Reinforcement Learning

Policy Gradient

袁路展

北京邮电大学

2022 年 4 月 26 日

Bottelneck of RL

为什么强化学习?

- 在高维空间下，动态规划、传统控制这些方法计算复杂度过高，不能表现的很好。

强化学习的问题

- 之前我们讲过的 value-based 和 policy-based 的方法，他们都需要和环境进行多轮的迭代，才能得到对策略的准确评估。
- 在很多真实的情形下，是不可能和环境进行海量采样的，这也是限制强化学习不能落地到真实场景的主要原因之一。

Approaches to the sample complexity of RL

怎样让智能体不用和真实环境进行海量的交互，也能得到一个策略？

- 构建一个环境，在环境里面智能体可以很廉价、安全的和智能体交互，同时也可以对策略进行评估和优化。

常见的方法有：

- 通过专家知识构建仿真环境。
  - 需要额外构建一个仿真环境，而且往往代价不菲。
  - 仿真环境与真实环境有偏差。
- 通过已有的样本学习环境模型（MBRL）。
  - 不需要额外的仿真环境，相对廉价。
  - 复杂场景，效果一般。

MBRL

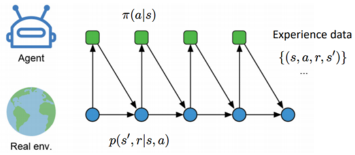


图 1: 无模型的强化学习

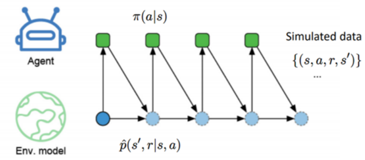


图 2: 基于模型的强化学习

- 无模型的强化学习：
  - 智能体和真实的环境交互，产生真实样本来优化自身。
- 基于模型的强化学习：
  - 智能体和学习得到的环境模型交互，产生虚拟样本来优化策略。

What is learning

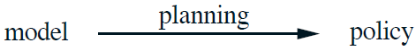
What is Learning:
A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Experience E

Experience E: 已经采集到的样本。
Learning 的核心是从已经见过的样本里面去学习。

What is planning

What is planning:
Any computational process that takes a model as input and produces or improves a policy for interacting with the modeled environment:

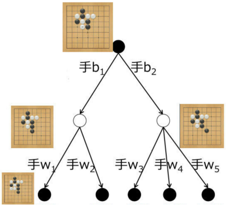


Model

Model: 环境模型。
Planning 的核心是根据环境模型，来预测我们没有见过的未来可能会发生什么来进行策略优化。

Demo of planning and learning

Planning:



Learning：根据之前的棋谱（交互信息），来决定现在这一步下什么。Planning 的核心是，在这一步，我去预测未来，然后来选择动作执行。

Review on MBRL

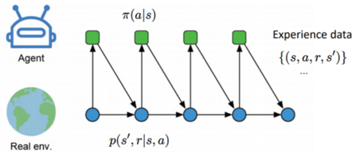


图 3: 无模型的强化学习

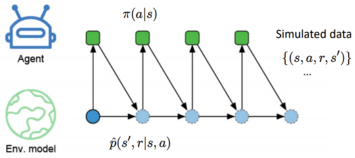
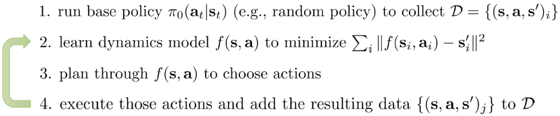


图 4: 基于模型的强化学习

- 无模型的强化学习：
  - Learning：智能体和真实的环境交互，产生真实样本来优化自身。
- 基于模型的强化学习：
  - Planning：智能体和学习得到的环境模型交互，产生虚拟样本来优化策略。

Naive MBRL

最原始的 MBRL 算法就可以表示为：



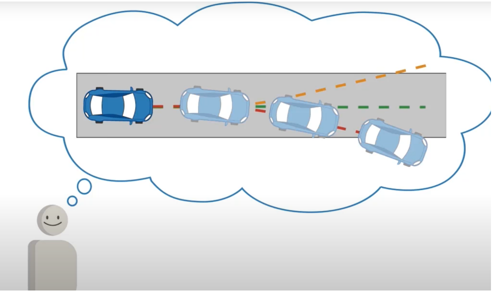
1. run base policy $\pi_0(\mathbf{a}_t|\mathbf{s}_t)$ (e.g., random policy) to collect $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}')_i\}$
2. learn dynamics model $f(\mathbf{s}, \mathbf{a})$ to minimize $\sum_i \|f(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{s}'_i\|^2$
3. plan through $f(\mathbf{s}, \mathbf{a})$ to choose actions
4. execute those actions and add the resulting data $\{(\mathbf{s}, \mathbf{a}, \mathbf{s}')_j\}$ to $\mathcal{D}$

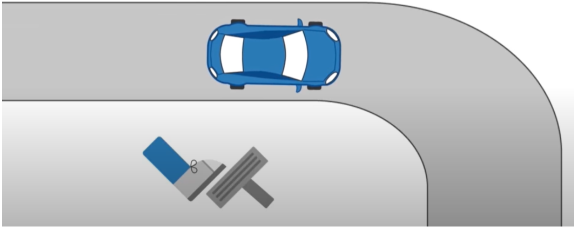
好处

- 训练收敛需要的海量数据和计算过程，都放在 planing 的过程里面了，和真实环境交互的成本大大降低。

存在的问题

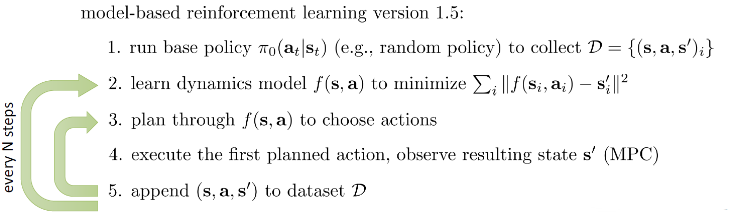
- 注意步骤 3，它产生的是一个动作序列，这会导致第四步执行的时候，不能很好的应对突发情况。





MPC

解决的思路也很简单，在每一步都重新做一次 plan，那么我就可以在突发情况之前有一个预警。

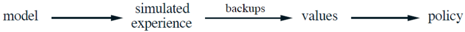


model-based reinforcement learning version 1.5:

1. run base policy $\pi_0(\mathbf{a}_t|\mathbf{s}_t)$ (e.g., random policy) to collect $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}')_i\}$
2. learn dynamics model $f(\mathbf{s}, \mathbf{a})$ to minimize $\sum_i \|f(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{s}'_i\|^2$
3. plan through $f(\mathbf{s}, \mathbf{a})$ to choose actions
4. execute the first planned action, observe resulting state $\mathbf{s}'$ (MPC)
5. append $(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ to dataset $\mathcal{D}$

Similarity of Planning and Learning

- 都是为了计算未来要发生的事情的期望.
- 在实际操作中，我们用 planning 的方式主要是通过和模型交互产生 simulated experience 来更新价值函数和策略。



- 所以在很多 learning 的方法里面，会把 real experience 换成 simulated experience。

Bottelneck of MBRL

Mismatch on model and real environment!

完全利用 planning 产生的虚拟样本来更新策略可能会为策略带来偏差。因此用真实的样本和模拟的样本都来进行策略更新。

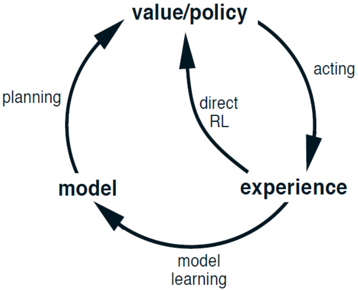
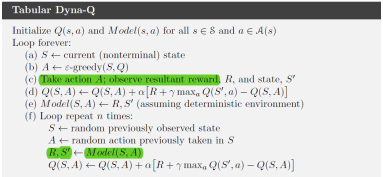


图 5: Dyna



```
Tabular Dyna-Q
Initialize $Q(s,a)$ and $Model(s,a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$
Loop forever:
   (a) $S \leftarrow$ current (nonterminal) state
   (b) $A \leftarrow \varepsilon$-greedy$(S,Q)$
   (c) Take action $A$; observe resultant reward, $R$, and state, $S'$
   (d) $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$
   (e) $Model(S,A) \leftarrow R, S'$ (assuming deterministic environment)
   (f) Loop repeat $n$ times:
          $S \leftarrow$ random previously observed state
          $A \leftarrow$ random action previously taken in $S$
          $R, S' \leftarrow Model(S,A)$
          $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$
```

图 6: Dyna-Q

现在绝大多数的 MBRL 的方法都是用 Dyna 的形式

Dyna:Integrated Planning, Acting, and Learning

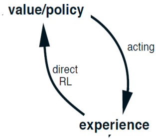


图 7: Model-free Learning

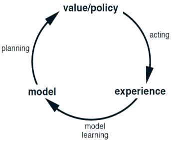
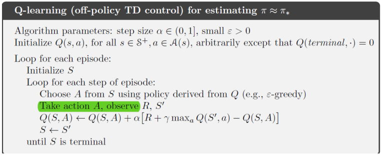


图 8: Model-based Planning

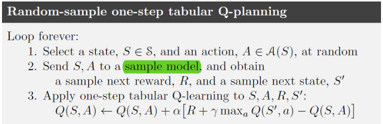


Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

```
Algorithm parameters: step size $\alpha \in (0,1]$, small $\varepsilon > 0$
Initialize $Q(s,a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal,\cdot) = 0$

Loop for each episode:
    Initialize $S$
    Loop for each step of episode:
        Choose $A$ from $S$ using policy derived from $Q$ (e.g., $\varepsilon$-greedy)
        Take action $A$, observe $R$, $S'$
        $Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$
        $S \leftarrow S'$
    until $S$ is terminal
```

图 9: Q Learning



Random-sample one-step tabular Q-planning

```
Loop forever:
    1. Select a state, $S \in \mathcal{S}$, and an action, $A \in \mathcal{A}(S)$, at random
    2. Send $S, A$ to a sample model, and obtain
           a sample next reward, $R$, and a sample next state, $S'$
    3. Apply one-step tabular Q-learning to $S, A, R, S'$:
           $Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$
```

图 10: Q-Planning

MBRL 是解决 RL 样本效率问题的重要途径之一，通过构建环境模型，让 agent 可以和模拟的环境模型交互产生样本来优化自己，大大降低了智能体和环境交互的需求。但是 MBRL 也存在一些问题。

- 环境模型的学习本身是一个小样本有监督问题，很难学到一个完美的环境模型，这会导致很多 MBRL 的算法得到的结果不如 MFRL 的。
- 在复杂环境中，模型学不出来，现在 MBRL 的算法主要应用场景是传统机械控制的简单环境和隐空间。

Review

- MBRL
    - 为什么 MBRL?
    - MBRL 存在的问题?
- Planning and Learning
    - Planning 是与 simulative model 交互产生样本。
    - Learning 是根据历史信息，学到最好的动作来执行。
- MPC: 每一步都进行 plan，降低 model 偏差带来的问题。
- Dyna: 把 simulative experience 当成 real experience.

Questions

Q&A