《人工智能实践》期末作业——兵棋推演实验报告

小组成员：

姓名	学号
俞泽斌	20337263
储致远	20337184
张祺	20337267

实验在兵棋推演环境中进行。实验内容为基于 QMIX 算法，根据结果进行分析，整理撰写该实验报告。

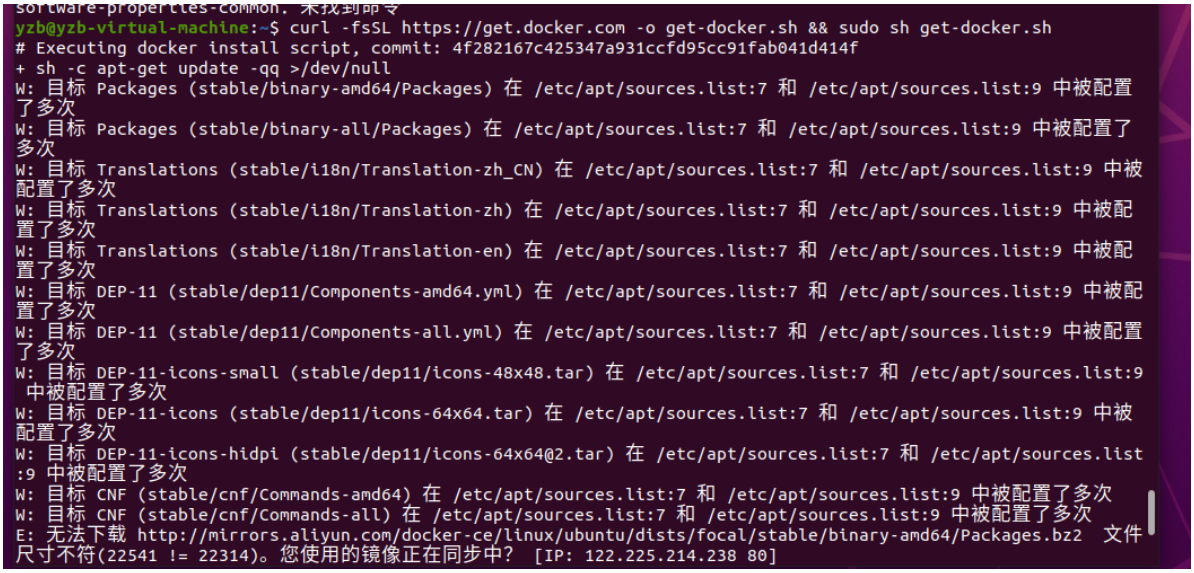
实验步骤

首先是进行环境配置以及下载课题指导提供的代码。

在Linux虚拟机环境下安装docker

```
curl -fsSL https://get.docker.com -o get-docker.sh && sudo sh get-docker.sh
```

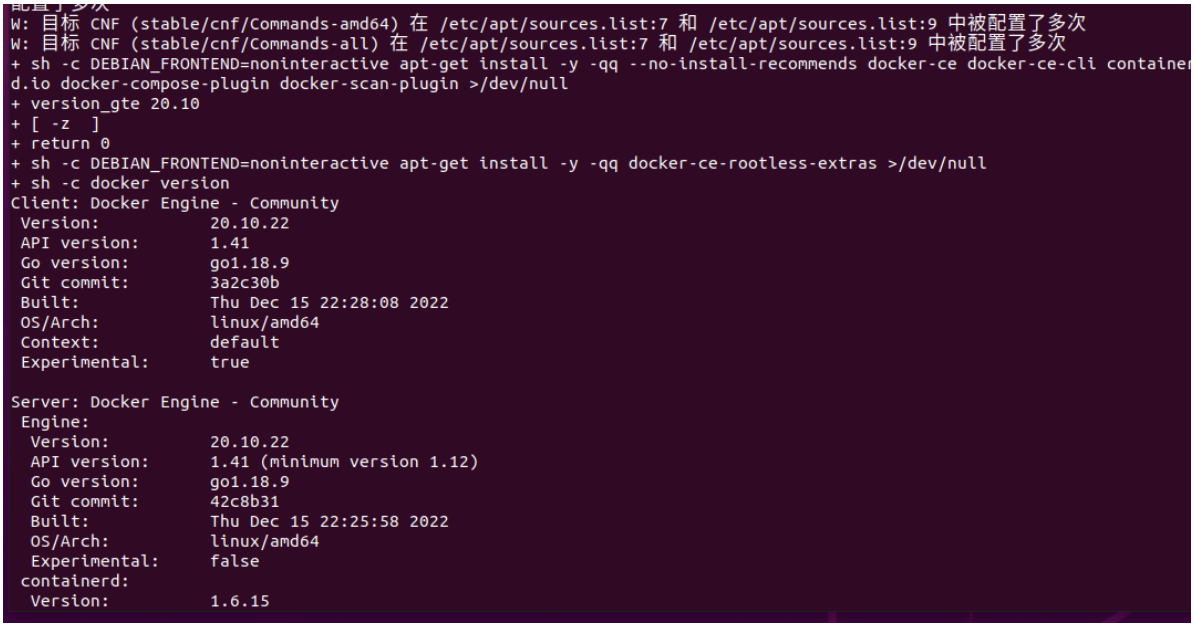
遇到文件尺寸不匹配的问题



换源后继续环境配置

```
sudo sed -i 's/aliyun.com/ustc.edu.cn/g' /etc/apt/sources.list
```

完成安装：



输入命令下载打包好的镜像

```
docke pull algernon6/homework:latest
```

但是在使用docker pull下载环境，速度受限，课题指导师兄将镜像重新上传到阿里云，重新尝试：

```
docker pull registry.cn-shenzhen.aliyuncs.com/algernon6/homework:latest
```

能够比较方便的下载好打包的镜像

输入命令运行镜像，镜像运行后得到容器

```
docker run -it --rm -v ~/data/POAC:/workspace algernon6/homework:latest
```

~/data/POAC 为电脑中代码的位置，/workspace 为容器内的位置。--gpus all 为允许容 器使用 gpu，由于本次实验机器性能限制，再虚拟机中构建docker环境，使用CPU进行本次实验。

实验使用QMIX算法，使用的任务难度为bq，运行容器后，在容器内运行代码

```
python3 src/main.py --config=qmix --env-config=bq --seed_y=1
```

由于实验时间过久，在reward图横坐标250左右停止运行。

实验所用的任务场景为兵棋推演。共有坦克、战车、步兵三个兵种，不同兵种属性不同 （具体属性值可参考对应论文）。最终需要己方的坦克、战车、步兵打赢对方的坦克、战 车、步兵。获胜标准有两个：一是游戏结束时己方总血量比对方高；二是对方血量为零。训练算法得到训练图像（共有三张：奖励曲线，第一个获胜标准，第二个获胜标准）。

实验原理

实验内容为基于 QMIX 算法

QMIX算法是一种基于集体行为的强化学习算法，它用于解决多智能体协作问题。QMIX算法通过学习一组质量度量来预测群体行为，并通过调整每个智能体的策略来达到协作目标。

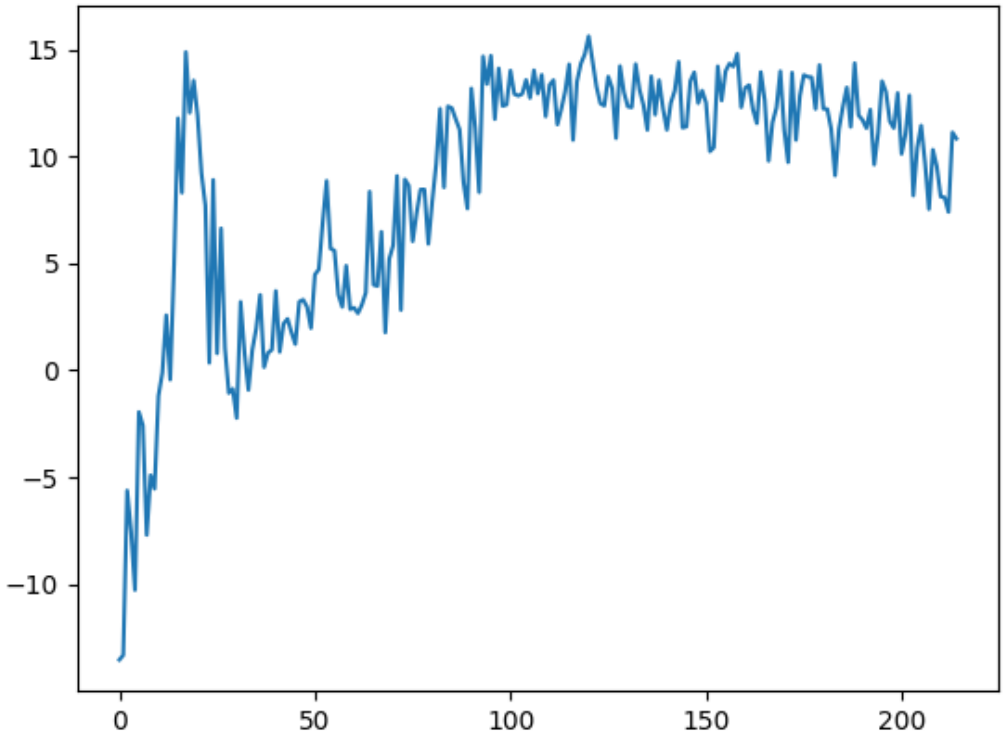
其整体上是在VDN基础上进行提高，VDN只是将每个智能体的局部动作值函数求和相加得到联合动作值函数，也没有在学习时利用状态信息。QMIX主要贡献在放松了VDN的动作价值函数计算方法，用了一些数学手段让agents的状态动作空间随着agents数量线性增长，并且使得值函数的分解变得更加合理。提高了多智能体系统下强化学习算法的准确度。QMIX采用集中式训练、分布式执行的架构，即CTDE，并且同VDN一样，也是所有agents共同维护一个奖励函数。

算法整体上使用了一个由 *agent networks*，一个 *mixing network* 和一组 *hypernetworks* 构成的架构。核心部分是一个Q-MIX层，它将所有智能体的策略进行组合并产生一个集体策略。这个层使用权重来组合各个智能体的策略，并在训练过程中不断调整权重来提高集体行为的质量。为了加强单调性约束，Mixing network 的权值被限制为非负值，这种单调性的方式限制更少，表达总体个体关系的能力更强。而其权重由不同的单个hypernetworks产生，进行的是全局状态和全局Q的衔接。

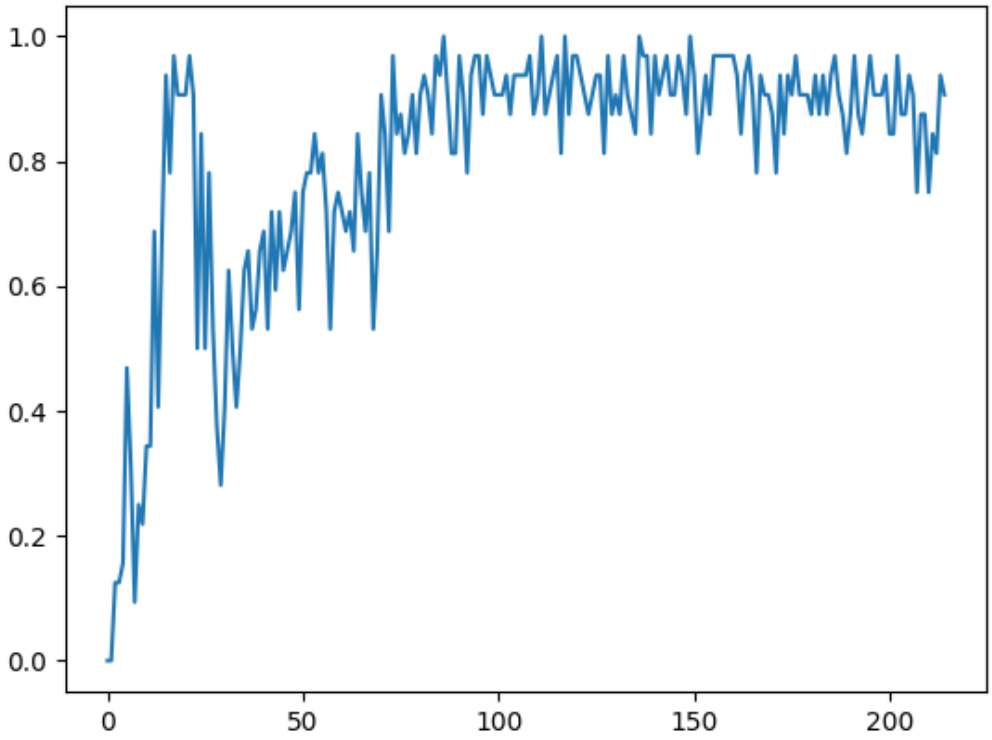
训练过程中，QMIX算法提到了使用权重更新方法来更新每个智能体在集体策略的权重。该算法使用当前策略的质量来更新权重，使得当前集体策略的质量越高，权重就越大，使用权重和当前策略来更新集体策略。

实验结果

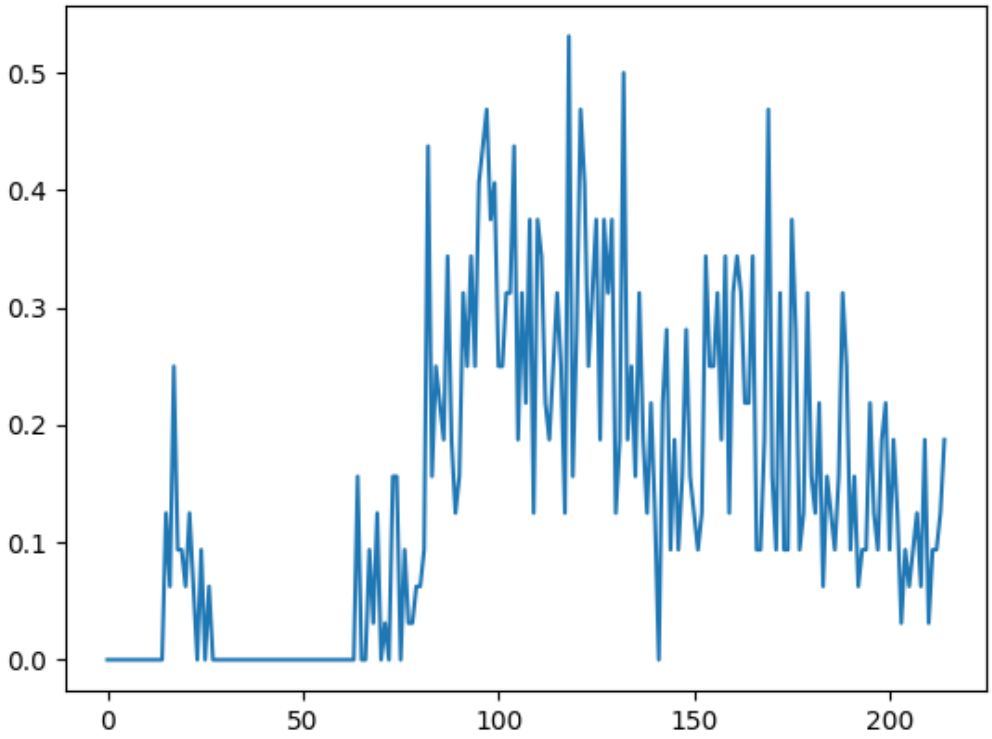
奖励曲线：



第一个获胜标准曲线：



第二个获胜标准曲线：



改进方法

本次实验在QMIX算法上进行，通过修改不同参数进行改进，通过训练算法得到训练图像像（共有三张：奖励曲线，第一个获胜标准，第二个获胜标准），并对此进行分析。

1. 修改parallel，调整并行环境数

   将qmix.yaml:runner的值改为"parallel"，即选择并行环境，并调整并行环境数如下：

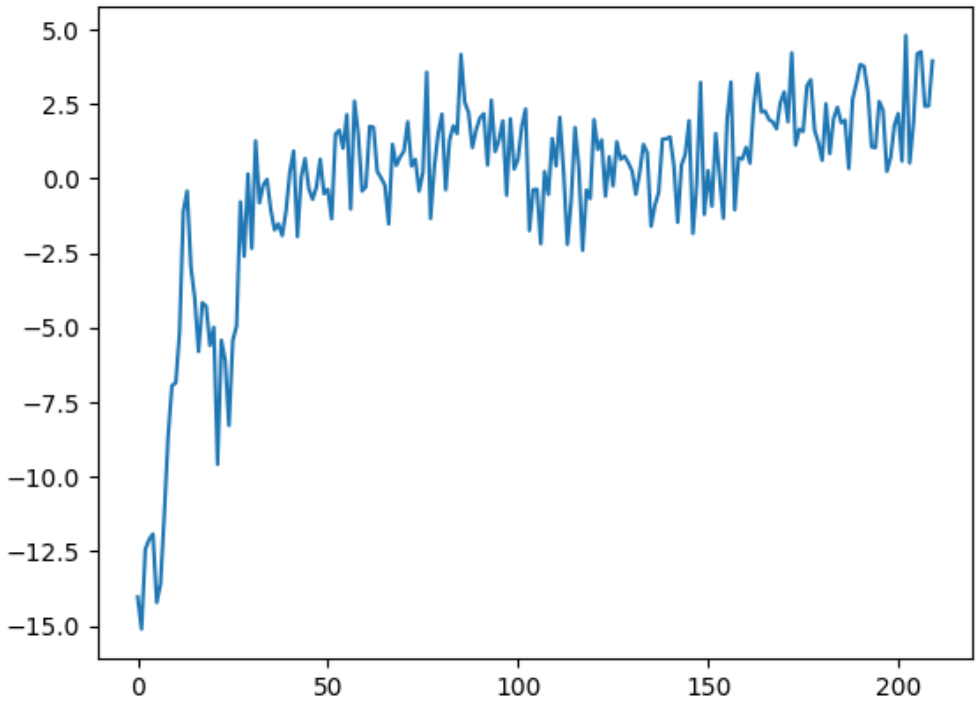
   runner: "episode" -> "parallel"

   batch_size_run: 1->8

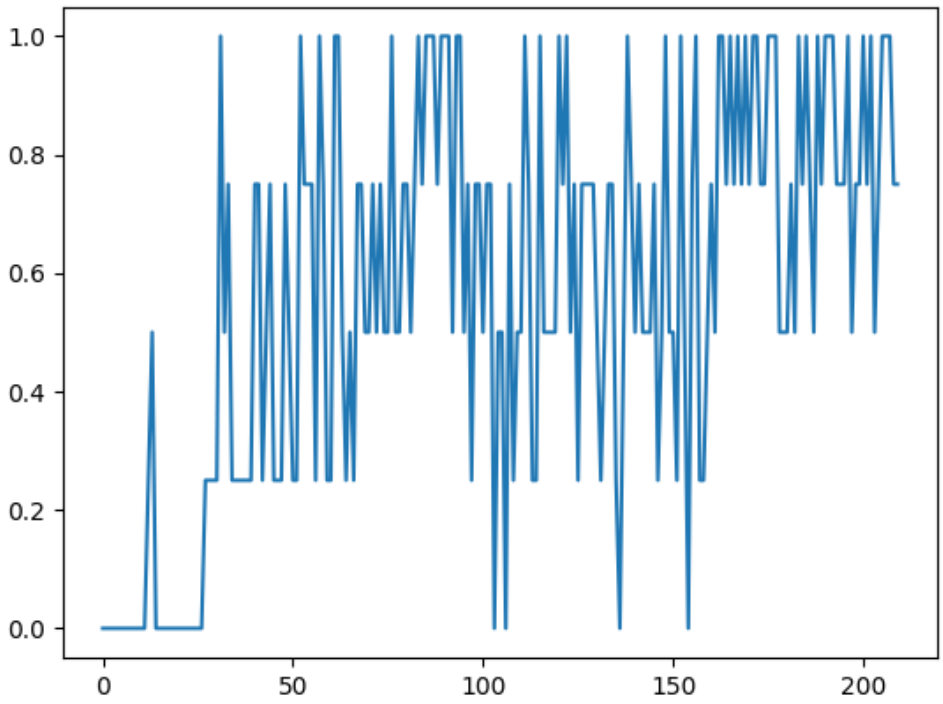
```yaml
runner: "parallel" # "parallel"
batch_size_run: 8 # 8 # batch_size_run=4, buffer_size = 2500, batch_size=64  for 3s5z_vs_3s6z
buffer_size: 5000
batch_size: 32 # 128
optimizer: 'adam'
```

   实验结果：

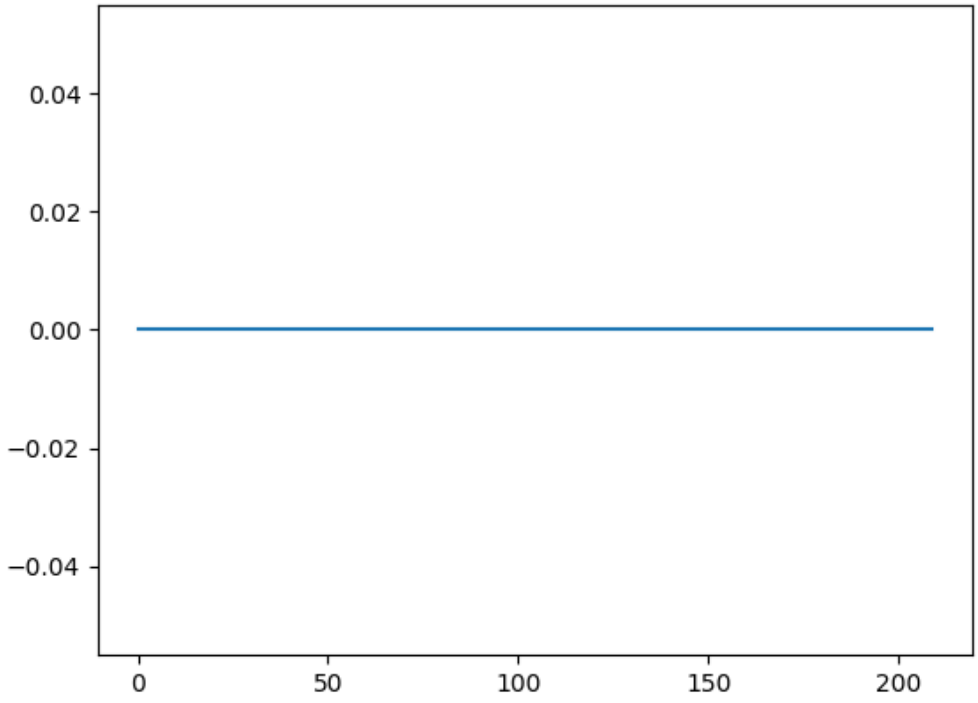
   奖励曲线：



   第一个获胜标准曲线：



第二个获胜标准曲线



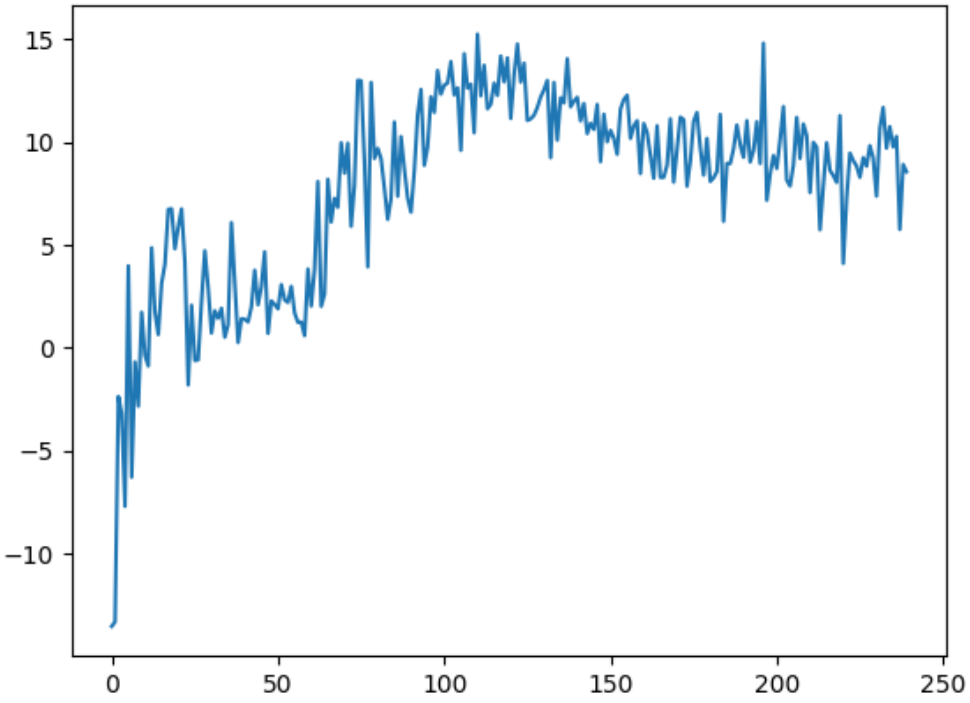
2. 学习率

学习率作为更新网络权值、各个动作价值函数的参数时，每次更新的步长大小，在人工智能实验过程中十分重要，调整到合适的值对良好的实验效果有很大影响，于是我们针对学习率进行进一步实验，调整实验参数如下：

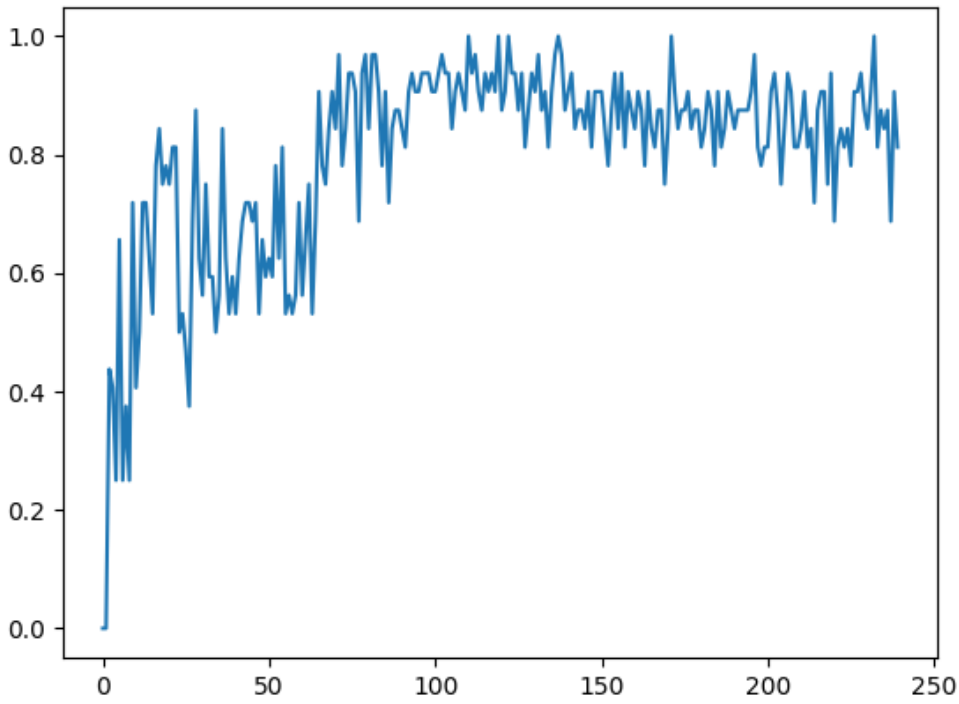
```yaml
learner: "nq_learner"
mixer: "qmix"
mixing_embed_dim: 32
hypernet_embed: 64
lr: 0.001 # 0.001 # Learning rate for agents
td_lambda: 0.6 # 0.3 for 6h_vs_8z
optimizer: 'adam'
q_lambda: False
```

实验结果

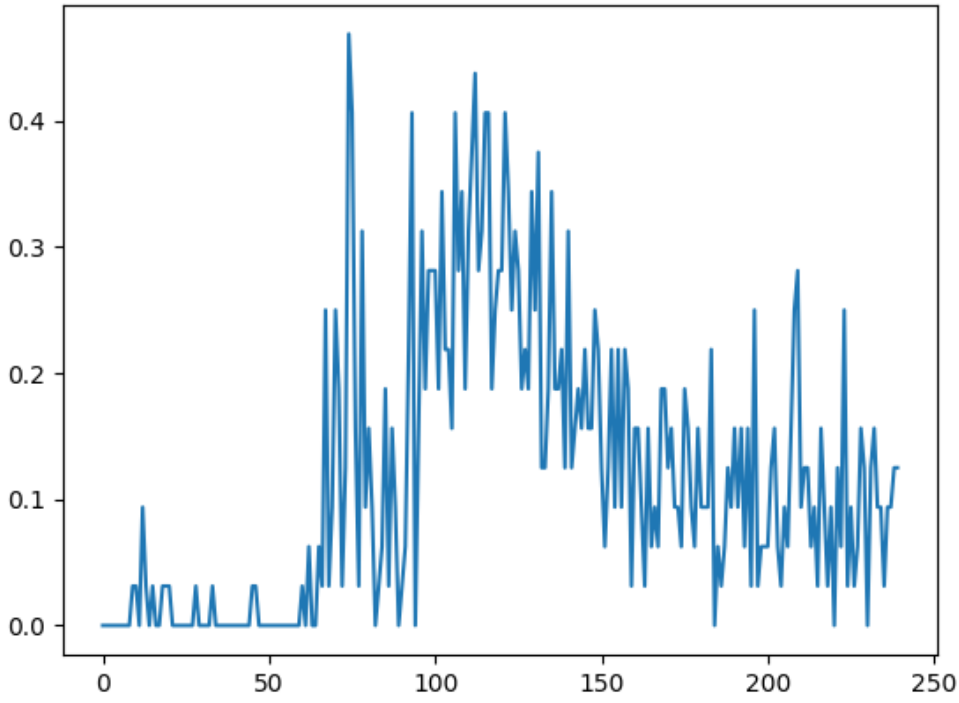
奖励曲线:



第一个获胜标准曲线:


1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250

第二个获胜标准曲线：


0.4
0.3
0.2
0.1
0.0
0
50
100
150
200
250

现象原因讨论和总结

1. 修改parallel，调整并行环境数

   可以看到，reward曲线增长缓慢，横坐标200以内时最高峰值仅为2.5左右，远低于原来的15，但可以看到其增长趋势，在横坐标150以后奖励函数值已经稳稳居于0以上。

   第一个获胜标准曲线波动极大，但是在可见范围内有缩小的趋势并趋近于1，第二个获胜标准曲线毫无波动，始终为0。

   采取了并行环境parallel运行，并设置batch_size_run为8，相较于原来的单个episode采集数据，运算量增大。

   奖励函数为己方总血量的减少值加上对方总血量的减少值，可以看到改进后的算法拥有更多的并行环境后采取更加求稳的胜利，放弃第二种获胜标准，即对方血量为0，而是采用更容易达到的第一种获胜标准，游戏结束时己方总血量比对方高。

   这种改进方法对于实验机器性能有一定要求，在进行的最简单难度bq的实验上无明显优势，在进行更高的难度，如bqhard3,bqhard4难度上，则会体现出显著优势。

2. 将学习率从0.005修改到0.001

   将学习率降低后，每次更新权值的步长也变小，网络权值变化的速度就会变慢。这意味着在训练过程中，网络权值需要更多的训练步骤才能达到最优解。

   在QMIX算法中，学习率是用来更新各个动作价值函数的参数的，学习率较小可能导致算法在训练过程中不能快速收敛。因为QMIX算法是一种基于Q-learning的多智能体协作学习算法，其目的是通过训练学习到多个智能体之间的协作策略来达到最优策略。在学习过程中，智能体需要快速学习到最优策略，而学习率较小会导致智能体在训练过程中不能快速学习到最优策略，从而影响最终的训练效果。

   而根据调整后的实验结果，在reward和第一个获胜标准图中可以看出的较大差别是：相比最初结果，没有在25次迭代左右获胜概率快速升高后又快速下降的情况，而学习过程更加平稳线性，分析如下：

   当将学习率从0.005调低到0.001后，由于学习率较小，权值更新的步长也变小，网络权值的变化速度也变慢。这意味着网络需要更多的训练步骤才能学会训练数据中的规律。因此，网络的学习过程变得更加稳定，更少的参数更新，就能够收敛了。因此，在训练过程中，reward和获胜概率变得更加线性上升，而不是像之前那样来回波动。而相比学习率较大时的模型初始情况，网络权值的变化较快，可能导致算法在收敛之前就过度更新了权值，导致过拟合、无法收敛等，直到更多轮次后又重新学习了规律，重新升高了reward，而降低学习率相对避开了该过程。

   但从实验结果同时可以注意到第二个获胜标准的成绩有所下降，最高值下降比例甚至在0.1左右，则可能因为网络可能需要更长的时间才能学会如何达到第二个获胜标准，从而导致结果下降。

不同获胜标准的差异是我们实验过程中所关注的，通过两个实验结果也可以看出不同的获胜标准体现出了对模型的不同要求，标准的设定在我们进行人工智能深度学习的实验中非常重要。

以本次实验过程为例，对算法进行不同方向的修改后，两个获胜标准展现出的实验结果变化也不尽相同，第一个获胜标准是在游戏结束时己方总血量比对方高，这要求模型在训练过程中学会在保持自己总血量高的同时进行攻击。而第二个获胜标准是对方血量为零，这要求模型在训练过程中学会更有效率地攻击对方。

调整参数的过程让模型的在不同维度的表现的变化趋势有所差异，在结果也对应表现出来。例如，我们因此在实验结果的分析中可以看到，在调整学习率为0.001后，第一个获胜标准可能更容易达成，而第二个获胜标准可能更难。