谷歌足球模拟器使用教程

#DL

开始安装

这是一份给 AI 导论的大作业准备的教程。
首先，请按照官网上的要求，安装一系列必须的库：

```
pip install gfootball
pip install tensorflow
pip install dm-sonnet==1.* psutil
pip install git+https://github.com/openai/baselines.git@master
```

已有文件介绍

完成后，解压我们群里的文件夹，找到 football-master/gfootball/example 目录，里面提供了数个可能有用处的文件：

- replaybuffer：经验回放池
- Normalization：PPO 中完成奖励归一化、状态归一化等所需的文件
- PPO_discrete：内含 PPO 的网络结构、PPO 智能体类的定义等
- Single_player_PPO：用于训练单智能体 PPO 算法
- GA_Agent：定义了遗传算法训练所需的网络结构、遗传算法智能体类。
- Utils：内有两个函数：处理状态、计算奖励
- Genetic_only_policy：训练遗传算法

状态和动作

- 在 Utils 中的 `calculating_state()` 中，我们对可观测状态进行了初步的整理，该函数可以读入环境输出的 $115 \times 4 = 460$ 维状态（环境每次更新时，会更新 4 步！可观测的状态是 115 维的），输出一个状态字典和一个状态列表。初始的 115 维状态包括：
  - 22 维：左侧球员位置
  - 22 维：左侧球员方向
  - 22 维：右侧球员位置
  - 22 维：右侧球员方向
  - 3 维：球位置
  - 3 维：球方向
  - 3 维：控球方
  - 11 维：活动的球员
  - 7 维：游戏模式的 Onehot Encoding
- 处理后的状态包括 33 维：
  - 2 维+2 维+2 维+2 维：左侧球员位置、速度、方向、方向变化
  - 2 维+2 维+2 维+2 维：右侧球员速度、位置、方向、方向变化
  - 3 维+3 维+3 维+3 维：球位置、速度、方向、方向变化
  - 3 维：控球方：无人、左侧、右侧
  - 2 维：活动的球员

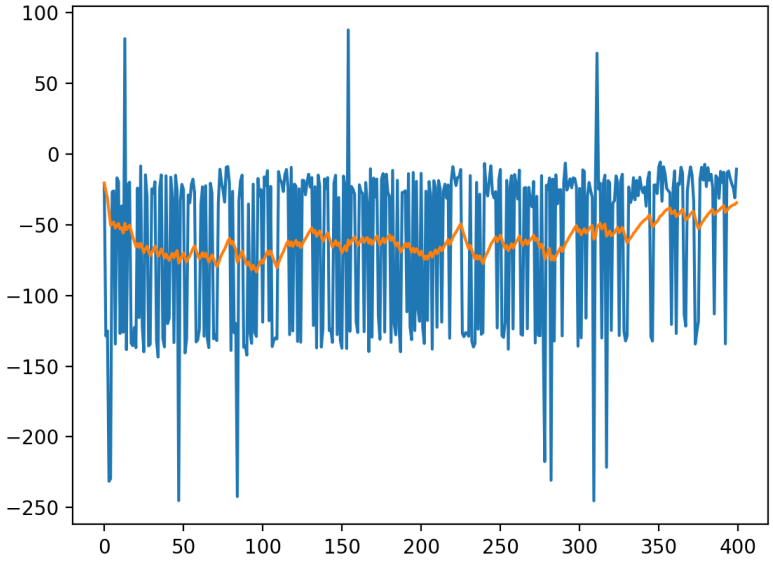
- 在 Utils 中的 `Calculating_reward()` 中，函数接受上面提到的状态字典和环境输出的奖励，计算一个新的奖励。奖励机制主要考虑了：
  - 进球加分、输球扣分（稀疏）
  - 离球越远，扣分越快（稠密）
  - 自己控球得分、对方控球扣分（稠密）
  - 速度方向向着自身指向球的矢量加分，背向这个矢量则扣分（稠密）
  - 球在己方半场扣分，在对方半场加分（稠密）

我们选择的动作是 19 个离散值：

- Idle actions
  - `action_idle` = 0, a no-op action, sticky actions are not affected (player maintains his directional movement etc.).
- Movement actions
  - `action_left` = 1, run to the left, sticky action.
  - `action_top_left` = 2, run to the top-left, sticky action.
  - `action_top` = 3, run to the top, sticky action.
  - `action_top_right` = 4, run to the top-right, sticky action.
  - `action_right` = 5, run to the right, sticky action.
  - `action_bottom_right` = 6, run to the bottom-right, sticky action.
  - `action_bottom` = 7, run to the bottom, sticky action.
  - `action_bottom_left` = 8, run to the bottom-left, sticky action.
- Passing / Shooting
  - `action_long_pass` = 9, perform a long pass to the player on your team. Player to pass the ball to is auto-determined based on the movement direction.
  - `action_high_pass` = 10, perform a high pass, similar to `action_long_pass`.
  - `action_short_pass` = 11, perform a short pass, similar to `action_long_pass`.
  - `action_shot` = 12, perform a shot, always in the direction of the opponent's goal.
- Other actions
  - `action_sprint` = 13, start sprinting, sticky action. Player moves faster, but has worse ball handling.
  - `action_release_direction` = 14, reset current movement direction.
  - `action_release_sprint` = 15, stop sprinting.
  - `action_sliding` = 16, perform a slide (effective when not having a ball).
  - `action_dribble` = 17, start dribbling (effective when having a ball), sticky action. Player moves slower, but it is harder to take over the ball from him.
  - `action_release_dribble` = 18, stop dribbling.

实用函数和预训练模型

- Single_player_PPO 和 genetic_only_policy 中，`create_single_football_env()` 用于创建单智能体环境。我们控制的是左半场的球员
- GA_agent 中，`def load_model_with_list()` 可以将一个列表中的数据作为网络参数放入到一个网络中
- `actor_parameters.pth` 是 PPO 算法训练 400 个 Epoch 的预训练模型。很菜。
- `model_weights.pth` 是遗传算法训练 50 代得到的预训练模型。也很菜。
  PPO 400 轮效果：


100
50
0
−50
−100
−150
−200
−250
0
50
100
150
200
250
300
350
400