# AI玩超级马里奥(super-mario)

## 摘要：

通过课程学习了有监督学习和无监督学习的深度学习，为了满足好奇，自学了强化学习的相关知识，做了AI玩Mario的一个小项目。介绍了强化学习中Agent, Environment, State, action, policy function 的概念；通过OpenAI-Gym这个工具包引入并初始化了环境，并对环境进行了灰度处理、矢量化处理和4帧堆叠的环境预处理。调用Stable-baselines3库，采用PPO算法的CnnPolicy的网络进行学习，每隔50000次保存一次模型，共跑了5M次，最终得到较好的训练结果。

## 引言：

通过课程学习了有监督学习和无监督学习的深度学习，但我一直比较好奇的是，如果追求的长远的收益，没有直接的数据来训练神经网络，追求的是最后的胜利，比如下围棋、象棋，暂时劣势没事只要最后赢了就行。所以我了解到了强化学习，学习了相关知识，并且由于我比较喜欢玩Mario游戏，最近新游戏Mario-wonder发售，我打算做一个ai玩Mario的项目来展示一下。

## 设计过程：

### 强化学习介绍

先介绍一下强化学习的一些概念

Agent,智能体：指的就是任何独立的能够思考并可以同环境交互的实体，很抽象的一个概念。以Mario游戏举例，我们操作的Mario就是Agent

Environment,环境：指的是能够与智能体进行交互并使智能体得到学习的世界，比如Mario游戏中的每一关都是一个环境

State,状态：指的是环境当前的特征，环境所能处于的所有状态的集合称为状态空间，比如Mario游戏中的每一帧的图像就是一个状态

Action,动作：指的是智能体对环境的响应，智能体所能做的所有动作的集合称之为动作空间。比如Mario游戏中Mario的‘跳’，‘向右走’等等就是动作

Reward,奖励：很重要的一个概念，奖励就是环境对智能体的反馈。智能体正是通过奖励才能与环境进行交互和学习，才能知道应该做什么动作。奖励是人为设定的，奖励设定的合理与否能直接地影响智能体的学习效率。比如Mario游戏中，目标是在不死亡的前提下更快地到达终点，在本项目中设置的奖励分为三部分。一是Mario向右走就会获得奖励；二是由于游戏有倒计时的机制，时间到了就结束了，所以每过1s就会获得一定的负奖励；三是Mario

碰到怪物或掉入坑中死亡就会获得一个非常大的负奖励。

Policy,策略函数,：指的是在状态s下采取动作a的概率，比如Mario游戏中在某一帧的画面中，Mario采取各种动作的概率可用这个函数表示。策略函数是未知的，如果知道了策略函数，我们要求智能体在状态s下采取概率最大的那个动作就可以了。因此要采用神经网络深度学习，近似出一个策略函数。

### 安装Mario环境并进行初始化

Open-ai官方编写了Gym工具包，它包含了各种模拟机器人(比如：四足机器人，直立行走的人)和各种复古的像素风的游戏，用来给开发者学习和比较不同的强化学习算法。如图1所示，这是一个直立行走的人的模型的环境，在未经过学习时，他可能直接就会摔倒或者走几步就会摔倒。但经过一段时间的学习，他就会走的步数更多，坚持的时间更长，速度也会更快(跑起来)。



图1 Open-ai-gym

本项目就是基于Open-ai-gym创建的Mario环境。链接：[gym-super-mario-bros · PyPI](https://pypi.org/project/gym-super-mario-bros/)

访问链接可以得知gym-super-Mario-bros包需要基于Nintendo Entertainment System (NES)的python版本的模拟器，所以首先要pip install nes-py 然后pip install gym-super-mario-bros, 就成功引入了Mario的环境。访问链接可以得到Mario训练前运行的一段代码，其中Mario的动作是随机的，它的动作空间有256个值，但实际用手柄操作时没有这么多值，为了简化操作，引入nes\_py的wrappers的JoypadSpace把Mario的动作空间简化为7个值(对应‘不动’，‘向右走’，‘向左走’，‘原地跳’，‘向右跳’，‘向右加速走’，‘向右加速跑跳’)。

在Mario游戏中环境由管子、栗宝宝(如图4所示的怪物)、砖块等组成。在状态下，Mario会按照算法的预测值做出动作，Mario给环境反馈。然后Mario做出动作，环境会得到下一个状态，奖励，环境给Mario反馈，一直反复直到达到训练次数，如下图2，3所示。正是通过这样的方式，使得Mario不断与环境交互，通过奖励得到学习，不断试错，最终学会正确的通关方式



图2 智能体与环境交互图

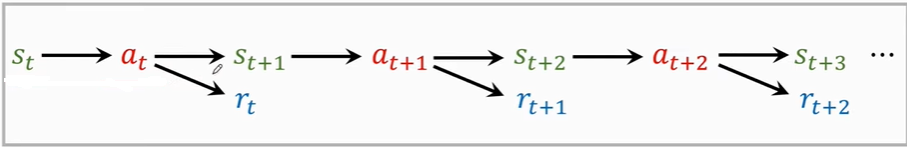


图3 迭代图

运行代码可以看到如图二所示，此时的Mario还未经过训练，还没有学习到游戏的一些技巧无法跳过这个水管



图4 Super-Mario运行图

### 预处理Mario环境

预处理环境和之后的训练模型中，需要用到stable-baselines3这个库。stable-baselines3是一个功能很强大的深度强化学习库，它基于pytorch建立，能够快速实现各种强化学习的算法和包装环境。

环境会给Mario一个状态，每个状态就是当前帧的图像，state.shape 后可以看到state由240\*256\*3的数组组成，这些信息比Mario需要的更多，我们可以使用Wrappers包装器在环境将信息发给Mario前对环境进行预处理以此来减少运算量。

首先state是彩色图像，但Mario的动作并不依赖于管道或天空的颜色，所以我们可以使用GrayScaleObservation来将RGB图像转换为灰度图像。处理后state的shape为240\*256\*1，计算量直接变为了原来的1/3而不会丢失有用的信息。彩色图像和灰色图像的对比图如图5所示。而且为了获得Mario在训练中的平均奖励和平均存活时间，要使用Monitor包装器来包装环境。

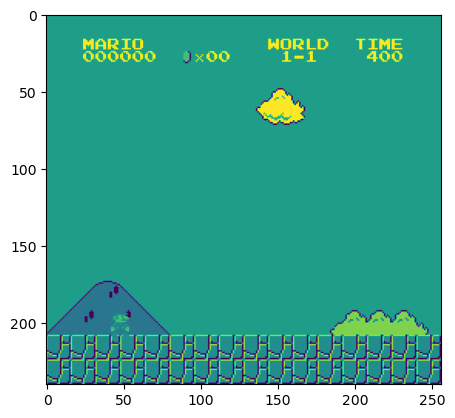
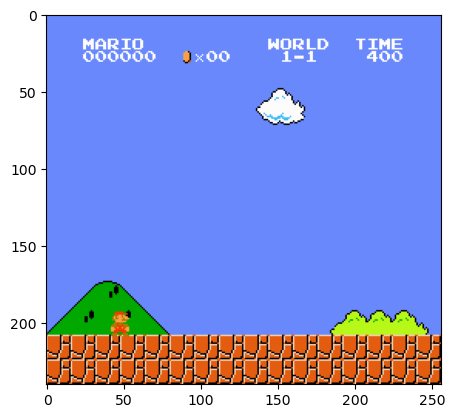


图5 彩色和灰度图像对比图

其次，要进行矢量化环境，它可以将多个独立环境堆叠到单个环境中。矢量化环境不是每一步在1个环境中训练Mario，而是可以每一步在n个环境中训练Mario，这使得Mario可以更快地收集经验，进行学习和Mario收集和学习到的经验会包含更多样化的状态，有利于提升Mario的探索性。在本项目中使用DummyVecEnv来矢量化环境，它会在同一进程上运行所有环境，DummyVecEnv会比SubprocVecEnv更快。矢量化环境后，state的shape变成了1\*240\*256\*1。

最后，使用FrameStack包装器，将环境的连续4帧堆叠到一个状态中来反馈到学习模型中。这样，我们就可以根据Mario在前几帧中的运动方向来判定Mario是在跳跃还是在着陆。如图6所示。

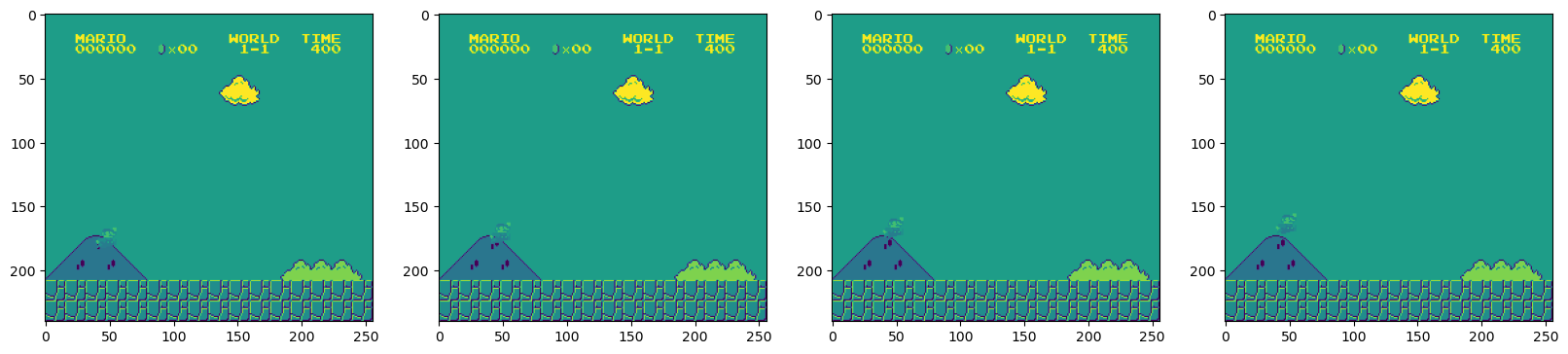


图6 连续4帧堆叠到一个状态

### 训练模型

强化学习学习的方式可以分为价值学习，策略学习，策略+价值学习，这里用的是策略+价值学习，调用stable-baselines库中的PPO(proximal policy optimization)算法函数和callback召回函数。查询Stable-baselines3官网中的策略网络架构如图7所示。[Policy Networks — Stable Baselines3 2.2.0a9 documentation (stable-baselines3.readthedocs.io)](https://stable-baselines3.readthedocs.io/en/master/guide/custom_policy.html)

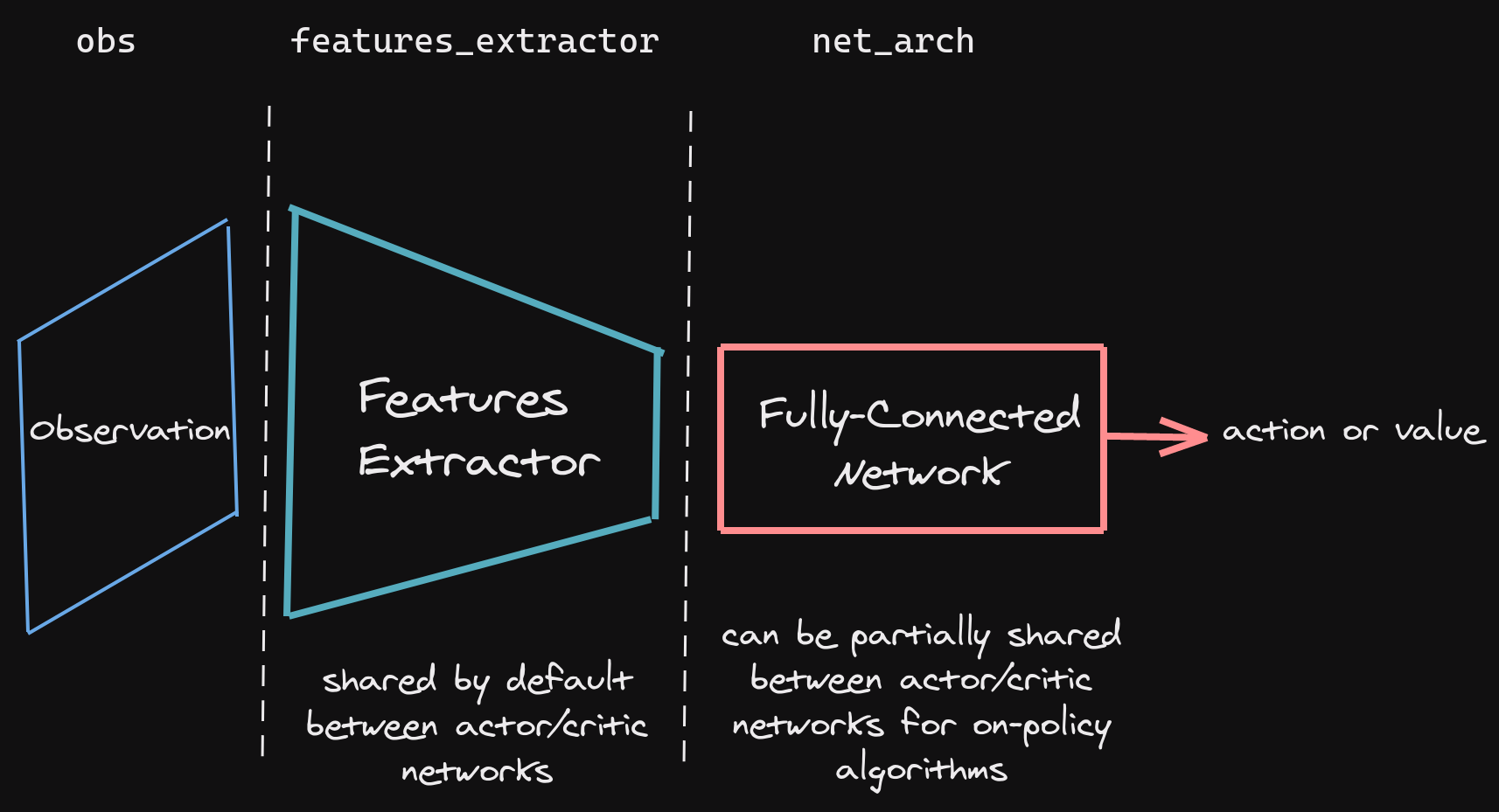


图7 策略网络架构

策略网络输入为状态s(state 或上图中的observation),经过特征提取层(若干层卷积层)转化为特征向量，比如Mario游戏中将需要用CNN从图像中提取特征。再经过全连接网络转化为动作action和价值value，从而达到了通过策略网络近似出策略函数，输入为状态state，输出为动作action和目前状态的价值value。

通过print( model.policy )打印了stable-baselines3库中的PPO算法CnnPolicy的网络架构如下图8所示。

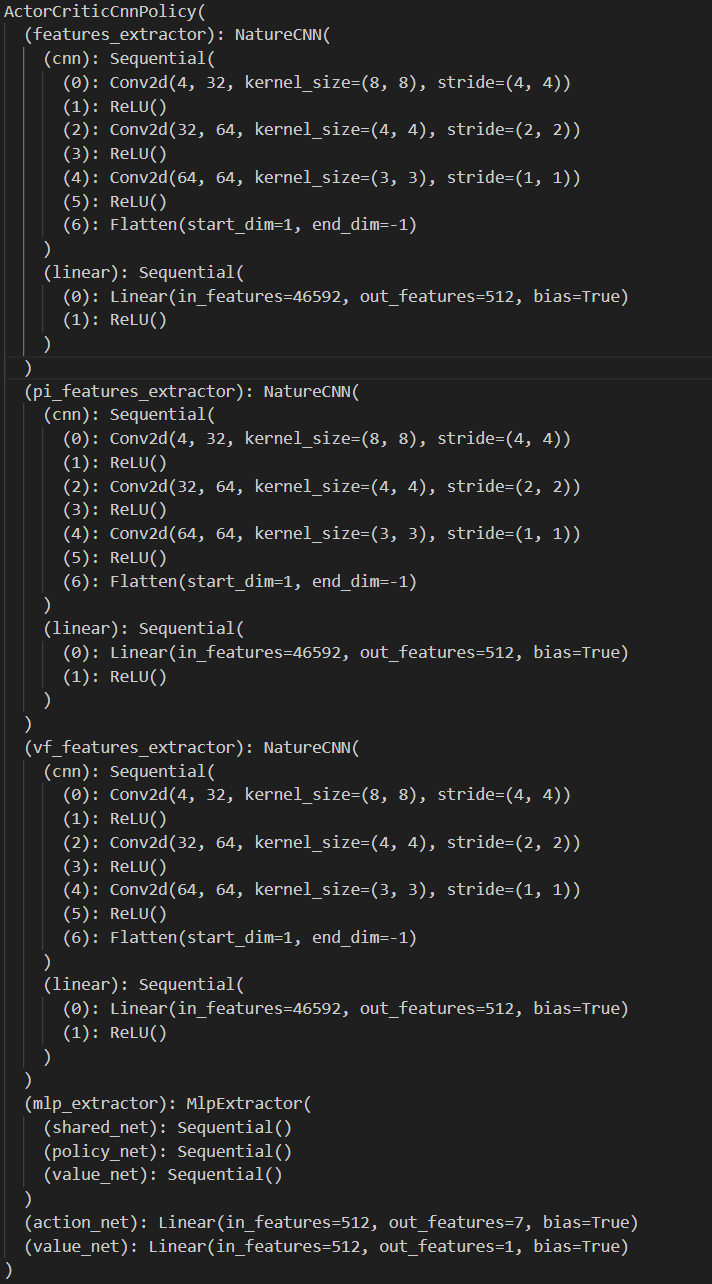


图8 PPO算法CNNPolicy的网络架构

继承BaseCallback基类并创建TrainAndLoggingCallback，每跑50000次保存一次模型，学习率为1e-6，n\_step为512，一共跑5M次。训练中的一些参数通过tensorboard读取，如图9所示

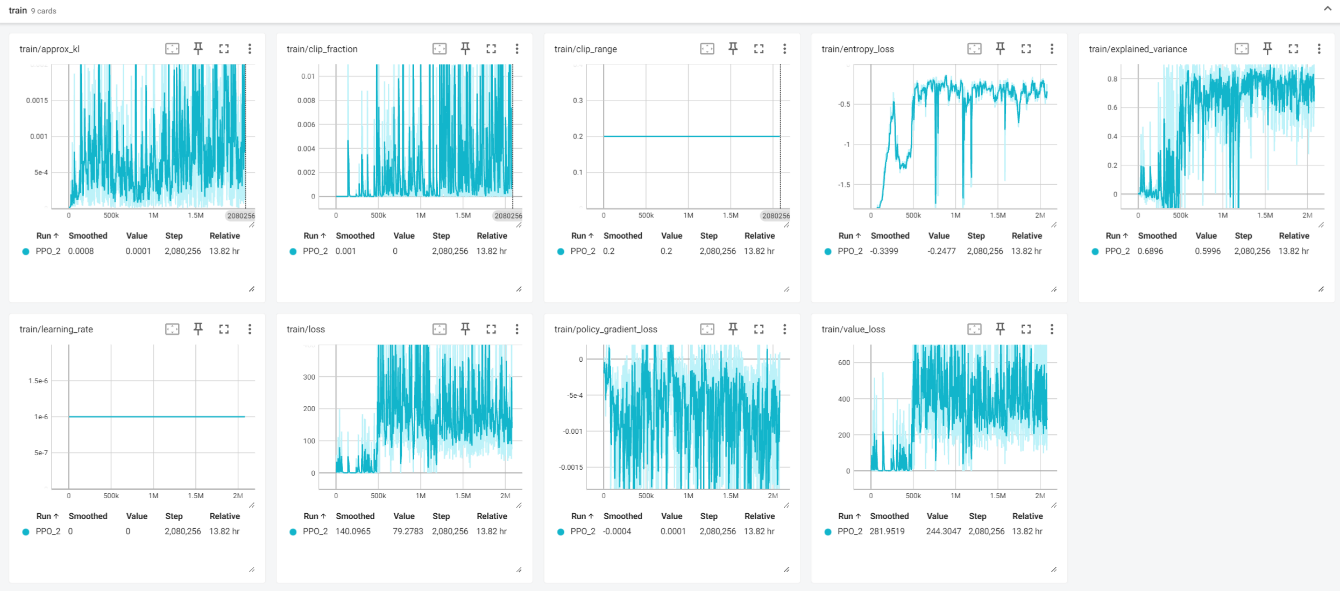


图9 训练过程中的一些参数

### 测试模型

按照上述1、2操作，初始化和预处理环境，然后PPO.load()读取训练好的模型，动作action为模型基于状态state的预测值model.predict(state)，通过env.step()和env.render()显示AI自动玩Mario的视频，在此不方便放视频展示，训练效果可见展示ppt和mario.mp4视频。运行代码时先运行mario-train.ipynb，开始训练，跑的模型会存到目录下train文件夹下，日志会存到logs文件夹下。再运行mario\_test.ipynb，读取要运行的模型，结果会从窗口显示。

## 心得与体会

1. 最大的一点感触就是自学过程中，大部分资源都是英文的，官方文档和介绍视频，还有遇到问题的交流论坛，一定要好好学英文啊！

2. 网上自学资源真的很多，目前还只可以调库，创建自定义的网络，奖励reward和自定义的召回函数还没尝试过。调库的话就要多去看官方文档和官方提供的视频

3. 找一个有兴趣的项目学习，会比较有动力，过程遇到问题，不断地去google学习，还是挺有收获的。

参考文献：

[1] [gym-super-mario-bros · PyPI](https://pypi.org/project/gym-super-mario-bros/)

[2][Policy Networks — Stable Baselines3 2.2.0a9 documentation (stable-baselines3.readthedocs.io)](https://stable-baselines3.readthedocs.io/en/master/guide/custom_policy.html)

[3] [Train a Mario-playing RL Agent — PyTorch Tutorials 2.1.0+cu121 documentation ---](https://pytorch.org/tutorials/intermediate/mario_rl_tutorial.html)

[4] [An introduction to Policy Gradient methods - Deep Reinforcement Learning - YouTube](https://www.youtube.com/watch?v=5P7I-xPq8u8)

[5] paper: <https://jmlr.org/papers/volume22/20-1364/20-1364.pdf>

[6] <https://github.com/DLR-RM/stable-baselines3>]