人工智能与机器学习基础
AI3002-25Fall Final Project

2025 年 12 月 23 日

1 实验主题

本次大作业分为两个阶段：

- **经典强化学习** (Warm-up)：参考附件 warmup 中的内容，学习并实现经典 RL 算法：
  - **Monte Carlo Learning**：Blackjack 环境下实现 MC Learning 的 first-visit 版本（推荐选做 every-visit）。
  - **Temporal-Difference Learning**：CliffWalking 环境下实现 SARSA 与 Q-Learning（推荐选做 double Q-learning）。
- **深度强化学习** (Main Task)：
  - 根据给定输入输出格式，使用 PyTorch API 设计合理的 Actor-Critic 模型并完成训练；
  - 博弈逻辑与环境交互已在 `utils.py` 中实现，但需理解其原理。

2 环境配置

配置 Torch 环境并截图。如果电脑支持 GPU 计算，建议完成相关配置并开启 GPU 加速：

- 对于 NVIDIA GPU，建议完成 CUDA 相关配置并开启 GPU 加速。
- 对于 MacBook（针对 M 系列芯片），支持 MPS（Metal Performance Shaders）加速，可在代码中使用 `torch.device("mps")` 开启加速。

```
# 创建一个虚拟环境
conda create -n ai25 python=3.10 -y

# 安装pytorch（查看https://pytorch.org/get-started/previous-versions 下载对应版本）
pip install torch==2.6.0 torchvision==0.21.0 torchaudio==2.6.0

# 安装其余的依赖
pip install matplotlib tqdm wandb pandas
```

请参考https://docs.wandb.ai/查看如何注册并使用（登陆）wandb，这是一个非常实用的实验记录工具，推荐大家以后跑实验的时候可以多多使用。

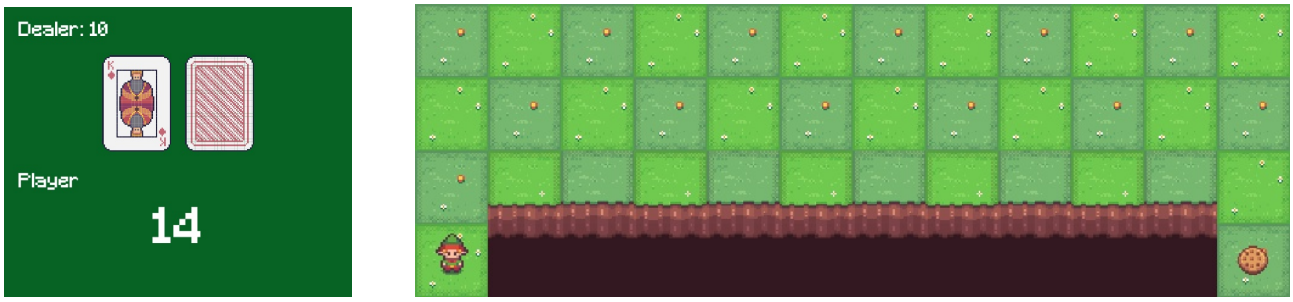
请仔细阅读实验文档、参考材料和 `readme.md` 以理解大体框架。

3 Warm-up：经典强化学习

本阶段用于建立强化学习的基本概念与算法直觉，强烈推荐每一位同学都独立完成一遍 Warm-up，为后续深度强化学习训练与曲线分析打基础。请参考附件内容完成以下任务，并在报告中展示关键结果与分析。

注：这一部分的实验报告中不用贴代码，只需要展示关键结果与结果分析。

使用环境：



Blackjack (MC Learning) Cliff Walking (TD Learning)

图 1: Blackjack（左，MC 方法）与 CliffWalking（右，TD 方法）的环境

依赖安装（仅 Warm-up 需要）

本实验需要安装特定版本的 gym，不同版本的接口可能有所不同，请执行以下命令：

```
pip install "gymnasium[classic-control]"
```

任务 A：Monte Carlo Learning

针对 Blackjack 环境，在 `mc.py` 中实现 MC Learning 的 first-visit 版本。

```python
def mc(env, num_episodes, discount_factor=1.0, epsilon=0.1):
    ...
```

并在报告中展示：

- 不同训练轮数下的价值函数可视化或等价指标（例如 1e4 vs 5e5 episodes 的对比图）。
- 对 first-visit MC 的收敛特点、方差来源的讨论。

任务 B：Temporal-Difference Learning

针对 CliffWalking 环境，在 `sarsa.py` 和 `qlearning.py` 中实现 SARSA 和 Q-learning。

`sarsa.py`

```python
def sarsa(env, num_episodes, discount_factor=1.0, alpha=0.5, epsilon=0.1):
    ...
```

`qlearning.py`

```python
def q_learning(env, num_episodes, discount_factor=1.0, alpha=0.5, epsilon=0.1):
    ...
```

并在报告中展示：

- 两种算法在 CliffWalking 的训练曲线对比（每回合回报/步数均可）。
- 结合“更安全路径 vs 更激进路径”的现象解释 SARSA（on-policy）与 Q-learning（off-policy）差异。

推荐选做

- every-visit MC：对比 first-visit 与 every-visit 的学习曲线或稳定性。
- double Q-learning：对比 double Q-learning 与 Q-learning 的表现差异，并解释其动机（Overestimation problem）。

4 Main Task：深度强化学习

完成 Warm-up 并理解相关曲线含义，然后完成 `submission.py` 并运行以下指令：

```
python submission.py --num_episodes=3000 --checkpoint=500
```

其中 `num_episodes` 是训练总轮数，`checkpoint` 是保存周期，还可以通过 `use_wandb` 参数开启 wandb 记录实验指标。

4.1 指标监控

模型训练时，每 500 个 episode 会中途保存模型至 `checkpoint_models` 文件夹。同时目录下会生成 `loss_tracker.png`，wandb 服务器端也会记录 Actor 和 Critic 的 Loss 值以及 Policy 的熵。

> 思考：在 *Actor-Critic* 架构中，为何希望 *Actor Loss* 最大化而 *Critic Loss* 最小化？*Policy* 对应的熵意味着什么？

4.2 模型评测准备

完成训练后，自行补全 `model_loader.py` 和 `opponent_loader.py` 中的 `get_model()` 和 `get_opponent()` 函数，使其能直接加载训练后的模型。

- `model`：黑棋棋手；
- `opponent`：白棋棋手。

4.3 运行评测

运行 `evaluator.py` 进行评测。可通过设置 `random_response` 参数选择对手为随机噪声（`True`）或指定的 `opponent` 模型（`False`）。

4.4 报告撰写

附上评测结果、训练记录（`loss_tracker.png`）并进行实验分析。若遇到题目要求以外的 Bug，请记录并给出 Debug 方案。

4.5 探索与改进

尝试不同的模型结构（如 CNN、Attention 机制等）或超参数组合，并记录他们的性能表现，但是最后请选出一个模型进行提交。记录实验中最初难以理解的点，并尝试给出解答或改进方案。

4.6 最终提交

将报告整理为 PDF（PDF 中需要声明小组内分工，但是小组内得分是一样的，建议大家合理分配任务哟），与 `submission.py`、`model_loader.py` 等一并提交。最终模型将与助教的模型进行对弈测评，必要时通过开展小组间对弈按排名给分。

5 报告要求

请按以下结构完成实验报告：

0. **Torch 配置**：展示环境搭建情况。
1. **Warm-up（经典强化学习）**：
   - MC：说明 first-visit MC 的核心思想、更新对象与策略改进方式，并展示 Blackjack 的关键结果图与分析。
   - TD：说明 SARSA 与 Q-learning 的更新差异，并展示 CliffWalking 的对比曲线与现象解释。
   - （选做）every-visit MC / double Q-learning 的对比与讨论。
2. **实验原理（主任务）**：用自己的话阐述二元零和马尔可夫博弈、Naive Self Play、Actor-Critic 等原理。
3. **模型设计**：附上 `__init__` 和 `forward` 方法的关键代码，并解释设计动机。
4. **约束策略分析**：分析 `forward` 部分如何解决置 0 限制和归一限制（Constrained Policy）。
5. **Bug 修复**：说明 `optimize` 函数中 3 个 Bug 的解决方案。
6. **其他问题记录**：记录在设计模型或处理张量形状（Shape Matching）、梯度计算（In-place 操作）时遇到的问题及对策。
7. **疑难点思考**：分析实验过程中最初不理解的地方及后续的理解。
8. **曲线分析（主任务）**：结合 `loss_tracker.png` 分析 Loss 和 Entropy 的变化趋势。
9. **（选做）模型优化**：记录超参数调整、模型深度调整或架构创新（CNN/Attention）的过程。
10. **（选做）思考题**：若模型权重收敛至 Actor/Critic Loss 最优点，黑白双方策略是否一定达成纳什均衡（Nash Equilibrium）？为什么？
11. **（必做）课程反馈**。

6 评分组成与提交

6.1 评分标准 (总分 200 分)

1. **报告规范性**：排版美观度、公式准确性。
2. **Warm-up 正确性与分析**：MC/TD 实现与曲线/现象解释（选做可加分）。

3. **主任务代码正确性**：逻辑合理，能正常、完整运行。
4. **主任务报告完整性与思考深度**：对问题的完成程度、原创性与分析质量。
5. **模型表现**：与助教模型的对弈结果或小组间对弈结果（需确保 `model_loader.py` 实现正确）。

各部分的分数比例会按照最终的实际情况进行调整（但会保证 **sum()=200**）。请尽可能保证质量地完成代码填空和报告的各个部分，并体现自己对整个问题建模、算法框架及理论的理解水平和思考深度。

6.2 提交规范

文件名格式：`PB24123456_ 小明 _project.zip`，目录结构如下：

```
PB24123456_小明_project/
    |-- report.pdf
    |-- gobang/
        |-- model.pth
        |-- model_loader.py
        |-- opponent_loader.py
        |-- submission.py
    |-- warmup/
        |-- mc.py
        |-- sarsa.py
        |-- qlearning.py
```

7 实验建议

以下是一些实验过程中的建议，供大家参考：

1. **环境配置**：有条件的同学可以使用本机 GPU 来加速训练。网上有很多相关教程，例如：https://blog.csdn.net/weixin_46334272/article/details/135307663
2. **模型设计**：设计一个小模型就可以达成本次实验的要求，用不着训练一个几百 M 甚至几个 G 的模型，任务简单的话，太大的模型反而容易过拟合。建议大家合理控制模型的深度、宽度和参数量。而且建议提交模型前检查一下自己的代码是否在测试阶段会过于耗时，如果时间（以及空间）开销过于离谱也有概率会被视为无法正常运行的。
3. **计算资源**：如果 PC 配置实在不大行，可以考虑使用 Kaggle 或者 Google Colab 提供的免费计算资源来训练模型。在模型设计不是过于离谱的情况下，训练最多几个小时应该就够用了（推荐 1 小时左右），注意没必要向实验室借显卡搞成军备竞赛那样，这次实验完全用不到。

4. **参考材料**：

   - Warm-up 参考：课程讲义。
   - CNN 参考材料：https://cloud.tencent.com/developer/article/2391027 https://zhuanlan.zhihu.com/p/66215918
   - Actor-Critic 入门：https://www.zhihu.com/question/56692640
   - PyTorch 官方文档：https://pytorch-cn.readthedocs.io/zh/latest/package_references/torch/

备注

在线 Issue（https://github.com/ChangshuoShen/USTC-AI3002-25fall/issues/7）**原则上不予解答任何原理性问题**。实验旨在考察学生的理解水平与思考深度，建议仔细推敲 `utils.py` 中的代码细节。