采用序列特征与知识引导的未知网络应用早期识别方法

刘祎彤[1,2]，王平辉[1,2]，赵俊舟[1,2]

（1. 西安交通大学自动化科学与工程学院，710049，西安；
2. 西安交通大学智能网络与网络安全教育部重点实验室，710049，西安）

摘要：针对现有网络流量分类方法难以在样本稀缺场景下快速识别早期未知应用这一问题，提出一种基于序列特征与知识引导的未知网络应用早期识别方法。一方面基于Transformer-encoder框架构建流量分类模型(FAIN)，该模型利用自注意力机制挖掘流量序列中数据包之间的全局依赖关系，生成具有可分辨性的流量表示向量用于分类。另一方面，为提升FAIN模型在样本稀缺场景下的适应能力，采取监督预训练与元学习相耦合的模型优化策略，赋予模型在小样本场景下快速学习流量分类任务的能力，使其满足识别早期未知网络应用的需求。在公开数据集与真实校园网流量合成的小样本数据集上进行了深入的对比实验。结果表明：所提出的流量分类模型FAIN在公开分类任务上优于现有方法，且优化后的FAIN模型在XJTU-FSTC和CSTNET数据集的5类和10类小样本分类任务上，准确率最高分别提升了16.75%、10.08%和11.57%、8.24%。该研究结果为未知网络应用的早期识别提供了有效的方法支撑。

关键词：网络应用识别；注意力机制；预训练；小样本学习
中图分类号：TP393.0 **文献标志码**：A
DOI：10.7652/xjtuxb202311018 **文章编号**：0253-987X(2023)11-0181-13

Early Identification of Unknown Applications Based on Sequence Features and Knowledge Guidance

LIU Yitong[1,2], WANG Pinghui[1,2], ZHAO Junzhou[1,2]

(1. School of Automation and Engineering, Xi'an Jiaotong University, Xi'an 710049, China; 2. Ministry of Education Key Lab for Intelligent Networks and Network Security, Xi'an Jiaotong University, Xi'an 710049, China)

Abstract: To address the problem that the existing network traffic classification methods are not able to promptly identify early unknown applications in case of data scarcity, a method for the early identification of unknown applications based on sequence features and knowledge guidance is proposed in this paper. On the one hand, a traffic classification model, named FAIN, is constructed using the Transformer-encoder framework. By leveraging the self-attention mechanism, FAIN model effectively captures global dependencies among packets and generates distinguishable representation for classification. On the other hand, to improve the adaptability of FAIN model to data scarcity, a model optimization strategy that couples supervised pre-training with meta-learning is proposed. This strategy empowers the model to quickly learn unseen tasks in a few-shot setting.

收稿日期：2023-02-02。 作者简介：刘祎彤(1994—)，男，博士生；王平辉(通信作者)，男，教授，博士生导师。 基金项目：国家自然科学基金资助项目(61922067)；教育部-中国移动"人工智能"资助项目(MCM20190701)。
网络出版时间：2023-06-01 网络出版地址：https://kns.cnki.net/kcms2/detail/61.1069.T.20230531.1552.002.html

In this study, in-depth comparative experiments are conducted on few-shot datasets synthesized from real campus network traffic. The results show that the proposed FAIN traffic classification model is superior to existing methods in terms of public network traffic classification. The optimized FAIN model improves the accuracy on the 5-class and 10-class few-shot classification tasks of the XJTU-FSTC and CSTNET datasets, with the maximum accuracy increases of 16.75%, 10.08% and 11.57%, 8.24%, respectively. This model provides effective support for the early identification of unknown applications.

Keywords: network traffic classification; self-attention mechanism; pre-training; few-shot learning

计算机网络作为互联网时代的基础设施,是现代社会生活中不可或缺的组成部分。随着网络通信技术的快速发展,各类新型网络协议和应用层出不穷,如 TLS 加密协议[1]、5G 通信[2]、云服务[3]等。与此同时,互联网用户对高通量网络传输的需求不断增长,Cisco 发布的 2021 年度互联网报告[4]显示,预计到 2023 年,全球固定宽带速度将是 2018 年的 2.4 倍。面对日益复杂的网络生态环境和庞大的实时流量数据,运营商亟需提升自身的管理水平和运维能力以保障网络空间的安全与稳定。网络流量分类[5]作为先进网络管理的前提,旨在从不同应用和协议的混合流量中准确识别每条流所属的应用类型,进而帮助运营商提升用户服务质量[6],实现差异化网络定价,提供精细化服务[7]以及诊断分析网络异常[8]等。

近年来,随着人工智能技术在自然语言处理[9-10]和计算机视觉[11-12]等领域取得成功,国内外研究人员也开始利用深度学习模型来解决流量分类任务[13-19]。迄今为止,基于深度学习的流量分类方法已经在绝大多数公开数据集上实现了最优结果,这很大程度上归功于数据集中包含的大量标注样本。但是,随着网络监管政策和数据隐私保护的加强,流量采集和标注工作变得越发困难,导致难以生成高质量的数据集用于模型训练。此外,由于实际网络环境动态变化,各种类型的未知应用不断涌现,如果能在未知应用出现早期对其进行快速有效识别,不仅能为应用提供合适的带宽和时延保障,而且能及时检测恶意应用,避免用户信息泄露和财产损失。然而,未知应用在早期阶段产生的流量规模小,难以在短时间内获取充足的样本用于已有分类器的微调或重新训练,而现有方法很大程度上忽略了这一问题,因此对未知应用的早期识别效果不佳。综上,尽管目前基于深度学习的方法在公开流量分类任务上成果斐然[20-23],但在实际中,尤其是在未知应用早期识别场景下,经常会面临样本稀缺的问题。众所周知,深度神经网络参数庞大,训练样本的质量和数量对模型性能起着决定性作用,样本稀缺会导致模型过拟合,进而降低模型的泛化能力,影响模型的鲁棒性和适用性[24-27]。

针对上述问题,本文提出了一种基于序列特征与知识引导的未知网络应用早期识别方法。该方法首先基于 Transformer-encoder 框架构建流量分类模型(FAIN),利用自注意力机制挖掘流量序列中数据包之间的全局依赖关系,提取流量序列的高维隐式特征,进而将其编码为具有可分辨性的表示向量用于分类。在之后的训练阶段,为提升 FAIN 模型在样本稀缺场景下的适应能力,本文采取监督预训练与元学习相耦合的模型优化策略。首先,利用大规模已知网络应用的流量样本,按照监督学习范式对模型进行预训练。其次,构建小样本流量分类任务,并将预训练后的模型参数作为元学习过程的初始化参数,利用预训练参数中蕴含的有关流量时序特征的通用领域知识,引导元学习过程中梯度更新方向。与此同时,为降低元学习过程中求解高阶梯度带来的高昂计算成本,采用一阶梯度更新算法 Reptile[25],对模型参数进行多轮更新。优化后的 FAIN 模型具备了在样本稀缺场景下快速学习流量分类任务的能力,能够满足准确识别早期未知网络应用的需求。

本文主要贡献如下:

(1)设计了一种基于 Transformer-encoder 框架的流量分类模型 FAIN,利用自注意力机制挖掘流量序列中数据包之间的全局依赖关系,进而构建具有可分辨性的流量表示向量用于分类;

(2)提出了一种监督预训练与一阶梯度元学习相耦合的模型优化策略,用于提升流量分类模型 FAIN 在小样本场景下的快速适应能力,使得模型满足准确识别早期未知应用的需求;

(3)在公开数据集 ISCXTor 和 ISCXVPN 以及由真实校园网流量合成的小样本数据集 XJTU-FSTC 上进行了对比实验,结果表明,FAIN 模型在

公开数据集 3 个分类任务上的综合表现优于对比方法，性能指标 F_1 值（模型精确率和召回率的调和平均值）分别为 93.17%、92.93%和91.10%；优化后的 FAIN 模型在 XJTU-FSTC 和 CSTNET 数据集的 5 类和 10 类小样本分类任务上，准确率显著提升。

1　相关工作

基于深度学习的流量分类方法的主要特点在于，将特征提取和流量分类两方面工作整合到统一的端到端深度神经网络之中，能够自动从原始流量中提取高维隐式表示向量，降低了特征构建过程中对专家经验和领域知识的依赖，模型具有良好的泛化能力，因此更加适合真实网络环境。

Lotfollahi 等利用流量序列中数据包负载中的字节序列构建特征向量，提出了基于堆叠自动编码器（SAE）和一维卷积神经网络（1D-CNN）的流量分类模型 Deep Packet，该方法从数据包层面对流量进行分类，在 ISCXVPN 公开数据集上的分类性能相较于机器学习类方法有明显提升[13]。Li 等提出 BSNN 流量分类方法，该方法将数据包负载中的字节序列分割成多个片段，基于 LSTM 和 GRU 循环神经网络构建编码器提取片段内和不同片段之间时序特征，同样实现了数据包层面的流量分类[15]。Liu 等考虑流量中数据包之间的时序特征，提出了流层面的流量分类模型（FS-Net）。该模型采用数据包包长序列作为输入，基于双向门控单元（Bi-GRU）构建流量特征提取器，并引入重构损失保证编码向量的有效性[14]。Xiao 等对 BSNN 模型进行拓展，提出了可适用数据包层面和流层面的流量分类模型（EBSNN）。具体来讲，该方法在 BSNN 模型之上增加了包信息聚合模块，进一步提升了模型的性能和可解释性[17]。

2　未知网络应用早期识别方法

本节首先给出网络流量分类和未知网络应用早期识别问题的公式化定义，其次介绍基于 Transformer-encoder 框架的流量分类模型，最后阐述基于监督预训练和梯度元学习的模型优化策略。

2.1　问题定义

2.1.1　网络流量分类

网络流是指原始混合流量中具有相同五元组（源 IP 地址、源端口、目的 IP 地址、目的端口及传输层协议）的一组数据包序列。网络流量分类旨在从包含不同应用和协议的混合流量中准确识别每条流所属的应用类型。给定训练数据集 D，集合 $Y=\{1, 2, \cdots, y\}$ 包含 y 类网络应用标签。数据集 D 中第 i 个样本可以表示为 $X_i=\{p_1^i, p_2^i, \cdots, p_n^i\}$，其中 n 表示序列长度（即包含 n 个数据包），$p_t^i \in \mathbf{R}^d$ 表示第 t 个数据包的 d 维初始特征，样本的真实标签 $y_i \in Y$。网络流量分类的目标是构建一个端到端的深度神经网络模型 f_θ 用于预测流量样本 X_i 所属的应用类别，即 $f_\theta(X_i) \rightarrow \hat{y}_i$，并使得预测标签 $\hat{y}_i$ 与样本真实标签 y_i 相匹配。

2.1.2　未知网络应用早期识别

本文将未知应用早期识别问题抽象为元学习范式下的小样本流量分类任务，通过小样本学习赋予模型 f_θ 快速适应样本稀缺场景的能力，进而满足准确识别早期未知网络应用的需求。一般来讲，元学习将任务 T 视作学习单元，每个任务由支持集 S 和查询集 Q 构成，支持集 S 通常设定为 N 类 K 样本形式，即包含 N 类标签和 $N \times K$ 个训练样本。在本文中，训练任务 $\mathcal{T}_{\text{train}}$ 和目标任务 $\mathcal{T}_{\text{target}}$ 从给定的任务分布 $p(\mathcal{T})$ 中采样得到，其中 $\mathcal{T}_{\text{train}}$ 中的样本标签均属于已知应用集合 C_{known}，$\mathcal{T}_{\text{target}}$ 中的样本标签均属于未知应用集合 C_{unknown}，且 $C_{\text{known}} \cap C_{\text{unknown}}=\varnothing$。在训练阶段，从分布 $p(\mathcal{T})$ 中不断采样训练任务，对模型参数 θ 进行更新直至收敛，收敛后的模型参数 θ 对流量分类任务高度敏感，具备小样本场景下快速适应新任务的能力。在测试阶段，模型 f_θ 在测试任务 $\mathcal{T}_{\text{target}}$ 上仅用少数几步更新，就能达到良好的分类准确率。

2.2　流量分类模型—FAIN

2.2.1　Transformer 模型

Transformer 作为当下应用最为广泛的序列表示学习模型，已经在自然语言处理和计算机视觉领域取得了瞩目的成就。Transformer 是基于自注意力机制构建的编码-解码架构，设计初衷是为了更好地解决机器翻译任务。相较于基于循环神经网络建模方法，Transformer 通过自注意力机制挖掘序列中不同时刻特征之间的全局依赖关系，有效避免了模型在训练过程中存在的梯度爆炸和梯度消失问题，而且能够并行进行注意力计算，极大程度上缩短了模型训练和推理所需的时间。

（1）编码-解码架构。Transformer 架构中的编码和解码模块均由 6 个结构相同的子模块堆叠而成，每个子模块由多头注意力层、残差归一化层和前向传播层 3 部分构成。使用 Transformer 进行序列建模时首先给定一组输入序列 $\boldsymbol{X}=[x_1, x_2, \cdots,$

$x_n]\in\mathbf{R}^{n\times d_{\text{in}}}$，使用编码模块对 $\boldsymbol{X}$ 进行特征编码，得到高维隐式嵌入表示 $\boldsymbol{Z}=[z_1, z_2, \cdots, z_n]\in\mathbf{R}^{n\times d_{\text{model}}}$，之后 $\boldsymbol{Z}$ 作为解码模块的输入，经过带有掩码操作的自回归运算得到输出序列 $\boldsymbol{Y}=[y_1, y_2, \cdots, y_m]\in\mathbf{R}^{n\times d_{\text{out}}}$。

(2)自注意力机制。Transformer 的核心是自注意力机制。数学上，自注意力机制是通过矩阵点积操作来进行向量相似度计算。给定矩阵 $\boldsymbol{Q}\in\mathbf{R}^{n\times d_k}$、$\boldsymbol{K}\in\mathbf{R}^{n\times d_k}$、$\boldsymbol{V}\in\mathbf{R}^{n\times d_k}$ 分别表示 queries、keys 和 values 向量，在注意力计算过程中，首先将矩阵 $\boldsymbol{Q}$ 和矩阵 $\boldsymbol{K}$ 进行点积并将结果归一化，之后再点积矩阵 $\boldsymbol{V}$ 得到输出结果

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V})=\text{Softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{1}$$

$$\text{MultiHead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V})=\text{Concat}(\text{head}_1, \text{head}_2, \cdots, \text{head}_h)\boldsymbol{W}^{\text{o}} \tag{2}$$

$$\text{head}_i=\text{Attention}(\boldsymbol{Q}\boldsymbol{W}_i^{\boldsymbol{Q}}, \boldsymbol{K}\boldsymbol{W}_i^{\boldsymbol{K}}, \boldsymbol{V}\boldsymbol{W}_i^{\boldsymbol{V}}) \tag{3}$$

式中：$\boldsymbol{W}_i^{\boldsymbol{Q}}\in\mathbf{R}^{d_{\text{in}}\times\frac{d_k}{h}}$；$\boldsymbol{W}_i^{\boldsymbol{K}}\in\mathbf{R}^{d_{\text{in}}\times\frac{d_k}{h}}$；$\boldsymbol{W}_i^{\boldsymbol{V}}\in\mathbf{R}^{d_{\text{in}}\times\frac{d_k}{h}}$；$\boldsymbol{W}^{\text{o}}\in\mathbf{R}^{d_k\times d_{\text{model}}}$，均表示映射矩阵；Contact(·)表示拼接操作；h 表示注意力头数。

2.2.2 流量表示学习模块

类似于 Bert[10]，FAIN 模型摒弃了 Transformer 架构中的解码模块，采用完全堆叠编码层的方式构建流量表示学习模块，这一做法主要出于两方面考虑：①Transformer 中的解码模块主要适用于长度不定的序列生成任务，如机器翻译、对话生成等，且在注意力计算过程中需要对序列进行掩码操作，因此并不适用于分类任务；②不使用解码模块能够减少大约一半的模型参数，可以大大缩短模型训练和推理所需时间。

如图 1 所示，流量表示学习模块中的编码层包含多头注意力层、残差归一化层以及前向传播层 3 个部分。值得注意的是，本文参考文献[26]在残差归一化层中将 Transformer 中使用的层归一化替换成批归一化，目的是减轻序列中存在的异常值的影响，同时在位置编码阶段采用可学习编码替换 Transformer 的正弦位置编码方式。给定一条包含 n 个数据包的网络流，表示为矩阵 $\boldsymbol{X}=[p_1, p_2, \cdots, p_n]\in\mathbf{R}^{n\times m}$，其中 m 表示每个数据包的原始特征维度(包长、间隔到达时间等)，流量表示学习模块将从矩阵 $\boldsymbol{X}$ 中学习网络流的时序特征表示。

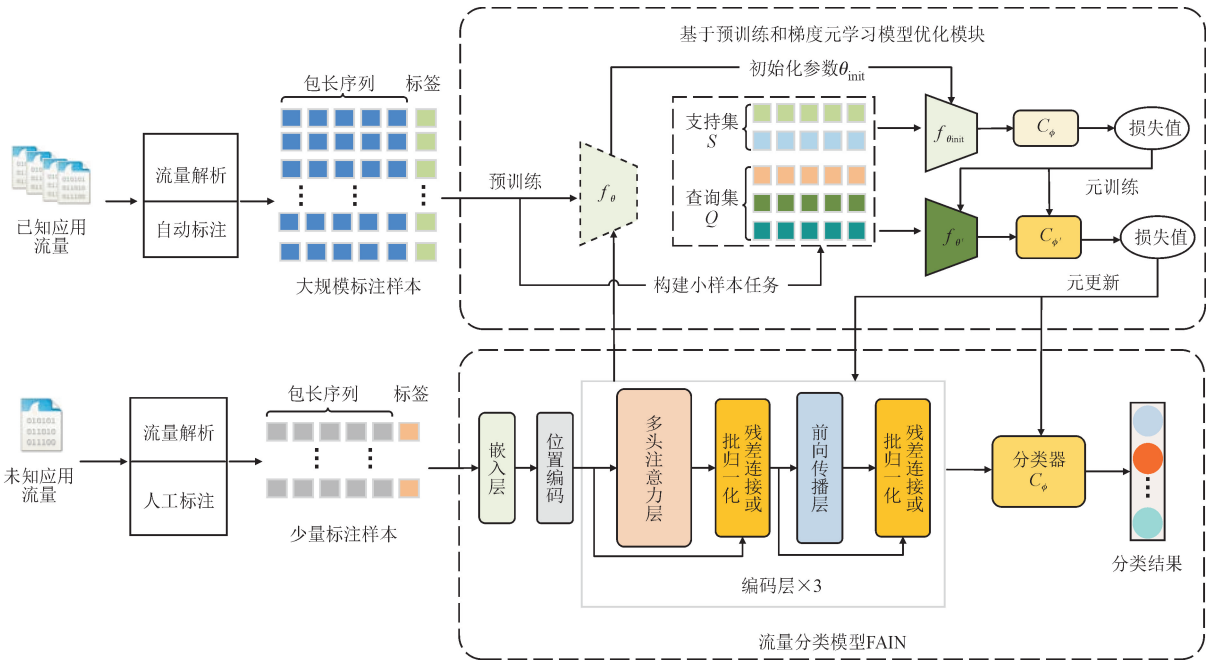


图 1 本文方法整体框架图

Fig. 1 Overall framework of the proposed method

具体地，对矩阵 $\boldsymbol{X}$ 编码时，首先将每个时刻的特征 $\boldsymbol{p}_i$ 经过线性层 $\boldsymbol{W}_{\text{proj}}$ 映射到 d 维嵌入空间中，添加位置编码后，得到每个数据包的初始表示向量 $\boldsymbol{e}_i$，可写为

$$\boldsymbol{e}_i=(\boldsymbol{W}_{\text{proj}}\boldsymbol{p}_i+\boldsymbol{b}_{\text{proj}})+\boldsymbol{e}_i^{\text{pos}} \tag{4}$$

式中：$\boldsymbol{W}_{\text{proj}}\in\mathbf{R}^{d\times m}$、$\boldsymbol{b}_{\text{proj}}\in\mathbf{R}^d$ 分别表示线性层参数和偏置向量；$\boldsymbol{e}_i^{\text{pos}}\in\mathbf{R}^d$ 表示位置编码向量。之后将得到的嵌入矩阵 $\boldsymbol{E}_0=[\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_n]\in\mathbf{R}^{n\times d}$ 输入到 FAIN

模型中进行逐层编码，每一层计算过程为

$$\boldsymbol{H}_l = \mathrm{MultiHead}(\boldsymbol{Q}_l, \boldsymbol{K}_l, \boldsymbol{V}_l) \tag{5}$$

$$\boldsymbol{E}_l = \mathrm{BatchNorm}(\mathrm{FFN}(\boldsymbol{E}_{l-1} + \boldsymbol{H}_l)) \tag{6}$$

式中：$\boldsymbol{E}_l$，$\boldsymbol{H}_l \in \mathbf{R}^{n \times d_{\mathrm{model}}}$；$l$ 表示层数；FFN(·)表示前向传播；BatchNorm(·)表示归一化。将最后一层的输出 $\boldsymbol{E}_l$ 经线性层映射后得到表示向量 $\boldsymbol{Z}$

$$\boldsymbol{Z} = W_{\mathrm{out}}\boldsymbol{E}_l + \boldsymbol{b}_{\mathrm{out}} \tag{7}$$

式中：$\boldsymbol{Z} \in \mathbf{R}^{n \times d_{\mathrm{out}}}$；$\boldsymbol{W}_{\mathrm{out}} \in \mathbf{R}^{n \times d_{\mathrm{out}}}$；$\boldsymbol{b}_{\mathrm{out}} \in \mathbf{R}^{d_{\mathrm{out}}}$，均为线性层参数。

2.2.3　流量分类模块和损失函数

在分类阶段，采用双层前馈神经网络构建分类器，将流量表示学习模块编码得到的表示向量 $\boldsymbol{Z}$ 作为输入，输出该条流所属网络应用的预测结果 $\hat{\boldsymbol{y}}$

$$\hat{\boldsymbol{y}} = \mathrm{Softmax}(\mathrm{Mean}(\mathrm{ReLU}(\boldsymbol{W}_1 Z + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2)) \tag{8}$$

式中：$\boldsymbol{W}_1$、$\boldsymbol{W}_2$、$\boldsymbol{b}_1$、$\boldsymbol{b}_2$ 为线性层参数；ReLU(·)表示激活函数；Mean(·)表示平均操作。本文采用多分类问题常用的交叉熵损失作为损失函数

$$\mathcal{L} = -\frac{1}{N}\sum_{i}^{N}\sum_{c=1}^{C} I(\hat{y}_{ic} = y_i)\ln q(\hat{y}_{ic}) \tag{9}$$

式中：N 为流量样本数量；C 为应用标签数目；$I(\cdot) \in \{0, 1\}$ 为指示函数，当预测结果 $\hat{y}_{ic}$ 和真实标签 y_i 一致时，取值为 1；$q(\cdot)$ 为预测标签的概率。

2.3　基于预训练和梯度元学习的模型优化策略

2.3.1　MAML 算法

MAML (model-agnostic meta-learning)[27] 作为一种基于梯度更新的元学习算法，其学习过程主要包含元训练和元更新两个阶段。给定模型 f_θ 和任务分布 $p(\mathcal{T})$，MAML 算法的优化目标为

$$\min_{\theta}\sum_{\mathcal{T}_i \sim p(\mathcal{T})}\mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'}) = \sum_{\mathcal{T}_i \sim p(\mathcal{T})}\mathcal{L}_{\mathcal{T}_i}(f_{\theta - \alpha\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)}) \tag{10}$$

式中：α 表示元训练阶段的学习率；$\mathcal{L}_{\mathcal{T}_i}(\cdot)$ 表示任务 i 的损失值。在每一轮训练过程中，首先从任务分布 $p(\mathcal{T})$ 中随机采样 M 个训练任务 $\{\mathcal{T}_{\mathrm{train}}^1, \mathcal{T}_{\mathrm{train}}^2, \cdots, \mathcal{T}_{\mathrm{train}}^M\}$；然后，在元训练阶段，每个训练任务 $\mathcal{T}_{\mathrm{train}}^i$ 独立地对模型参数 θ 进行 k 步梯度更新

$$\theta_i^{(k)} = \theta_i^{(k-1)} - \alpha\nabla_{\theta_i^{(k-1)}}\mathcal{L}_{\mathcal{T}_i}(f_{\theta_i^{(k-1)}}) \tag{11}$$

在元更新阶段，聚合 M 个训练任务的损失梯度用于更新模型参数 θ

$$\theta_{i+1} = \theta_i - \beta\nabla_\theta\sum_{i=1}^{M}\mathcal{L}_{\mathcal{T}_i}(\theta_i^{(k)}) \tag{12}$$

式中 β 表示元更新阶段的学习率。

为了减少元学习过程中的梯度更新带来的高昂计算成本，文献[27]同时提出了 MAML 的一阶近似算法 FOMAML。FOMAML 省略了式(10)中的二阶导数的求解过程，用 θ_i' 作为元梯度的更新方向。实验证明，采用 FOMAML 算法不但提升了元学习效率，而且能够保证模型性能不发生明显下降。

2.3.2　基于预训练和梯度元学习的模型优化策略

由式(9)和(10)可知，MAML 算法对参数 θ 的每轮更新都需要求解其二阶甚至更高阶导数，这将使得训练计算成本过大，且训练过程易受超参数和异常样本的影响，使得训练不稳定。除此之外，FAIN 模型中基于 Transformer 框架构建的特征提取模块参数量庞大，在小样本学习过程中可能出现过拟合，将严重影响模型的泛化能力，导致训练失败。而且，直接对特征提取模块参数进行随机初始化，一方面会导致训练时间过长，另一方面由于样本数量不足，训练过程中模型可能收敛到局部最优，导致无法获得最优性能。

为此，本文提出监督预训练和一阶梯度元学习相结合的模型优化策略，如图 1(b)所示。具体地，首先使用大量已知网络应用样本对 FAIN 模型按照有监督的方式进行预训练，之后去除分类层参数，将特征提取模块的预训练参数 $\theta_{\mathrm{pretrained}}$ 作为元学习过程的初始参数，目的在于利用预训练参数中蕴含的有关流量时序特征的通用领域知识，引导元学习过程中参数的梯度更新方向，缓解随机初始化参数可能造成的模型收敛到局部最优问题。与此同时，为进一步提高训练效率，元学习过程采用一阶梯度算法 Reptile 用以降低求解高阶梯度带来的高昂计算成本，缩短模型训练所需时间。模型参数为

$$\theta_{i+1} = \theta_i + \mu\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}(\mathcal{L}_{\mathcal{T}_i}(\theta_i^{(k)}) - \theta_i) \tag{13}$$

式中：μ 表示学习率；N 表示训练轮数。

算法 1　基于监督预训练和 Reptile 的参数优化算法

输入：模型 f_θ，任务分布 $p(\mathcal{T})$，预训练参数 $\theta_{\mathrm{pretrained}}$，学习率 α 及 μ，梯度更新步数 k，训练轮次 N

输出：优化后模型参数 θ_{opt}

初始化模型参数：$\theta_0 \leftarrow \theta_{\mathrm{pretrained}}$
while $i \leqslant N$ do
　# 元训练阶段
　采样批训练任务 $\{\mathcal{T}_{\mathrm{train}}^1, \mathcal{T}_{\mathrm{train}}^2, \cdots, \mathcal{T}_{\mathrm{train}}^M\} \sim p(\mathcal{T})$
　for j in range(M)：
　　使用 $\mathcal{T}_{\mathrm{train}}^j$ 中的支持集 S_j，按式(9)计算任务损失值；
　　根据式(11)对参数 θ_i 进行 k 步更新；
　　使用 $\mathcal{T}_{\mathrm{train}}^j$ 中的查询集 Q_j 计算梯度 $\nabla_{\theta_i}\mathcal{L}_{\mathcal{T}_{\mathrm{train}}^j}(\theta_i^{(k)})$；

9 end for
10 # 元更新阶段
11 收集 $\{\mathcal{T}_{\text{train}}^{1}, \mathcal{T}_{\text{train}}^{2}, \cdots, \mathcal{T}_{\text{train}}^{M}\}$ 任务梯度值，按式(13)更新模型参数 θ_i；
12 $i=i+1$
13 end while
14 return θ_{opt}
15 采样测试任务 $\mathcal{T}_{\text{test}} \sim p(\mathcal{T})$
16 使用优化后的参数 θ_{opt} 计算分类结果

3 实验与结果分析

3.1 实验设置

3.1.1 实验环境

本文所有实验使用的服务器配置了 2 块 Intel(R) Xeon(R) E5-2690 V4 2.60 Hz CPU 处理器以及 4 块 NVIDIA Tesla V100 32 GB GPU 处理器，操作系统为 Ubuntu 16.04，代码环境为 Python 3.8。

3.1.2 数据集与数据预处理

基于 2 个公开网络流量分类数据集 ISCXTor2016[28] 和 ISCXVPN2016[29]，以及由真实校园网流量合成的小样本数据集 XJTU-FSTC[30] 和公开数据集 CSTNET-TLS 1.3[31] 合成的小样本数据集开展相关实验。

(1)ISCXTor2016 数据集从 18 种具有代表性的网络应用(Facebook、Spotify、Gmail 等)中，收集了包含邮件、聊天、音频、文件传输等 8 种常用网络服务的流量，并且根据流量是否通过 Tor 浏览器代理，将采集到流量分为 Tor 和 Non-Tor 两类，数据集共计 16 类标签。

(2)ISCXVPN2016 数据集从 Facebook、Skype、Spotify、YouTube 等 16 种网络应用中采集了包含聊天、邮件、文件传输等 6 种常用网络服务的流量，并且根据流量是否通过 VPN 隧道代理，将每种服务的流量分为 VPN 和 Non-VPN 两类，数据集包含 Application classification 和 Service classification 两个分类任务，分别包含 16 类标签和 12 类标签。

(3)XJTU-FSTC 小样本数据集根据实验室收集的真实校园网流量合成，包含 BitTorrent、YOUKU、QQLive、TikTok 等 64 种网络应用产生的流量，经过滤、合并、标注等一系列预处理工作后，每类应用包含 50 个流量样本，每个样本由应用会话的前 256 个数据包构成。选取其中 50 类应用用于构建训练任务，剩余 14 类应用构建测试任务。

(4)CSTNET 小样本数据集由公开数据集 CSTNET-TLS 1.3 合成得到，数据集包含 120 类应用标签，每类标签包含 50 个流量样本，每个样本由应用会话前 30 个数据包构成。选取其中 96 类应用用于构建训练任务，剩余 24 类应用用于构建测试任务。

数据集详细的统计信息如表 1 所示，在预处理过程中，按照相同五元组从原始 pcap 文件中筛选出网络流并保留每条流的前 N 个数据包，移除数据包中链路层包头、传输层包头中源 IP 地址、目的 IP 地址、端口号等，得到网络流序列样本并标注。提取序列中数据包包长、TTL、时间窗口大小、Flags、包间隔到达时间等作为候选特征集合。

表 1 实验所用数据集统计信息

Table 1 Statistics of the datasets

数据集	流数量	包数量	标签数
ISCXTor2016	14 505	80 000	16
ISCXVPN2016	6 023	137 163	12(Service), 16(Application)
XJTU-FSTC	3 200	598 400	64
CSTNET	6 000	180 000	120

3.1.3 实验细节

采用 Pytorch 深度学习框架实现实验代码，使用 Scapy 和 nfstream 工具库解析原始 pcap 文件，得到序列长度为 30 的数据包包长序列。在 ISCXTor2016 和 ISCXVPN2016 公开数据集上按照 6∶2∶2 的比例划分训练集、验证集和测试集。在 XJTU-FSTC 数据集上按照 8∶2 的比例划分已知应用和未知应用，并构建 5 类 K 样本和 10 类 K 样本小样本学习任务。设置 FAIN 模型的编码层数为 3，自注意力模块头数为 8，初始嵌入维度为 128，采用 ReLU 激活函数。在预训练阶段，采集大规模已知应用流量样本对模型进行预训练，采用 Radam 优化器，学习率设为 10^{-3}，最大训练轮数为 200，批样本数量为 32，训练过程中采用早停机制防止过拟合。在小样本学习阶段，从训练集中随机采样 5 000 个训练任务进行训练，元训练阶段采用 SGD 优化器，学习率 10^{-4}，迭代步数为 5，元更新阶段采用Adam优化器，学习率为 10^{-5}。

3.2 基线方法与评价指标

3.2.1 基线方法

为验证所提方法的有效性，本文选取了 6 个网络流量分类方法作为基线进行对比试验，包括 packet-level 方法：BSNN 和 DeepPacket，以及 flow-level 方法：AppScanner、AWF、DeepFinger 和 FSNet。

(1) DeepPacket[13]。利用数据包负载字节信息，基于堆叠自动编码器(SAE)和一维卷积神经网络(1D-CNN)构建的流量分类模型。

(2)BSNN[15]。将数据包负载中的字节序列分割成多个片段,基于 LSTM 和 GRU 循环神经网络构建编码器提取片段内和不同片段之间时序特征。

(3)AppScanner[32]。使用数据包包长序列和 40 余类序列统计特征作为输入,结合 SVC 和随机森林算法,实现手机 App 流量分类。

(4)AWF[33]。基于卷积神经网络构建特征提取器,用于自动检测网站指纹攻击流量。

(5)DeepFinger[34]。在 AWF 基础上改进网络框架,用于深度检测增加防御策略后的 Tor 浏览器流量。

(6)FS-Net[14]。提取流中的数据包包长序列作为输入,基于 Bi-GRU 循环神经构建流量特征提取器,并引入重构损失保证所编码特征的有效性。

3.2.2　评价指标

本文采用召回率(R)、精准率(P)、准确率(A)和 F_1 值来评估所提方法在各个分类任务上的性能。

$$R=\frac{T_P}{T_P+F_N} \tag{14}$$

$$P=\frac{T_P}{T_P+F_P} \tag{15}$$

$$A=\frac{T_P+T_N}{T_P+F_N+F_P+T_N} \tag{16}$$

$$F_1=\frac{2PR}{P+R} \tag{17}$$

式中:T_P 表示正确分类正样本;F_P 表示将负样本分类为正样本;F_N 表示将正样本分类为负样本;T_N 表示正确分类负样本。考虑到数据集中存在的样本不均衡问题,本文采用加权平均计算分类结果。

3.3　实验结果与分析

3.3.1　公开数据集实验结果

本文与现有基于深度学习的 packet-level 流量分类方法 DeepPcaket 和 BSNN,以及 flow-level 流量分类方法 AppScanner、AWF、DeepFinger 和 FS-Net,在 ISCXTor2016 和 ISCXVPN2016 公开数据集的 3 个分类任务上进行对比实验,实验结果如表 2 所示。

表 2　ISCXTor2016 和 ISCXVPN2016 公开数据集对比实验结果

Table 2　Comparison results on ISCXTor2016 and ISCXVPN2016

方法	ISCXTor			ISCXVPN-Service			ISCXVPN-Application		
	召回率	精准率	F_1	召回率	精准率	F_1	召回率	精准率	F_1
BSNN	0.901 7	0.902 4	0.896 1	0.893 1	0.894 5	0.886 0	0.918 4	0.921 7	0.916 0
DeepPacket	0.916 4	0.919 3	0.912 4	0.924 2	0.937 9	0.921 7	**0.946 7**	**0.951 3**	**0.946 7**
AppScanner	0.442 2	0.375 6	0.391 3	0.722 5	0.733 9	0.719 7	0.519 8	0.486 4	0.493 5
AWF	0.802 8	0.789 0	0.783 9	0.687 9	0.709 2	0.665 2	0.770 1	0.778 7	0.767 5
DeepFinger	0.847 6	0.845 0	0.840 9	0.820 2	0.828 7	0.810 0	0.809 7	0.831 2	0.808 9
FS-Net	0.917 6	0.921 1	0.915 6	0.911 7	0.914 3	0.909 5	0.901 8	0.906 4	0.899 7
FAIN	**0.933 8**	**0.932 5**	**0.931 7**	**0.931 5**	**0.931 8**	**0.929 3**	0.912 7	0.917 1	0.911 0

注:加粗部分表示最优值。

由表 2 可知,在 ISCXTor 和 ISCXVPN-Service 分类任务上,FAIN 相较其余 baseline 方法,在 3 个评价指标上均排名第一。

(1)在 ISCXTor 任务上,FAIN 的 F_1 值相较于 DeepPacket 和 FS-Net 分别提升了 1.74% 和 1.62%;对于 ISCXVPN-Service 分类任务,FAIN 的 F_1 值相较于 DeepPacket 和 FS-Net 分别提升了 0.76%和 1.98%。

(2)在 ISCXVPN-Application 分类任务上,FAIN 相较于次优 flow-level 方法 FS-Net,其 F_1 值提升了 1.13%。但是,在该任务上 FAIN 的表现不及 packet-level 方法 DeepPacket 和 BSNN。

实验结果表明,本文提出的基于 Transformer-encoder 框架的流量分类模型 FAIN,能够从原始流量中有效地挖掘流量序列的时序特征,并构建具有可分辨性的表示向量用于流量分类。

在 ISCXVPN-Application 分类任务上,FAIN 模型表现不及 packet-level 方法 BSNN 和 DeepPacket,这是由于 ISCXVPN2016 数据集提供的是原始 pcap 文件,packet-level 方法提取 pacp 文件中每个数据包的字节序列生成训练数据集和测试数据集,而作为 flow-level 方法的 FAIN 是提取 pacp 文件中每条流的包长序列生成训练数据集和测试数据集。由于样本形式不同,导致这两类方法的数据集样本数量和标签分布存在差异。如图 2 所示,packet-level 方法的训练集从样本数量和样本均衡度上均远好于 flow-level 方法的训练集,导致 FAIN 在该任务上的表现不及 BSNN 和 DeepPacket。

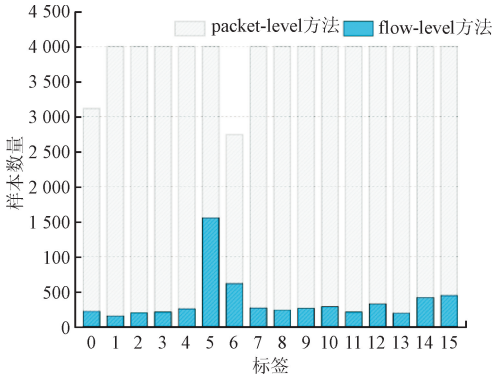


图 2 ISCXVPN-App 任务训练集数据分布

Fig. 2 The distribution of ISCXVPN-App training set

3.3.2 小样本数据集实验结果

在小样本实验中，本文选取 5 种 flow-level 流量分类方法 AWF、AppScanner、DeepFinger、BiLSTM 和 FS-Net 作为主干网络，分别结合 FOMAML 和Reptile 梯度元学习算法作为 baseline 模型，同时也和当前较为常用的小样本学习方法 Prototypical Network、Relation Network 进行了对比。表 3 和表 4 分别展示了上述方法与本文方法在 XJTU-FSTC 以及 CSTNET 小样本流量分类数据集上的对比结果。其中，XJTU-FSTC 数据集对比实验结果表明，本文方法在所有设置的小样本分类任务中效果最优。

表 3 XJTU-FSTC 数据集对比实验结果

Table 3 Comparison results on XJTU-FSTC

方法	标签数为 5 类时的准确率			标签数为 10 类时的准确率		
	1 样本	5 样本	10 样本	1 样本	5 样本	10 样本
FOMAML＋AWF	0.248 0	0.273 1	0.274 5	0.140 5	0.154 6	0.161 8
FOMAML＋DeepFinger	0.396 5	0.491 8	0.532 3	0.248 2	0.305 9	0.329 4
FOMAML＋BiLSTM	0.334 0	0.381 2	0.394 6	0.206 0	0.235 9	0.219 1
FOMAML＋FsNet	0.440 5	0.643 2	0.714 1	0.351 8	0.496 2	0.536 4
Reptile＋AWF	0.277 2	0.335 1	0.362 3	0.156 4	0.207 8	0.217 6
Reptile＋DeepFinger	0.405 5	0.456 8	0.481 1	0.267 3	0.297 0	0.317 9
Reptile＋BiLSTM	0.337 1	0.527 6	0.594 5	0.174 2	0.264 9	0.313 4
Reptile＋Fs-Net	0.392 8	0.631 7	0.722 2	0.294 9	0.509 8	0.597 3
Relation Network	0.193 6	0.201 2	0.204 4	0.108 3	0.127 2	0.156 6
Prototypical Network	0.555 0	0.707 1	0.756 5	0.430 7	0.582 2	0.636 8
FAIN＋Reptile	0.426 4	0.694 0	0.771 1	0.325 2	0.634 3	0.702 8
FAIN＋Reptile＋Pre-train	**0.608 0**	**0.795 2**	**0.876 3**	**0.452 6**	**0.728 4**	**0.786 7**

注：加粗部分表示最优值。

表 4 CSTNET 数据集对比实验结果

Table 4 Comparison results on CSTNET

方法	标签数为 5 类时的准确率			标签数为 10 类时的准确率		
	1 样本	5 样本	10 样本	1 样本	5 样本	10 样本
FOMAML＋AWF	0.320 8	0.339 5	0.371 1	0.183 4	0.235 9	0.249 3
FOMAML＋DeepFinger	0.387 0	0.592 9	0.659 7	0.285 8	0.440 5	0.511 9
FOMAML＋BiLSTM	0.413 1	0.440 6	0.458 0	0.224 2	0.291 1	0.234 3
FOMAML＋FsNet	0.617 9	0.809 7	0.839 5	0.484 0	0.565 8	0.616 6
Reptile＋AWF	0.333 7	0.425 1	0.491 0	0.199 2	0.270 7	0.318 4
Reptile＋DeepFinger	0.202 6	0.239 1	0.225 4	0.101 0	0.101 6	0.105 2
Reptile＋BiLSTM	0.334 0	0.381 2	0.593 5	0.159 4	0.223 7	0.272 2
Reptile＋Fs-Net	0.487 2	0.738 5	0.808 5	0.313 9	0.598 9	0.670 3
Relation Network	0.193 6	0.201 2	0.204 4	0.108 3	0.127 2	0.156 6
Prototypical Network	**0.679 8**	0.826 1	0.832 5	0.451 5	0.668 9	0.763 7
FAIN＋Reptile	0.539 6	0.835 2	0.875 4	0.398 9	0.710 1	0.772 5
FAIN＋Reptile＋Pre-train	0.655 3	**0.887 4**	**0.901 2**	**0.481 3**	**0.758 2**	**0.805 4**

注：加粗部分表示最优值。

（1）在 5 类 1、5、10 样本任务上分别取得了 68.80%、79.52%和 87.63%的分类准确率，在10 类 1、5、10 样本任务上分别取得了 45.26%、72.84%和 78.67%的分类准确率。

（2）值得注意的是，当不使用预训练参数对 FAIN 的模型特征提取模块进行初始化时，FAIN 分类准确率在 5 类 1、5、10 样本任务上降低了26.16%、10.12%以及 10.52%，在 10 类 1、5、10 样本任务上，分类准确率分别降低了 12.74%，9.41%和 8.39%，而且在 5 类 1 样本和 10 类 1 样本条件下不及 FOMAML＋FsNe 和 Prototypical Network 方法。因为相比于 FOMAML＋FsNet 和 Prototypical Network，本文使用了参数量更大的 Transformer-based 流量特征提取模块，所以在样本稀缺场景下，模型难以进行有效学习；但是随着样本数目增加，在 5 类 5、10 样本和 10 类 5、10 样本条件下，即使不使用预训练参数进行模型初始化，本文方法依旧取得了最优结果，表明了 FAIN 模型在流量表示学习方面的优越性。

CSTNET 数据集对比实验结果同样表明，本文方法在所有设置的小样本分类任务上综合表现优于其余对比方法。

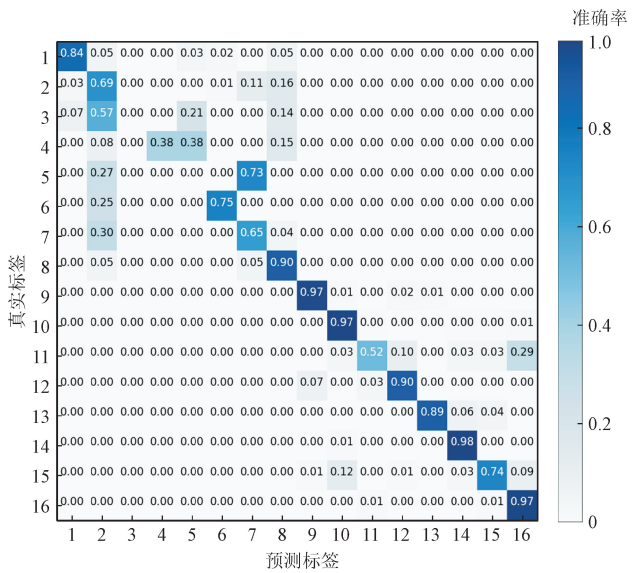
（1）除了在 5 类 1 样本条件下准确率低于 Prototypical Network，在其余的 5 类 5、10 样本和 10 类 1、5、10 样本任务上，均取得了最优的准确率，其值分别为 88.74%、90.12%、48.13%、75.82%和 80.54%。

（2）当不使用预训练参数对 FAIN 模型的特征提取模块进行初始化时，CSTNET 数据集上的对比结果与 XJTU-FSTC 数据集上结果表现出相同的规律。FAIN 分类准确率在 5 类 1、5、10 样本任务上分别降低了 11.57%、5.22%、2.58%，在 10 类 1、5、10 样本任务上，分类准确率分别降低了 8.24%、4.81%、3.29%，在 5 类 1 样本和 10 类 1 样本条件下不及 FOMAML＋FsNe 和 Prototypical Network 方法。

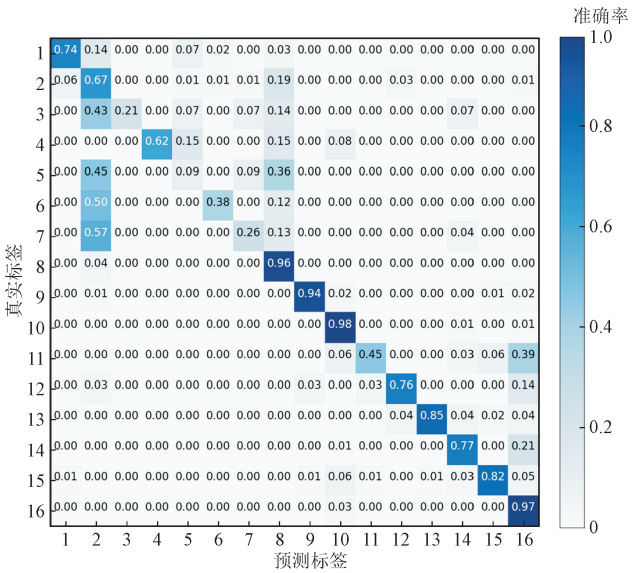
总的来看，由于流量的数据包序列不包含明显语义信息，因此基于深度学习的流量分类方法往往需要大量训练样本来学习不同应用的时序特征，而当训练样本稀缺时，就要求模型具备更强的表示学习能力以及泛化能力。在 XJTU-FSTC 和 CSTNET 小样本流量分类数据集上的对比结果表明，本文提出的流量分类模型 FAIN 经过小样本学习后的综合表现优于其余方法，充分体现了基于自注意力机制的表示学习模块在流量时序特征提取方面的有效性。同时，使用预训练参数作为元学习的初始化参数能够显著提升 FAIN 的性能，说明使用大量已知应用样本对 FAIN 进行预训练，能够赋予模型参数关于流量时序特征的通用领域知识。将预训参数作为 FAIN 在元学习过程的初始化参数，这些领域知识将作为先验知识，引导整个元学习过程中参数的梯度更新方向，能够有效提升训练稳定性，避免训练过程陷入局部最优，从而使得模型获得最优性能。

3.3.3　公开数据集细粒度分类结果

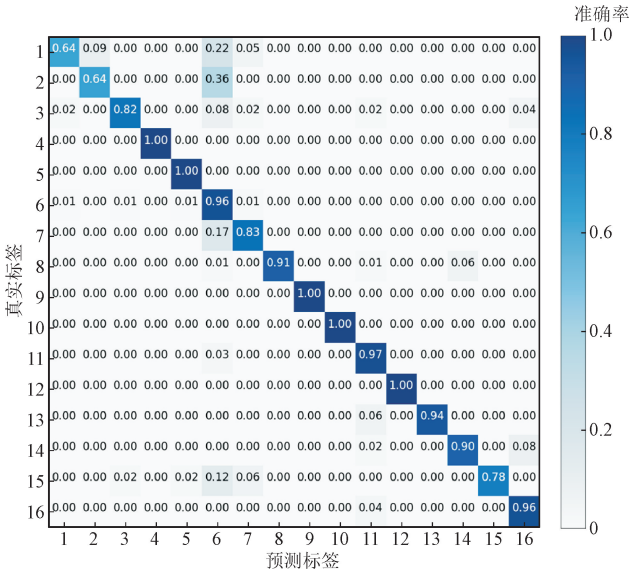
本文对比了同为 flow-level 方法的 FAIN 和 FS-Net 在不同分类任务上对每类标签的分类准确率。图 3(a)、(c)、(e)展示了 FAIN 在 ISCXTor、ISCXVPN-Appliction 和 ISCXVPN-Service 任务上的混淆矩阵，图 3(b)、(d)、(f)展示了 FS-Net 的结果。可以看出，在各个分类任务上，FAIN 对于绝大



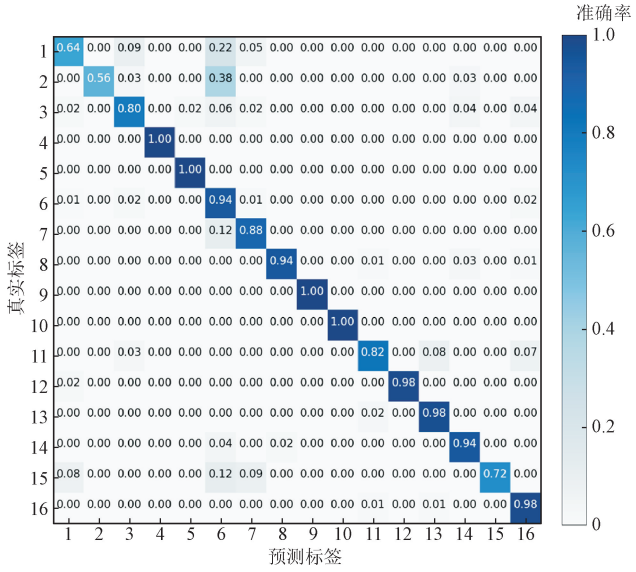
(a)Tor 任务上 FAIN 混淆矩阵



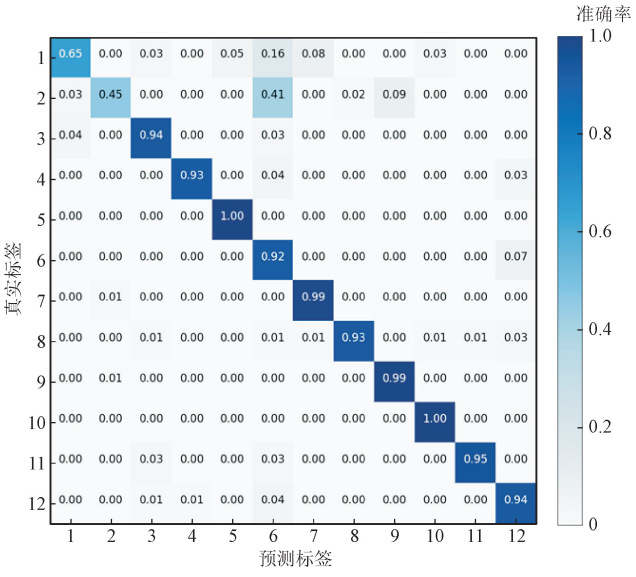
(b)Tor 任务上 FS-Net 混淆矩阵



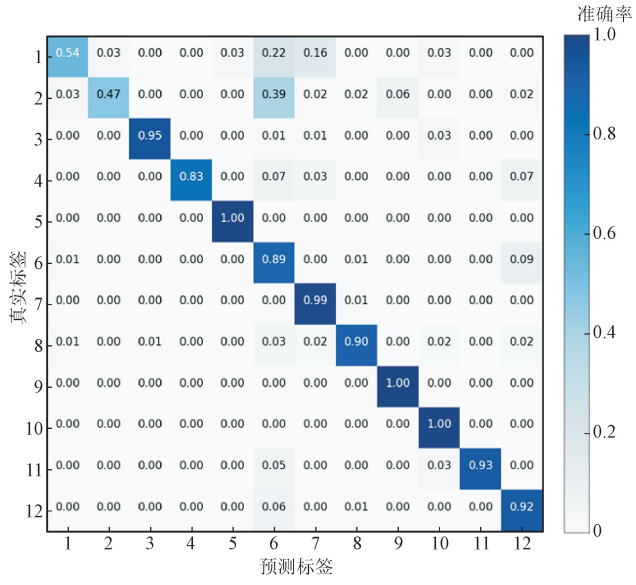
(c) VPN-App 任务上 FAIN 混淆矩阵



(d) VPN-App 任务上 FS-Net 混淆矩阵



(e) VPN-Service 任务上 FAIN 混淆矩阵



(f) VPN-Service 任务上 FS-Net 混淆矩阵

图 3 FAIN 和 FS-Net 在不同分类任务上的细粒度分类结果对比

Fig. 3 Comparison of fine-grained classification results between FAIN and FS-Net on different classification tasks

多数标签的分类准确率均优于 FS-Net。证明了 FAIN 在流量时序特征提取方面具有更强的鲁棒性和泛化能力,能更好地适用于不同的应用分类场景。同时,FAIN 和 FS-Net 在 ISCXTor 任务上对 Tor Browsing、Tor Chat、Tor File-Transfer、Tor Email 这几类标签的分类错误率较高,说明经由 Tor 浏览器转发的流量,其数据包之间的时序特征变得更加模糊难以被提取,这可能与 Tor 浏览器的路由机制有关。

3.4 参数影响分析

为了评估超参数对 FAIN 分类性能的影响,在 ISCXTor 数据集上分别对序列长度、编码层数、嵌入维度以及小样本学习过程内循环步数 α 进行参数分析,其中序列长度取值范围为{10, 20, 30, 60, 90, 120},编码层数取值范围为{2, 3, 4, 5, 6},嵌入维度取值范围为{16, 32, 64, 128, 256, 512},内循环步数的取值范围为{3, 4, 5, 6, 7, 8}。为保证实验公平性,其余参数不做更改。

图 4 展示了序列长度对 FAIN 的各项评价指标的影响。可以看出,当序列长度小于 30 时,模型性能随着序列长度的增加呈线性提升,当序列长度大于 30 时,模型性能不再明显提升,各项评价指标趋于稳定。这一结果说明,对于流量序列而言,前 30 个数据包相较于其余数据包蕴含更为重要的序列特征信息。

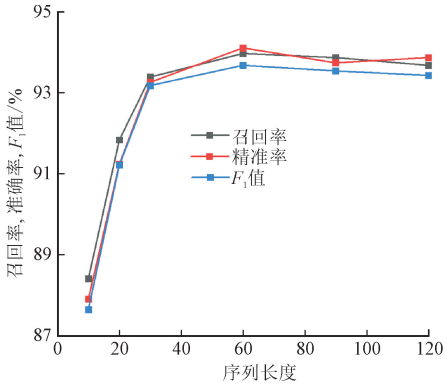


图 4　序列长度对 FAIN 分类性能的影响

Fig. 4　Effect of sequence length on FAIN performance

表示向量的维度对于分类效果有着重要影响，增加嵌入维度使得模型能够感知更加高维的隐式特征，但同时意味着特征空间更加稀疏，而且模型容易受到噪声和异常值的影响导致过拟合。图 5展示了模型性能随嵌入维度的变化趋势。可以看出，随着嵌入维度的增加，FAIN 的各项评价指标缓慢提升，而当嵌入维度大于 128 时，模型性能有了明显下降，当嵌入维度从 128 增至 512 时，F_1 值下降了 24.83%。

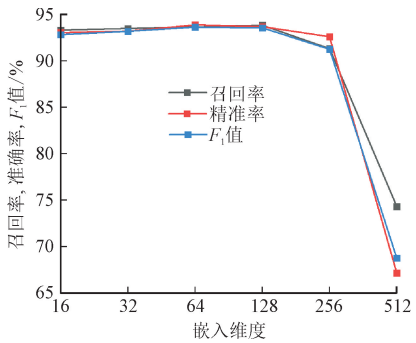


图 5　编码维度对 FAIN 分类性能的影响

Fig. 5　Effect of embedding dimension on FAIN performance

图 6 展示了编码层数对模型性能的影响。当编码层数小于 4 时，随着层数的不断增加，各项评价指标没有发生明显变化，而当层数从 4 层增至 6 层时，F_1 值下降了 19.68%。这一结果说明，基于自注意力机制构建的编码层具有良好的序列建模能力，即使是浅层网络，依然可以学习到丰富的特征信息，而加深编码层数会使得模型参数呈指数增加，当训练样本不足时就会引起过拟合，导致模型性能劣化。

图 7 展示了 5 类 1、5 样本和 10 类 1、5 样本条件下小样本学习过程中内循环更新步数对模型准确率的影响。为排除先验知识的影响，步数分析时不对模型进行预训练。从图中可以看出，随着内循环更新步数的不断增加，在 5 类 5 样本和 10 类 5 样本条件下，模型的准确率呈现缓慢上升趋势，模型准确率最高提升 2%；而在 5 类 1 样本和 10 类 1 样本条件下，模型的准确率呈现先缓慢上升之后下降的现象。总体来看，相较于其他参数，小样本学习过程中内循环更新步数对模型性能的影响有限。

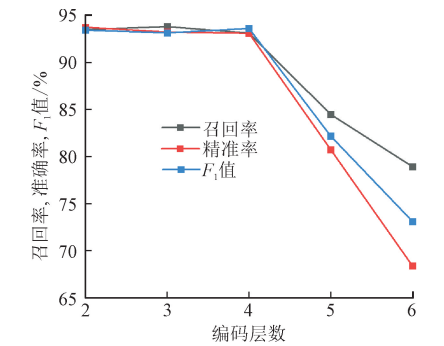


图 6　编码层数对 FAIN 分类性能的影响

Fig. 6　Effect of encoder layer on FAIN performance

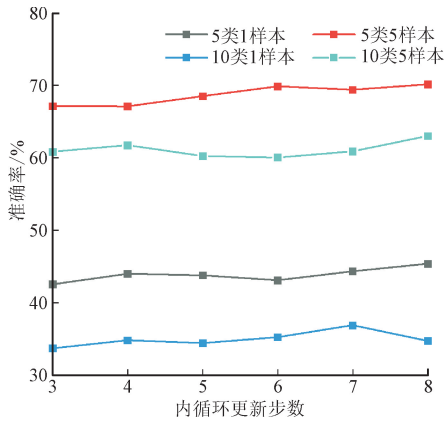


图 7　小样本学习中内循环更新步数对模型准确率的影响

Fig. 7　Effect of update steps of inner loop in few-shot learning on accuracy

4　结　论

(1)本文提出的 FAIN 流量分类模型在公开数据集上综合性能优于现有方法，且细粒度分类结果表明，FAIN 模型在流量时序特征提取方面具有更强的鲁棒性和泛化能力，能更好地适用于不同的应用分类场景。

(2)优化后的 FAIN 模型在小样本数据集上的综合性能优于对比方法，其中在 XJTU-FSTC 数据集的 5 类和 10 类分类任务上准确率最高分别提升了 16.75%、10.08%和 11.57%、8.24%。这表明使用预训练参数作为元学习的初始化参数能够显著提升 FAIN 模型在样本稀缺场景下的性能。

未来,将会考虑在计算资源有限的边缘网络设备上,部署能够实时分类流量的轻量化模型,进一步提升对网络空间的管理能力。

参考文献:

[1] DIERKS T, RESCORLA E. The transport layer security (TLS) protocol version 1.2 [EB/OL]. [2023-01-01]. https://www.rfc-editor.org/rfc/rfc5246.

[2] BARAKABITZE A A, AHMAD A, MIJUMBI R, et al. 5G network slicing using SDN and NFV: a survey of taxonomy, architectures and future challenges [J]. Computer Networks, 2020, 167: 106984.

[3] HÖFER C N, KARAGIANNIS G. Cloud computing services: taxonomy and comparison [J]. Journal of Internet Services and Applications, 2011, 2(2): 81-94.

[4] Cisco. Cisco annual internet report (2018-2023) white paper [EB/OL]. (2020-03-10)[2023-01-01]. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html.

[5] REZAEI S, LIU Xin. Deep learning for encrypted traffic classification: an overview [J]. IEEE Communications Magazine, 2019, 57(5): 76-81.

[6] TAYLOR V F, SPOLAOR R, CONTI M, et al. Robust smartphone app identification via encrypted network traffic analysis [J]. IEEE Transactions on Information Forensics and Security, 2018, 13(1): 63-78.

[7] QURESHI H N, MANALASTAS M, ZAIDI S M A, et al. Service level agreements for 5G and beyond: overview, challenges and enablers of 5G-healthcare systems [J]. IEEE Access, 2021, 9: 1044-1061.

[8] YAN Xiaodan, XU Yang, XING Xiaofei, et al. Trustworthy network anomaly detection based on an adaptive learning rate and momentum in IIoT [J]. IEEE Transactions on Industrial Informatics, 2020, 16(9): 6182-6192.

[9] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need [C]//Proceedings of the 31st International Conference on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc., 2017: 6000-6010.

[10] DEVLIN J, CHANG Mingwei, LEE K, et al. BERT: pre-training of deep bidirectional transformers for language understanding [EB/OL]. (2019-05-24)[2023-01-01]. https://arxiv.org/abs/1810.04805.

[11] TAN Mingxing, LE Q V. EfficientNet: rethinking model scaling for convolutional neural networks [C]// Proceedings of the 36th International Conference on Machine Learning. Chia Laguna Resort, Sardinia, Italy: PMLR, 2019: 6105-6114.

[12] DOSOVITSKIY A, BEYER L, KOLESNIKOV A, et al. An image is worth 16x16 words: transformers for image recognition at scale [EB/OL]. (2021-06-03)[2023-01-01]. https://arxiv.org/abs/2010.11929.

[13] LOTFOLLAHI M, JAFARI SIAVOSHANI M, SHIRALI HOSSEIN ZADE R, et al. Deep packet: a novel approach for encrypted traffic classification using deep learning [J]. Soft Computing, 2020, 24(3): 1999-2012.

[14] LIU Chang, HE Longtao, XIONG Gang, et al. FS-Net: a flow sequence network for encrypted traffic classification [C]//IEEE INFOCOM 2019-IEEE Conference on Computer Communications. Piscataway, NJ, USA: IEEE, 2019: 1171-1179.

[15] LI Rui, XIAO Xi, NI Shiguang, et al. Byte segment neural network for network traffic classification [C]// 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS). Piscataway, NJ, USA: IEEE, 2018: 1-10.

[16] 言洪萍, 周强, 王世豪, 等. 基于 SDN 的实际网络流中 Tor 网页复合特征提取方法 [J]. 通信学报, 2022, 43(3): 76-87.
YAN Hongping, ZHOU Qiang, WANG Shihao, et al. Composite tor traffic features extraction method of webpage in actual network flow based on SDN [J]. Journal on Communications, 2022, 43(3): 76-87.

[17] XIAO Xi, XIAO Wentao, LI Rui, et al. EBSNN: extended byte segment neural network for network traffic classification [J]. IEEE Transactions on Dependable and Secure Computing, 2022, 19(5): 3521-3538.

[18] 张小莉, 程光, 张慰慈. 基于改进深度卷积神经网络的网络流量分类方法 [J]. 中国科学(信息科学), 2021, 51(1): 56-74.
ZHANG Xiaoli, CHENG Guang, ZHANG Weici. Network traffic classification method based on improved deep convolutional neural network [J]. Scientia Sinica(Informationis), 2021, 51(1): 56-74.

[19] 谢绒娜, 马铸鸿, 李宗俞, 等. 基于卷积神经网络的加密流量分类方法 [J]. 网络与信息安全学报, 2022, 8(6): 84-91.
XIE Rongna, MA Zhuhong, LI Zongyu, et al. Encrypted traffic classification method based on convolutional neural network [J]. Chinese Journal of Network and Information Security, 2022, 8(6): 84-91.

[20] LI Jianfeng, ZHOU Hao, WU Shuohan, et al. FOAP: fine-grained open-world android app fingerprinting

[C]//Proceedings of the 31st USENIX Security Symposium. Boston, MA: USENIX Association, 2022: 1579-1596.

[21] 段雪源，付钰，王坤，等. 基于简单统计特征的 LDoS 攻击检测方法 [J]. 通信学报，2022，43(11)：53-64. DUAN Xueyuan, FU Yu, WANG Kun, et al. et al LDoS attack detection method based on simple statistical features [J]. Journal on Communications, 2022, 43(11): 53-64.

[22] ZHENG Wenbo, GOU Chao, YAN Lan, et al. Learning to classify: a flow-based relation network for encrypted traffic classification [C]//Proceedings of The Web Conference 2020. New York, USA: Association for Computing Machinery, 2020: 13-22.

[23] ACETO G, CIUONZO D, MONTIERI A, et al. Mobile encrypted traffic classification using deep learning: experimental evaluation, lessons learned, and challenges [J]. IEEE Transactions on Network and Service Management, 2019, 16(2): 445-458.

[24] SNELL J, SWERSKY K, ZEMEL R. Prototypical networks for few-shot learning [C]//Proceedings of the 31st International Conference on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc., 2017: 4080-4090.

[25] NICHOL A, ACHIAM J, SCHULMAN J. On first-order meta-learning algorithms [EB/OL]. (2018-10-22)[2023-01-01]. https://arxiv.org/abs/1803.02999.

[26] ZERVEAS G, JAYARAMAN S, PATEL D, et al. A transformer-based framework for multivariate time series representation learning [EB/OL]. (2020-10-06)[2023-02-01]. http://arxiv.org/pdf/2020.02803.pdf.

[27] FINN C, ABBEEL P, LEVINE S. Model-agnostic meta-learning for fast adaptation of deep networks [C]//Proceedings of the 34th International Conference on Machine Learning. Chia Laguna Resort, Sardinia, Italy: PMLR, 2017: 1126-1135.

[28] LASHKARI A H, DRAPER-GIL G, MAMUN M S I, et al. Characterization of tor traffic using time based features [C]//Proceedings of the 3rd International Conference on Information Systems Security and Privacy-ICISSP. Setúbal, Portugal: SciTePress, 2017: 253-262.

[29] DRAPER-GIL G, LASHKARI A H, MAMUN M S I, et al. Characterization of encrypted and vpn traffic using time-related [C]//Proceedings of the 2nd International Conference on Information Systems Security and Privacy-ICISSP. Setúbal, Portugal: SciTePress, 2016: 407-414.

[30] ANON. XJTU-FSTC dataset [EB/OL]. [2023-01-01]. https://drive.google.com/drive/folders/15gg0bvRpX3mQjEhqsve_LOvwtJIaT-c8?usp=share_link.

[31] LIN Xinjie, XIONG Gang, GOU Gaopeng, et al. ETBERT: A contextualized datagram representation with pre-training transformers for encrypted traffic classification [C]//Proceedings of the ACM Web Conference 2022. New York, USA: Association for Computing Machinery, 2022: 633-642.

[32] TAYLOR V F, SPOLAOR R, CONTI M, et al. AppScanner: automatic fingerprinting of smartphone apps from encrypted network traffic [C]//2016 IEEE European Symposium on Security and Privacy (EuroS&P). Piscataway, NJ, USA: IEEE, 2016: 439-454.

[33] RIMMER V, PREUVENEERS D, JUAREZ M, et al. Automated website fingerprinting through deep learning [EB/OL]. (2017-12-05)[2023-01-01]. https://arxiv.org/abs/1708.06376.

[34] SIRINAM P, IMANI M, JUAREZ M, et al. Deep fingerprinting: undermining website fingerprinting defenses with deep learning [C]//Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. New York, USA: Association for Computing Machinery, 2018: 1928-1943.

（编辑　亢列梅）