

Машинное обучение

Лекция 1.

Часть 3.

Обучение с подкреплением.

Продолжение

Сравнение ОП с обучением с учителем

- Цель обучения с учителем – научиться обобщать, располагая лишь фиксированным набором данных с ограниченным количеством примеров. Каждый пример состоит из входа и желательного выхода (или метки), так что реакция на выбор агента следует незамедлительно.
- Напротив, в ОП акцент ставится на последовательных действиях, которые можно предпринять в конкретной ситуации. В данном случае единственное, что дает учитель, – сигнал вознаграждения. Какое действие правильно при данных условиях, неизвестно, в отличие от обучения с учителем.

Основные характеристики ОП:

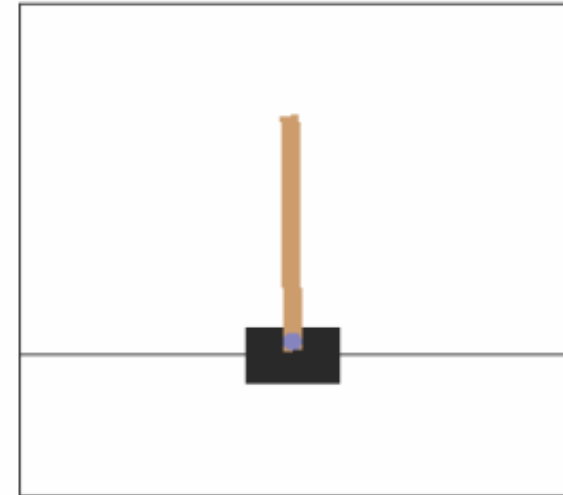
- вознаграждение может быть плотным, разреженным или приходиться с большим запаздыванием. Во многих случаях вознаграждение становится известно только в конце задания (например, в шахматах);
- задача последовательная и зависящая от времени – выбранные действия могут влиять на следующие действия, которые, в свою очередь, влияют на возможные вознаграждения и состояния;
- агент должен совершать действия, которые с высокой вероятностью приведут к достижению цели (использование), но в то же время должен пробовать иные действия, чтобы другие части окружающей среды не остались неисследованными (исследование). Эту двойственность называют дилеммой (или компромиссом) исследования использования, она призвана решить трудную проблему поиска баланса между исследованием и использованием окружающей среды. Она важна также и потому, что, в отличие от обучения с учителем, агент ОП может влиять на окружающую среду, т. к. вправе собирать новые данные, коль скоро считает это полезным;
- окружающая среда стохастическая и недетерминированная, и агент должен учитывать это в процессе обучения и для предсказания следующего действия. Многие компоненты ОП можно спроектировать так, что они будут выдавать либо только одно детерминированное значение, либо диапазон значений, каждому из которых сопоставлена вероятность.

Главные элементы, необходимые для создания алгоритмов глубокого ОП

- Язык программирования.
 - Python – первый кандидат для разработки алгоритмов машинного обучения благодаря своей простоте и богатству написанных на нем сторонних библиотек.
- Библиотека глубокого обучения.
 - TensorFlow,
 - PyTorch
 - Caffe.
- Окружающая среда.
 - OpenAI Gym предлагает общий интерфейс, а также широкий спектр окружающих сред.

Пример базового цикла ОП

```
import gym
# создать окружающую среду
env = gym.make("CartPole-v1")
# привести среду в исходное состояние
# перед началом
env.reset()
# повторять 10 раз
for i in range(10):
    # предпринять случайное действие
    env.step(env.action_space.sample())
    # нарисовать состояние игры
    env.render()
# закрыть окружающую среду
env.close()
```



Выводимое состояние



Более полный алгоритм ОП

```
import gym
# создать и инициализировать окружающую среду
env = gym.make("CartPole-v1")
env.reset()
# сыграть 10 игр
for i in range(10):
    # инициализировать переменные
    done = False
    game_rew = 0
    while not done:
        # выбрать случайное действие
        action = env.action_space.sample()
        # выполнить один шаг взаимодействия с окружающей средой
        new_obs, rew, done, info = env.step(action)
        game_rew += rew
        # если завершено, напечатать полное вознаграждение в игре и сбросить среду
        if done:
            print('Эпизод %d завершен, Вознаграждение:%d' % (i, game_rew))
            env.reset()
```

Результаты:

Эпизод: 0, Вознаграждение:13
Эпизод: 1, Вознаграждение:16
Эпизод: 2, Вознаграждение:23
Эпизод: 3, Вознаграждение:17
Эпизод: 4, Вознаграждение:30
Эпизод: 5, Вознаграждение:18
Эпизод: 6, Вознаграждение:14
Эпизод: 7, Вознаграждение:28
Эпизод: 8, Вознаграждение:22
Эпизод: 9, Вознаграждение:16

В следующей таблице показаны результаты метода `step()` для последних четырех действий:

Observation	Reward	Done	Info
<code>[-0.05356921, -0.38150626, 0.12529277, 0.9449761]</code>	1.0	False	<code>{}</code>
<code>[-0.06119933, -0.57807287, 0.14419229, 1.27425449]</code>	1.0	False	<code>{}</code>
<code>[-0.07276079, -0.38505429, 0.16967738, 1.02997704]</code>	1.0	False	<code>{}</code>
<code>[-0.08046188, -0.58197758, 0.19027692, 1.37076617]</code>	1.0	False	<code>{}</code>
<code>[-0.09210143, -0.3896757, 0.21769224, 1.14312384]</code>	1.0	True	<code>{}</code>

А.В.Гаврилов
кафедра ВТ, НГТУ

Некоторые окружающие среды:

- Gym Atari (<https://gym.openai.com/envs/#atari>). Включает игры для Atari 2600, в которых входными данными служат изображения на экране. Полезны для измерения качества алгоритмов ОП на широком спектре игр с одним и тем же пространством наблюдений.
- Gym Classic control (https://gym.openai.com/envs/#classic_control). Классические игры, которые можно использовать для оценки и отладки алгоритма.
- Gym MuJoCo (<https://gym.openai.com/envs/#mujoco>). Включает непрерывные задачи управления (например, Ant и HalfCheetah), построенные поверх MuJoCo, физического движка с платной лицензией (бесплатная лицензия доступна студентам).
- MalmöEnv (<https://github.com/Microsoft/malmo>). Среда, построенная поверх Minecraft.
- Pommerman (<https://github.com/MultiAgentLearning/playground>). Отличная среда для обучения многоагентных алгоритмов. Игра Pommerman – вариант знаменитой игры Bomberman.
- Roboschool (<https://github.com/openai/roboschool>). Среда эмуляции роботов, интегрированная с OpenAI Gym. Включает копию окружающей среды MuJoCo, показанной на рис. 2.10, две интерактивные среды для повышения надежности агента и одну многопользовательскую среду.
- Duckietown (<https://github.com/duckietown/gym-duckietown>). Эмулятор беспилотного автомобиля с различными картами местности и препятствиями.
- PLE (<https://github.com/ntasfi/PyGame-Learning-Environment>). Включает много аркадных игр, в т. ч. Monster Kong, FlappyBird и Snake.
- Unity ML-Agents (<https://github.com/Unity-Technologies/ml-agents>). Среда, построенная поверх Unity с реалистичными физическими законами. ML-agents допускает большую свободу и позволяет создавать собственные окружающие среды с использованием Unity.
- CoinRun (<https://github.com/openai/coinrun>). Среда, предназначенная для решения проблемы переобучения в ОП. Генерирует различные среды для обучения и тестирования.
- DeepMind Lab (<https://github.com/deepmind/lab>). Комплект трехмерных сред для решения задач навигации и сборки пазлов.
- DeepMind PySC2 (<https://github.com/deepmind/pysc2>). Среда для обучения сложной игре, StarCraft II.

Литература

- Саттон Р. С., Барто Э. Дж. Обучение с подкреплением. М.: ДМК Пресс, 2020.
- Лонца А. Обучение с подкреплением на Python. – М.: ДМК Пресс, 2020.
- Памперла М., Фергюсон К. Глубокое обучение и игра в го. М.: ДМК Пресс, 2020.