1.环境配置

安装mmcv与ffmpeg

安装mmdet与mmaction2并测试



打开Anaconda Prompt

开始环境的配置

conda create -n mmact python=3.8

conda activate mmact

conda install pytorch==1.8.0 torchvision==0.9.0 torchaudio==0.8.0 cudatoolkit=10.2 -c pytorch

之后进入虚拟环境安装mmcv

pip install mmcv\_full-1.4.5-cp38-cp38-win\_amd64.whl

安装ffmpeg

打开mmaction2/mmaction2-0.22.0目录下的ffmpeg解压包



将解压后的bin目录添加到系统环境变量，具体操作如下：

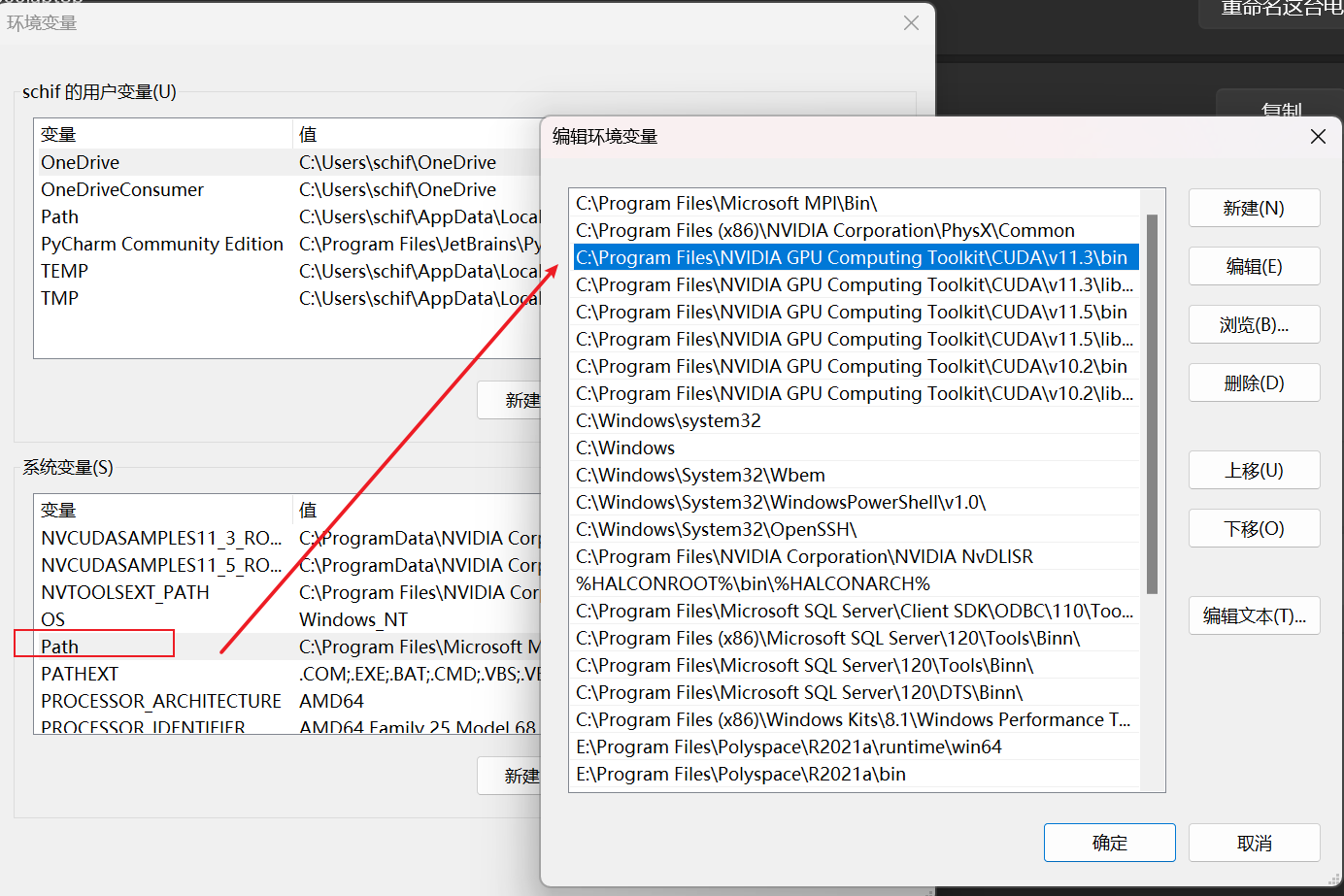


右键计算机点击属性



进入高级系统设置在“高级”选项卡中点击环境变量

在环境变量中找到Path，选择新建，将解压的bin目录添加即可



cmd输入ffmpeg有命令提示则成功

安装mmaction

cd mmaction2-master

pip install -r requirements.txt

python setup.py develop

安装mmdet

cd mmdetection-2.20.0

conda install git

pip install -re requirements.txt

（如果上一步报关于mmtrack的错误，是网络原因，pip install git+https://github.com/open-mmlab/mmtracking#egg=mmtrack，若执行该命令还是报错，这里作者是采用科学上网的工具解决）

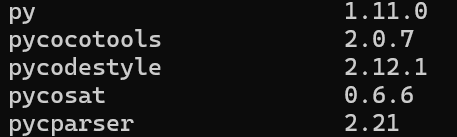
（如果上一步报关于error:Microsoft Visual C++ 14.0 or greater is required，默认安装VisualCppBuildToolsFull.exe）

Python setup.py develop

至此已配置完毕

激活环境后输入pip list命令





若环境列表中存在如上图所示的一些关键包，说明配置成功，安装无误。

2.AVA数据集介绍及创建自己的数据集

在mmaction2/mmaction2-0.22.0/tools/data/ava目录下有AVA\_annotation\_explained.md的针对AVA数据集的详细解释的英文文件。下面针对该文件的内容作简要介绍。

AVA数据集包含430个视频，其中235 个用于训练、64 个用于验证和 131 个用于测试。每个视频有 15 分钟的注释，间隔为 1 秒。

pkl后缀的文件指示由人体检测器生成的检测框集合。

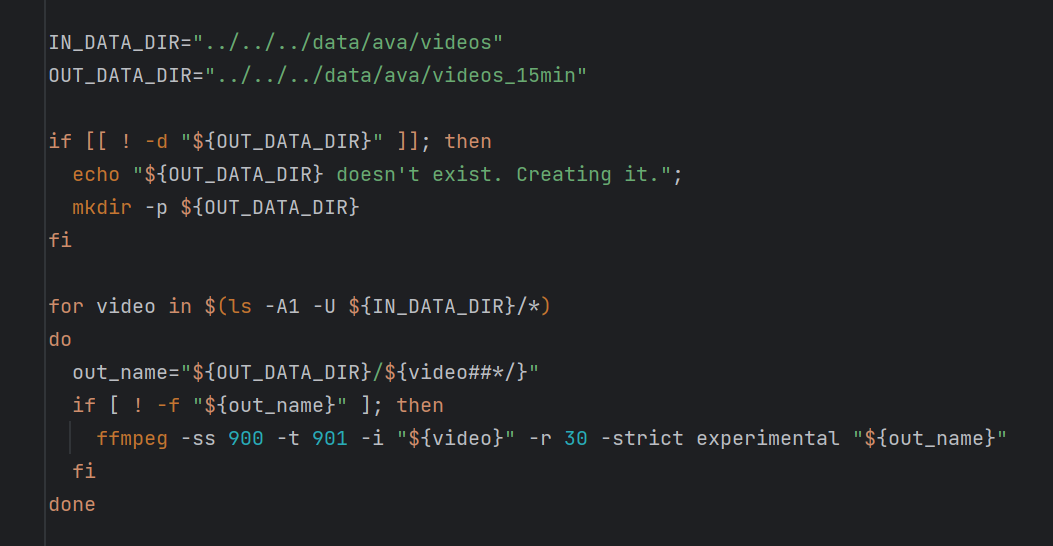
以ava\_dense\_proposals\_train.FAIR.recall\_93.9.pkl文件为例，该文件是一个大小为203626的字典集。字典的键由视频ID和时间戳组成。例如键-5KQ66BBWC4,0902’所对应的值是在视频-5KQ66BBWC4的第902秒的检测结果。字典的值是在对应的帧生成的人体检测框的数值，由[x1,y1,x2,y2,score]五个数值组成，其中x2，y2表示检测框右下角的坐标；x1，y1表示检测框左上角的坐标，score表示该检测框的置信度。AVA数据集只标注置信度在0.9以上的框，下面介绍标注文件。

以行为标签文件ava\_train\_v2.1.csv为例，该文件共有837318行，每一行是针对在一个检测框内的人类实例的标签。例如该文件中的第一行为:

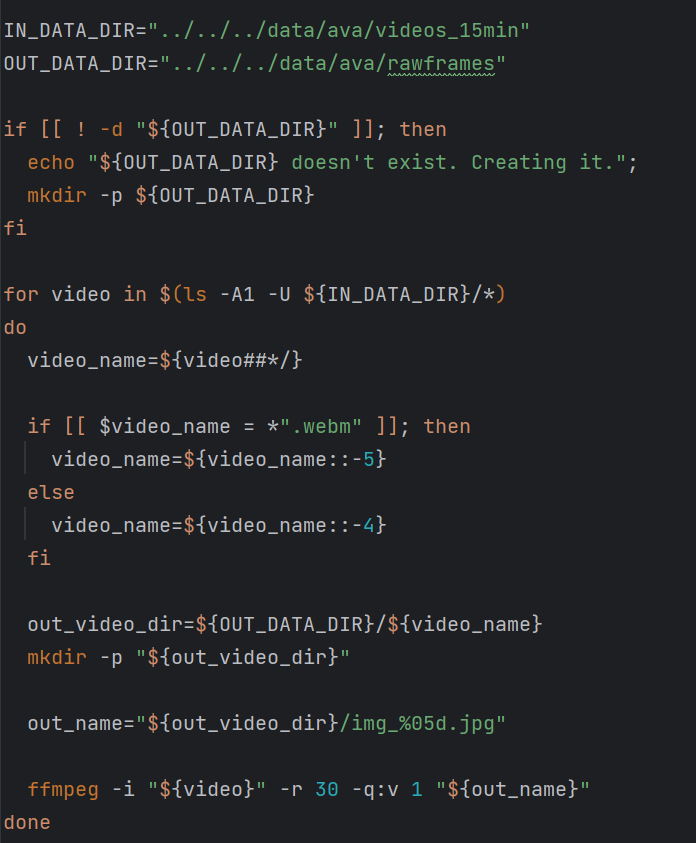
‘-5KQ66BBWC4,0902,0.077,0.151,0.283,0.811,80,1’ ，其中-5KQ66BBWC4和0902表示该标签打在了名称为-5KQ66BBWC4的视频的第902秒。紧跟着的四项0.077,0.151,0.283,0.811表示检测框的坐标（依次为x1，y1，x2，y2的数据）。80表示动作标签，1表示检测框的ID（检测框ID实际作用不大）。

下载的AVA数据集均为未处理的视频文件，要投入使用还需要进行下面步骤的处理。

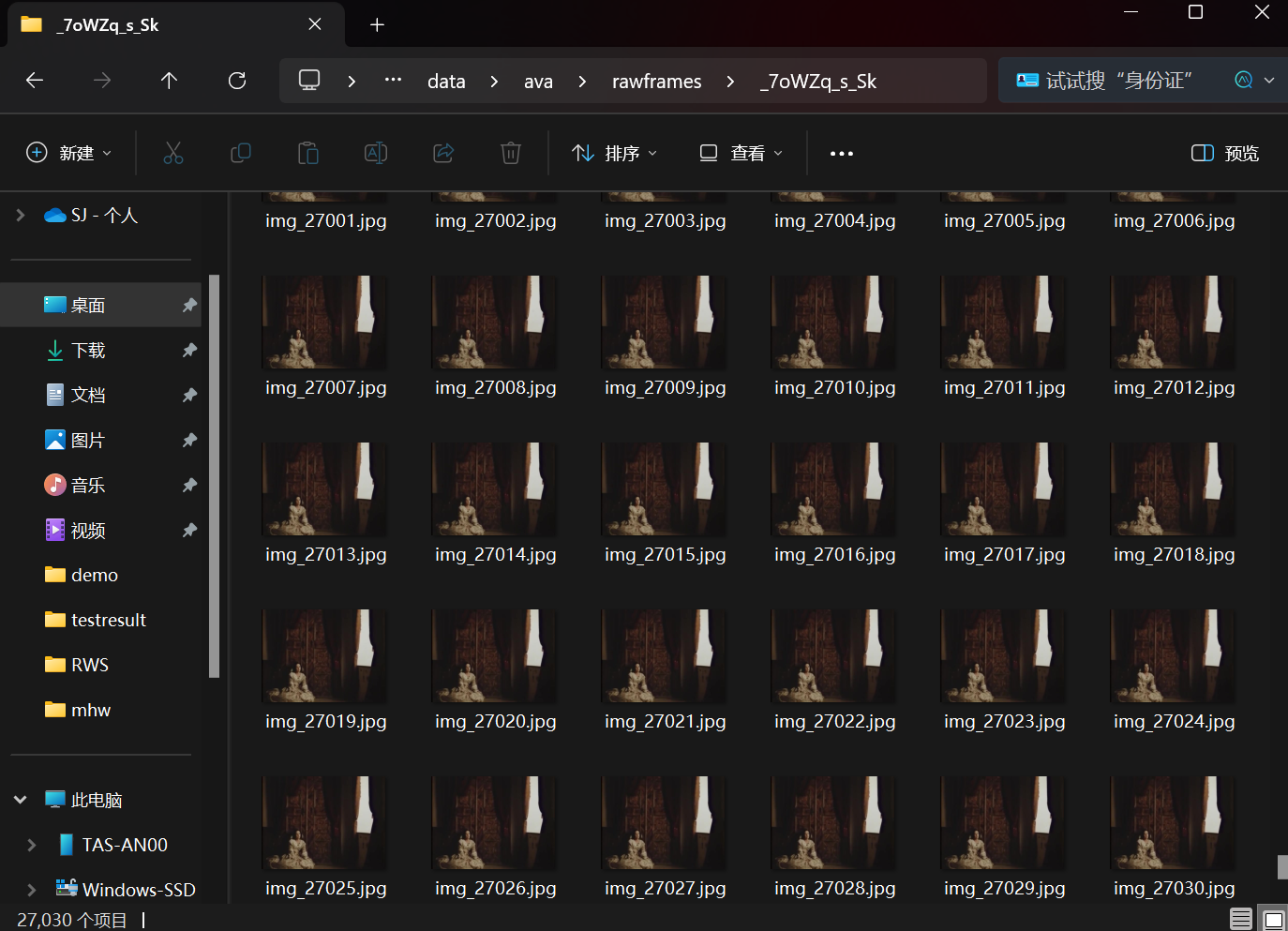
在mmaction2/mmaction2-0.22.0/tools/data/ava目录下运行cut\_video.sh即在终端中输入bash cut\_video.sh。cut\_video.sh的代码如下图所示:



IN\_DATA\_DIR与OUT\_DATA\_DIR根据自行下载的目录进行修改，IN\_DATA\_DIR为AVA数据集原始视频文件路径，OUT\_DATA\_DIR为处理后的视频保存路径。代码中”fmpeg -ss 900 -t 90…-r 30”频的第900秒开始往后的视频每秒30帧截取901秒。

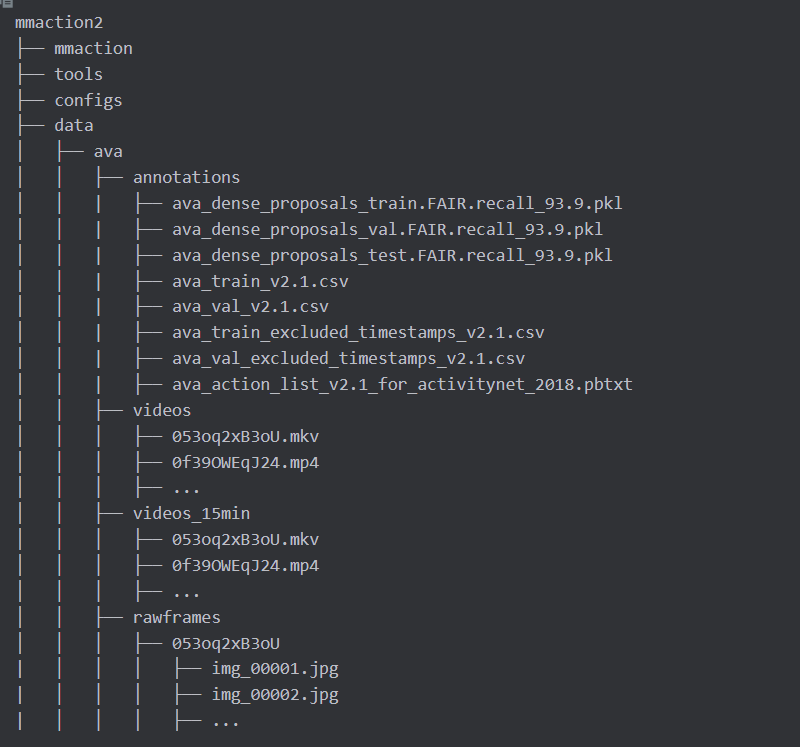
截取视频后在动作检测之前还需要提取RGB或者提取光流。由于提取flow的方法所占的空间过大，这里仅介绍提取RGB的方法在mmaction2/mmaction2-0.22.0/tools/data/ava目录下运行extract\_rgb\_frames\_ffmpeg.sh中输入bash extract\_rgb\_frames\_ffmpeg.sh。extract\_rgb\_frames\_ffmpeg.sh的代码如下图所示:

IN\_DATA\_DIR与OUT\_DATA\_DIR根据自行下载的目录进行修改，IN\_DATA\_DIR为上一步截取视频的保存路径，OUT\_DATA\_DIR为提取rgb处理后的图片保存路径。



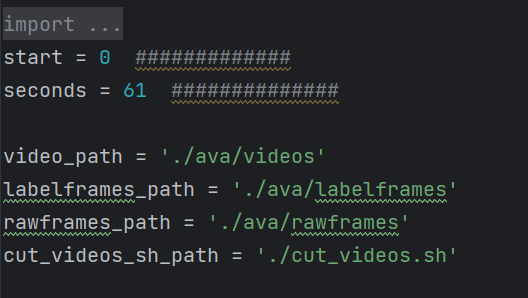
打开处理后的rawframes文件夹可以看到每一个视频在rgb提取的处理后被分为了27030张图片，而27030/30的结果为901，验证了上一步中的cut\_video是以每秒30帧截取901秒。

下一步为下载pkl文件，在mmaction2/mmaction2-0.22.0/tools/data/ava目录下运行fetch\_ava\_proposals.sh中输入bash fetch\_ava\_proposals.sh。（此步骤若发生报错，为网络原因，可自行上网搜索下载pkl或者使用科学上网工具。）

最后数据集文件的构成应如下图所示

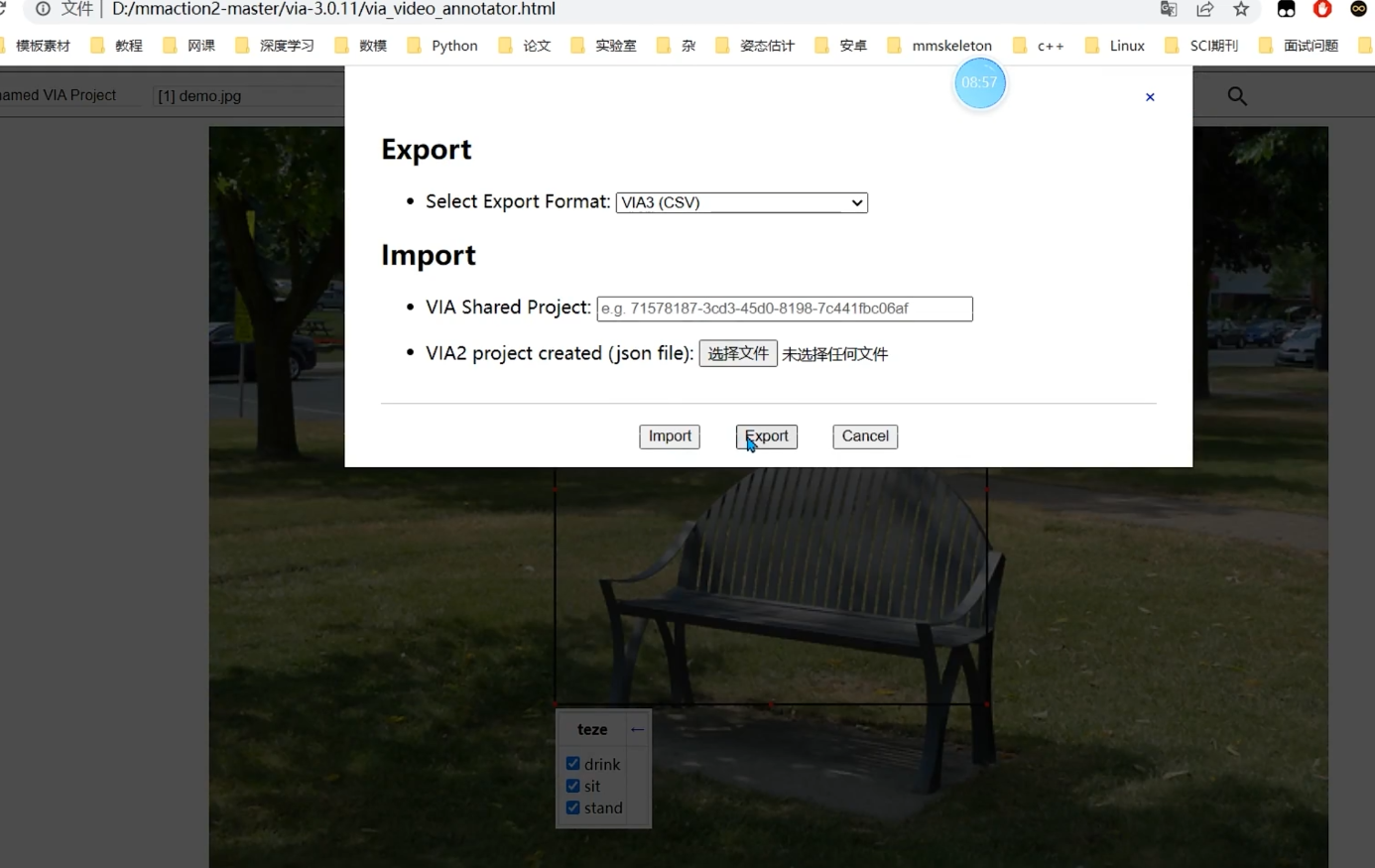
其中训练用到的文件为annotations和rawframes下的文件。其中anootations文件夹下的pkl和csv文件前文已经介绍过了为训练所必需的文件，带excluded\_timestamps的csv文件标注了被排除训练的时间帧（在后文制作自己的训练集时，若没有需要排除训练的时间帧，该文件可省去），pbtxt文件中记录了所有的动作标签及其ID。

下面进行自建数据集的方法介绍：

与上述步骤相似，将要训练的视频放在mmaction2/mmaction2-0.22.0/data/ava/videos文件夹下，在mmaction2/mmaction2-0.22.0/data/ava路径下打开video2img.py，部分代码如下图所示：

自行修改start和seconds，二者的含义分别为截取开始时间和截取总时间，即该文件的功能包含了上述的cut\_video和提取rbg步骤的功能。类似的，处理后的文件被保存到了rawframes文件夹中待训练。

对于标签的制作，将mmaction2/mmaction2-0.22.0路径下的via-3.0.11.zip解压，解压后进入并打开via\_video\_annotator.html，进入网站上传图片自行手动标注标签。

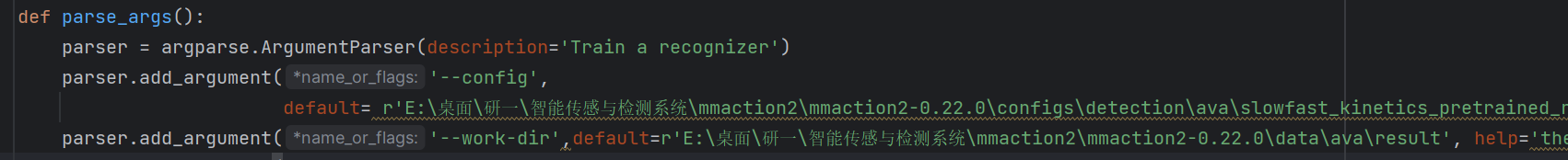


导出的标签文件并不能直接使用，运行mmaction2/data路径下的via2ava.py即可输出相应的标签csv文件。自此数据集自建完毕。

由于AVA数据集过于庞大，作者仅下载了其中的50个视频用于训练，因此对于标签文件需要做相应的处理：运行mmaction2/mmaction2-0.22.0/data/ava路径下的csvtest.py与pkl.py文件即可实现对csv和pkl文件的修改。修改的根据是str2.txt中用到的50个视频文件名，读者可根据自身下载情况自行修改。

3.训练数据集

打开mmaction2/mmaction2-0.22.0/tools路径下的train.py文件。



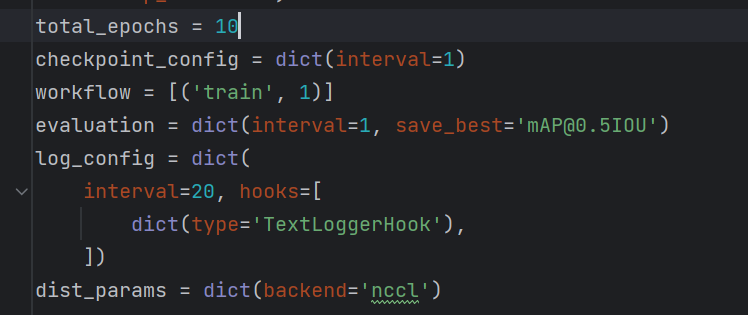
如上图所示，根据自身情况修改训练config和work-dir。训练的config可在mmaction2/mmaction2-0.22.0/configs/detection/ava路径下自行选择，这里以选择slowfast\_kinetics\_pretrained\_r50\_8x8x1\_20e\_ava\_rgb.py为例；work-dir为训练输出的保存文件夹路径（保存训练日志和最终模型）。

打开选择的config文件进行相应的修改：

在代码的42、136、147行左右将num\_classes修改为你的类别数加一。（AVA数据集的类别数为80，故这里为81）；在代码的63行左右，如下图所示：



根据自己的文件路径进行修改。（若为自己创建的数据集，没有需要忽略的时间帧，则相应的excluded\_timestamps文件留空即可。）

在代码的164行到171行左右，如下图所示：

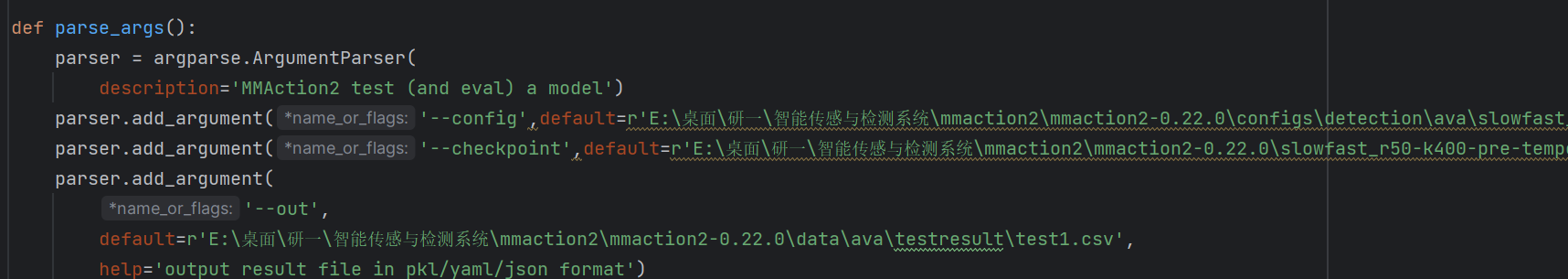
此处total\_epoch为训练总轮数，evaluation中interval为输出的轮数间隔，图中的含义即为每一轮输出训练效果最好的模型保存为名称[mAP@0.5IOU](mailto:mAP@0.5IOU)的pth文件。Log\_config为日志输出的配置文件，图中含义为每一轮的训练中，每20个训练样本输出一次训练日志。读者可根据自己的需要自行修改。

在代码的176行，load\_from为加载预训练模型，若不需要可注释，并将load\_from的参数改为None。

自此数据集训练文件配置完成。

3.测试训练模型

打开mmaction2/mmaction2-0.22.0/tools路径下的test.py文件。

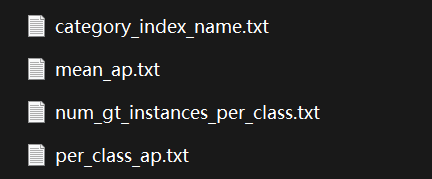


如上图所示修改相应的config，checkpoint以及out参数。checkpoint为上一步训练完毕后保存在work-dir下的模型参数pth文件，config与训练时配置的config相同，为模型的基本框架配置；out为测试的输出目录，测试结果将会生成一个csv文件。

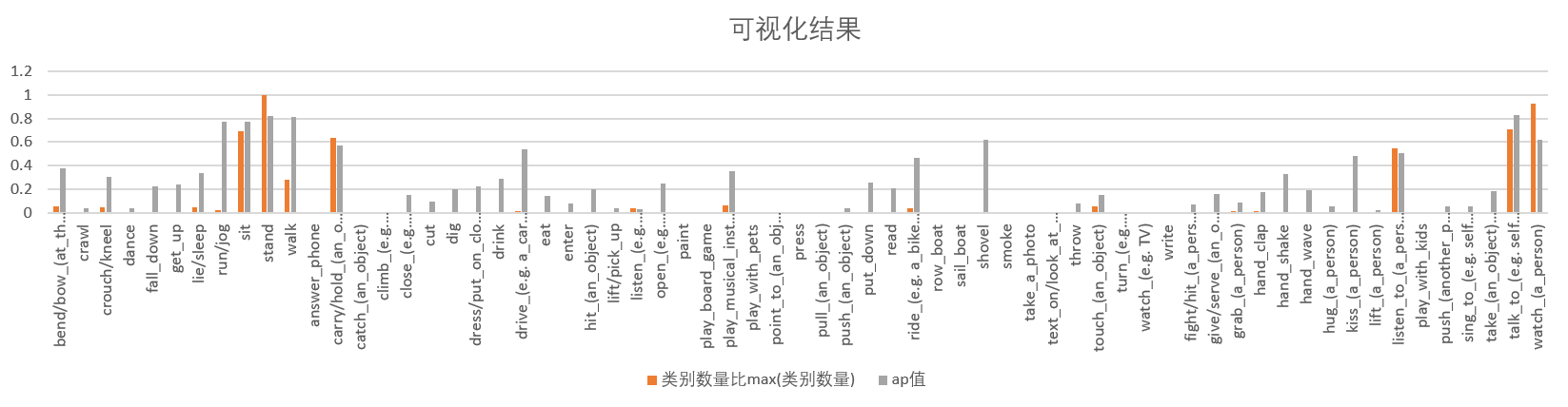
同时代码对

mmaction2\mmaction20.22.0\mmaction\core\evaluation\ava\_evaluation\object\_detection\_evaluation.py文件进行了相应的修改用于后续的测试结果可视化。具体的修改参考博客：<https://blog.csdn.net/WhiffeYF/article/details/126349432?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522C97B7D1D-135D-493C-89DA-B9BCBFB776EC%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fblog.%2522%257D&request_id=C97B7D1D-135D-493C-89DA-B9BCBFB776EC&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~blog~first_rank_ecpm_v1~rank_v31_ecpm-2-126349432-null-null.nonecase&utm_term=mmaction2&spm=1018.2226.3001.4450>

修改完毕后运行的结果将包含下列文件：

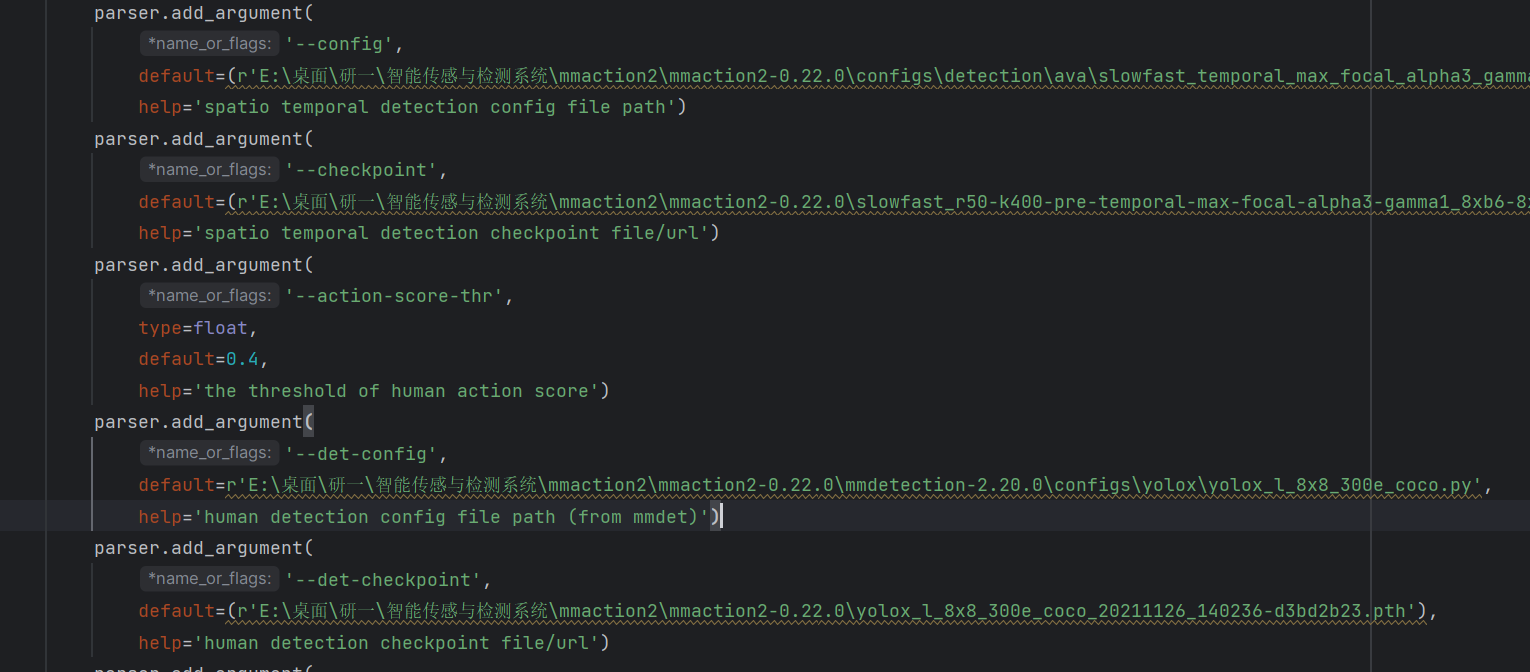


category\_index\_name记录了行为类别的名称，mean\_ap记录了该模型的测试平均ap值，num\_gt\_instances\_per\_class记录了每个行为类别对应的样本数量，per\_class\_ap记录了每个行为类别的测试ap值。默认修改后的文件保存目录在mmaction2\mmaction20.22.0\plot下，运行该目录下的analy.py文件将会输出all.txt文件在同一目录下，该文件为上述四个文本文件的整理整合。将all.txt文件中的数据复制到excel中进行处理（分列，去0列）后，即可将测试结果可视化。如下图所示：



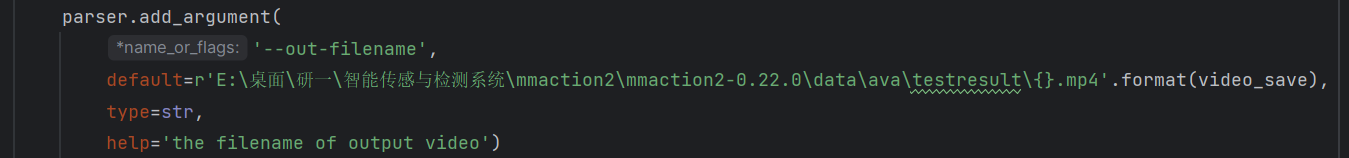
4.前端使用说明

训练测试完毕后，为了快速调用训练后最好的模型投入使用，这里设计了一快捷ui交互界面。在mmaction2\qt前端的目录下，打开finaltest.py文件，如下图所示：

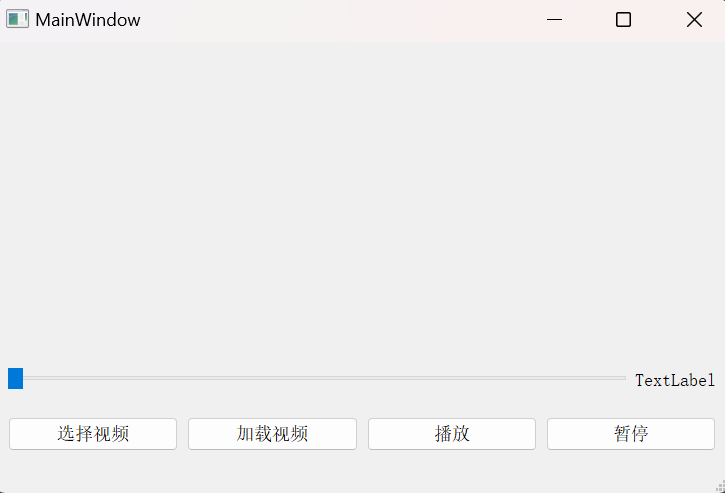


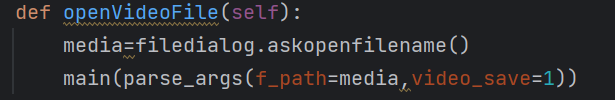
修改对应的config和checkpoint，与你要投入使用的模型一一对应。det-config和det-checkpoint指示识别框，默认是使用fast rcnn，但yolox在该方面的表现更好，yolox对应的config在图示的文件目录下可找到，相应的checkpoint可到官网下载。

<https://github.com/open-mmlab/mmdetection>



如上图所示，大约在90行作用可配置模型处理后的视频的输出地址，根据需要自行配置。

配置完毕后启动同一目录下的ui.py，右击运行即可将模型投入使用：

如上图所示，选择视频键的功能为选择所要进行时空动作检测的目标视频，选择完毕后即进行处理，在python终端可看到处理进度条，处理完毕后点击加载视频按键，选择处理后的视频即可进行播放。

在ui.py代码的50行左右，这里可以自行修改video\_save的值，其含义为输出视频的文件名称。本前端的编写使用的是python qt，有该方面基础的读者若是想要对前端控件进行修改，可对mmaction2\my\_windows下的文件右键external tools调用qt designer进行编写修改。修改完毕后右键external tools点击pyuic，即可生成对应的GUI.py文件，将其移入qt前端文件夹即可。相应的快捷配置可参考博客：

[https://blog.csdn.net/qq\_51210361/article/details/131582764?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%25223325049C-EF32-4190-B605-8E899960A052%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=3325049C-EF32-4190-B605-8E899960A052&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~sobaiduend~default-1-131582764-null-null.142^v100^pc\_search\_result\_base1&utm\_term=external%20tools&spm=1018.2226.3001.4187](https://blog.csdn.net/qq_51210361/article/details/131582764?ops_request_misc=%257B%2522request%255Fid%2522%253A%25223325049C-EF32-4190-B605-8E899960A052%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request_id=3325049C-EF32-4190-B605-8E899960A052&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduend~default-1-131582764-null-null.142%5ev100%5epc_search_result_base1&utm_term=external%20tools&spm=1018.2226.3001.4187)

针对视频播放器的代码参考博客：

<https://blog.csdn.net/u012552296/article/details/89295273>

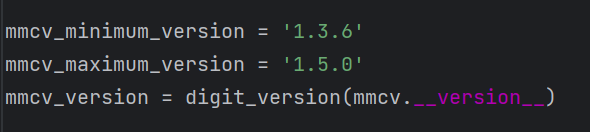
常见问题：

1. NVIDIA GeForce RTX 4060 Laptop GPU with CUDA capability sm\_89 is not compatible with ...

报错原因：显卡算力与cuda版本不匹配

作者的显卡为40系显卡，若按照前文所述步骤配置环境，由于40系显卡不支持10.2版本的cuda，故报错，若您在训练过程中也出现类似的错误，需要按照下述的步骤进行环境配置更改。

打开mmaction2-0.22.0/mmaction/\_\_init\_\_.py，可以看到



mmaction2的运行需要mmcv的版本在1.3.6到1.5.0之间

查询自己的显卡支持哪种算力的cuda，如下图所示为算力和cuda的版本对应关系



<https://developer.nvidia.com/cuda-gpus#compute> 该链接可查询自己显卡的算力

您的显卡算力不能大于cuda的最大cc（计算能力）

已作者的4060显卡为例，查询到4060显卡的算力为8.9，按照图中所示似乎只有cuda11.8版本的cc符合要求，但11.8版本的cuda对应的mmcv版本不符合要求，经过长时间的查询，发现cuda11.3的cc也能支持4060显卡，故进行cuda11.3的安装。此步骤之后进行cuda和对应的torch版本的查询，得到cuda11.3支持的torch版本为1.11.0，在终端中输入以下命令

pip install torch==1.11.0+cu113

这里注意若末端不加上“cu113”即对应的cuda语句，可能会导致安装的torch为cpu版本无法使用GPU加速；安装完毕后可打开mmaction2-0.22.0/cuda\_test.py文件并运行



若出现如上所示的结果，表明安装成功。

在显卡算力和cuda版本和torch版本一一对应匹配后，进行mmcv的安装

<https://download.openmmlab.com/mmcv/dist/cu115/torch1.11/index.html>

（根据您的配置自行修改网址即可即cu115表示的是cuda11.5，troch1.11表示torch的版本为1.11，修改后进入网址搜寻可下载的mmcv即可）

在上述网址中找到相应的mmcv版本，这里作者选择的是

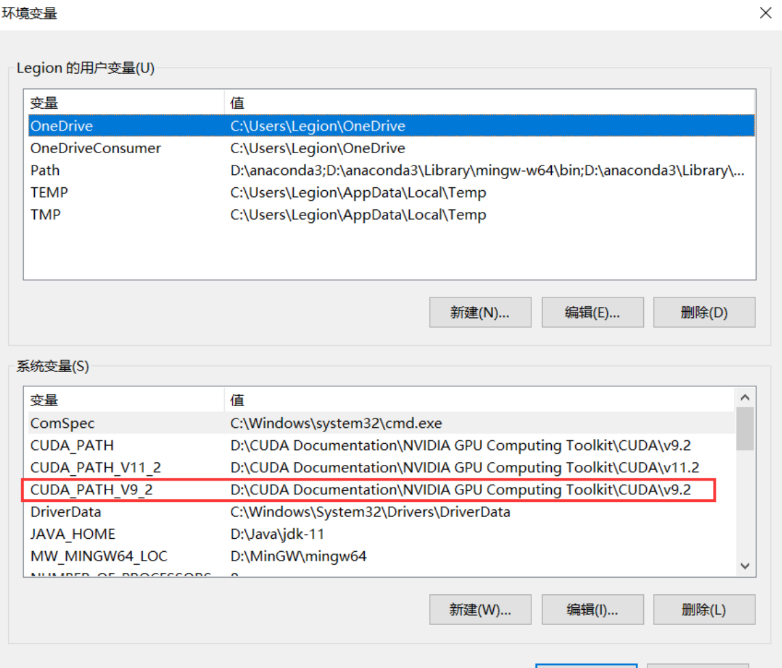
mmcv\_full-1.4.7-cp38-cp38-win\_amd64.whl即1.4.7版本，符合要求，将下载后的文件放到对应的目录中，pip install即可。

2.多个版本的cuda切换问题。

安装mmcv时可能新安装的cuda版本和原来已经安装的版本冲突，下面介绍安装和切换多个cuda版本的方法。（参考<https://blog.csdn.net/Pharaoh_cool/article/details/139843766> 的相关教程。）

在官网下载相关的安装包后进入安装

如上图所示，若之前已经安装过别的版本的cuda，这里只需要勾选第一个选项即可。

安装完成后右键点击此电脑->属性->高级系统设置->环境变量。

如上图所示，系统变量中出现了新安装的cuda环境变量



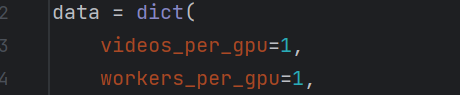
如上图所示，在系统变量中找到CUDA\_PATH 和 NVCUDASAMPLES\_ROOT 两个环境变量修改为想要的cuda版本的环境变量即可实现cuda版本的切换。最后Win + R 打开命令行窗口，输入 nvcc -V，即可输出 CUDA 版本校验。（对应版本的cudnn安装参考前文提到的网站内容即可。）

3. numpy.core.\_exceptions.MemoryError: Unable to allocate 10.5 MiB for an array with shape (1440, 2560, 3) and data type uint8

在测试训练后的模型时可能出现类似的错误，该错误出现的原因为，视频画幅的比例过大，计算机内存不足无法处理，此时需要通过第三方工具修改导入的视频素材的尺寸，或更换视频素材即可。

4.gpu out of memory

训练时可能会因为计算机gpu内存不足出现上述的报错，此时在选定config的配置文件中的第122到124行左右，如下图所示：



上一行参数表示单个gpu的批大小，下一行参数表示单个gpu的dataloader的进程，将这两个参数改小即可。