

Beijing Forest Studio  
北京理工大学信息系统及安全对抗实验中心



# DQN深度强化学习算法

DQN深度强化学习算法

硕士研究生 秦泉喃

2020年05月31日

- 背景简介
- 基本概念
- 算法原理
  - Deep Q-Network算法
  - Deep Q-Network的调参细节
- 优劣分析
- 应用总结
- 参考文献

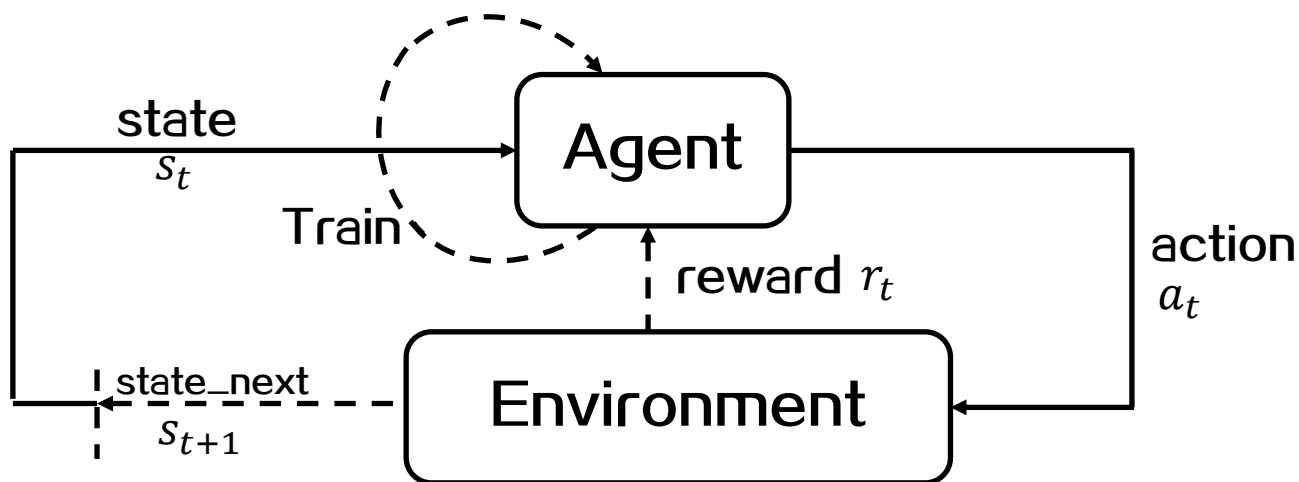
- 预期收获
  - 1. 复习强化学习的基本知识。
  - 2. 深入理解DQN算法原理及其调参细节。

- 强化学习的概念
  - 强化学习是指与环境相互作用的**智能体**，通过反复训练，为指定工程或项目的**顺序决策问题**学习**最优策略**。
- 案例
  - 2016年，AlphaGo Master击败围棋大师李世石；
  - 2019年1月25日，AlphaStar在《星际争霸2》中以10:1击败了人类顶级职业玩家；
  - 2019年4月13日，OpenAI在《Dota2》的比赛中战胜了世界冠军OG；
  - 2020年5月，王者荣耀觉悟AI实现吊打各平台主播；

## • 基本架构

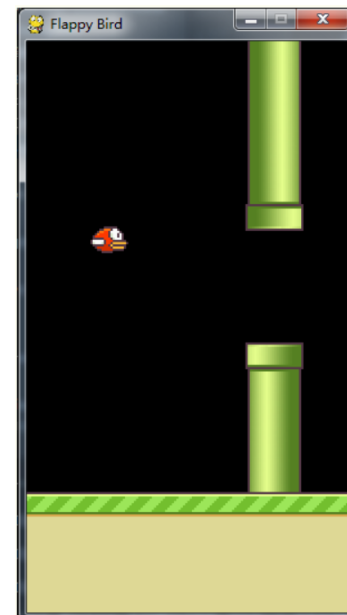
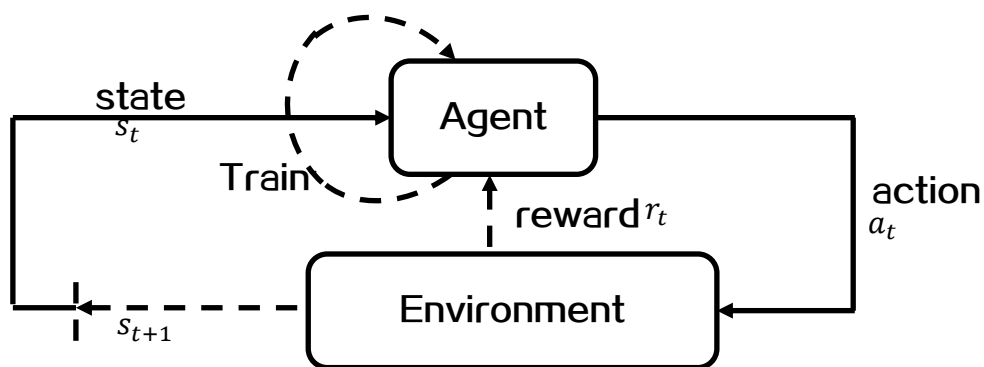
– 强化学习的基本框架一般包含：智能体、环境、动作、状态、奖励这5个部分。

- 智能体：依据当前的状态来做出最优行为决策；
- 动作：预先设置的行为决策；
- 环境：根据输入的动作来更新下一个阶段的状态；
- 状态：表征事物性质的特征；
- 奖励：用于评价当前动作的好坏程度；



## • 基本架构

- **Flappy Bird小游戏**: 障碍物以一定速度往左前行, 小鸟拍打翅膀向上或向下飞翔来避开障碍物, 如果碰到障碍物, 游戏就GAME OVER!
  - 智能体: 依据当前的状态来对小鸟进行操作;
  - 动作: 向上或向下;
  - 环境: 游戏的运行机制;
  - 状态: 根据卷积神经网络提取出来的特征;
  - 奖励: 游戏继续给正奖励, 失败则给予大的负奖励;



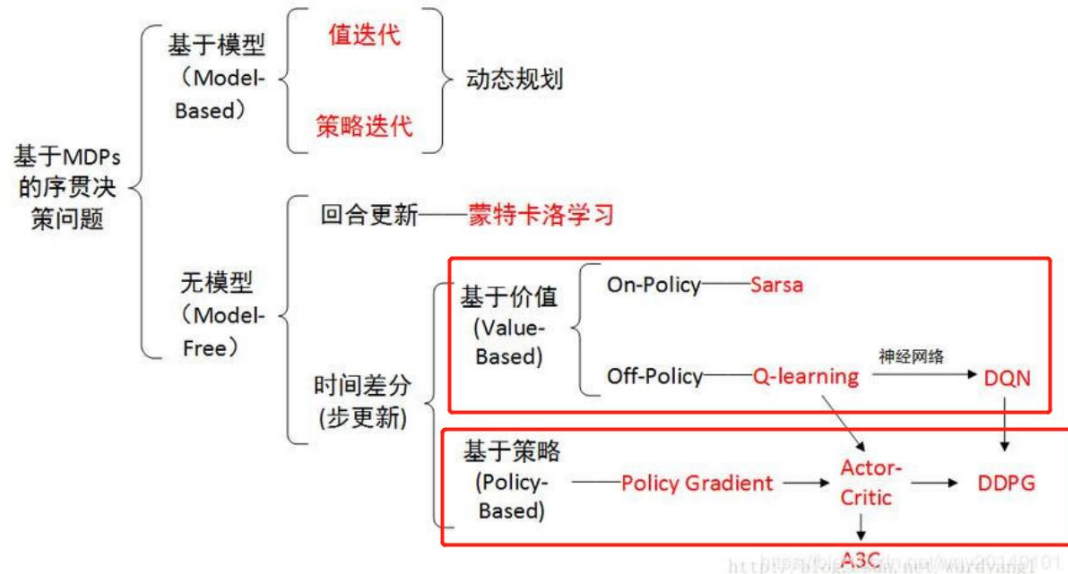
## • 分类

– 强化学习的分类：**免模型学习**、有模型学习

• 分类的标准：智能体能否完整的了解或学习到所在环境的模型。

– 免模型学习：不依赖于环境进行建模能够对所在环境有完整的了解。

• 主要代表算法是：Q Learning、Policy Gradient等



- Q learning算法

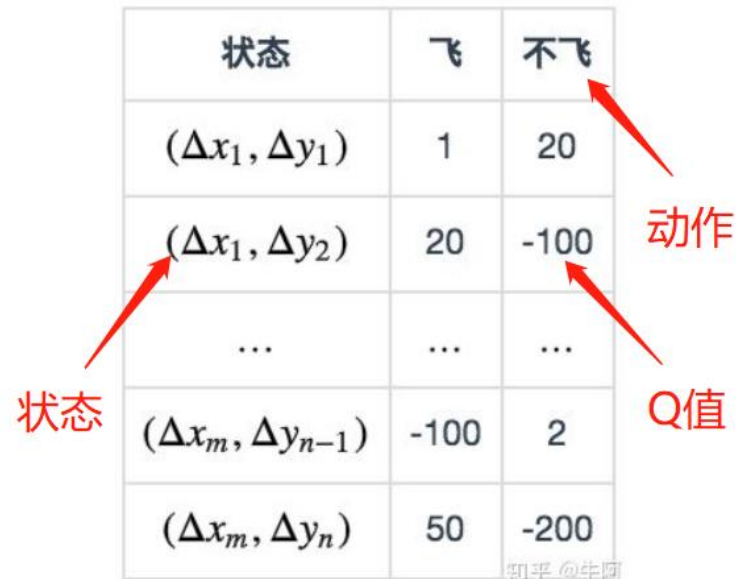
- Q learning算法是基于价值的算法，其主要思想是将状态（state）和动作（action）构建一张Q-table来存储Q值，然后依据Q值来选取能够获得最大收益的动作。

- Q值指的是在当前状态下预期的长期收益。

$$Q_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

**奖励：**在给定状态下收到的即时反馈。

**Q值：**在给定状态下预期得到的所有奖励加权后的总和。



状态	飞	不飞
$(\Delta x_1, \Delta y_1)$	1	20
$(\Delta x_1, \Delta y_2)$	20	-100
...	...	...
$(\Delta x_m, \Delta y_{n-1})$	-100	2
$(\Delta x_m, \Delta y_n)$	50	-200

动作

Q值

状态

知乎 @牛阿



## • Deep Q-Network

- Q learning算法面临状态空间或动作空间太大时，导致难以训练Q表。
- 利用神经网络具有较强的函数拟合能力，来逼近值函数，对Q表进行拟合。

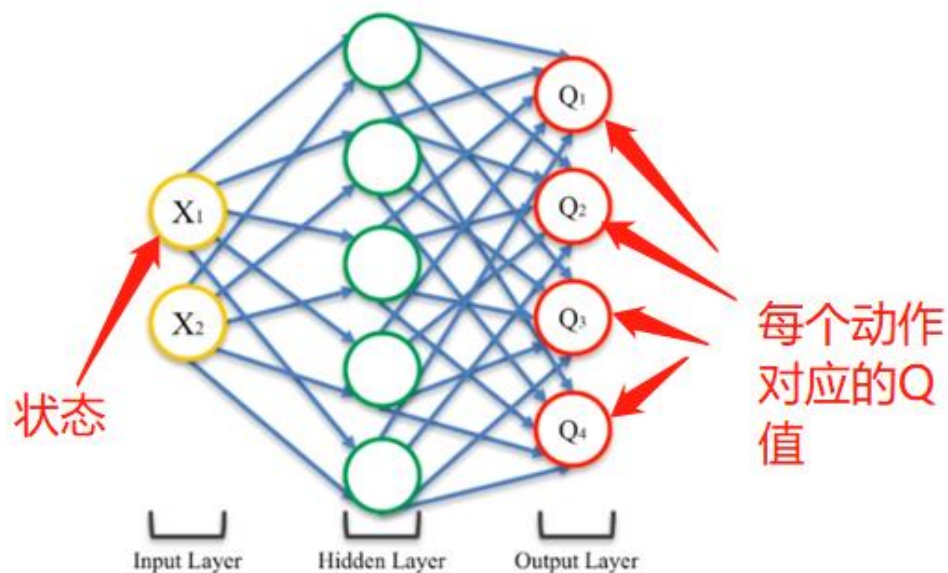
状态	飞	不飞
$(\Delta x_1, \Delta y_1)$	1	20
$(\Delta x_1, \Delta y_2)$	20	-100
...	...	...
$(\Delta x_m, \Delta y_{n-1})$	-100	2
$(\Delta x_m, \Delta y_n)$	50	-200

状态

动作

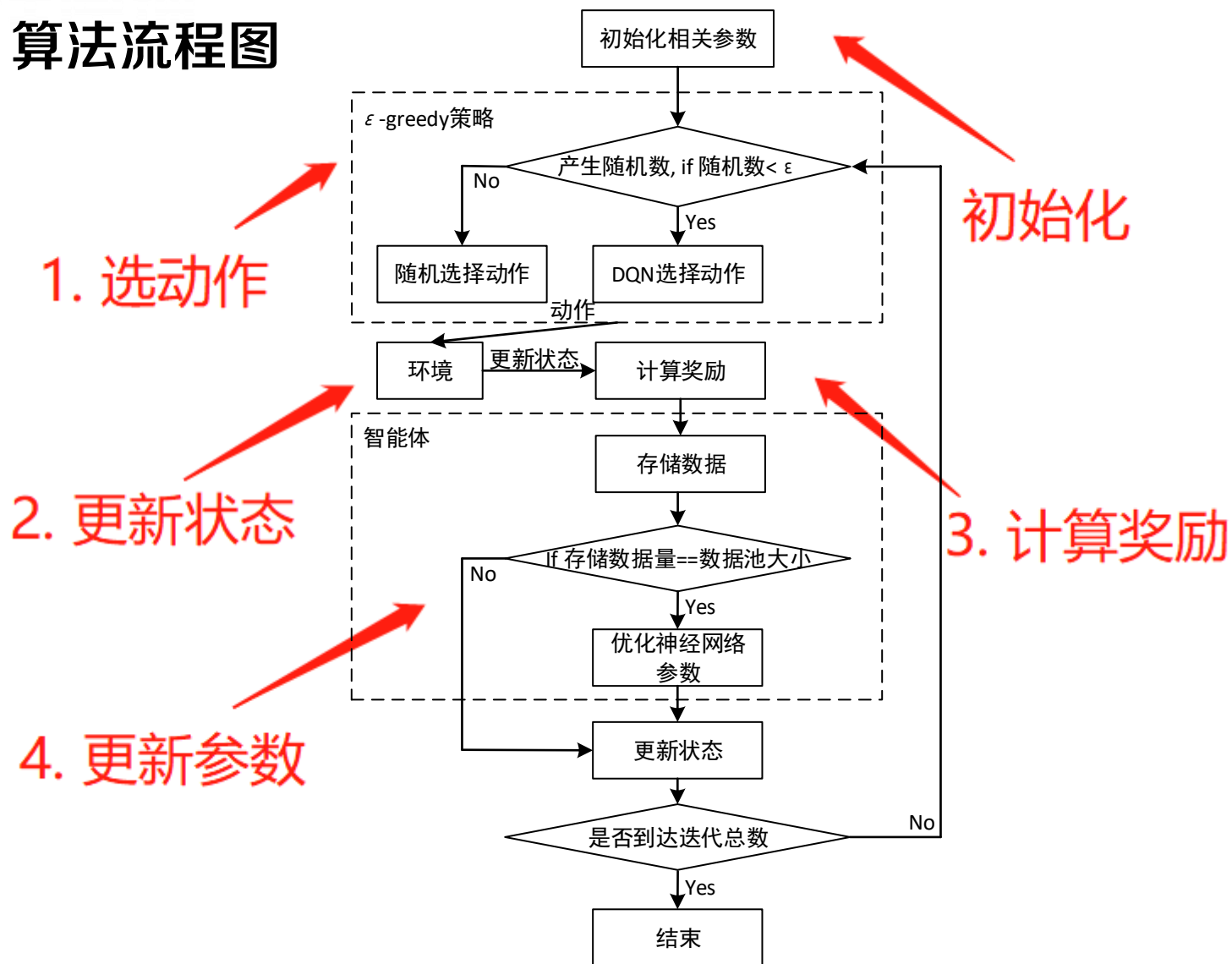
Q值

知乎@牛阿



T	训练智能体在顺序决策问题上做出最优行为策略
I	环境变量、奖励规则、动作集
P	初始化相关参数 For episode = 1, M do: 1. 依据当前状态选择动作 2. 将动作输入到环境中获得下一阶段状态 3. 计算奖励 4. 更新神经网络的参数
O	能做出最优行为决策的智能体
P	解决Q-learning算法中由于动作空间或状态空间太大而难以训练Q表的问题
C	顺序决策问题存在最优行为策略
D	训练成本高，超参数多而难以进行有效地调整
L	Nature

## • 算法流程图



## • 公式推导

- 目标：求出累计奖励最大的策略期望。

$$\max_{\pi} E[\sum_{t=0}^H \gamma^t R(S_t, A_t, S_{t+1}) | \pi]$$

- 贝尔曼方程：通过bellman方程求解马尔科夫决策过程的最佳决策序列，通过状态价值函数 $V_{\pi}(s)$ 来评价当前状态的好坏。

$$V_{\pi}(s) = E(U_t | S_t = s)$$

$$V_{\pi}(s) = E_{\pi} [R_{t+1} + \gamma [R_{t+2} + \gamma [\dots]] | S_t = s]$$

$$V_{\pi}(s) = E_{\pi} [R_{t+1} + \gamma V(s') | S_t = s]$$

- 最优状态价值函数 $V^*(s)$

$$V^*(s) = \max_{\pi} E[\sum_{t=0}^{\dot{H}} \gamma^t R(S_t, A_t, S_{t+1}) | \pi, s_0 = s]$$

- 转化为最优价值动作函数

$$Q^*(s, a) = \sum_{s'} P(s' | s, a) (R(s, a, s') + \gamma \max_{a'} Q_k^*(s', a'))$$

- 公式推导

- 时间差分方法：结合了蒙特卡罗的采样方法和动态规划方法的 bootstrapping(利用后继状态的值函数估计当前值函数)来进行快速的单步更新。

- Q值更新公式：

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

其中 $\alpha$ 为学习率， $\gamma$ 为奖励折扣因子， $\gamma$ 越接近于1代表它越有远见会着重考虑后续状态的价值， $\gamma$ 越接近于0代表就会变得近视只考虑当前利益的影响。

- 神经网络更新公式：

$$\theta_{t+1} = \theta_t + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s', a')] \nabla Q(s, a; \theta)$$

经验回放：为打破数据间的关联性，避免忘记以前的经验，从而设立经验池，随机抽取数据更新神经网络，存储形式 $\langle s, a, r, s' \rangle$ 。

- 过估计问题

- 普通的DQN会出现Q值过估计（overestimate）的问题。

- 具体表现：对某个动作的价值过高的估计从而影响判决性能。

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

- 动作选择：  $Q(s, a)$

- 动作评估：  $r + \gamma \max_{a'} Q(s', a')$

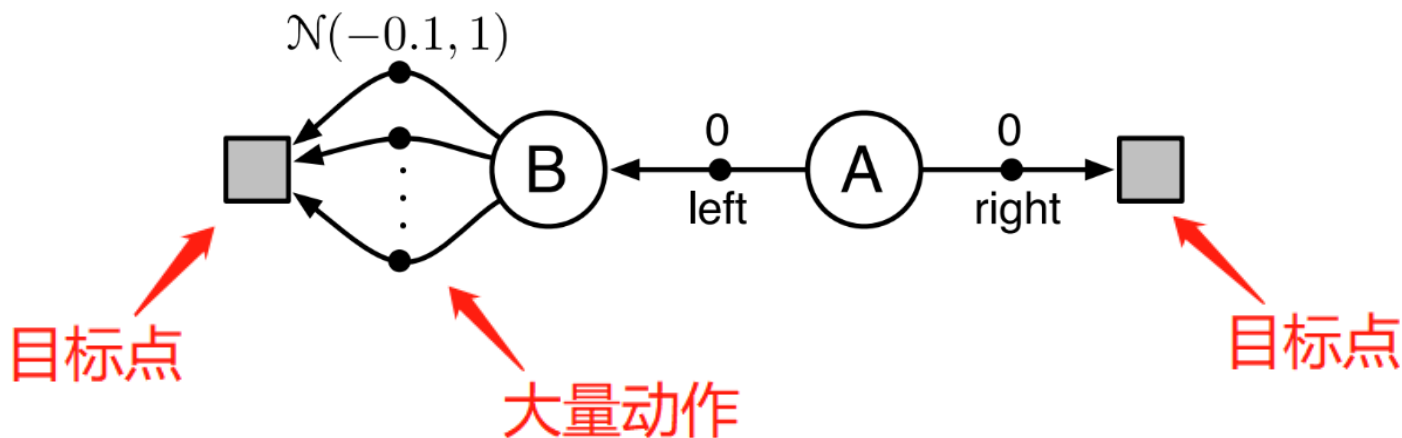
- 产生原因：

- 环境：环境中存在噪声和随机性，导致奖励的扰动。

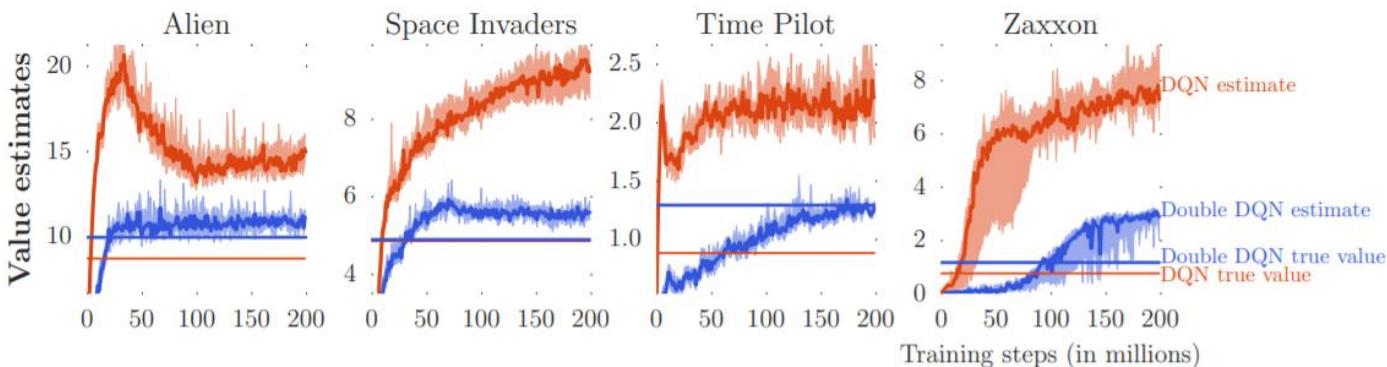
- 模型：弹性函数的逼近不足，且使用了同一个Q值预估模型进行动作选择和动作评估。

## • 过估计问题

- 初始状态A、中间状态B、目标点；
- 存在两条路径：
  - A→right→目标点；✓
  - A→left→B→大量动作→目标点；✗
- 大量动作：所有动作都从 $\mathcal{N}(-0.1, 1)$  采样，这是均值-0.1和方差1的正态分布。模拟由于环境中的噪声或其他因素扰动导致奖励扰动。
- $Q^*(A, left) \leftarrow Q(A, left) + \alpha[r + \gamma \max_{a'} Q(B, a') - Q(A, left)]$
- $Q^*(A, left) \leftarrow Q(A, left) + \alpha[r + \gamma \max_{a'} Q(-0.05, 0.25, -0.15) - Q(A, left)]$



- Double DQN
  - 使用两个独立的神经网络，一个用于进行动作选择，另一个用于动作评估，分别称之Evaluate\_net和Target\_net;
  - $Q^*(A, left) \leftarrow Q(A, left) + \alpha[r + \gamma \max_{a'} Q_{Target} (B, a') - Q_{Eval} (A, left) ]$
  - 更新方式：
    - Evaluate\_net为DQN网络更新的方式;
    - Target\_net为延迟更新的方式，通过设定超参数，将Evaluate\_net的参数传递给Target\_net;

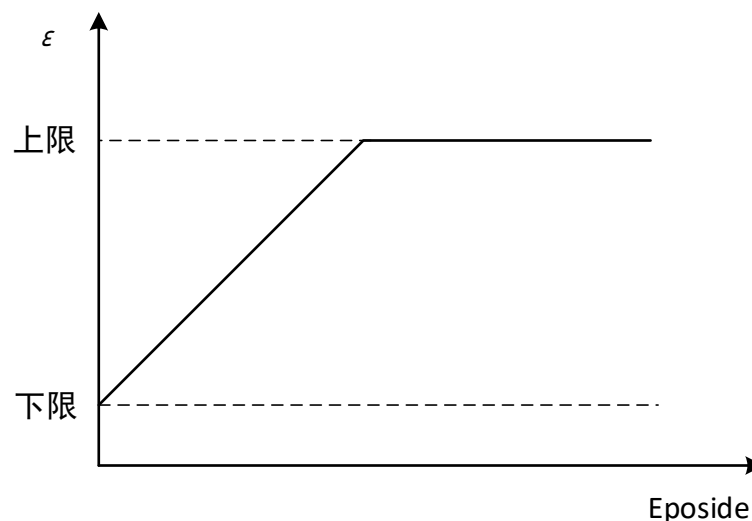
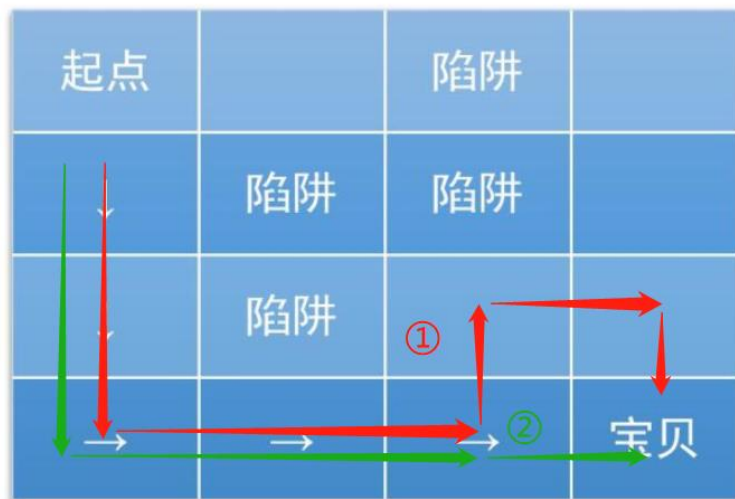




- 超参数调整
  - 1. 预先设置的动作集
    - $\epsilon$ -greedy中的概率  $\epsilon$
  - 2. 环境中的超参数
  - 3. 计算奖励的规则
  - 4. 智能体中的超参数
    - 学习率
    - 奖励衰减系数
    - 数据池大小
    - 批处理数据条数
    - 神经网络的层数和大小
    - Double DQN中Target Net的更新频率

- $\epsilon$ -greedy中的概率  $\epsilon$

- 这个参数的意思是按照  $\epsilon$  的概率依据已有经验选择动作，有  $(1 - \epsilon)$  的概率随机选择动作。
- 通常设置成0.9



- 智能体中的超参数

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

- 学习率 $\alpha$

- 学习率越大，降低神经网络的拟合效果，不易找到最优解；
    - 学习率越小，能够提高性能，但会增加智能体的训练次数和训练时间；

- 奖励衰减系数 $\gamma$

- 越接近于1代表它越有远见会着重考虑后续状态的价值；
    - $\gamma$ 越接近于0代表就只考虑当前奖励的影响；
    - 一般取0.8；

- **智能体中的超参数**

经验回放：为打破数据间的关联性，避免忘记以前的经验，从而设立经验池，随机抽取数据更新神经网络，存储形式  $\langle s, a, r, s' \rangle$ 。

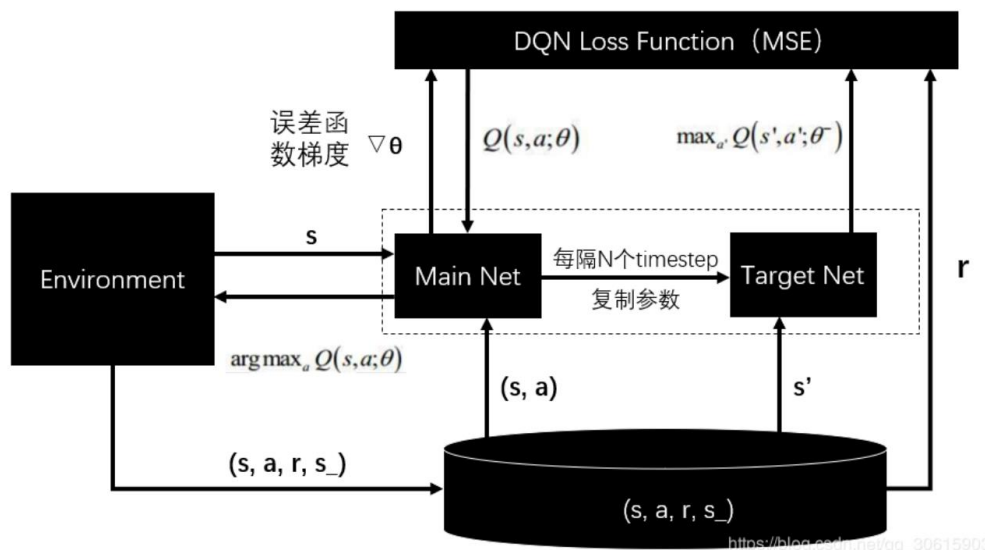
- **数据池大小**

- 数据池容量小，神经网络训练时将易于发散；
- 数据池容量大，存储时间将会增加，从而增加智能体训练时间；

- **批处理数据条数**

- 处理条数过小，神经网络的参数更新速度慢，增加智能体训练时间；
- 处理条数过大，无法打破数据间的关联性，从而出现操作值剧烈波动或发散；

- 智能体中的超参数
  - 神经网络的层数和大小
    - 根据任务的复杂程度而设定；
- Double DQN中Target Net的更新频率
  - 设置过小，将难以解决过估计问题；
  - 设置过大，会增加训练成本和时间；



- 智能体超参数调整总结

超参数	过小	过大
学习率	难以找到最优解	训练成本增加
奖励衰减系数	只考虑当前奖励	过度关注未来奖励
数据池大小	易导致神经网络发散	训练成本增加
批处理数据条数	训练成本增加	无法打破数据间的关联性
神经网络层数和大小	难以拟合Q值	训练成本增加
Target Net更新频率	无法解决过估计问题	训练成本增加

- **DDQN存在的一些问题**
  - **经验池累积会耗费很多的内存和计算力;**
  - **对需要长时奖励的任务上效果表现不佳;**
  - **调参的难度大;**

- Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 529–533.
- <https://medium.com/@ameetsd97/deep-double-q-learning-why-you-should-use-it-bedf660d5295>
- <https://towardsdatascience.com/double-deep-q-networks-905dd8325412>





# 谢谢!

大成若缺，其用不弊。大盈若冲，其用不穷。大直若屈。大巧若拙。大辩若讷。静胜躁，寒胜热。清静为天下正。

