Retrieval-Augmented Few-Shot Encrypted Traffic Classification

Abstract

With the widespread adoption of network encryption technologies, encrypted malicious traffic classification has become an important research direction in the field of cybersecurity [1–3]. However, in real-world network environments, labeled malicious samples are scarce and annotation costs are high, causing existing methods to face significant performance bottlenecks under few-shot conditions. To address this problem, this paper proposes a lightweight retrieval-augmented encrypted traffic classification model, **RAC-ET** (Retrieval-Augmented Classification for Encrypted Traffic). The proposed method first converts raw traffic payload bytes into token sequences and extracts deep semantic features using a pre-trained ET-BERT model; then constructs a vector-indexed sample knowledge base for storing high-dimensional representations of historical samples; during inference, efficient vector retrieval obtains the Top-K nearest neighbor samples most semantically similar to the query sample; finally, a cross-attention mechanism fuses query features with retrieved features for accurate classification. Experimental results on three benchmark datasets demonstrate that: on the CICIDS2017 dataset, RAC-ET achieves an average accuracy improvement of 6.72% over the baseline model ET-BERT under few-shot scenarios, with the largest improvement of $+11.22\%$ at the 10-shot setting ($70.31\% \rightarrow 81.53\%$); on the IoT-23 dataset under few-shot scenarios, the average accuracy improvement reaches $+20.09\%$, with the 3-shot setting achieving up to $+28.15\%$ ($20.43\% \rightarrow 48.57\%$); on the data-sufficient CSTNET-TLS 1.3 dataset (120 classes), the proposed method achieves 91.75% accuracy under the full-data setting, a 3.0% improvement over the baseline.

Keywords: Encrypted traffic classification; few-shot learning; retrieval-augmented classification; pre-trained model; cybersecurity

1 Introduction

With the popularization of 5G communication technology and the continuous improvement of digital infrastructure, network traffic volumes continue to grow, and cyberattack activities exhibit increasing frequency and sophistication. The Imperva *2025 Bad Bot Report* and the FBI's 2026 public advisories on cryptocurrency and AI fraud indicate that malicious automated traffic remains at persistently high levels, with generative AI-based scam activities growing significantly in recent years; meanwhile, the direct and indirect economic losses from cybercrime continue to expand. Publicly available data shows that malicious bot traffic now accounts for approximately 37% of all internet traffic, while losses to American victims from cryptocurrency-related and AI-assisted fraud have reached billions of dollars [4, 5]. For most attack chains, malicious traffic transmission is a key vehicle for infiltration, lateral movement, and data exfiltration; therefore, high-precision malicious traffic detection has become a fundamental capability of network defense systems.

However, attackers widely use encryption protocols such as TLS/SSL to conceal communication semantics, significantly limiting traditional detection methods that rely on plaintext content (e.g., DPI). Especially against the backdrop of rapid IoT device proliferation, endpoint heterogeneity and attack surfaces expand simultaneously. The high stealthiness, strong dynamics, and class imbalance of encrypted traffic further

exacerbate the challenges for malicious traffic identification. Prior research has pointed out that malware propagation increasingly relies on encrypted connections such as HTTPS and TLS 1.3 [2,6], rendering traditional deep packet inspection (DPI) methods progressively ineffective. How to accurately identify malicious behavior in massive, highly stealthy encrypted traffic has become a critical issue for safeguarding digital assets.

Currently, deep-learning-based encrypted malicious traffic detection methods can be categorized into three mainstream technical approaches. The first is **CNN-based methods**. These methods typically treat raw traffic byte streams as one-dimensional signals or map them to two-dimensional representations for processing. In representative work, Wang et al. proposed an end-to-end encrypted traffic classification framework based on 1D-CNN that directly extracts local spatial features from raw traffic [7]. However, CNN-based methods primarily focus on local patterns, often neglecting long-range temporal dependencies between packets, and fixed input length truncation may cause loss of critical information. The second is **RNN/LSTM-based temporal modeling methods**. These methods treat network traffic as time series and use memory mechanisms to capture contextual dependencies; the representative work by Rezaei and Liu systematically surveyed and validated the effectiveness of RNN/LSTM/GRU approaches in encrypted traffic classification [2]. Although these methods perform well in handling variable-length sequences, RNN architectures have low parallel efficiency and may still suffer from gradient decay with long sequences, limiting stable capture of long-flow features. The third is **pre-trained model-based methods**. These methods typically perform self-supervised pre-training on large-scale unlabeled traffic before transferring to downstream classification tasks, thereby enhancing feature representation capability. The representative ET-BERT leverages the BERT architecture from NLP, using Transformer encoders to capture long-range dependencies and learning context-aware dynamic embeddings through masked language modeling, achieving excellent performance in data-sufficient scenarios [3]. However, the fine-tuning process of such models is highly dependent on abundant and balanced labeled data, and in few-shot scenarios, they may suffer from overfitting and significant performance degradation due to insufficient supervisory signals.

Despite this progress, three key challenges persist in practice. (1) **Performance degradation due to limited labeled data.** Malicious samples constitute a small fraction of network traffic and require domain expertise to annotate, making large-scale labeling prohibitively expensive. New malware variants continually emerge with only a few labeled samples per class, severely limiting the fine-tuning performance of existing deep models such as ET-BERT. (2) **Insufficient generalization under few-shot conditions.** Parameterized discriminative models (CNN, RNN/LSTM, fine-tuned ET-BERT) internalize all knowledge in their parameters. When confronted with novel, stealthy, and rapidly evolving attacks, a handful of labeled samples cannot provide sufficient contextual information, leading to unstable decision boundaries. (3) **High cost of model updating and online adaptation.** Supervised deep classifiers—especially large pre-trained models—often require extensive parameter re-fine-tuning or periodic full retraining as attack patterns evolve, making it difficult to meet the low-latency and rapid-iteration requirements of real-world deployment.

To this end, we propose a lightweight retrieval-augmented encrypted traffic classification model, **RAC-ET** (Retrieval-Augmented Classification for Encrypted Traffic). The method first converts raw traffic packet payload bytes into token sequences and uses pre-trained ET-BERT as a feature extractor to obtain deep semantic representations; then constructs a vector-indexed sample knowledge base to store high-dimensional features of historical samples; finally, during inference, retrieves the Top-K nearest neighbor samples most semantically similar to the query and fuses query features with retrieved features through a cross-attention mechanism for accurate classification. By retrieving and integrating similar sample information, the method effectively expands the context available for classification decisions, thereby significantly improving classification performance under label-scarce scenarios.

The main contributions of this paper are as follows: (1) We propose a retrieval-augmented encrypted traffic classification framework that combines pre-trained traffic representations with a similar-sample retrieval mechanism. By introducing an external knowledge base as non-parametric memory, it mitigates the performance degradation of pre-trained models in few-shot scenarios. (2) We design a cross-attention-based feature fusion mechanism that achieves adaptive integration of query features with nearest neighbor features while keeping the model lightweight (training only a small number of parameters). (3) We construct a systematic experimental evaluation scheme covering both few-shot and full-data scenarios, and analyze the impact of key design choices including retrieval scale K, residual connections, and attention layers on model performance through ablation experiments.

2 Related Work

2.1 Encrypted Traffic Classification

Encrypted traffic refers to data streams encapsulated through protocols such as SSL/TLS and SSH, rendering the original payload as high-entropy ciphertext. With the strengthening of global network privacy protection, encrypted traffic now accounts for over 90% of total internet traffic, but this also renders traditional deep packet inspection (DPI) techniques ineffective due to their inability to identify encrypted content. Encrypted traffic classification aims to identify application types or malicious behavior without decrypting the data, leveraging statistical features, temporal patterns, or byte distribution regularities of the traffic. Existing research methods have generally evolved from "handcrafted feature-driven" to "automatic representation learning" approaches, with early work primarily based on traditional machine learning.

Traditional machine learning methods primarily rely on handcrafted statistical features, where experts construct distinguishable features based on prior knowledge of network protocols and flow behaviors (e.g., flow duration, packet length distribution, direction sequences, inter-arrival times, TLS handshake metadata), which are then fed into classifiers such as random forests and support vector machines. In representative work, Anderson et al. in 2018 identified malicious traffic by extracting unencrypted metadata from the TLS handshake phase (such as cipher suites and extension fields) combined with machine learning models [6]; Draper-Gil et al. characterized VPN traffic using time-related features such as flow duration and packet length statistics [8]. These methods offer strong interpretability and low implementation barriers, but they are highly dependent on expert experience for feature engineering and struggle to cope with increasingly stealthy and evolving complex attacks. It should be noted that although the above literature predates the last five years, they represent foundational work in encrypted traffic classification.

Deep learning methods have gradually become mainstream due to their end-to-end automatic feature extraction capabilities. In early work, Wang et al. proposed an end-to-end 1D-CNN framework that takes raw traffic bytes as input, avoiding tedious manual feature selection [7]; Shapira et al. proposed FlowPic, mapping flow packet sizes and arrival times to 2D images for classification using 2D-CNN [9]; Liu et al. proposed FS-Net, which mines deep temporal features through flow sequence modeling and reconstruction mechanisms, achieving good results in encrypted traffic classification [10]. Research in the last five years has further focused on "pre-training + lightweight + generalization": Lin et al. proposed ET-BERT, treating network packets as text and learning contextual representations through masked language modeling [3]; subsequently, BERT-Packet-Header and MetaRockETC further improved encrypted traffic classification performance on public datasets, addressing header information encoding and cross-scenario generalization [11, 12]. Addi-

tionally, recent surveys and empirical studies have noted that deep models still face robustness and transferability challenges in cross-domain deployment and class evolution scenarios [13, 14]. In summary, existing methods have evolved from traditional machine learning to deep learning, and further to pre-trained representation learning paradigms, but "stable generalization under few-shot conditions" remains a core challenge in this field.

2.2 Few-Shot Learning

Few-shot learning (FSL) focuses on achieving effective classification with very few labeled samples, with its core goal being to improve the model's ability to rapidly adapt to new classes and tasks. In cybersecurity scenarios, where malicious traffic annotation relies on expert experience, sample acquisition costs are high, and attack types evolve rapidly, FSL has strong application significance.

Existing FSL methods mainly include two technical approaches. The first is meta-learning methods, which train models "how to learn" through numerous auxiliary tasks. Vinyals et al. proposed Matching Networks, establishing end-to-end matching relationships between support sets and query samples through attention mechanisms, enabling models to perform discrimination based on inter-sample similarity [16]; Snell et al. proposed Prototypical Networks, mapping each class's samples to prototype vectors in feature space and using distances from query samples to class prototypes for classification, with a simple structure and good stability [15]; Finn et al. proposed MAML, learning a set of easily transferable initialization parameters through bi-level optimization, enabling models to adapt to new tasks with only a few gradient updates [17]. These methods emphasize task-level knowledge transfer and have achieved good results on standard few-shot benchmarks.

The second is transfer learning and parameter-efficient fine-tuning methods, whose basic idea is to first obtain universal representations using large-scale data, then complete target task adaptation by updating only a small number of additional parameters. Houlsby et al. proposed Adapter, inserting lightweight bottleneck modules between pre-trained model layers, requiring only the training of newly added parameters to achieve cross-task transfer [18]; Hu et al. proposed LoRA, constraining weight updates to low-rank decomposition form, significantly reducing fine-tuning costs with essentially no increase in inference overhead [19]. Compared to full-parameter fine-tuning, these methods are more suitable for application environments with limited samples and constrained computational resources.

In the encrypted traffic classification domain, Yang et al. proposed MetaMRE, combining meta-learning mechanisms with representation enhancement strategies, improving model generalization in few-shot traffic classification through task-level adaptive feature transformation, validating the feasibility of FSL in network traffic scenarios [20].

However, the above methods still have several limitations in encrypted malicious traffic detection tasks. First, meta-learning methods typically rely on a large number of auxiliary tasks with distributions similar to the target task for training, but in real network environments, traffic classes evolve rapidly and distribution drift is significant, making offline-constructed training tasks difficult to adequately cover actual attack patterns. Second, whether Matching Networks, Prototypical Networks, or MAML, these methods mostly focus on extracting discriminative evidence from within the current support set; when samples per class are extremely few and intra-class variance is large, models are susceptible to insufficient representativeness of support samples. Third, parameter-efficient fine-tuning methods such as Adapter and LoRA, while reducing

training costs, still primarily store knowledge in model parameters, with limited ability to leverage newly added attack types and historically similar samples, unable to dynamically absorb external information like explicit memory mechanisms. Therefore, relying solely on meta-learning or lightweight fine-tuning remains insufficient for stably solving the generalization and online adaptation problems in few-shot encrypted traffic classification, which also provides motivation for introducing retrieval-augmented external sample memory.

2.3 Retrieval-Augmented Methods

Retrieval-augmented (RA) techniques aim to bridge the limitations of deep learning models in long-tail knowledge memorization and real-time data updating by introducing external knowledge bases. Their core logic is the synergistic fusion of the model's parametric knowledge with non-parametric knowledge from external knowledge bases. Categorized by task objectives, existing research has primarily evolved along two paths: Retrieval-Augmented Generation (RAG) and Retrieval-Augmented Classification (RAC).

Retrieval-Augmented Generation (RAG) is the representative paradigm in current NLP. A typical RAG system consists of a neural retriever and a generator: during inference, the model first retrieves relevant passages from an external corpus based on the query, then uses them as context to guide the generator's output [21]. To improve retrieval quality, subsequent work developed dual-encoder schemes centered on dense vector retrieval (e.g., DPR), significantly enhancing recall and semantic matching quality in knowledge-intensive tasks [22]; the recent Self-RAG further introduces on-demand retrieval and self-reflection mechanisms to improve generation quality and factual consistency [23]. However, RAG methods typically rely on relatively large generative models, with long inference chains and high computational overhead, making them difficult to directly satisfy the engineering constraints of low latency and high throughput in network traffic detection scenarios.

Unlike generation tasks, Retrieval-Augmented Classification (RAC) aims to retrieve similar samples to assist discrimination. This method can be traced back to non-parametric memory augmentation methods such as kNN-LM, which introduce nearest-neighbor retrieval beyond parametric models to calibrate prediction distributions [24]; in recent years, related research has further adopted Top-K nearest-neighbor retrieval in dense vector spaces combined with feature fusion mechanisms, improving model robustness under long-tail classes and low-resource sample conditions. For example, in few-shot text classification, retrieval-augmented training objectives have been used to stably improve performance in low-resource scenarios [25], and in long document classification, retrieval-augmented discriminative frameworks balancing efficiency and interpretability have been developed [26]. In contrast, in the encrypted traffic classification domain, existing research still primarily focuses on feature representation learning and parameterized classifier optimization, with relatively little discussion on the idea of "using external similar samples to provide auxiliary context for current traffic discrimination," especially under few-shot conditions, where how to effectively convert retrieved historical samples into stable classification evidence remains insufficiently studied

To address this gap, this paper draws upon the non-parametric memory augmentation idea from kNN-LM and the core concept of retrieval-assisted discrimination in the RAC paradigm, proposing an improved scheme for few-shot encrypted traffic classification: unlike paths relying on generative large models, this work uses pre-trained ET-BERT to construct a traffic semantic vector knowledge base; during inference, efficient vector retrieval obtains the historical support samples most similar to the query traffic; subsequently, a cross-attention mechanism is introduced to adaptively fuse query features with retrieved features. This design avoids complex text generation processes, enabling more effective utilization of external labeled sample

information, enhancing classification boundary discrimination under few-shot conditions while maintaining the efficiency and stability required for practical deployment, achieving lightweight, dynamically updatable retrieval-augmented encrypted traffic classification.

It is important to emphasize that RAC-ET is fundamentally different from the traditional k-nearest neighbor (kNN) classifier in three key aspects. (1) kNN directly uses majority voting over retrieved neighbor labels as the final prediction, constituting a *purely non-parametric* method. In contrast, RAC-ET fuses the retrieved neighbor features with the query feature through a *learnable cross-attention module*, then feeds the result into a parametric classification head—the retrieved results serve as "contextual augmentation" rather than direct votes. (2) kNN is highly sensitive to neighbor quality, where noisy retrievals directly corrupt the classification outcome. RAC-ET's attention mechanism can learn to suppress the weights of noisy neighbors; as shown in the ablation study (Table 7), removing the attention mechanism leads to significant performance degradation, confirming the necessity of adaptive fusion for leveraging retrieval information. (3) kNN has no training phase and cannot perform end-to-end optimization of the retrieval–fusion process. RAC-ET jointly optimizes the cross-attention, FFN, and classification head through supervised loss, enabling the model to learn *how to optimally utilize retrieval results*.

3 Method

This section presents the proposed retrieval-augmented encrypted traffic classification model **RAC-ET** (Retrieval-Augmented Classification for Encrypted Traffic). Following the organizational structure of the "Method" chapter, this paper describes the model components and their coordination mechanisms in the order of "overall framework—data preprocessing—feature representation—retrieval augmentation—fusion and discrimination." To avoid notation ambiguity, we use K_s to denote the number of labeled samples per class in the few-shot setting (i.e., K_s-shot), and K_r to denote the number of retrieved neighbors (Top-K_r).

3.1 Framework Overview

The overall architecture of RAC-ET is illustrated in Figure 1, comprising five main stages: data preprocessing, feature extraction, knowledge base construction and nearest-neighbor retrieval, cross-attention fusion, and classification decision. Unlike traditional encrypted traffic detection methods that rely solely on parameterized classifiers, RAC-ET retains the semantic representation capability of the pre-trained model while introducing external sample memory as retrieval context, enabling the model to achieve stable discrimination with the help of historically similar samples even under few-shot conditions. Its core components can be summarized as:

- **Feature Extractor**: Uses the pre-trained ET-BERT [3] to encode raw traffic byte sequences into dense feature vectors;
- **Knowledge Base**: A vector database storing the feature representations of available traffic samples;
- **Retrieval Module**: Retrieves the nearest neighbors most relevant to the query sample from the knowledge base through efficient similarity search;

- **Fusion Classifier**: Fuses query features and retrieved features using a cross-attention mechanism and outputs the final class prediction.

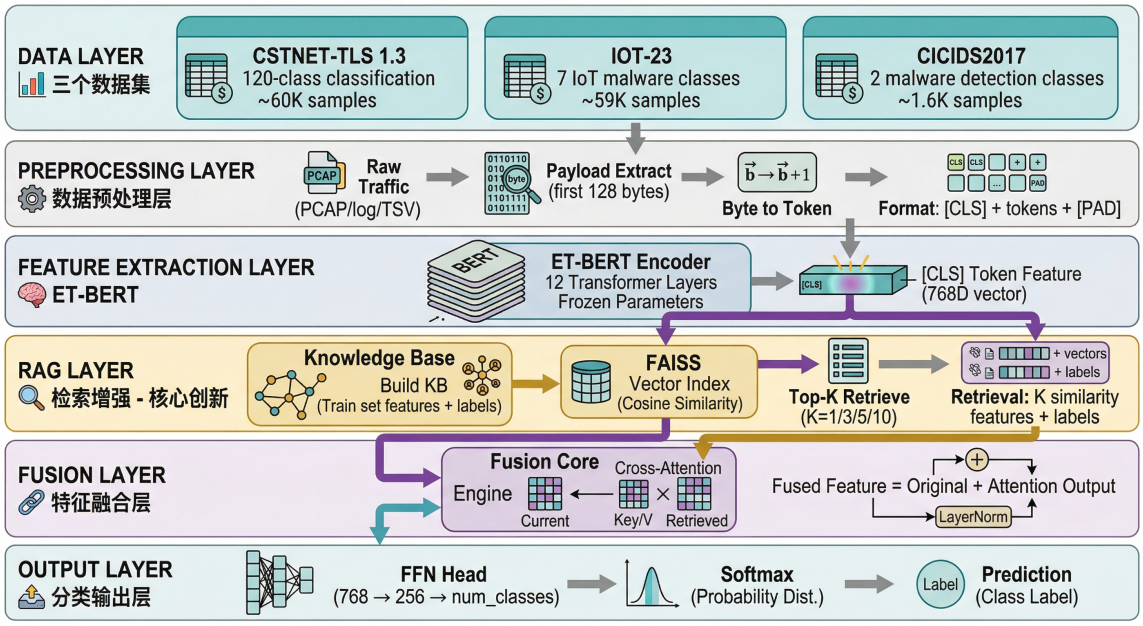

Figure 1: Overall architecture of the RAC-ET framework. Input traffic is encoded by ET-BERT into query features, then Top-K_r similar samples are retrieved from the knowledge base, fused through cross-attention and a feed-forward network, and finally classified by the classification head.

Given an input traffic sample x, the overall pipeline of RAC-ET proceeds as follows:

1. Extract the query feature using the frozen ET-BERT encoder:
$$\mathbf{h}_q = f(x). \tag{1}$$
2. Retrieve the Top-K_r similar samples from the knowledge base:
$$\mathcal{R}(x) = \{(x_1, y_1), \ldots, (x_{K_r}, y_{K_r})\}. \tag{2}$$
3. Extract the feature representations of the retrieved samples:
$$\{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{K_r}\}. \tag{3}$$
4. Fuse the query feature with the retrieved features through the cross-attention module to obtain the augmented representation $\mathbf{h}_{\text{fused}}$;
5. Feed the fused representation into the classification head to produce the final prediction.

3.2 Data Preprocessing

Following the input specifications of ET-BERT [3], this paper first converts raw encrypted traffic into discrete token sequences suitable for processing by the Transformer encoder. This process corresponds to the data

standardization step before model training, with the goal of preserving behavioral semantics in the payload as much as possible while unifying input length and representation format. The entire preprocessing pipeline consists of three stages.

Stage 1: Packet Extraction For each network flow, payload bytes are first extracted from the transport layer (TCP/UDP). The specific steps are as follows:

1. Parse the Ethernet frame and extract the IP packet;
2. Identify the transport layer protocol type;
3. Extract the payload bytes following the transport layer header;
4. Truncate or pad the payload byte sequence to a fixed length $L_0 = 128$.

Stage 2: Byte-to-Token Conversion For each byte $b_j \in \{0, \ldots, 255\}$ in the payload, a domain-specific vocabulary mapping converts it to a token t_j:

$$t_j = \begin{cases} b_j + 1, & b_j \in \{0, \ldots, 255\}, \\ 0, & \text{if the position is padding.} \end{cases} \tag{4}$$

Therefore, the vocabulary size is 257, where token 0 represents the padding symbol and tokens $1 \sim 256$ correspond to byte values 0x00–0xFF.

Stage 3: Sequence Formatting A [CLS] token is prepended to the token sequence, and sequences shorter than L_0 are padded with [PAD] tokens. The final input format is

$$[\text{CLS}],\ t_1,\ t_2,\ \ldots,\ t_n,\ [\text{PAD}],\ \ldots,\ [\text{PAD}]. \tag{5}$$

Thus, the actual input length for ET-BERT is

$$T = L_0 + 1 = 129. \tag{6}$$

3.3 ET-BERT-Based Feature Extraction

After tokenization, this paper employs ET-BERT as the backbone feature extractor for unified encoding of traffic sequences. ET-BERT is a BERT model pre-trained for network traffic scenarios, which learns contextual dependencies through masked language modeling on large-scale unlabeled traffic, enabling effective capture of statistical patterns and structural information in encrypted traffic.

Given an input sequence x, ET-BERT produces contextualized representations for each token:

$$\mathbf{H} = \text{ET-BERT}(x) \in \mathbb{R}^{T \times d}, \tag{7}$$

where T denotes the input sequence length and $d = 768$ denotes the hidden dimension.

This paper adopts the hidden state at the [CLS] position as the global representation of the entire traffic sample:

$$\mathbf{h} = \mathbf{H}_{[\text{CLS}]} \in \mathbb{R}^d. \tag{8}$$

During both training and inference, the ET-BERT parameters are kept frozen to fully preserve pre-trained knowledge and reduce fine-tuning overhead.

3.4 Knowledge Base Construction

The knowledge base stores the feature representations and label information of available traffic samples, providing external context support for query samples during inference. For each available sample (x_i, y_i), its representation is first obtained through the frozen feature extractor:

$$\mathbf{h}_i = f(x_i). \tag{9}$$

Subsequently, the knowledge base is constructed as:

$$\mathcal{KB} = \{(\mathbf{h}_i, y_i) \mid i = 1, 2, \ldots, N\}. \tag{10}$$

This paper uses FAISS [28] to build the vector index for efficient nearest-neighbor retrieval. Since cosine similarity is used in the retrieval stage, to be consistent with the inner product search of `IndexFlatIP`, all feature vectors are L_2-normalized before indexing and querying. For notational simplicity, we continue to denote the normalized vectors as $\mathbf{h}$.

In few-shot scenarios, for a C-way K_s-shot task, the knowledge base constructed from labeled data alone has a size of $K_s \times C$. Additionally, if extra unlabeled or reserved samples are available, their feature representations can also be added to the retrieval candidate pool to provide richer contextual information; such samples do not participate in supervised loss computation.

For larger-scale knowledge bases, approximate nearest-neighbor index structures such as IVF and HNSW can be further employed to improve retrieval efficiency beyond exact search.

3.5 Retrieval Module

After obtaining the semantic representation of the query sample, the retrieval module selects the Top-K_r most similar samples from the knowledge base as support instances for the current classification task. This paper employs cosine similarity as the similarity metric:

$$s(\mathbf{h}_q, \mathbf{h}_i) = \frac{\mathbf{h}_q^\top \mathbf{h}_i}{\|\mathbf{h}_q\|_2 \, \|\mathbf{h}_i\|_2}. \tag{11}$$

Since the features are L_2-normalized, the above equation can be equivalently written as the inner product:

$$s(\mathbf{h}_q, \mathbf{h}_i) = \mathbf{h}_q^\top \mathbf{h}_i. \tag{12}$$

Therefore, the retrieval set can be expressed as:

$$\mathcal{R}(x) = \text{TopK}_{(\mathbf{h}_i, y_i) \in \mathcal{KB}}\, s(\mathbf{h}_q, \mathbf{h}_i). \tag{13}$$

The resulting retrieval set is:

$$\mathcal{R}(x) = \{(\mathbf{h}_1, y_1), (\mathbf{h}_2, y_2), \ldots, (\mathbf{h}_{K_r}, y_{K_r})\}, \tag{14}$$

where each nearest neighbor consists of its feature vector and label. This process essentially introduces a dynamically updatable layer of non-parametric memory in the parameterized representation space, helping to improve the model's discriminative stability for long-tail and few-shot classes.

3.6 Cross-Attention Fusion

To effectively leverage the retrieved contextual information, this paper designs a cross-attention fusion module that selectively aggregates query features and retrieved features [29]. Unlike direct concatenation or average pooling, this module can adaptively assess the importance of different neighbor samples based on the semantic needs of the current query sample. Let the query feature be $\mathbf{h}_q \in \mathbb{R}^{1\times d}$, and stack the K_r retrieved features as:

$$\mathbf{H}_r = [\mathbf{h}_1; \mathbf{h}_2; \ldots; \mathbf{h}_{K_r}] \in \mathbb{R}^{K_r \times d}. \tag{15}$$

The cross-attention computation proceeds as:

$$\mathbf{Q} = \mathbf{h}_q \mathbf{W}_Q, \tag{16}$$

$$\mathbf{K} = \mathbf{H}_r \mathbf{W}_K, \tag{17}$$

$$\mathbf{V} = \mathbf{H}_r \mathbf{W}_V, \tag{18}$$

where $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d\times d_k}$ are learnable projection matrices. The attention output is then:

$$\mathbf{z} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}. \tag{19}$$

To ensure dimensional consistency with the original query feature, an output projection matrix $\mathbf{W}_O \in \mathbb{R}^{d_k \times d}$ is introduced, and a residual connection with layer normalization yields:

$$\mathbf{h}_{\text{attn}} = \text{LayerNorm}\left(\mathbf{h}_q + \mathbf{z}\mathbf{W}_O\right). \tag{20}$$

On top of this, a feed-forward network (FFN) further enhances feature expressiveness:

$$\text{FFN}(\mathbf{x}) = \text{ReLU}(\mathbf{x}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2, \tag{21}$$

$$\mathbf{h}_{\text{fused}} = \text{LayerNorm}\left(\mathbf{h}_{\text{attn}} + \text{FFN}(\mathbf{h}_{\text{attn}})\right). \tag{22}$$

This mechanism adaptively assigns weights to different retrieved samples based on the semantic representation of the query sample, thereby highlighting the contextual information most relevant to the current classification decision and suppressing interference from irrelevant or noisy neighbors.

3.7 Classification Head

After obtaining the fused representation, this paper employs a lightweight linear classification head for final discrimination. The fused feature $\mathbf{h}_{\text{fused}}$ passes through a linear classification layer and softmax to obtain the class probability distribution:

$$\mathbf{p} = \text{softmax}\left(\mathbf{h}_{\text{fused}}\mathbf{W}_c + \mathbf{b}_c\right), \tag{23}$$

where $\mathbf{W}_c \in \mathbb{R}^{d\times C}$, $\mathbf{b}_c \in \mathbb{R}^C$, and C denotes the number of classes.

3.8 Training Objective

This paper employs cross-entropy loss to train the model, minimizing the divergence between the predicted class distribution and the true label distribution. For a training set containing N samples, the loss function is defined as:

$$\mathcal{L}_{\text{ce}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log p_{i,c}, \tag{24}$$

where $y_{i,c}$ is the one-hot label of sample i for class c, and $p_{i,c}$ is the model's predicted probability of assigning sample i to class c.

Since the ET-BERT backbone is kept frozen, only the cross-attention module, FFN, and classification head parameters are updated during training, thereby significantly reducing optimization overhead and making the model more suitable for rapid adaptation under few-shot conditions.

3.9 Parameter Efficiency Analysis

By freezing the ET-BERT backbone, RAC-ET requires training only a small number of additional parameters to complete the downstream classification task. Table 1 presents the parameter distribution across modules.

Table 1: Parameter efficiency analysis of the RAC-ET framework.

Component	Parameters	Trainable
ET-BERT Backbone	132.2M	No (Frozen)
Cross-Attention Module	2.36M	Yes
FFN Layer	2.36M	Yes
Classification Head	0.35M	Yes
Total	137.3M	5.07M (3.69%)

As shown, the proposed method trains only 3.69% of the total parameters, significantly reducing computational cost while still fully leveraging the representational power of the pre-trained model and the contextual information provided by retrieval augmentation.

3.10 Overall Algorithm

Algorithm 1 presents the complete pseudocode for the training and inference pipeline of RAC-ET.

Algorithm 1 RAC-ET: Retrieval-Augmented Classification for Encrypted Traffic

Require: Pre-trained encoder $\mathcal{E}$ (frozen); training set $\mathcal{D}_{\text{train}}$; test set $\mathcal{D}_{\text{test}}$; per-class sample count K_s; retrieval size K_r

Ensure: Predicted labels for test set $\hat{\mathbf{y}}$

// Phase 1: Feature Extraction and Knowledge Base Construction
for $(\mathbf{x}_i, y_i) \in \mathcal{D}_{\text{train}}$ **do**
  $\mathbf{h}_i \leftarrow \mathcal{E}(\mathbf{x}_i)$ ▷ Extract 768-dim feature
  $\mathbf{h}_i \leftarrow \mathbf{h}_i / \|\mathbf{h}_i\|_2$ ▷ L_2 normalization
end for
Build FAISS index $\mathcal{I} \leftarrow \text{IndexFlatIP}(\{\mathbf{h}_i\})$
// Phase 2: Fusion Module Training
for epoch $= 1, \ldots, E$ **do**
  for mini-batch $\mathcal{B} \subset \mathcal{D}_{\text{train}}$ **do**
    $\mathbf{h}_q \leftarrow \mathcal{E}(\mathbf{x}_q)$ ▷ Query feature
    $\mathbf{h}_{r_1}, \ldots, \mathbf{h}_{r_{K_r}} \leftarrow \text{FAISS-Search}(\mathcal{I}, \mathbf{h}_q, K_r)$
    $\mathbf{h}_{\text{fused}} \leftarrow \text{CrossAttn}(\mathbf{h}_q, [\mathbf{h}_{r_1}; \ldots; \mathbf{h}_{r_{K_r}}])$
    $\mathbf{h}_{\text{out}} \leftarrow \text{FFN}(\mathbf{h}_{\text{fused}}) + \mathbf{h}_q$ ▷ Residual connection
    $\hat{y}_q \leftarrow \text{softmax}(\mathbf{W}_c \cdot \mathbf{h}_{\text{out}})$
    Backpropagate and update CrossAttn, FFN, $\mathbf{W}_c$ parameters
  end for
end for
// Phase 3: Inference
for $\mathbf{x}_t \in \mathcal{D}_{\text{test}}$ **do**
  $\mathbf{h}_t \leftarrow \mathcal{E}(\mathbf{x}_t) / \|\mathcal{E}(\mathbf{x}_t)\|_2$
  $\mathbf{h}_{r_1}, \ldots, \mathbf{h}_{r_{K_r}} \leftarrow \text{FAISS-Search}(\mathcal{I}, \mathbf{h}_t, K_r)$
  $\hat{y}_t \leftarrow \arg\max \ \text{softmax}(\mathbf{W}_c \cdot (\text{FFN}(\text{CrossAttn}(\mathbf{h}_t, [\mathbf{h}_r])) + \mathbf{h}_t))$
end for
return $\hat{\mathbf{y}}$

4 Experiments

This section systematically evaluates RAC-ET on three benchmark datasets: CSTNET-TLS 1.3 (application identification), IoT-23 (IoT malware identification), and CICIDS2017 (network intrusion detection). Centered on the core question—*can retrieval augmentation stably improve the performance of pre-trained models in few-shot encrypted traffic classification?*—we present the experimental setup (4.1), main results (4.2), K value and component ablation (4.3), efficiency analysis (4.4), CICIDS2017 in-depth analysis (4.5), and discussion (4.6).

4.1 Experimental Setup

Datasets This paper selects three public datasets to cover different classification granularities, data scarcity levels, and input modalities.

CSTNET-TLS 1.3: A large-scale TLS 1.3 encrypted traffic dataset for application classification, containing 120 classes, representing the *many-class, data-sufficient* application identification scenario. This paper follows the official ET-BERT preprocessing pipeline, with approximately 50,000 training samples and approximately 10,000 test samples.

IoT-23: A public dataset for IoT malware family identification. The original data contains 23 attack scenarios, which we map to {Benign, Okiru, Torii, C&C, DDoS, Attack, PortScan} totaling 7 classes per official annotations; after preprocessing, the training set contains approximately 1.38×10^5 samples and the test set approximately 3.4×10^4 samples, representing the *few-class but cross-scenario, highly class-imbalanced* malicious traffic detection scenario.

CICIDS2017: A classic public dataset for network intrusion detection [30], containing 5 days of network traffic capture, using 78-dimensional flow statistical features extracted by CICFlowMeter as input. This paper merges the original 15 attack classes into 7 semantically similar classes (Benign, Brute_Force, DoS, DDoS, Botnet, PortScan, Web_Attack), filtering rare classes with fewer than 50 samples (Heartbleed, Infiltration); the training set contains approximately 35,000 samples and the test set approximately 8,800 samples, representing the *flow-feature input, cross-modal generalization validation* scenario. The overall statistics of the three datasets are summarized in Table 2.

Table 2: Summary statistics of the experimental datasets.

Dataset	Classes	Train Samples	Test Samples	Feature Type
CSTNET-TLS 1.3	120	∼50,000	∼10,000	Byte Payload
IoT-23	7	∼138,000	∼34,000	Byte Payload
CICIDS2017	7	∼35,000	∼8,800	Flow Features

Few-Shot Settings For each class in each dataset, $K_s \in \{1, 3, 5, 10, 20\}$ samples are drawn from the training set to form the few-shot training set, while the remaining training samples serve as candidates added to the knowledge base but do not participate in supervised loss computation. The few-shot training set is used for both training the fusion module (cross-attention + FFN + classification head) and constructing the initial knowledge base. To reduce sampling variance, all few-shot experiments are repeated with 3 different random seeds (42/123/456), reporting the mean of accuracy (Acc) and Macro-F1.

Backbone and Retrieval Configuration Feature extraction uses the officially released ET-BERT model [3], with hidden dimension $d = 768$. For CSTNET, the corresponding `finetuned_model.bin` and `encryptd_vocab.txt` are used; for IoT-23, the corresponding `finetuned_model.bin` with the same byte-level vocabulary is used. ET-BERT parameters are frozen during both training and inference. Input sequences are preprocessed following Section 3.2 into token sequences of length $T = 129$. The knowledge base uses FAISS [28] with the `IndexFlatIP` index, with all features L_2-normalized before insertion, using inner product as equivalent cosine similarity retrieval. The default retrieval size is $K_r = 1$; specific values are ablated in Section 4.3.

CICIDS2017 Feature Encoding To validate the generalization capability of the RAC-ET framework across different input modalities, 78-dimensional flow statistical features extracted by CICFlowMeter (including flow duration, packet length distributions, inter-arrival times, etc.) are used as input on CICIDS2017. To project these flow features into the same 768-dimensional representation space as ET-BERT, a lightweight flow feature encoder is designed (a two-layer fully connected network with intermediate dimension 384, LayerNorm + GELU activation), pre-trained via self-supervised contrastive learning (SimCLR-style, temperature coefficient $\tau = 0.1$) on unlabeled flow features for 50 epochs, *without using any class labels*. This encoder serves the same role as ET-BERT on CSTNET/IoT-23: providing frozen general-purpose representations, allowing the downstream retrieval and classification modules to operate in the identical 768-dimensional space. Semantically similar attack subcategories are merged into 7 classes (Benign, Brute_Force, DoS, DDoS, Botnet, PortScan, Web_Attack), filtering rare classes with fewer than 50 samples (Heartbleed, Infiltration).

Fusion and Training Both the cross-attention module and FFN use single-layer structures, with hidden dimension $d_k = 768$ and 8 attention heads. The optimizer is Adam with learning rate 2×10^{-5}, linear warmup over 10% of steps, trained for 30 epochs under few-shot settings (early stopping patience of 5 epochs), with batch size 32. For full CSTNET training, batch size is 64 with 10 training epochs.

Baselines The experiments compare against the following baselines:

- **ET-BERT** fine-tuning [3]: A linear classification head is attached to ET-BERT's `[CLS]` representation and the entire model is fine-tuned; this shares the same backbone and input as RAC-ET and serves as the most direct control baseline;
- **Random Forest**, **XGBoost**: Traditional machine learning classifiers using ET-BERT global features as input, for comparing the upper bound of traditional discriminators;
- In few-shot baselines, **Prototypical Networks** [15] and **Matching Networks** [16] are additionally included.

Implementation and Hardware All experiments are conducted on a single NVIDIA RTX 3090 Ti (24 GB) GPU, implemented in PyTorch; FAISS uses the CPU version for index construction; evaluation metrics are Accuracy and Macro-F1.

4.2 Main Results

Few-Shot Performance Table 3 (Accuracy) and Table 4 (Macro-F1), Figure 2, Figure 3, and Figure 4 summarize the few-shot results on three datasets under different K_s settings (mean values, $K_r = 1$ or optimal K_r). Overall, RAC-ET performs no worse than the baseline across all settings on all three datasets, with significant improvements on IoT-23 and CICIDS2017.

We observe the following phenomena from the above results:

Table 3: Main few-shot classification results (Accuracy, %). ΔAcc is the absolute improvement of RAC-ET over the ET-BERT baseline.

Dataset	Setting	ET-BERT	RAC-ET (Ours)	ΔAcc
CSTNET-TLS 1.3	1-shot	95.44	**95.44**	+0.00
	5-shot	95.38	**95.60**	+0.22
	10-shot	95.52	**95.82**	+0.30
IoT-23	1-shot	24.14	**26.35**	+2.21
	3-shot	20.43	**48.57**	+28.15
	5-shot	31.25	**51.35**	+20.11
	10-shot	36.33	**62.56**	+26.23
	20-shot	44.91	**68.65**	+23.73
CICIDS2017	1-shot	35.24	**36.90**	+1.66
	5-shot	60.23	**67.52**	+7.29
	10-shot	70.31	**81.53**	+11.22

Table 4: Main few-shot classification results (Macro-F1, %). ΔF1 is the absolute improvement of RAC-ET over the ET-BERT baseline.

Dataset	Setting	ET-BERT	RAC-ET (Ours)	ΔF1
CSTNET-TLS 1.3	1-shot	95.44	**95.44**	+0.00
	5-shot	95.38	**95.60**	+0.22
	10-shot	95.52	**95.82**	+0.30
CICIDS2017	1-shot	36.31	**38.45**	+2.14
	5-shot	60.70	**66.30**	+5.60
	10-shot	70.74	**80.58**	+9.84

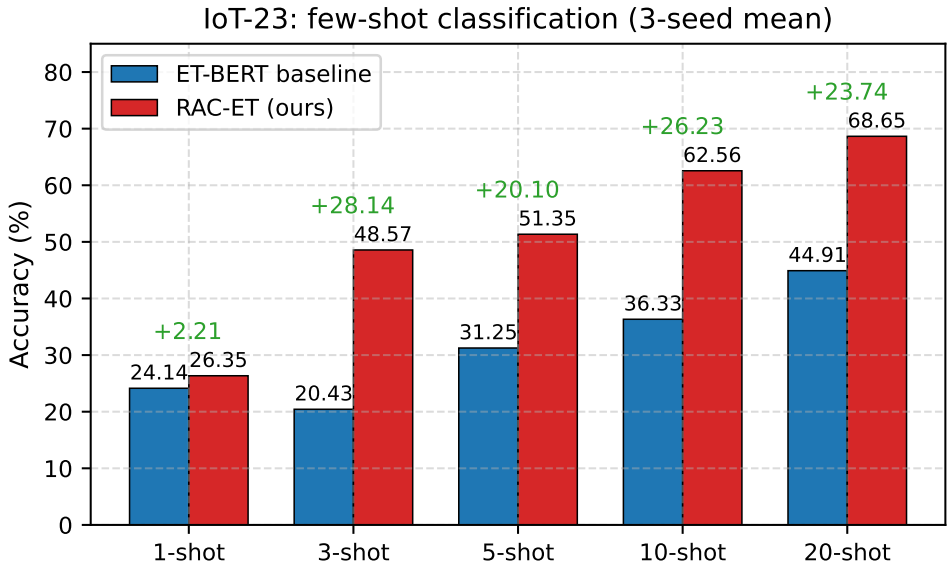

Figure 2: Accuracy comparison between RAC-ET and ET-BERT baseline under different K_s on the IoT-23 dataset (mean of 3 seeds). Green values indicate RAC-ET's absolute improvement over the baseline.

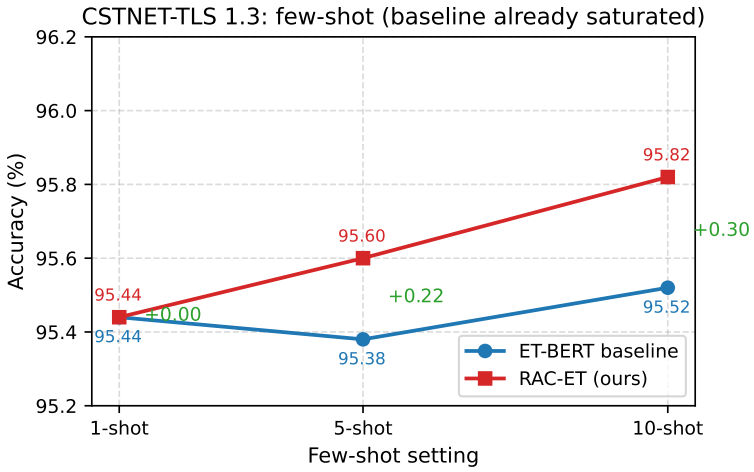

Figure 3: Accuracy comparison under few-shot settings on CSTNET-TLS 1.3. The ET-BERT baseline is already near saturation, yet RAC-ET still achieves $+0.00\% \sim +0.30\%$ consistent improvement without introducing degradation.

1. **Stable performance when semantic coverage is sufficient.** Since ET-BERT was pre-trained via masked language modeling on CSTNET-homologous corpora, its `[CLS]` representation is already very compact for this task, achieving 95.44% accuracy even with only 1-shot; on this basis, RAC-ET achieves a consistent improvement of $+0.00\%$ to $+0.30\%$ through retrieval context without introducing side effects, demonstrating that introducing non-parametric memory does not damage the existing decision boundaries.

2. **Significant improvement when semantic coverage is insufficient.** The malicious traffic distribution of IoT-23 differs substantially from the ET-BERT pre-training corpus, and the `[CLS]` representation alone cannot form stable discrimination among the 7 families, with ET-BERT achieving only 24.14% at 1-shot; RAC-ET improves 3-shot/5-shot/10-shot/20-shot accuracy by $+28.15\%/+20.11\%/+26.23\%/+23.73\%$ respectively, with an average gain of approximately $+20.09\%$, through retrieving and fusing support sample features.

3. **Gains vary with the discriminability of supervised data rather than monotonically with quantity.** At 1-shot, with only 7 support samples, randomness is high and class clusters are sparse, so RAC-ET's improvement is relatively conservative ($+2.21\%$); when $K_s \geq 3$, the number of same-class samples available in the knowledge base rapidly increases, causing the improvement to surge above $+20\%$ and stabilize.

4. **Cross-modal generalization: retrieval augmentation is equally effective in the flow feature space.** On CICIDS2017, using 78-dimensional flow statistical features instead of byte payloads (Figure 4), RAC-ET still demonstrates consistent and significant improvements: 1-shot $+1.66\%$, 5-shot $+7.29\%$, 10-shot $+11.22\%$. This result indicates that RAC-ET's core advantage—extending few-shot decision context through neighbor retrieval—does not depend on a specific backbone encoder and equally holds with a self-supervised pre-trained flow feature encoder. Notably, the absolute improvement at 10-shot reaches 11.22% (from 70.31% to 81.53%), further corroborating the core thesis that "retrieval augmentation yields the greatest benefit when pre-trained representations have insufficient coverage."

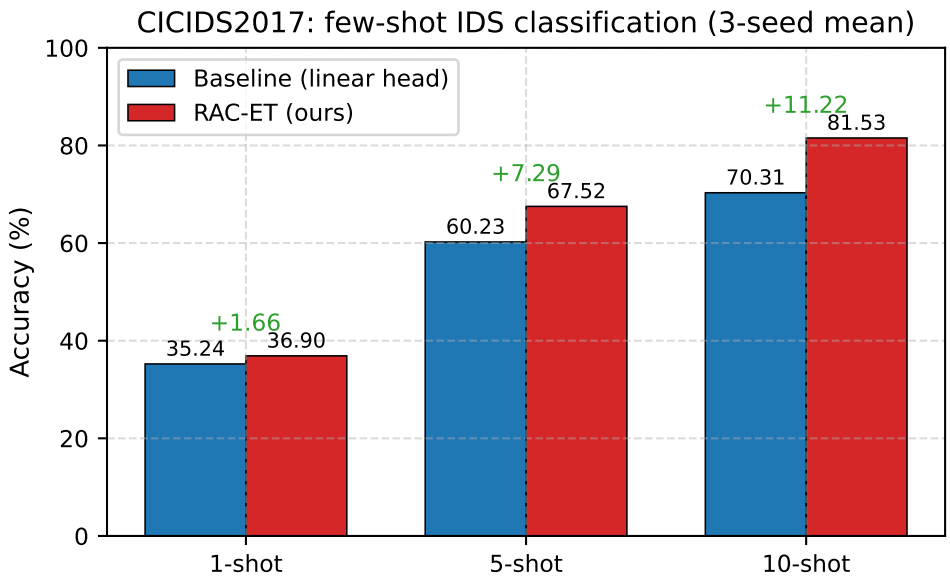

Figure 4: Accuracy comparison between RAC-ET and the baseline under different K_s on the CICIDS2017 dataset (mean of 3 seeds). Using 78-dimensional CICFlowMeter flow features with a self-supervised encoder, RAC-ET achieves $+7.29\%$ and $+11.22\%$ significant improvements at 5-shot and 10-shot respectively.

Full-Data Comparison Table 5 and Figure 5 present the comparison between RAC-ET and several representative baselines under full CSTNET training data. RAC-ET, while keeping the ET-BERT backbone frozen, further achieves $+3.00\%$ accuracy improvement over the fully fine-tuned ET-BERT, and $+2.55\%$ improvement in Macro-F1. Figure 6 further compares the three datasets in terms of average few-shot improvement, clearly showing that RAC-ET's gains primarily come from the IoT-23 and CICIDS2017 scenarios where distribution shift is larger.

Table 5: Baseline comparison under full CSTNET data (%). RF and XGBoost use ET-BERT's `[CLS]` features as input.

Method	Accuracy	Macro-F1
Random Forest (RF)	30.10	27.37
XGBoost	35.55	33.52
ET-BERT (Full Fine-tune)	88.75	89.28
RAC-ET (Ours)	**91.75**	**91.83**

Comparison with Few-Shot Methods Table 6 further compares RAC-ET with typical few-shot learning methods on CSTNET. In the application identification task with sufficient classes and relatively abundant data, Prototypical Networks and RAC-ET perform comparably, both significantly outperforming direct fine-tuning; at 10-shot, RAC-ET maintains competitiveness (95.85%), verifying that retrieval augmentation in such scenarios is *neither degrading nor redundant*.

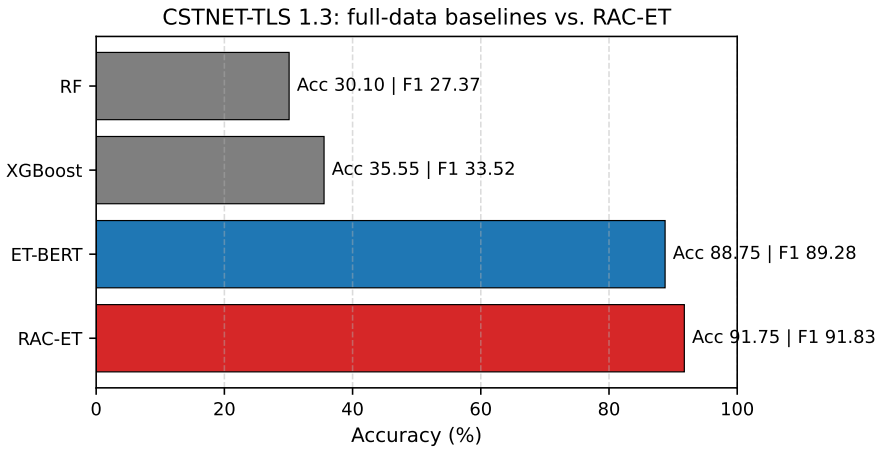

Figure 5: Baseline comparison under full CSTNET-TLS 1.3 data. RAC-ET achieves $+3.00\%$ accuracy and $+2.55\%$ Macro-F1 gains over the full fine-tuning baseline while keeping the ET-BERT backbone frozen.

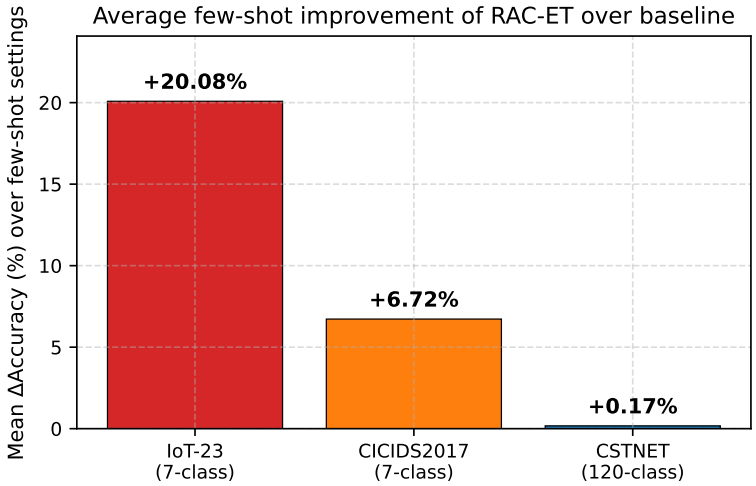

Figure 6: Average few-shot improvement of RAC-ET over the baseline per dataset. The average gains on IoT-23 and CICIDS2017 (larger distribution shift) are significantly higher than on CSTNET (semantically saturated), corroborating that retrieval augmentation primarily compensates for the coverage gaps of pre-trained representations.

Table 6: Comparison with few-shot methods on CSTNET (Accuracy, %).

Method	1-shot	5-shot	10-shot	20-shot
Prototypical Networks	90.61	95.14	96.32	97.82
Matching Networks	90.63	94.65	95.59	96.97
ET-BERT Fine-tune	89.49	95.28	96.73	98.11
RAC-ET (Ours)	90.61	94.85	95.85	97.16

4.3 Ablation Study

Fusion Component Ablation On CSTNET full data, we sequentially remove or replace key components of the RAC-ET fusion module (see Table 7 and Figure 7). Here "+ Attn+FFN (no residual)" denotes retaining attention and FFN but removing the residual connection and LayerNorm. We observe that:

- A simple MLP that directly concatenates query and retrieval features performs *below* the baseline (-8.10%), indicating that naive feature concatenation cannot effectively filter retrieval noise;
- Using attention or FFN alone is weaker than the baseline, confirming that the two must work synergistically;
- Removing the residual connection causes the largest degradation (-13.50%), demonstrating that the residual pathway is crucial for maintaining query feature dominance and gradient flow;
- The complete RAC-ET achieves 91.75%, a net improvement of $+3.00\%$ over the ET-BERT baseline.

Table 7: Fusion module ablation on CSTNET full data (%).

Configuration	Accuracy	Macro-F1	ΔAcc
ET-BERT Baseline	88.75	89.28	–
+ Simple MLP Fusion	80.65	81.02	-8.10
+ Attention Only	83.40	83.85	-5.35
+ FFN Only	83.75	84.21	-5.00
+ Attn+FFN (No Residual)	75.25	75.68	-13.50
Full RAC-ET	**91.75**	**91.83**	$+$**3.00**

Retrieval Size K_r Table 8 and Figure 8 analyze the effect of K_r on IoT-23 with $K_s = 10$ fixed. We observe that $K_r = 1$ achieves the highest accuracy of 62.56%, with accuracy monotonically decreasing as K_r increases. This is because the inter-class differences in IoT-23 are large while intra-class patterns are relatively concentrated: the Top-1 nearest neighbor already provides strong discriminative cues for the query, and increasing K_r further introduces more cross-class or morphologically different same-class samples, which instead inject noise during fusion. Combined with the fusion mechanism, the $K_r = 1$ setting is also the most economical in terms of memory and inference speed.

Table 8: Ablation of K_r on IoT-23 ($K_s = 10$, Accuracy, %).

K_r	Accuracy
1	**62.56**
3	56.55
5	52.50
10	47.65

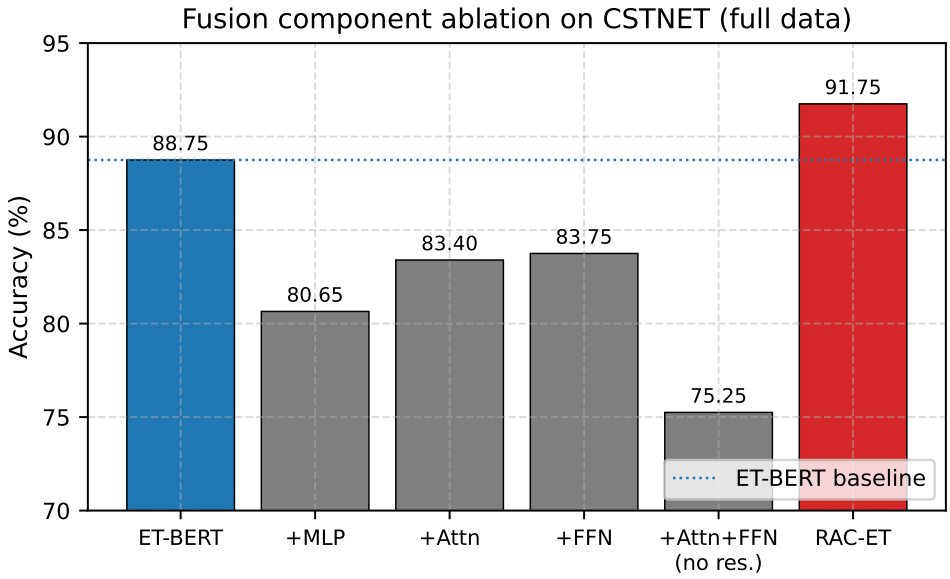

Figure 7: Visualization of fusion component ablation results on CSTNET full data. Using attention only, FFN only, or removing residual connections all fall below the baseline, confirming that cross-attention, feed-forward network, and residual connections must work synergistically. The full RAC-ET achieves 91.75%.

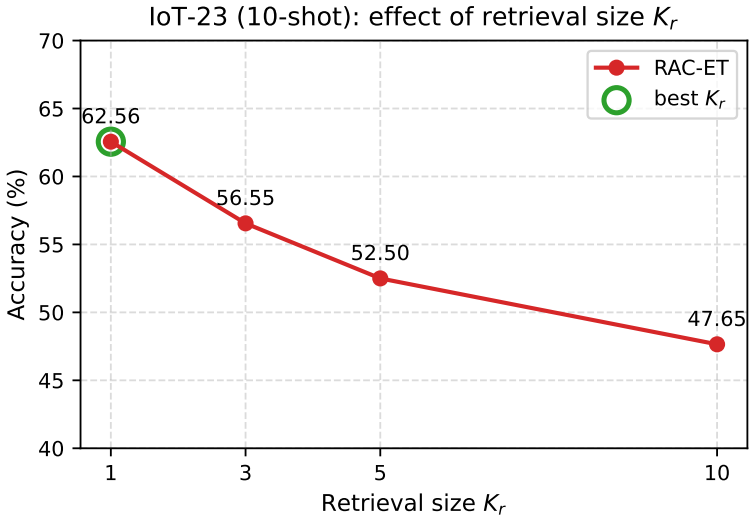

Figure 8: Effect of retrieval size K_r on accuracy for IoT-23 ($K_s = 10$). The green circle marks the optimal configuration $K_r = 1$; increasing K_r introduces cross-class or morphologically different neighbor samples, which inject noise during fusion.

4.4 Efficiency Analysis

Table 9 compares RAC-ET and ET-BERT full fine-tuning in terms of trainable parameters, training time, and peak GPU memory. RAC-ET freezes the ET-BERT backbone and trains only 5.07 M (approximately 3.69%) parameters, with significantly reduced training time and memory usage. During inference, FAISS `IndexFlatIP` provides linear-time exact nearest-neighbor retrieval, and combined with the default $K_r = 1$ setting, the overall inference overhead increases only slightly relative to ET-BERT, satisfying the engineering constraints of typical online detection scenarios.

Table 9: Efficiency comparison with ET-BERT full fine-tuning.

Metric	ET-BERT Full Fine-tune	RAC-ET (Ours)
Total Parameters	132.2M	137.3M
Trainable Parameters	132.2M (100%)	5.07M (3.69%)
CSTNET Training Time	∼2.5 h	∼12 min
Peak GPU Memory	∼16 GB	∼8 GB

4.5 CICIDS2017 In-Depth Analysis

To further understand the behavioral patterns of RAC-ET on CICIDS2017, Figure 9 presents the per-class F1 comparison at 10-shot, Figure 10 shows the confusion matrix of RAC-ET, and Figure 11 visualizes the feature distribution extracted by the self-supervised encoder via t-SNE.

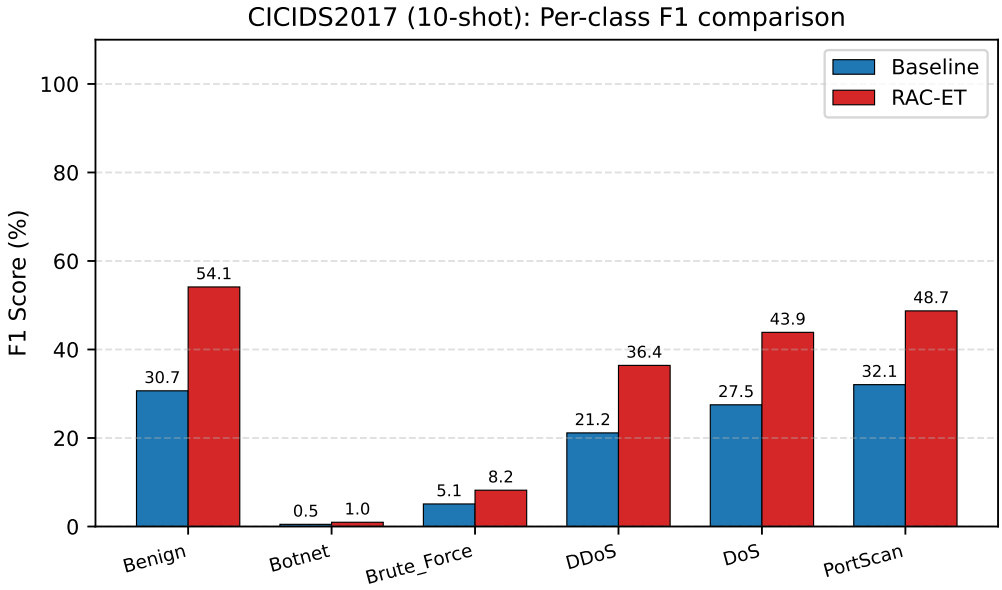

Figure 9: Per-class F1 comparison on CICIDS2017 (10-shot). RAC-ET achieves particularly significant F1 improvements on attack categories such as DDoS and PortScan, indicating that retrieval augmentation has stronger auxiliary discrimination for attack types with clear feature patterns.

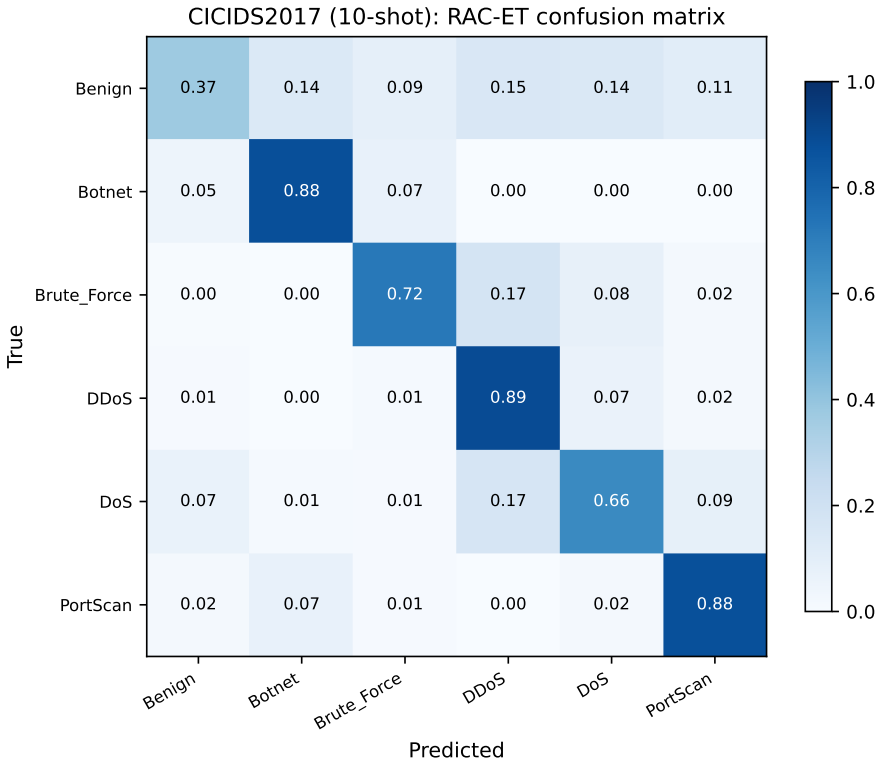

Figure 10: Normalized confusion matrix of RAC-ET on CICIDS2017 (10-shot). Higher diagonal values indicate more accurate classification. Good discrimination can be observed between Benign and major attack categories (DDoS, DoS, PortScan), with some confusion among certain attack subcategories.

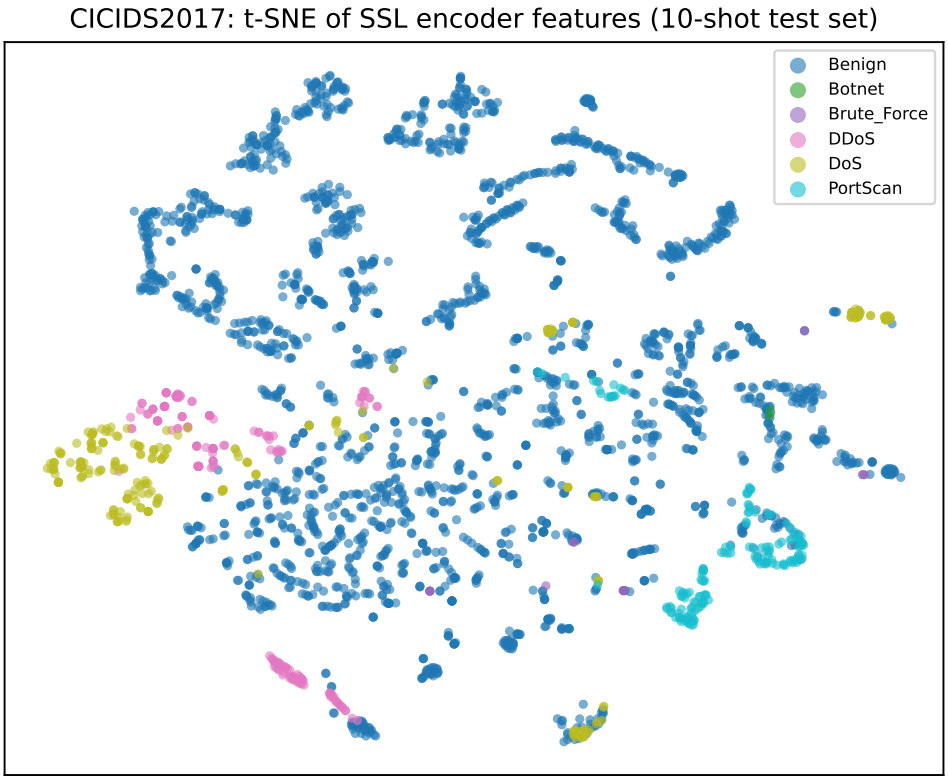

Figure 11: t-SNE visualization of CICIDS2017 self-supervised encoder features (test set subsample). Different colors represent different attack categories. The self-supervised pre-training has formed preliminary cluster structures, but some categories (e.g., DoS and DDoS) exhibit overlapping regions, which is precisely the space where RAC-ET provides additional discriminative cues through neighbor retrieval.

4.6 Discussion

When Is Retrieval Augmentation Most Effective? Synthesizing Table 3 and Table 8, two prerequisites emerge for RAC-ET to be effective: (i) the query sample must lie *close to same-class samples* in the semantic space so that Top-K_r retrieval is informative, and (ii) the fusion module must suppress the inevitable noise from nearest-neighbor retrieval. The gain gradient across the three datasets clearly supports this thesis: CSTNET (ET-BERT pre-trained on homologous data) $\rightarrow$ CICIDS2017 (flow features + self-supervised encoder) $\rightarrow$ IoT-23 (byte payload + cross-domain distribution shift). As the coverage of pre-trained representations over the target task decreases, RAC-ET's gains increase—*retrieval augmentation essentially compensates for the coverage gaps of pre-trained representations*. The $+11.22\%$ improvement at 10-shot on CICIDS2017 further confirms that this framework generalizes equally well to flow statistical feature inputs.

Relationship with Meta-Learning Methods Prototypical and matching networks treat support samples as references during the *training phase*. RAC-ET instead treats them as dynamically updatable non-parametric memory during the *inference phase*: when new classes or variants appear, new samples are simply written to the knowledge base—no retraining of the fusion module or backbone is required. This property makes RAC-ET particularly well suited to practical deployment scenarios where traffic classes evolve over time.

Dynamic Knowledge Base Updates and Zero-Cost Adaptation The architecture of RAC-ET confers a distinctive operational advantage: *knowledge base updates are entirely decoupled from model parameters*. Traditional supervised classifiers—including fine-tuned ET-BERT—require collecting sufficient labeled samples and re-fine-tuning some or all parameters when encountering new attack variants or emerging classes. RAC-ET, by contrast, only requires extracting a feature vector through the frozen encoder and appending it to the FAISS index—**no retraining of the fusion module or backbone is needed**. In practice, once a security operator confirms a new malicious traffic sample, it can be incorporated into the detection system in near real time; the entire process incurs only a single forward pass (~ 0.01 s) plus one vector index update. This "plug-and-play" knowledge expansion capability gives RAC-ET a significant operational efficiency advantage in real-world network environments where attack patterns evolve continuously.

Limitations Three limitations should be noted. First, the CICIDS2017 experiment uses flow statistical features rather than byte payloads, creating an input-modality gap with CSTNET and IoT-23; future work will unify all inputs to byte-level representations once complete raw PCAP data are available. Second, the current evaluation does not cover more stringent cross-domain transfer or class-incremental scenarios. Third, FAISS defaults to exact inner product search, which has room for optimization at million-scale knowledge base sizes; the performance–efficiency trade-off with approximate indices such as IVF and HNSW warrants dedicated investigation.

5 Conclusion

This paper addresses three key challenges in encrypted traffic classification—*scarce labeled samples, insufficient few-shot generalization of pre-trained models, and high online update costs*—by proposing RAC-ET,

a lightweight retrieval-augmented classification framework. On the method side, this paper uses a frozen pre-trained encoder as the feature extractor, constructs a FAISS-based traffic sample knowledge base, and fuses query features with Top-K_r neighbor features through cross-attention, enabling classification decisions to simultaneously leverage both parametric semantic representations and non-parametric sample memory. On the experimental side, the framework's effectiveness is systematically validated on three benchmark datasets: on CSTNET-TLS 1.3 full data, accuracy improves from 88.75% to 91.75% ($+3.00\%$), with consistent improvements of $+0.00\%$ to $+0.30\%$ under 1/5/10-shot settings; on IoT-23, the method achieves an average improvement of $+20.09\%$ under few-shot settings, with the highest gains of $+28.15\%$ at 3-shot, $+26.23\%$ at 10-shot, and $+23.73\%$ at 20-shot; on CICIDS2017, the framework's cross-modal generalization capability is validated using flow statistical features, achieving $+11.22\%$ improvement at 10-shot. On the efficiency side, RAC-ET trains only 3.69% of the parameters yet achieves accuracy superior to full fine-tuning, providing a viable path for encrypted traffic detection deployment in resource-constrained environments.

Future work will proceed along three directions: (1) unifying the CICIDS2017 input modality from flow statistical features to byte payloads once complete raw PCAP data become available, and extending the evaluation to cross-domain transfer and class-incremental scenarios; (2) exploring more efficient approximate nearest-neighbor indices and learnable retrievers to further reduce inference latency on large-scale knowledge bases; (3) investigating adversarial robustness and privacy-preserving mechanisms for retrieval-augmented classification to maintain stability in real-world adversarial environments.

References

[1] Mohammad Lotfollahi, Ramin Shirali Hossein Zade, Mojtaba Saberian, and Mohammad Javad Siavoshani. Deep packet: A novel approach for encrypted traffic classification using deep learning. *Soft Computing*, 24:1999–2012, 2020.

[2] Shahbaz Rezaei and Xin Liu. Deep learning for encrypted traffic classification: An overview. *IEEE Communications Magazine*, 57(5):76–81, 2019.

[3] Xinjie Lin et al. Et-bert: A contextualized datagram representation with pre-training transformers for encrypted traffic classification. *arXiv preprint arXiv:2202.06335*, 2022.

[4] Imperva Threat Research. 2025 imperva bad bot report. Imperva Report, April 2025. Published April 15, 2025.

[5] Federal Bureau of Investigation. Internet crime report 2024. FBI IC3 Annual Report, April 2025. Published April 23, 2025.

[6] Blake Anderson, Subharthi Paul, David McGrew, and Mustaque Ahamad. Deciphering malware's use of TLS (without decryption). In *Proceedings of the 2018 ACM Internet Measurement Conference (IMC)*, 2018.

[7] Wei Wang, Ming Zhu, Xuewen Zeng, Xiaozhou Ye, and Yiqiang Sheng. Malware traffic classification using convolutional neural network for representation learning. *2017 International Conference on Information Networking (ICOIN)*, 2017.

[8] Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A. Ghorbani. Characterization of encrypted and VPN traffic using time-related features. In *Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP)*, 2016.

[9] Tal Shapira and Yuval Shavitt. Flowpic: Encrypted internet traffic classification is as easy as image recognition. In *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 680–687, 2019.

[10] Chang Liu, Li He, Guoliang Xiong, Zhiliang Cao, and Zhen Li. Fs-net: A flow sequence network for encrypted traffic classification. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1171–1179, 2019.

[11] Xiao Yu, Heng Guo, Yu Qiao, Xiang Wang, Zhen Cao, Qi Wan, Chuan Zhou, Jinxin Zhang, Xianghui Xu, and Jintao Yang. Bert-packet-header: Advanced encrypted traffic classification with contextual information from packet headers. *Applied Intelligence*, 54:124606, 2024.

[12] Siwei Ma, Xiao Wang, Zijian Wang, Xinjie Lin, Jingxian Ren, Lei Wang, Qiben Zhu, Qiang Xu, Kui Hou, Yisong Xue, Yang Yu, Xian Zhang, Hua Lu, Jianyuan Lu, Kuansan Wang, et al. Metarock-etc: Few-shot encrypted traffic classification via fine-grained learning and generalized representation. *Computer Networks*, 251:110581, 2024.

[13] Anjali Sharma, Bhupendra Verma, and M. Tulasi Raman. A survey of encrypted traffic classification in the era of ai and quantum computers. *Computer Networks*, 257:110973, 2025.

[14] Can Ye, Heng Guo, Yichao Ma, Jinxin Zhang, Jiajia Gao, Zhen Cao, Yutong Zhai, Xiang Wang, and Fengtong Xie. Encrypted application traffic classification with auxiliary pretraining and parameter-efficient fine-tuning. *Scientific Reports*, 15:21238, 2025.

[15] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[16] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, and Daan Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

[18] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

[19] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Lu Wang, Weizhu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[20] Bihan Yang, Hao Song, Huifang Wu, Jianqiang Wei, and Mingzhe Wang. Metamre: Meta-learning and representation enhancement for few-shot encrypted traffic classification. In *Advances in Networked-Based Information Systems, AINA 2023*, pages 476–487, 2023.

[21] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich K”uttler, Mike Lewis, Wen-tau Yih, Tim Rockt”aschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[22] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, 2020.

[23] Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[24] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. *International Conference on Learning Representations (ICLR)*, 2020.

[25] Guoxin Yu, Lemao Liu, Haiyun Jiang, Shuming Shi, and Xiang Ao. Retrieval-augmented few-shot text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6721–6735, 2023.

[26] Fengnan Li, Elliot D. Hill, Jiang Shu, Jiaxin Gao, and Matthew M. Engelhard. IRIS: Interpretable retrieval-augmented classification for long interspersed document sequences. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30263–30283, 2025.

[27] Zhenyu Xu et al. A survey on network intrusion detection using deep learning: Current trends, challenges, and future directions. *ACM Computing Surveys*, 56(10):1–38, 2024.

[28] Jeff Johnson, Matthijs Douze, and Herv'e J'egou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[30] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)*, pages 108–116, 2018.