## **5** 实验

5.1实验设置

为了验证本文算法的可行性，本文在表5.1所示的实验环境下进行实验。

表5.1 实验环境

|  |  |
| --- | --- |
| 类别 | 参数 |
| 处理器 | Intel(R) Core(TM) i5-6300 CPU |
| 显卡 | NVIDIA GeForce GTX 960M |
| Python3 | 3.7 |
| Tensorflow | 2.8.0 |
| Scikit-learn | 1.0.2 |

**参数设置**：在级联森林中每个层级包含4个随机森林和4个完全随机森林；每个森林通过3折交叉验证生成样本的类别分布向量。对于原始深度森林将每和森林设置100棵决策树。对改进深度森林第一层使用10棵决策树。随着层数的增加剩余训练样本数目不断减少，每个森林中的决策树会随着训练样本数目的减少而线性增加。若连续4层准确没有得到提升，则自动停止训练。

5.2 UNB-CIC Tor 数据集简介与处理

为了评估提出方法的有效性，本文使用了UNB（University of New Brunswick）发布的Tor流量有标签数据集来验证本文所提出的方法。研究人员首先在物理计算机中组成两个虚拟机：网关和工作站，如图5.2所示。由于所有Tor流量必须通过网关才能到达工作站，因此Tor流量变得透明。Tor流量数据包在网关处捕获，非Tor流量在通过网关的工作站处捕获。研究人员使用Wireshark 和 tcpdump 工具捕获流量，总共生成了22 GB的PCAP文件。

本文使用ISCXFlowMeter 应用程序来解析pcap数据包。按照具有相同<目的 IP，目的端口，源 IP，源端口，协议> 值的数据包定义为流。FlowMeter 生成双向流，以第一个数据包确定正向（从源到目的地）和反向（从目的地到源）方向，因此统计时间相关的特征也在正向和反向分别计算。请注意，TCP 流通常在连接断开时终止（由 FIN 数据包），而 UDP 流则由流超时终止。流超时值可以由单独的方案任意分配。

表5.2 Tor流量检测数据分布

|  |  |  |  |
| --- | --- | --- | --- |
| 流超时值 | Tor | nonTor | 总计 |
| 2s | 32901 | 162552 | 195453 |
| 4s | 17556 | 137155 | 154711 |
| 8s | 9713 | 102970 | 112683 |
| 16s | 5329 | 86495 | 91824 |
| 32s | 2924 | 69633 | 72557 |
| 64s | 1626 | 61012 | 62638 |
| 128 | 930 | 57655 | 58585 |

本文目标是从混合流量中检测出Tor流量，因此对数据集划分为Tor流量和非Tor流量。在本文中研究了几个流超时值，以确定流超时对最终结果的影响。特别是，我们将流的持续时间设置为2、4、8、32，64和128秒。表5.2显示了在不同流超时值下，对Tor流量行为检测提取的数据分布。

5.3 实验结果

本节实验使用流超时时间为2、4、8、32，64和128秒的经过预处理的数据集作为输入，按照2：8的比例划分了训练集和测试集，使用了准确率、精确率、F1-Score、召回率、训练时长、测试时长和内存开销作为模型性能的评价指标。使用原始深度森林模型和改进深度森林模型gcCS的Tor流量检测实验如表5.3所示。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 方法 | 准确率% | 精确率% | F1-Score% | 召回率% | 训练时长s | 测试时长s | 内存开销M | 流超时时间 |
| gcCS | 99.90 | 99.24 | 98.96 | 98.68 | 189.54 | 11.94 | 24.67 | 2s |
| gc | 99.90 | 99.62 | 99.20 | 98.78 | 692.11 | 29.06 | 25.14 | 2s |
| gcCS | 99.85 | 99.56 | 99.40 | 99.24 | 197.87 | 3.89 | 20.84 | 4s |
| gc | 99.87 | 99.67 | 99.48 | 99.3 | 467.00 | 21.25 | 25.55 | 4s |
| gcCS | 99.83 | 99.62 | 99.46 | 99.54 | 159.12 | 3.20 | 20.02 | 8s |
| gc | 99.85 | 99.68 | 99.59 | 99.51 | 378.96 | 20.94 | 23.125 | 8s |
| gcCS | 99.86 | 99.82 | 99.72 | 99.62 | 114.92 | 2.72 | 20.00 | 16s |
| gc | 99.87 | 99.82 | 99.68 | 99.75 | 291.86 | 32.57 | 22.60 | 16s |
| gcCS | 99.85 | 98.61 | 97.61 | 96.93 | 213.00 | 8.76 | 20.14 | 32s |
| gc | 99.89 | 98.96 | 98.37 | 97.78 | 259.00 | 29.43 | 23.09 | 32s |
| gcCS | 99.85 | 98.76 | 96.67 | 94.66 | 162.98 | 6.42 | 20.09 | 64s |
| gc | 99.85 | 98.46 | 96.68 | 94.96 | 148.86 | 17.75 | 21.13 | 64s |
| gcCS | 99.85 | 94.39 | 95.36 | 96.35 | 166.86 | 2.80 | 22.04 | 128s |
| gc | 99.90 | 96.88 | 96.88 | 96.88 | 141.00 | 24.55 | 22.18 | 128s |

由表5.3的实验结果可知随着流超时时间越来越大数据集越不平衡（数据集中Tor流量越来越少）导致精确率、F1-Score、召回率出现了下降。原始深度森林和改进深度森林模型在预测效果不相上下，但其只须花费更少训练时长、测试时长和内存。改进深度森林较原始深度森林模型用更少的计算资源得到了更好的预测性能。

5.4 对比实验

为了验证改进深度森林在Tor流量检测的优势，通过综合考虑选择流超时时间为16s的数据集和多种常见机器学习模型进行对比实验。本文选择了决策树、随机森林、支持向量机和深度神经网络四种机器学习方法进行对比。实验结果如表5.4所示。

表5.4 对比实验结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分类器 | 准确率% | 精确率% | F1-Score% | 召回率% |
| gcCS | 99.86 | 99.82 | 99.72 | 99.62 |
| SVM | 98.43 | 97.00 | 94.66 | 92.43 |
| RandomForest | 99.79 | 99.64 | 99.31 | 98.98 |
| DecisionTree | 99.69 | 99.07 | 98.99 | 98.9 |
| CNN |  |  |  |  |