

成绩：\_\_\_\_\_



# 网络攻防技术

## 课程设计

学 院： 网络空间安全学院

学 号： 19373474

姓 名： 屈佳宁

2023 年 6 月 25 日

# 目 录

一.	课程设计背景与目的 .....	4
1.1	课程设计背景 .....	4
1.2	课程设计目的 .....	5
二.	技术设计与实现 .....	6
2.1	数据集描述 .....	6
2.1.1	数据源 .....	6
2.1.2	数据描述性统计分析 .....	6
2.2	PCAP 特征提取与双向流匹配 .....	7
2.2.1	PCAP 文件的特征提取 .....	7
2.2.2	CSV 文件的双向流筛选 .....	9
2.2.3	PCAP 对 CSV 的匹配与合并 .....	9
2.3	流量特征分析 .....	11
2.3.1	数据抽样与处理 .....	11
2.3.2	箱型图分析 .....	12
2.4	特征选择 .....	13
2.4.1	基于语义的特征选择 .....	13
2.4.2	基于标准差的特征选择 .....	13
2.4.3	基于非参数统计的特征选择 .....	14
2.5	特征预处理 .....	15
2.5.1	数据清洗 .....	15
2.5.2	数据编码 .....	17
2.5.3	数据标准化 .....	17
2.6	分类模型简介 .....	18
2.6.1	逻辑回归 (Logistic Regression) .....	18
2.6.2	决策树 (Decision Tree) .....	18
2.6.3	随机森林 (Random Forest) .....	19

2.6.4	卷积神经网络 (CNN)	19
三.	结果测试	20
3.1	分类结果展示	20
3.1.1	准确率 (Accuracy)	20
3.1.2	F1-Score	21
3.1.2	ROC 曲线与 AUC	22
3.2	分类指标对比	25
3.3	算法分析	25
四.	创新点说明	26
4.1	数据特征提取标准多元	26
4.2	数据分类模型横纵对比	26
五.	团队成员及分工	27
六.	总结	27
七.	参考文献	28

## 一. 课程设计背景与目的

### 1.1 课程设计背景

随着信息技术不断发展，互联网广泛应用到各行各业中，网络流量日益规模化、复杂化，网络安全问题也接踵而至。恶意流量，即网络攻击、流量欺诈、恶意软件等，对网络和系统的安全造成严重威胁。在数字化时代，网络攻击者致力于捕捉网络漏洞，利用相应的技术手段，不断迭代和改进攻击方式，这使得网络安全形势更加严峻。

在真实世界中，我们需要应对复杂多样的恶意软件和计算机病毒，这些恶意软件通过各种方式，如网络钓鱼、恶意链接、潜在的下载源等，传播到用户的计算机中，并可能窃取敏感信息、加密用户数据，甚至控制用户设备或干扰正常的系统操作。此外，分布式拒绝服务攻击（DDoS）也具有一定危害性。攻击者利用大量的请求并占用系统资源，使得网络服务无法正常运行，导致服务中断、延迟或降级，给用户和企业带来严重的经济和声誉损失。

基于此背景，保护用户免受恶意流量的侵害，防止信息被窃取和服务被影响，成为保障网络安全的首要问题。作为网络安全领域的关键手段，网络流量分析可以帮助企业、用户及时了解网络中的数据传输和行为，分析并识别恶意流量，以便于及时采取相应的安全措施和应对策略。

通过对网络流量进行抓取，我们可以捕获到网络中的数据传输过程。对网络流量进行分析并提取出关键特征，包括通信的源地址和目的地址、传输的数据类型、协议类型及使用的端口号等，辅助以机器学习或深度学习模型，可实现异常行为和攻击模式的识别，从而对良性流量和恶意流量进行区分。这些数据具有丰富的信息利用价值，蕴含网络的运行机制和行为规律。

传统的网络安全防护手段往往依赖于规则和签名的匹配，随着恶意攻击的日益复杂和隐蔽性的增加，仅依靠传统手段已经难以有效应对。因此，通过对网络流量进行分析，可以获取更全面、深入的信息，识别恶意流量的特征和行为模式，从而提供更有效的网络安全防护策略。

## 1.2 课程设计目的

本课程设计基于传统网络流量在复杂网络下特征维度高，特征处理复杂度高等背景，实现良性与恶意流量的分类，为网络安全防护提供支持。

具体步骤包括：

a) **数据源选择：**本项目采用公共流量数据集 CIC-AndMal2017，使用 PCAP 文件提取分析网络流量，并用 CSV 文件中的数据作为标签辅助分析。

b) **流量特征分析：**为了区分恶意流量和良性流量，本项目采用数据抽样处理以及箱型图分析方法，判断选择体现两种流量显著差异的特征。

c) **流量特征选择：**特征进一步筛选，分别基于语义、标准差、非参数统计方法，便于后续训练分类模型。

d) **特征预处理：**关注特征的数据类型，处理可能出现的异常值，便于提高训练效果和判断精度，同时对数据进行标准化处理。

e) **分类模型训练：**通过恰当的特征训练，调整训练比例和测试比例，采用逻辑回归、决策树、随机森林、卷积神经网络等方法，区分正常流量和恶意流量。

f) **分类结果分析：**分别从准确率、精确率和召回率、敏感性和特异性等方面评估分类结果，判断最佳分类模型。

该项目旨在降低网络攻击和威胁对系统和用户的影响，通过完成上述步骤，为网络安全领域的研究和实践提供有益的参考和指导。通过抓取大量网络流量并提取流量数据特征进行深入分析，结合机器学习和数据挖掘技术，可以实现流量的检测和分类，实现良性与恶意流量的区分，为网络安全决策提供有力的支持，提高网络安全的防护能力。该项目成功实施将在网络安全领域产生重要的应用和研究价值，结合未来可能出现的完善流量分析算法和模型，提高恶意流量检测的准确性和效率，以应对日益复杂和隐蔽的网络攻击。此外，还可以探索网络流量抓取与分析在其他领域的应用，如网络性能优化、业务智能分析等，进一步拓展其应用范围和潜力。

## 二. 技术设计与实现

### 2.1 数据集描述

#### 2.1.1 数据源

本项目使用公共流量数据集 CIC-AndMal2017，使用 PCAP 文件提取分析网络流量，并用 CSV 文件中的数据作为标签辅助分析，整体流程图如图 1 所示。

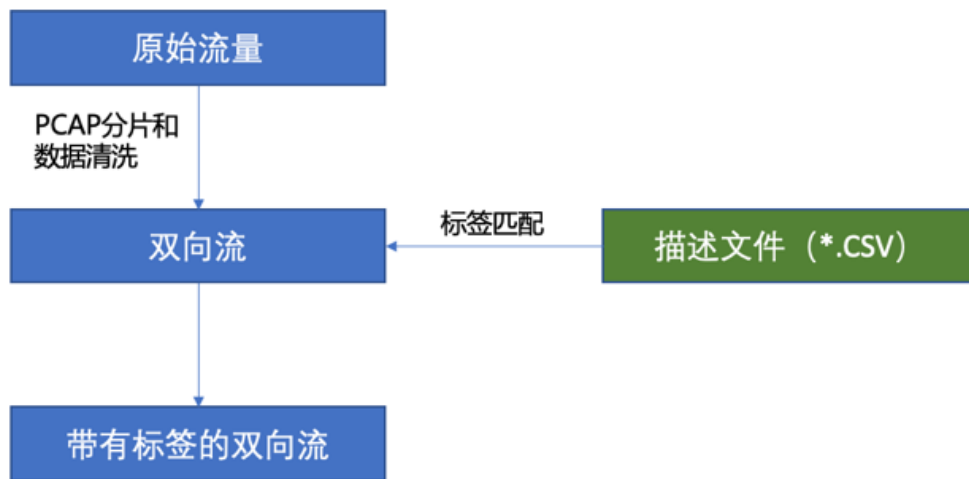


图 1 项目流程图

#### 2.1.2 数据描述性统计分析

经统计分析，数据集中的 CSV 文件约有 261 万条记录，共有包含 Flow\_ID、Source IP、Source Port、Destination IP、Destination Port、Protocol 等字段在内的 84 个特征。

Label 为该数据集的标签，用于标识该流量属于何种流量。CICAndMal2017 数据集中包括广告软件(Adware)、勒索软件(Ransomware)、恐吓软件(Scareware)、短信恶意软件(SMS Malware)四类恶意软件流量，以及大量的良性流量(Benign)。下图给出基于五分类和二分类的流量类型分布图。

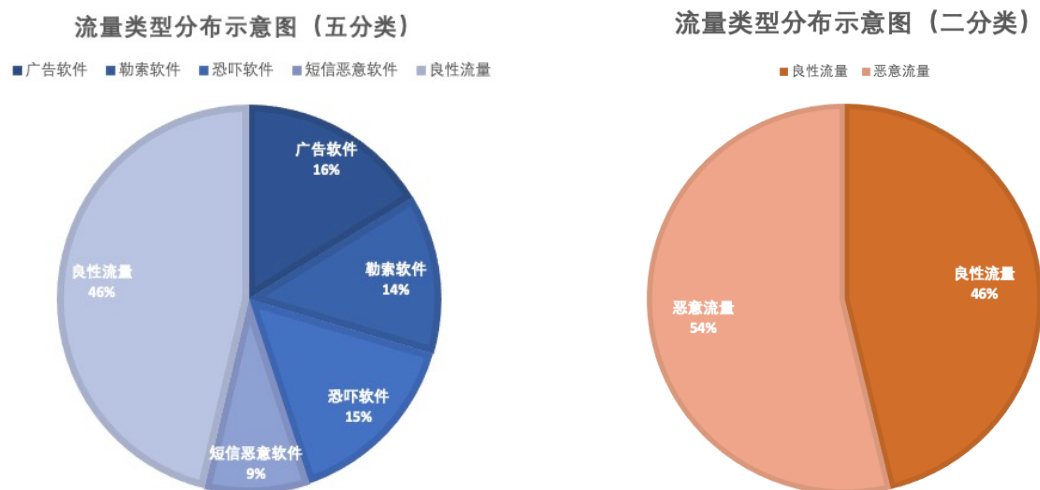


图 2 数据集流量类型分布示意图

## 2.2 PCAP 特征提取与双向流匹配

公开数据集中的双向流信息以 PCAP 格式存储，在利用其与 CSV 文件结合并进行特征提取与模型训练之前，需要进行以下步骤：

- PCAP 文件的特征提取
- CSV 文件的双向流筛选
- PCAP 对 CSV 的匹配与合并

### 2.2.1 PCAP 文件的特征提取

这一步将实现把 PCAP 文件中的流量特征提取到 CSV 文件中。Python 处理 PCAP 文件使用 Scapy 库，其中的 rdpcap() 函数用于提取一个 .pcap 文件中的所有数据包信息储存在 PacketList 形式的数据结构中；提取之后可以通过 show() 函数显示数据包信息，还可通过索引协议层名 ('IP', 'TCP', 'tls' 等) 与 fields() 函数调用具体的数据信息。

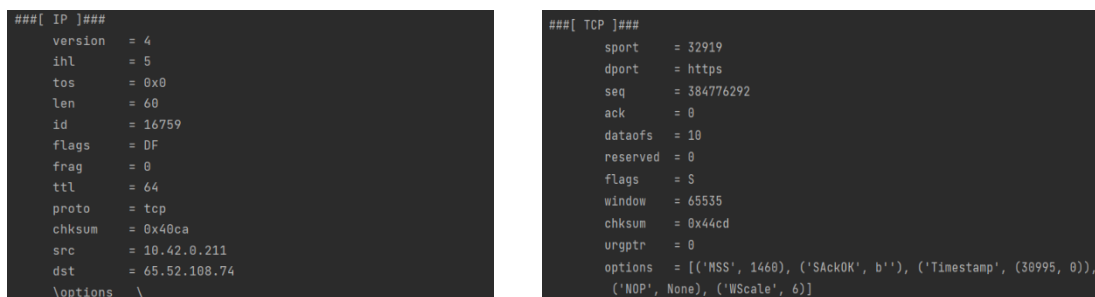


图 3 PCAP 文件读取结果

对于原始的 PCAP 文件，需要进行一定的清理与筛选处理：

- (1) 需要过滤无关网络报文，如 ARP 或 ICMP 报文；
- (2) 需要删除重复的，损坏的，不必要的，不完全捕获的流量流或信息；
- (3) 每个 PCAP 包选前 5 条数据包作为处理样本。

对于明文数据包和加密数据包还需要分别提取不同的特征：

- (1) 对于加密的数据流（源/目的端口号为 443），主要提取：

- TLS 协议类型（TLS.type）
- 列出的密码套件列表（Cipher Suites）
- 支持的扩展列表（Extensions）
- 密钥交换算法中选用的公钥长度（Public Key Length）

- (2) 对于未加密的数据流，主要提取：

- 报文头部信息（Source/Destination IP/Port、Timestamp 等）
- 流入和流出 HTTP 字段的种类
- HTTP 标志位（flags）
- HTTP 内容类型（Content-Type）

其余可用于分类的特征字段在 CSV 文件中已经存在，在 PCAP 文件中不再需要提取。

提取到的特征以行为单位存储在元组中，然后调用 csv 库，将字段名和特征行用 writerows 写入到 csv 文件中，完成 PCAP 到 CSV 文件的转换。完成 PCAP 提取后的样例字段如下：

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9722 entries, 0 to 9721
Data columns (total 10 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Source IP             9722 non-null   object
1   Source Port           9722 non-null   int64
2   Destination IP        9722 non-null   object
3   Destination Port      9722 non-null   int64
4   Timestamp             9722 non-null   object
5   flags                 9722 non-null   object
6   HTTP-type             9722 non-null   object
7   TLS-type              8717 non-null   object
8   Cipher Suites Len     8717 non-null   float64
9   Extensions Len        8716 non-null   float64
dtypes: float64(2), int64(2), object(6)
```

图 4 PCAP 文件提取的特征值

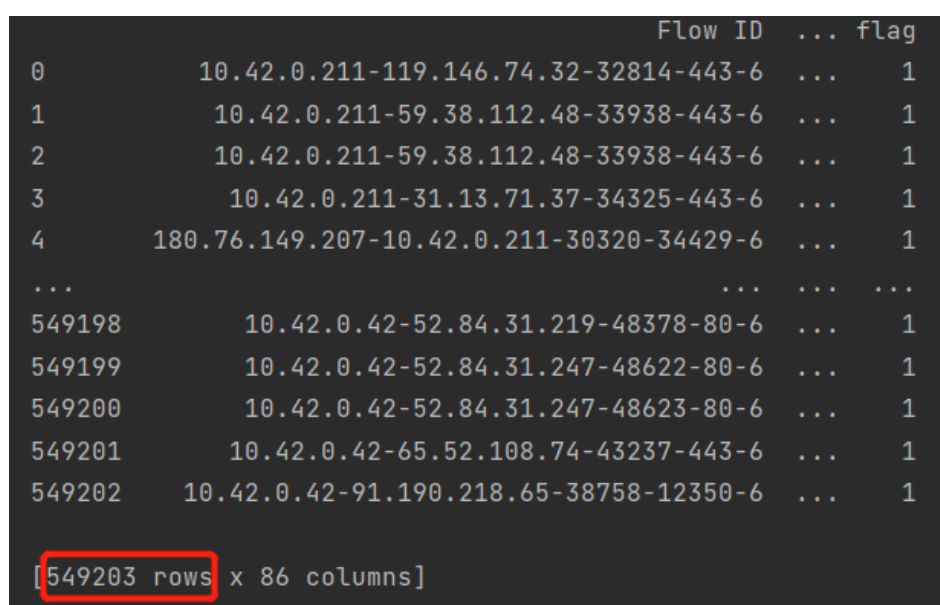


### 2.2.2 CSV 文件的双向流筛选

CSV 中存在大量重复的、单向的干扰数据，因此在 CSV 文件与 PCAP 文件进行双向流匹配前，需要先过滤部分 CSV 文件中的数据。这一步的难点在于如何删除单向数据并保留双向流数据。

由于数据量极大，用一般的数据结构对每条数据都进行双向匹配需要 $O(n^2)$ 级的时间复杂度，运算耗时极大。因此，我们选择用 pandas 库中的 DataFrame 结构来高效处理 CSV 文件。用 read\_csv 读入数据后，遍历每一个 CSV 文件中的数据，用 loc() 函数和逻辑表达式检测当前数据的源/目的 IP 与端口号颠倒后是否仍存在于读入的 DataFrame 结构中，若存在，则保留该行，添加标志位为 1；否则，标志位设为 0。最后过滤掉标志位为 0 的行（使用 drop 函数）即得到经过双向流筛选的 CSV 文件。

最终过滤掉了大量单向流数据，从原来的 261 万条数据精简到约 55 万条：



	Flow ID	...	flag
0	10.42.0.211-119.146.74.32-32814-443-6	...	1
1	10.42.0.211-59.38.112.48-33938-443-6	...	1
2	10.42.0.211-59.38.112.48-33938-443-6	...	1
3	10.42.0.211-31.13.71.37-34325-443-6	...	1
4	180.76.149.207-10.42.0.211-30320-34429-6	...	1
...	...	...	...
549198	10.42.0.42-52.84.31.219-48378-80-6	...	1
549199	10.42.0.42-52.84.31.247-48622-80-6	...	1
549200	10.42.0.42-52.84.31.247-48623-80-6	...	1
549201	10.42.0.42-65.52.108.74-43237-443-6	...	1
549202	10.42.0.42-91.190.218.65-38758-12350-6	...	1
[549203 rows x 86 columns]			

图 5 双向筛选后的 CSV 文件数据量

### 2.2.3 PCAP 对 CSV 的匹配与合并

经过双向流筛选的 CSV 文件中仍有大量实际不存在的无效数据，即 CSV 的来源中并未提供原始流量文件（PCAP 文件），CSV 文件与 PCAP 文件的关系如图 6 所示。因此，还需要利用提取特征后的 PCAP 数据与双向流筛选后的 CSV 文件进行匹配与合并，得到最终需要的数据。

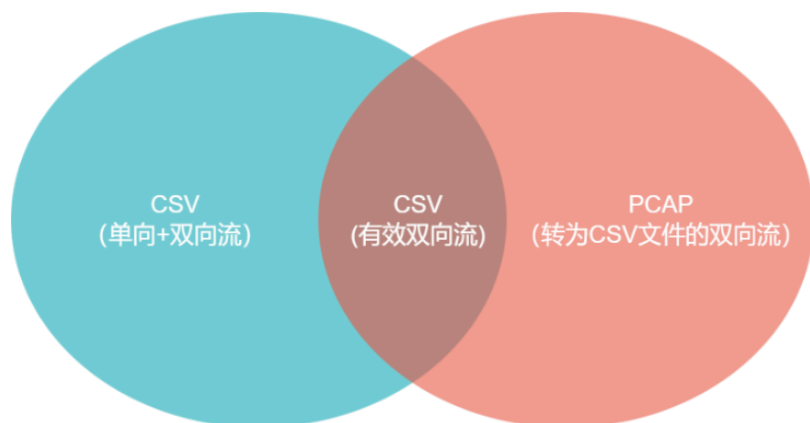


图 6 CSV 文件与 PCAP 文件的关系

在进行匹配与合并前，先对两部分文件根据['Source IP', 'Source Port', 'Destination IP', 'Destination Port']的优先级进行排序，以便处理过程中更高效地进行查找操作。

匹配方法与 2.2.2 中的类似，用 pandas 库读入 CSV 文件后，遍历其中的每行数据，查询当数据的源/目的 IP/端口在以 CSV 格式保存的 PCAP 文件中是否存在对应行，若存在，则保留该行，添加标志位为 1；否则，标志位设为 0。

合并过程需要建立在匹配完成的基础上，完成匹配后，再将 PCAP 文件中独有的特征添加到当前 CSV 数据的新列之中。最后过滤掉标志位为 0 的行，得到最终需要的数据文件。最终得到的数据量大约在 47w:

	Flow ID	...	Label
0	183.61.27.21-10.42.0.211-443-32810-6	...	ADWARE_DOWGIN
1	183.56.172.96-10.42.0.211-80-33000-6	...	ADWARE_DOWGIN
2	10.42.0.211-65.52.108.74-33397-443-6	...	ADWARE_DOWGIN
3	172.217.0.234-10.42.0.211-443-33496-6	...	ADWARE_DOWGIN
4	172.217.0.234-10.42.0.211-443-33496-6	...	ADWARE_DOWGIN
...	...	...	...
469824	10.42.0.42-52.84.31.219-48378-80-6	...	SMSMALWARE_ZZONE
469825	10.42.0.42-52.84.31.247-48622-80-6	...	SMSMALWARE_ZZONE
469826	10.42.0.42-52.84.31.247-48623-80-6	...	SMSMALWARE_ZZONE
469827	10.42.0.42-65.52.108.74-43237-443-6	...	SMSMALWARE_ZZONE
469828	10.42.0.42-91.190.218.65-38758-12350-6	...	SMSMALWARE_ZZONE
[469829 rows x 8 columns]			

图 7 有效双向流的数据量

## 2.3 流量特征分析

为了分类模型具有更好的预测效果，需要选择在恶意流量与良性流量之间具有显著差异的特征；为了使得良性与恶意流量之间的差异更显著并可视化，在本节进行流量特征分析。

### 2.3.1 数据抽样与处理

在可视化部分，对所有特征进行随机抽样，选择其中六个特征进行分析。抽样选择后的特征集合是['Packet Length Std', 'Packet Length Variance', 'Fwd Packet Length Mean', 'act\_data\_pkt\_fwd', 'Total Length of Fwd Packets', 'Subflow Fwd Packets']。

在选择特征之后，利用散点图观察数据分布情况。由于数据量较大，需要再次对数据进行随机抽样，为使图表中散点较为清晰，在实际中对一个特征下抽样选择七百个良性与恶意流量的数据。如图 8 所示构造散点图，其中良性流量散点颜色为红色，恶意流量为蓝色。

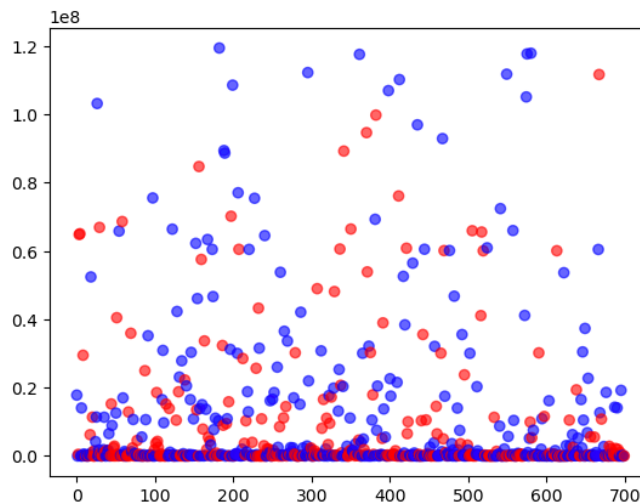


图 8 单一特征的数据抽样检测的散点图

从图中分析可知，良性与恶意流量的分布较为复杂与密集，直接分析难以直接展示差异性，连续数据具有较明显的长尾效应，即大部分数据集中在某一区间，极少数据分布在其他位置。因此需要先对数据进行标准化。在标准化方法选择时，选择 z-score 方法或者 0-1 标准化法需要进一步考虑。

在统计学中，通常可以对数据分布范围较大的数据，以及变量尺度较大的数据集取对数处理。取对数后不会改变数据的性质和相关关系，但压缩了变量尺度，使得数据更加平稳，构造的图表中展示效果更好。并且选择基底为自然指数  $e$  的对数函数，该函数为递增函数不会改变数据之间的大小关系，更为常用。

因此为了对数据进行对数处理，我们选择分布范围大于 0 的 0-1 标准化方法，具体操作为：对于序列数据  $x$ ，变换后的数据为  $x'$ ，则有：

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

同时数据经过标准化后存在较小的数据，在计算过程中会被计算机识别为 0，在取对数时出现数学性报错。因此对数据集整体加一处理，改变其在数轴上的位置而不改变分布情况。对数处理后再进行可视化。

### 2.3.2 箱型图分析

箱型图（Boxplot）是利用数据中的五个统计量：最小值、上四分位数、中位数、下四分位数与最大值来描述数据的一种统计图。它能够直观的显示数据分布的离散情况以及数据的对称性。常见的箱型图结构如图 9 所示。

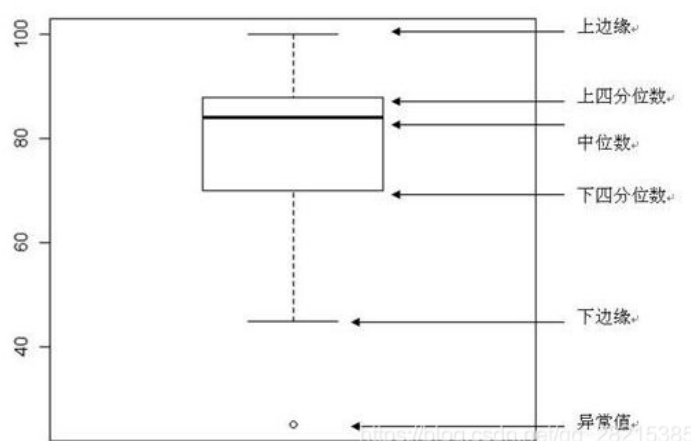


图 9 常见的箱型图结构

利用不同数据集的不同箱型，可以直观的展示数据分布差异，从而展示良性恶意流量特征的差异性。为了更好的描述数据批的分布形状，这里结合了均值来分析。对上述抽样的六种特征对应的经过标准化处理的数据进行箱型图构造，得到图 10 所示。

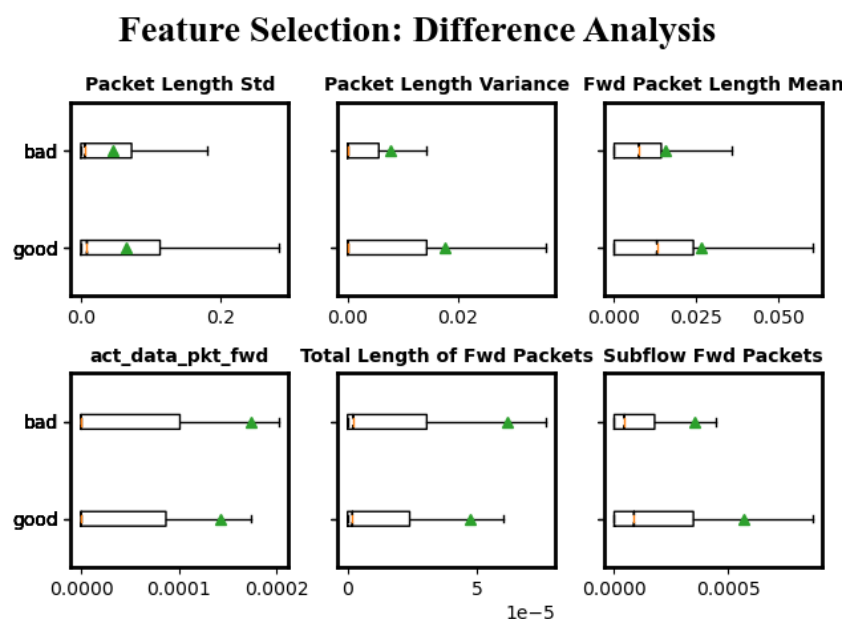


图 10 经过标准化以及取对数处理后的箱型图

上述六种特征的箱型图中，绿色正三角表示数据集合的均值，红色直线表示中位数。可以看到，对于有差异性的特征，数据集合的均值、分布范围有着较为明显的不同，利用这些特点可以选择相应特征用于训练分类模型。由于特征较多，不能直接构造所有的箱型图直接肉眼分析，所以需要进一步的方法进行特征选择。

## 2.4 特征选择

### 2.4.1 基于语义的特征选择

针对采集到的特征（包括 CSV 和 PCAP），通过分析可知，流 ID (Flow\_ID)、源 IP 地址 (Source IP)、目的 IP 地址 (Destination IP)、时间戳 (Timestamp) 在语义上与恶意流量不相关，属于干扰特征，故予以剔除。

### 2.4.2 基于标准差的特征选择

标准差用于描述一组序列数据的波动情况。针对采集到的特征，对所有特征向量求标准差，发现 ECE Flag Count、Fwd URG Flags、Bwd URG Flags 等特征标准差为 0，说明在整体数据集中，这些特征没有波动。换言之，标准差为 0 的特征在恶意流量和良性流量中不具有可区分性，属于无效特征，故予以剔除。

### 2.4.3 基于非参数统计的特征选择

经过上述筛选后，已剔除部分特征，下面寻找更有效、更可解释的特征选择方法。非参数统计（Nonparametric Statistics）是一种统计学方法，用于对数据进行分析 and 推断，而不对数据分布进行具体的假设。

非参数统计在以下情况下特别适用：

1. 数据分布未知或不符合特定的分布假设：当我们无法确定数据的概率分布形式，或者数据不满足正态分布或其他特定分布假设时，非参数统计提供了一种无需依赖特定分布的方法。

2. 数据包含离群值或异常值：离群值的存在可能会影响参数统计方法的可靠性。非参数统计方法基于数据的秩次或排序，对离群值相对较为鲁棒，能够更好地应对这种情况。

3. 非线性关系或异方差性：某些参数统计方法假设数据间存在线性关系或异方差性。如果数据具有非线性关系或异方差性，非参数统计方法能够更好地捕捉数据之间的复杂关系。

对于数据集中离散的数据，正如网络流量一般不具有较明显的数据分布的情况，非参数统计有着比较好的统计性能。

我们尝试了三种检验方法，分别是 MannWhitney U 检测法、K-S 检测法以及 Kruskal-Wallis H 检验法。

在 MannWhitney U 检测法中，我们将每一个特征所对应的所有数据按照良性流量与恶意流量分别聚合成两类，对于这两类样本，进行该检测，分析两种样本的中位数是否在同一水平，也就是两类样本的差异性是否明显。当结果大于标准值时，认为在当前显著性水平下可以认为两种样本没有显著差异。且这个标准值越低表示对差异描述越严格。K-S 检测与 Kruskal- Wallis H 检测法原理与上述方法类似，也是用来描述两种样本的分布情况是否相同（是否来自同一总体）和差异情况的方法，具体不再重复说明。在实现时，利用 `scipy` 模块中的 `mannwhitneyu` 模块、`ks_2samp` 模块以及 `kruskal` 模块实现。设置标准值为 0.01，即检测结果 `pvalue` 的值小于 0.01 时认为其数据在两类样本中有显著差异，并把有显著性差异的特征输出，结果如下图所示。

```
['Source Port', 'Destination Port', 'Protocol', 'Flow Duration', 'Total Fwd Packets', 'Total Backward Packets', 'Total Length of Fwd Packets', 'Total Length of Bwd Packets', 'Fwd Packet Length Max', 'Fwd Packet Length Min', 'Fwd Packet Length Mean', 'Fwd Packet Length Std', 'Bwd Packet Length Max', 'Bwd Packet Length Min', 'Bwd Packet Length Mean', 'Bwd Packet Length Std', 'Flow Packets/s', 'Flow IAT Mean', 'Flow IAT Std', 'Flow IAT Max', 'Flow IAT Min', 'Fwd IAT Total', 'Fwd IAT Mean', 'Fwd IAT Std', 'Fwd IAT Max', 'Fwd IAT Min', 'Bwd IAT Total', 'Bwd IAT Mean', 'Bwd IAT Std', 'Bwd IAT Max', 'Bwd IAT Min', 'Fwd PSH Flag Count', 'Fwd Header Length', 'Bwd Header Length', 'Fwd Packets/s', 'Bwd Packets/s', 'Min Packet Length', 'Max Packet Length', 'Packet Length Mean', 'Packet Length Std', 'Packet Length Variance', 'FIN Flag Count', 'SYN Flag Count', 'PSH Flag Count', 'ACK Flag Count', 'URG Flag Count', 'Down/Up Ratio', 'Average Packet Size', 'Avg Fwd Segment Size', 'Avg Bwd Segment Size', 'Fwd Header Length.1', 'Subflow Fwd Packets', 'Subflow Fwd Bytes', 'Subflow Bwd Packets', 'Subflow Bwd Bytes', 'Init_Win_bytes_forward', 'Init_Win_bytes_backward', 'act_data_pkt_fwd', 'min_seg_size_forward', 'Active Mean', 'Active Std', 'Active Max', 'Active Min', 'Idle Mean', 'Idle Std', 'Idle Max', 'Idle Min', 'Label_1']
```

图 11 MannWhitney U 检测输出结果

```
['Source Port', 'Destination Port', 'Protocol', 'Flow Duration', 'Total Fwd Packets', 'Total Backward Packets', 'Total Length of Fwd Packets', 'Total Length of Bwd Packets', 'Fwd Packet Length Max', 'Fwd Packet Length Min', 'Fwd Packet Length Mean', 'Fwd Packet Length Std', 'Bwd Packet Length Max', 'Bwd Packet Length Min', 'Bwd Packet Length Mean', 'Bwd Packet Length Std', 'Flow Bytes/s', 'Flow Packets/s', 'Flow IAT Mean', 'Flow IAT Std', 'Flow IAT Max', 'Flow IAT Min', 'Fwd IAT Total', 'Fwd IAT Mean', 'Fwd IAT Std', 'Fwd IAT Max', 'Fwd IAT Min', 'Bwd IAT Total', 'Bwd IAT Mean', 'Bwd IAT Std', 'Bwd IAT Max', 'Bwd IAT Min', 'Fwd PSH Flags', 'Fwd Header Length', 'Bwd Header Length', 'Fwd Packets/s', 'Bwd Packets/s', 'Min Packet Length', 'Max Packet Length', 'Packet Length Mean', 'Packet Length Std', 'Packet Length Variance', 'FIN Flag Count', 'SYN Flag Count', 'PSH Flag Count', 'ACK Flag Count', 'URG Flag Count', 'Down/Up Ratio', 'Average Packet Size', 'Avg Fwd Segment Size', 'Avg Bwd Segment Size', 'Fwd Header Length.1', 'Subflow Fwd Packets', 'Subflow Fwd Bytes', 'Subflow Bwd Packets', 'Subflow Bwd Bytes', 'Init_Win_bytes_forward', 'Init_Win_bytes_backward', 'act_data_pkt_fwd', 'min_seg_size_forward', 'Active Mean', 'Active Std', 'Active Max', 'Active Min', 'Idle Mean', 'Idle Std', 'Idle Max', 'Idle Min', 'Label_1']
```

图 12 K-S 检测输出结果

```
['Source Port', 'Destination Port', 'Protocol', 'Flow Duration', 'Total Fwd Packets', 'Total Backward Packets', 'Total Length of Fwd Packets', 'Total Length of Bwd Packets', 'Fwd Packet Length Max', 'Fwd Packet Length Min', 'Fwd Packet Length Mean', 'Fwd Packet Length Std', 'Bwd Packet Length Max', 'Bwd Packet Length Min', 'Bwd Packet Length Mean', 'Bwd Packet Length Std', 'Flow Packets/s', 'Flow IAT Mean', 'Flow IAT Std', 'Flow IAT Max', 'Flow IAT Min', 'Fwd IAT Total', 'Fwd IAT Mean', 'Fwd IAT Std', 'Fwd IAT Max', 'Fwd IAT Min', 'Bwd IAT Total', 'Bwd IAT Mean', 'Bwd IAT Std', 'Bwd IAT Max', 'Bwd IAT Min', 'Fwd PSH Flag Count', 'Fwd Header Length', 'Bwd Header Length', 'Fwd Packets/s', 'Bwd Packets/s', 'Min Packet Length', 'Max Packet Length', 'Packet Length Mean', 'Packet Length Std', 'Packet Length Variance', 'FIN Flag Count', 'SYN Flag Count', 'PSH Flag Count', 'ACK Flag Count', 'URG Flag Count', 'Down/Up Ratio', 'Average Packet Size', 'Avg Fwd Segment Size', 'Avg Bwd Segment Size', 'Fwd Header Length.1', 'Subflow Fwd Packets', 'Subflow Fwd Bytes', 'Subflow Bwd Packets', 'Subflow Bwd Bytes', 'Init_Win_bytes_forward', 'Init_Win_bytes_backward', 'act_data_pkt_fwd', 'min_seg_size_forward', 'Active Mean', 'Active Std', 'Active Max', 'Active Min', 'Idle Mean', 'Idle Std', 'Idle Max', 'Idle Min', 'Label_1']
```

图 13 Kruskal-Wallis H 检验输出结果

## 2.5 特征预处理

### 2.5.1 数据清洗

主要关注字段格式、数据类型、异常值与空值处理等。

对于字段格式，观察到从 CSV 文件导入的字段存在两端空格的问题，这里使用 pandas 库的 strip()方法实现。

2	Source IP	object
3	Source Port	float64
4	Destination IP	object
5	Destination Port	float64
6	Protocol	float64
7	Timestamp	object
8	Flow Duration	float64
9	Total Fwd Packets	float64
10	Total Backward Packets	float64
11	Total Length of Fwd Packets	float64
12	Total Length of Bwd Packets	float64
13	Fwd Packet Length Max	float64

图 14 字段格式不统一



对于数据类型，观察到从 CSV 文件导入的 Flow IAT Min、Packet Length Std、CWE Flag Count、Down/Up Ratio、Fwd Avg Bytes/Bulk 等字段出现格式错误（浮点数类型被识别为字符串类型），以及 Source Port、Destination Port、Protocol（整型被识别成浮点数类型），应该人工予以矫正。数据类型转换可以使用 pandas 库中的 `astype()` 方法实现。

经转换后，数据可以分为离散型、连续型和字符串型三类，离散型主要包括 Source Port、Destination Port、Protocol，剩余特征均为连续型，分类标签 Label 为字符串型。

51	SYN Flag Count	float64
52	RST Flag Count	float64
53	PSH Flag Count	float64
54	ACK Flag Count	float64
55	URG Flag Count	float64
56	CWE Flag Count	object
57	ECE Flag Count	float64
58	Down/Up Ratio	object
59	Average Packet Size	float64
60	Avg Fwd Segment Size	float64
61	Avg Bwd Segment Size	float64
62	Fwd Header Length 1	float64
63	Fwd Avg Bytes/Bulk	object
64	Fwd Avg Packets/Bulk	float64

图 15 数据类型错误

对于异常值，观察到有部分特征的浮点数过大，可能存在越界的危险，这里使用空值 NaN 对无穷大（Inf/-Inf）进行替换。异常值处理可使用 pandas 库中的 `replace()` 方法实现。

对于空值，可以使用 pandas 库的 `isnull()` 方法判断是否为空，使用 `dropna()` 方法进行剔除。本项目中空值数量较少，这里采取剔除的方式处理空值。

# 查看空值	
<code>df.isnull().sum()</code>	
Flow ID	4
Source IP	0
Source Port	0
Destination IP	0
Destination Port	0
..	
Idle Mean	11
Idle Std	11
Idle Max	11
Idle Min	11
Label	11

图 16 空值处理



### 2.5.2 数据编码

对源数据集中的五分类标签（Label）进行条件判断，得到二分类使用的标签（Label\_1）。根据前文描述，本实验共使用五种流量，枚举值如下：

- Benign：良性流量
- Adware：广告软件
- Ransomware：勒索软件
- Scareware：恐吓软件
- SMS Malware：短信恶意软件

其中，仅有 Benign 为良性流量，其余均为恶意流量。为方便数据处理，减少数据冗余，良性流量使用整数 0 编码，恶意流量使用整数 1 编码。

### 2.5.3 数据标准化

通过对相关特征的数据进行分析，可以发现连续数据具有较明显的长尾效应，即大部分数据集中在某一区间，极少的数据分布在其他位置。为了消除数据长尾对特征学习造成的影响，这里使用 z-score 方式对数据进行标准化。

z-score 标准化方法通过对数据进行伸缩变换，使其映射到标准正态分布的空间上，具体操作为：对于序列数据 $x$ ，变换后的数据为 $x'$ ，则有：

$$x' = \frac{x - \text{mean}(x)}{\text{std}(x)}$$

其中， $\text{mean}(x)$ 和 $\text{std}(x)$ 为序列 $x$ 的均值和标准差。图 17、18 展示了对于部分特征，数据标准化对数据长尾的缓解。

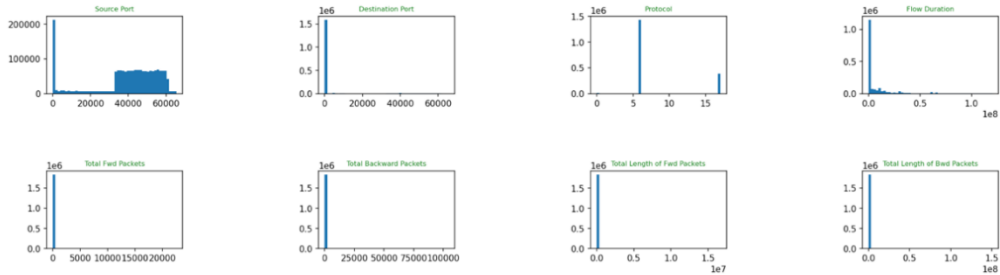


图 17 标准化前的数据分布直方图

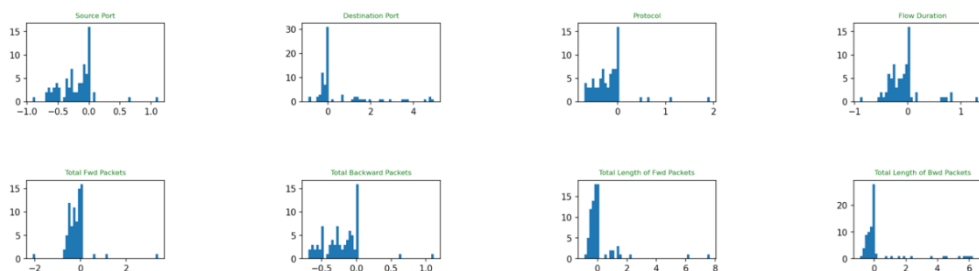


图 18 标准化后的数据分布直方图

## 2.6 分类模型简介

### 2.6.1 逻辑回归（Logistic Regression）

逻辑回归是一种用于分类问题的机器学习算法，其核心思想是通过建立一个二分类模型，将输入特征映射到输出标签。在本次实验中，需要评估一条网络流量是恶性或是良性的（即标签），逻辑回归方法正好可用于评估一个二元变量的概率。具体而言，逻辑回归模型由两部分组成，特征函数和分类函数，前者用于将输入特征映射到高维空间中的点，而后者则将这些点映射到输出标签上。在训练过程中，逻辑回归将输入特征与权重相乘，使用最大似然估计来确定权重的值，以最大化正确分类的概率，从而实现分类任务。

具体实现上，逻辑回归可以通过调用 Python 机器学习库 `scikit-learn` 中的 `LogisticRegression` 模块实现。本项目实例化了一个逻辑回归模型 `lr`，对流量数据进行分类。

### 2.6.2 决策树（Decision Tree）

决策树是一种基于树形结构的机器学习算法，同样可以用于数据分类和回归问题，将输入特征映射到输出标签，从而实现分类任务。该算法基于一个贪心策略，即从已知数据集中的每个样本开始，选择最符合条件的特征或属性，并将它们划分到不同的子节点中，直到所有的样本都被划分到不同的子节点中，或者无法再划分为止。每个节点表示一个特征，每个分支表示该特征的取值，最终的叶子节点表示数据的分类结果。决策树易于理解，可以处理离散和连续变量，对缺失数据有较好的容错性。但是，决策树容易过拟合，需要进行剪枝操作。

具体实现上，决策树可以调用 Python 机器学习库 `scikit-learn` 的子库 `tree` 中的 `DecisionTreeClassifier` 模块实现。本项目实例化了一个决策树模型 `clf`，对流量数据进行分类。

### 2.6.3 随机森林 (Random Forest)

随机森林是一种基于决策树的集成学习算法，用于数据分类和回归问题，其核心思想是将决策树随机化，生成一组决策树，并使用这些决策树进行分类。在随机森林中，每个决策树的输出是另一个决策树的输出的加权平均。这个加权平均函数通常称为森林加权平均函数，它使用每个决策树的分类结果来计算出一个新的分类结果。具体而言，随机森林包括随机抽样、特征随机选择、构建决策树，再进行不断重复构建多棵决策树，最后将所有决策树的结果进行集成，得到最后的分类结果。

具体实现上，随机森林算法可以调用 Python 机器学习库 `scikit-learn` 的子库 `ensemble` 中的 `RandomForestClassifier` 模块实现。本项目实例化了一个随机森林模型 `rfc`，对流量数据进行分类。

### 2.6.4 卷积神经网络 (CNN)

卷积神经网络是一种深度学习模型，可用于图像识别、分割和分类等计算机任务。具体而言，CNN 模型第一步对输入数据进行归一化预处理，接下来再进行卷积操作，提取数据中的局部特征。在卷积操作之后，模型通常会使用池化操作来减少模型的参数数量和计算量，同时保留最显著的特征。最后，模型会对输入数据进行全连接层操作，将特征映射到类别标签上，全连接层通常包含多个神经元，每个神经元对应一个类别标签。重复上述过程，通过不断地卷积、池化和全连接操作，CNN 模型可以逐渐提取输入数据中的高级特征，从而提高模型的分类能力。

具体实现上，CNN 可以调用 Python 深度学习库 `tensorflow` 中的 `Conv1D` 等模块实现。本项目实例化了一个 CNN 模型 `cnn_model`，对流量数据进行分类，并将其导出为 `cnn_model.h5`。

### 三. 结果测试

#### 3.1 分类结果展示

针对清洗后的 261 万条 CSV 流量数据,与 PCAP 提取出的流量进行匹配后,作为总的数据集进行分类。分别使用逻辑回归、决策树、随机森林和卷积神经网络算法进行训练,得到的训练结果见下表。(70%作为训练集,30%为测试集)

表 1 训练结果汇总表

	逻辑回归	决策树	随机森林	卷积神经网络
准确率	0.541	0.659	0.686	0.777
F1-score	0.696	0.685	0.712	0.771
AUC	0.527	0.657	0.761	0.801

##### 3.1.1 准确率 (Accuracy)

为了评价一个分类模型的性能,这里引入混淆矩阵。

	预测为正	预测为负
实际为正	TP	FN
实际为负	FP	TN

图 19 混淆矩阵

混淆矩阵展示了 TP (True Positive)、TN(True Negative)、FP (False Positive) 和 FN (False Negative) 等指标的表现。其中,FP 表示负类被判断为正类的样本比例,FN 表示正类被判断为负类的表现,这两个指标均代表判断错误。由此,准确率计算公式为:

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

准确率可以衡量一个模型的分类能力。下图展示了使用四种不同分类算法计算的准确率。其中,逻辑回归算法表现最差,准确率只有 54.1%;CNN 算法表现

最好，准确率达到 77.7%。

```
[40]: score = lr.score(x_test, y_test)
      print(score)
0.5413148967016111
```

图 20 逻辑回归算法准确率

```
[50]: score_c = clf.score(x_test, y_test)
      print(score_c)
0.6593567307337072
```

图 21 决策树算法准确率

```
[46]: score_r = rfc.score(x_test, y_test)
      score_r
[46]: 0.6868830091762754
```

图 22 随机森林算法准确率

```
938/938 [=====] - 2s 2ms/step - loss: 0.5619 - accuracy: 0.7768
938/938 [=====] - 2s 2ms/step
Loss: 0.5619326829910278
Accuracv: 0.7767999768257141
```

图 23 CNN 算法准确率

### 3.1.2 F1-Score

为了更好的评定模型，避免出现因数据集不平衡、数据过拟合等原因出现的错误，机器学习又引入了精确率（Precision）和召回率（Recall）等概念。其中，精确率用于描述，所有预测为 1 的样本中，有多少正类被正确分类，精确率的公式为：

$$Precision = \frac{TP}{TP + FP}$$

召回率则用来描述在所有真实值为 1 的样本中，有多少正类被选择出来。召回率的公式为：

$$Recall = \frac{TP}{TP + FN}$$

F1-Score 是综合了精确率和召回率的指标，是精确率和召回率的加权平均。

F1-Score 的公式为:

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

下图展示了四种算法不同的 F1-Score，其中决策树具有最低的 F1-Score，仅有 0.685；CNN 具有最高的 F1-Score，达到了 0.771。

```
[60]: f1_lr = f1_score(y_test, y_pred)
```

```
[60]: 0.6961855378682721
```

图 24 逻辑回归算法的 F1-Score

```
[62]: f1_clf = f1_score(y_test, y_pred)
```

```
[62]: 0.6859252420536059
```

图 25 决策树算法的 F1-Score

```
[64]: f1_rfc = f1_score(y_test, y_pred)
```

```
[64]: 0.7115563358463287
```

图 26 随机森林算法的 F1-Score

```
Precision: tf.Tensor(0.8490382, shape=(), dtype=float32)
Recall: tf.Tensor(0.70680594, shape=(), dtype=float32)
F1 Score: tf.Tensor(0.7714208, shape=(), dtype=float32)
```

图 27 CNN 算法的 F1-Score

### 3.1.2 ROC 曲线与 AUC

ROC 曲线，即接收者操作特征曲线（receiver operating characteristic curve），是反映敏感性和特异性连续变量的综合指标，ROC 曲线上每个点反映着对同一信号刺激的感受性。

AUC (Area Under Curve) 被定义为 ROC 曲线下的面积，显然这个面积的数值不会大于 1。又由于 ROC 曲线一般都处于  $y = x$  这条直线的上方，所以 AUC

的取值范围一般在 0.5 和 1 之间。使用 AUC 值作为评价标准是因为很多时候 ROC 曲线并不能清晰的说明哪个分类器的效果更好,而作为一个数值,对应 AUC 更大的分类器效果更好。下表给出从 AUC 判断分类器优劣的标准:

表 2 AUC 说明

AUC 区间	描述
$AUC = 1$	是完美分类器, 采用这个预测模型时, 存在至少一个阈值能得出完美预测。绝大多数预测的场合, 不存在完美分类器。
$0.5 < AUC < 1$	优于随机猜测。这个分类器(模型)妥善设定阈值的话, 能有预测价值。
$AUC = 0.5$	与随机猜测没有区别, 模型没有预测价值。
$AUC < 0.5$	比随机猜测还差; 但只要总是反预测而行, 就优于随机猜测。

AUC 还可以理解为, 给正负样本打分高低的概率。例如 0.7 的 AUC, 其含义可以大概理解为: 给定一个正样本和一个负样本, 在 70%的情况下, 模型对正样本的打分高于对负样本的打分。通常情况下, AUC 超过 0.7 的分类器已经算是效果比较好的分类器。

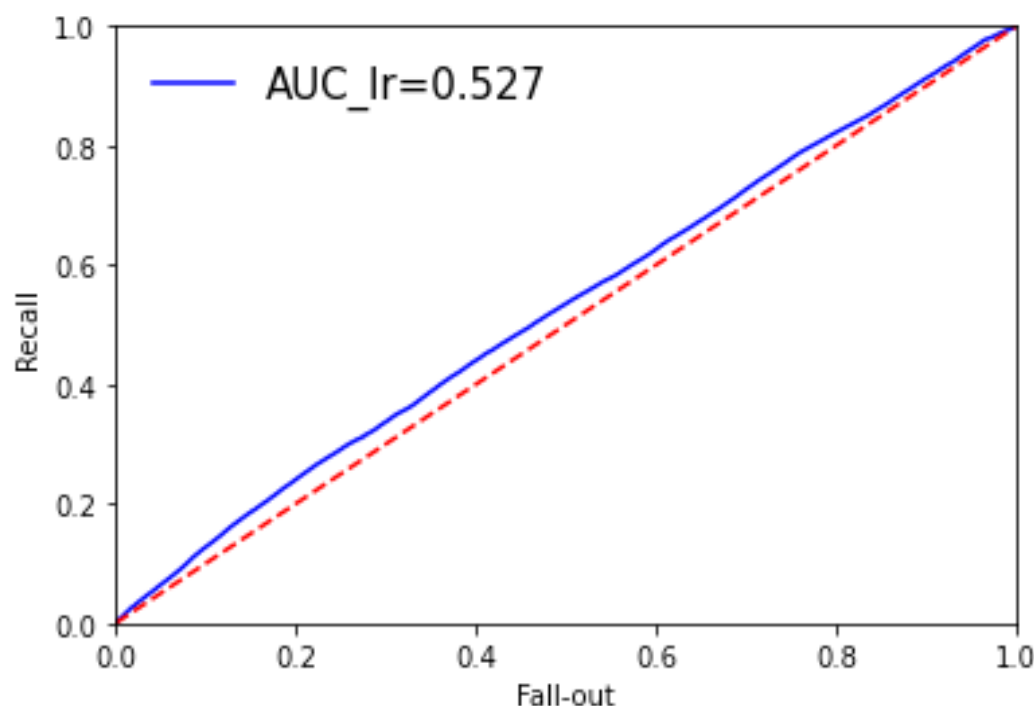
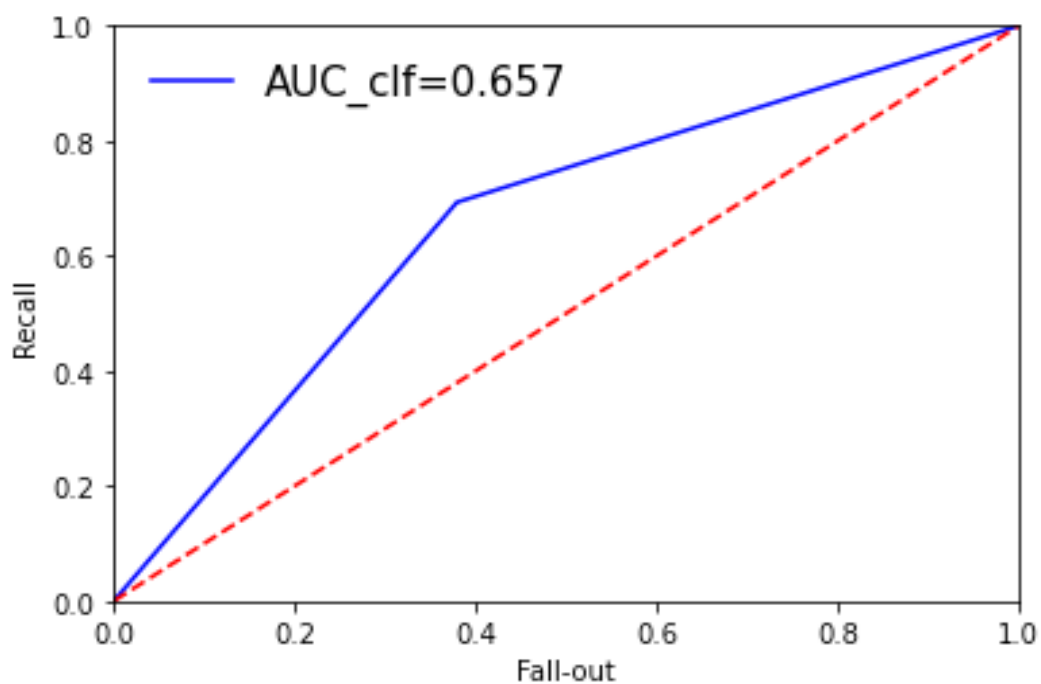


图 28 逻辑回归算法的 ROC 曲线



z

图 29 决策树算法的 ROC 曲线与 AUC

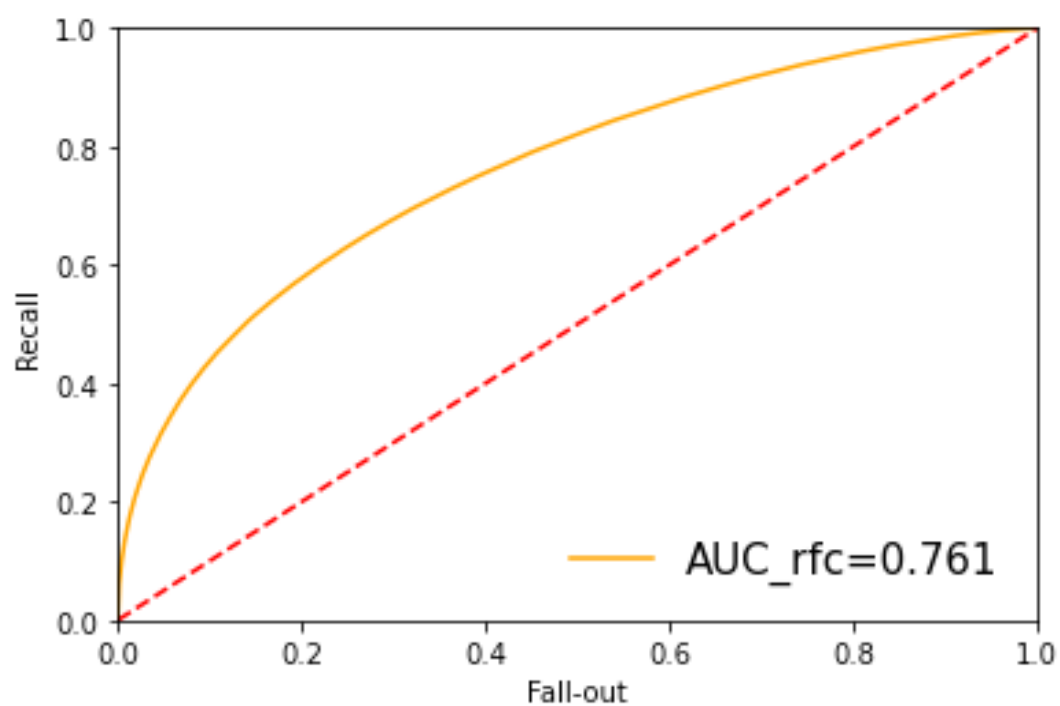


图 30 随机森林算法的 ROC 与 AUC 曲线



## 3.2 分类指标对比

下图给出四种不同的分类算法在准确率 (ACC)、F1-Score 和 AUC 的表现，可以发现 CNN 在解决流量分类问题上具有显著优势，准确率可以达到 77% 左右。

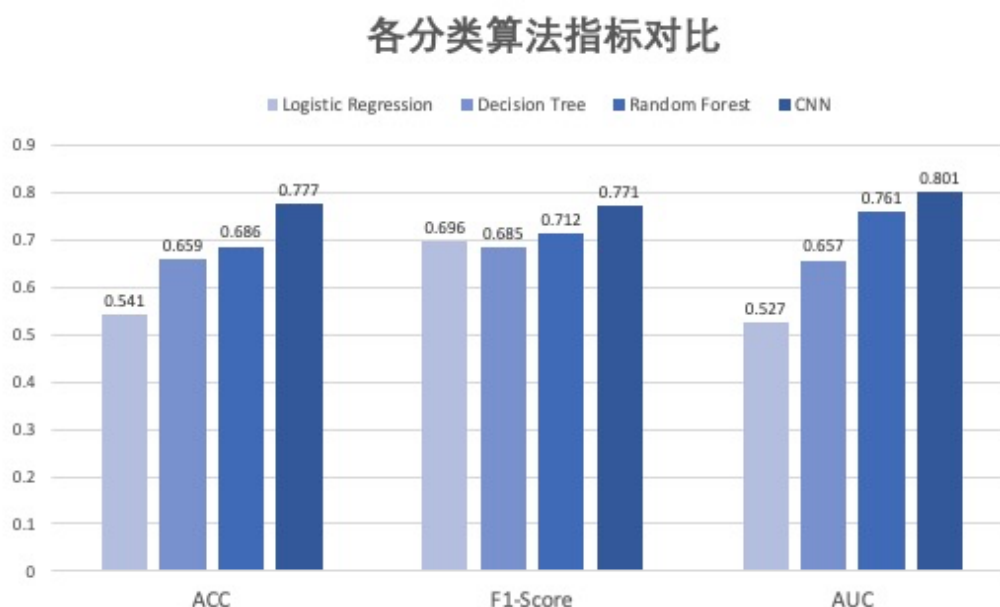


图 31 分类指标对比图

## 3.3 算法分析

我们最关心分类模型是否能正确分类恶意流量和良性流量，即模型的准确率高。从图 31 中，我们可以直观看到，逻辑回归、决策树、随机森林、卷积神经网络算法的准确率依次增高，这与该项目的数据集有关。

逻辑回归使用线性函数来拟合分类边界，简单易懂且易于实现，具有计算速度快、可解释性强等优点。但是，它对于异常值极为敏感，仅能处理线性可分问题，因此难以应对较复杂的应用场景。同时，它连续型特征的拟合能力不足，容易出现过拟合现象。因此，逻辑回归在处理离散型特征时准确率会很低。

决策树是一种树形结构的分类算法，可以将数据集分成小的子集，从而方便算法处理。但是，决策树虽然在本项目中较优于逻辑回归，但同样地，它在处理特征关联性比较强的数据时表现得不是太好，容易出现过拟合现象，导致准确率不尽人意。

随机森林是一种集成学习算法，它由多个决策树组成，可以处理较为复杂的问题，并且具有较好的鲁棒性。但是，对于连续型特征，随机森林拟合能力不足，会使准确率降低。相比较于其他分类方法，随机森林的分类准确率通常更高。另外，随机森林算法可处理大规模问题（即多样本单元、多变量），可处理训练集中有大量缺失值的数据，也可以应对变量多于样本单元的数据。可计算袋外预测误差、度量变量重要性也是随机森林的两个明显优势。

卷积神经网络是一种深度学习算法，它使用卷积操作和池化操作来处理数据。在卷积操作之后，模型通常会使用池化操作来减少模型的参数数量和计算量，同时保留最显著的特征。卷积神经网络对于数据具有很好的处理能力，并且可以处理连续型特征。

在本项目中，卷积神经网络模型能显著区分恶意流量和良性流量的特征，也即优秀的判断分类能力，因此在准确率上达到了令人满意的成绩。不同的场景有不同的需求，需要我们根据数据集选择最恰当的方法。

## 四. 创新点说明

### 4.1 数据特征提取标准多元

本项目的核心之一是进行流量数据的特征选择，因此在特征分析与选择上施加较为严苛的标准。一方面，对数据进行抽样可视化处理，又利用箱型图分析，以初步确定具有显著差异的特征集；另一方面，又基于语义、标准差、非参数统计等方法进一步进行选择，加强数据特征的合理程度，提高模型分类的准确性。

### 4.2 数据分类模型横纵对比

采用了较为常见的四种模型对恶意流量和良性流量进行分类，分别是逻辑回归、决策树、随机森林和卷积神经网络算法。针对分类结果，又设置准确率、F1-score、AUC 得分参数，利用图表可视化进一步直观看出不同算法之间的差异，最终得出卷积神经网络是最适用于此项目的算法模型。

## 五. 团队成员及分工

姓名	学号	班级	联系方式	具体分工
屈佳宁	19373474	203913	18810599394	担任队长，主要负责特征预处理、分类模型训练、结果测试与评价等部分，并完成代码书写、报告书写和 Demo 录制。
赵碧琪	20373669	203911	13835179240	主要负责流量特征数据处理、差异性分析以及流量特征提取部分，并完成代码书写与报告排版。
李浦旭	20373670	203913	18510333177	主要负责查找参考文献、实现 PCAP 文件的提取与双向流匹配，并书写相关代码和报告。
黄亮	20373953	203913	18090099847	主要负责项目背景调研、流程设计、算法原理学习，并书写相关报告。

## 六. 总结

网络流量庞大且错综复杂，本项目基于公开的大数据集，对数据的特征进行细致分析以及提取，对数据特征进行预处理，并采用四种不同的算法，包括逻辑回归、决策树、随机森林和卷积神经网络算法对数据流量进行分类，最终根据准确率、F1-score、AUC 等参数得分评判，得出卷积神经网络是一个判断恶意流量的行之有效的方法。本文方法在确保极大部分数据均使用的情况下，模型仍有较高的精度以及效率，在网络异常检测中具有重要的应用价值。

但是，本文仍有一些问题需要解决。一是特征选取影响到模型分类的准确度，其标准是否可以进一步优化；二是使用了公开的数据集，而没有具体深入到现实

随机网络中，其中恶意流量更加繁多复杂，需要有新的算法或工具完成分类。

## 七. 参考文献

- [1] 邓华伟,李喜旺. 基于深度学习的网络流量异常识别与检测[J].计算机系统应用,2023,32(02):274-280.
- [2] Zihao Wang, Kar-Wai Fok, Vrizlynn L. L. Thing. Machine Learning for Encrypted Malicious Traffic Detection: Approaches, Datasets and Comparative[J]//Study Computers & Security 2022/02.
- [3] 王勇,周慧怡,俸皓等.基于深度卷积神经网络的网络流量分类方法[J].通信学报,2018,39(01):14-23.
- [4] 包文博,沙乐天,曹晓梅.基于特征融合的恶意加密流量识别[J].计算机系统应用,2023,32(01):358-367.