Beijing Forest Studio

北京理工大学信息系统及安全对抗实验中心

预训练加密流量分类方法

硕士研究生 李国浩

2025年02月09日

问题回溯

- 相关内容
  - 九尾狐
  - 2023.08.07 巩锟《预训练加密流量表征方法》
  - 2022.04.24 吴泽瀚《加密移动流量分析方法》
  - 2022.03.21 张钊《高准确率的鲁棒加密恶意流量实时检测方法》

内容提要

- 预期收获
- 题目内涵解析
- 研究背景与意义
- 研究历史与现状
- 知识基础
- 算法原理
  - BERT and Packet Headers
  - YaTC
- 特点总结与知识展望
- 参考文献

预期收获

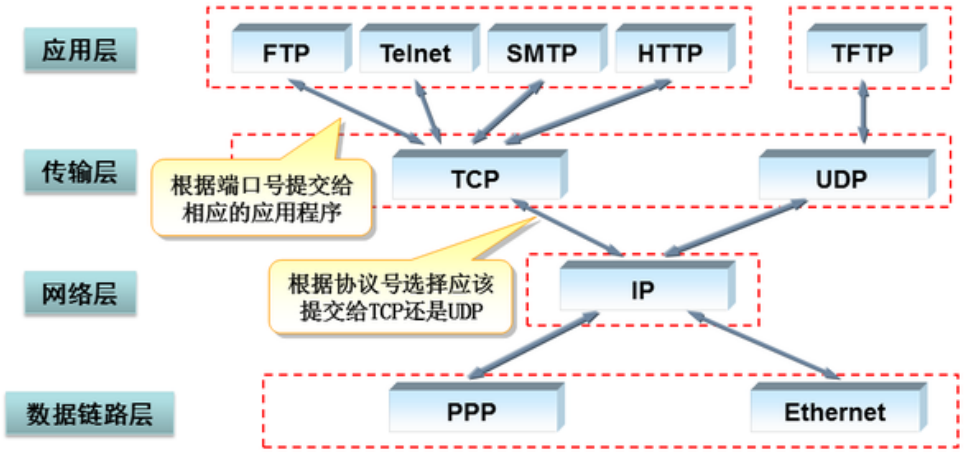
- 预期收获
  - 1.了解加密流量分类的基本概念和研究方向
  - 2.理解两种流量分类方法的基本原理
  - 3.了解现有方法的缺陷及未来的发展方向

题目内涵解析

- 加密流量
  - 由加密算法生成的流量，主要是指在通信过程中所传送的被加密过的实际明文内容
  - 网络流量的加密可以通过虚拟专用网络 (VPN)、传输层安全协议(TLS)、安全套接层(SSL)或其他加密技术实现
- 识别的对象
  - 加密流量识别对象是指识别的输入形式，包括流级、包级、主机级和会话级
  - 流级主要关注流的特征及到达过程，IP流根据传输方向可以分为单向流和双向流
  - 包级主要关注数据包的特征及到达过程，包级特征主要有包大小分布、包到达时间间隔分布等

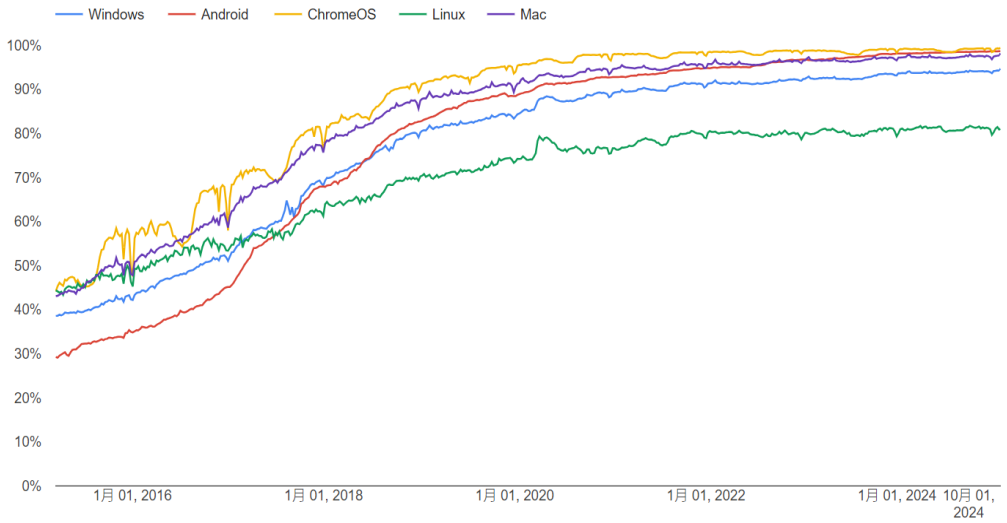
题目内涵解析

- 识别的类型
  - 加密流量识别类型指识别结果的输出形式，根据流量识别的应用需求确定识别类型
  - 加密与未加密流量：识别出哪些流量属于加密的，剩余则是未加密的
  - 协议识别：识别加密流量所采用的加密协议或应用层协议
  - 应用识别：识别流量所属的应用程序，如Zeus和YouTube等
  - 服务识别：识别加密流量所属的服务类型，如网页浏览、流媒体、IP电话等
  - 异常流量识别：识别出 DDoS、APT等恶意流量



研究背景与意义

- 研究背景
  - 加密流量快速增长
    - 采用HTTPS加密协议有利于搜索引擎排名
    - 加密协议良好的兼容性和可扩展性
  - 加密流量识别与非加密流量识别存在不少差异
  - 加密流量的快速增长带来许多互联网安全问题
- 研究意义
  - 流量分析和网络管理需要精细化识别加密流量
    - 恶意软件和入侵检测
    - 链路拥塞时改变路由策略



研究历史与现状

1992 描述互联网相关协议、方法等一系列文档的RFC定义了应用层协议的标准端口号。因此,通过标准端口号及其与应用协议之间的对应关系来识别网络流量的技术逐渐发展并得到广泛应用

2004 Sen等人提出一种利用应用程序级签名来识别 P2P 应用程序流量的方法，该方法通过分析文件和数据包级别的流量来识别应用程序的特征,并基于这些特征构建在线过滤系统,使其能在高速网络环境中有效追踪 P2P 流量

2013 Amoli等人提出一种实时无监督加密流量异常检测方法，用于检测正常和加密通信中的复杂攻击。该方法，通过分析字节、数据包、网络流的数量和时间,判断是否为异常流量

2017 Wang等人首次在加密流量应用识别研究中将特征选择、提取及分类集成到一个端到端框架中,使模型能够自动识别原始流量数据中的非线性关系

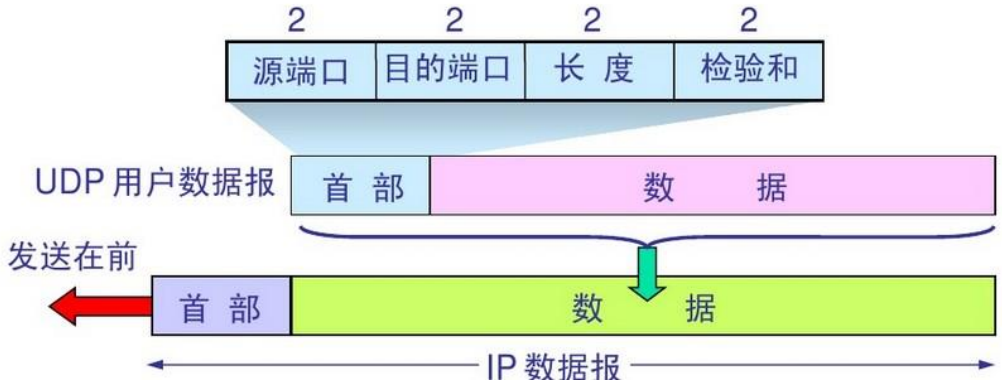
2019 Liu等人提出一种适合流序列特征的神经网络结构Fs-net,该方法将原始流量看作包长序列,利用多层编码器–解码器结构深入提取流的潜在序列特征,并通过引入重构机制增强特征的有效性。

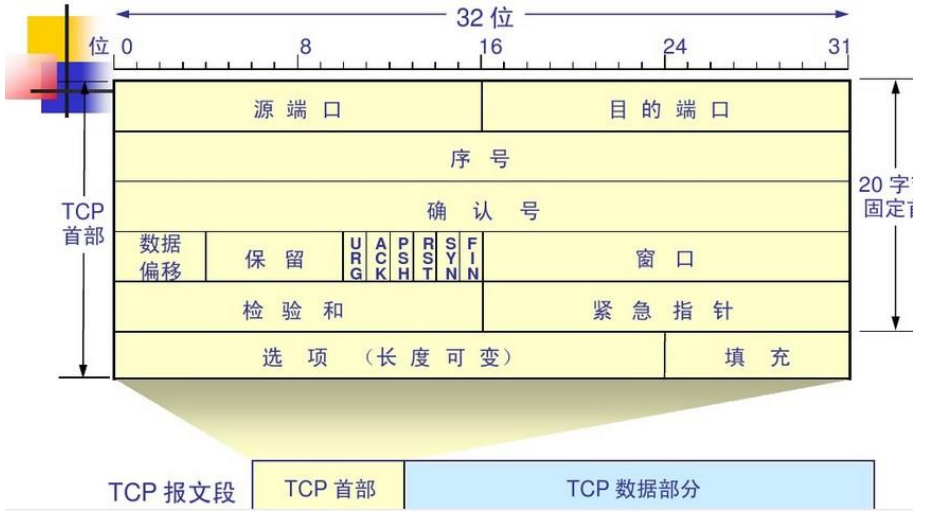
2022 Lin等人加密流量表征模型ET-BERT使用BURST预测任务和同源BURST预测任务,挖掘流量上下文信息,通过微调预训练好的模型即可实现不同场景下的流量类型识别

2023 Zhao等人提出了一种基于MAE的流量transformer用于流量分类，采用基于MAE的自监督学习范式，在预训练阶段从大量未标记的流量数据中学习通用的潜在表示，然后用少量标记数据对一系列流量分类任务进行监督微调

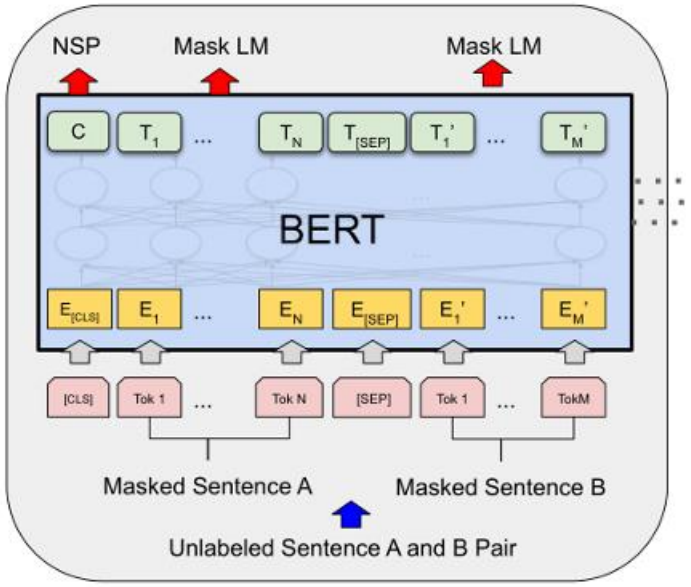
2024 Yu等人提出了一种基于BERT的新型服务类型和应用分类系统，该系统使用加密流量的包头信息，创建保留字段独特特征和上下文的句子，使用BERT模型对加密流量进行分类，实现了一个具有增强泛化性能的分类模型

- TCP/IP结构
  - Ethernet II帧格式
  - IP数据包格式
  - TCP数据段格式
  - UDP数据段格式





- **BERT**
  - 网络结构：多层双向Transformer编码器（块个数、隐藏层大小、多头数量）
  - 分隔符
    - [CLS]：每个序列的第一个标记，表示整个句子或句子对
    - [SEP]：用于分隔不同的句子或句子对
  - 输入表示
    - token嵌入
    - 分割嵌入
    - 位置嵌入
  - 预训练
    - MLM任务
    - 下一句预测（NSP）
  - 微调



Pre-training

算法原理

A novel approach for application classification with encrypted traffic using BERT and packet headers

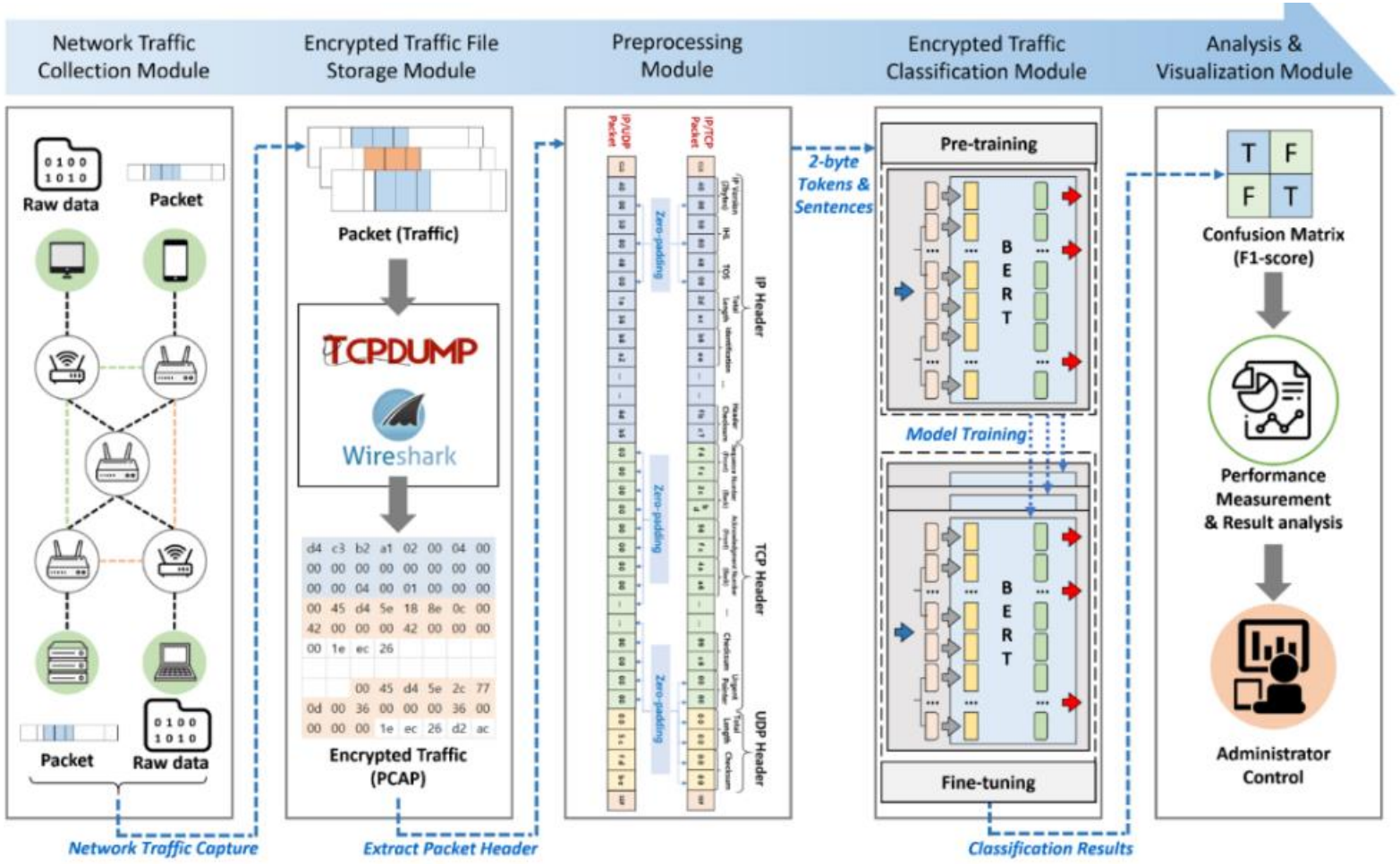
目标与难点

T	目标	利用加密流量中的包头信息实现服务类型和应用分类
I	输入	加密流量中不包括5元组和有效载荷的包头信息
P	处理	1.从PCAP或PCAPNG文件中提取数据包头，然后删除5元组和有效载荷 2.将每个标头字段的值转换为2字节的十六进制token 3.掩码15%的tokens进行重建 4.监督微调
O	输出	服务类型分类与应用分类结果

P	问题	5元组的使用在加密流量分类模型引入偏见，随机加密的有效载荷很难确保分类模型的泛化能力
C	条件	仅使用报头字段信息，不包括5元组和有效载荷
D	难点	包头字段单元长度的选择
L	水平	2024 CCF-B (Computer Networks)

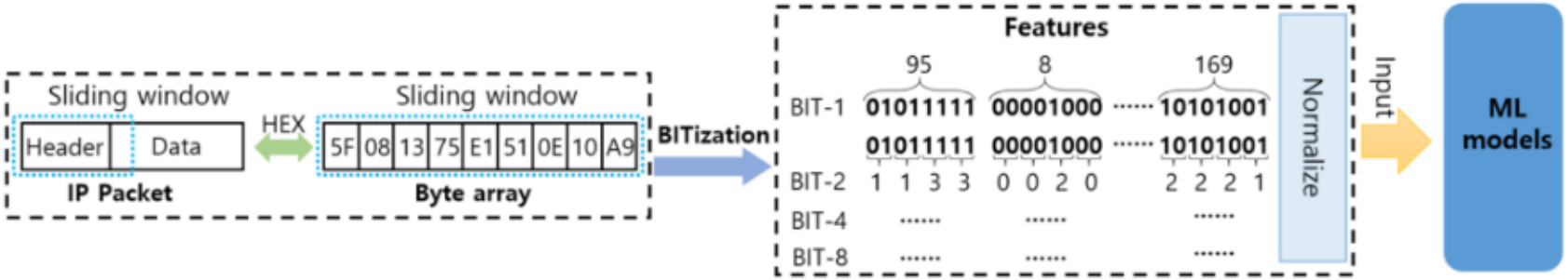
算法原理

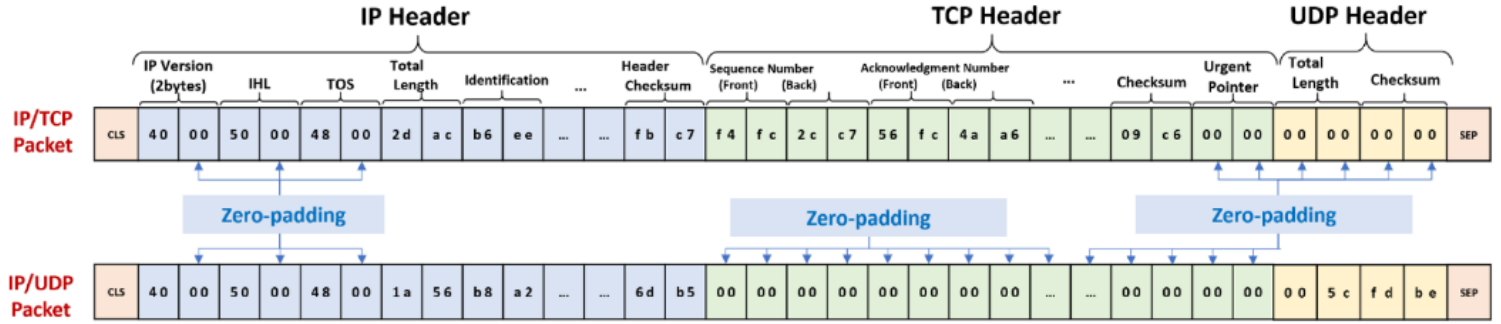
- 系统结构
  - 网络流量采集模块
  - 加密流量文件存储模块
  - 预处理模块
  - 加密流量分类模块
  - 分析和可视化模块



算法原理

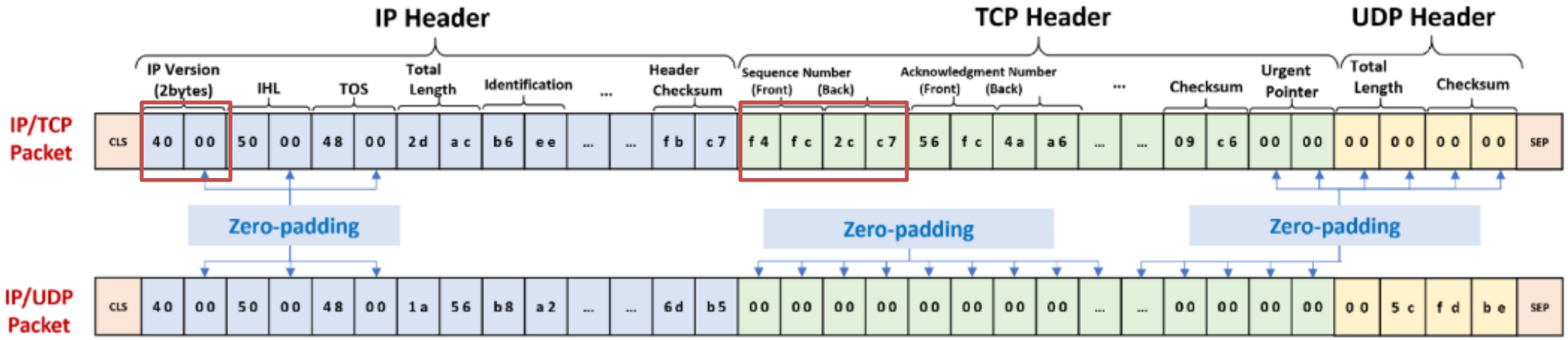
- 预处理模块
  - 先前方法：将数据包报头和数据转换为字节级十六进制值，然后将这些值提取到四个不同的特征集中：BIT-1、BIT-2、BIT-4和BIT-8
  - 从PCAP或PCAPNG文件中提取数据包标头，然后删除5元组和有效载荷
  - 包头字段的值转换为2字节的十六进制令牌，用于创建BERT预训练和微调的句子





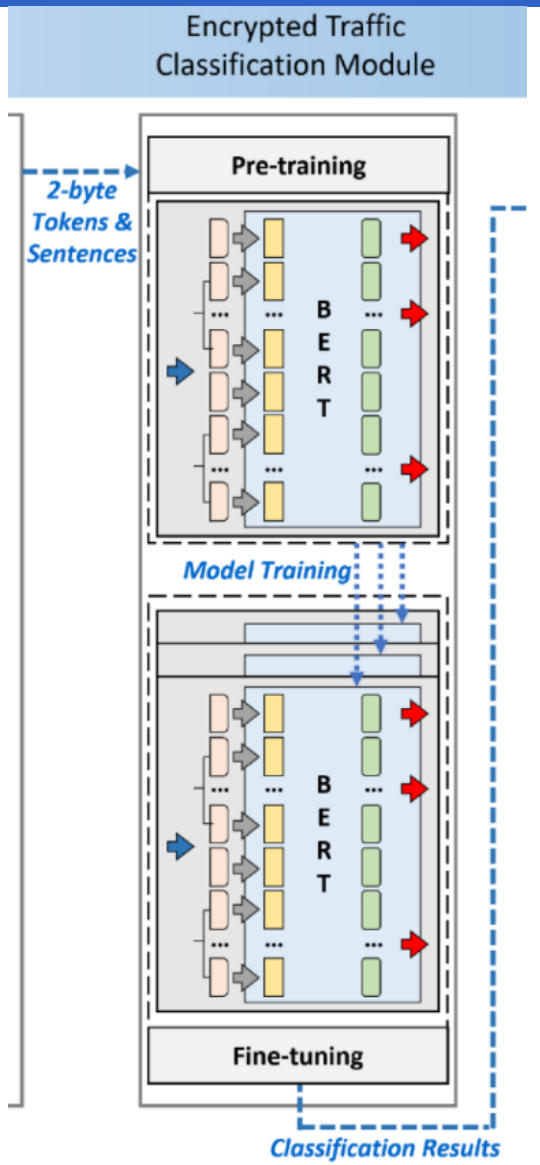
算法原理

- 预处理模块
  - 以2字节令牌的形式可以保持语义定义，未使用的字段进行零填充
  - TCP报头的序列号和确认号字段为4字节，仍以2字节划分



算法原理

- 加密流量分类模块
  - 输入
    - 将得到的令牌嵌入、位置嵌入和分段嵌入合并
  - 预训练
    - 仅使用掩码语言建模任务
    - 一个句子15%的被随机屏蔽
    - 掩蔽率太高，模型就很难学习上下文；掩码率太低，则需要很长时间才能收敛
  - 微调
    - 使用少量未在预训练中使用的标记数据进行微调
    - 数据是根据服务类型或应用程序的分类目标定制



算法原理

- 实验数据集
  - ISCX VPN-nonVPN数据集分为12种服务类型和17种应用程序
  - 排除了ISXC VPN-VPN数据集中未包含的服务类型的点对点（P2P）、应用程序的Tor和非VPN的Torrent
  - 每个服务类别和应用程序中随机提取最多100000个数据包

Dataset	Services	Applications
ISCX VPN-nonVPN	Chat	AIM-Chat
	Email	Email
	File-Transfer (FT)	Facebook
	Streaming	Gmail
	VoIP	Hangout
	VPN—Chat	ICQ-Chat
	VPN-Email	Netflix
	VPN-FT	SCP
	VPN-P2P	Skype
	VPN-Streaming	Spotify
	VPN-VoIP	Vimeo
		VoipBuster
		VPN-Ftps
		VPN-Sftp
		Youtube

Dataset	Services Types	PCAP file list
ISCX VPN	VPN—Chat	aim_chat, facebook_chat, skype_chat, hangouts_chat, icq_chat
	VPN-Email	email, gmail
	VPN-FT	skype_files, ftps, sftp
	VPN-P2P	bittorrent
	VPN-Streaming	vimeo, youtube, netflix, spotify
	VPN-VoIP	facebook_audio, hangouts_audio, skype_audio, voipbuster

Dataset	Classification tasks	Number of total samples	Number of labels
VPN-nonVPN	Service types	1100,000	11 types
	Each application	1297,134	15 types

算法原理

- 服务类型分类
  - 与10种最受认可的方法比较
    - FlowPrint：生成指纹进行分类
    - CUMUL, AppScanner and BIND：提取并利用统计特性
    - DeepPacket, FS-Net and GraphDApp：深度学习方法
    - ET-BERT（流），ET-BERT（包），BFCN，PERT：预训练方法

Method	Accuracy	Precision	Recall	F1-score
AppScanner [27]	71.82	73.39	72.25	71.97
CUMUL [23]	56.10	58.83	56.76	56.68
BIND [28]	75.34	75.83	74.88	74.20
FlowPrint [19-20]	79.62	80.42	78.12	78.20
FS-Net [31]	72.05	75.02	72.38	71.31
GraphDApp [32]	59.77	60.45	62.20	60.36
DeepPacket [29]	93.29	93.77	93.06	93.21
PERT [38]	93.52	94.00	93.49	93.68
BFCN [11]	99.12	99.13	99.11	99.11
ET-BERT (flow) [10]	97.29	97.56	97.31	97.33
ET-BERT (packet) [10]	98.90	98.91	98.90	98.90
Proposed	99.25	99.26	99.24	99.24

算法原理

- 应用程序分类
  - 与ET-BERT（包）和BFCN模型相比，F1得分的性能指标分别降低了0.63%和0.67%
  - 上述方法失去数据包报头独特特性，数据包特征和语义信息的丢失
    - IP标头中的IP标志和片段偏移字段被组合成一个令牌，UDP的校验和和和有效载荷被提取为一个令牌
  - 19个token代表数据包报头的独特特征

Method	Accuracy	Precision	Recall	F1-score
AppScanner [27]	62.66	48.64	51.98	49.35
CUMUL [23]	53.65	41.29	45.35	42.36
BIND [28]	67.67	51.52	51.53	49.65
FlowPrint [19-20]	87.67	66.97	66.51	65.31
FS-Net [31]	66.47	48.19	48.48	47.37
GraphDApp [32]	63.28	59.00	54.72	55.58
DeepPacket [29]	97.58	97.85	97.45	97.65
PERT [38]	82.29	70.92	71.73	69.92
BFCN [11]	99.65	99.36	99.47	99.41
ET-BERT (flow) [10]	85.19	75.08	72.94	73.06
ET-BERT (packet) [10]	99.62	99.36	99.38	99.37
Proposed	98.74	98.76	98.73	98.74

算法原理

- 算法总结
  - 算法贡献
    - 提出了一种基于双向编码表示变换器（BERT）的新型服务类型和应用分类系统
      - 该系统仅利用了加密流量中的包头信息，确保了分类模型的准确性和泛化性能
    - 所提出的系统可以根据服务类型和应用程序对分类目标进行精细调整
  - 算法不足
    - 仅在单个数据集上验证
    - 识别目标单一
    - 模型复杂度较高，实时处理能力差

Yet Another Traffic Classifier: A Masked Autoencoder Based Traffic Transformer with Multi-Level Flow Representation

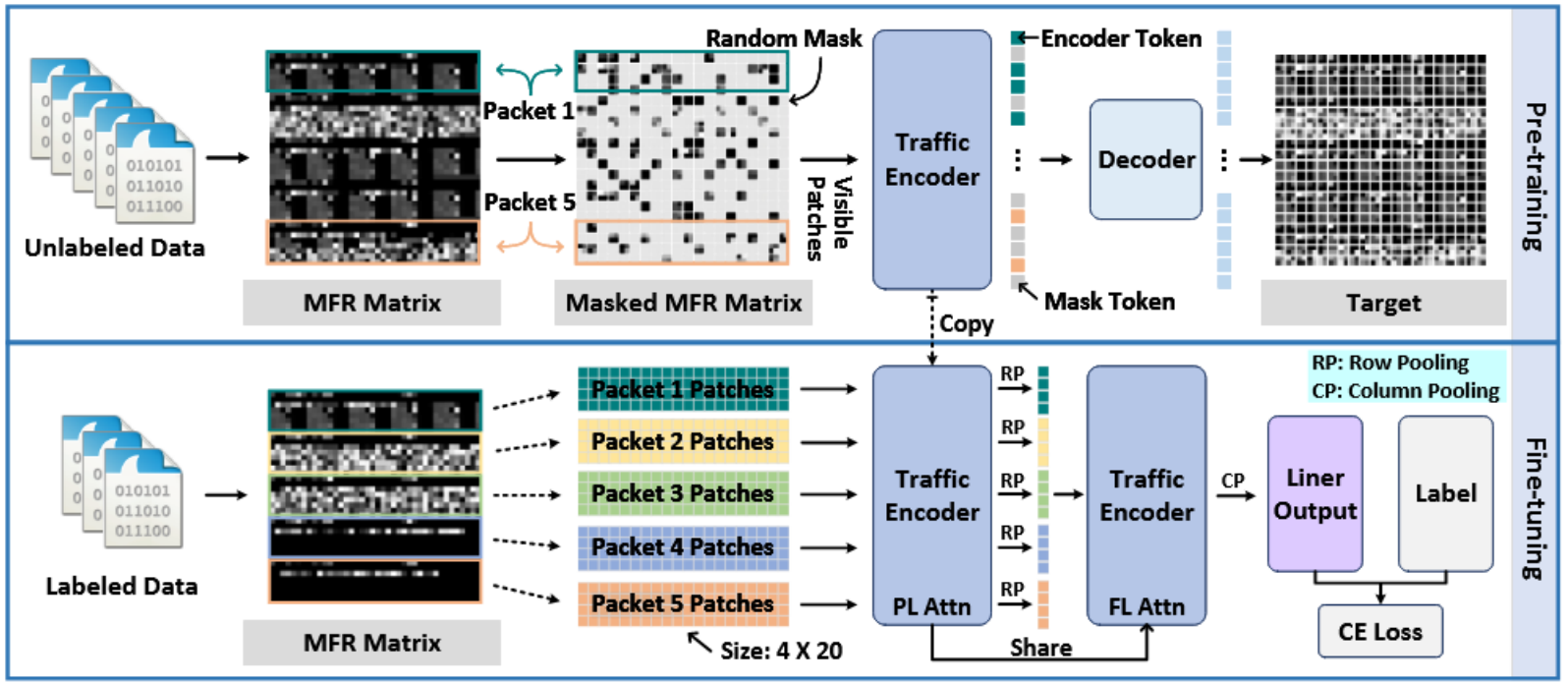
目标与难点

T	目标	设计较低复杂度和高效特征提取的流量分类器
I	输入	5个原始流量数据包
P	处理	1.根据原始数据包创建一个多级流表示（MFR）矩阵 2.构建了一种具有数据包级注意力模块和流级注意力模块的新型流量transformer 3.基于掩码自编码器（MAE）分两个阶段训练
O	输出	加密流量不同类型分类结果

P	问题	特定场景的分类器训练通常需要劳动密集型和耗时的过程来标记数据
C	条件	设计具有分层流量信息的格式化流量表示矩阵
D	难点	如何根据MFR矩阵设计包级注意力模块和流级注意力模块
L	水平	2023 CCF-A (AAAI)

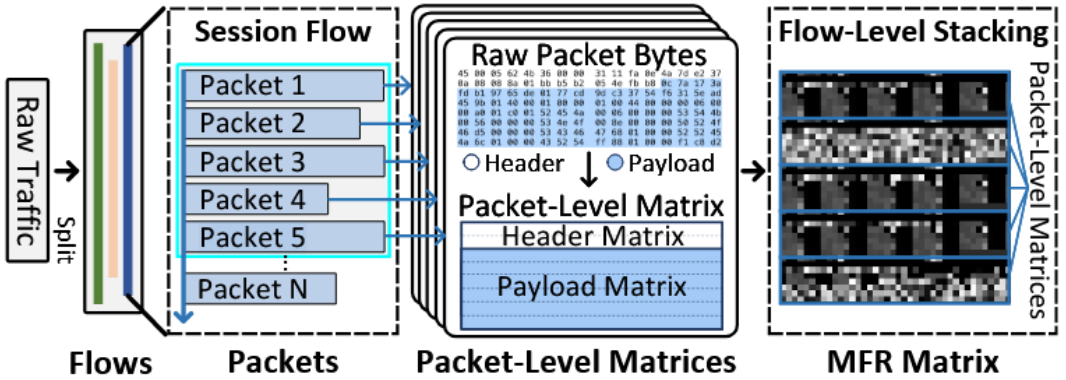
算法原理

- 系统架构
  - 多级流表示
    - 字节级别
    - 数据包级别
    - 流级别
  - 流量Transformer
    - 嵌入模块
    - 包级注意力模块
    - 流级注意力模块



算法原理

- 多级流表示
  - 先前方法
    - 直接截取流中前面固定数量的字节，形成二维矩阵
    - 矩阵中低级语义信息过多，影响了这些模型的有效性和效率
    - 在某些流中，第一个长数据包将占据整个矩阵
  - 本方法
    - 根据IP地址、端口号和协议类型将原始流量拆分为流
    - 删除了流的以太网报头，将端口号设置为零，并用随机地址替换IP，但保持其方向
    - 捕获流中的M个相邻数据包，并将其格式化为大小为H*W的二维矩阵



算法原理

- 流量Transformer
  - 嵌入模块
    - MFR矩阵$x \in R^{H*W}$被分割成大小为P×P的非重叠二维块patch，记为$x_p \in R^{N*P^2}$
    - 通过线性层将patch映射到D维向量作为patch嵌入
    - 将位置嵌入添加到patch嵌入中作为流量编码器的输入

$$x_0 = [x_p^1 E; x_p^2 E; ...; x_p^N E] + E_{pos}.$$

  - 包级注意力模块
    - 只在同一数据包中的patch之间执行多头自注意
    - 优先学习数据包内报头补丁或有效载荷补丁之间的依赖关系

$$Q = x_l W^Q, K = x_l W^K, V = x_l W^V,$$

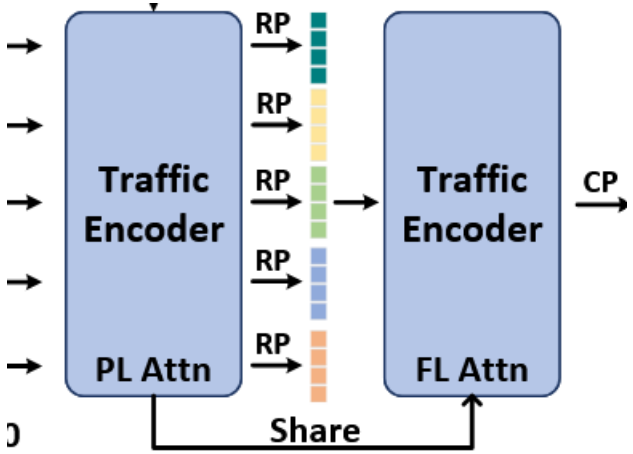
$$Attn(Q, K, V) = softmax(\frac{QK^T}{\sqrt{D_k}})V,$$

算法原理

- 流量Transformer
  - 流级注意力模块
    - 在包级注意力模块之后，MFR矩阵$x'_p \in R^{N*D}$的每个补丁的显著包级特征都被输出
    - 以更粗的粒度学习数据包间的关系
      - 按行池化（RP），$x_r \in R^{\sqrt{N}*D}$

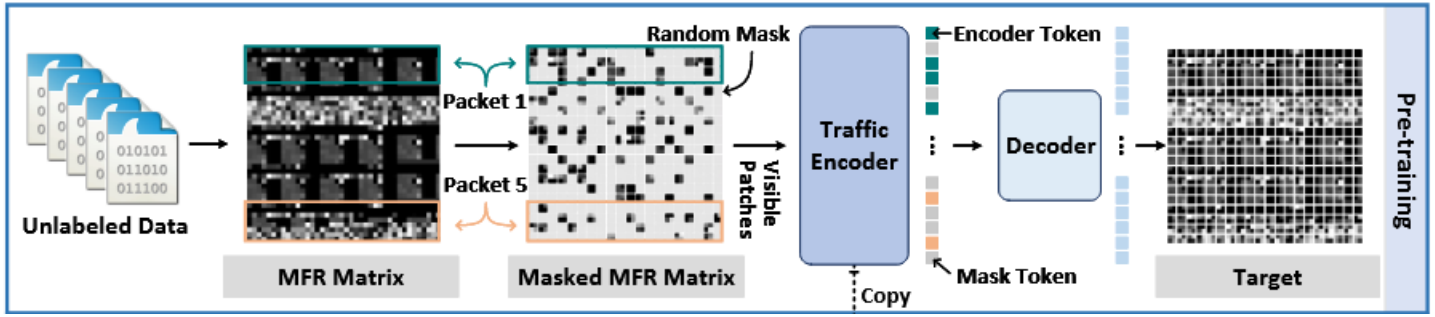
$$x_r = Pooling(x'_p),$$

      - 将MFR矩阵中的所有行补丁输入到流量编码器，输出一列行补丁特征$x_c \in R^{\sqrt{N}*D}$
      - 按列池化（CP），获得整个MFR矩阵的最终表示



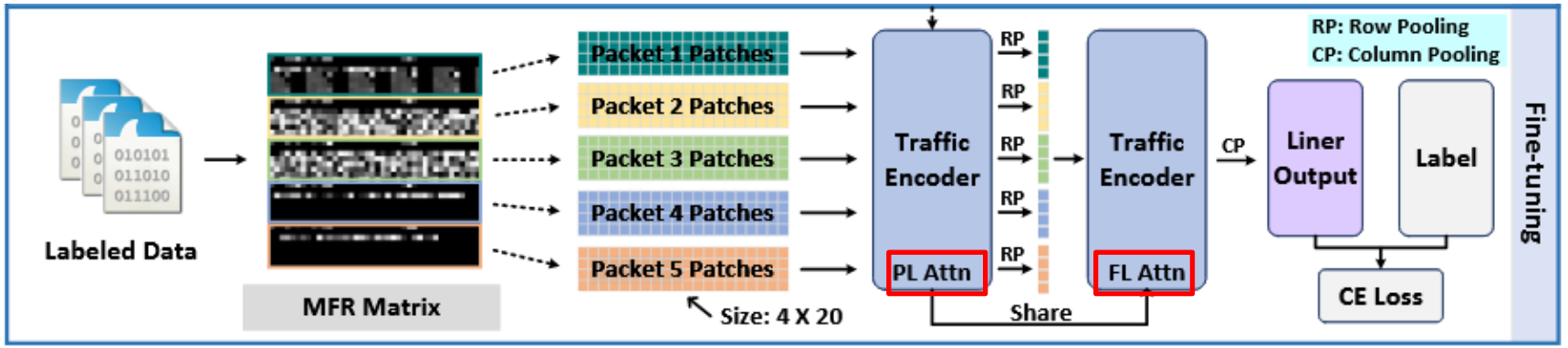
算法原理

- 训练策略
  - 非对称编码器-解码器架构的MAE
  - 很高比例的MFR补丁被随机屏蔽，只有一小部分补丁（即可见的未屏蔽补丁）被输入到模型中
    - 高掩码比导致缺乏原始信息来捕获数据包内和数据包之间的依赖关系
    - 在预训练期间，执行全局注意力
  - 流量编码器从这部分补丁中提取尽可能多的有效特征，然后输出编码器令牌
  - 小型解码器使用编码器令牌和掩码令牌恢复MFR矩阵的掩码区域
  - 重建损失（均方误差）进行训练 $\mathcal{L}_{rec} = MSE(y_{rec}, y_{real}).$



算法原理

- 训练策略
  - 在下游任务中，来自预训练的编码器参数被加载到流量transformer
  - 切换为包级注意力模块和流级注意力模块，并用于分组级和流级的特征提取
  - MFR矩阵的分类特征被展平并输入到MLP中，以获得预测分布$\hat{y} \in R^C$
  - 根据预测分布$\hat{y}$和地面真值标签y之间的交叉熵损失计算分类损失



算法原理

- 实验设置
  - 数据集
    - ISCX-VPN2016，ISCX-Tor2016，USTC-TFC2016，CIC-IoT2022，Cross-Platform
    - 前四个训练数据集形成了一个大规模的未标记训练数据集，用于预训练
    - 微调阶段，使用五个训练数据集进行监督学习
  - 对比方法
    - FlowPrint（2020）和AppScanner（2016）是基于机器学习的方法，使用统计特征进行流量分类
    - DF（2018）、Deeppacket（2020）、2D-CNN（2017）、3D-CNN（2020）和FS-Net（2019）是基于DL的流量分析方法，使用原始数据包信息进行监督学习
    - PERT（2020），ET-BERT（2022）将流量表示提取视为预训练的NLP任务，然后用有限的标记数据微调分类器

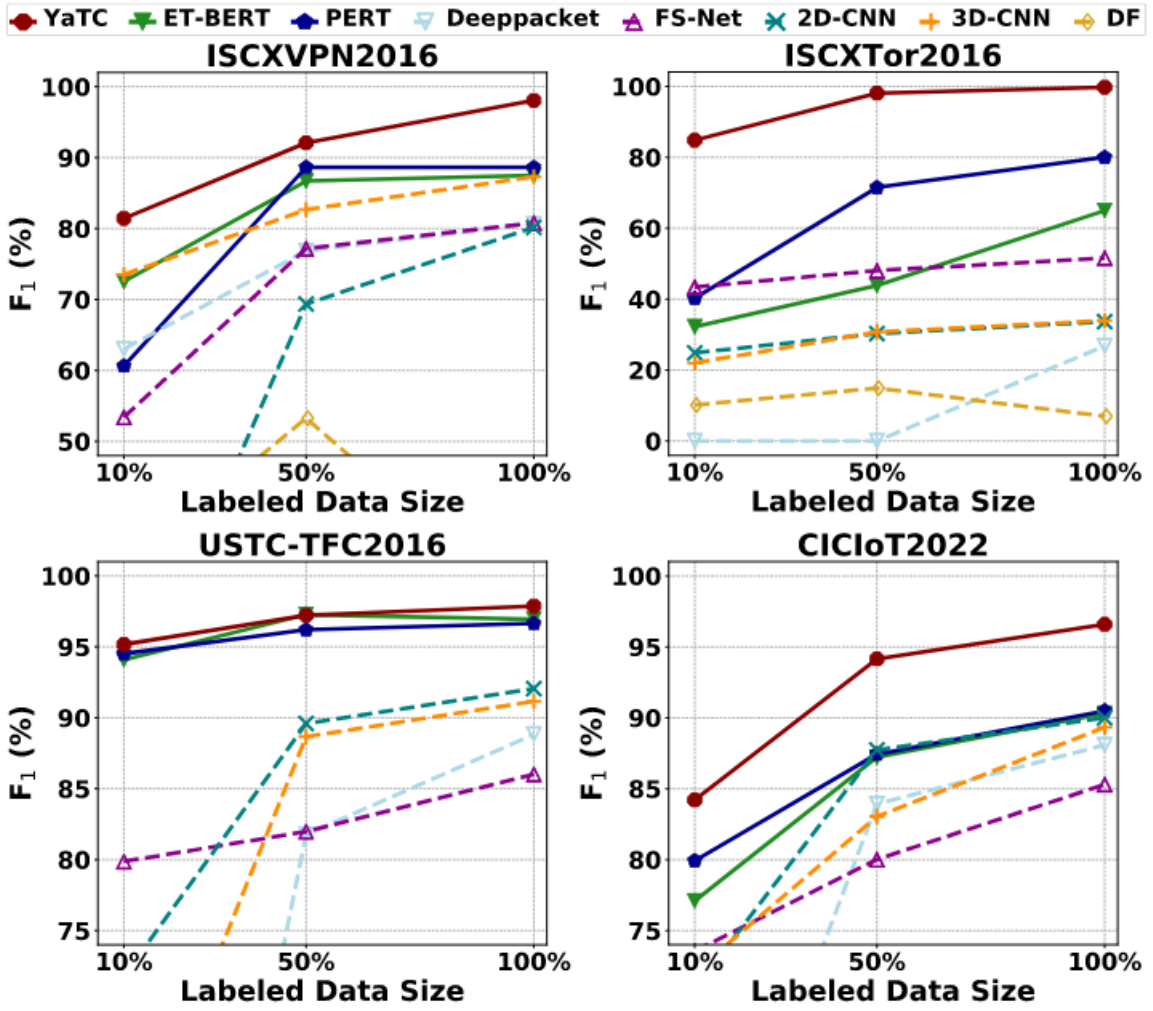
算法原理

- 实验结果
  - 基于DL的方法的有效性和基于ML的具有统计特征的方法的不足
  - 没有在ISCXTor2016数据集上进行预训练的方法表现不佳
    - 匿名流量的加密和混淆技术使得直接分析有效载荷变得困难
  - 预训练方法在除CIC-IoT2022之外的所有数据集上都优于其他方法

Method	ISCXVPN2016		ISCXTor2016		USTC-TFC2016		CICIoT2022	
	Acc.	F_1	Acc.	F_1	Acc.	F_1	Acc.	F_1
FlowPrint	30.29%	14.09%	25.27%	10.19%	25.30%	12.47%	50.46%	49.14%
AppScanner	79.93%	80.85%	50.27%	49.68%	60.41%	58.36%	76.52%	76.81%
DF	62.87%	25.40%	33.24%	7.00%	58.45%	49.15%	60.13%	46.35%
Deeppacket	80.21%	80.17%	36.81%	26.81%	88.49%	88.83%	88.28%	88.08%
2D-CNN	81.26%	80.64%	34.62%	33.66%	92.26%	92.05%	90.07%	90.00%
3D-CNN	81.09%	80.79%	34.89%	33.96%	91.55%	91.16%	89.39%	89.33%
FS-Net	87.64%	87.30%	52.03%	51.64%	87.05%	86.02%	85.37%	85.30%
PERT	88.62%	88.61%	80.22%	79.99%	96.63%	96.64%	90.52%	90.49%
ET-BERT	87.74%	87.47%	65.38%	64.98%	96.95%	96.95%	90.35%	90.31%
Ours (YaTC)	**98.07%**	**98.04%**	**99.72%**	**99.72%**	**97.86%**	**97.86%**	**96.58%**	**96.58%**

算法原理

- 小样本分析
  - 将标记的数据大小设置为10%、50%、100%
  - YaTC、ET-BERT和PERT，在小样本场景中通常优于其他监督方法
  - YaTC的性能都优于ET-BERET和PERT，表明其具有出色的鲁棒性



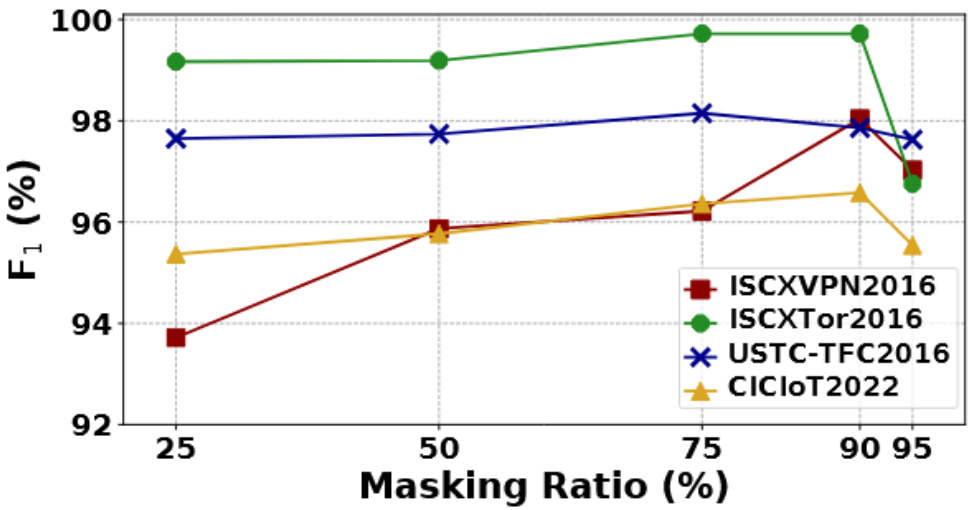
算法原理

- 消融实验
  - 与应用全局注意力相比，本方法降低了复杂性，取得了更好的结果
  - 去除包级注意力都会导致性能显著下降
  - 没有预训练的流级注意力有时会导致性能下降
    - 在没有预训练的情况下进一步关注小数据更容易导致过度拟合
  - 微调过程中应用参数共享可以提高轻量化和性能
  - 直接转换原始流量字节或仅删除MFR中的流级堆叠也会导致性能较弱

Method	ISCXVPN2016		ISCXTor2016		USTC-TFC2016		CICIoT2022	
	Acc.	F_1	Acc.	F_1	Acc.	F_1	Acc.	F_1
Ours (YaTC)	**98.07%**	**98.04%**	**99.72%**	**99.72%**	**97.86%**	**97.86%**	**96.58%**	**96.58%**
Ours with GA	95.27%	95.14%	98.63%	98.62%	97.86%	97.86%	95.64%	95.61%
Ours w/o PA	90.19%	90.03%	78.02%	77.28%	96.03%	96.03%	92.84%	92.78%
Ours w/o FA	95.62%	95.49%	99.18%	99.18%	97.66%	97.62%	95.81%	95.80%
Ours w/o FS	92.47%	92.35%	97.80%	97.77%	93.48%	93.48%	94.58%	94.57%
Ours w/o PS	97.55%	97.53%	99.45%	99.45%	97.45%	97.40%	95.41%	95.39%
Ours w/o PT	87.74%	87.22%	92.03%	91.90%	95.32%	95.25%	92.70%	92.65%
Ours w/o PT & PA	78.63%	77.58%	39.84%	38.58%	93.28%	93.22%	90.88%	90.79%
Ours w/o PT & FA	87.74%	87.40%	85.99%	85.84%	95.52%	95.46%	93.19%	93.17%
Ours w/o PT & FS	81.96%	81.84%	83.52%	83.15%	91.75%	91.48%	91.59%	91.59%
Ours w/o PT & MFR	80.91%	80.49%	42.86%	42.11%	93.99%	93.90%	91.36%	91.26%

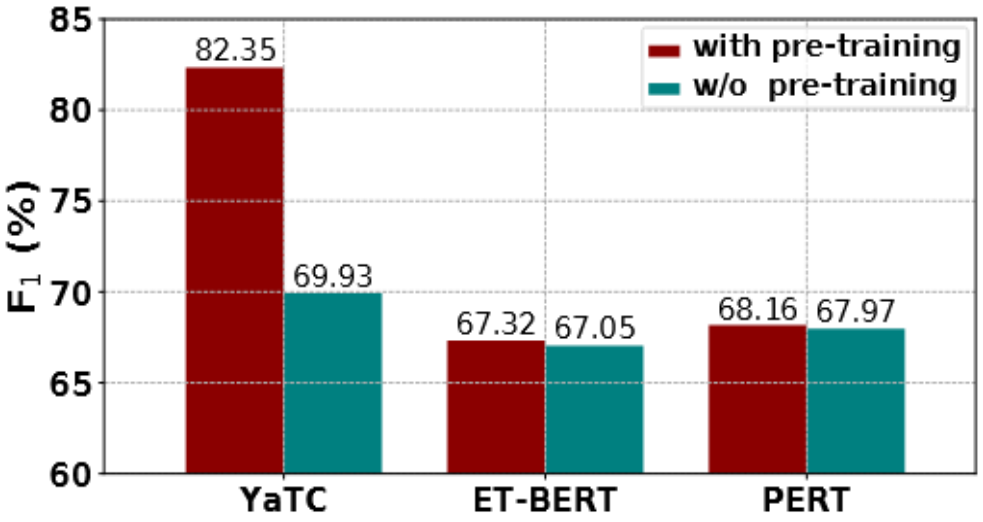
算法原理

- 讨论
  - 掩码率的影响
    - 更高的掩模比将带来更好的性能，但过高的掩蔽率使得重建任务过于困难
    - 对于分类任务，单词是高级和抽象的信息，但流量字节只是稀疏的特征，没有明确的语义单位，更类似于像素



算法原理

- 讨论
  - 迁移学习
    - 在跨平台数据集上评估迁移学习
    - YaTC将F1从69.93%显著提高到82.35%，比没有预训练的情况提高了12.42%
    - ET-BERT和PERT使用预训练进行弱提升，表明他们的预训练模型很难转移到新的下游流量分类任务



算法原理

- 算法总结
  - 算法贡献
    - 设计了一个MFR矩阵，充分考虑了流层次结构来表示原始流量
    - 构建了一种具有分组级和流级注意机制的新型流量Transformer，以较低的复杂度和较少的参数进行更有效的特征提取
  - 算法不足
    - 应对不同类型加密流量的细节可能不足，尤其是对新型加密技术的适应性
    - 降低了复杂度，但整体模型的计算开销仍然较大

特点总结与未来展望

研究工作总结

- 特点总结
  - BERT and Packet Headers
    - 提出了一种基于BERT的新型服务类型和应用分类系统，确保了分类模型泛化性能
    - 仅使用包头字段信息，并有助于用户隐私保护
  - YaTC
    - 提出了一种基于MAE和MFR的流量分类器，从流量表示、分类器结构和训练策略三个方面突破了传统的流量分析方法
    - 预训练模型对于新的下游分类分类任务表现出了出色的转移能力
- 未来展望
  - 不同层级流量特征的设计
  - 加密流量分类器的迁移性能，复杂度和实时处理能力

参考文献

- [1] YU J, CHOI Y, KOO K, et.al. A novel approach for application classification with encrypted traffic using BERT and packet headers[J]. Computer Networks, 2024, 254: 110747-110759.
- [2] ZHAO R, ZHAN M, DENG X, et.al. Yet Another Traffic Classifier: A Masked Autoencoder Based Traffic Transformer with Multi-Level Flow Representation[C]. Proceedings of the AAAI Conference on Artificial Intelligence, 2023, 37(4): 5420-5427.

知人者智，自知者明。胜人者有力，自胜者强。知足者富。强行者有志。不失其所者久。死而不亡者，寿。

谢谢！