强化学习：作业一

黄彦骁 502023370016

October 7, 2023

1 作业内容

在“蒙特祖马的复仇”环境中实现Dagger算法。

2 实现过程

最开始观察代码的实现思路是Beheavior Cloning的想法，由人来游玩游戏，利用正确游玩游戏的路径对应的状态和动作作为训练数据集，让模型在该训练集上进行训练，从而迅速拟合我玩该游戏的动作行为，这样既简洁又省事（偷懒行为）。

然而在针对dagger算法进行分析之后，伪代码如图1所示：

Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in Π.
for $i = 1$ **to** N **do**
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
  Sample T-step trajectories using π_i.
  Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by π_i and actions given by expert.
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
  Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
end for
Return best $\hat{\pi}_i$ on validation.

Algorithm 3.1: DAGGER Algorithm.

Figure 1: Dagger算法伪代码

可以看出dagger算法的行为与经典的Beheavior Cloning并不完全一致，其本质是利用专家的正确行为来对数据集进行有益的增广，从而不断增强模型的能力，同时不全采用模型探索的方式，避免一直生成重复的状态。

第二次实现的思路是在每次探索产生了新的状态之后，人为地对新的状态标记对应的动作标签，最后将状态-动作对作为训练集交给模型训练。然而由于标记状态所需要的时间太长，标记时精力有限，经常会出现错误标记的情况，针对该方法标记了900-1200个有效数据进行训练，然而模型性能没有提升（无法得分），甚至标记的状态也并没有出现较大的改变（即模型的探索基本无变化）。

和多位同学[1]讨论后打算在github上寻找一个可用的专家轨迹来作为我们的专家指导，即为最后使用带参数的Policy Model，由于该专家模型采用的游戏环境与原始代码中包装的环境不一致，通过阅读专家模型的实现和调用代码，针对Env进行了重新包装（主要关键在于专家模型运行的帧数与kuan框架环境不同提取了输出动作所需要的相关函数，均在`Expert/Expert.py`中实现，相关调用专家的主函数部分在`main2.py`中实现。

算法最后实现如下：模型采用pytorch自带resnet18进行训练，batch_size大小为400，每次训练10个epoch；由于数据不断增量后后续训练所需要的时间开销太大，在dataset size达到10000后将其维持在这个数量；训练所用显卡为GTX2080。

3 复现方式

在主文件夹下运行 `python main2.py`.

4 实验效果

见图 2。每次得到一个样本都需要访问一次专家得到对应动作，因此二者大小相同，而累计奖励大致会随着训练量的增大有一定增大，但不够稳定。

5 小结

按照道理采用了专家轨迹进行指导之后，算法应该能取得一个较为不错的效果，但最后的效果图却有不小的模型震荡情况，仔细思考探寻这背后的原因我认为有以下几种可能：在数据集大小达到10000之后，采取的措施为每次随机丢弃掉400个数据，在下一次循环后加上探索的400个数据，将大

[1]李安琦，习三卓

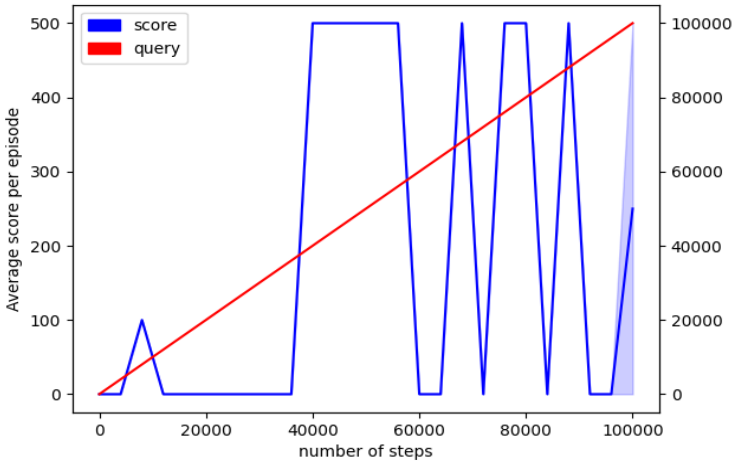

Figure 2: Dagger算法最后效果

小维持在10000，这样的策略可能会导致模型对一些关键数据训练不够，而对一些无用的状态进行了过多的训练，导致模型性能下降；在算法进行探索过程中，采用的是原框架的模型+随机探索，而真正的dagger算法是模型+专家的探索策略，随机探索可能会探索到很多无用状态，对模型性能产生干扰。

思考题：在玩游戏的过程中标注数据与 Dagger 算法中的标注数据方式有何不 同？这个不同会带来哪些影响?

玩游戏过程中标注数据时，标注的标签会对数据后续的状态产生影响，而Dagger算法是在状态全部产生后对其进行标注，你的标注无法影响产生的状态。前者更像一种Beheavior Cloning，如果标注正确能很快的取得较高的reward，但是会对标注的行为过拟合；而后者由于模型探索会遍历更多的状态，算法取得效果更慢，但因为见过更多的状态，不易过拟合。