Лекция 5: Value Function Approximation

Антон Романович Плаксин

Markov Decision Process

Markov Property

$$\mathbb{P}[S_{t+1}|S_t, A_t] = \mathbb{P}[S_{t+1}|S_1, A_1, S_2, A_2 \ldots, S_t, A_t]$$

$$\mathbb{P}[R_t|S_t, A_t] = \mathbb{P}[R_t|S_1, A_1, S_2, A_2 \ldots, S_t, A_t] = 1$$

Markov Decision Process $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$

- $\mathcal{S}$ — бесконечное пространство состояний
- $\mathcal{A}$ — конечное пространство действий
- $\mathcal{P}$ — неизвестная функция (тензор) вероятностей переходов между состояниями

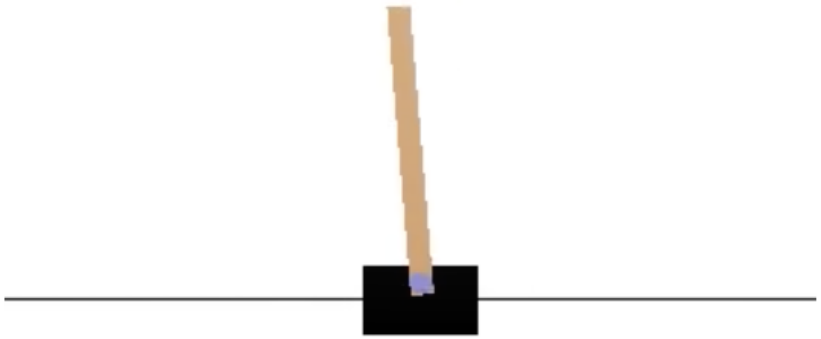
$$\mathcal{P}(s'|s, a) = \mathbb{P}[S_{t+1} = s'|S_t = s, A_t = a]$$

- $\mathcal{R}$ — неизвестная функция (матрица) вознаграждений

$$\mathcal{R}(s, a) = R_t \quad \Leftrightarrow \quad \mathbb{P}[R_t|S_t = s, A_t = a] = 1$$

- $\gamma \in [0, 1]$ — коэффициент дисконтирования

Пример: Cartpole



- Состояния: $\mathbb{R}^4$
  или пиксели с экрана
- Действия: $\rightarrow$, $\leftarrow$, «0»
- Награда: $+1$ на каждом шаге

Пример: Atari Games

- Состояния: пиксели с экрана
- Действия: $\rightarrow$, $\leftarrow$, «0»
- Награда: очки в игре

Monte-Carlo Algorithm

Пусть $Q(s,a) = 0$, $N(s,a) = 0$ и $\varepsilon = 1$.
Для каждого эпизода $k \in \overline{1,K}$ делаем:

- Согласно $\pi = \varepsilon$-greedy(Q) получаем траекторию $\tau = (S_0, A_0, \ldots, S_T)$ и награды $(R_0, \ldots, R_{T-1})$. По ним определяем $(G_0, \ldots, G_{T-1})$.
- Для каждого $t \in \overline{0,T-1}$ обновляем Q и N:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t) + 1}\big(G_t - Q(S_t, A_t)\big),$$

$$N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$$

Определяем $\varepsilon = 1/k$

Monte-Carlo Algorithm

Пусть $Q(s,a) = 0$, $N(s,a) = 0$ и $\varepsilon = 1$.
Для каждого эпизода $k \in \overline{1,K}$ делаем:

- Согласно $\pi = \varepsilon$-greedy(Q) получаем траекторию $\tau = (S_0, A_0, \ldots, S_T)$ и награды $(R_0, \ldots, R_{T-1})$. По ним определяем $(G_0, \ldots, G_{T-1})$.
- Для каждого $t \in \overline{0, T-1}$ обновляем Q и N:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t) + 1}\big(G_t - Q(S_t, A_t)\big),$$

$$N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$$

Определяем $\varepsilon = 1/k$

SARSA Algorithm

Пусть $Q(s, a) = 0$ и $\varepsilon = 1$.
Для каждого эпизода k делаем:
Пока эпизод не закончен делаем:

- Находясь в состоянии S_t совершаем действие $A_t \sim \pi(\cdot|S_t)$, где $\pi = \varepsilon$-greedy(Q), получаем награду R_t, переходим в состояние S_{t+1}, совершаем действие $A_{t+1} \sim \pi(\cdot|S_{t+1})$
- По $(S_t, A_t, R_t, S_{t+1}, A_{t+1})$ обновляем Q:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_t + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))$$

Полагаем, например, $\varepsilon = 1/k$

Q-Learning Algorithm

Пусть $Q(s, a) = 0$ и $\varepsilon = 1$.
Для каждого эпизода k делаем:
Пока эпизод не закончен делаем:

- Находясь в состоянии S_t совершаем действие $A_t \sim \pi(\cdot|S_t)$, где $\pi = \varepsilon$-greedy(Q), получаем награду R_t переходим в состояние S_{t+1}.
- По (S_t, A_t, R_t, S_{t+1}) обновляем Q:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_t + \gamma \max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t))$$

Полагаем, например, $\varepsilon = 1/k$

Аппроксимация

Идея аппроксимации

- Задать функцию $Q^\theta(s, a)$, параметризованную $\theta \in \mathbb{R}^N$
- Найти такое θ, чтобы

$$Q^\theta(s, a) \approx q_\pi(s, a) \quad \text{или} \quad Q^\theta(s, a) \approx q_*(s, a)$$

Аппроксимация

Идея аппроксимации

- Задать функцию $Q^\theta(s, a)$, параметризованную $\theta \in \mathbb{R}^N$
- Найти такое θ, чтобы

$$Q^\theta(s, a) \approx q_\pi(s, a) \quad \text{или} \quad Q^\theta(s, a) \approx q_*(s, a)$$

Дифференцируемые аппроксиматоры

- Линейные комбинации функций
- Нейронные сети

Линейная комбинация функций

$$Q^{\theta}(s,a) = \sum_{i=1}^{n} \theta_i \varphi_i(s,a),$$

где $\varphi_i(s,a)$ — заданные функции

Линейная комбинация функций

$$Q^\theta(s,a) = \sum_{i=1}^{n} \theta_i \varphi_i(s,a),$$

где $\varphi_i(s,a)$ — заданные функции

Градиент

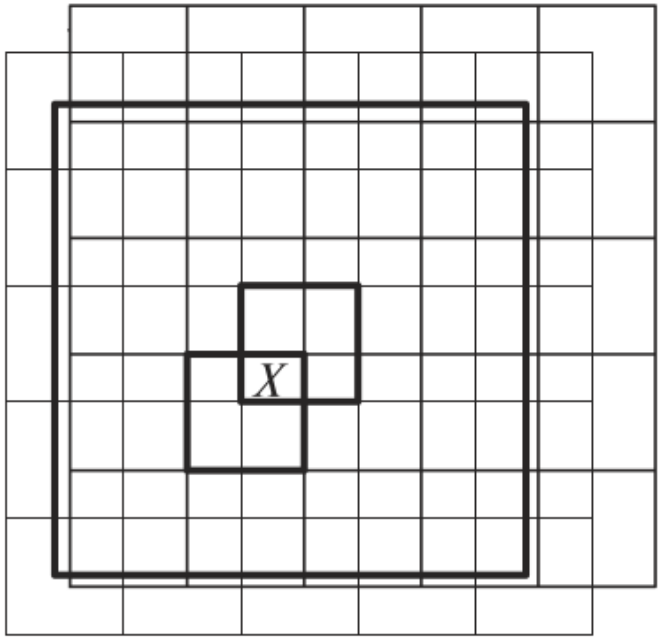
$$\nabla_\theta Q^\theta(s,a) = \begin{pmatrix} \varphi_1(s,a) \\ \vdots \\ \varphi_n(s,a) \end{pmatrix}$$

Пример $\varphi_{i,j}(s, a)$

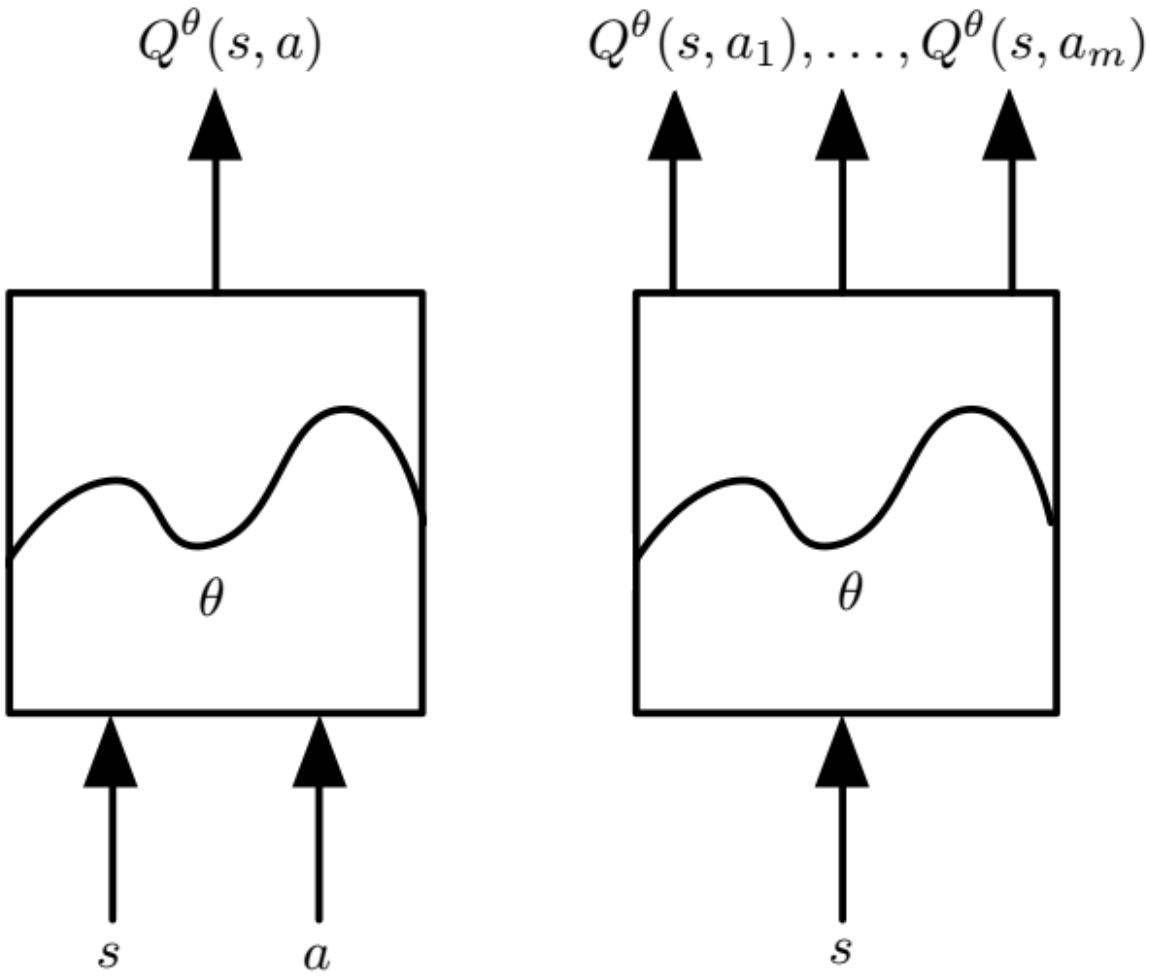
i \ j				
	0	0	0	0
	0	0	0	0
	0	1	0	0
	0	0	0	0

Пример $\varphi_{i,j}(s, a)$

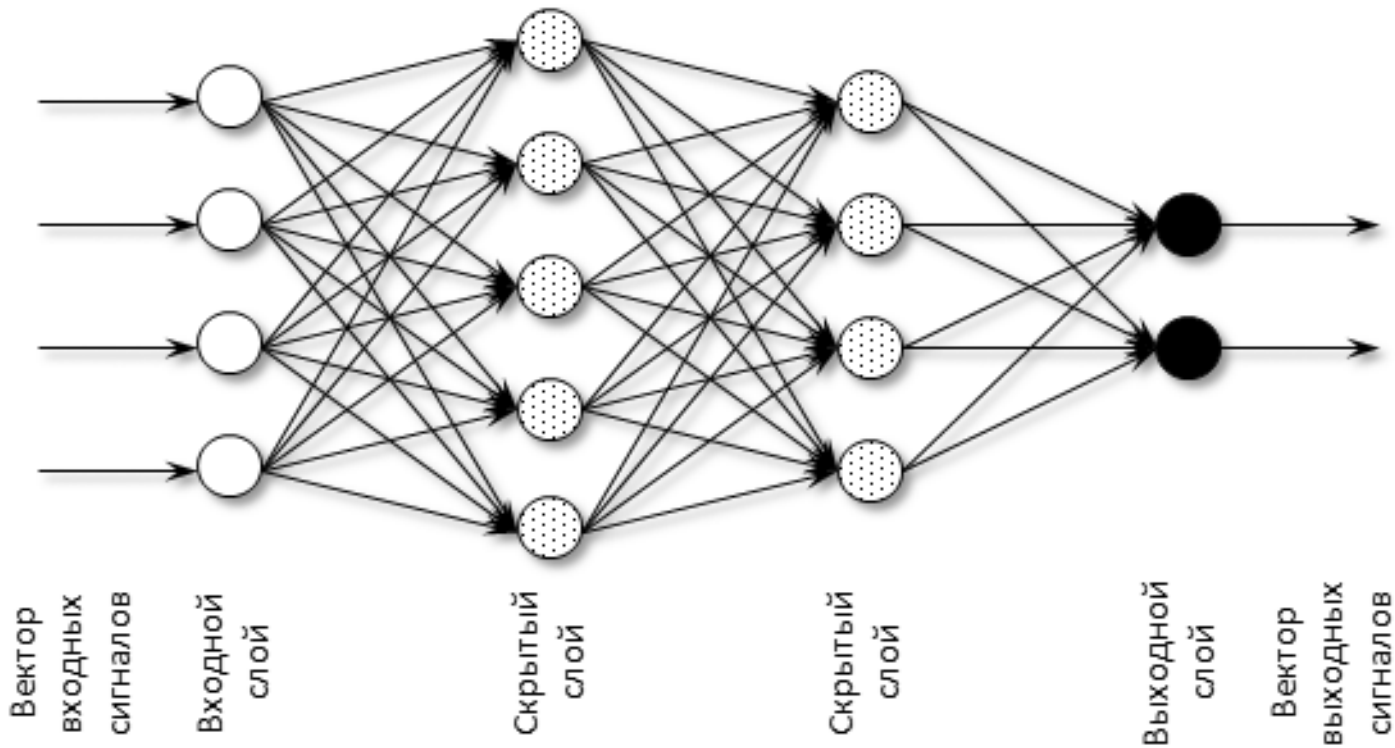
i \ j				
	0	0	0	0
	0	0	0	0
	0	1	0	0
	0	0	0	0



Нейронная сеть



Нейронная есть



$$F_j^\theta(X) = f_{out}\left(b_j + \sum_{k=1}^{4} w_{j,k} f\left(\hat{b}_k + \sum_{l=1}^{5} \hat{w}_{k,l} f\left(\tilde{b}_l + \sum_{i=1}^{4} \tilde{w}_{l,i} x_i\right)\right)\right), \quad j \in \overline{1,2}.$$

$$F^\theta(X) \in \mathbb{R}^2, \quad X \in \mathbb{R}^4, \quad \theta \in \mathbb{R}^{59}$$

Monte-Carlo Update

Определение q_π

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t \mid S_t = s,\ A_t = a]$$

Monte-Carlo Update

Определение q_π

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t \mid S_t = s,\ A_t = a]$$

$$\Downarrow$$

Monte-Carlo Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t) + 1}\big(G_t - Q(S_t, A_t)\big),$$

$$N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$$

Monte-Carlo Update

Определение q_π

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t \mid S_t = s,\ A_t = a]$$

$$\Downarrow$$

Monte-Carlo Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t) + 1}\big(G_t - Q(S_t, A_t)\big),$$

$$N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$$

$$\Downarrow$$

Monte-Carlo Update for Q^θ

$$Loss(\theta) = \big(G_t - Q^\theta(S_t, A_t)\big)^2$$

Monte-Carlo Update

Определение q_π

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t \mid S_t = s,\ A_t = a]$$

$$\Downarrow$$

Monte-Carlo Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t) + 1}\big(G_t - Q(S_t, A_t)\big),$$

$$N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$$

$$\Downarrow$$

Monte-Carlo Update for Q^θ

$$Loss(\theta) = \big(G_t - Q^\theta(S_t, A_t)\big)^2$$

$$\nabla_\theta Loss(\theta) = -\big(G_t - Q^\theta(S_t, A_t)\big)\nabla_\theta Q^\theta(S_t, A_t)$$

Monte-Carlo Update

Определение q_π

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t \mid S_t = s,\ A_t = a]$$

$$\Downarrow$$

Monte-Carlo Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t) + 1}\big(G_t - Q(S_t, A_t)\big),$$

$$N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$$

$$\Downarrow$$

Monte-Carlo Update for Q^θ

$$Loss(\theta) = \big(G_t - Q^\theta(S_t, A_t)\big)^2$$

$$\nabla_\theta Loss(\theta) = -\big(G_t - Q^\theta(S_t, A_t)\big)\nabla_\theta Q^\theta(S_t, A_t)$$

$$\theta \leftarrow \theta - \alpha\nabla_\theta Loss(\theta)$$

SARSA Update

Bellman Expectation Equation для q_π

$$q_\pi(s, a) = \mathbb{E}_\pi[R_t + \gamma q_\pi(S_{t+1}, A_{t+1}) \,|\, S_t = s,\, A_t = a]$$

SARSA Update

Bellman Expectation Equation для q_π

$$q_\pi(s,a) = \mathbb{E}_\pi[R_t + \gamma q_\pi(S_{t+1}, A_{t+1}) \,|\, S_t = s,\ A_t = a]$$

$$\Downarrow$$

SARSA Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\big(R_t + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\big)$$

SARSA Update

Bellman Expectation Equation для q_π

$$q_\pi(s,a) = \mathbb{E}_\pi[R_t + \gamma q_\pi(S_{t+1}, A_{t+1}) \,|\, S_t = s,\ A_t = a]$$

$$\Downarrow$$

SARSA Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\big(R_t + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\big)$$

$$\Downarrow$$

SARSA Update for Q^θ

$$Loss(\theta) = \big(R_t + \gamma Q^\theta(S_{t+1}, A_{t+1}) - Q^\theta(S_t, A_t)\big)^2$$

SARSA Update

Bellman Expectation Equation для q_π

$$q_\pi(s,a) = \mathbb{E}_\pi[R_t + \gamma q_\pi(S_{t+1}, A_{t+1}) \,|\, S_t = s,\ A_t = a]$$

$$\Downarrow$$

SARSA Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\big(R_t + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\big)$$

$$\Downarrow$$

SARSA Update for Q^θ

$$Loss(\theta) = \big(R_t + \gamma Q^\theta(S_{t+1}, A_{t+1}) - Q^\theta(S_t, A_t)\big)^2$$

$$\nabla_\theta Loss(\theta) \approx -\big(R_t + \gamma Q^\theta(S_{t+1}, A_{t+1}) - Q^\theta(S_t, A_t)\big)\nabla_\theta Q^\theta(S_t, A_t)$$

SARSA Update

Bellman Expectation Equation для q_π

$$q_\pi(s,a) = \mathbb{E}_\pi[R_t + \gamma q_\pi(S_{t+1}, A_{t+1}) \,|\, S_t = s,\ A_t = a]$$

$$\Downarrow$$

SARSA Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\big(R_t + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\big)$$

$$\Downarrow$$

SARSA Update for Q^θ

$$Loss(\theta) = \big(R_t + \gamma Q^\theta(S_{t+1}, A_{t+1}) - Q^\theta(S_t, A_t)\big)^2$$

$$\nabla_\theta Loss(\theta) \approx -\big(R_t + \gamma Q^\theta(S_{t+1}, A_{t+1}) - Q^\theta(S_t, A_t)\big)\nabla_\theta Q^\theta(S_t, A_t)$$

$$\theta \leftarrow \theta - \alpha\nabla_\theta Loss(\theta)$$

Q-Learning Update

Bellman Optimality Equation для q_*

$$q_*(s,a) = \mathbb{E}[R_t + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s,\ A_t = a]$$

Q-Learning Update

Bellman Optimality Equation для q_*

$$q_*(s,a) = \mathbb{E}[R_t + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s,\ A_t = a]$$

$$\Downarrow$$

Q-Learning Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\big(R_t + \gamma \max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t)\big)$$

Q-Learning Update

Bellman Optimality Equation для q_*

$$q_*(s,a) = \mathbb{E}[R_t + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s,\ A_t = a]$$

$$\Downarrow$$

Q-Learning Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\big(R_t + \gamma \max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t)\big)$$

$$\Downarrow$$

Q-Learning Update for Q^θ

$$Loss(\theta) = \big(R_t + \gamma \max_{a'} Q^\theta(S_{t+1}, a') - Q^\theta(S_t, A_t)\big)^2$$

Q-Learning Update

Bellman Optimality Equation для q_*

$$q_*(s,a) = \mathbb{E}[R_t + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s,\ A_t = a]$$

$$\Downarrow$$

Q-Learning Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\big(R_t + \gamma \max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t)\big)$$

$$\Downarrow$$

Q-Learning Update for Q^θ

$$Loss(\theta) = \big(R_t + \gamma \max_{a'} Q^\theta(S_{t+1}, a') - Q^\theta(S_t, A_t)\big)^2$$

$$\nabla_\theta Loss(\theta) \approx -\big(R_t + \max_{a'} \gamma Q^\theta(S_{t+1}, a') - Q^\theta(S_t, A_t)\big)\nabla_\theta Q^\theta(S_t, A_t)$$

Q-Learning Update

Bellman Optimality Equation для q_*

$$q_*(s,a) = \mathbb{E}[R_t + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s,\ A_t = a]$$

$$\Downarrow$$

Q-Learning Update for Q

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\big(R_t + \gamma \max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t)\big)$$

$$\Downarrow$$

Q-Learning Update for Q^θ

$$Loss(\theta) = \big(R_t + \gamma \max_{a'} Q^\theta(S_{t+1}, a') - Q^\theta(S_t, A_t)\big)^2$$

$$\nabla_\theta Loss(\theta) \approx -\big(R_t + \max_{a'} \gamma Q^\theta(S_{t+1}, a') - Q^\theta(S_t, A_t)\big)\nabla_\theta Q^\theta(S_t, A_t)$$

$$\theta \leftarrow \theta - \alpha \nabla_\theta Loss(\theta)$$

Конечный случай

$$\mathcal{S} = \{0, 1, \ldots, n-1\}, \quad \mathcal{A} = \{0, 1, \ldots, m-1\}$$

Конечный случай

$$\mathcal{S} = \{0, 1, \ldots, n-1\}, \quad \mathcal{A} = \{0, 1, \ldots, m-1\}$$

Положим

$$\varphi_{i,j}(s, a) = \begin{cases} 1, & \text{если } s = i,\ a = j \\ 0, & \text{иначе} \end{cases}$$

Конечный случай

$$\mathcal{S} = \{0, 1, \ldots, n-1\}, \quad \mathcal{A} = \{0, 1, \ldots, m-1\}$$

Положим

$$\varphi_{i,j}(s,a) = \begin{cases} 1, & \text{если } s = i,\ a = j \\ 0, & \text{иначе} \end{cases}$$

$$Q^{\theta}(s,a) = \theta_{s,a}$$

Конечный случай

$$\mathcal{S} = \{0, 1, \ldots, n-1\}, \quad \mathcal{A} = \{0, 1, \ldots, m-1\}$$

Положим

$$\varphi_{i,j}(s,a) = \begin{cases} 1, & \text{если } s = i,\ a = j \\ 0, & \text{иначе} \end{cases}$$

$$Q^{\theta}(s,a) = \theta_{s,a}$$

$$\theta \leftarrow \theta - \alpha \nabla_{\theta} Loss(\theta)$$

Конечный случай

$$\mathcal{S} = \{0, 1, \ldots, n-1\}, \quad \mathcal{A} = \{0, 1, \ldots, m-1\}$$

Положим

$$\varphi_{i,j}(s,a) = \begin{cases} 1, & \text{если } s = i,\ a = j \\ 0, & \text{иначе} \end{cases}$$

$$Q^\theta(s,a) = \theta_{s,a}$$

$$\theta \leftarrow \theta - \alpha \nabla_\theta Loss(\theta)$$

$$\Downarrow$$

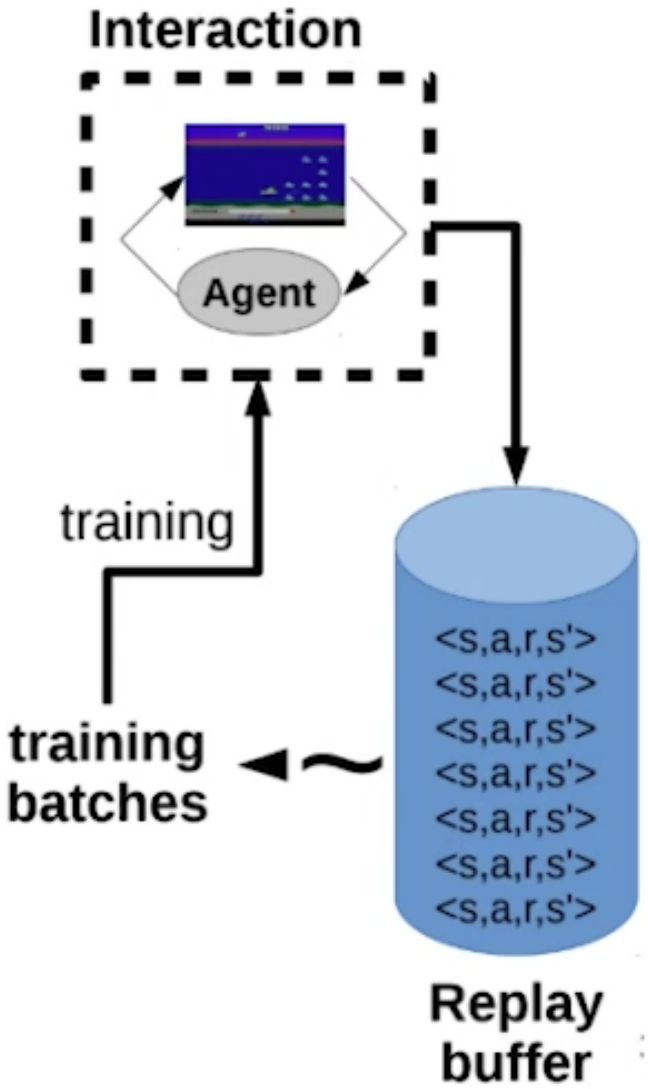
$$\begin{pmatrix} \theta_{0,0} \\ \vdots \\ \theta_{s,a} \\ \vdots \end{pmatrix} = \begin{pmatrix} \theta_{0,0} \\ \vdots \\ \theta_{s,a} \\ \vdots \end{pmatrix} - \alpha \begin{pmatrix} 0 \\ \vdots \\ -\left(R_t + \gamma Q^\theta(S_{t+1}, A_{t+1}) - Q^\theta(S_t, A_t)\right) \\ \vdots \end{pmatrix}$$

Сходимость

Algorithm	Table Lookup	Linear	Non-Linear
Monte-Carlo Control	✓	(✓)	✗
Sarsa	✓	(✓)	✗
Q-learning	✓	✗	✗

(✓) = chatters around near-optimal value function

Experience Replay



- Сохраняем четверку: $(S_t, A_t, R_t, S_{t+1}) \rightarrow Memory$
- Учимся на батче четверок: $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^{n} \leftarrow Memory$

DQN Algorithm

Задаем структуру аппроксимации Q^θ, начальные вектор параметров θ, вероятность исследования среды $\varepsilon = 1$.
Для каждого эпизода k делаем:
Пока эпизод не закончен делаем:

DQN Algorithm

Задаем структуру аппроксимации Q^θ, начальные вектор параметров θ, вероятность исследования среды $\varepsilon = 1$.
Для каждого эпизода k делаем:

Пока эпизод не закончен делаем:

- Находясь в состоянии S_t совершаем действие $A_t \sim \pi(\cdot|S_t)$, где $\pi = \varepsilon$-greedy(Q), получаем награду R_t переходим в состояние S_{t+1}. Сохраняем $(S_t, A_t, R_t, S_{t+1}) \to Memory$

DQN Algorithm

Задаем структуру аппроксимации Q^θ, начальные вектор параметров θ, вероятность исследования среды $\varepsilon = 1$.
Для каждого эпизода k делаем:

Пока эпизод не закончен делаем:

- Находясь в состоянии S_t совершаем действие $A_t \sim \pi(\cdot|S_t)$, где $\pi = \varepsilon$-greedy(Q), получаем награду R_t переходим в состояние S_{t+1}. Сохраняем $(S_t, A_t, R_t, S_{t+1}) \to Memory$
- Берем $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^n \leftarrow Memory$, определяем целевые значения

$$y_i = \begin{cases} r_i, & \text{если } s'_i \text{ -терминальное,} \\ r_i + \gamma \max\limits_{a'} Q^\theta(s'_i, a'), & \text{иначе} \end{cases}$$

функцию потерь $Loss(\theta) = \frac{1}{n} \sum\limits_{i=1}^{n} \left(y_i - Q^\theta(s_i, a_i)\right)^2$ и обновляем вектор параметров

$$\theta \leftarrow \theta - \alpha \nabla_\theta Loss(\theta)$$

DQN Algorithm

Задаем структуру аппроксимации Q^θ, начальные вектор параметров θ, вероятность исследования среды $\varepsilon = 1$.
Для каждого эпизода k делаем:

Пока эпизод не закончен делаем:

- Находясь в состоянии S_t совершаем действие $A_t \sim \pi(\cdot|S_t)$, где $\pi = \varepsilon$-greedy(Q), получаем награду R_t переходим в состояние S_{t+1}. Сохраняем $(S_t, A_t, R_t, S_{t+1}) \to Memory$
- Берем $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^n \leftarrow Memory$, определяем целевые значения

$$y_i = \begin{cases} r_i, & \text{если } s'_i \text{ -терминальное,} \\ r_i + \gamma \max\limits_{a'} Q^\theta(s'_i, a'), & \text{иначе} \end{cases}$$

функцию потерь $Loss(\theta) = \frac{1}{n} \sum\limits_{i=1}^{n} \left(y_i - Q^\theta(s_i, a_i)\right)^2$ и обновляем вектор параметров

$$\theta \leftarrow \theta - \alpha \nabla_\theta Loss(\theta)$$

- Уменьшаем ε

Пример: Atari Games

- Состояния: пиксели с экрана
- Действия: $\rightarrow$, $\leftarrow$, «0»
- Награда: очки в игре

Нейронная сеть для Atari games

- На вход подаются 4 последних предобработанных изображения экрана

Нейронная сеть для Atari games

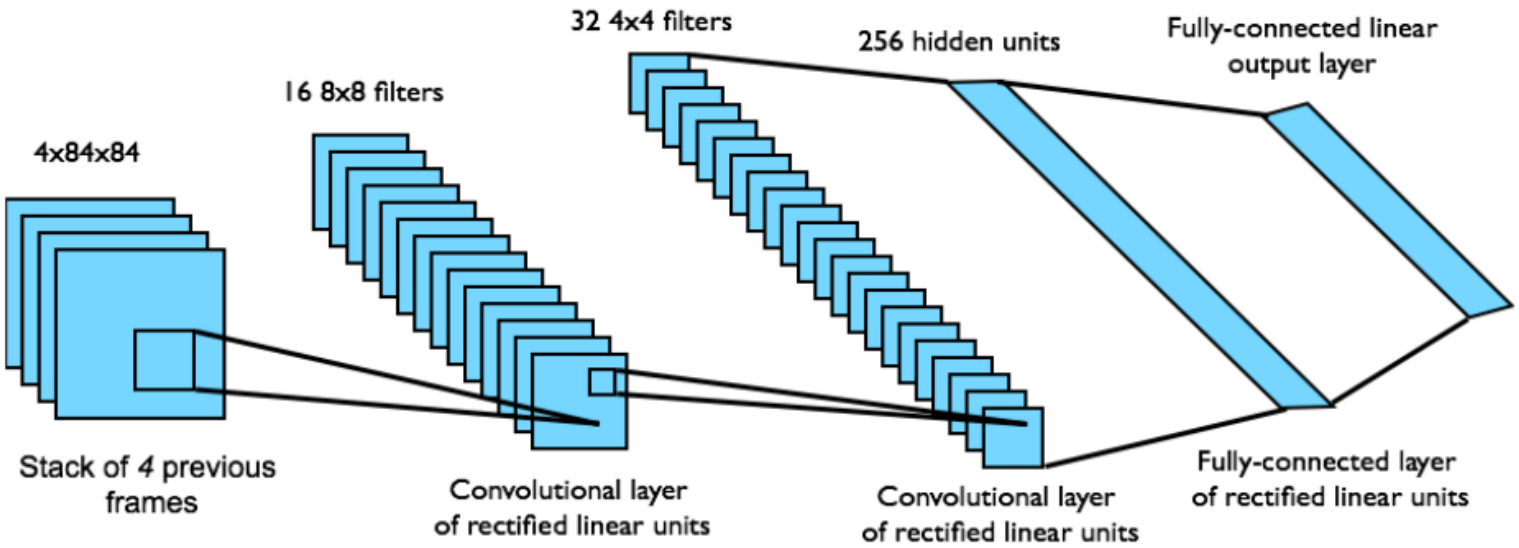
- На вход подаются 4 последних предобработанных изображения экрана
- На выходе $Q^\theta(s, a_1), \ldots, Q^\theta(s, a_{18})$

Нейронная сеть для Atari games

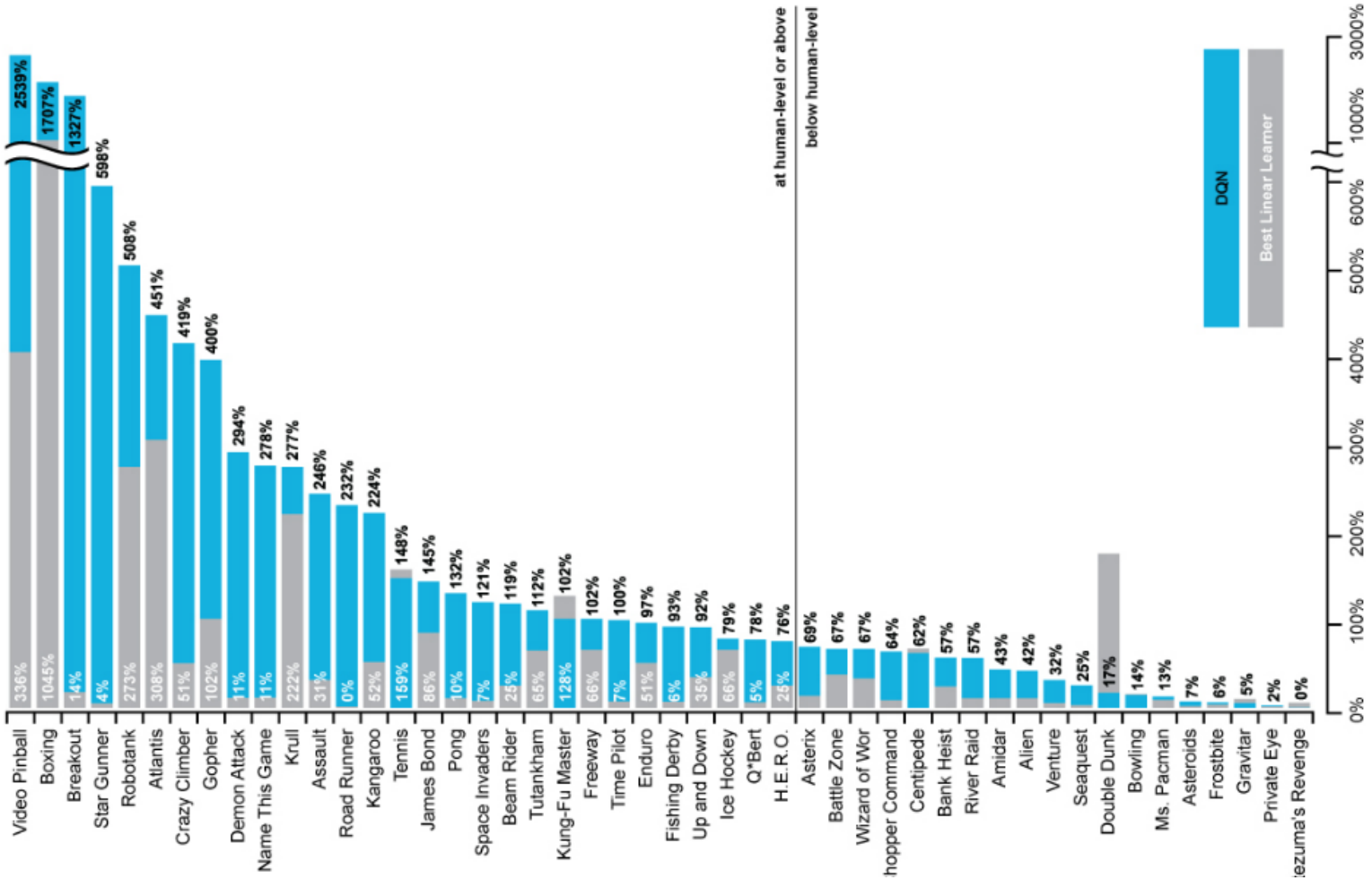
- На вход подаются 4 последних предобработанных изображения экрана
- На выходе $Q^\theta(s, a_1), \ldots, Q^\theta(s, a_{18})$
- Структура сети и гиперпараметры не меняются для всех игр

Нейронная сеть для Atari games

- На вход подаются 4 последних предобработанных изображения экрана
- На выходе $Q^\theta(s, a_1), \ldots, Q^\theta(s, a_{18})$
- Структура сети и гиперпараметры не меняются для всех игр



Результаты в Atari games



Mnih V., at el. Playing Atari with Deep Reinforcement Learning. 2013.

Можно ли использовать Experience Replay для MC и SARSA?

Можно ли использовать Experience Replay для MC и SARSA?

НЕТ

Можно ли использовать Experience Replay для MC и SARSA?

НЕТ

Потому что тройки (S_t, A_t, R_t) и пятерки $(S_t, A_t, R_t, S_{t+1}, A_{t+1})$ зависят от Policy, а она все время меняется

Autocorrelation

Q-Learning Update for Q^θ

- $y = r + \gamma \max_{a'} Q^\theta(s', a')$
- $Loss(\theta) = \left(y - Q^\theta(s, a)\right)^2$
- $\theta \leftarrow \theta - \alpha \nabla_\theta Loss(\theta)$

Autocorrelation

Q-Learning Update for Q^θ

- $y = r + \gamma \max\limits_{a'} Q^\theta(s', a')$
- $Loss(\theta) = \big(y - Q^\theta(s, a)\big)^2$
- $\theta \leftarrow \theta - \alpha \nabla_\theta Loss(\theta)$

Проблема

Если Reward в двух близких состояниях сильно отличается, то при Q-Learning Update возможно $Q^\theta(s, a) \to \infty$

Hard Target Networks $Q^{\theta'}(s, a)$

- Определяем $\theta = \theta'$
- Делаем достаточно много итераций:
  - $y = r + \gamma \max\limits_{a'} Q^{\theta'}(s', a')$
  - $Loss(\theta) = \big(y - Q^{\theta}(s, a)\big)^2$
  - $\theta \leftarrow \theta - \alpha \nabla_{\theta} Loss(\theta)$
- Полагаем $\theta' = \theta$

Soft Target Networks $Q^{\theta'}(s, a)$

- $y = r + \gamma \max_{a'} Q^{\theta'}(s', a')$
- $Loss(\theta) = \big(y - Q^{\theta}(s, a)\big)^2$
- $\theta \leftarrow \theta - \alpha \nabla_{\theta} Loss(\theta)$
- $\theta' \leftarrow \tau\theta + (1 - \tau)\theta'$

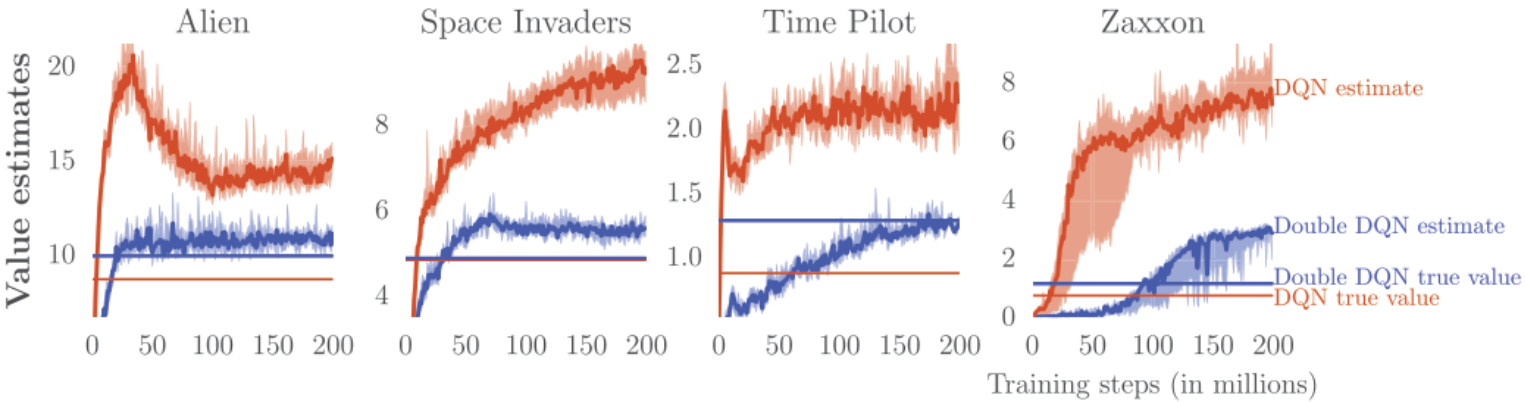
Результаты Experience Replay и Target Network

	Replay Fixed-Q	Replay Q-learning	No replay Fixed-Q	No replay Q-learning
Breakout	316.81	240.73	10.16	3.17
Enduro	1006.3	831.25	141.89	29.1
River Raid	7446.62	4102.81	2867.66	1453.02
Seaquest	2894.4	822.55	1003	275.81
Space Invaders	1088.94	826.33	373.22	301.99

Double DQN

- $y = r + \gamma Q^{\theta}\left(s, \text{argmax}_{a'} Q^{\theta'}(s', a')\right)$
- $Loss(\theta) = \left(y - Q^{\theta}(s, a)\right)^2$
- $\theta \leftarrow \theta - \alpha \nabla_{\theta} Loss(\theta)$
- $\theta' \leftarrow \tau\theta + (1 - \tau)\theta'$

Результаты Double DQN



Van Hasselt H., Guez A., Silver D. Deep Reinforcement Learning with Double Q-Learning. 2016.

Markov Decision Process

Markov Property

$$\mathbb{P}[S_{t+1}|S_t, A_t] = \mathbb{P}[S_{t+1}|S_1, A_1, S_2, A_2 \ldots, S_t, A_t]$$

$$\mathbb{P}[R_t|S_t, A_t] = \mathbb{P}[R_t|S_1, A_1, S_2, A_2 \ldots, S_t, A_t] = 1$$

Markov Decision Process $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$

- $\mathcal{S}$ — бесконечное пространство состояний
- $\mathcal{A}$ — бесконечное пространство действий
- $\mathcal{P}$ — неизвестная функция (тензор) вероятностей переходов между состояниями

$$\mathcal{P}(s'|s, a) = \mathbb{P}[S_{t+1} = s'|S_t = s, A_t = a]$$

- $\mathcal{R}$ — неизвестная функция (матрица) вознаграждений

$$\mathcal{R}(s, a) = R_t \quad \Leftrightarrow \quad \mathbb{P}[R_t|S_t = s, A_t = a] = 1$$

- $\gamma \in [0, 1]$ — коэффициент дисконтирования

Пример: Pendulum

- Состояния: $\mathbb{R}^2$
  или пиксели с экрана
- Действия: $[-2, 2]$
- Награда: $\psi^2 + 0.1\dot{\psi}^2 - 0.001a^2$

DQN Algorithm

Задаем структуру аппроксимации Q^θ, начальные вектор параметров θ, вероятность исследования среды $\varepsilon = 1$.
Для каждого эпизода k делаем:

Пока эпизод не закончен делаем:

- Находясь в состоянии S_t совершаем действие $A_t \sim \pi(\cdot|S_t)$, где $\pi = \varepsilon$-greedy(Q), получаем награду R_t переходим в состояние S_{t+1}. Сохраняем $(S_t, A_t, R_t, S_{t+1}) \to Memory$
- Берем $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^n \leftarrow Memory$, определяем целевые значения

$$y_i = \begin{cases} r_i, & \text{если } s'_i \text{ -терминальное,} \\ r_i + \gamma \max\limits_{a'} Q^\theta(s'_i, a'), & \text{иначе} \end{cases}$$

функцию потерь $Loss(\theta) = \frac{1}{n} \sum\limits_{i=1}^{n} \left(y_i - Q^\theta(s_i, a_i)\right)^2$ и обновляем вектор параметров

$$\theta \leftarrow \theta - \alpha \nabla_\theta Loss(\theta)$$

- Уменьшаем ε

DQN Algorithm

Задаем структуру аппроксимации Q^θ, начальные вектор параметров θ, вероятность исследования среды $\varepsilon = 1$.
Для каждого эпизода k делаем:

Пока эпизод не закончен делаем:

- Находясь в состоянии S_t совершаем действие $A_t \sim \pi(\cdot|S_t)$, где $\pi = \varepsilon\text{-greedy}(Q)$, получаем награду R_t переходим в состояние S_{t+1}. Сохраняем $(S_t, A_t, R_t, S_{t+1}) \to Memory$
- Берем $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^n \leftarrow Memory$, определяем целевые значения

$$y_i = \begin{cases} r_i, & \text{если } s'_i \text{ -терминальное,} \\ r_i + \gamma \max\limits_{a'} Q^\theta(s'_i, a'), & \text{иначе} \end{cases}$$

функцию потерь $Loss(\theta) = \frac{1}{n} \sum\limits_{i=1}^{n} \left(y_i - Q^\theta(s_i, a_i)\right)^2$ и обновляем вектор параметров

$$\theta \leftarrow \theta - \alpha \nabla_\theta Loss(\theta)$$

- Уменьшаем ε

Continuous DQN

Gu S., Lillicrap T., Sutskever I., Levine S. Continuous Deep Q-Learning with Model-based Acceleration. 2016.

$$Q^{\theta}(s,a) = V^{\theta_V}(s) + A^{\theta_A}(s,a), \quad \theta = (\theta_V, \theta_A)$$

где

Continuous DQN

Gu S., Lillicrap T., Sutskever I., Levine S. Continuous Deep Q-Learning with Model-based Acceleration. 2016.

$$Q^{\theta}(s,a) = V^{\theta_V}(s) + A^{\theta_A}(s,a), \quad \theta = (\theta_V, \theta_A)$$

где

$$A^{\theta_A}(s,a) = -(a - \mu^{\theta_\mu}(s))^T P^{\theta_P}(s)(a - \mu^{\theta_\mu}(s)), \quad \theta_A = (\theta_\mu, \theta_P)$$

где

Continuous DQN

Gu S., Lillicrap T., Sutskever I., Levine S. Continuous Deep Q-Learning with Model-based Acceleration. 2016.

$$Q^{\theta}(s,a) = V^{\theta_V}(s) + A^{\theta_A}(s,a), \quad \theta = (\theta_V, \theta_A)$$

где

$$A^{\theta_A}(s,a) = -(a - \mu^{\theta_\mu}(s))^T P^{\theta_P}(s)(a - \mu^{\theta_\mu}(s)), \quad \theta_A = (\theta_\mu, \theta_P)$$

где

$$P^{\theta_P}(s) = L^{\theta_P}(s) L^{\theta_P}(s)^T$$

Continuous DQN

Gu S., Lillicrap T., Sutskever I., Levine S. Continuous Deep Q-Learning with Model-based Acceleration. 2016.

$$Q^{\theta}(s,a) = V^{\theta_V}(s) + A^{\theta_A}(s,a), \quad \theta = (\theta_V, \theta_A)$$

где

$$A^{\theta_A}(s,a) = -(a - \mu^{\theta_\mu}(s))^T P^{\theta_P}(s)(a - \mu^{\theta_\mu}(s)), \quad \theta_A = (\theta_\mu, \theta_P)$$

где

$$P^{\theta_P}(s) = L^{\theta_P}(s) L^{\theta_P}(s)^T$$

Тогда

- $\max\limits_a Q^{\theta}(s,a) = ???$
- $\text{argmax}_a Q^{\theta}(s,a) = ???$

Continuous DQN

Gu S., Lillicrap T., Sutskever I., Levine S. Continuous Deep Q-Learning with Model-based Acceleration. 2016.

$$Q^{\theta}(s,a) = V^{\theta_V}(s) + A^{\theta_A}(s,a), \quad \theta = (\theta_V, \theta_A)$$

где

$$A^{\theta_A}(s,a) = -(a - \mu^{\theta_\mu}(s))^T P^{\theta_P}(s)(a - \mu^{\theta_\mu}(s)), \quad \theta_A = (\theta_\mu, \theta_P)$$

где

$$P^{\theta_P}(s) = L^{\theta_P}(s) L^{\theta_P}(s)^T$$

- $\max\limits_a Q^{\theta}(s,a) = V^{\theta_V}(s)$
- $\text{argmax}_a Q^{\theta}(s,a) = \mu^{\theta_\mu}(s)$

Организационные вопросы

- Пятница, 17:50, аудитория 622
- Отчетность: домашние работы
- Страничка курса: https://github.com/imm-rl-lab/UrFU_course
- E-mail для связи:

ВОПРОСЫ?