Electronic Research Archive

https://www.aimspress.com/journal/era

ERA, 34(1): 173–195.
DOI: 10.3934/era.2026009
Received: 18 August 2025
Revised: 26 November 2025
Accepted: 18 December 2025
Published: 07 January 2026

Research article

TGM: A fine-grained classification method for DoH tunneling tools based on Transformer-GRU-MLP

Youwen Li[1,2,*], Qin Liu[1], Lejun Shen[2], Tao Wang[2], Jiangtao Zhai[2] and Guangjie Liu[2]

[1] Nanjing SAC Rail Traffic Engineering CO., LTD., Nanjing 210032, China
[2] School of Electronics and Information Engineering, Nanjing University of Information Science and Technology, Nanjing 210044, China

* **Correspondence:** Email: youwen-li@sac-china.com.

Abstract: DNS-over-HTTPS (DoH) effectively improves domain name system (DNS) security by encapsulating DNS query and response content in HTTPS packets, and also provides an opportunity for DoH-based covert tunneling attacks. Aiming at the problems of poor feature representation ability and low classification accuracy in the classification research of existing DoH tunneling tools, a multi-modal fusion deep learning classification model of DoH tunneling tools based on transformer-gated unit recurrent (GRU)-multilayer perceptron (MLP) (TGM) is proposed to help security personnel accurately locate specific threat types and take corresponding defensive measures. The model combines the spatial sequence feature extraction ability of the transformer encoder and the time sequence feature learning advantage of GRU to capture deep-level features between traffic generated by different DoH tunneling tools, and uses MLP to learn statistical features. Finally, we fuse the output embeddings of the three branches and learn the weights of different branch features through an attention mechanism. The experimental results show that our method achieves classification accuracy of 98.87% and F1-score of 98.02%, which are all better than the existing state-of-the-art methods.

Keywords: DNS-over-HTTPS; tunnel detection; deep learning; cyber security

1. Introduction

As the core component of the Internet infrastructure, the security of the domain name system (DNS) directly affects the stable operation of the Internet and the security of user data and privacy, and it has now become the basic support for most applications on the Internet [1]. The DNS protocol has long used the unencrypted user datagram protocol for plaintext transmission. This design was feasible when the scale of the early Internet was limited, but it has been exposed to serious flaws in the modern network environment [2]. First, plaintext transmission allows the query content to be bypassed and eavesdropped, resulting in the leakage of user access behavior. Second, the protocol lacks an integrity verification mechanism, allowing attackers to implement DNS hijacking or response tampering through man-in-the-middle attacks. More seriously, distributed attacks based on cache poisoning can cause regional service paralysis. In this context, the Internet Engineering Task Force officially released the DNS-over-HTTPS (DoH) protocol standard in 2018. This technology implements end-to-end encryption transmission and strong certificate verification mechanism by encapsulating DNS packets in the HTTP protocol layer and using transport layer security protocols. Compared with the 53-port plaintext communication of traditional DNS, DoH uses the standard 443 port, which makes DNS traffic mixed with conventional Web traffic, significantly improving the ability to resist traffic feature recognition [3].

Although DoH enhances DNS security and protects user privacy, it also provides a new covert channel for criminals. Attackers can use trusted DoH resolvers to build covert tunnels to evade security regulations, and thus carry out malicious activities such as command control, data theft, and ransomware communication [4], as shown in Figure 1. In this process, the client side of the attacker will encode the data to be transmitted into the DNS query field and initiate a domain name query request to the DoH resolver. After a complete domain name resolution process, the request finally reaches the authoritative DNS server controlled by the attacker, thus completing a covert data transfer [5]. Because DoH traffic is almost the same as ordinary HTTPS traffic in appearance, it has strong concealment, which brings severe challenges to cyber security supervision [6, 7].

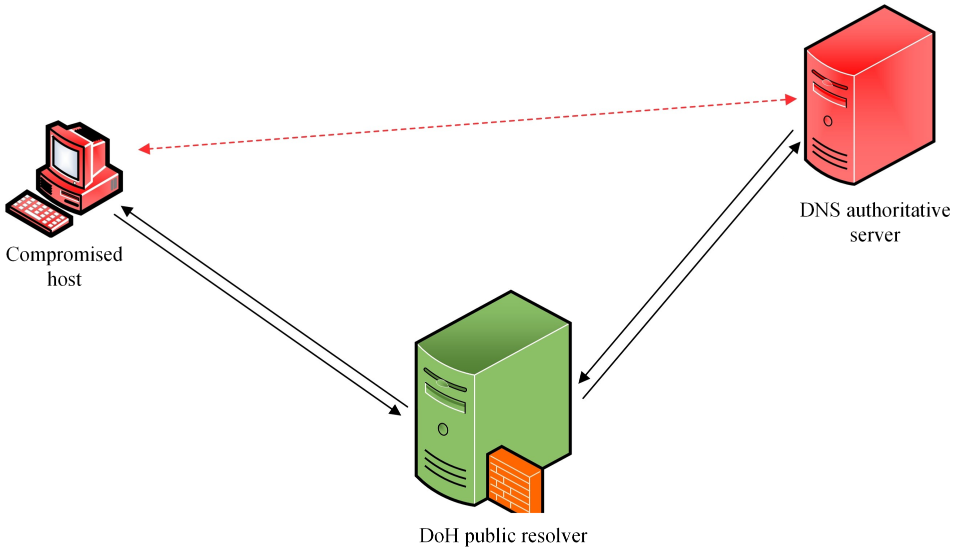


Figure 1. DoH tunnel communication threat model.

Aiming at the above problems, this paper proposes a TGM-based tunneling tools classification model. The model combines the spatial sequence feature extraction ability of transformer encoder with the time sequence feature learning advantage of gated recurrent unit (GRU) to capture deep-level features between flows generated by different DoH tunneling tools, and uses multilayer perceptron (MLP) to learn statistical features. Specifically, the method first extracts feature sequences containing data packet length, time, and direction information from DoH tunnel traffic, as well as corresponding statistical features, and designs an efficient feature fusion mechanism. Experiments show that, compared with traditional deep learning models and machine learning methods, TGM has obvious advantages in handling DoH tunneling tools classification tasks, especially in dealing with sample imbalance problems and capturing complex time sequence features.

The main contributions of this paper are as follows:

(1) A novel TGM model is proposed to effectively improve the classification accuracy of DoH tunneling tools. The model innovatively fuses the advantages of transformer encoder in extracting spatial sequence features, GRU in learning time sequence features, and MLP in learning statistical features. TGM can capture the deep-level features of the traffic generated by different DoH tunneling tools, and significantly improve the classification accuracy through an efficient feature fusion mechanism, effectively solving the problems of insufficient feature representation ability and low classification accuracy in existing studies.

(2) Construct a multi-scale feature system and design an adaptive feature fusion mechanism. Extract the original feature sequence containing data packet length, timestamp, and direction information from the DoH tunnel traffic, and combine eight key statistical features to form a multi-scale feature system. Meanwhile, an attention mechanism is designed to adaptively learn the weights of different branch features. This scheme performs well in optimizing the length of the input sequence and dealing with sample imbalance, providing a robust and reliable feature support for classification tasks.

(3) Based on CIRA-CIC-DoHBrw-2020 [11] and DoH-Tunnel-Traffic-HKD [22] datasets, the complementary advantages of transformer and GRU were confirmed through ablation experiments. Comparative experiments showed that the accuracy of the TGM model reached 98.87% and the F1-score reached 98.02%, which was significantly better than traditional deep learning models such as one-dimensional convolutional neural network (1D-CNN) and bidirectional long short-term memory (Bi-LSTM) and machine learning methods such as random forest (RF) and LightGBM. It provides an effective technical path for the accurate identification of DoH tunneling tools in cyber security defense.

The structure of the rest of this paper is as follows: Section 2 is related to the work, summarizing the progress of the existing DoH tunnel detection and classification research, as well as summarizing the methods based on machine learning and deep learning, respectively; Section 3 elaborates the overall architecture and core modules of the DoH tunneling tools classification model based on TGM; Sections 4 and 5 are experiments and conclusions.

2. Related works

In recent years, with the wide application of DoH technology, its covert features provide convenience for malicious tunnel attacks, and the detection and classification of DoH tunnels have become a hot topic in the field of cyber security [8, 9]. The existing research mainly focuses on machine learning and deep learning models, aiming to improve the recognition accuracy of DoH tunneling tools, but there are still problems such as insufficient feature representation ability and insufficient capture of complex timing patterns.

2.1. Methods based on machine learning

In machine learning methods, researchers rely on artificially designed statistical features to achieve classification [10]. For example, some studies extract statistical features such as data packet length distribution, time interval statistics, and the ratio of upstream and downstream traffic, and they combine integrated learning models such as RF, LightGBM, and XGBoost to perform DoH tunneling detection. Montazeri et al. [11] proposed a two-layer method to distinguish non-DoH and DoH traffic through time sequence analysis of data packet flows, and further divide benign and malicious DoH traffic. Wu et al. [12] developed the DOHUNTER system to detect DoH connections from HTTPS traffic and further identify its DoH parser. Jafar et al. [13] used many machine learning algorithms, among which support vector machine (SVM) and k-nearest neighbors (KNN) are relatively slow in the training stage, while gaussian naive bayes (GNB) is faster, but the detection effect is slightly inferior to other algorithms. Zebin et al. [14] implemented an explainable artificial intelligence solution using a novel machine learning framework, proposing a balanced and stacked RF that achieves very high accuracy in the DoH attack classification task. Mitsuhashi et al. [22] used continuous DoH traffic analysis and identified DoH tunneling tools. The system is able to continuously update knowledge about emerging malicious DoH tunneling tools on machine learning models, thereby reducing the threat that emerging DoH tunneling tools pose to users. Mahdavifar et al. [15] counted and extracted 30 stateful and stateless features, and applied them to the machine learning model of DNS tunnel detection, comparing the detection effects of MLP, SVM, and RF. This kind of method can achieve certain results when the feature engineering design is reasonable, but its performance is highly dependent on the manual screening quality of the features. It is difficult to capture the deep-level patterns implied in the traffic, and it is sensitive to the problem of sample imbalance. The classification effect on small sample categories is often poor.

2.2. Methods based on deep learning

Deep learning methods are widely used in traffic classification tasks due to their ability to learn automatic features. Liu et al. [16] proposed an attention-based Bi-GRU method for classifying encrypted HTTPS traffic. The bidirectional gated recurrent unit (Bi-GRU) model effectively captures data packet features through forward and backward GRU operations and attention mechanisms. Liang et al. [28] used CNN-based modules to extract features, and then evaluated the homogeneity and exclusivity of features based on clustering methods, which greatly improved the detection rate of the model for classification tasks and unknown types of DNS tunnels, and further reduced the false positive rate. Wang et al. [17] proposed a combination of multi-head attention and recurrent neural networks (RNNs) techniques, combining statistical features with byte sequence features extracted by RNNs, to identify important features through multi-head attention mechanisms. Casanova and Lin [18] proposed using LSTM and Bi-LSTM models and developed a comprehensive data processing flow, including feature selection and solutions to data imbalance problems. However, the global dependence of spatial sequences and fine-grained time sequence features has not been fully integrated, and the accuracy in distinguishing subtle behavioral differences of DoH tunneling tools is limited. In the study of fine grain classification for DoH tunneling tools, there are two limitations in the existing work: first, the feature dimension is single. Most methods only rely on statistical features or single sequence features, and fail to fully utilize the multi-modal information of traffic; second, the feature fusion mechanism is simple, and there is a lack of adaptive learning for the importance of different types of features, resulting in the lack of the model's ability to represent complex tunneling behaviors. In addition, some studies do not fully consider the similarity between DoH traffic and conventional HTTPS traffic, as well as the unique differences in database sharding, encapsulation strategy, and timing mode of different tunneling tools, resulting in the need to improve the classification accuracy and robustness.

In summary, the existing methods still have room for optimization in multi-modal feature fusion, complex timing, and spatial feature collaborative learning. The proposed TGM model integrates the advantages of transformer, GRU, and MLP to build a multi-branch feature learning and attention fusion mechanism, aiming to make up for the shortcomings of existing research and realize high-precision classification of DoH tunneling tools.

3. The framework of the proposed method

Aiming at the issues of insufficient feature representation and low classification accuracy in the classification of DoH tunneling tools in existing studies, this chapter proposes a TGM-based DoH tunneling tools classification method. This method combines the spatial sequence feature extraction ability of transformer encoder and the time sequence feature learning ability of GRU for dual feature learning and uses MLP to learn the statistical features of traffic. This enables the model to understand the feature patterns of different DoH tunneling tools more comprehensively, so as to accurately complete the classification of DoH tunneling tools. The classification process of the model as a whole is shown in Figure 2. It is mainly divided into three modules, namely the data preprocessing module, feature learning module, and feature fusion module.

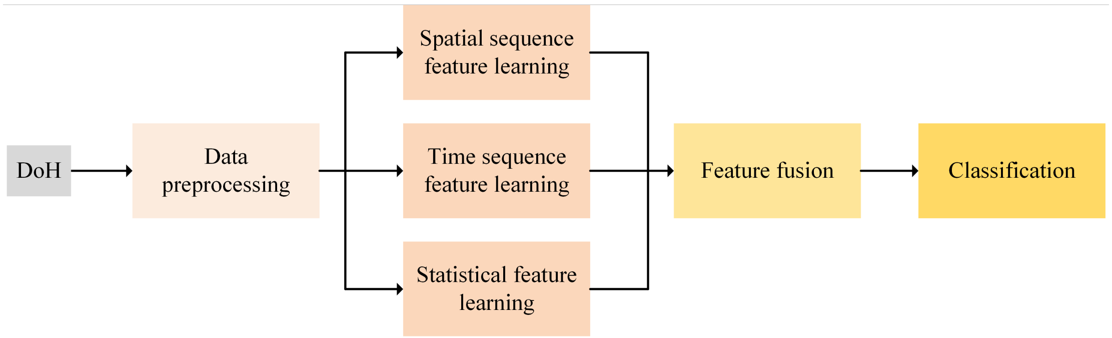

DoH
Data preprocessing
Spatial sequence feature learning
Time sequence feature learning
Statistical feature learning
Feature fusion
Classification

Figure 2. Overall classification process.

3.1. Data preprocessing

In the data preprocessing module, we first use the standard five-tuple (source IP, destination IP, source port, destination port, protocol) to extract all relevant session flows from the original packet capture (PCAP) file. Then, data cleaning and filtering are carried out. At this stage, all incomplete or empty sessions are removed. Feature sequence extraction is responsible for extracting the metadata of sessions from network traffic and constructing feature sequences as the input for subsequent modules. This model extracts three types of packet metadata from network traffic, namely relative time, packet length, and transmission direction, as the time sequence and spatial sequence features for the model input. We define the relative time sequence as a time sequence because it explicitly captures the timing and rhythm of packet transmissions. Correspondingly, we define the sequence composed of packet length and transmission direction together as a spatial sequence, as it describes the patterns of the data flow in terms of its transmission form and structure, rather than its temporal dynamics. Specifically, we extract the metadata of the first N packets of each session, that is, a fixed sequence length. For sessions with a length less than N, we use zero-padding; for sessions that are too long, we only retain the first N packets. The selection of these three types of information is based on an in-depth understanding of the DoH tunneling communication mechanism, considering both the differences in time patterns among different tunneling tools and the characteristics of data transmission behavior. In terms of the selection of statistical features, we have selected eight statistical features, as shown in Table 1. Before being input into the MLP module, all statistical features are standardized by z-score to eliminate the dimensional differences among different features.

Table 1. Selection of statistical features.

Statistical features	**Feature description**
Downlink Time Mean	The average time taken for downlink data transmission over a specific period
Uplink Time STD	The standard deviation of the time taken for uplink data transmission
Downlink Length Mean	The average size of data transmitted in the downlink direction over a certain period
Downlink Length STD	The standard deviation of the size of downlink transmitted data
Uplink Length Mean	The average size of data transmitted in the uplink direction over a specific time frame
Uplink Length STD	The standard deviation of the size of uplink transmitted data
Downlink Block Size STD	The standard deviation of the size of data blocks in downlink transmission
Uplink Block Size STD	The standard deviation of the size of data blocks in uplink transmission

The relative time features reflect the distribution of time intervals between data packets. Different tunneling tools adopt different time strategies to balance transmission efficiency and concealment [19]. For example, Dns2tcp uses a nearly uniform time interval in a stable network environment but dynamically adjusts the interval to maintain the reliability of TCP connections during packet loss or congestion; Dnscat2, as a command-and-control tool, adopts an active randomization strategy through the base interval of ±30% jitter and dynamically adjusts according to the type of task, so its communication mode is more flexible and the time interval distribution is more irregular; Iodine relies on pre-configured fixed intervals, its transmission rhythm is affected by the server's response, and its autonomous adjustment ability is relatively low. Data packet length features describe the distribution of data packet size in the communication process. In DoH tunnel communication, different tools use different sharding and encapsulation strategies for data [20]. For example, Dns2tcp tends to use fixed-size data blocks to provide stable TCP tunneling but will reduce the sharding size according to the maximum transmission unit (MTU) limit of the server; Dnscat2 distinguishes between the size of the command packet and the data packet, usually compressing the command packet, expanding the data packet to improve throughput, and actively perturbing the packet length distribution through encryption padding and redundancy fields to enhance the ability to resist detection; while Iodine is an IP layer tunnel, its data packet length is affected by many factors like encoding method, DNS record type, and compression algorithm, making its packet length distribution more diverse. Overall, the differences in database sharding and encapsulation strategies between these three tools provide exploitable features for their classification tasks.

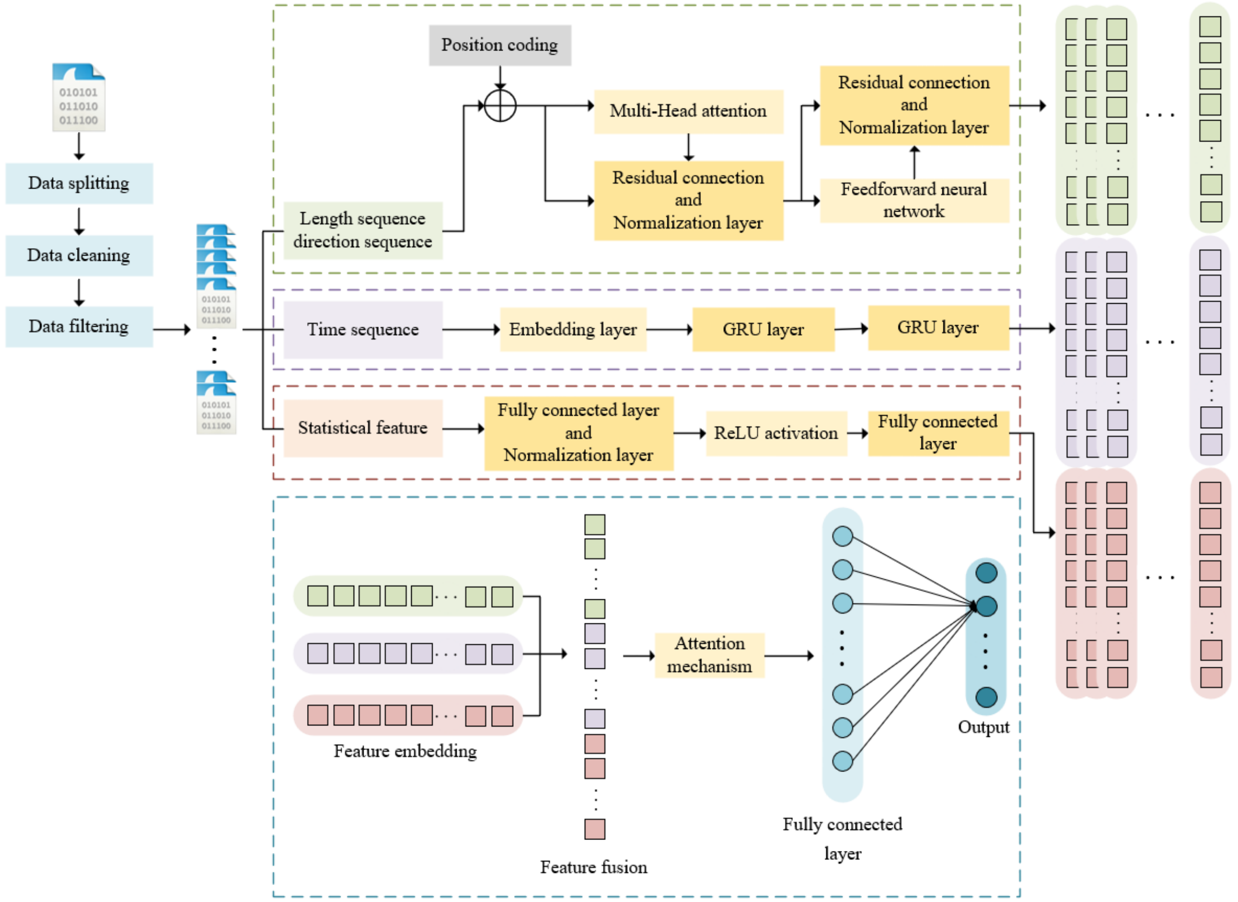


Figure 3. Model framework.

3.2. *Feature learning*

In the module of feature learning, this model combines transformer encoder, GRU, and MLP. Among them, the transformer branch processes the spatial sequences (length and direction sequences), the GRU branch processes the time sequence (relative time sequence), and the MLP branch processes the statistical features.

Then, the output embedding of the three is fused. The specific model framework is shown in Figure 3.

3.2.1. Spatial sequence feature learning

The spatial sequence feature learning module uses a transformer encoder based on the attention mechanism. In actual tunnel communication, data transmission often exhibits complex long-range dependence. This long-range dependence is difficult to capture with traditional sequence models, and transformer self-attention mechanism can just solve this problem. First, we map the input features to a high-dimensional space and add location information. For the input sequence X, the process can be expressed as

$$E = W_e X + b_e, X \in \mathbb{R}^{n \times d_{emb}}, \tag{3.1}$$

where n represents the sequence length. Then, position encoding PE is added to preserve position information in the sequence

$$PE_{(pos,\ 2i)} = sin\left(\frac{pos}{10000^{2i/d_{model}}}\right), PE_{(pos,\ 2i+1)} = cos\left(\frac{pos}{10000^{2i/d_{model}}}\right), \tag{3.2}$$

where pos is the location, i is the dimensional index of the location encoding, and d_{model} is the embedding dimension. In DoH tunnel traffic analysis, location encoding is crucial for understanding the transmission order of data packets. For example, Dns2tcp has significant differences in data packet patterns during the session establishment phase and the data transmission phase, and location information helps to distinguish the features of these different phases.

The multi-head self-attention mechanism is the core of the transformer, which allows the model to learn the dependencies in the sequence from multiple angles [21]. For DoH tunnel traffic, this mechanism allows the model to learn the differences exhibited by the DoH tunneling tools from different feature dimensions. First, the query (Q), key (K), and value (V) are generated through a linear transformation, and then attention weights are calculated:

$$Q = XW_Q, K = XW_K, V = XW_V, \tag{3.3}$$

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{3.4}$$

In Eq (3.4), $\sqrt{d_k}$ is the scaling factor, which is used to prevent excessive inner product growth from causing the softmax layer to disappear. Multi-head attention learns feature relationships from different representation subspaces by parallel computing multiple attention heads:

$$MultiHead(Q, K, V) = Concat(head_1, \ldots, head_h)W_o, \tag{3.5}$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V), \tag{3.6}$$

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2. \tag{3.7}$$

In Eqs (3.4)–(3.7), W_i^Q, W_i^K, W_i^V, and W_o are learnable parameter matrices. Each attention layer is followed by a feedforward neural network that introduces nonlinear transformations to enhance the expressiveness of the model, where W_1, W_2 and b_1, b_2 are the weights and biases of the network. Then, we have

$$Z = \mathrm{LayerNorm}(x + MultiHead(x)), \tag{3.8}$$

$$LayerNorm(x) = \gamma\frac{x - \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta, \tag{3.9}$$

$$h_{trans} = \frac{1}{n}\sum_{i=1}^{n} Z_i^{(N)}, \tag{3.10}$$

where μ and σ are the mean and standard deviation, γ and β are learnable scaling and bias parameters. Through the design of this structure, the model can effectively learn the spatial features of DoH tunnel traffic and capture the complex relationship between data packets at different time points. The output $Z \in \mathbb{R}^{n\times d_{emb}}$ is obtained through the above two layers of transformer encoder, and finally h_{trans} pooled.

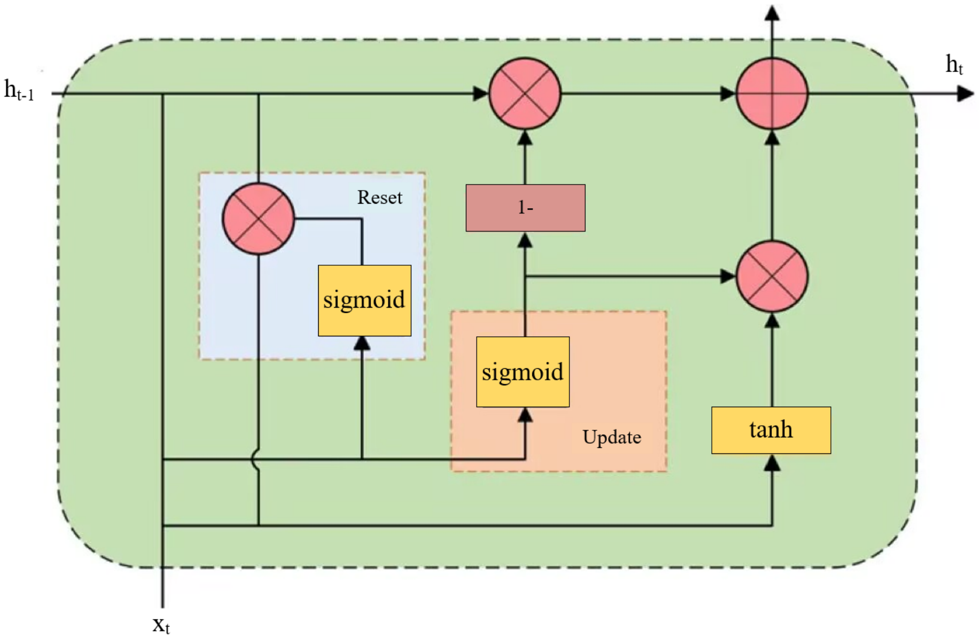


Figure 4. GRU structure.

3.2.2. Time sequence feature learning

The learning of time sequence feature is crucial for distinguishing among different tunneling tools, and the traffic generated by different DoH tunneling tools often exhibits unique time sequence feature. This model uses GRU networks as a time sequence feature learning module to capture these complex temporal patterns. GRU networks are an improved RNN architecture, which solves the vanishing gradient problem of traditional RNNs when processing long sequences through a well-designed gating mechanism. Compared to LSTM, GRU adopts a more concise structure but maintains similar performance. As shown in Figure 4, the core of GRU is its unique gating mechanism, which includes update gates and reset gates. The mechanism together controls the flow of information and the update of memory, which enables the model to dynamically adjust its internal state according to changes in traffic features.

The update gate plays a key role in the GRU structure, which determines how much information needs to be preserved in the hidden state of the previous moment. The calculation process can be expressed as

$$z_t = sigmoid(W_z[h_{t-1}, x_t] + b_z). \tag{3.11}$$

In Eq (3.11), x_t is the current input feature, h_{t-1} is the hidden state of the previous moment, W_z is the weight matrix, and b_z is the bias vector. When performing DoH tunneling tools classification, the update gate can dynamically adjust the memory policy according to the features of the data packet sequence. The reset gate is used in the GRU network to control the degree of influence of historical information on the current state, and its calculation formula is

$$r_t = sigmoid(W_r[h_{t-1}, x_t] + b_r). \tag{3.12}$$

In the DoH tunnel traffic analysis, the role of the reset gate is reflected in many aspects. When the session state transition is detected, such as the transition from the authentication stage to the data transmission stage, the reset gate can help the model switch the focus. When dealing with multiple rounds of interaction, it can help distinguish the features of different interaction cycles. In the face of anomalies or noise, the reset gate can reduce the influence of irrelevant historical information, so that the model remains sensitive to the current input. Based on the output of the update gate and the reset gate, the GRU model generates a new hidden state in the following steps. First, the model calculates the candidate hidden state, and then, through the control of the update gate, the historical information and the new information are fused to obtain the final hidden state. The specific formula is as follows:

$$\tilde{h}_t = tanh(W[r_t \odot h_{t-1}, x_t] + b), \tag{3.13}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t, \tag{3.14}$$

$$h_{gru} = h_n. \tag{3.15}$$

This update mechanism has unique advantages when dealing with DoH tunnel traffic. It can automatically adjust the retention ratio of historical information according to traffic features, alleviate the problem of layer disappearance through linear paths, and maintain selective memory for different types of tunnel behavior.

Based on the time sequence feature learning module, the model can fully understand the time sequence features of DoH tunnel traffic, providing a reliable feature representation for subsequent tunneling tools classification. Experimental results show that GRU-based time sequence feature learning can effectively distinguish the behavioral features of different DoH tunneling tools, significantly improving the accuracy and reliability of classification.

3.2.3. Statistical feature learning

In the classification task of DoH tunneling tools, the introduction of statistical features is a key necessity. The length and direction sequences only reflect the local features at the data packet level, while the statistical features provide a conversation-level global view and are insensitive to the order of data packets. Statistical features are used as meso-level representations to establish an interpretable bridge between the micro-packet features and the macro-application semantics, which is difficult to achieve with pure sequence models. The introduction of statistical features builds a multi-scale feature system, which makes the model have both fine grain sequence analysis capabilities and conversation-level global cognition, which significantly improves the accuracy, robustness, and interpretability of DoH tunneling tools classification.

In the statistical feature learning module, we first transform the normalized feature vector $S \in \mathbb{R}^m$ linearly through the fully connected layer, then apply the rectified linear unit (ReLU) activation function to introduce nonlinearity, and finally generate the final representation through the second fully connected layer to get

$$h_{mlp} = ReLU[W_{m2} \cdot ReLU(W_{m1}S + b_{m1}) + b_{m2}]. \tag{3.16}$$

3.3. Features fusion and classification

In the feature fusion module, we first stitch the feature vectors extracted from the three branches to form a fused feature vector:

$$H_{fused} = [h_{trans}; h_{gru}; h_{mlp}]. \tag{3.17}$$

In order to learn the importance weights of each feature branch adaptively, we introduce an attention mechanism. First, the attention score e of each branch feature is calculated through a fully connected network, where $W_a \in \mathbb{R}^{d_a \times d_h}$ is the weight matrix and $b_a \in \mathbb{R}^{d_a}$ is the bias vector. Then, the normalized attention weight α is generated through linear transformation and softmax function, and $v_a \in \mathbb{R}^{d_a}$ is the attention weight vector. Finally, the attention weight is used to weigh the original fusion features to generate a context vector c. The specific formula is as follows:

$$e = tanh(W_a H_{fusion} + b_a), \tag{3.18}$$

$$\alpha = softmax(v_a^T e), \tag{3.19}$$

$$c = \sum_{i=1}^{J} \alpha_i H_{fusion}^{(i)}. \tag{3.20}$$

In the classification layer, the context vector c is input into a fully connected network for classification, and the loss function adopts the label-smoothed cross entropy loss. The specific formula is as follows:

$$\hat{y} = softmax\left(W_c \cdot ReLU(W_f c + b_f) + b_c\right), \tag{3.21}$$

$$Loss = -\sum_{k=1}^{J} \left(0.9 y_j + \frac{0.1}{J}\right) \log(\hat{y}_j). \tag{3.22}$$

Among them, W_f and b_f are the hidden layer parameter, W_c and b_c are the output layer parameter, J is the number of classes, and $\hat{y}$ is the probability distribution of prediction. The algorithm for feature learning and classification of the above whole is shown in Algorithm 1.

4. Experimental results and analysis

4.1. Experimental environment and datasets

The experiment of this study was completed under Ubuntu 20.04 LTS, 64-bit operating system. The hardware environment configuration is as follows: CPU was Intel (R) Xeon (R) Silver 4210R CPU @2.40 GHz, GPU was NVIDIA Tesla T4, and GPU memory was 16 GB. The programming language was Python 3.8, and the deep learning framework was PyTorch 1.9.1.

Table 2. Distribution of dataset samples.

DoH tunneling tools	Number of samples	Sample size
Dns2tcp	111271	2.1 G
Dnscat2	10290	5.3 G
Iodine	12295	9.8 G
Dnstt	12163	223 MB
Tcp-over-dns	7929	89 MB
Tuns	7665	47 MB

This experiment used CIRA-CIC-DoHBrw-2020 [11] and DoH-Tunnel-Traffic-HKD [22] datasets for experimental evaluation. CIRA-CIC-DoHBrw-2020 is a dataset created by the Canadian Institute

Algorithm 1 TGM feature learning and fusion

```
Require: $\mathbf{T}$=Time sequence (relative timestamps)
    $\mathbf{L}$=Packet length sequence
    $\mathbf{D}$ = Direction sequence ($d_i \in \{0, 1\}$)
    $\mathbf{S} \in \mathbb{R}^8$=Statistical feature vector
Ensure: Classification probabilities $P(y|\mathbf{T}, \mathbf{L}, \mathbf{D}, \mathbf{S})$
// 1. Spatial Sequence Feature Learning (Transformer)
$\mathbf{X} \leftarrow Concat(\mathbf{L}, \mathbf{D})$ {Combine length and direction}
$\mathbf{E} \leftarrow W_e\mathbf{X} + b_e$ {Embedding projection}
$\mathbf{E} \leftarrow \mathbf{E} + PE$ {Add positional info}
for $layer = 1$ to $2$ do
    $\mathbf{Q} \leftarrow \mathbf{E}W_q, \mathbf{K} \leftarrow \mathbf{E}W_k, \mathbf{V} \leftarrow \mathbf{E}W_v$
    $Attention \leftarrow softmax\left(\frac{\mathbf{QK}^T}{\sqrt{d_k}}\right)\mathbf{V}$
    $\mathbf{Z} \leftarrow LayerNorm(\mathbf{E} + MultiHead(Attention))$
    $\mathbf{Z} \leftarrow LayerNorm(\mathbf{Z} + FFN(\mathbf{Z}))$
end for
$\mathbf{h}_{trans} \leftarrow MeanPooling(\mathbf{Z})$
// 2. Time Sequence Feature Learning (GRU)
$\mathbf{h}_0 \leftarrow 0$ {Initialize hidden state}
for $i = 1$ to $n$ do
    $\mathbf{z}_i \leftarrow sigmoid(W_z[t_i; \mathbf{h}_{i-1}] + b_z)$ {Update gate}
    $\mathbf{r}_i \leftarrow sigmoid(W_r[t_i; \mathbf{h}_{i-1}] + b_r)$ {Reset gate}
    $\tilde{\mathbf{h}}_i \leftarrow tanh(W_h[t_i; \mathbf{r}_i \odot \mathbf{h}_{i-1}] + b_h)$
    $\mathbf{h}_i \leftarrow (1 - \mathbf{z}_i) \odot \mathbf{h}_{i-1} + \mathbf{z}_i \odot \tilde{\mathbf{h}}_i$
end for
$\mathbf{h}_{gru} \leftarrow \mathbf{h}_n$
// 3. Statistical Feature Learning (MLP)
$\mathbf{h}_{mlp} \leftarrow W_{m2}ReLU(W_{m1}\mathbf{S} + b_{m1}) + b_{m2}$
// 4. Multi-modal Fusion
$\mathbf{H}_{fusion} \leftarrow [\mathbf{h}_{trans}; \mathbf{h}_{gru}; \mathbf{h}_{mlp}]$
$\mathbf{e} \leftarrow tanh(W_aH_{fusion} + b_a)$
$\alpha \leftarrow softmax(v_a^Te)$ {Attention weights}
$\mathbf{c} \leftarrow \sum_i \alpha_i v_i$ {Context vector}
// 5. Classification
$\mathbf{u} \leftarrow W_c\mathbf{c} + b_c$
$\mathbf{p} \leftarrow softmax(\mathbf{u})$ {Predicted probabilities}
return p
```

for Cybersecurity (CIC) and the Canadian Internet Registration Authority (CIRA), mainly used to detect and distinguish between benign DoH traffic, malicious DoH traffic, and non-DoH traffic. The malicious DoH traffic category was mainly generated using Dns2tcp, Dnscat2, and Iodine. DoH-Tunnel-Traffic-HKD was usually used in conjunction with CIRA-CIC-DoHBrw-2020 as an important supplement to the latter. It focuses on other DoH tunneling tools not included in CIRA-CIC-DoHBrw-2020, mainly including malicious traffic generated using Dnstt, Tcp-over-dns, and Tuns. These two datasets are currently the most representative DoH traffic datasets, which contain traffic samples generated using different DoH tunneling tools (Dns2tcp, Dnscat2, Iodine, Dnstt, Tcp-over-dns, and Tuns). Using the above two standardized public datasets ensured the transparency and reproducibility of our experimental results. Most importantly, this enabled our TGM model to conduct fair and direct performance comparisons with existing advanced methods that also use these datasets. The above datasets are all provided in PCAP file format. We consider each DoH tunnel traffic session as an independent sample. In the processed dataset, the distribution of the number of samples (the number of sessions) for various tools is shown in Table 2. We can observe that there is a significant class imbalance in the dataset, and the number of samples in Dns2tcp is significantly higher than that of the other five tools. To solve this problem, we not only adopted the method of stratified sampling but also used the label-smoothed cross entropy loss function (Eq (3.22)) during training. Different from the standard cross entropy, which may lead to over fitting to the majority classes, label smoothing effectively regularizes the model by penalizing low-entropy output distributions and improves the generalization ability of the minority classes. The stratified sampling strategy ensured that the training set and the test set had the same class distribution, guaranteeing the reliability of the evaluation metrics. We divided the data according to the ratio of 80% for the training set and 20% for the test set.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}, \tag{4.1}$$

$$Precision = \frac{TP}{TP + FP}, \tag{4.2}$$

$$Recall = \frac{TP}{TP + FN}, \tag{4.3}$$

$$F1 - Score = \frac{2Precision \cdot Recall}{Precision + Recall}. \tag{4.4}$$

We used a range of evaluation metrics to evaluate the classification performance of our model, including: Accuracy, Precision, Recall, and F1-Score. These metrics facilitate subsequent evaluation of our model against other models. Where TP refers to the number of samples that the model correctly predicted as positive, TN refers to the number of samples that the model correctly predicted as negative, FP refers to the number of samples that the model incorrectly predicted as positive, and FN refers to the number of samples that the model incorrectly predicted as negative.

4.2. Model parameter training

In the input sequence length evaluation experiment, the optimal parameters of each model with different input sequence lengths were determined using a grid search strategy. Table 3 lists the optimal parameter settings of the TGM model when the input sequence length is 16. Among them, the transformer encoder part adopts a four-layer structure, and each layer contains four attention

Table 3. Model parameter configuration.

Parameter type	Parameter value
Transformer encoder layers	4
Transformer hidden layer dimension	64
GRU layer	2
GRU hidden layer dimension	128
MLP hidden layer dimension	64
Dropout	0.5
Batch size	256

heads. This configuration enables the model to focus on the dependence of DoH tunnel traffic from multiple different angles. The GRU network adopts a two-layer structure, and the hidden layer dimension is set to 128, which allows the model to fully learn the timing features. The MLP adopts a single hidden layer structure, and the hidden layer dimension is set to 64. The design focuses on learning the global statistical features of DoH tunnel traffic. Through the nonlinear mapping ability of the fully connected layer, the session-level statistical laws are transformed into highly discriminative feature representations, which make up for the limitations of the sequence model in global pattern capture. During the training process, we adopted a small batch training method with a batch size of 256, used the Adam optimizer, and set the initial learning rate to 0.001. To prevent overfitting, a dropout layer was added between the key layers, and the ratio was set to 0.5. The entire training process was set to 100 rounds, and the early stop mechanism was adopted. When the performance on the validation set did not improve for 10 consecutive rounds, the training was terminated early.

4.3. Input sequence length evaluation and selection

In order to determine the optimal input sequence length, several experiments were carried out on the TGM model, and each experiment only changed its input sequence length to study the impact of different sequence lengths on the model's performance. As shown in Figure 5, this chapter uses different configurations with sequence lengths from 5 to 30 to observe its impact on the F1-score. When the sequence length is 5, it is difficult for the model to accurately capture the behavioral features of the DoH tunneling tools due to less timing information provided, and the F1-score is only 70.73%. As the sequence length increases, the model performance gradually improves, reaching 98.02% when the sequence length is 16. When the sequence length continues to increase, the model performance begins to stabilize, and even has a downward trend. This is mainly because the overly long sequence introduces redundant information, increases the complexity of the model, and instead affects the classification effect. This result shows that the sequence length is a key parameter affecting the performance of the model. Sequences that are too short will lead to insufficient feature information to fully describe the behavioral patterns of the DoH tunneling tools; sequences that are too long will introduce noise and redundant information, increase computational overhead, and may lead to overfitting. Among the six tunneling tools classification tasks, a sequence length of 16 can achieve the best balance between information volume and computational complexity.

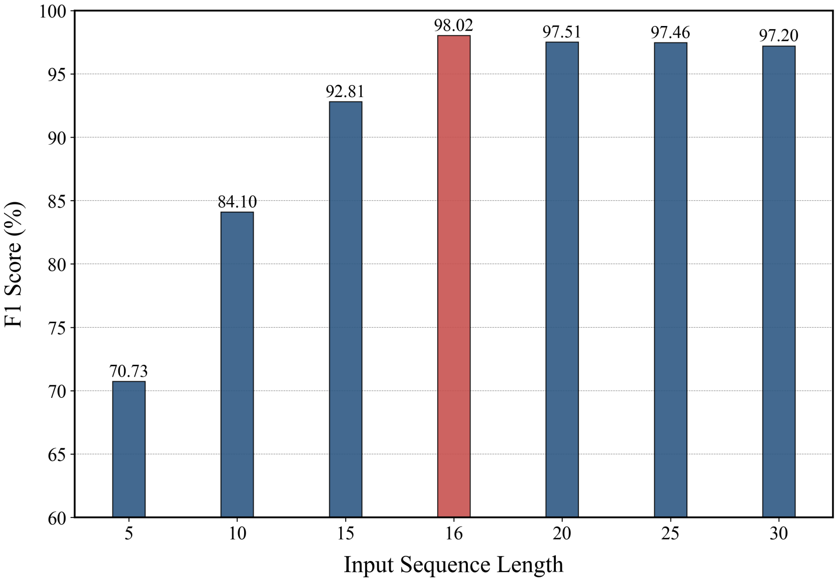


Figure 5. Effect of input sequence length on F1-score.

4.4. Ablation experiment

To systematically evaluate the role of individual key components in the TGM model, a series of detailed ablation experiments were designed to analyze the impact of each component on the classification performance of the DoH tunneling tools. Specifically, we designed the following groups of controlled experiments: model using transformer-MLP (TM), model using GRU-MLP (GM), model using transformer-GRU (TG) and variant model replacing GRU with LSTM (TLM). During the experiment, the same data preprocessing methods and datasets were used, and only the model structure was changed to ensure the comparability of experimental results. Table 4 shows the performance of different model variants on the test set.

Table 4. Comparison of model ablation test results.

Model	Accuracy (%)	Precision(%)	Recall(%)	F1-score(%)
TM	97.41	96.86	96.35	96.54
GM	96.46	95.59	95.84	95.72
TG	94.27	92.86	93.58	94.07
TLM	98.36	97.75	97.12	97.26
TGM	**98.87**	**98.71**	**98.32**	**98.02**

First, by comparing the complete TGM model (F1-score 98.02%) with the TG model (F1-score 94.07%), we observed the most significant performance difference. The TG model completely removed the MLP branch for processing statistical features, resulting in a serious decline in performance. This strongly demonstrates that the global statistical features learned by the MLP are a crucial and indispensable component for the model to achieve high performance, providing a global session-level perspective that cannot be captured by the sequence models (T and G) alone. Second, by comparing the TGM model (F1-score 98.02%) with the TM (F1-score 96.54%) and GM

(F1-score 95.72%) models, we found that the two sequence feature extraction methods have a significant complementary effect. TM is good at capturing long-distance dependencies in the packet sequence, while GRU is more adept at learning local temporal change patterns. Whether using TM or GM alone, their performance cannot reach the level of the complete TGM model. This confirms that both global dependencies and local temporal features are essential elements in DoH tunnel traffic analysis, jointly constituting a comprehensive understanding of tunnel behavior. Finally, by comparing TGM (Accuracy 98.87%) with TLM (Accuracy 98.36%), we found that the simplified gating mechanism of GRU is more efficient in handling DoH tunnel traffic. Although LSTM has a more complex gating structure, in the DoH tunnel tool classification task, the combination of the update gate and reset gate of GRU can fully capture temporal features while reducing computational complexity and the number of parameters. The accuracy of the TLM model at 98.36% is slightly lower than that of TGM at 98.87%. Considering the computational efficiency advantage of GRU, this performance difference further validates that GRU is a more suitable temporal feature extractor in the DoH tunnel traffic classification scenario.

4.5. Comparative experiment

To validate the effectiveness and superiority of the proposed TGM model in the DoH tunneling tools classification task, we designed two sets of comparative experiments. The first set of experiments compared TGM with deep learning models commonly used in the field of network traffic classification; the second set of experiments compared the performance of DoH tunneling tools classification methods in the existing literature.

First, we compared three representative deep learning models in the field of encrypted traffic classification: 1D-CNN [23], Bi-LSTM [24], and CNN-LSTM [25] hybrid models. 1D-CNN uses local perception to capture short-range dependencies between data packets and extracts hierarchical feature representations through multi layer convolutional stacking, which can efficiently learn recognizable features from traffic data. The Bi-LSTM model adopts a bidirectional LSTM structure, which processes sequence data in both forward and backward directions, and can comprehensively capture context information. This bidirectional modeling method is particularly suitable for tasks that need to consider the complete sequence context. It can effectively handle long-range dependencies and ensure that the model takes into account all the information before and after the sequence when understanding the sequence. CNN-LSTM is a classic combined architecture for statistical feature extraction that combines CNNs and LSTMs. The CNN layer first extracts local spatial features to capture spatial patterns in the sequence; then these features are fed into the LSTM layer for temporal modeling to capture long-distance dependencies. To ensure fairness in comparison, all models used the same extracted feature sequences as inputs, and only the input format was adjusted to suit the structural features of each model. Table 5 shows the performance comparison results of commonly used deep learning methods on the classification task of DoH tunneling tools.

As can be seen from the table, the accuracy rate of our model reaches 98.87%, which is 1.29 percentage points higher than that of CNN-LSTM's 97.58%, and 2.75 and 2.36 percentage points higher than that of 1D-CNN and Bi-LSTM, respectively. This shows that this model can not only accurately identify various types of DoH tunneling tools, but also maintain a low misclassification rate. As a harmonized average of accuracy rate and recall rate, the F1-score can more comprehensively evaluate the performance of classification methods. The F1-score of our model

Table 5. Comparison of classification metrics with commonly used encrypted traffic classification models.

Model	Accuracy (%)	Precision(%)	Recall(%)	F1-score(%)
1D-CNN [23]	96.12	95.48	94.21	95.10
Bi-LSTM [24]	96.51	94.79	95.21	95.23
CNN-LSTM [25]	97.58	96.97	96.52	96.61
TGM (our)	**98.87**	**98.71**	**98.32**	**98.02**

reaches 98.02%, which is significantly higher than that of other comparison models. This result fully demonstrates the excellent performance of TGM on the classification task of DoH tunneling tools. In order to show how TGM classifies each category on the DoH tunneling tools classification task, we also did corresponding comparison experiments. As shown in Figure 6, TGM significantly outperforms the comparison model in both accuracy and F1-score, especially in the DoH tunneling tools category with a small sample size. However, there is still room for optimization in the performance of IP layer tunneling tools such as Iodine. In addition, TGM also outperforms the comparison model on mainstream tools such as Dns2tcp, with an accuracy rate of 99.31% and an F1-score of 98.42%.

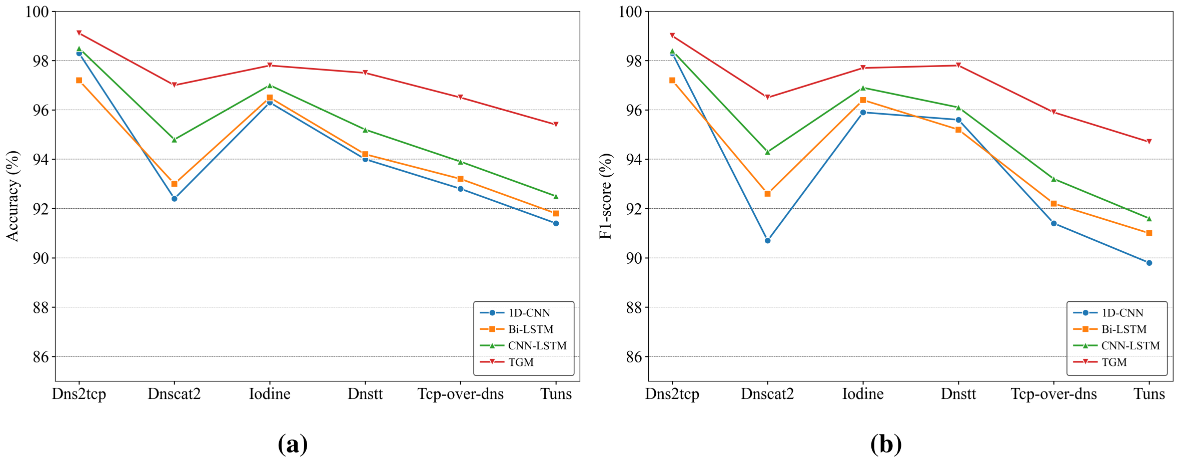


Figure 6. Accuracy and F1-scores of different models for classifying different DoH tunneling tools. (a) Accuracy of different models for classifying different DoH tunneling tools. (b) F1-scores of different models for classifying different DoH tunneling tools.

Table 6. Comparison of classification indicators with existing models.

Model	Accuracy (%)	Precision(%)	Recall(%)	F1-score(%)
RF [26]	96.82	94.51	94.79	94.65
LightGBM [22]	97.22	94.97	95.43	95.19
XGBoost [26]	97.06	94.73	95.18	94.95
CatBoost [27]	96.91	94.46	94.94	94.69
TGM (our)	**98.87**	**98.71**	**98.32**	**98.02**

Table 7. Comparison of computational costs of different models.

Model	Parameters(M)	Total Training Time(Min)	Inference Latency(Ms/Batch)
1D-CNN [23]	0.8	27	1.8
Bi-LSTM [24]	1.4	46	4.2
CNN-LSTM [25]	1.7	51	4.5
TGM (our)	2.2	59	4.7

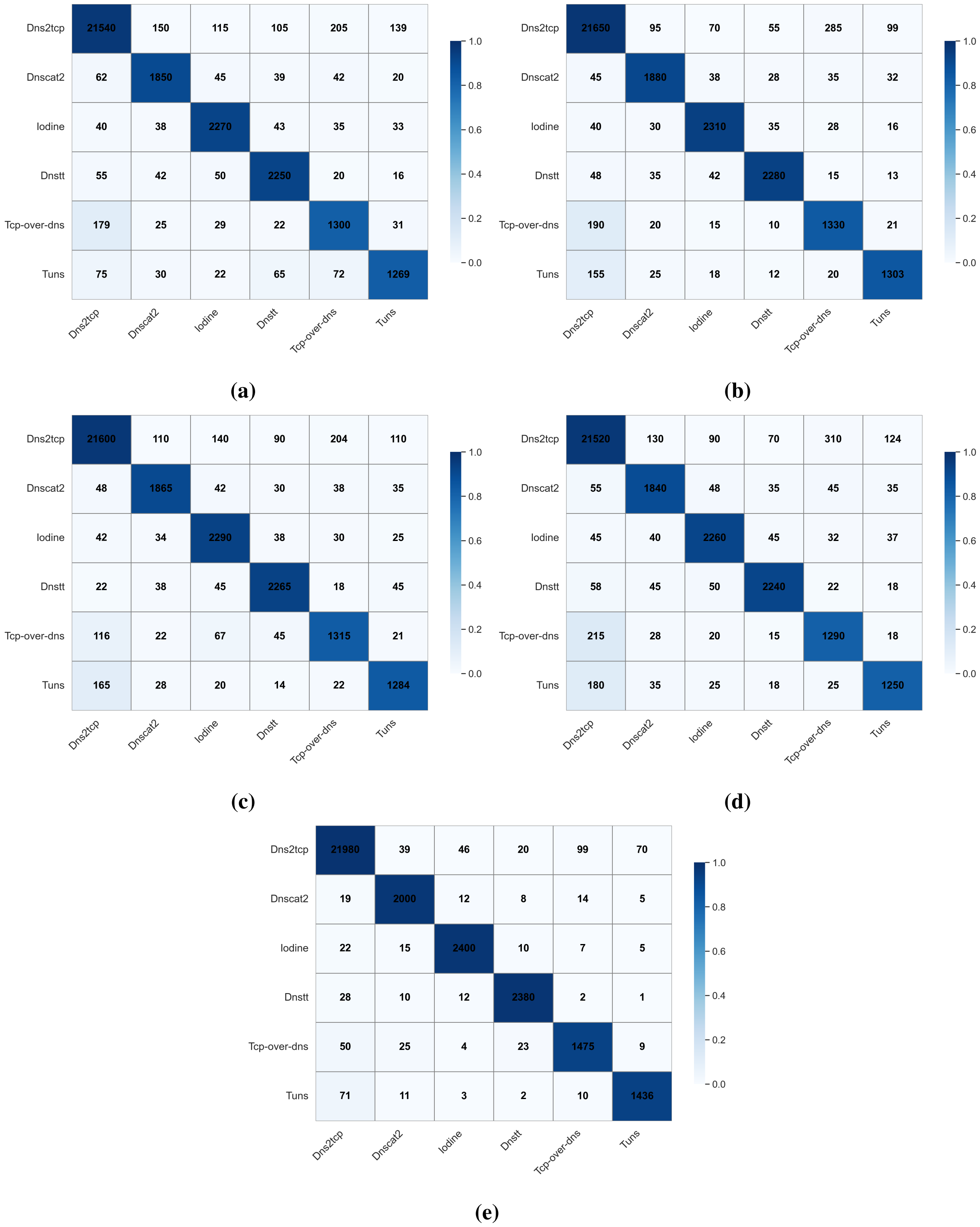
To further demonstrate the effectiveness of TGM, we compared the performance of DoH tunneling tools classification methods in the existing literature, which include RF [26], LightGBM [22], XGBoost [26], and CatBoost [27]. To ensure a fair and direct comparison, we re-implemented the proposed baseline methods. The datasets used are all the six-class mixed datasets constructed in Section 4.1. Unlike that model method, these model methods only used statistical features as input. Table 6 shows the performance comparison of each model on the same dataset. As can be seen from the table, the accuracy rate of TGM reaches 98.87%, which is 1.65 percentage points higher than that of LightGBM. This significant gap proves that the deep learning method that fuses sequence features and statistical features has obvious advantages over traditional machine learning methods. This performance difference is mainly due to two factors: The first is that although the statistical features can reflect the overall features of the traffic, they inevitably lose the timing information and patterns of the fine grain; the second is that the predefined statistical features may not fully capture the unique behavioral features of DoH tunneling tools. In contrast, the TGM model can automatically learn richer and more discriminating feature representations by fusing the sequence of data packets with the statistical features. Figure 7 shows the confusion matrix for each model classification. It can be seen from the figure that the classification performance of TGM for each individual is superior to other models, and the classification of a small number of sample categories also has a higher classification effect.

By synthesizing the results of the two sets of comparative experiments, it can be concluded that the TGM model proposed in this paper has significant advantages in the DoH tunneling tools classification task. It is not only better than the commonly used deep learning models in the field of network traffic classification, but it also greatly surpasses the classification methods in the existing literature. These results fully demonstrate the effectiveness of the deep learning method that fuses sequence features and statistical features in capturing DoH tunnel traffic features.

4.6. Computational cost analysis

In order to evaluate the feasibility of TGM in practical deployment, this section conducts a quantitative analysis of the model's computational cost. We compared the computational overhead of TGM with that of 1D-CNN, Bi-LSTM, and CNN-LSTM. The comparison metrics include: the number of model parameters (Parameters), total training time (Total Training Time), and average inference latency (Inference Latency). The batch size was set to 256, and the tests were carried out on a GPU (NVIDIA Tesla T4). The specific results are shown in Table 7.

As can be seen from Table 7, 1D-CNN has the lowest values in all cost indicators. Due to the integration of three feature learning branches, TGM has the most complex structure. Therefore, its number of parameters (2.2 M) and training time (59 minutes) are the highest among all models. In



	Dns2tcp	Dnscat2	Iodine	Dnstt	Tcp-over-dns	Tuns
Dns2tcp	21540	150	115	105	205	139
Dnscat2	62	1850	45	39	42	20
Iodine	40	38	2270	43	35	33
Dnstt	55	42	50	2250	20	16
Tcp-over-dns	179	25	29	22	1300	31
Tuns	75	30	22	65	72	1269

(a)

	Dns2tcp	Dnscat2	Iodine	Dnstt	Tcp-over-dns	Tuns
Dns2tcp	21650	95	70	55	285	99
Dnscat2	45	1880	38	28	35	32
Iodine	40	30	2310	35	28	16
Dnstt	48	35	42	2280	15	13
Tcp-over-dns	190	20	15	10	1330	21
Tuns	155	25	18	12	20	1303

(b)

	Dns2tcp	Dnscat2	Iodine	Dnstt	Tcp-over-dns	Tuns
Dns2tcp	21600	110	140	90	204	110
Dnscat2	48	1865	42	30	38	35
Iodine	42	34	2290	38	30	25
Dnstt	22	38	45	2265	18	45
Tcp-over-dns	116	22	67	45	1315	21
Tuns	165	28	20	14	22	1284

(c)

	Dns2tcp	Dnscat2	Iodine	Dnstt	Tcp-over-dns	Tuns
Dns2tcp	21520	130	90	70	310	124
Dnscat2	55	1840	48	35	45	35
Iodine	45	40	2260	45	32	37
Dnstt	58	45	50	2240	22	18
Tcp-over-dns	215	28	20	15	1290	18
Tuns	180	35	25	18	25	1250

(d)

	Dns2tcp	Dnscat2	Iodine	Dnstt	Tcp-over-dns	Tuns
Dns2tcp	21980	39	46	20	99	70
Dnscat2	19	2000	12	8	14	5
Iodine	22	15	2400	10	7	5
Dnstt	28	10	12	2380	2	1
Tcp-over-dns	50	25	4	23	1475	9
Tuns	71	11	3	2	10	1436

(e)

Figure 7. Confusion matrix for different model classifications. (a) Confusion matrix for RF classifications. (b) Confusion matrix for LightGBM classifications. (c) Confusion matrix for XGBoost classifications. (d) Confusion matrix for CatBoost classifications. (e) Confusion matrix for TGM classifications.

terms of the inference latency, a key indicator for evaluating the actual deployment ability, the latency of TGM is 4.7 ms, which is also the highest. However, the magnitude of this increase is completely controllable. In the GPU deployment environment commonly adopted by modern network security devices, the inference latency of TGM (4.7 ms) is of the same order of magnitude as that of Bi-LSTM (4.2 ms) and CNN-LSTM (4.5 ms), only 0.2 ms (about 4.4%) slower than that of the CNN-LSTM model. A comprehensive analysis shows that the design of the TGM architecture achieves excellent cost-effectiveness. As shown in Table 5, the F1-score of TGM (98.02%) is significantly 1.41 percentage points higher than that of the second-best CNN-LSTM (96.61%). The TGM model has achieved a huge improvement in classification accuracy at the cost of a small and controllable increase in inference overhead, which fully demonstrates the superiority and practicality of the TGM architecture in the fine-grained classification task of DoH tunneling tools.

5. Conclusions and future works

This paper proposes a classification model for DoH tunneling tools based on TGM. The model innovatively combines the spatial sequence feature extraction ability of encoder and the time sequence feature learning advantages of GRU. Meanwhile, MLP is used to learn the statistical features, and an efficient classification framework is designed for the flow features of DoH tunnels. In the method design, the metadata is extracted from the original flow to construct the feature sequence and statistical features, and the accurate classification of DoH tunneling tools is realized through the TGM architecture. Using CIRA-CIC-DoHBrw-2020 and DoH-Tunnel-Traffic-HKD datasets for comprehensive evaluation, TGM achieved 98.87% accuracy and 98.02% F1-score, which is significantly better than the existing technologies.

Although the TGM model performs well on the CIRA-CIC-DoHBrw-2020 and DoH-Tunnel-Traffic-HKD datasets, both of these datasets were generated in a controlled laboratory environment. This may result in them lacking the noise, variability, and evasion strategies that are unique to real enterprise networks. In the future, we will enhance the generality of the model, evaluate its performance on live or enterprise-scale traffic, the generalization ability of the TGM model, and improve its performance in detecting unknown tunnel traffic.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grants No. U2436601)

Conflict of interest

The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. M. K. Bansal, M. Sethumadhavan, Survey on domain name system security problems-DNS and blockchain solutions, in *Futuristic Trends in Networks and Computing Technologies. FTNCT 2019. Communications in Computer and Information Science* (eds. P. Singh, S. Sood, Y. Kumar, M. Paprzycki, A. Pljonkin, W.C. Hong), Springer Singapore, **1206** (2020), 634–647. https://doi.org/10.1007/978-981-15-4451-4_50
2. L. Jiao, Y. Zhu, X. Fu, Y. Zhou, F. Qin, Q. Liu, CCSv6: A detection model for DNS-over-HTTPS tunnel using attention mechanism over IPv6, in *2023 IEEE Symposium on Computers and Communications (ISCC)*, (2023), 1327–1330, https://doi.org/10.1109/ISCC58397.2023.10218057
3. C. Dong, J. Yang, Y. Li, Y. Wu, Y. Chen, C. Li, et al., E-DoH: Elegantly detecting the depths of open DoH service on the internet, *Cybersecurity*, **8** (2025), 101. https://doi.org/10.1186/s42400-025-00390-5
4. N. Sharma, M. Swarnkar, DLAZE: Detecting DNS tunnels using lightweight and accurate method for zero-day exploits, *IEEE Trans. Network Ser. Manage.*, **22** (2025), 2343–2353. https://doi.org/10.1109/TNSM.2025.3541234
5. X. Liu, W. Mao, A. Wang, Z. Li, H. Xue, Y. Zhang, et al., DNS tunnel detection for low throughput data exfiltration via time-frequency domain analysis, in *ICC 2023-IEEE International Conference on Communications*, (2023), 2331–2337, https://doi.org/10.1109/ICC45041.2023.10279472
6. M. Dawood, S. Tu, C. Xiao, M. Haris, H. Alasmary, M. Waqas, et al., The impact of domain name server (DNS) over hypertext transfer protocol secure (HTTPS) on cyber security: Limitations, challenges, and detection techniques, *Comput. Mater. Continua*, **80** (2024), 4513–4542. https://doi.org/10.32604/cmc.2024.050049
7. M. Zuo, C. Guo, H. Xu, Z. Zhang, Y. Cheng, METC: A hybrid deep learning framework for cross-network encrypted DNS over HTTPS traffic detection and tunnel identification, *Inf. Fusion*, **121** (2025), 103125. https://doi.org/10.1016/j.inffus.2025.103125
8. J. Tong, Y. Zhao, C. Jin, W. Chen, Y. Zhang, L. Wu, An adaptive DoH encrypted tunnel detection method based on contrastive learning, *IEEE Internet Things J.*, **12** (2025), 25936–25950. https://doi.org/10.1109/JIOT.2025.3561015
9. M. Moure-Garrido, C. Campo, C. Garcia-Rubio, Real time detection of malicious DoH traffic using statistical analysis, *Comput. Networks*, **234** (2023), 109910. https://doi.org/10.1016/j.comnet.2023.109910
10. S. Yadav, R. K. Patel, V. P. Singh, Bearing fault classification using TKEO statistical features and artificial intelligence, *J. Intell. Fuzzy Syst.*, **45** (2023), 4147–4164. https://doi.org/10.3233/JIFS-224221
11. M. MontazeriShatoori, L. Davidson, G. Kaur, A. H. Lashkari, Detection of DoH tunnels using time-series classification of encrypted traffic, in *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, (2020), 63–70. https://doi.org/10.1109/DASC-PICom-CBDCom-CyberSciTech49142.2020.00026

12. J. Wu, Y. Zhu, B. Li, Q. Liu, B. Fang, Peek inside the encrypted world: Autoencoder-based detection of doh resolvers, in *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, (2021), 783–790. https://doi.org/10.1109/TrustCom53373.2021.00113

13. M. T. Jafar, Analysis and investigation of malicious DNS queries using CIRA-CIC-DoHBrw-2020 dataset, *Manchester J. Artif. Intell. Appl. Sci.* , **2** (2021), 1–10.

14. T. Zebin, S. Rezvy, Y. Luo, An explainable AI-based intrusion detection system for DNS over HTTPS (DoH) attacks, *IEEE Trans. Inf. Forensics Secur.*, **17** (2022), 2339–2349. https://doi.org/10.1109/TIFS.2022.3183390

15. S. Mahdavifar, A. H. Salem, P. Victor, A. H. Razavi, M. Garzon, N. Hellberg, et al., Lightweight hybrid detection of data exfiltration using DNS based on machine learning, in *Proceedings of the 2021 11th International Conference on Communication and Network Security (ICCNS '21)*, (2022), 80–86. https://doi.org/10.1145/3507509.3507520

16. X. Liu, J. You, Y. Wu, T. Li, L. Li, Z. Zhang, et al., Attention-based bidirectional GRU networks for efficient HTTPS traffic classification, *Inf. Sci.*, **541** (2020), 297–315. https://doi.org/10.1016/j.ins.2020.05.035

17. Y. Wang, C. Shen, D. Hou, X. Xiong, Y. Li, FF-MR: A DoH-encrypted DNS covert channel detection method based on feature fusion, *Appl. Sci.*, **12** (2022), 12644. https://doi.org/10.3390/app122412644

18. L. F. Gonzalez Casanova, P. C. Lin, Generalized classification of DNS over HTTPS traffic with deep learning, in *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, (2021), 1903–1907.

19. E. Guillen, B. Uguen, C. Moy, J. Le Masson, Temporal analysis of LoRaWAN data packets: Unveiling patterns for improving secure-oriented IoT designs, in *2024 20th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*, (2024), 532–539. https://doi.org/10.1109/DCOSS-IoT61029.2024.00084

20. X. Liu, Y. Zhang, X. Yang, W. Gai, B. Sun, Mfc-doh: DoH tunnel detection based on the fusion of maml and F-CNN, in *Proceedings of the 21st ACM International Conference on Computing Frontiers (CF '24)*, (2024), 267–275. https://doi.org/10.1145/3649153.3649207

21. H. Li, D. Pimentel-Alarcón, Deep fusion: Capturing dependencies in contrastive learning via transformer projection heads, preprint, arXiv: 2403.18681. https://doi.org/10.48550/arXiv.2403.18681

22. R. Mitsuhashi, Y. Jin, K. Iida, T. Shinagawa, Y. Takai, Malicious DNS tunnel tool recognition using persistent DoH traffic analysis, *IEEE Trans. Network Ser. Manage.*, **20** (2023), 2086–2095. https://doi.org/10.1109/TNSM.2022.3215681

23. D. Li, D. Sun, C. Zeng, Research on abnormal network traffic detection based on 1D-CNN, in *2021 International Conference on Neural Networks, Information and Communication Engineering*, **11933** (2021), 138–145. https://doi.org/10.1117/12.2615167

24. K. J. Pradeep, P. Mishra, Detection and prevention of DDoS attack packets on the distributed network using Bi-LSTM network, *Webology*, **19** (2022), 1–12.

25. R. Liu, Y. Ma, X. Gao, L. Zhang, Real-time traffic intrusion detection based on CNN-LSTM deep neural networks, in *International Conference on Computer Network Security and Software Engineering (CNSSE 2024)*, **13175** (2024), 82–89. https://doi.org/10.1117/12.3031914

26. R. Alenezi, S. A. Ludwig, Classifying DNS tunneling tools for malicious DoH traffic, in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, (2021), 1–9. https://doi.org/10.1109/SSCI50451.2021.9660136

27. R. Mitsuhashi, A. Satoh, Y. Jin, K. Iida, T. Shinagawa, Y. Takai, Identifying malicious DNS tunnel tools from DoH traffic using hierarchical machine learning classification, in *Information Security. ISC 2021. Lecture Notes in Computer Science* (eds. J.K. Liu, S. Katsikas, W. Meng, W. Susilo, R. Intan), Springer, Cham, **13118** (2021), 245–260. https://doi.org/10.1007/978-3-030-91356-4_13

28. J. Liang, S. Wang, S. Zhao, S. Chen, FECC: DNS tunnel detection model based on CNN and clustering, *Comput. Secur.*, **128** (2023), 103132. https://doi.org/10.1016/j.cose.2023.103132