ML-Agents入门级使用方法

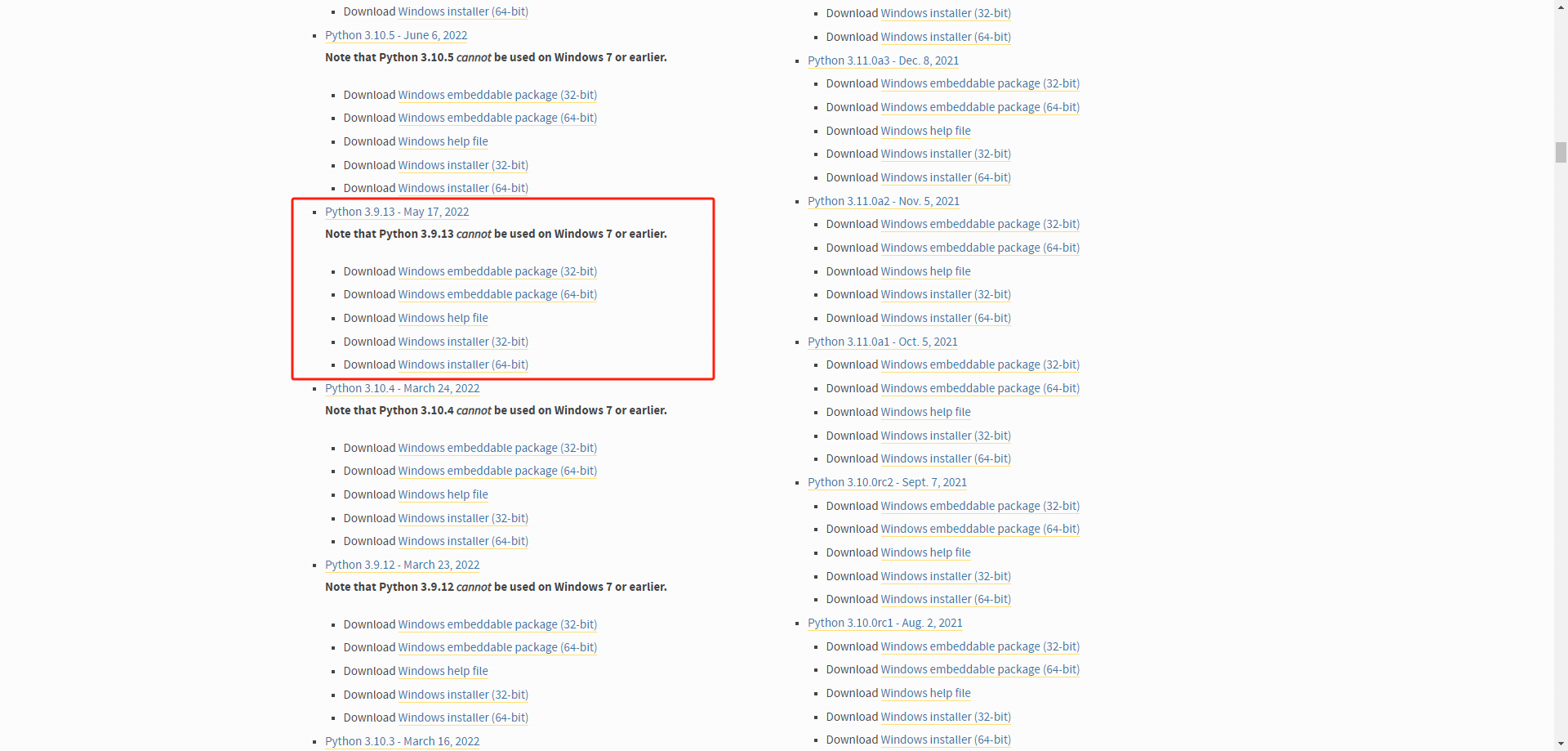
参考视频：<https://www.youtube.com/watch?v=RANRz9oyzko&t=906s>

unity文档：<https://docs.unity3d.com/Packages/com.unity.ml-agents@1.0/api/Unity.MLAgents.Academy.html>

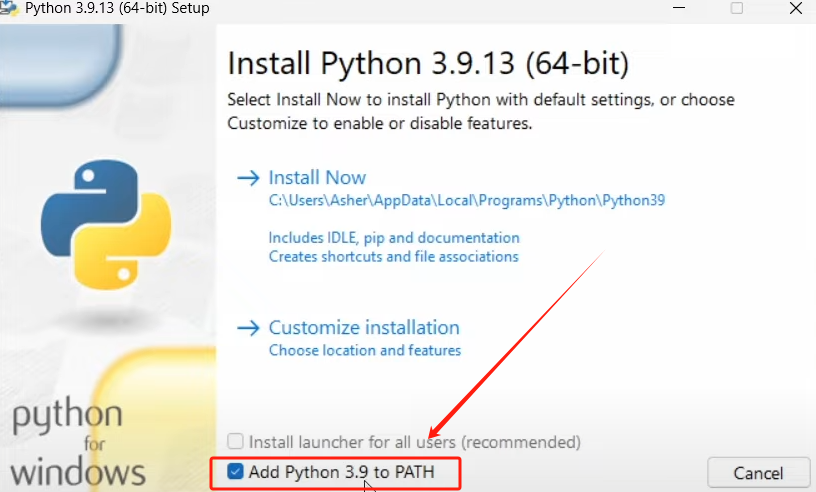
Git Hub：<https://github.com/Unity-Technologies/ml-agents>

## Python安装

要运行ML-Agent需要Python，从<https://www.python.org/downloads/>下载Python，必须3.9.13否则无法使用（2023年12月22日）。



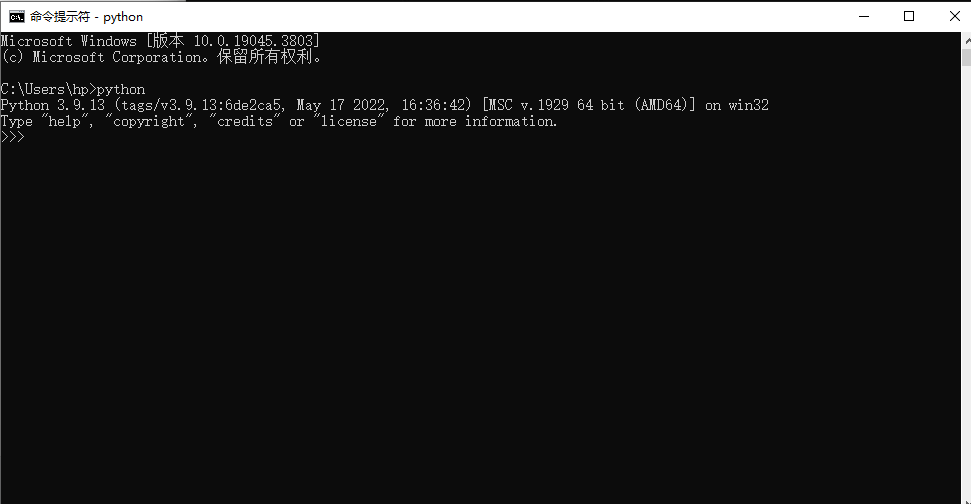
下载后直接双击安装即可，注意安装时需要勾选“Add Python 3.9 to PATH“，安装时可自动配置环境变量。



若环境变量没有配置成功，需手动配置，手动配置即在安装路径中将文件目录和文件目录下的Scripts文件添加到环境变量中即可。



在cmd中输入python查看是否安装成功，若弹出微软商店，可通过输入py查看，与python一样，安装成功应如图所示。



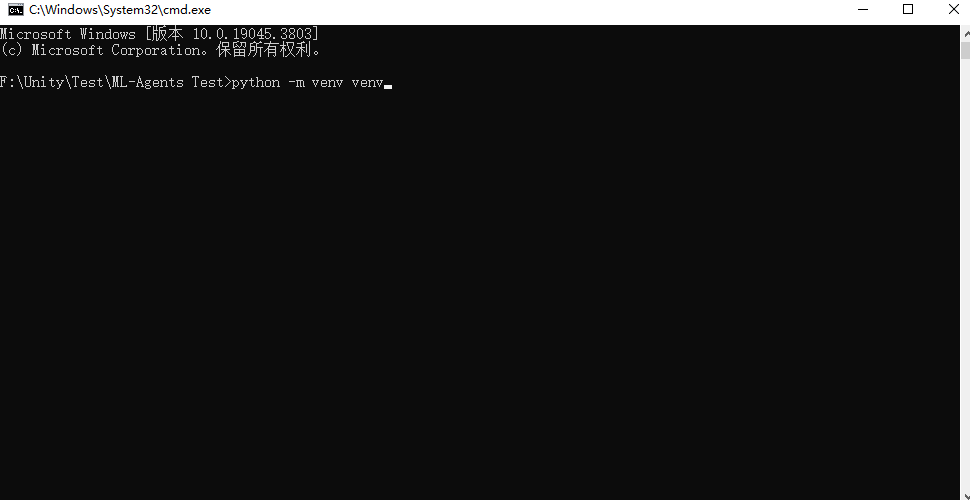
## 环境配置

1. 创建虚拟环境，让ml-agents运行在虚拟环境中，创建Unity项目，在项目根目录中运行cmd命令行。输入

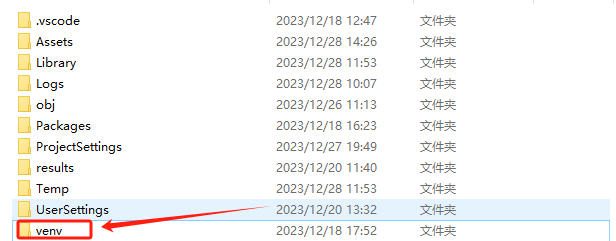
python -m venv 文件名

或 py -m venv 文件名

创建虚拟环境，我在这里直接使用了venv作为文件名。



执行完后会在当前文件夹中创建一个venv文件夹。



1. 虚拟环境就创建完毕，接下来运行虚拟环境，直接运行venv文件夹中的Scripts文件夹下的activate，即使用命令：

venv\Scripts\activate

执行后会在路径前出现虚拟环境的名称。

手机屏幕的截图

描述已自动生成

1. 升级python的pip包，命令为：

python -m pip install –upgrade pip

1. 执行后会升级pip包，通过pip安装mlagents，命令如下：

pip install mlagents

1. 安装后再执行以下命令，安装torch

pip3 install torch torchvision torchaudio

1. 安装好后执行以下命令，这里是更换一下protobuf的版本，mlagents支持的版本：

pip3 install protobuf==3.20.3

1. 运行命令

mlagents-learn -h

查看是否可以运行，若成功运行会展示mlagents的帮助界面

文本

描述已自动生成

1. 如果报错，大概率是缺少包，根据报错信息使用

pip install 包名

命令安装缺失的包即可。

## Unity中加入包

在package manager中选择unity Registry，搜索ML-Agents导入即可。

图形用户界面, 文本

描述已自动生成

## 使用方法介绍

说明：训练模型需要python环境，使用加减分进行训练，在unity中自行定义加减分机制进行训练，运行时需要同时在后台先使用python运行mlagents-learn。

### 脚本方法重要介绍

1. 负责加减分逻辑的脚本需要继承Agent类。

文本

描述已自动生成

1. 通过重写OnActionReceived方法，获得AI传来的数据

文本

描述已自动生成

1. 通过重写CollectObservations方法将数据传给AI

文本

描述已自动生成

1. 通过调用SetReward方法，进行加减分操作

文本

描述已自动生成

1. 上图中有EndEpisode方法的作用是结束本轮训练，不是结束整个训练，而是结束这一次，相当于在加减分后，结束本次。有种类似于结束这节课，开始下节课的感觉。
2. OnEpisodeBegin重写方法可以在训练开始时做出一些操作，该方法在开始一次训练时被调用，可以用于初始化。

### 组件介绍

在unity中将脚本拖到游戏对象上，可以看到还会自动添加一个名为“BehaviorParameters”的组件。

图形用户界面

描述已自动生成

该组件中几个比较重要的参数如下：

图形用户界面, 应用程序

描述已自动生成

1. Space Size是观测数组的大小，可以理解为AI接收的数据是个数组，而这个参数是设置数组的大小所用。需要注意的一点：比如像Vector3这类的数值，占用三个位置，也就是如果只传一个Vector3类型的数值，需要定义的数组长度为3。
2. Actions是AI给出的值，下面有两个数组：

actions.ContinuousActions数组来记录所有连续型的操作，中数组内的每一个值都在[-1,1]之间。

actions.DiscreteActions 数组记录所有离散的操作，每个值应该是0到x中的整数，x表示该操作的所有可能性的总数（跳跃操作的话可以分为跳或不跳x为2，如果是移动的话可以分为不动、上、下、左、右，x为5）。

电脑游戏的截图

描述已自动生成

也就是说，通过重写CollectObservations传入的数据就是给到一个Space Size大小的数组，具体方法是sensor.AddObservation(数值)，sensor是该方法的参数。如传入两个位置。

文本

描述已自动生成

通过重写OnActionReceived而得到得数据就是action（action为该方法的参数），action下有两个数组，即ContinuousActions与DiscreteActions，对传回的数据进行操作即可。比如通过传回的数据控制位置：

手机屏幕的截图

描述已自动生成

1. Behavior Type是指行为模式，有三个选项，

图形用户界面, 文本, 应用程序

描述已自动生成

正常训练使用Deafault即可，而Heuristic Only则是用手动去顶替AI的输入，可以在后台不运行模型的情况下使用，顶替的方法是在脚本中重写Heuristic方法，如我用输入代替AI的输入：

电脑屏幕的截图

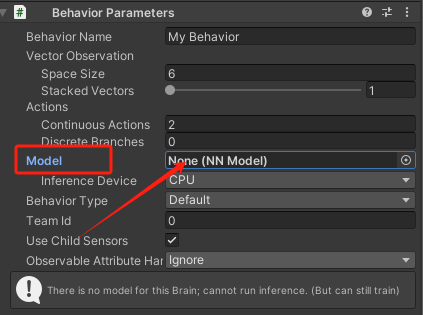
描述已自动生成

## 开始训练及训练结果

在命令行中运行设置的虚拟环境，输入mlagents-learn --run-id=【训练id】就可以开始训练啦，运行后，在unity中运行即可。



训练之后会在目录出生成一个results文件夹，该文件夹下有训练后的数据，找到.onnx的文件，这就是训练的结果，将其拖进项目的Asset文件夹后将其拖到“BehaviorParameters”脚本的model处，即可使用。



如果在一次训练后继续训练mlagents-learn --run-id=【训练的id】 --resume即可。

ML Agents还提供了可视化窗口，在训练时，再次打开一个cmd窗口，依旧是进入虚拟环境，运行如下命令：

tensorboard --logdir results

运行后在浏览器中打开localhost:6006网址即可查看。

文本

描述已自动生成

