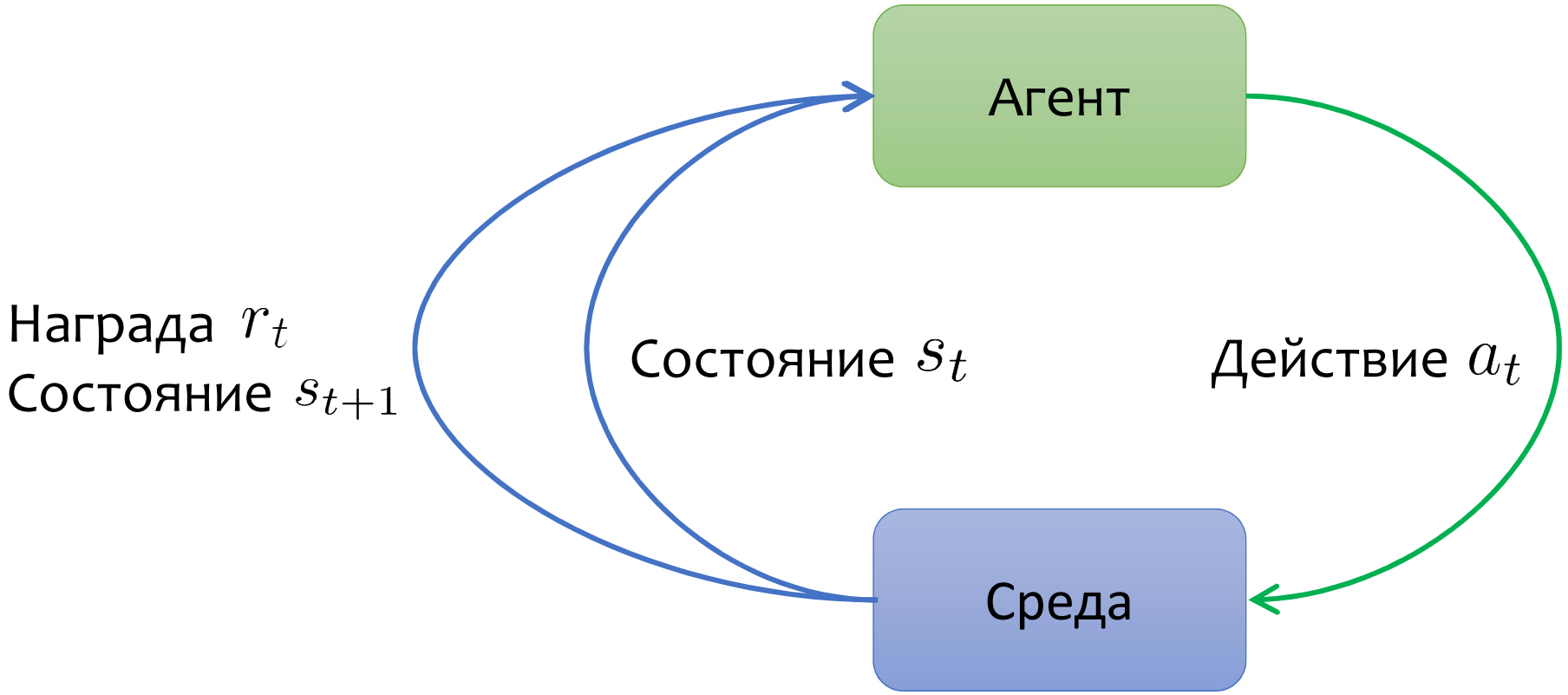
Deep Q-learning

Подготовил:

Пугачев Александр, 151

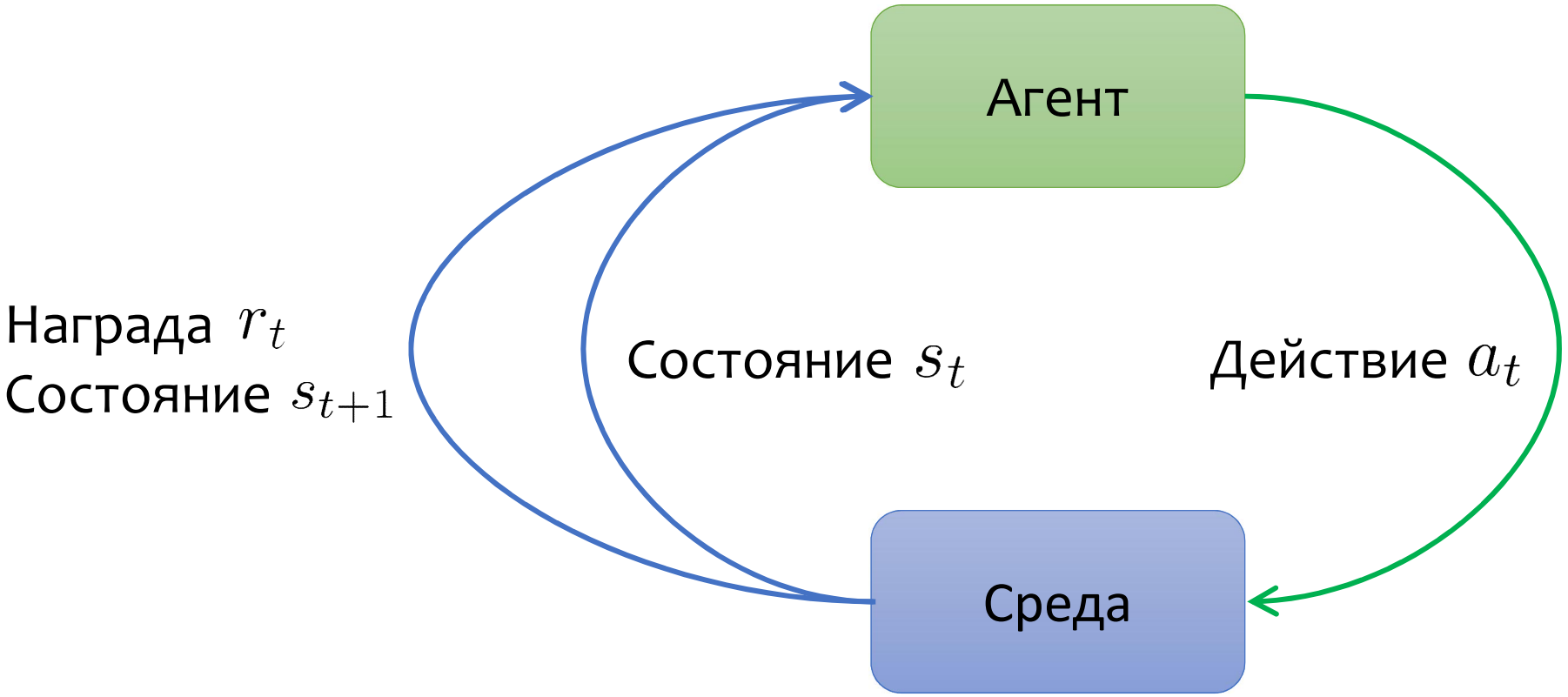
Reinforcement learning



Reinforcement learning

- Агент: платформа, отражающая мяч
- Состояние: пиксельное изображение состояния игры
- Действие: игровое управление (влево, вправо)
- Награда: увеличение/уменьшение счета игры

Как переформулировать математически?



Марковский процесс принятия решений

Определяется как: (S, A, R, P, γ)

- S – множество возможных состояний
- A – множество возможных действий
- R – множество вознаграждений
- P – распределение вероятностей перехода в новое состояние
- γ – discount factor

Марковский процесс принятия решений

- Первый шаг: $t = 0$, среда принимает состояние s_0
- Пока не достигли терминального состояния:
  - Агент выбирает действие a_t
  - Среда генерирует награду $r_t \sim R(\cdot|s_t, a_t)$
  - Среда генерирует новое состояние $s_{t+1} \sim \mathbb{P}(\cdot|s_t, a_t)$
  - Агент получает награду r_t и следующее состояние s_{t+1}

Оптимальная стратегия

Стратегия π – функция из S в A, определяющая какое действие принимать в каждой ситуации

Цель: найти оптимальную стратегию π*, максимизирующую: $\sum_{t>0} \gamma^t r_t$

Оптимальная стратегия π*:

$$\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | \pi\right]$$

V – function & Q – function

Траектория: $s_0,\ a_0,\ r_0,\ s_1,\ a_1,\ r_1, \ldots$

Насколько хорошим является текущее состояние?

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t \geq 0} \gamma^t r_t | s_0 = s, \pi\right]$$

Насколько хорошей является пара состояние-действие?

$$Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{t \geq 0} \gamma^t r_t | s_0 = s, a_0 = a, \pi\right]$$

Уравнение Беллмана

$$Q^*(s,a) = \max_{\pi} \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | s_0 = s, a_0 = a, \pi\right]$$

Q^* удовлетворяет уравнению Беллмана:

$$Q^*(s,a) = \mathbb{E}_{s'}\left[r + \gamma \max_{a'} Q^*\left(s', a'\right) | s, a\right]$$

Value iteration algorithm

Используем уравнение Беллмана в качестве итерации:

$$Q_{i+1}(s,a) = \mathbb{E}_{s'}\left[r + \gamma \max_{a'} Q_i\left(s', a'\right) | s, a\right]$$

$$Q_i \to Q^*$$

В чем проблема?

Нам необходимо вычислять $Q(s, a)$ для каждой пары состояние-действие

Какое решение?

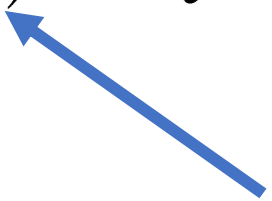
Использовать аппроксиматор $Q(s, a)$

Например, нейросеть!

Нахождение оптимальной стратегии

Используем аппроксимирующую функцию

$$Q(s, a, \theta) \approx Q^*(s, a)$$



параметры - веса

Нахождение оптимальной стратегии

Хотим найти функцию, удовлетворяющую уравнению Беллмана:

$$Q^*(s,a) = \mathbb{E}_{s'}\left[r + \gamma \max_{a'} Q^*(s',a') \,|s,a\right]$$

Forward pass:

Функция потерь: $L_i(\theta_i) = \mathbb{E}_{s,a}\left[(y_i - Q(s,a;\theta_i))^2\right]$

$$y_i = \mathbb{E}'_s\left[r + \gamma \max_{a'} Q(s',a';\theta_{i-1})|s,a\right]$$

Backward pass:

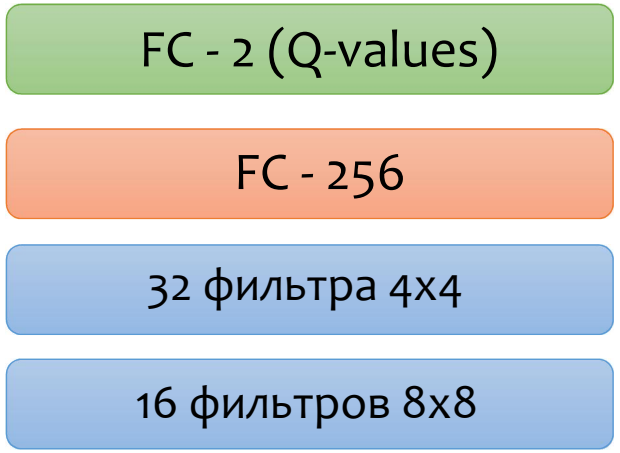
$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}_{s,a,s'}\left[\left(r + \gamma \max_{a'} Q(s',a';\theta_{i-1}) - Q(s,a;\theta_i)\right)\nabla_{\theta_i} Q(s,a;\theta_i)\right]$$

Пример

- Агент: платформа, отражающая мяч
- Состояние: пиксельное изображение состояния игры
- Действие: игровое управление (влево, вправо)
- Награда: увеличение/уменьшение счета игры

Алгоритм

$Q(s, a; \theta)$:



Последний полносвязный слой имеет выход длины 2, соответствующий:

$Q(s_t, a_1)$
$Q(s_t, a_2)$

s_t : 4 картинки 84x84

Experience Replay

- Обучение на последовательных изображениях проблематично:
  - Изображения коррелируют

- Решение: использование Experience Replay:
  - Создадим таблицу replay memory размера N , состоящую из переходов (s_t, a_t, r_t, s_{t+1})
  - Обучаем Q-сеть на случайном наборе переходов из replay memory

Deep Q-learning & Experience Replay

- Инициализируем replay memory D, и веса Q-сети
- Для каждого эпизода:
  - Инициализируем состояние s_1
  - Пока не достигли терминального состояния:
    - Выбираем $a_t = \max_a Q^*(s_t, a; \theta)$
    - Совершаем действие a_t, получаем награду r_t и новое состояние s_{t+1}
    - Сохраняем переход (s_t, a_t, r_t, s_{t+1})
    - Генерируем случайно переход (s_j, a_j, r_j, s_{j+1}) из D
    - $y_j = r_j + \gamma \max_{a'} Q(s_{j+1}, a'; \theta)$
    - Совершаем шаг градиентного спуска: $\left((y_j - Q(s_j, a_j; \theta)\right)^2$

Сравнение с аналогами

Atari Breakout

	SARSA	Contingency	Human	DQN
Average total reward	5.2	6	31	**168**

Сравнение с аналогами

	B. Rider	Breakout	Enduro	Pong	Seaquest	S. Invaders
SARSA	996	5.2	129	-19	665	271
Contingency	1743	6	159	-17	723	268
Human	7456	31	368	-3	28010	3690
DQN	4092	168	470	20	1705	581

Источники

- https://youtu.be/lvoHnicueoE
- https://youtu.be/V1eYniJ0Rnk
- https://arxiv.org/pdf/1312.5602.pdf
- https://habrahabr.ru/post/279729/