基于强化学习和质量多样性算法的多样策略生成

杨可芸 12012438 曾宪清 12012338 陈驿来 12013025
指导老师：刘佳琳

October 2022

目录

1 问题背景

伴随计算机人工智能的发展，如今深度强化学习 (Deep Reinforcement Learning,DRL) 在游戏 [5]、蛋白质结构预测 [2]、无人驾驶等领域不断取得突破。同时，深度强化学习相关的研究也产生了新的细分，例如多目标强化学习、约束强化学习、质量多样性强化学习。其中，质量多样性强化学习，关注质量多样性优化（Quality Diversity Optimisation,QDO），训练具有多样行为（按照行为特征空间分布）的高质量（根据环境交互设置的奖励）策略集合，在游戏人工智能、机器人控制等领域有极大的应用潜力。本项目尝试融合质量多样性优化算法和传统深度学习算法，提出通用、高效、可拓展的质量多样性深度强化学习算法框架，最终实现具有行为多样性的策略集合生成。（参考 [7]）。

2 问题描述

2.1 强化学习问题描述

传统的强化学习（Reinforcement Learning,RL）的主要过程是构建最大化马尔可夫决策（Markov Decision Process, MDP）获得更高奖励的策略模型。进行学习，并做出决策的对象称为智能体。以如下图 1 为例，智能体（Agent）与环境（Environment）（所有与之相互作用的事物）中持续交互，智能体选择行为，环境做出对应的响应，并使得智能体呈现新的状态。环境在给定状态和动作后会转移到下一个状态并产生收益（reward）。这也是智能体在不断尝试动作的过程中想要最大化的目标。如图 1.

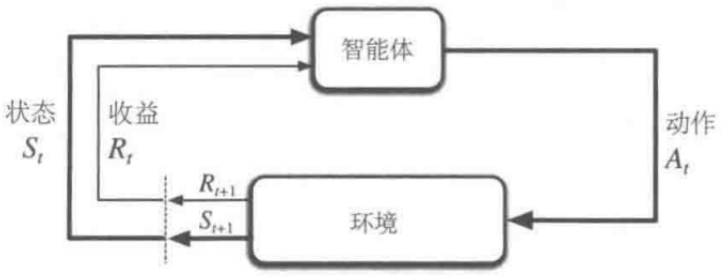


图 1: 图来自 [6]Fig.3.1。马尔可夫决策过程中的“智能体”与“环境”的交互

传统 RL 算法（DRL 算法）寻找最大的期望（折扣）奖励目标。为了能够不失一般性地考虑有限长度时间决策，我们让 R_t 表示智能体在环境中第 t 个时刻得到的奖励，让 h 表示决策时长，两种目标（有无折扣）分别可以定义如下：

$$J = \frac{1}{h}\mathbb{E}\left[\sum_{t=0}^{h-1} R_t\right], \tag{1}$$

$$J = \mathbb{E}\left[\sum_{t=0}^{h-1} \gamma^t R_t\right], \gamma \in [0,1], \tag{2}$$

其中 γ 为折扣率，它决定了未来收益的现值。
。本小节内容参考 [6][7]。

2.2 质量多样性优化问题描述

质量多样性优化最早在 [3] 中被提出，文章中的 MAP-Elites 算法旨在找到特征空间中每一个细胞所对应的质量函数值最大的解。令 $\mathcal{X}$ 为解空间，考虑连续的质量函数 $f(\cdot): \mathcal{X} \mapsto \mathbb{R}$，以及一系列特

征函数 $b_i(\cdot): \mathcal{X} \mapsto \mathbb{B}_i$，其中 $\mathbb{B}_i$ 为第 i 个特征的特征空间，QDO 的目标是找到满足以下条件的解集 X^*：

$$X^* = \left\{ \arg\max_x [f(x),\ \text{s.t. } b_i(x) = b_i] \,\middle|\, \forall b_i \in \mathbb{B}_i \right\} \tag{3}$$

本小节参考 [7]。

2.3 质量多样性强化学习

质量多样性强化学习是强化学习的重要子课题。在强化学习的设定下，智能体在与环境交互的过程中会得到一些奖励，而强化学习的目标则是最大化累积奖励，即智能体在环境中要得到最高的奖励分数。时至今日，在强化学习领域已经取得了一系列突破性的进展。例如，智能体可以在很多复杂的游戏中取得超越人类的表现，例如自适应控制，通过强化学习算法，使机械臂能够根据不同的材料和场景自适应地搭建积木桥。 所以质量多样性强化学习，允许程序自定义一系列行为特征函数（比如在场景中，AI 如何决定任意特定动作频率，）。我们此时将解集 $\mathcal{X}$ 替换为环境中所有可能的策略，质量指标 $f(\cdot)$ 替换为公式 (1) 或 (2)，即为质量多样性强化学习问题。

3 项目目标

质量多样性强化学习算法文章 [4] 提出了 PGA-Map-Elites 算法，它将基于 actor-critic 架构的面向连续动作空间的深度强化学习算法 TD3 与质量多样性算法 Map-Elites 相结合，成功地阐明了搜索空间中性能与关心的维度之间的关系，并在整个可能的行为空间内找到了高性能的解决方案。

本学期的最终目标为复现文章 [4]，并结合已有研究，对当前算法框架提出改进，或解决现有算法中的一些不足 [7]。对于测试所用的强化学习环境，计划从 BipedalWalker 控制模拟环境、Mujoco 机器人控制模拟环境、基于 Mario-AI-Framework 修改的强化学习环境中选择 [7]。

4 项目规划

4.1 项目内容

本项目最终所需复现的文章 [4] 提出了一个可更换基础 DRL 算法的框架。为了理解其中的 PGA-Map-Elites 算法运用到的深度强化学习算法 TD3，我们制定了具体的算法学习路线：

Q-learning → Policy Gradient →Deep-Q-Network → Deep Deterministic Policy Gradient (DDPG) → Twin Delayed Deep Deterministic policy gradient (TD3)

在强化算法中，Q-learning 和 Deep-Q-Network 是基于价值的（value-based），Policy-Gradient 是基于策略的（policy-based），DDPG 则是两者的融合（actor-critic），TD3 在 DDPG 基础上实现了三大方面的优化。我们计划学习 Q-learning 和 Policy Gradient 以入门强化学习，通过学习涉及神经网络的 DQN 算法来了解深度学习的基础概念，再进一步研究并复现结合了 value-based 和 policy-based 两类强化算法的 DDPG 来加深强化学习的了解。在此基础上继续复现 DDPG 的优化版本——DRL 算法 TD3[1]，最终在复现的 TD3 算法基础上实现 [4] 中的 PGA-Map-Elites 算法，针对不足之处提出改进，并最终运用于强化学习环境进行测试。

4.2 初期规划

在项目初期阶段，我们计划首先阅读文献 [3][4]，了解质量多样性算法的产生背景及伪代码过程和相关应用。其次，通过学习分别基于价值的和基于策略的两种算法以入门强化学习，理解其核心思想。除此之外，搜索相关可用于强化学习算法测试的环境库并尝试配置文章 [4] 中的测试环境 QDgym。

4.3 中期规划

在项目中期阶段，我们计划继续沿着既定算法学习路线复现 DQN, DDPG 算法，并调研、配置项目内容中提及的强化学习环境以对算法进行调参测试。为实现此阶段目标，我们还将配合著名强化学习教材 [6]，对强化学习进行更深入的研究。

4.4 后期规划

在项目的后期阶段，我们预期将完成 PGA-MAP-Elites 算法的复现，熟悉掌握其算法框架。在此基础上总结其不足及改进方向，尝试替换其他 DRL 算法进行对比测试，进行优化，以期最终实现基于强化学习和质量多样性算法的多样策略生成目标。

5 背景研究

5.1 MAP-Elites 算法 [3]

5.1.1 背景和概述

MAP-Elites 是一个高效的搜索算法。在该算法被提出之前，一些传统搜索算法具有如下缺点：只关注在搜索空间中找到一个或一组高质量的解决方案；无法解决一些需要“黑盒”优化的问题，因为对于这类问题我们难以获取决定性能的评估函数，因而不能使用需要计算函数的优化方法，例如梯度上升/下降；此外它们可能会因为依赖于随机启发式算法（即对好的解决方案进行随机更改，来尝试获得更好的解决方案）而导致局部最优，但却无法获得全局最佳的解决方案。相比之下，MAP-Elites 算法能够返回一组高性能且多样化的解决方案；阐明性能与解决方案中所关注的维度之间的关系；改进优化性能，因为它探索了更多的特征空间以避免陷入局部最优，从而找到不同的且更好的性能峰值。

如图2，MAP-Elites 算法在高维空间中搜索，以在低维特征空间中的每个点处找到性能最高的解决方案。我们将这种类型的算法称为“照明算法（Illumination algorithm）”，因为它照亮了特征空间中每个区域的性能潜力，包括性能变化和感兴趣特征之间的权衡。

5.1.2 优化算法（Optimization algorithms ）与照明算法（Illumination algorithms ）

优化算法试图在搜索空间中找到性能最高的解决方案，而优化算法并不是为了返回特征空间区域中可能的性能最高的解而设计的。照明算法被设计为在特征空间中的每个点返回性能最高的解决方案。任何照明算法（如 MAP Elites 算法）也可以用作优化算法，使得照明算法成为优化算法的超集。

5.1.3 MAP-Elites 算法的细节

Map-Elites 算法的伪代码如图3所示。首先，用户选择一个计算解决方案 x 的性能度量 $f(x)$。例如，如果寻找机器人的形态，性能指标可以是机器人的速度。其次，用户选择 N 个感兴趣的变化维度，组

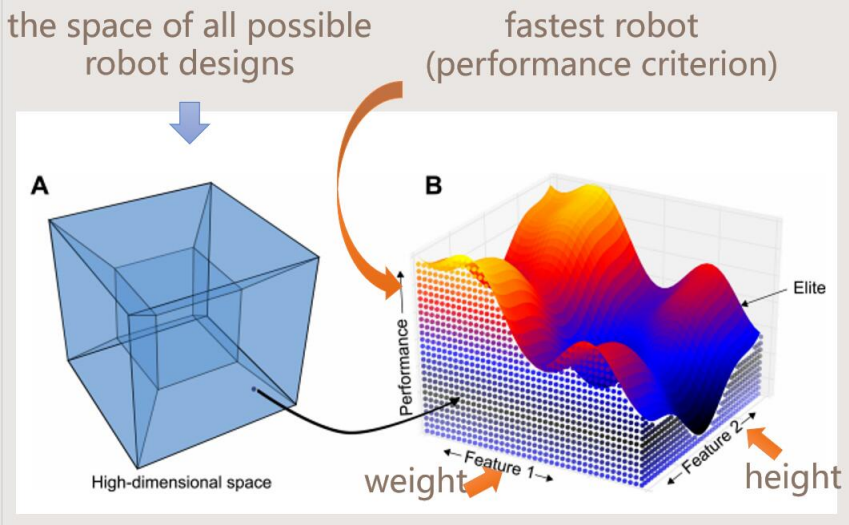


图 2: 修改自 [3]Fig.1。
照亮了特征空间每个区域的适应度潜力，包括性能和感兴趣特征之间的权衡

procedure MAP-ELITES ALGORITHM (SIMPLE, DEFAULT VERSION)
$(\mathcal{P} \leftarrow \emptyset, \mathcal{X} \leftarrow \emptyset)$ ▷ *Create an empty, N-dimensional map of elites: {solutions $\mathcal{X}$ and their performances $\mathcal{P}$}*
for iter $= 1 \rightarrow I$ **do** ▷ *Repeat for I iterations.*
  if iter $< G$ **then** ▷ *Initialize by generating G random solutions*
    $\mathbf{x}' \leftarrow$ random_solution()
  else ▷ *All subsequent solutions are generated from elites in the map*
    $\mathbf{x} \leftarrow$ random_selection($\mathcal{X}$) ▷ *Randomly select an elite x from the map $\mathcal{X}$*
    $\mathbf{x}' \leftarrow$ random_variation($\mathbf{x}$) ▷ *Create x', a randomly modified copy of x (via mutation and/or crossover)*
  $\mathbf{b}' \leftarrow$feature_descriptor($\mathbf{x}'$) ▷ *Simulate the candidate solution x' and record its feature descriptor $\mathbf{b}'$*
  $p' \leftarrow$performance($\mathbf{x}'$) ▷ *Record the performance p' of x'*
  if $\mathcal{P}(\mathbf{b}') = \emptyset$ or $\mathcal{P}(\mathbf{b}') < p'$ **then** ▷ *If the appropriate cell is empty or its occupants's performance is $\leq p'$, then*
    $\mathcal{P}(\mathbf{b}') \leftarrow p'$ ▷ *store the performance of x' in the map of elites according to its feature descriptor $\mathbf{b}'$*
    $\mathcal{X}(\mathbf{b}') \leftarrow \mathbf{x}'$ ▷ *store the solution x' in the map of elites according to its feature descriptor $\mathbf{b}'$*
return feature-performance map ($\mathcal{P}$ and $\mathcal{X}$)

图 3: [4] 中的 Fig.2, 展示 Map-Elites 的 Pseudo code

成一个名为 b 的向量，定义一个感兴趣的特征空间。比如说，我们对于机器人的形态，感兴趣的一个维度可能是机器人有多高，一个维度可能是它的重量，另一个可能是每米移动它的能量消耗，等等。给定一个特定的离散化程度划分特征空间，Map-Elites 将寻找 N 维特征空间中每个细胞的最高性能解（我们将特征空间中的每个单元格称为一个细胞）。例如，Map-Elites 将寻找以高、重、高效为特征的机器人中速度最快的机器人；以高，重，低效为特征的速度最快的机器人；以高，轻，高效为特征的速度最快的机器人等等。搜索在搜索空间中进行，搜索空间是 x 的所有可能值的空间，其中 x 是对候选解的描述。在我们的示例中，搜索空间包含所有可能的机器人形态的描述。我们称 x 描述符为基因组；函数 $f(x)$ 称为适应度函数，返回每个 x 的性能；行为函数 $b(x)$ 则是对于每个 x，确定 x 在每个 N 个特征维度中的值。换句话说，$b(x)$ 返回 b_x，它是一个描述 x 的特征的 n 维向量。在我们的例子中，$b(x)$ 的第一个维度是机器人的高度，第二个维度是它的重量，第三个维度是它的每米移动的能量消耗，等等。特征向量的一些元素可以直接在表型中进行测量（如身高、体重），但其他的元素（如能量消耗）需要在模拟环境中测量。

伪代码的过程：首先随机生成 G 个基因组，并确定每个基因组的性能和特征。这些基因组被放入它们的特征对应的细胞中（如果多个基因组映射到同一个细胞，则保留每个细胞中表现最高的一个基因组）。接下来的每次迭代中，特征空间中的一个细胞被随机选择，该细胞中的基因组通过突变和交叉产生一个后代。如果该基因组变异后对应的细胞是空的，或者后代的性能比当前的细胞对应的基因组更高，那么后代基因组将被放置在细胞中。

5.1.4 两个值得注意的地方

由于如下两个原因，Map-Elites 算法无法保证特征空间中的所有细胞都将被填充：在特征空间中可能没有映射到特定细胞的基因组 x。例如受限于物理定律，不可能有任意高度和重量的机器人；即使存在生成映射到特征中细胞的基因组，搜索算法可能无法成功生成。另外，如果在给定的问题中可以直接在特征空间中进行操作，那么就没有必要采用 MAP-Elites，因为可以简单地在特征空间执行穷举搜索。因而 MAP-Elites 算法可视为尝试在特征空间中执行这种穷举搜索的一种方式，但仍存在一个额外的挑战，即如何为该特征空间中的每个单元找到性能最高的解决方案。

5.1.5 Map-Elites 算法和它之前的算法的区别

1. Search + Local Competition (NS+LC 算法)
   鼓励特征空间中搜索的多样性，但让每个有机体仅与特征空间中靠近它的其他有机体在性能上竞争；时间复杂度更高，因为要计算每一代与其它有机体的特征距离；同时需要一个储存器来记录那些特征细胞已经被访问过。

2. MOLE 算法
   MOLE 有两个目标：第一个是性能，第二个是每个有机体尽可能远离其他有机体，距离在用户指定的特征空间中测量。仅保留该区域中性能相对最佳的单元 (Map-Elites 保留所有单元的性能) 它不会同时均匀地搜索地图所有区域的改进。

5.1.6 Map-Elites 的两个应用

神经网络

由于神经网络中，最小化神经元的连接成本可以促进模块化的发展，使用 MAP-Elites 研究神经网络的连接成本同模块化程度，创建一个二维特征空间，两个特征分别是连接成本和模块化程度，用 MAP-Elites 来寻找最优的连接情况。结果表明，在全局性能 (找到一个最高性能的解决方案)、可靠性 (所有可填充细胞的平均性能)、精度 (仅由算法填充的细胞的平均性能)、覆盖率 (填充的细胞数量) 四个指标上，Map-Elites 都显著高于传统搜索，NS+LC，随机等其他搜索算法。

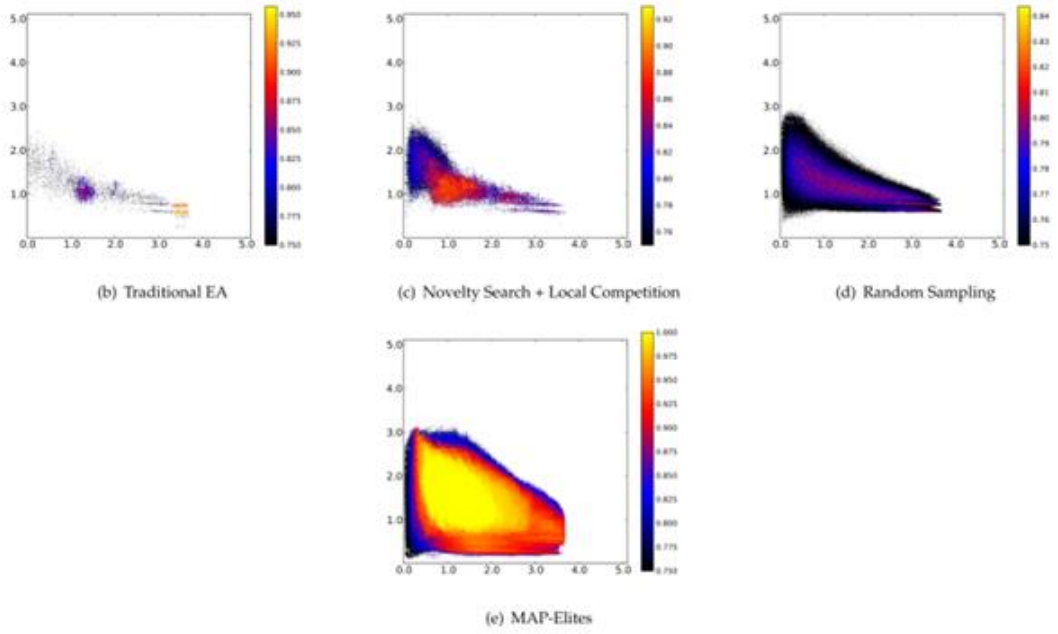


图 4: [4] 中的 Fig.3, 横坐标是连接成本，纵坐标是模块化程度颜色表示 performance

仿真的软运动机器人形态

用其它搜索算法得到的最终进化的软机器人的形态大多相似。如同进化算法的经典情形，在每次运行的搜索过程中，易发生找到局部最值后难以跳脱该区域以寻求另一峰值的情况。最终进化结果很少包含某一特定材料（骨骼），因而无法全面地展示这种材料对机器人的形态及表现的影响。若采用奖励函数激励骨骼含量的增加，可能会导致过度投资，且很难提前知道产生成功的设计所需的骨骼比例。而采用 Map-Elites 进行搜索，能够全面地探索搜索空间以寻求全局最优点，观察在不同骨骼含量下表现最好的软机器人，从而了解骨骼的使用如何影响机器人的速度和形态设计，除此以外，还可增加感兴趣的维度以探究其变化的影响（如不同尺寸）。

5.1.7 Map-Elites 算法的总结

与传统优化算法相比，Map-Elites 和其他照明算法映射整个特征空间，告诉用户特征和性能之间的各种权衡; 另外，Map-Elites 学习了自然界产生多样性的特点，因为它同时奖励了众多不同领域中表现最好的解决方案，最后，它在用户定义的特征空间的每个点上找到性能最好的解，是帮助我们了解复杂搜索空间的有价值的新工具。

5.2 Policy Gradient Assisted MAP-Elites 算法 [4]

5.2.1 优势体现

1. Map-Elites 使用简单开环系统，即系统的输出取决于输入，而输入却独立于系统输出。这种系统因为不包含反馈环节而被称为无反馈系统。而 PGA-MAP-Elites 通过引入基于梯度的变异算子，结合反向传播，以利用从批评性神经网络获得的梯度估计值来调整参数，从而实现快速找到性能更高的解。

2. 之前介绍的 Map-Elites 依赖于遗传算法，因而 MAP-Elites 被限制于只能适用于低维度问题（优化参数的数量通常保持在 100 个以下），相比之下，PGA-MAP-Elites 算法通过引入基于梯度的变异算子，结合反向传播，利用从批评性神经网络获得的梯度估计值，快速找到性能更高的解。同时，对于一些需求复杂且需要精确控制变量的问题，Deep-Neural-Network （DNN）需要加入巨量的参数来解决，而 PGA-MAP-Elites 算法可以相对轻松地完成相同工作，并且能够在随机的环境中，保持较好的学习鲁棒性。

6 强化学习算法的学习

强化学习的最终目的是学习“做什么来使数值化之后的收益信号最大化”，学习体通过不断试错，尝试，去发现怎样可以产生更大的收益。“‘试错’和‘延迟收益’—-是强化学习两个最重要最显著的特征”(此句来自《强化学习》[6]) 强化学习最重要的四个要素为：策略、收益、价值函数、对环境建立的模型。

6.1 基于“价值”与基于“策略”

价值函数是所有强化学习算法不可缺少的要素，价值函数是指确定状态（或确定状态以及确定动作的二元组）的函数，目的是评估确定状态（和动作）下有多“好”，是否“好”则取决于收益的期望值。选取期望值最高的动作来作为下一步行为，则认为是基于“价值”(value-based) 的强化学习算法。但同时，学习体对未来的收益的期望取决于它具体选择了什么动作，所以有时价值函数与特定的行为方式（选择动作的概率）相关，此时我们称之为基于“策略”(policy-based)，“策略”可以认为是从状态到每个动作的选择概率之间的映射。[6]

PGA-Map-Elite 算法将强化学习算法 TD3 与 Map-Elite 相结合。实现了搜索空间的强大照明，在整个可能的行为范围内找到高性能的解决方案。为了理解在 PGA-Map-Elite 中运用到的 TD3 强化学习算法，我们按照如下顺序进行学习 Q-learning → Policy Gradient → Deep-Q-Network → DDPG →TD3, 其中，Q-learning 和 Deep-Q-Network 是基于价值的（value-based），Policy Gradient 是基于策略的 (policy-based)，DDPG 则是两者的融合（actor-critic），TD3 是 DDPG 的改进版本。

6.2 Q-learning 算法

Q-learning 是强化学习算法中 value-based 的算法，伪代码见 Algorithm 1。Q 即为 $Q(s,a)$ 就是在某一时刻的 s 状态下 ($s \in S$)，采取 $a(a \in A)$ 动作的未来奖励期望（公式4，5），环境会根据 agent 的动作反馈相应的回报 reward r。算法的主要思想就是将 State 与 Action 构建成一张 Q-table 来存储 Q 值（初始化为 0），然后根据 Q 值来选取能够获得最大的收益的动作，获取 reward 后根据公式 (6) 来更新 Q-table 的值。对于动作的选取，还可结合 $\epsilon - greedy$ 策略，即 ϵ 概率选择 Q 值最大动作，$(1-\epsilon)$ 概率选择随机动作。引入 ϵ 可更好地权衡开发与探索。

$$Q(s_t, a_t) = E(G_t|s_t, a_t) \tag{4}$$

其中, G_t 为当前 t 时刻对未来的累计折扣奖励,

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \tag{5}$$

$$Q^{new}(s,a) = Q^{old}(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q^{old}(s,a)] \tag{6}$$

来自https://blog.csdn.net/qq_30615903/article/details/80739243?ops_request_misc

Algorithm 1 Q-learning 算法伪代码

Initialize $Q(s,a)$ arbitrarily

repeat(for each episode):

  Initialize S

  repeat(for each step of episode):

    Choose A from S using policy derived from$Q$(e.g.ϵ-greedy)

    Take action A, observe R, S'

    $Q^{new}(S,A) \leftarrow Q^{old}(S,A) + \alpha[R + \gamma \max_{a'} Q(S',a') - Q^{old}(S,A)]$

    $S \leftarrow S'$

  until S is terminal

until M episodes done

6.2.1 出租车调度环境测试

Gym 库（https://gym.openai.com）是 OpenAI 推出的强化学习实验环境库。本次测试选用了 Gym 中 Taxi-V3 出租车调度问题作为测试环境。其状态空间大小为 500，环境中观察到的状态是一个 [0,500) 的整数值，其中出租车的位置的行数是 0-4，列数是 0-4。乘客的位置是 0-4，其中 4 表示乘客在车上, 目的地的取值是 0-3。状态示例 $(1, 0, 1, 3)$ 见图5。因此总的状态数为 $5 * 5 * 5 * 4 = 500$。动作空间大小为 6，动作取值是 [0-5]，其中 0-3 表示上下左右，4 表示乘客上车，5 表示乘客下车。状态转移限制为只能在地图范围内上下左右移动一格，并且在有竖线阻拦地方不能横向移动。关于奖励函数 R，设置如下，完成一次任务可以得到 20 奖励，每次试图移动得到-1 奖励，不合理地邀请乘客上车（例如目前车和乘客不在同一位置，或乘客已经上车）或让乘客下车（例如车不在目的地，或车上没有乘客）得到-10 奖励。当乘客在车上且已移动到乘客下车的目标位置并下车时，即可判断任务完成，测试终止。最终测试结果如图6 所示。

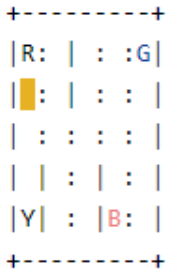
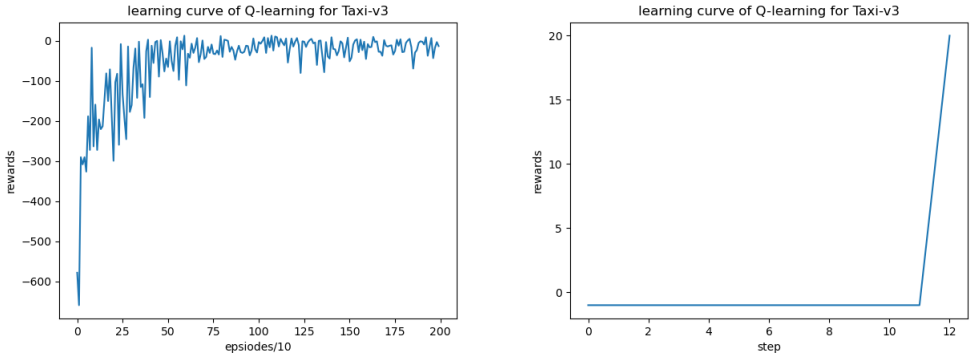


图 5: 状态示例



(a) 在训练过程中随着迭代次数增加，每回合根据 Q-learning 算法获得的奖励总值显著上升

(b) 利用训练所得 Q 表进行单次测试，结果显示出租车能够选择正确的行为序列接送乘客

图 6: Q-learning 算法在 Taxi-V3 环境的训练表现及测试结果

6.3 Q-learning 算法与 Sarsa 算法对比

Q-learning 与 Sarsa 在强化学习中都属于基于价值的算法，他们具有许多相似之处如选择 action 都采用 -greedy 方法，都通过更新 Q-value 的方式进行学习。两者的差异主要体现在 Q 值更新上：Q-learning (off-policy) 选取下一步 Q 值最大的动作作为更新

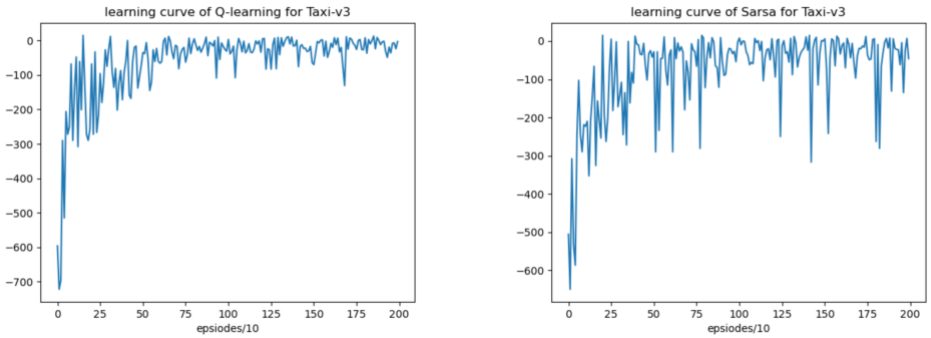
$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)] \tag{7}$$

而 Sarsa (on-policy) 使用下一步的实际动作来作为更新

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma Q(s',a') - Q(s,a)] \tag{8}$$

这两种不同的更新方式形成了 off-policy 和 on-policy 对比，即可通过产生数据的策略与评估和要改善的策略是否为同一个策略来判断该算法采用 off-policy 或 on-policy。

我们还将 Q-learning 算法与 Sarsa 算法运用于出租车调度环境中进行对比测试，并保证测试过程中不同算法的相同参数一致，结果如图7所示。可见，在此测试环境的训练过程中，off-policy 较 on-policy 表现更优。从算法过程中分析，Sarsa 优化的是实际上执行的策略，更新 Q 值所用的 action 都是实际发生的，受到探索过程的影响，可能无法学习到最优解，因而收敛到局部最优。而 Q-learning 则进行了改进，它不仅拥有行为策略，还有目标策略。行为策略用于大胆探索，目标策略根据行为策略来优化自己，两者在一定程度上相互独立，从而在保证探索性的同时仍能继续学习最优策略。



(a) Q-learning　　(b) Sarsa

图 7: 将 Q-learning 算法与 Sarsa 算法运用于 Taxi-V3 环境的训练过程对比

6.4 Policy Gradient 算法

当状态空间变得高维时，q-learning 方法则变得不适用，因为 Q-table 无法存储过多的状态，此时就需要借助神经网络来处理。Policy Gradient 算法使用神经网络输入当前状态，输出在这个状态下采取每个动作的概率。网络使用反向传播算法，设置一个误差函数，通过梯度下降来使损失最小。但对于强化学习来说，我们不知道动作的正确与否，只能通过奖励值来判断这个动作的相对好坏。我们把基于上面的想法，产生非常简单的想法：如果一个动作得到的期望折扣奖励大，那么我们就使其出现的概率增加，如果一个动作得到的 reward 少，我们就使其出现的概率减小。我们定义 $J(\theta)$ 为采用 θ 为参数的神经网络得到的期望奖励的函数，我们希望的是通过改进参数 θ 来使得 $J(\theta)$ 值最大。改进 θ 的公式如9。

$$\theta_{t+1} = \theta_t + \alpha\widehat{\nabla J(\theta_t)} \tag{9}$$

来自https://blog.csdn.net/ygp12345/article/details/109009311

7 小结

该项目本学期的目标为复现 [4] 中的 PGA-Map-Elites 算法，是一个将强化学习与传统 Map-Elites 相结合的算法。对于质量多样性优化，我们首先研读了提出 Map-ELites 算法的文章 [3]，对质量多样性算法的目的和内容有了初步认识。其次，我们阅读了项目目标需要复现的算法的文章 [4]，了解 PGA-Map-Elites 算法的大致框架，并尝试配置论文中提到的的开源环境 QDgym(https://github.com/ollenilsson19/QDgym)。对于强化学习，我们首先学习并复现了最基础的 Q-learning 和 Sarsa 算法（代码见 Github 仓库https://github.com/tsukii7/Quality-diversity-with-reinforcement-learning），并调整参数进行测试。这两种基于价值的强化算法都采用了 ϵ-greedy，但他们根据更新 Q 值时使用的策略分别属于异策略 (off-policy) 及同策略 (on-policy) 两种类型。通过比较他们的测试结果，我们发现在出租车调度问题中，异策略算法具有更佳的表现。其次我们研究了 Policy Gradient 算法原理，对其具体实现有了一定了解。该算法使用神经网络输入当前的状态，网络就可以输出在这个状态下采取每个动作的概率，网络使用反向传播算法，设置一个误差函数，通过梯度下降来使损失最小。Policy Gradient 在处理高维的状态空间和连续的动作空间时比 Q-learning 更有优势，因为 Q-table 的大小有限，无法存储过多的状态和动作。后半学期的目标为根据项目规划继续学习并动手实现 DQN,DDPG 与 TD3，在此基础上复现 PGA-Map-Elites，提出改进，并最终运用于强化学习环境进行测试。

8 项目分工

1. 曾宪清：阅读学习相关参考论文，实现 Q-learning 和 Sarsa，并在两种不同环境下进行测试，编辑修改报告和 ppt。
2. 杨可芸：阅读学习相关参考论文，学习并运行 Policy Gradient 算法，编辑修改报告和 ppt。
3. 陈驿来：阅读学习相关参考论文，学习并运行 TD3 算法并配置相关环境，编辑修改报告和 ppt。

参考文献

[1] S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. *International conference on machine learning*, page 1587–1596, 2018.

[2] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K.Tunyasuvu nakool, R. Bates, A. Žídek, and A. Potapenko et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[3] J.-B. Mouret and J. Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.

[4] O. Nilsson and A. Cully. Policy gradient assisted map-elites. *Genetic and Evolutionary Computation Conference*, page 866–875, 2021.

[5] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrit twieser, I. Antonoglou, V. Panneershelvam, and M. Lanctot et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[6] R. S. Sutton and A. G. Barto. Reinforcement learning: An introduction (2nd edition). *MIT press*, 2018.

[7] 王子祺. 基于强化学习和质量多样性算法的多样策略生成. 创新实践课题介绍, 2022.