1. 环境安装

**1、本项目运行需要的基本条件包括：**

（1）anaconda虚拟环境管理软件；

（2）基于anaconda创建的包含pytorch、opencv等库的虚拟环境；

（3）图灵架构或安培架构的NVIDIA GPU（RTX20系及以上）。

因此，在开始环境配置前，请确保您的计算机至少满足以下条件：

（1）安装anaconda和项目本体的磁盘剩余10GB以上的磁盘空间；

（2）具备前述GPU设备且显存容量大于等于6GB。

另外，若您的计算机已经安装过CUDA和pytorch虚拟环境（torch>=1.7.1），只需在虚拟环境中执行本文档最后一句命令即可。

**2、环境配置步骤**

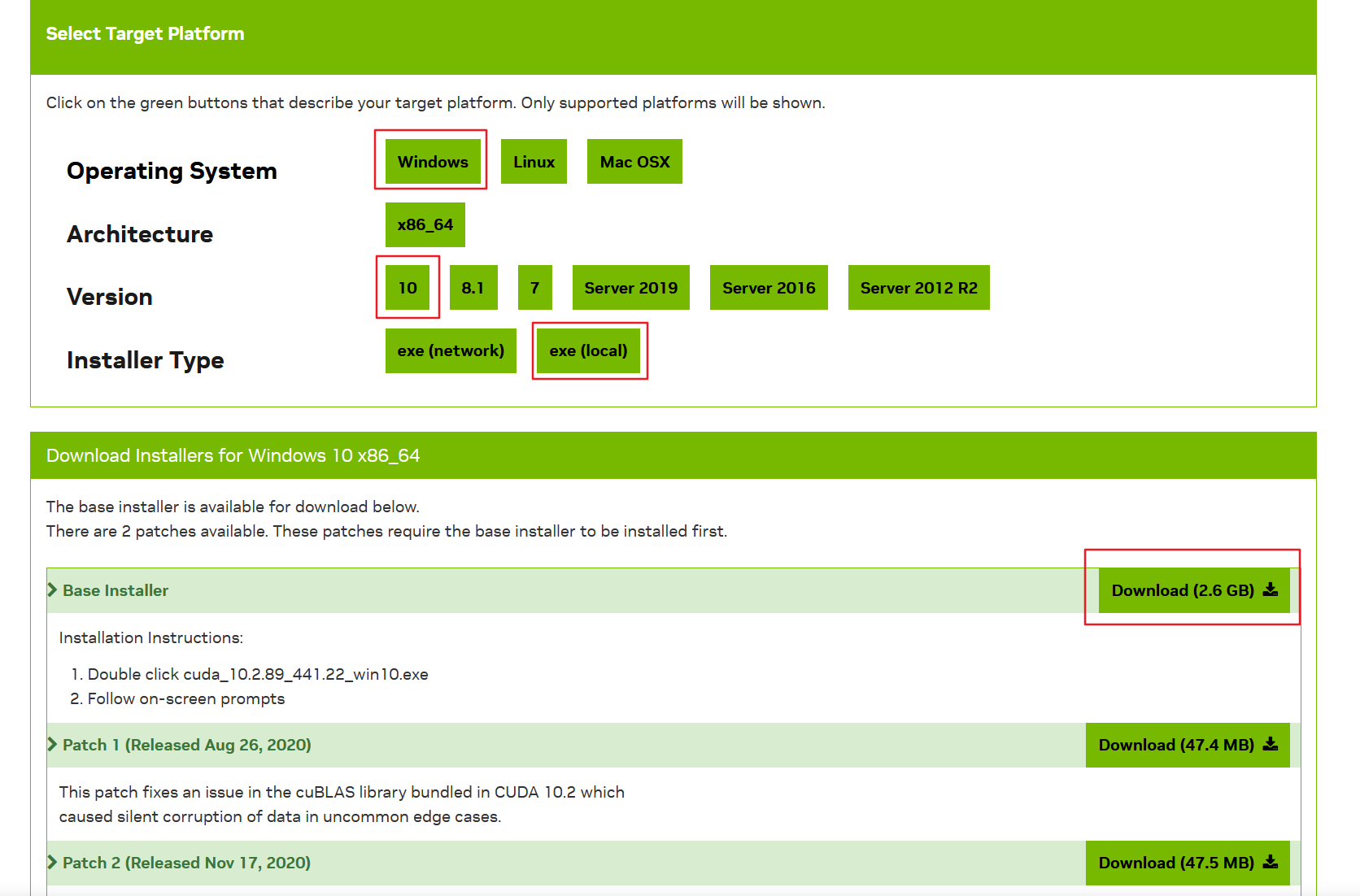
①、CUDA、cuDNN安装

1、打开网址https://developer.nvidia.com/cuda-toolkit-archive，如下图所示，图灵架构GPU（如RTX20系列）选择CUDA Toolkit 10.2，安培架构GPU（如RTX30系列、RTX40系列）选择CUDA Toolkit 11.3。



2、选择版本

以win10系统为例，按下图依次点击即可。下载后按默认路径安装即可。



3、编辑系统环境变量

右键“计算机”，依次选择属性→高级系统设置→高级→环境变量，在“系统变量”中选择path，点击“编辑”，将以下路径添加进去即可。

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.2\bin;

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.2\libnvvp;

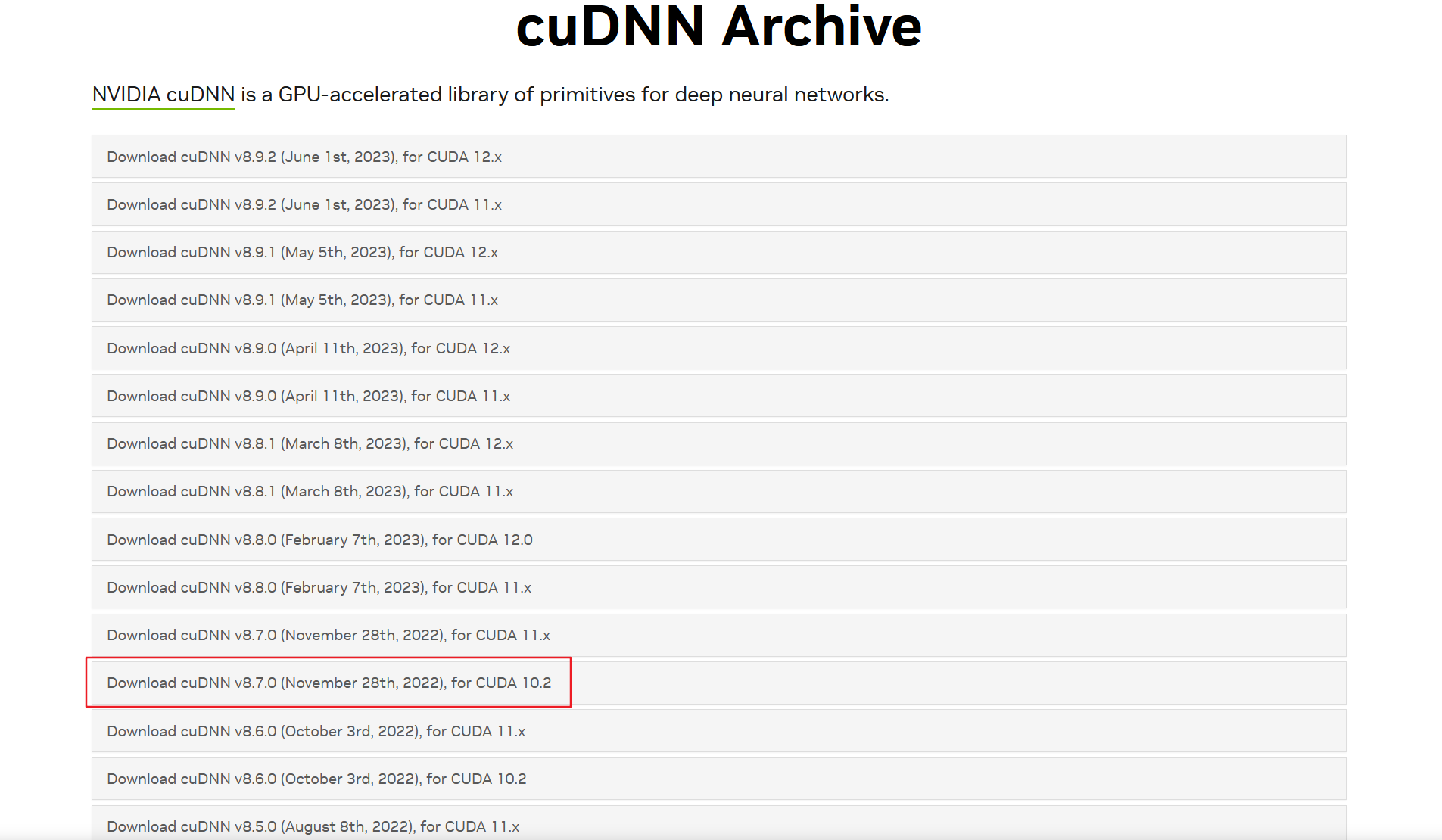
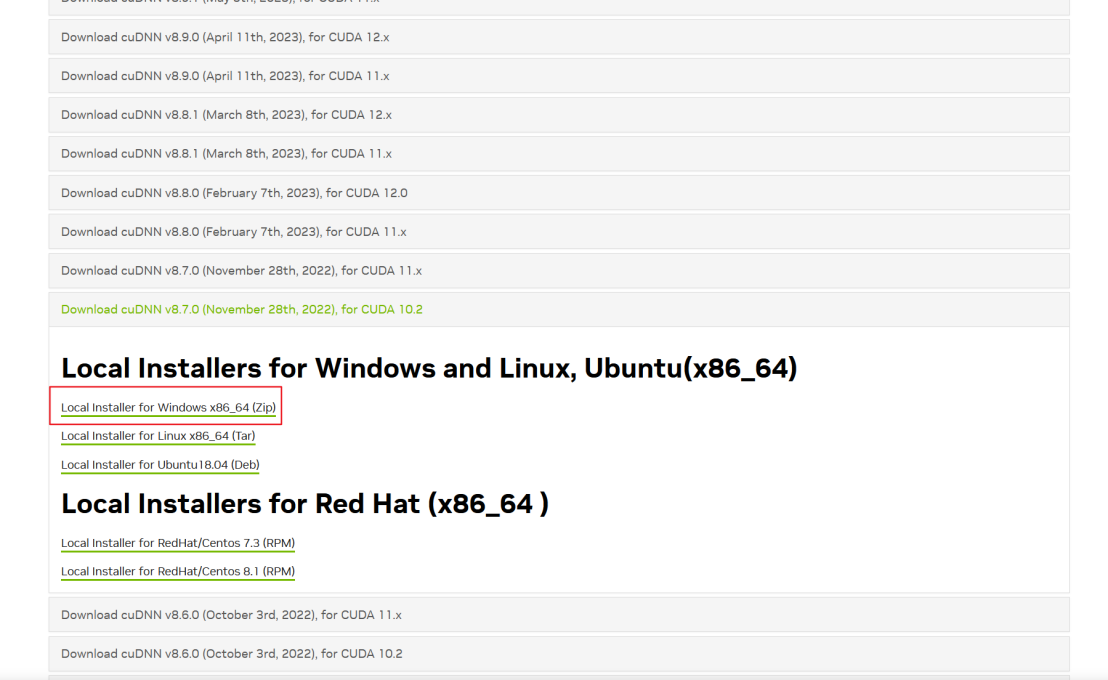
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.2;

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.2\lib\x64;

注意：若CUDA Toolkit安装版本和位置与本例不同，只需找到安装根目录中的bin、lib等文件夹，替换上述路径即可。

4、下载cuDNN

打开网址https://developer.nvidia.com/rdp/cudnn-archive，然后根据要求进行一个账号的注册，注册成功后便可以下载与CUDA版本相对应的cuDNN。以10.2版本CUDA Toolkit为例，如下图所示。



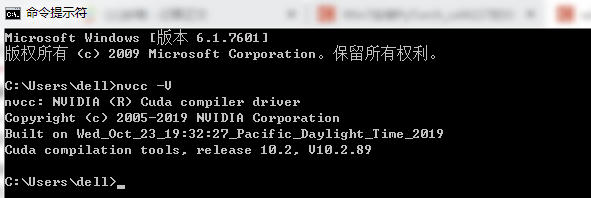
下载完成后进行解压得到一个CUDA文件夹，里面含有lib、include、bin文件夹。找到CUDA安装的位置，默认为C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.2，然后将CUDNN解压后的 lib、include、bin文件夹复制进去，若提示复制和替换，点击是即可。

5、验证CUDA安装是否成功

在“开始”菜单搜索栏中输入“cmd”打开命令行窗口，输入命令：

**nvcc -V**

得到下图所示结果（版本号和日期可以不同）则安装成功。

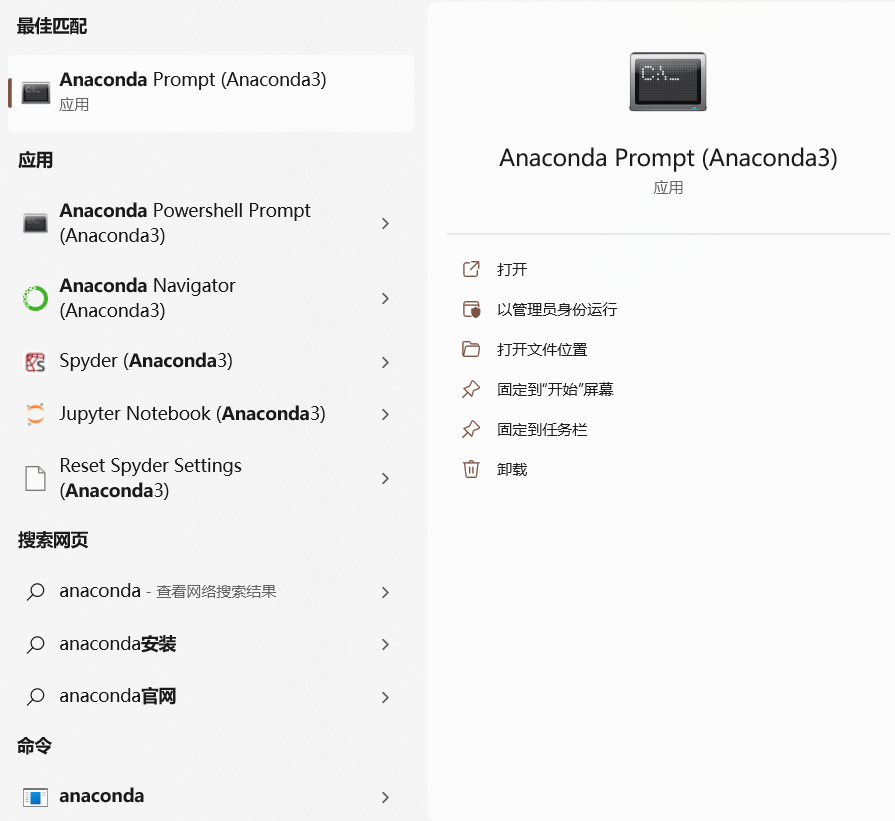


②Anaconda安装

Anaconda安装较为简单，下载、安装、配置环境变量等，可参考博客https://www.jianshu.com/p/d3a5ec1d9a08，注意在anaconda官网上选择最新版本安装即可（博客中为2019年老版本）

③虚拟环境配置：

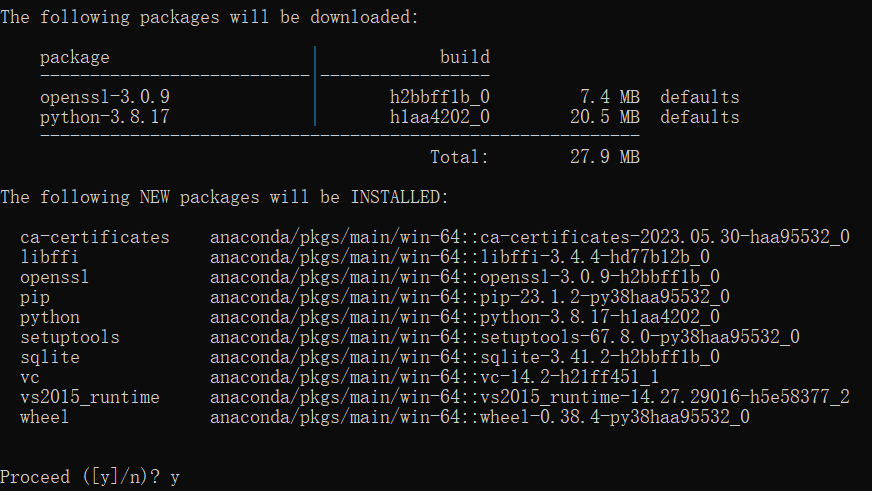
1. 创建虚拟环境



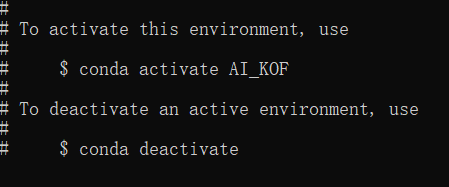
在“开始”菜单搜索栏输入“anaconda”即可找到anaconda命令行工具“Anaconda Prompt(Anaconda3)”，点击打开。在窗口中输入命令：

conda create -n AI\_KOF python=3.8

出现选项时输入“y”后回车：



出现如下提示即创建成功：



2、安装pytorch

在上一步窗口中继续输入命令：

conda activate AI\_KOF

如CUDA Toolkit版本为10.2，则输入以下命令：

conda install pytorch==1.10.1 torchvision==0.11.2 torchaudio==0.10.1 cudatoolkit=10.2 -c pytorch

如CUDA Toolkit版本为11.3，则输入以下命令：

conda install pytorch==1.10.1 torchvision==0.11.2 torchaudio==0.10.1 cudatoolkit=11.3 -c pytorch -c conda-forge

执行命令即开始pytorch安装。遇到选项一律输入“y”并回车，等待安装完成。

安装完成后输入：

python

回车后进入交互式界面。

再依次输入：

import torch

torch.cuda.is\_available()

若得到“True”，则证明pytorch成功安装。

验证后输入：

quit()

退出交互式界面

3、安装其他依赖

在上一步窗口中继续输入命令：

pip install numpy gym opencv-python ultralytics scikit-image pywin32 tensorboard

执行命令后等待安装完成即可。

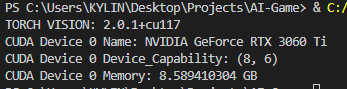
**至此，虚拟环境配置完成。**

1. 测试文件介绍

在代码的主目录下，有一个test文件夹，在test文件夹下，存有测试环境、测试cuda等必备设备的文件

1. test\_gpu.py

运行该文件后，会输出如下格式的信息：



此文件用来检测torch版本、以及是否存在CUDA设备，并输出CUDA设备的信息

1. 环境代码介绍
2. utils.py

存放用来截屏的函数、键盘的16进制映射、模拟键盘输入等函数，同时也有开始游戏、游戏重启、游戏结束的函数脚本，具体打开该文件后查看注释课了解其功能

1. GameENV.py

项目主文件，定义了一个ENV结构体，用以实现智能体（强化学习AI算法）与游戏环境（拳皇15）交互的过程，将智能体的决策转换成键盘输入，同时提取环境特征以及决策奖励返还给智能体，具体使用参考第五部分

1. Start.py

定义了一个用以开始游戏以及重启游戏的类，提供了写好的脚本供使用，也可以通过重新写该文件自定义实现游戏的开始和重启

1. Train.py

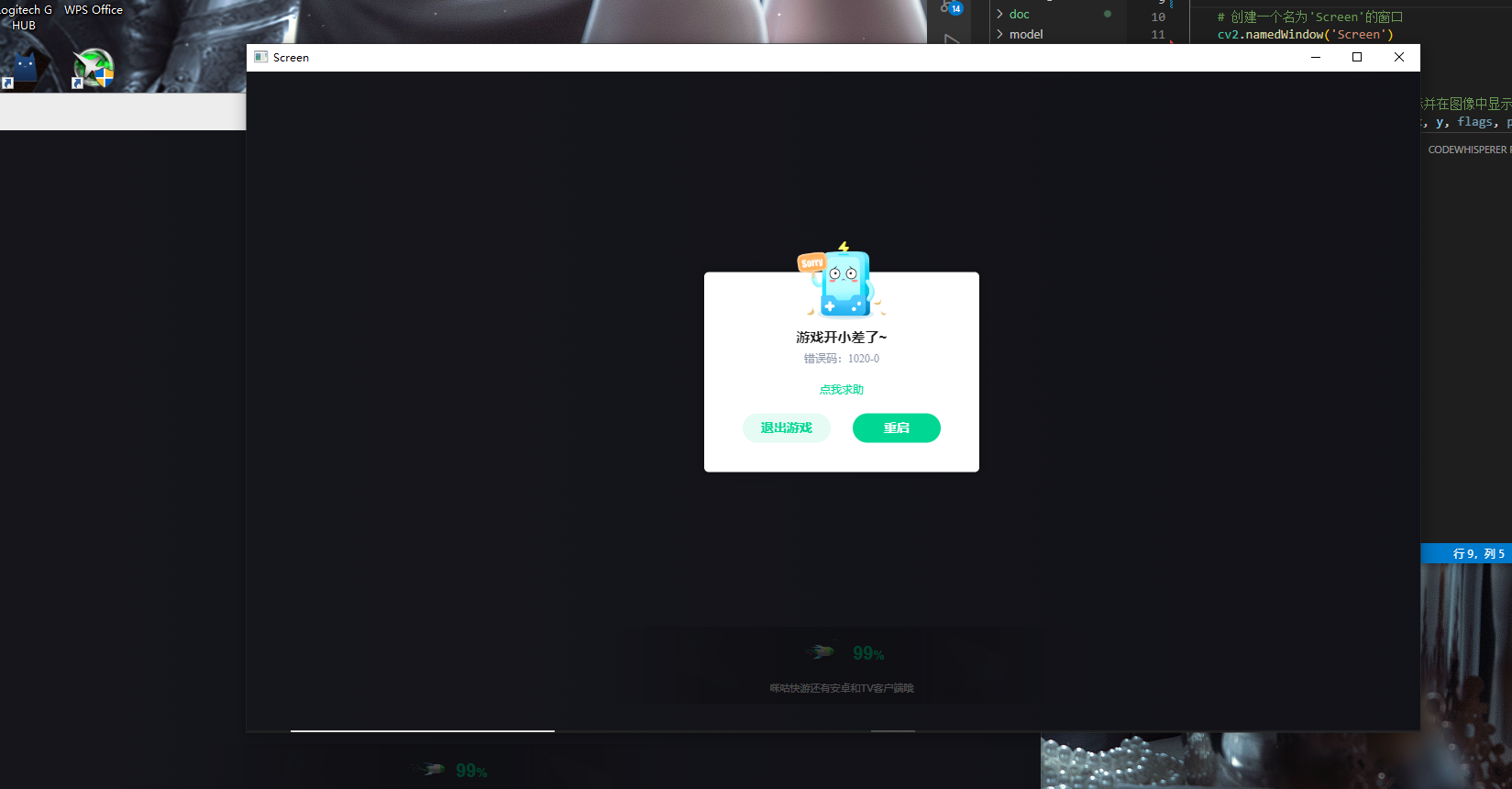
定义了一个用以训练模型的类，选手自行重写

1. Action\_keys.py

定义了AI可以选择的动作函数，因为拳皇连招丰富，需要选手自己动手写属于其自己的连招脚本，我们提供了一些范例，选手可以自行参考

1. Grab\_Screen.py

运行此文件进行截图定义窗口位置，鼠标在出现的截图上移动后会出现相对位置坐标，移动和缩放拳皇游戏界面，使得运行后出现的截图窗口上的游戏界面不能有黑边和其他浏览器上的内容



1. number\_rec.py

存放用以进行数字识别的模型代码

1. API
2. utils：
   1. 键盘映射变量：

# Keys

SPACE = 0x39

UP = 0xC8

LEFT = 0xCB

RIGHT = 0xCD

DOWN = 0xD0

W = 0x11

A = 0x1E

S = 0x1F

D = 0x20

ENTER = 0x9C

M = 0x32

J = 0x24

K = 0x25

LSHIFT = 0x2A

R = 0x13

V = 0x2F

U = 0x16

Q = 0x10

I = 0x17

O = 0x18

P = 0x19

C = 0x2E

F = 0x21

up = 0xC8

down = 0xD0

left = 0xCB

right = 0xCD

esc = 0x01

* 1. 、PressKey(hexKeyCode)

输入：hexKeyCode：键盘对应按键的16进制数字

输出：模拟摁下该16进制数字对应的键盘摁键

* 1. 、ReleaseKey(hexKeyCode)

输入：hexKeyCode：键盘对应按键的16进制数字

输出：模拟松开该16进制数字对应的键盘摁键

* 1. 、grab\_screen(region=None)

输入：region：一个有四个整数元素的元组，对应（xmin，ymin，xmax，ymax）

输出：由（xmin，ymin）和（xmax，ymax）两点确定的屏幕截图

* 1. 、init\_game()

输入：无

输出：自动从开始界面进入游戏的脚本

* 1. 、restart\_game()

输入：无

输出：自动从开始界面进入游戏的脚本

* 1. 、is\_win(situation)

输入：situation：situation: 1表示敌方劣势，-1表示己方劣势，一般不会有0出现

输出：己方是否胜利，胜利为True，否则为False

* 1. 、is\_loading(img)

输入：img(np.ndarray): 输入一帧图像(RGB or GRAYSCALE)

输出：正在加载返回True，否则返回False