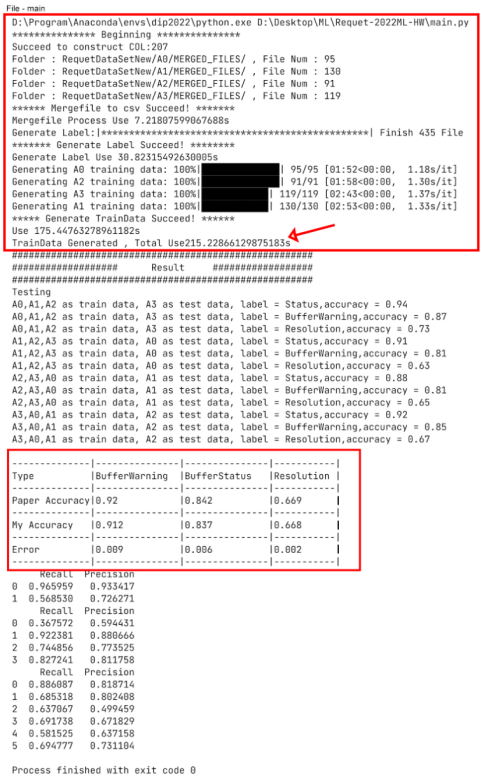
机器学习QoE论文复现说明

1 复现性能展示

1. 结果均来自运行项目中`main.py`文件控制台输出得到。
2. 生成训练数据集时间仅需要**215.23**s（测试电脑cpu：AMD Ryzen9 5900HX 16核，在python中我运用了**多进程**方式生成，可以大大缩减处理数据所需时间，同时我们采用了**其他方式解析pcap**文件，大大提高了解析速度），下图第一个红色方框所示。
3. 对A0/A1/A2/A3采用四折交叉验证，Accuracy如下图所示，偏差与论文数据均小于一个百分点。在下图第二个红色方框中，三个Label的Recall以及Precision也在下图展示。



```
File - main
D:\Program\Anaconda\envs\dip2022\python.exe D:\Desktop\ML\Requet-2022ML-HW\main.py
************ Beginning ***************
Succeed to construct COL:207
Folder : RequetDataSetNew/A0/MERGED_FILES/ , File Num : 95
Folder : RequetDataSetNew/A1/MERGED_FILES/ , File Num : 130
Folder : RequetDataSetNew/A2/MERGED_FILES/ , File Num : 91
Folder : RequetDataSetNew/A3/MERGED_FILES/ , File Num : 119
****** Mergefile to csv Succeed! *******
Mergefile Process Use 7.218075990676688s
Generate Label:|########################################| Finish 435 File
******* Generate Label Succeed! ********
Generate Label Use 30.823154926300055s
Generating A0 training data: 100%|██████████| 95/95 [01:52<00:00,  1.18s/it]
Generating A2 training data: 100%|██████████| 91/91 [01:58<00:00,  1.30s/it]
Generating A3 training data: 100%|██████████| 119/119 [02:43<00:00,  1.37s/it]
Generating A1 training data: 100%|██████████| 130/130 [02:53<00:00,  1.33s/it]
***** Generate TrainData Succeed! ******
Use 175.44763278961182s
TrainData Generated . Total Use215.22866129875183s
############################################################
##################      Result     ##################
############################################################
Testing
A0,A1,A2 as train data, A3 as test data, label = Status,accuracy = 0.94
A0,A1,A2 as train data, A3 as test data, label = BufferWarning,accuracy = 0.87
A0,A1,A2 as train data, A3 as test data, label = Resolution,accuracy = 0.73
A1,A2,A3 as train data, A0 as test data, label = Status,accuracy = 0.91
A1,A2,A3 as train data, A0 as test data, label = BufferWarning,accuracy = 0.81
A1,A2,A3 as train data, A0 as test data, label = Resolution,accuracy = 0.63
A2,A3,A0 as train data, A1 as test data, label = Status,accuracy = 0.88
A2,A3,A0 as train data, A1 as test data, label = BufferWarning,accuracy = 0.81
A2,A3,A0 as train data, A1 as test data, label = Resolution,accuracy = 0.65
A3,A0,A1 as train data, A2 as test data, label = Status,accuracy = 0.92
A3,A0,A1 as train data, A2 as test data, label = BufferWarning,accuracy = 0.85
A3,A0,A1 as train data, A2 as test data, label = Resolution,accuracy = 0.67

-------------|--------------|--------------|----------|
Type         |BufferWarning |BufferStatus  |Resolution|
-------------|--------------|--------------|----------|
Paper Accuracy|0.92          |0.842         |0.669     |
-------------|--------------|--------------|----------|
My Accuracy  |0.912         |0.837         |0.668     |
-------------|--------------|--------------|----------|
Error        |0.009         |0.006         |0.002     |
-------------|--------------|--------------|----------|
     Recall  Precision
0  0.965959   0.933417
1  0.568530   0.726271
     Recall  Precision
0  0.367572   0.594431
1  0.922381   0.880666
2  0.744856   0.773525
3  0.827241   0.811758
     Recall  Precision
0  0.886087   0.818714
1  0.685318   0.802408
2  0.637067   0.499459
3  0.691738   0.671829
4  0.581525   0.637158
5  0.694777   0.731104

Process finished with exit code 0
```

2 复现思路展示

2.1 概述

1. 复现过程分为

- MergeFileProcess：将txt格式的MergeFile提取成为可读的csv格式文件。`(MergeFileProcess.py)`
- GenerateLabel：将转换为csv格式的文件的复现论文中VideoStateLabeling过程，生成带有Label的csv文件。`(GenerateLabel.py)`
- GenerateTrainData：生成最终可以输入到训练模型中的csv文件（包含ChunkDetection以及FeatureExtraction）`(GenerateTrainData.py)`
- 生成pkl训练模型：在ipython文件`RF_Train`中，使用四折交叉验证，生成了3*4=12个RF模型。
- 测试模型结果：在`TestModel.py`中，对于四折交叉生成的模型进行测试，得到测试结果。
- `main.py`整合了上述1、2、3、5过程，测试所需模型采用了在我们本地生成的pkl模型，当然您也可以重新运行`RF_Train.ipthon`文件，覆盖我们生成的模型，再进行测试。

2.2 VideoStateLabeling过程

```
*************** Beginning ***************
Succeed to construct COL:207
Folder : RequetDataSetNew/A0/MERGED_FILES/ , File Num : 95
Folder : RequetDataSetNew/A1/MERGED_FILES/ , File Num : 130
Folder : RequetDataSetNew/A2/MERGED_FILES/ , File Num : 91
Folder : RequetDataSetNew/A3/MERGED_FILES/ , File Num : 119
****** Mergefile to csv Succeed! *******
Mergefile Process Use 7.21807599067688s
Generate Label:|**********************************************| Finish 435 File
******* Generate Label Succeed! ********
```

1. 该过程在上述`MergeFileProcess`与`GenerateLabel`中完成。将txt文件输入后，输出命名以`*_label.csv`结尾的label文件，保存在LabelDataSet文件夹下，同时复现了论文附录中的smooth算法。生成文件（部分）截图如下所示，BufferWarning的二分类结果以0、1区分，BufferStatus的四分类结果以0、1、2、3区分，Resolution从144p→1080p（舍去了大于1080p的结果）以0→5所分类。

EpochTime	BuffWarning	status	Resolution
1516221103763	1	3	4
1516221103964	1	3	4
1516221104164	1	3	4
1516221104264	1	3	4
1516221104364	1	3	4
1516221104464	1	3	4
1516221104564	1	3	4
1516221104764	1	3	4
1516221104864	1	3	4
1516221104964	1	3	4
1516221105164	1	3	4
1516221105364	1	3	4
1516221105564	1	3	4
1516221105764	1	3	4
1516221105964	1	3	4
1516221106064	1	3	4
1516221106164	1	3	4
1516221106264	1	3	4
1516221106364	1	3	4
1516221106466	1	3	4

2. 程序均采用了python中的`concurrent.futures`包中的进程池进行进行并行处理数据，我们采用的为8个进程并行的方式处理（要求cpu可用核心≥8），这样可以大大缩减处理数据的时间。

```
with concurrent.futures.ProcessPoolExecutor(max_workers=8) as executor:
        executor.map(cal_label, files)
```

2.3 ChunkDetection与FeatureExtraction过程

```
Generating A0 training data: 100%|██████████| 95/95 [01:52<00:00,  1.18s/it]
Generating A2 training data: 100%|██████████| 91/91 [01:58<00:00,  1.30s/it]
Generating A3 training data: 100%|██████████| 119/119 [02:43<00:00,  1.37s/it]
Generating A1 training data: 100%|██████████| 130/130 [02:53<00:00,  1.33s/it]
***** Generate TrainData Succeed! ******
Use 175.447632789611825
```

1. 该过程在`GenerateTrainData`中完成，对于pcap文件的解析，我们并没有使用`scapy`包解析（实在太慢），我们用了`dpkt`包解析文件，大大提高了解析速度。过程与论文基本保持一致。同样对于四个不同的文件夹中文件，我采用了多进程并发的方式进行上述过程。可以将这一步的时间缩短四倍。
2. 唯一有所差异的地方是：我们生产的Feature是采用了120+2的形式共使用的是122个feature对应3个label。除了120个时间窗口的chunk metrics以外，我们并没有按照论

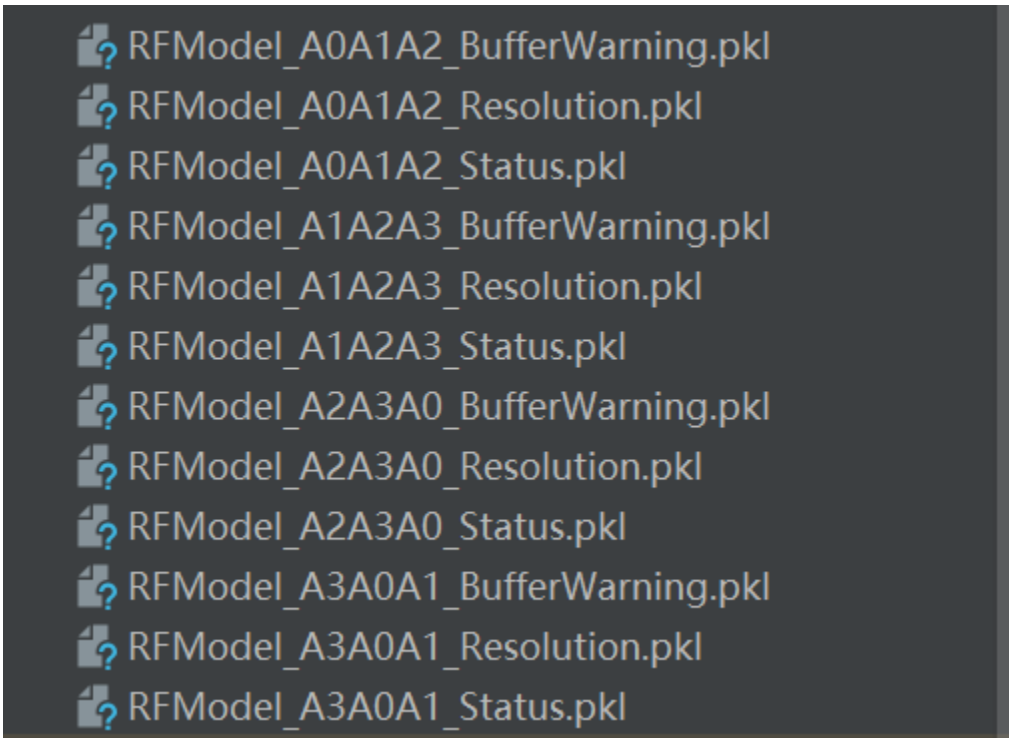
文采用7个chunk metrics，而是从这7个挑选了ChunkSize与DownLoadSize作为Feature的指标。我个人认为StartTime等其他五个似乎不具备Feature的特性。

3. 最终得到的结果是一个共120+7+3=130行数的大列表，共四张对应A0→A3文件，存储到TrainData文件夹下。但是对于Resolution的生成与其他两个label的生成不太一样，我们提取出了上述表格中只含有VideoChunk的列（在表格中对于type==0），然后生成了用于训练Resolution的csv数据表格，因此在TrainData文件中，一共有2*4=8个用于训练的文件

2.4 随机森林的训练与预测

1. 采用四折交叉验证的方法，及就是三个作为训练集，一个作为验证集。对于前两个label（Warning与Status）输入的训练矩阵形状是（122*TrainData数目），对于Resolution的训练输入的矩阵形状是（122*TrainData中type==Video数目）。生成的模型由3*4=12个。

2. 训练结果通过`joblib`包保存问pkl格式，A0A1A2代表这个模型用哪几个训练集合并生成，后面的是label代表是哪个label的模型数据。



3. 死者交叉验证的结果如下，我们计算了与论文的误差，均小于1个百分点。

Type	BufferWarning	BufferStatus	Resolution
Paper Accuracy	0.92	0.842	0.669
My Accuracy	0.912	0.837	0.668
Error	0.009	0.006	0.002