





TensorFlow 2 强化学习入门

Workshop Host



Eliyar Eziz
ML GDE @ Yodo1
github.com/BikerMan



01 RL 101

02 OpenAI Gym

03 Q Learning

04 Deep Q Learning



监督学习

输入特征和标签
学习特征到标签的映射关系

图像识别，回归预测

非监督学习

输入特征
对于特征进行聚类

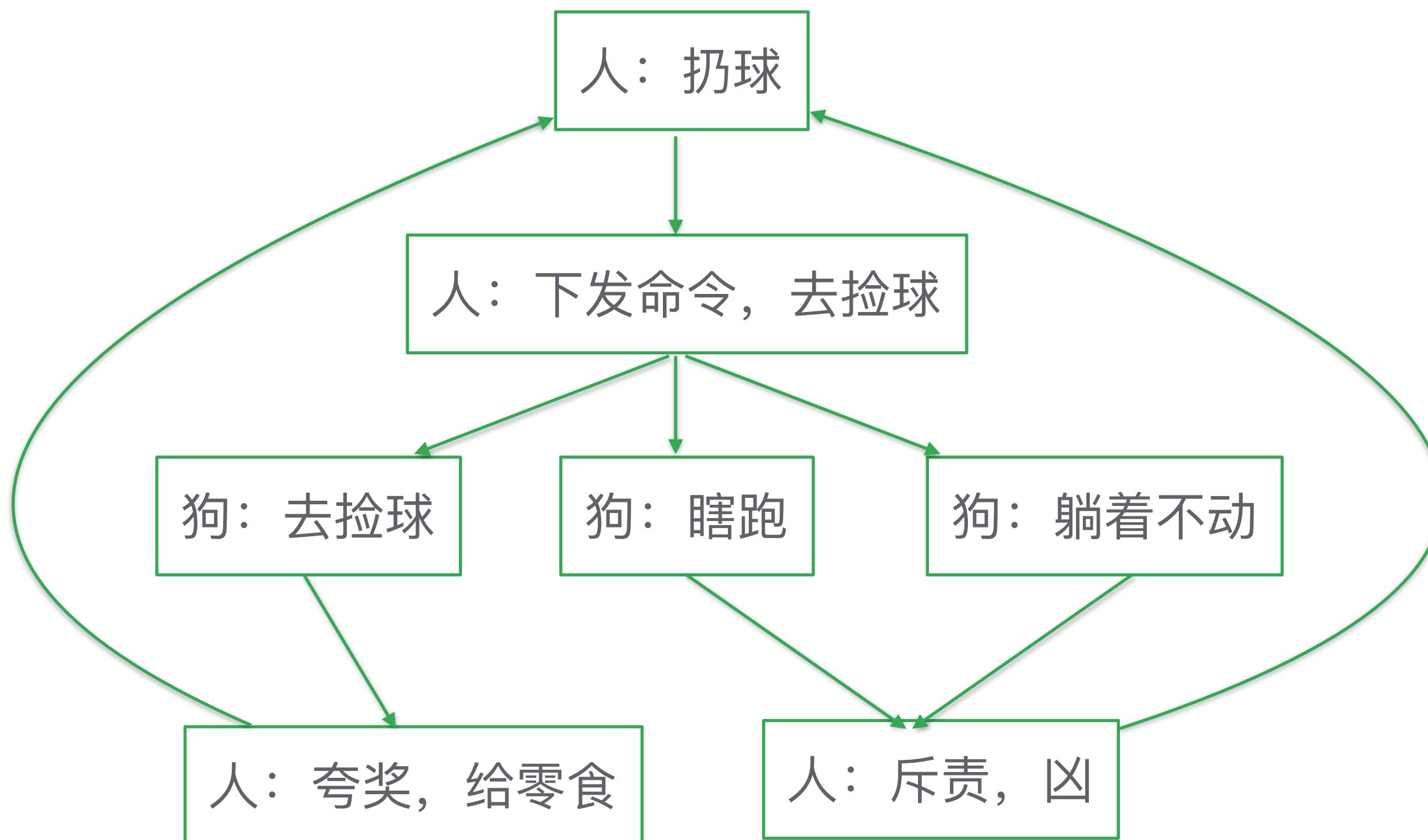
推荐系统

强化学习

通过环境对于输入的反馈
进行学习

AI 玩游戏，下象棋

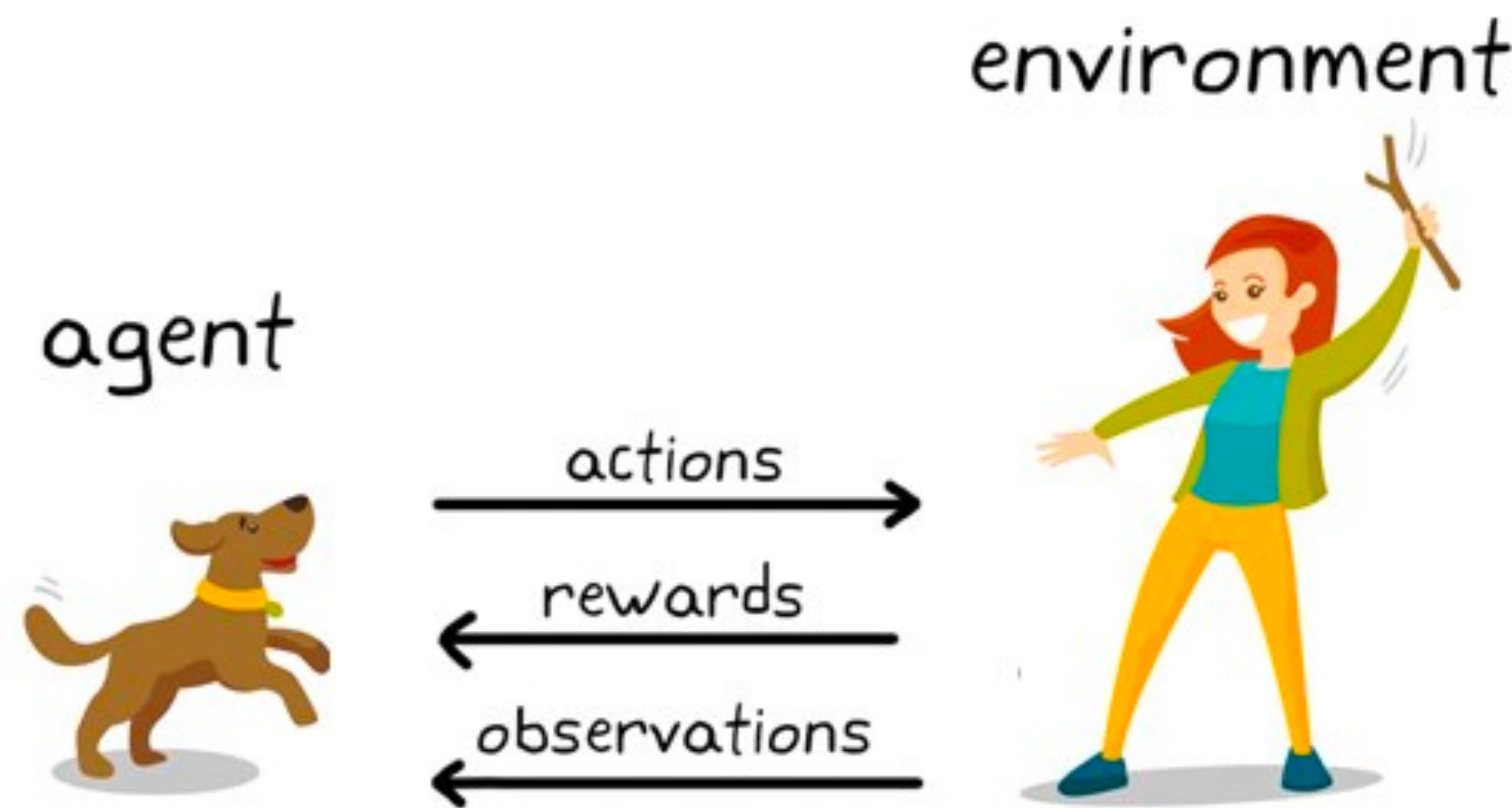
训练小狗捡球



训练小狗捡球

- 智能体 (Agent) : 小狗
- 环境 (Environment) : 智能体所在的场景, 比如操场
- 状态 (State) : 环境和智能体的状态, 包括小狗的状态, 我们发出的命令, 球的位置
- 动作 (Action) : 智能体的行为, 比如跑步或者趴下
- 奖励 (Reward) : 当智能体做出正确的行为的时候给予的奖励, 比如夸奖
- 惩罚 (Penalty) : 当智能体做出错误的行为的时候给予惩罚, 比如斥责
- 策略 (Policy) : 状态到动作的映射, 比如小狗的思考过程

训练小狗捡球





01 RL 101

02 OpenAI Gym

03 Q Learning

04 Deep Q Learning

<https://gym.openai.com/envs/>

The screenshot shows the OpenAI Gym environments page. At the top, there are three circular navigation icons. Below them is a teal header bar with the text "Environments" and "Documentation" on the left, and a logo on the right. On the left side, there is a sidebar with a vertical list of environment categories: "Algorithms", "Atari", "Box2D", "Classic control", "MuJoCo", "Robotics", "Toy text" (with a "EASY" button), and "Third party environments". The "Box2D" category is currently selected, indicated by a blue border. The main content area is titled "Box2D" and describes "Continuous control tasks in the Box2D simulator". It features five environment cards arranged in two rows. The first row contains "BipedalWalker-v2" (Episode 3), "BipedalWalkerHardcore-v2" (Episode 1), and "CarRacing-v0" (Episode 1). The second row contains "LunarLander-v2" (Episode 2) and "LunarLanderContinuous-v2" (Episode 2). Each card includes a small screenshot of the game and a brief description of the task.

Environments Documentation

Algorithms

Atari

Box2D

Classic control

MuJoCo

Robotics

Toy text **EASY**

Third party environments ↗

Box2D
Continuous control tasks in the Box2D simulator.

BipedalWalker-v2
Train a bipedal robot to walk.
Episode 3

BipedalWalkerHardcore-v2
Train a bipedal robot to walk over rough terrain.
Episode 1

CarRacing-v0
Race a car around a track.
Episode 1

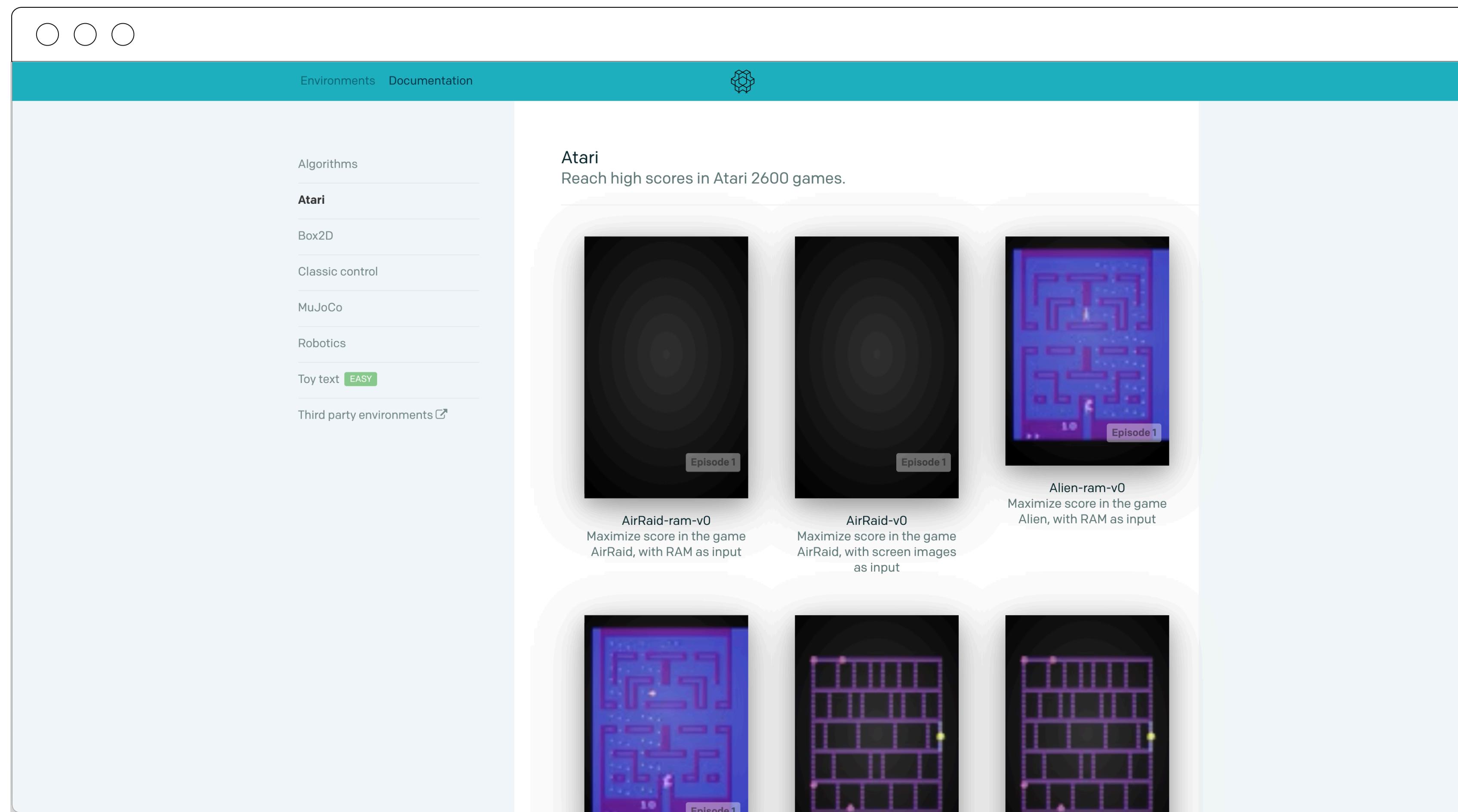
LunarLander-v2
Navigate a lander to its landing pad.
Episode 2

LunarLanderContinuous-v2
Navigate a lander to its landing pad.
Episode 2

Environments Documentation

OpenAI

<https://gym.openai.com/envs/>



```
import gym

# 初始化 Taxi-v3 环境
env = gym.make("Taxi-v3")
# 渲染环境当状态
# 需要注意，不同的环境会有不同的渲染模式。Taxi-v3 环境默认以字符串形式渲染。
env.render()
```

输出如下

```
+-----+
| R: | : : G | |
| : | : : |
| : | : : |
| : | : | : |
| Y | : | B: |
+-----+
```

```
# 三个关键方法
```

```
# 重置环境
```

```
env.reset()
```

```
# 采取动作
```

```
env.step(action)
```

```
# env.step 返回以下四个元素
```

```
# - observation: 智能体观察到的环境, 也就是 state
```

```
# - reward: 环境对于动作的奖惩
```

```
# - done: 当前一轮游戏是否结束
```

```
# - info: 其他附带信息, 用于 debug
```

```
# 渲染当前的状态
```

```
env.render()
```

```
# 两个关键属性, 分别表示动作空间和观察空间
```

```
# 动作空间, Taxi-v3 环境中动作空间为 `Discrete(6)` , 代表 6 个动作可选, 分别是上下左右移动、接乘客上车和乘客下车
```

```
env.action_space
```

```
# 观察空间, Taxi-v3 环境中动作空间为 `Discrete(500)` , 代表 5 × 5 个车的位置, 5 × 4 个乘客位置。
```

```
env.observation_space
```

Coding Time
Random Agent



01 RL 101

02 OpenAI Gym

03 Q Learning

04 Deep Q Learning

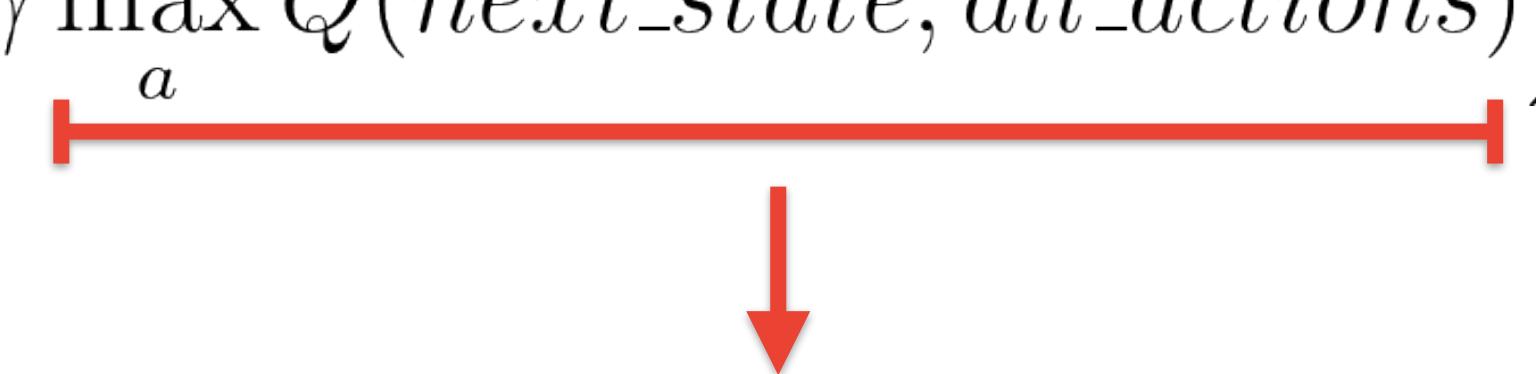


动作价值函数 Q

$$Q(state, action) = (1-\alpha)Q(state, action) + \alpha \left(reward + \gamma \max_a Q(next_state, all_actions) \right)$$

- α 是学习率，取值范围 0-1。新知识对于 Q 值更新的影响比例。
- γ 是历史经验的权重，取值范围 0-1。
 - 值越接近于 1 表示模型更加注重长期回报，越接近 0 表示模型越注重短期回报。
 - 如果设定为 0，那么模型只学习本次回报，抛弃之前的历史回报经验。

动作价值函数 Q

$$Q(state, action) = (1-\alpha)Q(state, action) + \alpha \left(reward + \gamma \max_a Q(next_state, all_actions) \right)$$


下一个状态所有动作中的最高奖励

假设我们用全 0 初始化了 Q 表, $\alpha=0.4$, $\gamma=0.8$ 。经过第一步, 发现了在状态 218 时如果向上移动, 那么会被扣 1 分, 状态变更为 217

$$\begin{aligned} Q[218, 1] &= (1 - 0.4) * 0 + 0.4 * (-1 + 0.7 * \max(Q[217])) \\ &= 0.4 \end{aligned}$$

Q Learning 步骤

1. 初始化一个全 0 的 Q 表。
2. 根据当前状态选择一个动作，探索阶段可以一定比例随机选择来探索更多的可能性。
3. 获得本次收益和下一个状态，然后通过行为将状态转换为下一个状态。
4. 使用上述价值函数更新 Q 表的值。
5. 不断重复步骤 2-4，尽可能完善 Q 表。

初始化 Q 表

Q-Table	动作					
	下 (0)	上 (1)	右 (2)	左 (3)	上车 (4)	下车 (5)
状态	0	0	0	0	0	0

	330	0	0	0	0	0

	499	0	0	0	0	0

训练

Q-Table	动作					
	下 (0)	上 (1)	右 (2)	左 (3)	上车 (4)	下车 (5)
状态	0	-2.132141	-1.4123	1.3123	0.432451	-1.312321

	330	32.54354	12.432432	6.24142	1.44313	3.134213

	499	9.312345	2.345555	3.124566	2.12354	2.3144132

Coding Time

Q-Learning Agent



Experts

01 RL 101

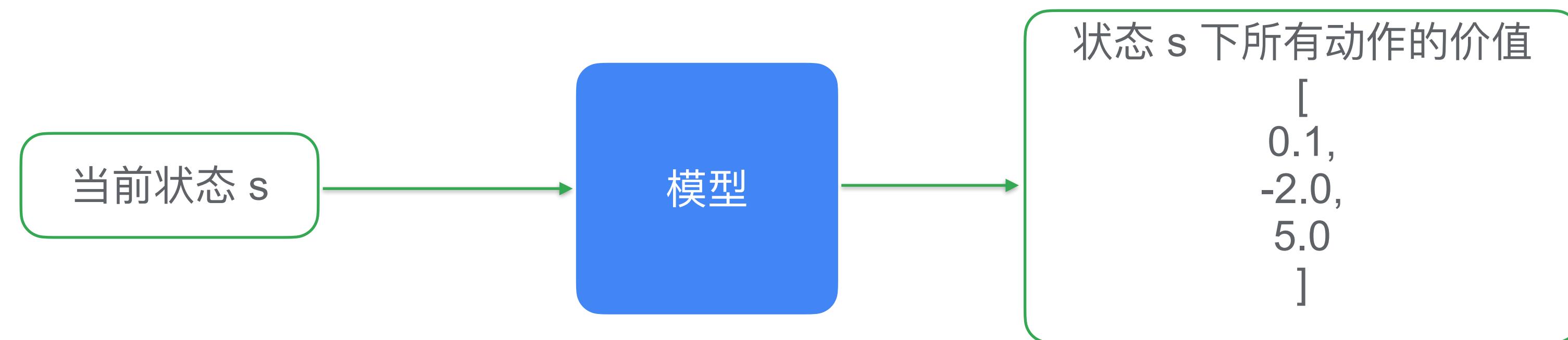
02 OpenAI Gym

03 Q Learning

04 Deep Q Learning

Deep Q Learning

- 使用模型替代了 Q Table，解决了 Q 表的维度爆炸
- 引入了经验池和经验回放概念





Deep Q Learning

1. 初始化模型，经验池。
2. 根据当前状态 S 选择一个动作 a_t ，探索阶段可以一定比例随机选择来探索更多的可能性。
3. 记录状态，动作，回报，下一个状态和是否完成到经验池。
4. 每 n 步进行一次经验回放，训练模型。
5. 更新探索比例参数 ϵ 。
6. 重复步骤 2-5，不断优化模型。

Coding Time

Deep Q-Learning Agent

TensorFlow 2 实战

By 艾力

<https://item.jd.com/13320676.html>

通过简洁易懂的语言，循序渐进的学习 TensorFlow 2
涵盖图像处理，自然语言处理，GAN 和强化学习知识点

异步图书
www.epubit.com



艾力〇编著



通过**简洁易懂**的语言，帮助读者**循序渐进**地学习 TensorFlow 2
精选图像处理和自然语言处理两方面的实例，帮助读者**掌握深度学习**的应用
直击生成对抗网络和强化学习的难点，帮助读者**精通深度学习**

中国工信出版集团 人民邮电出版社
POSTS & TELECOM PRESS

Thank you!



Eliyar Eziz
ML GDE @ Yodo1
github.com/BikerMan