E^{(k)}_{2\to i}, \cdots, E^{(k)}_{i\to i}, \cdots, E^{(k)}_{n\to i}] \tag{7}$$

$$\widetilde{y} = index(max(\sum_{k=1}^{m} \mu^{(k)}_{\widetilde{y^{(k)}}})) \tag{8}$$

where E refers to a certain evaluation criterion, $E^{(k)}_{i\to j}$ refers to an evaluation criterion unit that classifies all samples of the i-th category into the j-th category under the k-th classifier. If E is the accuracy rate, then $E^{(k)}_{i\to j}$ represents the proportion of class i samples classified into class j in the k classifier to the total number of class i samples, which generally adopts the classification accuracy rate of this category. $\mathbf{CMatrix}^{(k)}$ refers to the prior confusion matrix of the k-th classifier, $\widetilde{y^{(k)}}$ is the classification prediction result of the k-th classifier, $\widetilde{y}$ is the classification prediction result of C3WV.

As for the confidence of the Softmax vector, we directly weighted its average to get the voting result.

VI. EXPERIMENTAL EVALUATION

A. DATASET

To prove the effectiveness of the method, we select two datasets. One is the ISCX VPN-nonVPN dataset [2]. It is an encrypted traffic dataset used by most research institutes with much reliability. However, it contains some seriously expired encrypted traffic or unencrypted traffic. We reserved four classes for subsequent classification based on the previous encrypted traffic detection method SED-CapsNet [36].

At the same time, we used a dataset we had collected and used in the previous research. It is a fresh dataset in 2021 collected from the CERNET environment [6], including TLS-1.3, QUIC, and DTLS. In this study, we supplemented and improved this dataset and named it CERNET-2022-Service. Since the classification unit of this paper is flow, and the minimum input unit is a packet, we do not consider the highest visible protocol [6] above the transport layer and divide the dataset into two parts: TCP and UDP (that is, the traffic with TCP at the transport layer and UDP at the transport layer are respectively classified).

It is worth noting that our research faces an open-world environment, so the training set for training each GNN participant is completely separated from the test set. In addition, the part for testing EGNN is further sourced from the GNN test set (further division) to fully depict the situation that the new traffic in the actual network environment is different from the old traffic for training.

The statistical results of the two datasets are shown in table 2. It is worth noting that in the follow-up experiment, in order to reflect the fairness of the experiment and the uncertainty in the open-world scenario, we did not directly select the complete dataset for the experiment but randomly selected the same total number of samples and divided them into training sets and test sets according to 8:2 for the experiment.

The new dataset CERNET-2022-Service is published on **https://data.iptas.edu.cn/web/tbps**.

B. EXPERIMENT SETTINGS & COMPARED METHODS

We conduct our experiment in a high-performance workstation with an AMD 5950x CPU, 64-GB memory, and a RTX 3090 GPU, within a running environment of CUDA version 11.1, Pytorch version 1.8.1, and Python version 3.8.8.

To test the effectiveness of the method, precision rate (Pr), recall rate (Rc), and F1-score (F1), which are most commonly used, are selected as evaluation criteria.

Several state-of-the-art methods mentioned in Section II are chosen to compare our method:

- CNN [4] is a basic deep learning model using automatic feature selection.
- Deep Packet [18], a classic model for traffic classification, combines CNN and SAE.
- FS-Net [7] is one of the state-of-the-art models for encrypted application traffic classification.
- LS-LSTM [6] is currently one of our best methods for encrypted service traffic classification.

TABLE 2. Statistics of filtered ISCX VPN-nonVPN dataset and CERNET-2022-Service.

	Category	Flows (UDP: Packets)	Category	Flows (UDP: Packets)
ISCX	Chat	467	Email	247
	File Transfer	84	Streaming	1161
	P2P(UDP)	3337	VoIP(UDP)	8562012
CERNET	File upload	1757	Streaming	12458
	Web-https	11401	P2P	3857
	VoIP(UDP)	2259292	Streaming(UDP)	22249926
	P2P(UDP)	1581877		

As we use the stacking mode in EGNN, the classifiers used in the stacking mode also need to be verified experimentally. Because the inputs and outputs of the stacking second-level meta-model are relatively simple, we use traditional machine learning methods and ensemble learning methods to conduct experiments. There are five methods:

- C4.5, a representative decision tree model with decent interpretability.
- SVM, a high-dimensional classification model based on kernel functions.
- kNN, a classical classification model based on feature distance.
- XGBoost, an excellent ensemble learning model based on gradient boosting.
- RF, a powerful ensemble learning model based on tree bagging.

C. CLASSIFICATION EFFECT AND PERFORMANCE

Our experiment is divided into three parts. The first part is the experiment on the open dataset, comparing the classification effect of the GNN classifiers with other classifiers and further experimenting on the effect of EGNN. We then conducted further experiments on the CERNET-2022-Service dataset. Finally, we consider the possible bottlenecks in system deployment under the actual network environment and evaluate each model's performance requirements and resource consumption.

1) EFFECTIVENESS EVALUATION ON ISCX DATASET

First, we conducted a basic model classification experiment on the open dataset ISCX VPN-nonVPN. The classification result is shown in table 3. It is worth noting that the overall precision rate, recall rate, and F1-score of the classifier are obtained from the macro average of each category's results. Therefore, the F1-score may be lower than the F1-score calculated directly using the overall precision rate and the overall recall rate (resulting in the F1-score being lower than the precision and recall simultaneously).

Because different methods used for comparison have specific differences in input, to ensure respect for the methods being compared, we did not modify the original input and can only ensure the consistency of input size within a limited range. Therefore, we also clarified the input size of the four

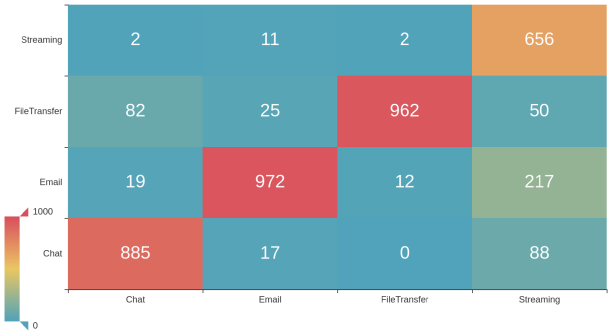


FIGURE 8. Heat map figure of GCN on ISCX VPN-nonVPN dataset.

compared methods and the three GNN methods we proposed in the table.

It can be found from the results that although the data required by CNN and Deep Packet is minimal, only 1500 bytes of an MTU, the classification precision rate and recall rate are too low to be used in the actual network environment. FS-Net has a good effect. However, its input size is too large (on average, each flow carrying data has about 1000 packets), and it is not easy to apply to the online classification scenario. From the perspective of effect alone, the best two models are the LS-LSTM model proposed in our previous work and the WFF-GGNN model presented in this paper, both of which are superior in precision rate and recall rate, respectively. However, from the perspective of input size, LS-LSTM uses PDUs instead of packets. As a result, the required number of packets is much larger than that of WFF-GGNN under the condition of the same length sequence size. At the same time, WFF-GCN and WFF-CapsGNN also have acceptable performance. Therefore, in general, WFF-GGNN outperforms all state-of-the-art methods in this dataset.

Although the overall classification effect of WFF-GCN and WFF-CapsGNN is not as good as that of WFF-GGNN, both models have unique feature extraction and description capabilities. In order to describe such capabilities, the performance of the three models on the ISCX dataset is described in detail in the form of a heat map, as shown in Figure 8, 9, and 10.

From the heat map figure of the confusion matrix, we can see that WFF-GCN performs well in the classification of

TABLE 3. Experimental results for the effect criteria on the ISCX VPN-nonVPN dataset.

Methods	Precision	Recall	F1-score	Input size
CNN	0.4082	0.3853	0.3824	1500 bytes
Deep Packet	0.4296	0.3818	0.3843	1500 bytes
FS-Net	0.9561	0.9535	0.9539	entire flow (1000 packets in average)
LS-LSTM	**0.9775**	0.9568	**0.9670**	50 non-zero PDUs (300 packets in average)
WFF-GCN	0.8820	0.8696	0.8648	**50 packets**
WFF-GGNN	0.9597	**0.9596**	**0.9596**	**50 packets**
WFF-CapsGNN	0.8970	0.8881	0.8853	**50 packets**

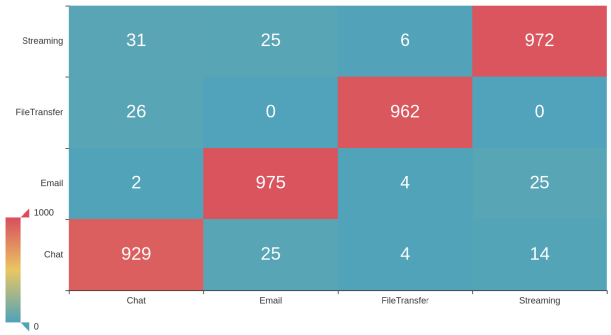


FIGURE 9. Heat map figure of GGNN on ISCX VPN-nonVPN dataset.

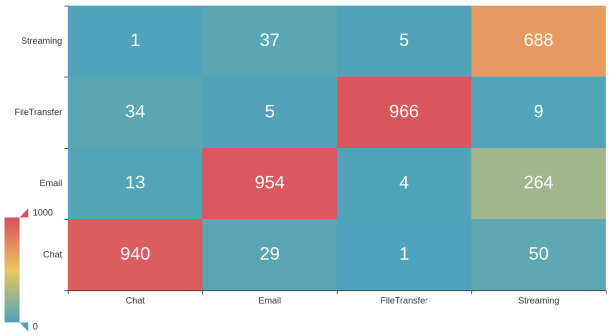


FIGURE 10. Heat map figure of CapsGNN on ISCX VPN-nonVPN dataset.

email and file transfer but poorly in the classification of streaming because WFF-GCN has the advantage of leveraging local dependencies in WFF. Email and file transfer are smaller samples in the ISCX dataset, so their length values and interactions are more varied locally than others. WFF-GCN effectively extracts these features. In streaming, the packet length sequence values are similar (due to MTU), and many continuous transmissions are in the same direction. Convolution will cover the minor interactions to a certain extent, resulting in poor classification effects of this category. WFF-GGNN is excellent for email and streaming and suitable for every category else. The result reflects the superiority of WFF. GGNN makes good use of various features in the graph. While WFF-CapsGNN performs very well in the chat, email, and file transfer categories (chat, in particular, is the best of

TABLE 4. Experimental results for the effect criteria on the ISCX VPN-nonVPN dataset.

Methods	Precision	Recall	F1-score
Baseline (3 WFF-GNNs average)	0.9129	0.9058	0.9032
EGNN-Voting-hard	0.9111	0.9035	0.9014
EGNN-Voting-president	0.9240	0.9193	0.9189
EGNN-Voting-soft	**0.9541**	**0.9534**	**0.9531**
EGNN-Voting-C3WV	0.9305	0.9252	0.9244

the three models), it does not perform as well in streaming. This result is similar to GCN, which has better short-range Markov utilization. Moreover, vector neurons of CapsGNN can better describe the directivity of continuous interaction in chat, which improves the classification effect. Therefore, the three GNN participants in EGNN have their unique value, and there is no waste of classification ability.

Then, we further compared the modes of voting and stacking in EGNN. First, we took the average classification effect of the three GNNs as the baseline to conduct a controlled experiment on the four modes of voting. With precision rate, recall rate, and F1-score as evaluation criteria, the experimental results are shown in Table 4. It is worth noting that the president we selected in president mode is the WFF-GGNN model, according to the result of Table 3.

As seen from the table, after voting is used, compared with the baseline model without EGNN, the overall classification effect is improved to a certain extent, except for the hard mode. We use the stacked bar diagram to show the improvement of the four models under the three evaluation criteria, as shown in Figure 11.

The figure shows that both president mode and C3WV mode have significantly improved the effect, while soft mode has the most pronounced effect. The main reason is that the primitive used for voting is more complex, which is about the classification confidence of all categories obtained by the current classifier. This kind of data's information entropy is much higher than the classification result (generally speaking, it is only an integer). Except for soft mode, our C3WV results are the best because it effectively uses the historical data distribution to predict and correct the misclassification. However, the hard mode not only does not improve but reduces the classification effect to a certain extent compared

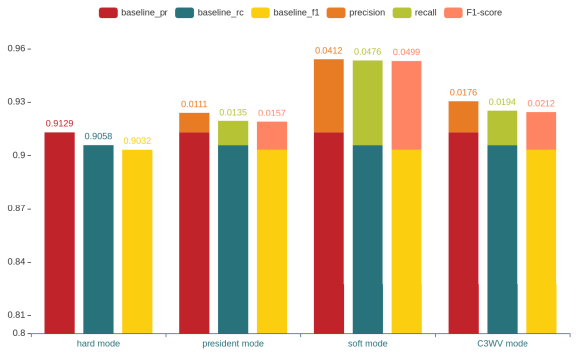


FIGURE 11. EGNN improvement of classification effect in voting modes on ISCX VPN-nonVPN.

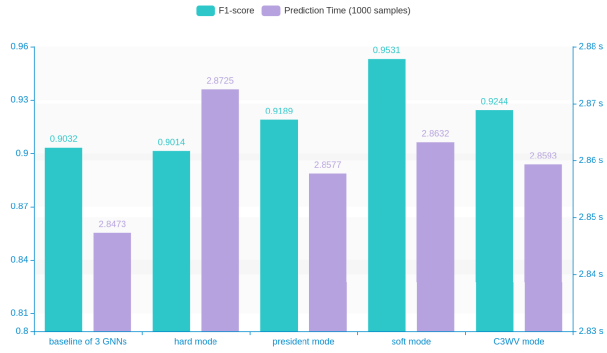


FIGURE 12. EGNN comparison of classification F1-score and prediction time consuming in voting modes on ISCX VPN-nonVPN.

with the baseline result. The main reason is that when the results conflict, a result will be uniformly randomized. If only one classifier classifies correctly, the probability of obtaining the correct result will be significantly reduced. However, the president mode effectively inhibits the classification error when the three results of classifiers are completely different.

However, the classification improvement brought by the four EGNN voting modes is not a free lunch. Therefore, we further experiment with the prediction time of four voting modes. In order to reflect the classification performance under the backbone network, we use the form of replay samples (as the ISCX dataset does not have enough samples) to test the prediction time of 1,000,000 flows. The overlap bar diagram of the prediction time combined with the F1-score is shown in Figure 12.

As can be seen from the results, hard mode does not improve the classification efficiency and affects the prediction time efficiency, so it should be deprecated. While soft mode improves the classification accuracy, it also significantly increases the prediction time. President mode and C3WV mode have better classification accuracy and adequate prediction time. Compared with the president mode, the C3WV mode has better classification performance under similar prediction time, and the president mode needs to select a president in advance. This process may be difficult

TABLE 5. Experimental results for the effect criteria on the ISCX VPN-nonVPN dataset of EGNN stacking hard mode.

Methods	Precision	Recall	F1-score
Baseline (3 WFF-GNNs average)	0.9129	0.9058	0.9032
EGNN-Stacking-hard-C4.5	**0.9775**	**0.9781**	**0.9777**
EGNN-Stacking-hard-kNN	0.9651	0.9659	0.9650
EGNN-Stacking-hard-SVM	0.9761	0.9764	0.9759
EGNN-Stacking-hard-RF	0.9685	0.9695	0.9685
EGNN-Stacking-hard-XGBoost	0.9753	0.9760	0.9754

on the premise of an insufficient understanding of the EGNN participants. Therefore, if the voting mode is used, whether soft or C3WV, is used needs to be balanced between classification accuracy and prediction time. According to the results, if the average number of packets per flow is 300 (in fact, the number of packets diverted is far more than this number), the throughput under voting modes can reach 1.226*Gbps*.

Further, we experimented with two modes and five meta-models in the stacking mode. In stacking, because the hyper-parameters of each meta-model also need to be determined, we use the *RandomizedSearchCV* function in the sklearn library to determine the optimal hyper-parameters. At the same time, the results shown in the experiment are also the results under the optimal hyper-parameters.

The first is the classification results in stacking hard mode. We also took the mean value of the classification results of three GNNs as the baseline to compare the classification effects of five different meta-models. The results are shown in Table 5.

The results show that the stacking hard mode significantly improves the classification effect, regardless of the base classifier used. The main reason for the stacking mode is that the meta-model of the stacking mode learns and corrects the misclassification of three EGNN participants again. When three participants have unique advantages, the stacking mode can better excavate the relationship between this feature perspective advantage and the real classification result from the feedback of GNN results of the first layer. Even though the GNN currently gives the wrong classification. In addition, the results show that the performance of the decision tree is the best under the ISCX dataset, while the performance of RF is reciprocal among several meta-models. This phenomenon has been thoroughly analyzed in the discussion.

Then we experimented with the classifications in stacking soft mode. The experimental results are shown in Table 6.

It can be seen from the results that stacking soft significantly improves the classification effect, and overall, the improvement of classification results is better than stacking hard. Similar to the advantage of voting soft for voting hard, the inputs to the meta-model of stacking soft are more complex and provide more information gain so that the meta-model can learn more features to correct the misclassification. In the ISCX dataset, RF has the best stacking

TABLE 6. Experimental results for the effect criteria on the ISCX VPN-nonVPN dataset of EGNN stacking soft mode.

Methods	Precision	Recall	F1-score
Baseline (3 WFF-GNNs average)	0.9129	0.9058	0.9032
EGNN-Stacking-soft-C4.5	0.9812	0.9815	0.9814
EGNN-Stacking-soft-kNN	0.9850	0.9854	0.9851
EGNN-Stacking-soft-SVM	0.9862	0.9867	0.9864
EGNN-Stacking-soft-RF	**0.9907**	**0.9901**	**0.9904**
EGNN-Stacking-soft-XGBoost	0.9900	0.9900	0.9900

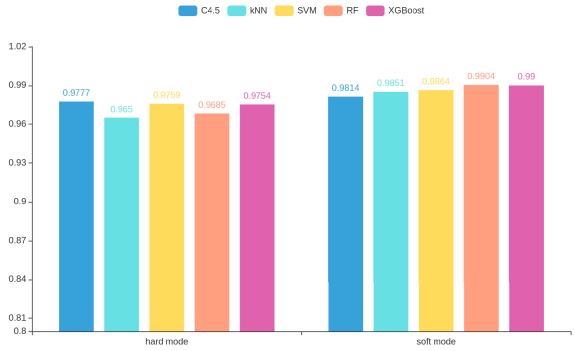


FIGURE 13. EGNN stacking modes classification F1-score among five different meta-models on ISCX VPN-nonVPN.

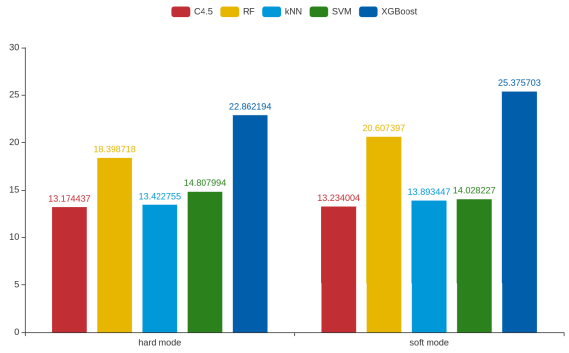


FIGURE 14. EGNN stacking modes training time cost among five different meta-models on ISCX VPN-nonVPN.

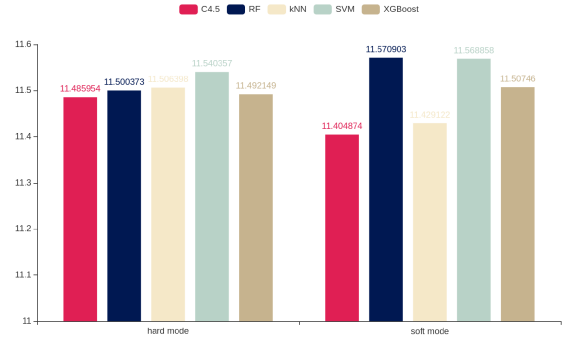


FIGURE 15. EGNN stacking modes prediction time cost among five different meta-models on ISCX VPN-nonVPN.

performance, and the decision tree has the worst stacking performance, and this is contrary to stacking hard. This phenomenon is also analyzed in depth subsequently.

In order to carefully evaluate the advantages of stacking hard and stacking soft, we first compared the improvement of the classification effects between stacking hard and stacking soft using the F1-score as the evaluation criterion. The bar diagram for comparison between the two is shown in Figure 13.

Then, using time consumption as the evaluation criteria, we compared the time consumed by stacking hard or soft modes on training and prediction. It is worth noting that since both rely on three GNN participants to conduct classification first, the time consumed includes the sample classification time of three participants. As stated in Section VI-A, the datasets used for stacking meta-model training and testing are further divided on top of the test sets of the first layer GNN classifier so that the input unit of stacking training is 3200 and the unit of the test is 800. The training time in stacking is shown in Figure 14.

The results show that the two ensemble learning meta-models (RF and XGBoost) require much more training than the three traditional machine learning meta-models. On the whole, the training time cost in soft mode is higher than in hard mode, which is also expected. However, the training time of SVM in soft mode is shorter than in hard mode. It may be because the boundary values in soft mode are all floating point numbers, and the calculation of hinge loss will be faster than the integer values with boundary mixed in hard mode (it is easier to find the hyperplane among the categories).

The prediction times of 10 models of two modes with 3,200,000 samples in stacking are shown in Figure 15.

It can be seen from the results that the prediction time of the models in hard mode and soft mode is uneven. Overall, the soft mode has a higher prediction time due to the complexity of the meta-model inputs. However, due to the gap between input features, the time required for soft mode prediction is shorter under the C4.5 and kNN models. It may be because the tree depth in C4.5 is smaller than in hard mode, and the mean value in kNN is calculated faster. Also converted by the results, the average throughput of stacking modes is $1.047 Gbps$.

Further, we crosswise the classification effect and prediction time in voting and stacking modes. We choose the most effective C3WV and soft modes in voting mode as the representatives, and in stacking mode, we choose C4.5 with the best effect in hard mode and RF with the best performance in soft mode as the comparison objects. Comparisons were made from precision, recall, EGNN model size (including 3 GNN participants), training time cost, and prediction time cost. A radar chart displays the comparison results, as shown in Figure 16.

It is worth noting that, in order to make each dimension of the radar map represent the larger, the better, the overall size

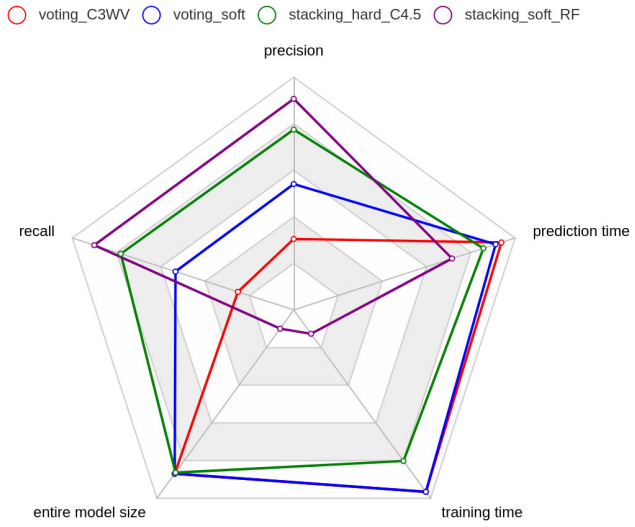


FIGURE 16. EGNN voting modes vs stacking modes in comprehensive assessments on ISCX VPN-nonVPN.

of the model is represented as the proportion of the remaining storage space under the 1MB storage unit to represent the module or agent in the actual intelligent switch device. In addition, the negative value is used directly as the training and prediction time benchmark.

It can be seen from the results that both voting mode and stacking mode have advantages and disadvantages. The classification effect of the stacking mode is better than voting. However, the more complex training process significantly increased training time cost, and colossal model memory occupation needs to be considered in the actual network environment. Therefore, it is difficult to conclude which mode of voting or stacking is better. As for the performance comparison, we conduct more in-depth experiments later in this paper.

2) EFFECTIVENESS EVALUATION ON CERNET-2022-SERVICE DATASET

We further carried out experiments in TCP and UDP scenarios of CERNET-2022-Service to prove its classification effectiveness in high-reliability networks and low-latency networks, respectively. The classification results in two scenarios are shown in Table 7. Note that the CERNET-2022-Service dataset, due to its complexity and strict segmentation of our experiment, can effectively represent the latest open-world actual network environment.

It can be seen from the results that the three WFF-GNN methods proposed by us all have satisfactory performance in the open-world environment, while the performance of other state-of-the-art models is significantly decreased compared with the ISCX dataset because they are not suitable for representing the features in the more complex open-world environment. WFF-GGNN has an excellent classification effect in both TCP and UDP data due to its advantage of Markov expression ability within the sequence, reflecting the superiority of WFF. Under TCP data, WFF-GCN is better than WFF-CapsGNN, while under UDP, WFF-CapsGNN is better. It also reflects the unique advantages of different WFF-GNN models under different data.

In order to further reflect the classification effect in each specific category, we use the Sankey figure to show the detailed classification of TCP and UDP data in this dataset, respectively. Figure 17 and Figure 18 are shown below.

It can be seen from the results that WFF-GGNN performs well in both TCP and UDP, with balanced classification distribution and few classification errors. In TCP, WFF-GCN and WFF-CapsGNN confuse and classify streaming and file upload obviously, while in UDP, P2P is misclassified seriously. The main reason is that in the CERNET dataset, streaming and file upload are both behaviors of continuous transmission of large amounts of data. Therefore, the local Markov properties are not as significant as the long-range Markov properties. At the same time, P2P, as a common means of downloading, can be understood as the reverse of file upload, namely download. WFF-GCN and WFF-CapsGNN have a certain deficiency in the expression ability of such large sequences because they pay more attention to the characteristics of local propagation.

Furthermore, under the CERNET-2022-Service dataset, we took the average classification capability of three GNN participants in TCP and UDP as the baseline and compared the voting and stacking in EGNN. The results are shown in Table 8.

The results show that EGNN performs better in an open-world environment than an older dataset like ISCX because EGNN is better suited to handling classification between categories with similar features in an open-world environment (with strong misclassification correction capability). In voting, the soft mode performs better in TCP, while the C3WV works better in UDP. In the stacking hard mode, XGBoost and RF models perform best in TCP and UDP, respectively. In stacking soft mode, RF outperforms other meta-models. This phenomenon shows that the ensemble learning meta-model has more advantages in classification ability in an open-world environment than the traditional machine learning meta-model.

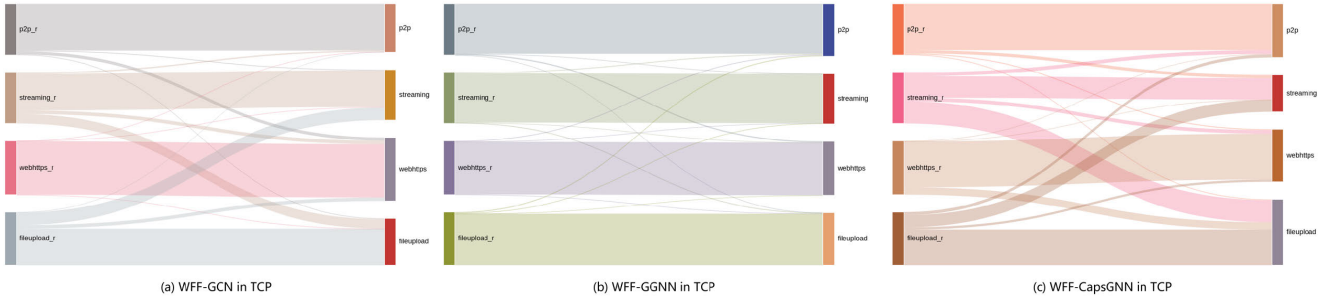
Then, we selected the same five dimensions as in the ISCX experiment to make a horizontal comparison of voting soft, voting C3WV, stacking hard, and stacking soft in the CERNET dataset. The results are shown in Figure 19.

In terms of the exact results, in TCP, if precision is to be pursued, the stacking mode should be used, whereas if speed is to be pursued, voting, especially C3WV mode, should be used. Under UDP, the stacking hard mode with RF has the most moderate performance, and other modes need to be selected according to the actual situation.

It is important to note that in TCP mode because the optimal meta-model RF of stacking soft is too large, it exceeds our 1MB limit. We manually set it to 0. Otherwise, the radar chart will be deformed.

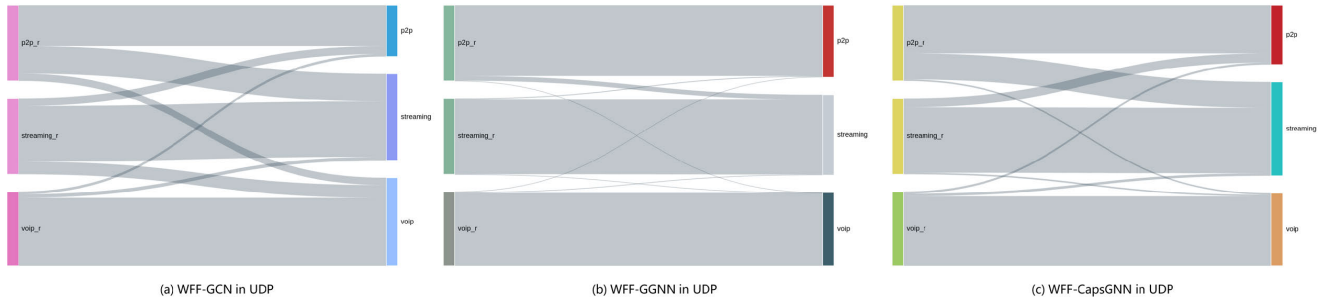
TABLE 7. Experimental results of single models for the effect criteria on the CERNET-2022-Service dataset (Include both TCP and UDP).

	TCP			UDP		
Methods	**Precision**	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
CNN	0.2770	0.2736	0.2753	0.2788	0.2517	0.2545
Deep Packet	0.3207	0.3063	0.3133	0.2225	0.1853	0.1422
FS-Net	0.8563	0.8402	0.8482	0.8342	0.7793	0.7775
LS-LSTM	0.8783	0.8781	0.8782	0.8750	0.8712	0.8727
WFF-GCN	0.8283	0.8287	0.8260	0.7375	0.7287	0.7218
WFF-GGNN	**0.9724**	**0.9722**	**0.9723**	**0.9697**	**0.9687**	**0.9687**
WFF-CapsGNN	0.7157	0.7147	0.7102	0.8210	0.8107	0.8093



(a) WFF-GCN in TCP (b) WFF-GGNN in TCP (c) WFF-CapsGNN in TCP

FIGURE 17. Sankey figure of each WFF-GNN model in TCP classification task on CERNET-2022-Service:(a) WFF-GCN, (b) WFF-GGNN, (c) WFF-CapsGNN.



(a) WFF-GCN in UDP (b) WFF-GGNN in UDP (c) WFF-CapsGNN in UDP

FIGURE 18. Sankey figure of each WFF-GNN model in UDP classification task on CERNET-2022-Service:(a) WFF-GCN, (b) WFF-GGNN, (c) WFF-CapsGNN.

In summary, EGNN in the open-world CERNET-2022-Service dataset has a considerable improvement compared to the existing state-of-the-art approach. Among them, the classification effect under TCP is improved, as shown in Figure 20.

3) OPEN-WORLD DEPLOYMENT-ORIENTED PERFORMANCE EVALUATION

In an open-world actual network environment, the model's accuracy, precision, and recall are only part of model usability. If we want to deploy the model in an actual network environment, the training cost of the model is also needed to be taken into account, as well as the predicted time/throughput, memory footprint.

First, we conducted experiments from the perspective of training time. The experiments here were mainly conducted around the EGNN stacking mode because it required additional training of the meta-model. In an open-world environment, if stacking mode is used as the ensemble mechanism for EGNN, we cannot predict what meta-models are better in the current data interval, so it is necessary to consider training all meta-models and obtain an optimal current model. Therefore, a control experiment was conducted for the overall stacking mode training time, as shown in Figure 21.

As can be seen from the results, the training time in stacking hard mode is shorter, but the training time in both modes increases with the amount of data. Furthermore, much time is spent on the optimal hyper-parameter grid search. Therefore, this needs to be considered when deployed in an

TABLE 8. Experimental results of EGNN modes for the effect criteria on the CERNET-2022-Service dataset (Include both TCP and UDP).

	TCP			UDP		
Methods	**Precision**	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
Baseline (3GNNs average)	0.8388	0.8385	0.8362	0.8427	0.8360	0.8330
EGNN-Voting-hard	0.9075	0.9080	0.9068	0.8459	0.8343	0.8320
EGNN-Voting-president	0.9212	0.9213	0.9204	0.8545	0.8417	0.8393
EGNN-Voting-soft	**0.9592**	**0.9590**	**0.9589**	0.9242	0.9180	0.9178
EGNN-Voting-C3WV	0.9559	0.9550	0.9549	**0.9341**	**0.9340**	**0.9339**
EGNN-Stacking-hard-C4.5	0.9813	0.9812	0.9812	0.9726	0.9717	0.9718
EGNN-Stacking-hard-RF	0.9726	0.9725	0.9725	**0.9743**	**0.9733**	**0.9735**
EGNN-Stacking-hard-kNN	0.9788	0.9788	0.9787	0.9675	0.9667	0.9668
EGNN-Stacking-hard-SVM	0.9726	0.9725	0.9725	0.9676	0.9667	0.9667
EGNN-Stacking-hard-XGBoost	**0.9838**	**0.9838**	**0.9837**	0.9694	0.9683	0.9684
EGNN-Stacking-soft-C4.5	0.9766	0.9762	0.9762	0.9868	0.9867	0.9866
EGNN-Stacking-soft-RF	**0.9925**	**0.9925**	**0.9925**	**0.9917**	**0.9917**	**0.9917**
EGNN-Stacking-soft-kNN	0.9788	0.9788	0.9787	0.9883	0.9883	0.9883
EGNN-Stacking-soft-SVM	0.9888	0.9888	0.9887	0.9783	0.9783	0.9783
EGNN-Stacking-soft-XGBoost	0.9900	0.9900	0.9900	0.9867	0.9867	0.9867

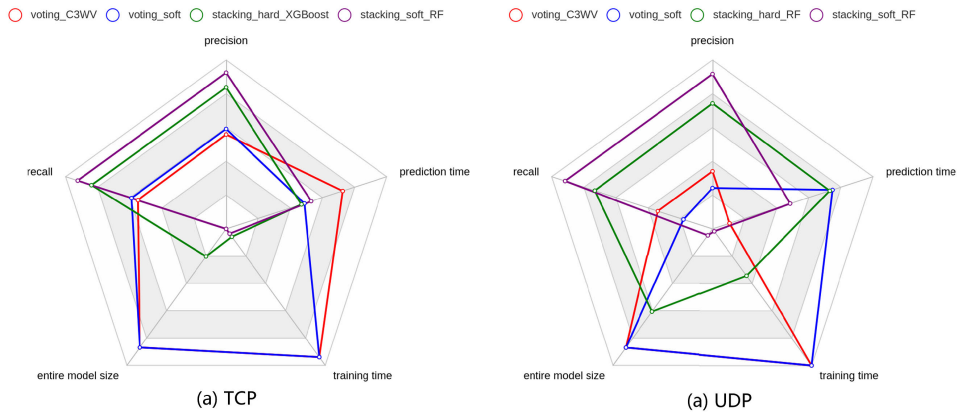


FIGURE 19. EGNN voting modes vs stacking modes in comprehensive assessments on CERNET-2022-Service.

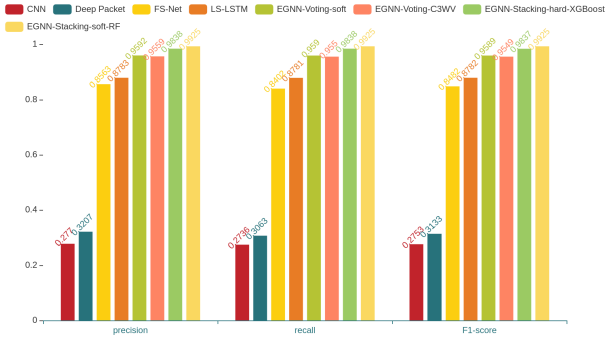


FIGURE 20. EGNN classification effect improvement on open-world CERNET-2022-Service dataset (TCP task).

open-world actual network environment, and voting mode is more applicable than stacking mode.

Next, we compared the memory usage of the models. First, we simultaneously compared three WFF-GNN and four state-of-the-art models in two datasets (three classification tasks). The results are shown in Table 9. It is worth noting that to reflect the minor differences between model sizes, we use bytes as units.

As can be seen from the results, even the largest (and with the best effect) WFF-GGNN model in the WFF-GNN models has a memory space occupation of only 126kB, much smaller than the state-of-the-art method. Although LS-LSTM has a good effect, especially in the ISCX dataset, its extremely large memory consumption makes it difficult to be applied in most classification scenarios. In addition, we can find that different classification tasks (the difference in the number of output categories) have little impact on the model's size, especially the model with a large size. This situation further shows the superiority of the WFF graphical expression method and the GNN classifiers.

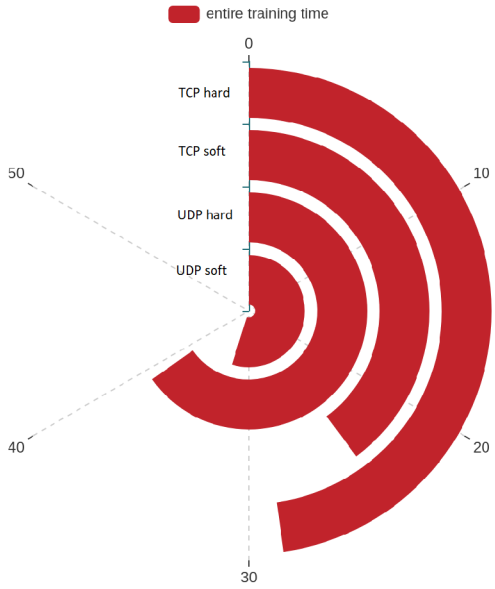


FIGURE 21. EGNN stacking mode entire training cost on open-world CERNET-2022-Service dataset.

TABLE 9. EGNN participants model size evaluation on two datasets in three classification tasks.

Methods	ISCX	TCP	UDP
CNN	332745	332745	332041
Deep Packet	2328127	2328127	2327397
FS-Net	43544832	43544832	43543133
LS-LSTM	154663507	154663507	154661459
WFF-GCN	4123	4123	4059
WFF-GGNN	127213	127213	127021
WFF-CapsGNN	5257	5257	5001

Furthermore, according to previous experiments, if the voting mode is adopted, the increase in the size of the overall model is tiny, generally within 5 to 6 bytes, which can be ignored compared with the size of GNN itself. However, because there is an additional meta-model in the stacking mode, its size cannot be ignored. Therefore, we further compared the model sizes of five models in two stacking modes in EGNN under three classification tasks, as shown in Figure 22.

As can be seen from the figure, kNN not only has an unsatisfactory classification effect but also has a large model size, so it is not applicable. RF and XGBoost, two ensemble learning meta-models, have good classification effects simultaneously but have tens or even hundreds of times of model size expansion compared with C4.5 and SVM. Therefore, if stacking mode is used, the impact of the meta-model size needs to be considered, especially in the context of tasks with numerous categories.

D. ANALYSIS

In general, the effect of single in the CERNET-2022-Service dataset is slightly lower than that in ISCX. It mainly lies

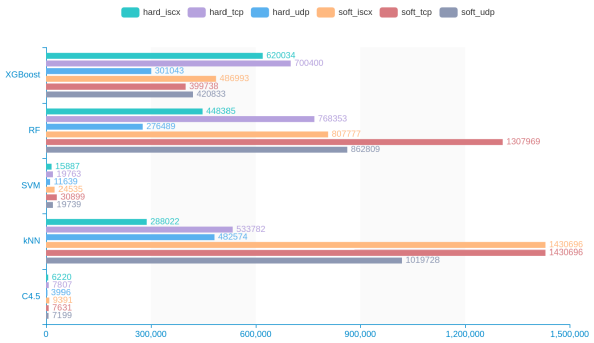


FIGURE 22. EGNN stacking mode meta-model size (with bytes) on two datasets in three classification tasks.

in the fact that this dataset contains new protocols (TLS-1.3, QUIC, DTLS) and applications, so the classification is more complicated. However, EGNN showed better classification accuracy in this dataset, indicating that EGNN has better applicability in the open-world environment and can better correct the misclassification caused by the similarity of features and the lack of effective learning of features.

For a single GNN, GCN has the fastest training speed of each epoch. Due to the complexity of the CapsGNN model and the lack of parallelization (graph convolution needs to be calculated concurrently), the training speed is slow, and the training procedure occupies plenty of memory resources. However, the convergence speed of CapsGNN is fast, and only about ten epochs are needed on average to achieve satisfactory results. GGNN has a good classification effect and fast convergence speed, but its operation is complex and requires many computing resources.

Although each of the three models has its own advantages and disadvantages, there is no perfect GNN model. However, EGNN effectively plays the advantages of the three models in various dimensions. No matter which dataset and classification task, EGNN greatly improves the classification effect, making it possible to classify encrypted traffic highly precisely in an open-world environment.

Next, we will analyze and discuss several specific details of the experiment.

1) WHY DO C4.5 AND RF SHOW OPPOSITE RESULTS IN STACKING HARD AND STACKING SOFT IN THE ISCX DATASET?

Based on a careful analysis of the generated meta-models, we find that the C4.5 decision tree in the stacking hard is good because its maximum depth is 54, while the maximum depth of trees in the RF is only 15. The meta-model input for hard mode is a class of integers, so getting RF to build multiple trees from different feature perspectives is difficult. The input in soft mode is the confidence floating point number, which effectively allows RF to build the tree from different angles.

2) EGNN USES THREE GNN MODELS AT THE SAME TIME. HOW TO ENSURE ITS TRAINING AND DEPLOYMENT EFFICIENCY IN THE ACTUAL NETWORK ENVIRONMENT?

EGNN is a good combination of the advantages of the three. It can train CapsGNN and GGNN, respectively, by using the CPU and GPU of the device simultaneously, effectively utilizing computing resources and significantly increasing efficiency. The experimental results show that the classification effect of EGNN is better than single GNN and various state-of-the-art baseline methods with only a tiny prediction time increasing.

In addition, different models can be deployed on different devices for parallel computation. In the experiment, we are serial so that the time consumption will be higher. There is only one data transfer problem to solve.

3) WHY ONLY USE 1MB AS A BASELINE FOR MODEL SIZE?

In the actual network environment of the open-world, encrypted traffic classification is usually obtained by bypass splitting or port mapping in the high-volume traffic scenario. In some cases, strict serial traffic classification is adopted to achieve real-time security protection. In either case, the service capability of the current network needs to be maintained, which means that system resources, especially memory resources, will be very precious (temporary storage of service data or cache of traffic information to be classified). Therefore, a large model size may affect regular services or traffic cache. In addition, the small model size enables the model to be effectively deployed on the portable chip or FPGA board further to realize the combination of hardware and software in classification and improve the classification efficiency.

4) WHY WASN'T GAT ADOPTED AS THE BASE MODEL FOR GNN?

Although GAT is a graph neural network based on the attention mechanism, in essence, the attention mechanism is not a particular neural network message transmission path. It is just a method to find feature advantages among existing features. Therefore, although we did not use GAT as the base model, we introduced the attention mechanism in all three WFF-GNN models, which was mainly realized by combining the linear layer with the activation function. It is also reflected in the three WFF-GNN network structure diagrams.

5) VOTING SOFT MODE USES THE MEAN VALUE OF THE SOFTMAX CONFIDENCE VECTOR AS THE VOTING PREFERENCE. CAN THE EFFECT BE IMPROVED IF THE MAXIMUM VALUE OF THE SOFTMAX VECTOR OF EACH CLASSIFIER IS USED FOR VOTING?

The answer is yes, but there are no specific advantages to this approach. If the highest value of the softmax confidence vector obtained by the different participants is taken as the result of the classification, then this is a way of falling between hard and soft. We name it semi-soft. We compare voting semi-soft with voting soft. The experimental results show that voting semi-soft can also correct misclassification. However, the effect was not as good as voting soft (the F1-score was 0.35% lower under the ISCX dataset), and the prediction time was 7.14% higher. Then this method is inferior to voting soft both in terms of classification effect and prediction speed. So we did not take voting semi-soft as an ensemble way to compare.

VII. CONCLUSION

In this paper, we deeply mine the Markov relationship paradigms in encrypted traffic and propose a novel flow graphical expression model named WFF. Compared with the existing graph construction methods, WFF can express the relevance relation of packets in encrypted traffic more completely and effectively. Furthermore, we proposed the WFF-GCN, WFF-GGNN, and WFF-CapsGNN models based on three GNN base models. They have their own unique feature expression advantage and data affinity. On top of this, we designed the EGNN architecture, which for the first time used GCN, GGNN, and CapsGNN to encrypted traffic classification, and it was also the first attempt that combined ensemble learning with GNNs. We apply the voting mode and stacking mode to EGNN, respectively, and present the president mode and C3WV mode based on the hard and soft modes of voting mode. In the stacking mode, multiple models are used in the hard and soft modes. We conducted detailed and sufficient experiments on the classification of individual GNN, the classification efficiency of EGNN, and the differences between various models within EGNN. The evaluation dimensions involved precision rate, recall rate, training time, prediction time, model size, and other dimensions. Experimental results show that the effect of our single WFF-GNN methods is better than multiple state-of-the-art methods. Furthermore, our EGNN model has outstanding classification effect improvement and is significantly superior to the existing state-of-the-art method in other dimensions, realizing the high-precision and high-efficiency encryption traffic classification in the open-world environment.

In future work, we will further explore the efficient operation of EGNN (involving further compression of features and multi-flow classification) and collaborative classification (including its federated deployment mode) and study more EGNN participants to expand its capabilities for more classification tasks.

REFERENCES

[1] Google. (2021). *HTTPS Encryption on the Web—Google Transparency Report*. [Online]. Available: https://transparencyreport.google.com/https/overview

[2] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and VPN traffic using time-related features," in *Proc. 2nd Int. Conf. Inf. Syst. Secur. Privacy*, 2016, pp. 407–414.

[3] Y. Shi, A. Ross, and S. Biswas, "Source identification of encrypted video traffic in the presence of heterogeneous network traffic," *Comput. Commun.*, vol. 129, pp. 101–110, Sep. 2018.

[4] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *Proc. IEEE Int. Conf. Intell. Secur. Informat. (ISI)*, Jul. 2017, pp. 43–48.

[5] Z. Chen, G. Cheng, B. Jiang, S. Tang, S. Guo, and Y. Zhou, "Length matters: Fast internet encrypted traffic service classification based on multi-PDU lengths," in *Proc. 16th Int. Conf. Mobility, Sens. Netw. (MSN)*, Dec. 2020, pp. 531–538.

[6] Z. Chen, G. Cheng, Z. Xu, S. Guo, Y. Zhou, and Y. Zhao, "Length matters: Scalable fast encrypted internet traffic service classification based on multiple protocol data unit length sequence with composite deep learning," *Digit. Commun. Netw.*, vol. 8, no. 3, pp. 289–302, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352864821000699

[7] C. Liu, L. He, G. Xiong, Z. Cao, and Z. Li, "FS-Net: A flow sequence network for encrypted traffic classification," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2019, pp. 1171–1179.

[8] T.-L. Huoh, Y. Luo, and T. Zhang, "Encrypted network traffic classification using a geometric learning model," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manag. (IM)*, May 2021, pp. 376–383.

[9] M. Shen, J. Zhang, L. Zhu, K. Xu, and X. Du, "Accurate decentralized application identification via encrypted traffic analysis using graph neural networks," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 2367–2380, 2021.

[10] G. Ciano, A. Rossi, M. Bianchini, and F. Scarselli, "On inductive-transductive learning with graph neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 2, pp. 758–769, Feb. 2022.

[11] J. Muehlstein et al., "Analyzing HTTPS encrypted traffic to identify user's operating system, browser and application," in *Proc. 14th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2017, pp. 1–6.

[12] S. Bagui, X. Fang, E. Kalaimannan, S. C. Bagui, and J. Sheehan, "Comparison of machine-learning algorithms for classification of VPN network traffic flow using time-related features," *J. Cyber Secur. Technol.*, vol. 1, no. 2, pp. 108–126, Apr. 2017.

[13] S. Fathi-Kazerooni and R. Rojas-Cessa, "GAN tunnel: Network traffic steganography by using GANs to counter internet traffic classifiers," *IEEE Access*, vol. 8, pp. 125345–125359, 2020.

[14] M. Song, J. Ran, and S. Li, "Encrypted traffic classification based on text convolution neural networks," in *Proc. IEEE 7th Int. Conf. Comput. Sci. Netw. Technol. (ICCSNT)*, Oct. 2019, pp. 432–436.

[15] H. Yao, C. Liu, P. Zhang, S. Wu, C. Jiang, and S. Yu, "Identification of encrypted traffic through attention mechanism based long short term memory," *IEEE Trans. Big Data*, vol. 8, no. 1, pp. 241–252, Feb. 2022.

[16] S. Cui, B. Jiang, Z. Cai, Z. Lu, S. Liu, and J. Liu, "A session-packets-based encrypted traffic classification using capsule neural networks," in *Proc. IEEE 21st Int. Conf. High Perform. Comput. Commun., IEEE 17th Int. Conf. Smart City, IEEE 5th Int. Conf. Data Sci. Syst. (HPCC/SmartCity/DSS)*, Aug. 2019, pp. 429–436.

[17] Y. Zeng, H. Gu, W. Wei, and Y. Guo, "Deep-full-range: A deep learning based network encrypted traffic classification and intrusion detection framework," *IEEE Access*, vol. 7, pp. 45182–45190, 2019.

[18] M. Lotfollahi, M. J. Siavoshani, R. S. H. Zade, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Soft Comput.*, vol. 24, no. 3, pp. 1999–2012, Feb. 2020.

[19] Y. Zhang, S. Zhao, J. Zhang, X. Ma, and F. Huang, "STNN: A novel TLS/SSL encrypted traffic classification system based on stereo transform neural network," in *Proc. IEEE 25th Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Dec. 2019, pp. 907–910.

[20] W. Maonan, Z. Kangfeng, X. Ning, Y. Yanqing, and W. Xiujuan, "CENTIME: A direct comprehensive traffic features extraction for encrypted traffic classification," in *Proc. IEEE 6th Int. Conf. Comput. Commun. Syst. (ICCCS)*, Apr. 2021, pp. 490–498.

[21] T. Shapira and Y. Shavitt, "FlowPic: Encrypted internet traffic classification is as easy as image recognition," in *Proc. IEEE INFOCOM Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2019, pp. 680–687.

[22] G. Baldini, "Analysis of encrypted traffic with time-based features and time frequency analysis," in *Proc. Global Internet Things Summit (GIoTS)*, Jun. 2020, pp. 1–5.

[23] H. Wu, L. Wang, G. Cheng, and X. Hu, "Mobile application encryption traffic classification based on TLS flow sequence network," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, Jun. 2021, pp. 1–6.

[24] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "MIMETIC: Mobile encrypted traffic classification using multimodal deep learning," *Comput. Netw.*, vol. 165, Dec. 2019, Art. no. 106944. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128619304669

[25] C. Dong, C. Zhang, Z. Lu, B. Liu, and B. Jiang, "CETAnalytics: Comprehensive effective traffic information analytics for encrypted traffic classification," *Comput. Netw.*, vol. 176, Jul. 2020, Art. no. 107258. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128619309466

[26] I. Guarino, G. Aceto, D. Ciuonzo, A. Montieri, V. Persico, and A. Pescapè, "Contextual counters and multimodal deep learning for activity-level traffic classification of mobile communication apps during COVID-19 pandemic," *Comput. Netw.*, vol. 219, Dec. 2022, Art. no. 109452. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128622004868

[27] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2017, *arXiv:1609.02907*.

[28] Y. Li, R. Zemel, M. Brockschmidt, and D. Tarlow, "Gated graph sequence neural networks," in *Proc. ICLR*, Apr. 2016, pp. 1–20. [Online]. Available: https://www.microsoft.com/en-us/research/publication/gated-graph-sequence-neural-networks/

[29] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. 6th Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, 2018, pp. 1–12. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[30] Z. Xinyi and L. Chen, "Capsule graph neural network," in *Proc. 7th Int. Conf. Learn. Represent.*, New Orleans, LA, USA, 2019, pp. 1–16. [Online]. Available: https://openreview.net/forum?id=Byl8BnRcYm

[31] B. Sun, W. Yang, M. Yan, D. Wu, Y. Zhu, and Z. Bai, "An encrypted traffic classification method combining graph convolutional network and autoencoder," in *Proc. IEEE 39th Int. Perform. Comput. Commun. Conf. (IPCCC)*, Nov. 2020, pp. 1–8.

[32] X. Ji and Q. Meng, "Traffic classification based on graph convolutional network," in *Proc. IEEE Int. Conf. Adv. Electr. Eng. Comput. Appl.*, Aug. 2020, pp. 596–601.

[33] R. Zhao et al., "Flow sequence-based anonymity network traffic identification with residual graph convolutional networks," in *Proc. IEEE/ACM 30th Int. Symp. Quality Service (IWQoS)*, Jun. 2022, pp. 1–10.

[34] T.-L. Huoh, Y. Luo, P. Li, and T. Zhang, "Flow-based encrypted network traffic classification with graph neural networks," *IEEE Trans. Netw. Service Manag.*, vol. 20, no. 2, pp. 1224–1237, Jul. 2023.

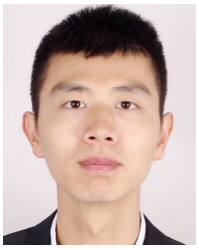
[35] F. Huang, G. Xie, and R. Xiao, "Research on ensemble learning," in *Proc. Int. Conf. Artif. Intell. Comput. Intell.*, vol. 3, Nov. 2009, pp. 249–252.

[36] Z. Chen, G. Cheng, Z. Xu, K. Xu, Y. Shan, and J. Zhang, "A3C system: One-stop automated encrypted traffic labeled sample collection, construction and correlation in multi-systems," *Appl. Sci.*, vol. 12, no. 22, p. 11731, Nov. 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/22/11731

ZIHAN CHEN (Member, IEEE) received the B.S. degree in software engineering from Central South University in 2017 and the Ph.D. degree in cyber security from Southeast University in 2023. He is currently a Post-Doctoral Researcher with the School of Cyber Science and Engineering, Southeast University. His major research interests include cyber security, encrypted traffic classification, encrypted traffic feature engineering, and deep learning. He works as a reviewer for multiple journals, such as IEEE Internet of Things Journal.

GUANG CHENG (Member, IEEE) received the B.S. degree in traffic engineering from Southeast University, Nanjing, China, in 1994, the M.S. degree in computer application from the Hefei University of Technology in 2000, and the Ph.D. degree in computer networks from Southeast University in 2003. He is currently a Full Professor with the School of Cyber Science and Engineering, Southeast University. He has authored or coauthored seven monographs and more than 100 technical papers, including top journals and top conferences. His research interests include network security, network measurement, and traffic behavior analysis. He is a Senior Member of CCF.

DANDAN NIU (Student Member, IEEE) received the B.S. degree in Internet of Things from Jiangnan University in 2021. She is currently pursuing the M.S. degree in cyberspace security with the School of Cyber Science and Engineering, Southeast University. Her major research interests include cyber security and encrypted traffic analysis.



XING QIU (Student Member, IEEE) received the B.S. degree in automation from Chongqing University in 2015 and the M.S. degree in communication networks and signal processing from the University of Bristol in 2018. He is currently pursuing the Ph.D. degree in cyberspace security with the School of Cyber Science and Engineering, Southeast University. His major research interests include security, encrypted traffic analysis, and artificial intelligence.

YUYU ZHAO (Member, IEEE) received the B.S. degree in software engineering from the Nanjing University of Science and Technology in 2016 and the M.S. and Ph.D. degrees in cyber security from Southeast University, Nanjing, China, in 2019 and 2023, respectively. He is currently a Lecturer with the School of Cyber Science and Engineering, Southeast University. His research interests include in-band telemetry, blockchain, and network processors.

YUYANG ZHOU (Member, IEEE) received the B.S. degree in electronic information engineering from the Nanjing University of Science and Technology in 2016 and the Ph.D. degree in cyber security from Southeast University in 2021. He is currently a Post-Doctoral Researcher with the School of Cyber Science and Engineering, Southeast University. He has published in some of the top journals and conferences, such as IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY (TIFS), IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, and ACM CCS. His major research interests include moving target defense, DDoS mitigation, security modeling, intrusion detection, and Android malware detection. He is involved as a reviewer and in technical program committees of several journals and conferences in the field.