Research on Network Traffic Classification Based on Graph Neural Network

Yue Han, Hong Dai*

***Abstract*—Network traffic classification is a critical concern in network security and management, essential for accurately differentiating among various network applications, optimizing service quality, and improving user experience. The exponential increase in worldwide Internet users and network traffic is continuously augmenting the diversity and complexity of network applications, rendering the Internet environment increasingly intricate and dynamic. Conventional machine learning techniques possess restricted processing abilities for network traffic attributes and struggle to address the progressively intricate traffic classification tasks in contemporary networks. In recent years, the swift advancement of deep learning technologies, particularly Graph Neural Networks (GNN), has yielded significant improvements in network traffic classification. GNN can capture the structured information among network nodes and extract the latent features of network traffic. Nonetheless, current network traffic classification models continue to exhibit deficiencies in the thoroughness of feature extraction. To tackle the problem, this research proposes a method for constructing traffic graphs utilizing numerical similarity and byte distance proximity by exploring the latent correlations among bytes, and it constructs a model, SDA-GNN, based on Graph Isomorphic Networks (GIN) for the categorization of network traffic. In particular, the Dynamic Time Warping (DTW) distance is employed to evaluate the disparity in byte distributions, a channel attention mechanism is utilized to extract additional features, and a Long Short-Term Memory Network (LSTM) enhances the stability of the training process by extracting sequence characteristics. Experimental findings on two actual datasets indicate that the SDA-GNN model surpasses other baseline techniques across multiple assessment parameters in the network traffic classification task, achieving classification accuracy enhancements of 2.19% and 1.49%, respectively.**

***Index Terms*—deep learning, graph isomorphism network, network traffic classification, multilayer perceptron**

I. Introduction

THE rapid growth of the Internet and the introduction of new network applications requiring substantial network resources, such as online music, live broadcasting, video, large-scale network games, and multifunctional application platforms, have rendered network management and security increasingly intricate. This poses considerable challenges to the efficiency and security of network services [1]–[3]. By analyzing user behavior data, network traffic categorization technology, a critical component of network control and information security management, can optimize network architecture, reduce security vulnerabilities, and enhance services for related network communications [4]–[5]. Designing effective algorithms for network traffic categorization is crucial for improving user experience, service quality, and the capability to differentiate among diverse network applications. Therefore, establishing a network flow classification method with a simplified framework that accurately identifies various types of network flows is essential.

Network traffic classification employs machine learning techniques that can be categorized into two main types: classical machine learning and deep learning. Given the increasing volume and complexity of network data, deep learning is becoming increasingly prominent in traffic identification and categorization [6]. Deep learning is widely utilized due to its independence from manually crafted features and its unique capability to autonomously extract deep features. An LSTM-based model with Hierarchical Attention Networks is proposed in the literature [7] for extensive experimentation using the ISCX VPN-nonVPN dataset. The significant progress in Graph Neural Networks research has enhanced the capability of related models to learn essential network characteristics [8]. The feasibility of using GNN for network traffic classification was demonstrated in the literature [9] by examining host behavior and employing a graph convolution model for network traffic classification. However, due to the absence of network traffic statistics, this approach proved inadequate for categorizing transport layer encrypted Internet traffic. The implementation of network traffic classification in GNN was demonstrated in the literature [10] by mining possible correlations between traffic bytes and transforming byte sequences into graphs. Nevertheless, this approach failed to account for the characteristics of byte location association, which limited its ability to handle complex byte sequences and led to incomplete feature extraction.

Synthesizing the issues identified in the above studies, this paper presents a method for constructing a byte positional correlation graph based on the byte distance proximity and numerical similarity traffic graph. This method incorporates DTW to accurately capture the positional relationships between bytes. We propose the SDA-GNN model to optimize the feature extraction process by employing two distinct Multilayer Perceptron (MLP) designs coupled with GIN. To enhance the model's classification accuracy, a novel channel attention mechanism is integrated to assign different weights to the features. To validate the superior performance of the

Manuscript received July 20, 2024; revised October 31, 2024. The research work was supported by the Fundamental Research Funds for the Liaoning Universities (NO. LJ212410146058).

Yue Han is a postgraduate student of University of Science and Technology Liaoning, Anshan, Liaoning, CO 114051, China. (e-mail: 296872286@qq.com).

Hong Dai* is a professor of School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, Liaoning, China. (corresponding author to provide phone: +086-186-4226-8599; fax:0412-5929818; e-mail: dear_red9@163.com).

SDA-GNN model over baseline approaches in the network traffic categorization task, experiments are conducted on two real-world datasets.

II. Related Work

A. Traditional Machine Learning

Conventional machine learning techniques include Bayesian Networks, Decision Trees, and Support Vector Machines, among others. A. W. Moore et al. [11] constructed 249 statistical features of network traffic for categorization by integrating various conventional machine learning techniques. Zhang et al. [12] proposed an improved K-means semi-supervised clustering algorithm capable of automatically optimizing model parameters, enabling fine-grained differentiation of unknown traffic while ensuring accurate classification of known traffic categories. M. Shafiq and colleagues [13] employed Bayesian Networks, C4.5 Decision Trees, Support Vector Machines, and Naive Bayes for network traffic categorization. The classification performance of traditional machine learning relies on the effectiveness of manually selected features, which require significant domain expertise. As network traffic categorization problems grow more complex, the development of relevant network traffic features becomes increasingly challenging.

B. Deep Learning

By converting bytes to grayscale images and classifying them using a one-dimensional convolutional neural network, W. Wang et al. [14] were the first to apply an end-to-end deep learning approach to network stream classification. However, the byte data contains excessive redundant information, resulting in low classification accuracy. Liu et al. [15] proposed FS-NET, a comprehensive classification model utilizing recurrent neural networks to address the challenge of encrypted traffic classification. Additionally, a multi-layer encoder-decoder architecture is implemented to thoroughly extract potential sequential features of stream data. Nevertheless, the encoder-decoder model architecture is overly complex. J. Busch et al. [16] collected directed edge characteristics of network traffic to construct a traffic graph, subsequently employing graph neural networks to detect and classify malware within network traffic. The deep learning model demonstrates exceptional stability and generalization capabilities, enabling it to autonomously extract intricate features from network traffic data and achieve precise classification.

III. Graph Construction

A. Byte Flow Graph Construction

We express it as a graph structure to efficiently categorize network traffic data. Specifically, we constructed a byte-flow graph $G=\{V,E,X\}$ .Here, V represents the set of nodes, where each node v_i corresponds a $byte_i$ in the network traffic data, located in the header or load portion of the packet. E denotes the set of edges, and an edge $e_{i,j}$ denotes the connection relationship between nodes if the byte pair association is within the threshold. In this case, then an edge $e_{i,j}$ is established between $byte_i$ and $byte_j$. X represents the set of features and we have used the raw values of the bytes as the features x_i of the nodes. To better represent the relationship between bytes, we introduce an adjacency matrix A ,where $a_{i,j} \in A$ represents the connections between nodes. In particular, the element $a_{i,j}$ of the adjacency matrix A indicates whether there is an edge connection between node v_i and node v_j. If $a_{i,j}=1$, an edge exists between node v_i and node v_j. Otherwise, no edge connection exists.

B. Byte Correlation

We assessed the degree of linkage between bytes by calculating the frequency of byte pair occurrences in the data window through a sliding window technique. Concretely, for each pair of $byte_i$ and $byte_j$, the frequency of their occurrence f is incremented by 1 if they occur simultaneously in each data window. If a byte pair appears more than once in the same window, for example $(byte_i, byte_j, byte_i)$, the frequency of its occurrences is aggregated. The window size is established as $window\_size$; hence, the maximum frequency of a byte pair occurring within the same window is calculated as follows:

$$F_{\max} = \left\lfloor \frac{window\_size}{2} \right\rfloor \times (window\_size - \left\lfloor \frac{window\_size}{2} \right\rfloor) \tag{1}$$

The sequence length is defined as $len(list)$. The aggregate quantity of windows is expressed as:

$$windows = len(list) - window\_size + 1 \tag{2}$$

Ultimately, by aggregating the frequencies of occurrences across all windows, the cumulative frequency of byte pair occurrences is derived as:

$$F(byte_i, byte_j) = \sum_{n=1}^{windows} f_n \tag{3}$$

The distance proximity computation method quantifies the degree of similarity between byte pairs based on their occurrence frequency. In this method, by default, the initial distance $default\_dis$ of each byte pair is the maximum distance at which the byte pair may exist in the sequence, i.e., the maximum frequency at which it may occur is $F_{\max} \times windows$. The distance proximity of the byte pair is defined as:

$$d(byte_i, byte_j) = \frac{1}{default\_dis - F(byte_i, byte_j)} \tag{4}$$

As shown in Fig. 1, the space between byte pairs diminishes as frequency increases, indicating a stronger correlation between the bytes.

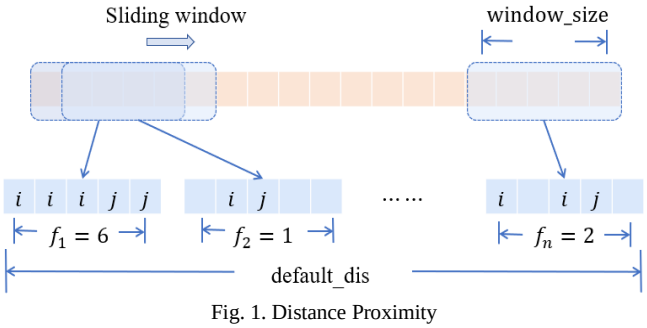


Fig. 1. Distance Proximity

The numerical similarity method quantifies the similarity in byte alteration trends between byte pairs. This method

documents the increment and decrement characteristics, or first-order differences, of each byte within a window. In particular, the first-order difference within the window is calculated as follows:

$$\delta(byte_i)=\begin{cases} byte_i - byte_{i-1}\ ,\ i\neq 0 \\ 0\qquad\quad ,\ i=0 \end{cases} \tag{5}$$

When a byte repeats within the window, its increment and decrement properties are averaged based on its frequency by aggregating all first-order differences. The equation is as follows:

$$\delta\left(byte_i\right)=\frac{1}{n}\sum_{1}^{n}\delta_n\left(byte_i\right) \tag{6}$$

The numerical similarity of each byte pair is calculated as follows:

$$s\left(byte_i,byte_j\right)= \begin{cases} \dfrac{1}{\left|\delta\left(byte_i\right)-\delta\left(byte_j\right)\right|}\ ,\ \left|\delta\left(byte_i\right)-\delta\left(byte_j\right)\right|>1 \\ 1\qquad\qquad\quad ,\ \left|\delta\left(byte_i\right)-\delta\left(byte_j\right)\right|\le 1 \end{cases} \tag{7}$$

By tracking the increase or decrease in each byte's attributes and analyzing byte changes, the trend and magnitude of changes in adjacent byte features can be captured.

The correlation of byte pairings is computed using distance similarity and value similarity, resulting in $c=d\times s$, yielding the correlation dictionary for all byte pairs, $Corr=\left\{byte_0,byte_1:c_{0,1},byte_1,byte_2:c_{1,2}...,byte_i,byte_j:c_{i,j}\right\}$. Using the "min-max normalization" method, the correlation is linearly transformed into the range [0,1], where the minimum value becomes 0 and the maximum value becomes 1. The process is calculated as follows:

$$c_{i,j}\_norm=\frac{c_{i,j}-c\_\min}{c\_\max-c\_\min} \tag{8}$$

The correlation dictionary is updated with the normalized value $c_{i,j}\_norm$ to derive the final correlation $c(byte_i,byte_j)$ for each byte pair.

A threshold $\tau=0.5$ is set for byte correlation. if $c<\tau$, the correlation between two bytes is deemed insufficient, resulting in the exclusion of the byte pair, thus obtaining the adjacency matrix of byte edges:

$$a_{i,j}=\begin{cases} 1\ ,\ c(byte_i,byte_j)>\tau \\ 0\ ,\quad otherwise \end{cases} \tag{9}$$

C. Location Correlation

Dynamic Time Warping [17] (DTW) is a method employed to assess the similarity between two sequences. For two sequences $X=[x_1,x_2,\cdots,x_n]$ and $Y=[y_1,y_2,\cdots,y_n]$, the DTW algorithm pairs each byte in sequence X with each byte in sequence Y using elasticity and calculates the distance of all byte pairs between them with the following formula:

$$D(i,j)=\left\|\left(x_i-y_i\right)^2\right\| \tag{10}$$

After that, a path is determined by dynamic programming to minimize the cumulative distances of the byte pairs along that path, as illustrated in Fig. 2. The equation is as follows:

$$\begin{aligned} C(i,j)=&D(i,j)\\ &+\min\left\{C\left(i,j-1\right),C(i-1,j),C(i-1,j-1)\right\} \end{aligned} \tag{11}$$

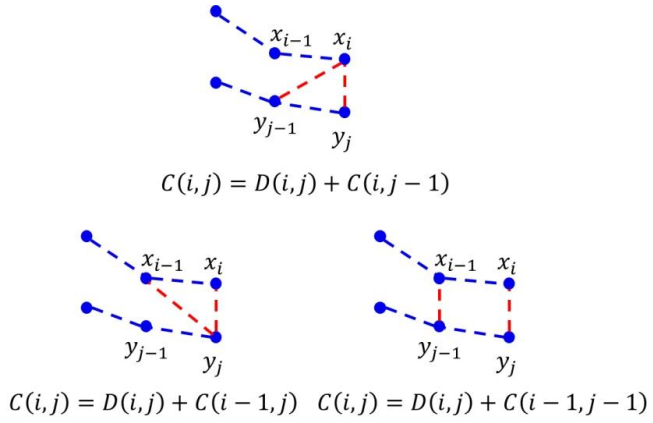


Fig. 2. DTW algorithm dynamic programming

The DTW distance between the two sequences, indicating their similarity, is the sum of byte pair distances along the path with minimal cumulative distance.

$$DTW(X,Y)=C(n,m) \tag{12}$$

The distance matrix is derived by computing the DTW distances between sequences, which are then normalized and constructed into a graph-structured adjacency matrix, as illustrated in Fig. 3. In this matrix, closer feature pairs yield values near 0, while more distant pairs approach 1, quantifying the disparity between byte distributions.

1	0.893	…	…	0.176	0.686
0.893	1	…	…	0.085	0.036
…	…	1	…	…	…
…	…	…	1	…	0.567
0.176	0.085	…	…	1	0.283
0.686	0.036	…	0.567	0.283	1

Fig. 3. Distance correlation matrix

The DTW distance quantifies similarity between different feature distributions, effectively captures morphological characteristics, measures feature distributions, and stabilizes the training process.

IV. Model

A. Graph Isomorphism Network

Graph Isomorphism Network (GIN) [18] represents a category of GNNs [19] that systematically aggregates the features of nodes and their neighbors, effectively capturing nuanced similarities among nodes and thereby allowing the network to discern distinct graph structures. The primary approach involves employing a Multilayer Perceptron (MLP)

to aggregate features from nodes and their neighbors, as illustrated in formula (13).

$$h_v^{(k)} = MLP^{(k)}\left(\left(1+\varepsilon^{(k)}\right)\cdot h_v^{(k-1)} + \sum_{u\in N(v)} h_v^{(k-1)}\right) \quad (13)$$

$h_v^{(k)}$ denotes the feature vector of node v at layer k. $MLP^{(k)}$ is a MLP utilized to acquire the feature representation of a node. $\varepsilon^{(k)}$ is a trainable parameter for modulating the significance of features for itself and its neighbors. $N(v)$ represents the collection of adjacent nodes to v.

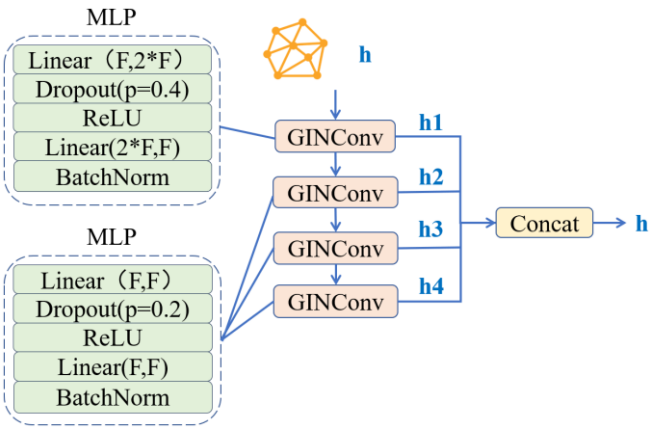


Fig. 4. GIN model structure

The GIN model, illustrated in Fig. 4, aggregates node properties layer by layer with a four-layer GINConv. It subsequently integrates the features to formulate a more resilient representation, improving classification performance. Two distinct MLP structures were intentionally incorporated into the GIN network design to enhance feature extraction. The initial layer GINConv employs a more complex MLP structure characterized by a higher-dimensional hidden layer and an increased dropout parameter. This strategy mitigates the risk of overfitting by using higher dropout values and capturing more complex characteristics during the initial stage. A three-layer GINConv is then used to further refine complex features and reduce information loss, employing a simpler MLP architecture with reduced dropout parameters. This hierarchical MLP architecture enhances the model's feature extraction capabilities while addressing regularization and information retention, thereby improving generalization and overall performance.

B. *Channel Attention Mechanism*

This research employs the channel attention mechanism [20], which weights and aggregates input features to suppress irrelevant ones and emphasize pertinent ones. The input data has the shape $(batch\_size, C, F)$, indicating that there are C channels in a network flow, each with F dimensions. The weights of each channel, structured as $(batch\_size, C, 1)$, are derived by initially mapping each channel to a weight value via a linear layer, followed by normalizing these weight values with a Softmax function. Subsequently, the input features are added to the feature weights, resulting in the final feature vector X_{att}, as shown in equation (14).

$$X_{att} = \sum_{i=1}^{N}\left(soft\max\left(fc(x_i)\right)\cdot x_i\right) \quad (14)$$

In contrast to maximum and average pooling methods, the Softmax weight assignment technique can dynamically modify channel weights, highlighting the significance of each channel and improving the model's performance in network traffic classification tasks. This differs from the conventional channel attention mechanism.

C. *SDA-GNN*

This research presents an SDA-GNN graph neural network model founded on GIN for the task of network traffic classification, as illustrated in Fig. 5.

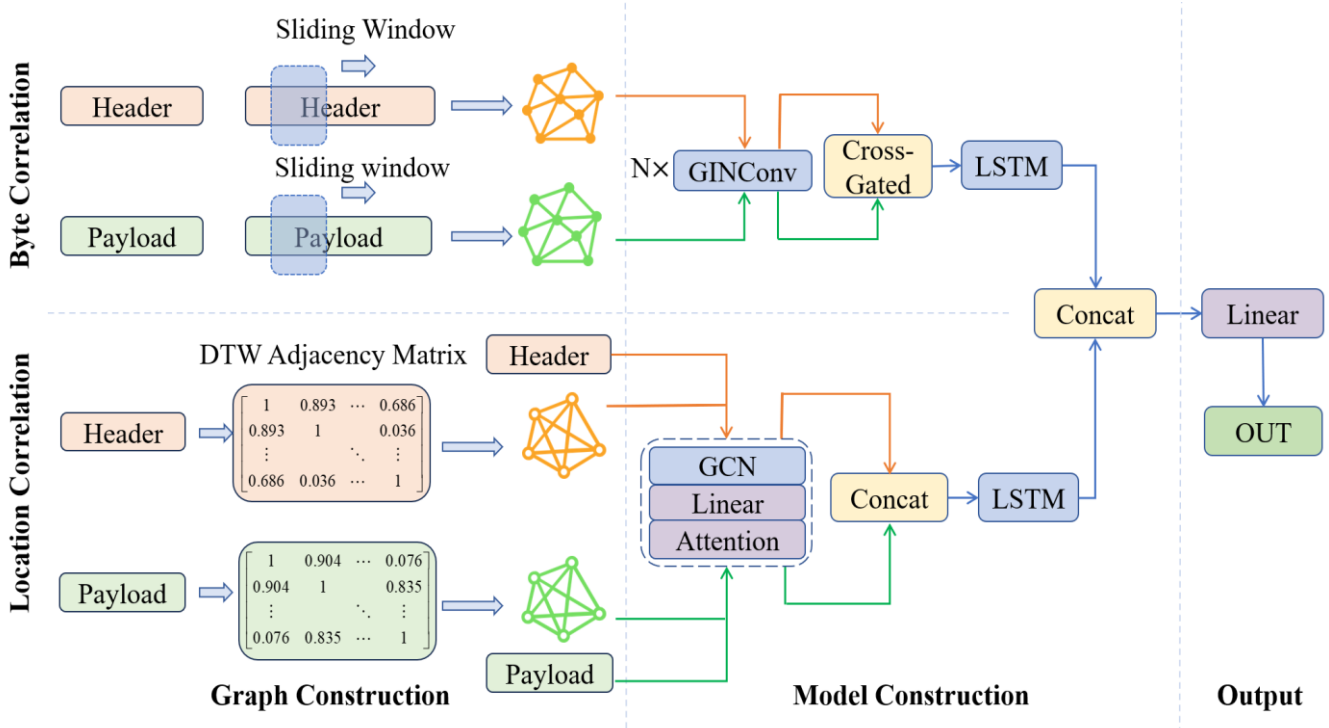


Fig. 5. SDA-GNN model structure

The model consists of two modules: one that addresses byte correlation and another that addresses positional correlation, thereby comprehensively capturing the features of network traffic. Firstly, for the byte correlation module, the model utilizes a sliding window mechanism to process the header and payload data of network traffic packets, constructing a byte traffic graph. The GIN module separately extracts the graph features of the header and payload data. This step captures local dependencies among different bytes, resulting in a more expressive feature representation. During the feature fusion phase, a cross-gating mechanism [10] is introduced to effectively integrate the features from the header and payload, allowing the model to adaptively focus on the significance of each feature. The fused features are further processed through LSTM [21] to extract temporal information, enhancing the model's ability to capture time-dependent characteristics in network traffic data.

The second module is the positional correlation module, which constructs a correlation matrix based on DTW to generate a positional correlation graph that reflects the spatial distribution of bytes. The positional correlation graph, along with the original header and payload data, is processed by GCN[22] to extract spatial features. This allows the model to capture the spatial characteristics of the data, while a channel attention mechanism assigns weights to different nodes, generating a more discriminative feature representation. Subsequently, the features are further processed by LSTM to capture additional temporal information, thereby strengthening the model's ability to represent sequential dependencies in the data. Finally, the output features from the byte correlation module and the positional correlation module are concatenated and fed into a linear layer for classification, producing the network traffic classification results.

V. EXPERIMENTS

A. Datasets and Preprocessing

The public traffic dataset ISCX VPN-nonVPN [23] is specified for the experiment and comprises two datasets: VPN and nonVPN. VPN traffic is transmitted through an additional encrypted tunnel, which differentiates its traffic properties from those of nonVPN traffic. Presented in Table I, the dataset comprises six distinct user behavior categories: Chat, Email, File, P2P, Streaming, and VoIP.

TABLE I
ISCX VPN-NONVPN DATASET CATEGORIES

Type	Application
Chat	ICQ, AIM, Skype, Facebook, Hangouts
Email	SMPTS, POP3S, IMAPS
File	Skype, FTPS, SFTP
P2P	uTorrent, Transmission
Streaming	Vimeo, Youtube
VoIP	Facebook, Skype, Hangouts

This process performs data extraction on the header and payload data of the dataset. The header data consists of five fields: source IP address, destination IP address, source port number, destination port number, and protocol type, which provide essential information about the network traffic. The payload data refers to the section of a network packet that includes the actual data transmitted by the sender. Together, these components are crucial for categorizing network traffic, as they contain most of the relevant information.

All header and payload data in the dataset have been sanitized, populated, and truncated. First, the data undergoes a cleaning process to eliminate empty packets and anomalous data. Then, data fields are filled and truncated to ensure uniform packet lengths, facilitating subsequent processing and analysis. When a specified threshold is exceeded, the first 12 bytes of the header and the portion from the 24*th* byte onward are extracted. Typically, the first 12 bytes contain the source IP address, destination IP address, and protocol type, while bytes from the 24*th* onward include port information. By removing unnecessary header fields, data dimensions are reduced, enhancing the efficiency of further processing. Finally, the dataset is partitioned into a training set of 90% and a test set of 10%.

B. Evaluation Criteria

The experiments use four metrics, Accuracy (ACC), Precision (PR), Recall (RC), and F1-score (F1), to evaluate the performance of the classification model. The formulas are shown in equations (15-18). Given the varying sample sizes across categories, a weighted average is used for the final evaluation to provide a more comprehensive performance assessment.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{15}$$

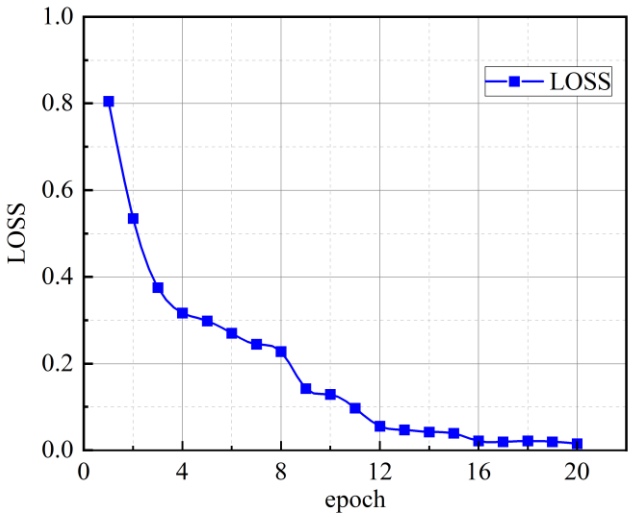
$$PR = \frac{TP}{TP + FP} \tag{16}$$

$$RC = \frac{TP}{TP + FN} \tag{17}$$

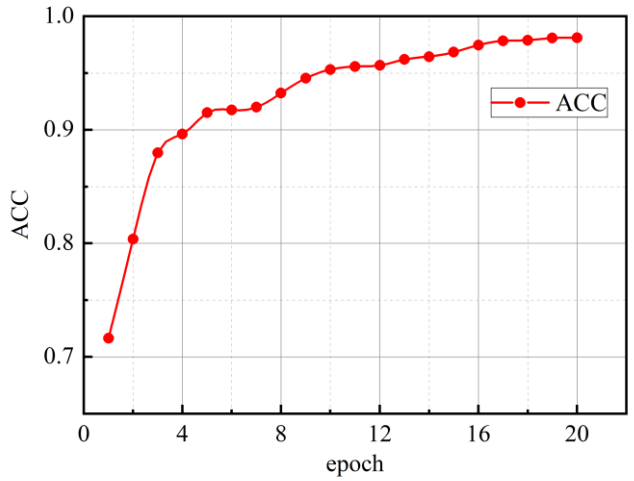
$$F1 = \frac{2 \cdot PR \cdot RC}{PR + RC} \tag{18}$$

C. Loss and Accuracy

To validate the efficacy of the model, we conducted experiments on the dataset, as depicted in Fig. 6. These experiments show changes in accuracy and loss during the model's training phase, with evaluations conducted every 5 training rounds.The results indicate a rapid increase in accuracy as the number of training rounds grows, suggesting that the model quickly achieves high classification performance within fewer training rounds. Subsequently, the accuracy stabilizes above 98%, demonstrating the model's strong capacity for learning and precision in network data classification. Additionally, the loss shows a sharp decline in the initial stages of training, particularly in the first few rounds, and then stabilizes. This trend suggests that the model successfully reduces errors during optimization and demonstrates the distinct feature of fast convergence. Furthermore, the stable accuracy and low loss values over extended training rounds indicate the model's resilience against overfitting, suggesting its adaptability to varying traffic patterns. Overall, the experimental findings confirm the model's efficacy and robustness in classifying network traffic and show that the model has strong generalization ability and high learning efficiency.



(a) Loss curve



(b) Accuracy curve

Fig. 6. Accuracy and loss variation

D. Comparison Experiments

In order to assess the efficacy of the model, we conduct a comparative analysis between the SDA-GNN model and many traditional network traffic classification algorithms. The models comprise the machine learning models GRAIN[24], FAAR[25], ETC-PS [26], and the deep learning models FS-NET [14], EDC[27], and TFE-GNN [10]. The models as mentioned above are detailed as follows:

1) GRAIN: A fine-grained network traffic classification technique that utilizes seven unique statistical features derived from packet payload length. It achieves multi-label classification by combining two random forest classifiers.
2) FAAR: A fast identification application model based on encrypted traffic that extracts trend characteristics from traffic generated by application activities and uses machine learning to identify categories.
3) ETC-PS: A method for classifying encrypted flows using path signatures is proposed. This approach generates traffic paths by analyzing sequences of session packet durations, then calculates multi-scale path features and applies conventional machine learning classifiers for classification.
4) FS-NET: An end-to-end classification model that uses multi-layer GRU stacking capture representative characteristics from the original stream. It utilizes a multi-layer encoder-decoder architecture to explore the stream's latent sequence features in greater depth.
5) EDC: A multi-label network traffic dataset is constructed, and a digital behavior-based classification framework is proposed to classify network traffic using feed-forward neural networks.
6) TFE-GNN: A byte-level traffic graph construction method based on point-by-point mutual information, and utilizes graph neural networks for feature extraction.

The results of the comparison are summarized in Table II. The table shows that the SDA-GNN model outperforms other baseline models across all evaluation criteria in comparative experiments. Deep learning methods, particularly GNN models, outperform conventional machine learning models in classification accuracy. The SDA-GNN model achieved classification accuracies of 98.1% and 91.89% in trials conducted on the ISCX-VPN and ISCX-nonVPN datasets, respectively. These results exceed those of the TFE-GNN model, which is currently considered the optimal model, by 2.19% and 1.49%, respectively.

The standardized encapsulation and encryption process of VPN traffic imparts a high degree of uniformity, as observed in our experiments. Conversely, nonVPN traffic is characterized by greater diversity, encompassing a more comprehensive array of activities and applications. This diversity introduces noise into the dataset, thereby increasing the model's difficulty in achieving accurate classification predictions.

TABLE II
PERFORMANCE COMPARISON OF SDA-GNN AND BASELINE MODELS

Dataset	Metric	GRAIN	FAAR	ETC-PS	FS-NET	EDC	TFE-GNN	**SDA-GNN**
ISCX-VPN	ACC (%)	81.29	83.63	88.89	92.98	78.36	95.91	**98.10**
	PR (%)	80.77	82.24	88.03	92.63	77.47	95.26	**98.19**
	RC (%)	81.09	84.04	89.37	92.11	81.08	95.93	**98.10**
	F1 (%)	80.27	82.91	88.51	92.34	78.88	95.36	**98.08**
ISCX-nonVPN	ACC (%)	66.67	73.74	72.73	76.26	69.70	90.40	**91.89**
	PR (%)	65.32	75.09	74.14	76.85	71.53	93.16	**94.02**
	RC (%)	66.64	71.21	71.33	75.34	70.00	91.90	**92.73**
	F1 (%)	65.67	72.52	72.08	75.55	69.78	92.40	**92.92**

E. Ablation Experiments

An ablation test is conducted on the model to validate the efficacy of each module. The experimental results are presented in Table Ⅲ, where the column labeled 'w/o' indicates that the module is not used.

TABLE III
ABLATION EXPERIMENTS OF SDA-GNN

Dataset	Model	ACC(%)	F1(%)
ISCX-VPN	SDA-GNN w/o	**98.10**	**98.08**
	SDA-GNN w/o Correlation	96.32	96.28
	SDA-GNN w/o Location	97.23	97.20
	SDA-GNN w/o Attention	97.74	97.70
	SDA-GNN w/o GIN	97.36	97.28
ISCX-nonVPN	SDA-GNN w/o	**91.89**	**92.92**
	SDA-GNN w/o Correlation	90.91	91.63
	SDA-GNN w/o Location	90.98	91.80
	SDA-GNN w/o Attention	91.19	92.11
	SDA-GNN w/o GIN	91.15	92.07

The table shows that the computation of byte correlation is crucial to the model. The experimental accuracies on the two datasets, without considering the calculation of byte correlation, are 96.32% and 90.91%, respectively. These results are 1.78% and 0.98% lower than the reported accuracies of 98.1% and 91.89% in the improved model proposed in the main article. Excluding the byte positional correlation graph module results in experimental accuracies of 97.23% and 90.98% on the two datasets, respectively, which are 0.87% and 0.91% lower than those of the improved model presented in the main article. Moreover, the model's performance is further enhanced by GIN and the channel attention mechanism. When the channel attention mechanism is removed from the improved model, the accuracies on the two datasets decrease to 97.74% and 91.19%, respectively. Similarly, when the GIN in the enhanced model is substituted with a classical GNN, the accuracies on the two datasets decrease to 97.36% and 91.15%, respectively. The experimental results indicate that each module is crucial for enhancing the model's performance.

F. Hyperparameters Analysis

The results of the hyperparameter testing used to determine the byte relevance threshold are shown in Fig. 7. Insufficiently setting the threshold may lead to an excessive number of weakly connected byte pairs being considered, hence increasing the noise level. Setting the threshold too low may lead to an excessive number of weakly connected byte pairs being considered, thereby increasing the noise level. Including a large amount of irrelevant information in the graph can have a detrimental effect on the model's learning progress and classification performance. Conversely, if the threshold is set too high, the model may lack sufficient data to accurately capture the characteristics in the data throughout the deep learning process, thereby diminishing accuracy. Thus, selecting an appropriate threshold is crucial for the model's classification performance. Achieving a balance between retaining enough information and reducing noise is essential to ensure that the model can accurately capture valuable features without being excessively affected by noisy input.

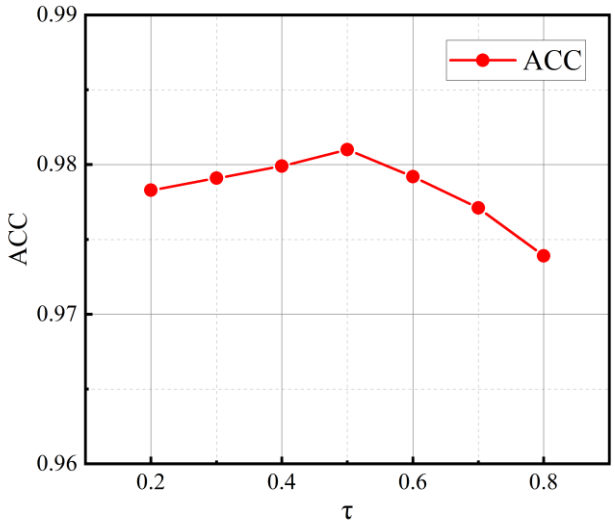


Fig. 7. Hyperparametric experimental results

Experimental results indicate that the model achieves its highest classification accuracy of 98.1% when the threshold is set at $\tau = 0.5$. A lower threshold yields better results than a higher threshold due to its ability to retain more feature information without introducing excessive noise. The diversity and richness of characteristics significantly impact the improvement of classification performance.

VI. Conclusion

This work presents a technique for constructing traffic graphs using numerical similarity and byte distance proximity. We introduce DTW to measure variations in byte distributions. Additionally, we extract features using the channel attention mechanism. Finally, we develop an SDA-GNN model for network traffic classification based on GIN. Empirical results on the ISCX VPN-nonVPN dataset indicate that the SDA-GNN model improves the accuracy of network traffic classification by 2.19% and 1.49% compared to the currently optimal TFE-GNN model.

Our future research endeavor aims to enhance feature representation by substituting the existing single-head attention mechanism with a multi-head attention mechanism in Transformer. Furthermore, incorporating an appropriate level of noise in the training set will improve the model's resilience and generalization. Implementing these adjustments will enhance the model's effectiveness in network data categorization tasks.

References

[1] D. Jiang, Y. Wang, Z. Lv, S. Qi, and S. Singh, "Big data analysis based network behavior insight of cellular networks for industry 4.0 applications," *IEEE Transactions on Industrial Informatics*, vol.16, no.2, pp.1310-1320, 2019.

[2] A. Domański, J. Domańska, T. Czachórski, J. Klamka, J. Szyguła, and D. Marek, "The IoT gateway with active queue management," *International Journal of Applied Mathematics and Computer Science*, vol. 31, no. 1, pp.165–178,2021.

[3] D. Nunez-Agurto, W. Fuertes, L. Marrone, and M. Macas, "Machine Learning-Based Traffic Classification in Software-Defined Networking: A Systematic Literature Review, Challenges, and Future Research Directions," *IAENG International Journal of Computer Science*, vol.49, no.4, pp.1002-1015, 2022.

[4] H. Tahaei, F. Afifi, A. Asemi, F. Zaki, and NB. Anuar, "The rise of traffic classification in IoT networks: A survey," *Journal of Network and Computer Applications*, vol.154, pp.102538, 2020.
[5] A. Dainotti, A. Pescape, and KC. Claffy, "Issues and future directions in traffic classification," *IEEE Network*, vol.26, no.1, pp.35-40,2012.
[6] SM. Rachmawati, DS. Kim, and JM. Lee, "Machine learning algorithm in network traffic classification," *2021 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, pp.1010-1013, 2021.
[7] 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 Transactions on Big Data*, vol.8, no.1, pp. 241-252, 2022.
[8] SR. Vinta, G. Singh, M. Kaur, A. Kaur, SF. Norbaevna, and A. Kumar, "Auto-encoder and Graph Neural Networks-Based Hybrid Model for Link Prediction on Complex Network," *International Conference on Intelligent Computing and Networking. Singapore: Springer Nature Singapore*, pp.483-492, 2023.
[9] X. Ji, and Q. Meng, "Traffic classification based on graph convolutional network,"*2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*. IEEE, pp.596-601, 2020.
[10] H. Zhang, L. Yu, X. Xiao, Q. Li, F. Mercaldo, X. Luo, et al., "TFE-GNN: A temporal fusion encoder using graph neural networks for fine-grained encrypted traffic classification," *Proceedings of the ACM Web Conference*, pp. 2066-2075, 2023.
[11] A. W. Moore, and D. Zuev, "Internet traffic classification using bayesian analysis techniques," *Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*. pp.50-60, 2005.
[12] J. Zhang, X. Chen, Y. Xiang, W. L. Zhou, and J. Wu, "Robust network traffic classification," *IEEE/ACM Transactions on Networking*, vol. 23, no.4, pp.1257-1270,2015.
[13] M. Shafiq, X. Yu, and D. Wang, "Network traffic classification using machine learning algorithms," *Advances in Intelligent Systems and Interactive Applications: Proceedings of the 2nd International Conference on Intelligent and Interactive Systems and Applications (IISA2017)*, Springer International Publishing, pp.621-627,2018.
[14] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks,"*2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*, IEEE, pp. 43-48, 2017.
[15] C. Liu, L. He, G. Xiong, Z. Cao, and Z. Li, "Fs-net: A flow sequence network for encrypted traffic classification," *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, IEEE, pp.1171-1179, 2019.
[16] J. Busch, A. Kocheturov, V. Tresp, and T. Seidl, "NF-GNN: network flow graph neural networks for malware detection and classification," *Proceedings of the 33rd International Conference on Scientific and Statistical Database Management*, pp. 121-132, 2021.
[17] Müller, and Meinard, "Dynamic time warping," *Information Retrieval for Music and Motion*, pp.69-84, 2007.
[18] K Xu, W Hu, J Leskovec, and S Jegelka, "How powerful are graph neural networks?" *arXiv preprint arXiv:1810.00826*, 2018.
[19] F. Scarselli, M. Gori, AC. Tsoi, M. Hagenbuchner, and G.Monfardini, "The Graph Neural Network Model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp.61-80, 2009.
[20] M. H. Guo, T. X. Xu, J. J. Liu, Z. N. Liu, P. T. Jiang, T. J. Mu, et al., "Attention mechanisms in computer vision: A survey," *Computational Visual Media*, vol. 8, no. 3, pp.331-368, 2022.
[21] H. Sen, "Time Series Prediction based on Improved Deep Learning," *IAENG International Journal of Computer Science*, vol.49, no.4, pp. 1133-1138, 2022.
[22] TN. Kipf, and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," *arXiv preprint arXiv:1609.02907*, 2016.
[23] G. D. Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and VPN traffic using time-related features," *Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP 2016)*, pp.407-414, 2016.
[24] F Zaki, F Afifi, S Abd Razak, A Gani, and NB Anuar, "GRAIN: Granular multi-label encrypted traffic classification using classifier chain," *Computer Networks*, vol.213, pp.109084, 2022.
[25] X. Liu, S. Zhang, H. Li, and W. Wang, "Fast application activity recognition with encrypted traffic," *Wireless Algorithms, Systems, and Applications: 16th International Conference*, pp.314-325, 2021.
[26] S. J. Xu, G. G. Geng, X. B. Jin, D. J. Liu, and J. Weng, "Seeing traffic paths: Encrypted traffic classification with path signature features," *IEEE Transactions on Information Forensics and Security*, vol.14, no. 8, pp.2166-2181, 2022.
[27] W. Li, and G. Quenard, "Towards a multi-label dataset of internet traffic for digital behavior classification," *2021 3rd International Conference on Computer Communication and the Internet (ICCCI)*. IEEE, pp.38-46, 2021.