Навчання з підкріпленням

Лекція 5: Методи апроксимації функції цінності

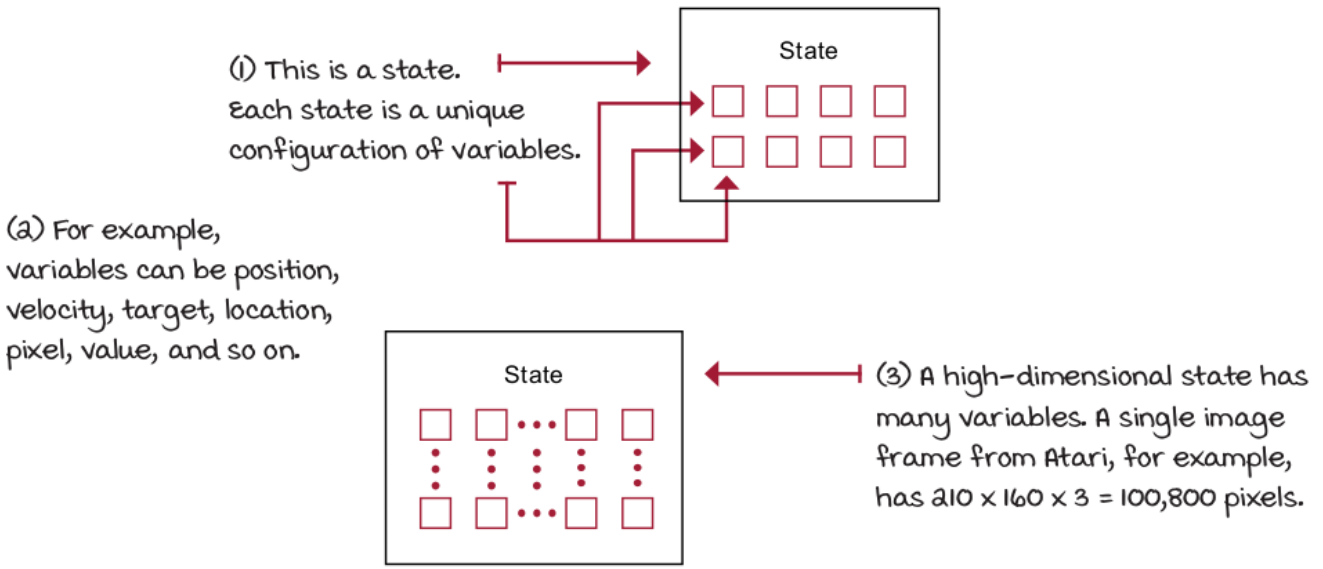
Кочура Юрій Петрович
iuriy.kochura@gmail.com
@y_kochura

Сьогодні

- Вступ
- Інкрементні методи
- Пакетні методи

Вступ

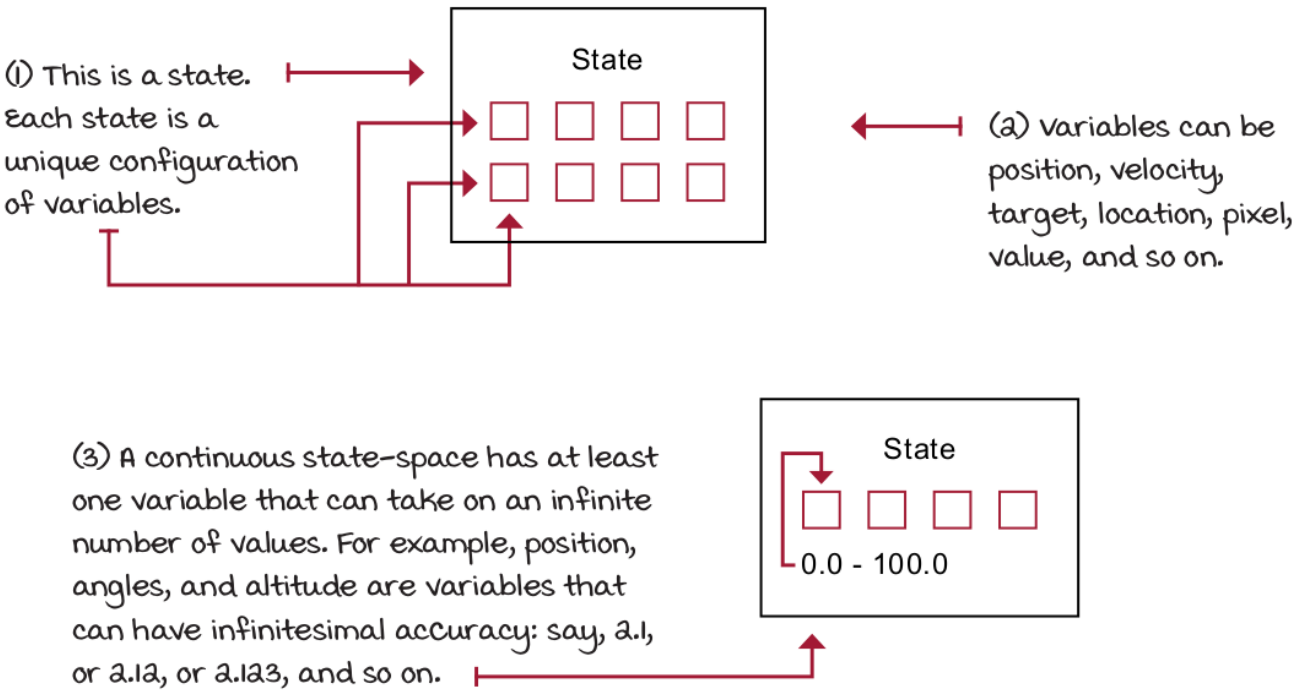
High-dimensional state spaces



—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

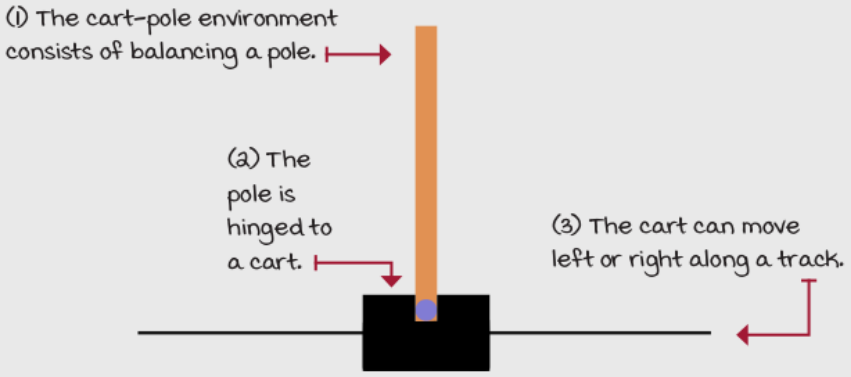
Continuous state spaces



—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

This is the cart-pole environment



Its state space is comprised of four variables:

- The cart position on the track (x-axis) with a range from –2.4 to 2.4
- The cart velocity along the track (x-axis) with a range from –inf to inf
- The pole angle with a range of ~–40 degrees to ~ 40 degrees
- The pole velocity at the tip with a range of –inf to inf

There are two available actions in every state:

- Action 0 applies a –1 force to the cart (push it left)
- Action 1 applies a +1 force to the cart (push it right)

You reach a terminal state if

- The pole angle is more than 12 degrees away from the vertical position
- The cart center is more than 2.4 units from the center of the track
- The episode count reaches 500 time steps (more on this later)

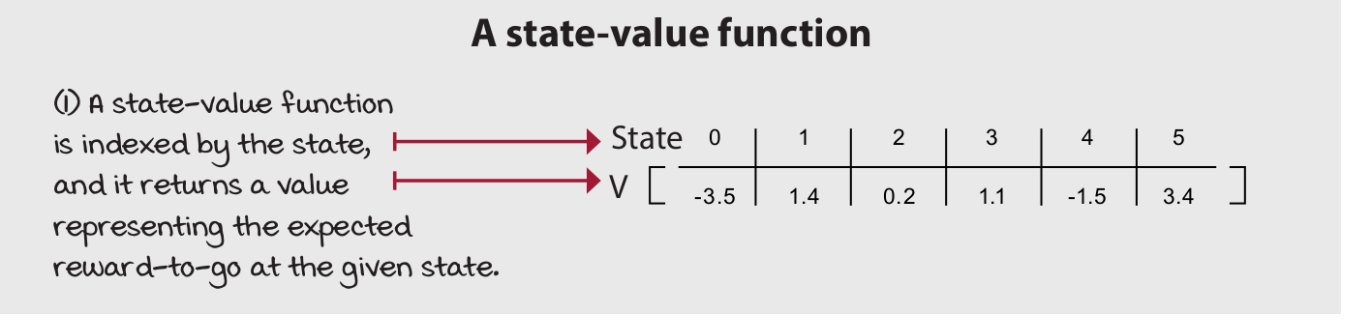
The reward function is

- +1 for every time step

—

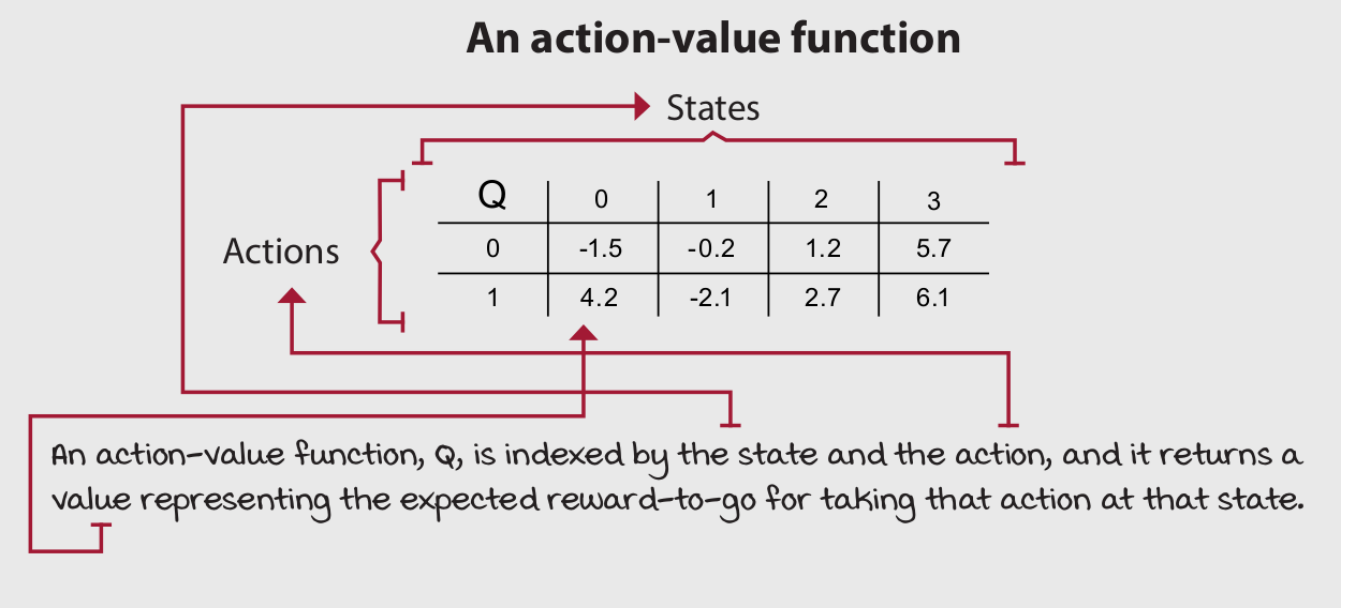
Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Апроксимація функцій має переваги



Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Апроксимація функцій має переваги



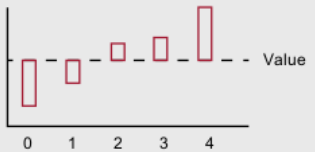
—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

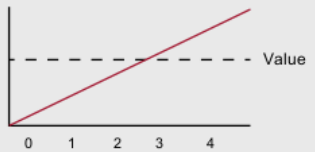
A state-value function with and without function approximation

(1) Imagine this state-value function. $V = [-2.5, -1.1, 0.7, 3.2, 7.6]$

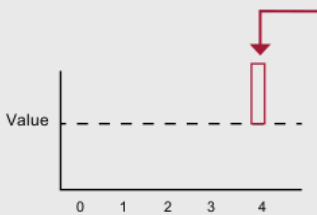
(2) Without function approximation, each value is independent.



(3) With function approximation, the underlying relationship of the states can be learned and exploited.

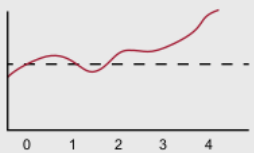


(4) The benefit of using function approximation is particularly obvious if you imagine these plots after even a single update.



(5) Without function approximation, the update only changes one state.

(6) With function approximation, the updates change multiple states.



(7) Of course, this is a simplified example, but it helps illustrate what's happening. What would be different in "real" examples?

First, if we approximate an action-value function, Q, we'd have to add another dimension.

Also, with a non-linear function approximator, such as a neural network, more complex relationships can be discovered.

—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Апроксимація функцій має переваги

Boil it Down

Reasons for using function approximation

Our motivation for using function approximation isn't only to solve problems that aren't solvable otherwise, but also to solve problems more efficiently.

—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

First decision point: Selecting a value function to approximate

- The state-value function $v(s)$
- The action-value function $q(s, a)$
- The action-advantage function $a(s, a)$

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Which Function Approximator?

There are many function approximators, e.g.

- Linear combinations of features
- Neural network
- Decision tree
- Nearest neighbour
- Fourier / wavelet bases
- ...

—

Джерело: David Silver.

Which Function Approximator?

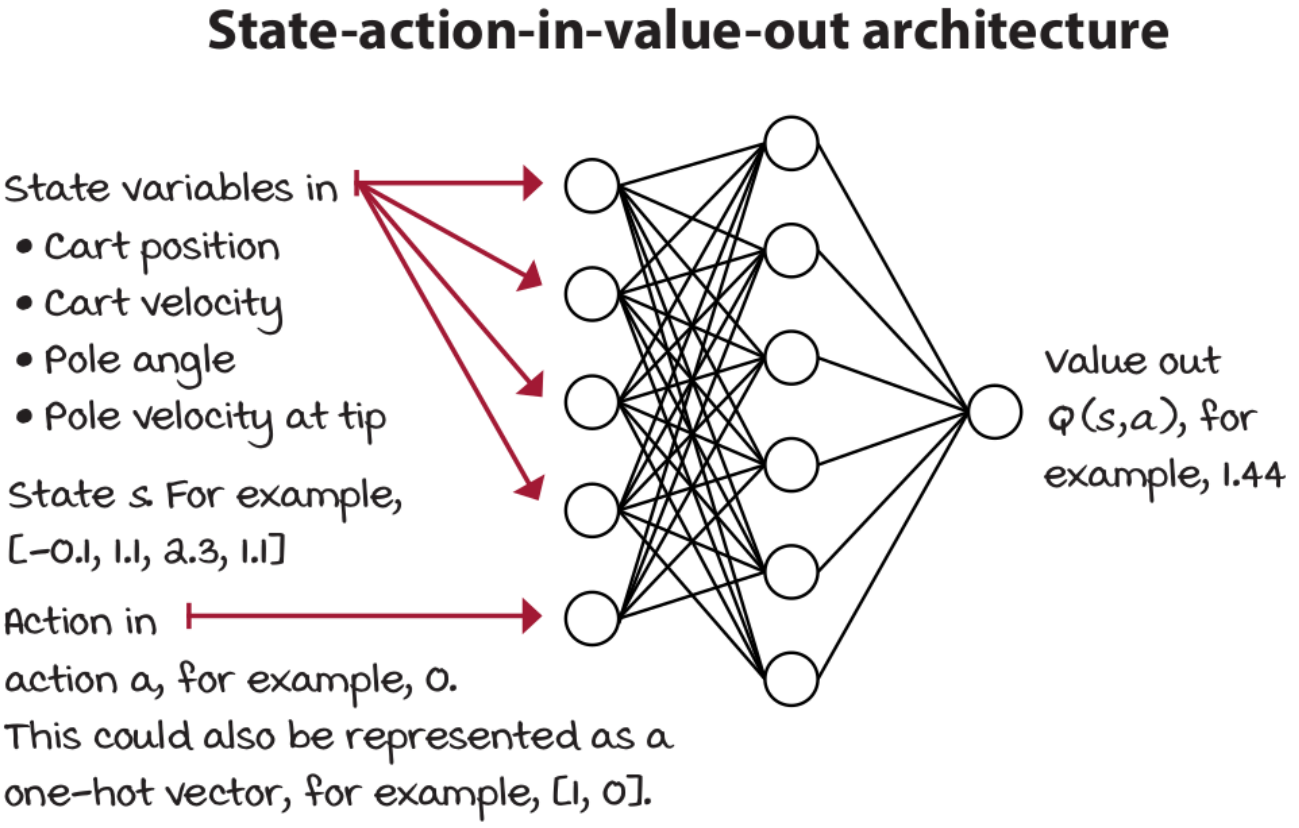
We consider differentiable function approximators, e.g.

- Linear combinations of features
- Neural network
- Decision tree
- Nearest neighbour
- Fourier / wavelet bases
- ...

—

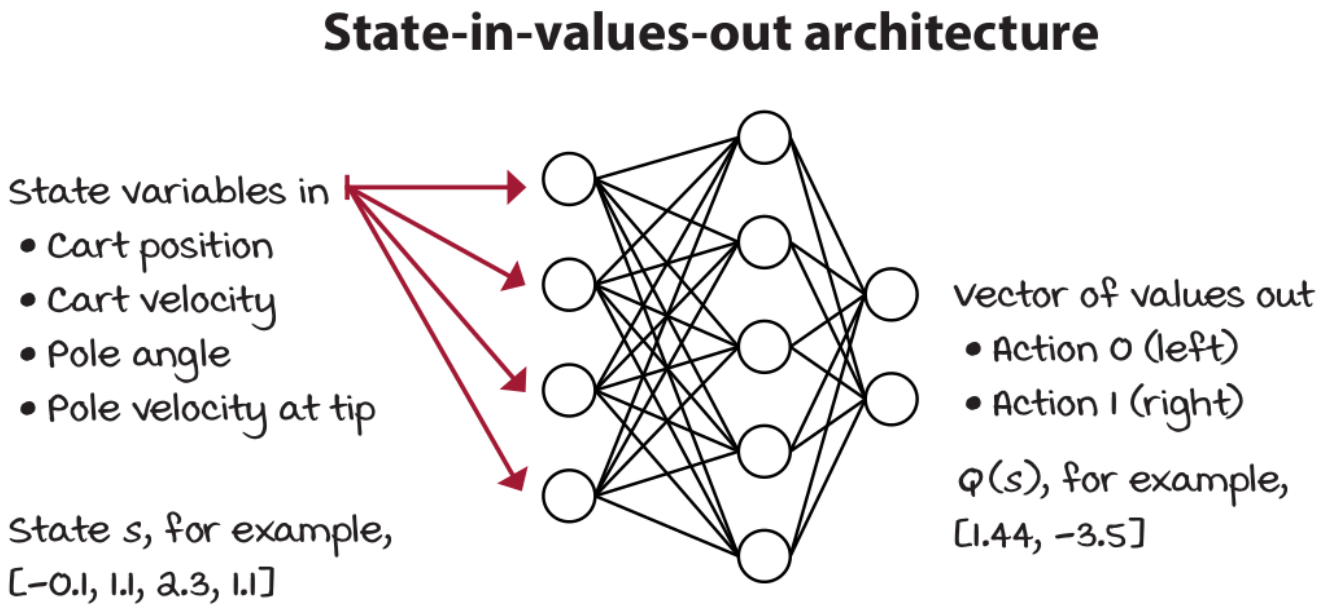
Джерело: David Silver.

Second decision point: Selecting a neural network architecture



Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

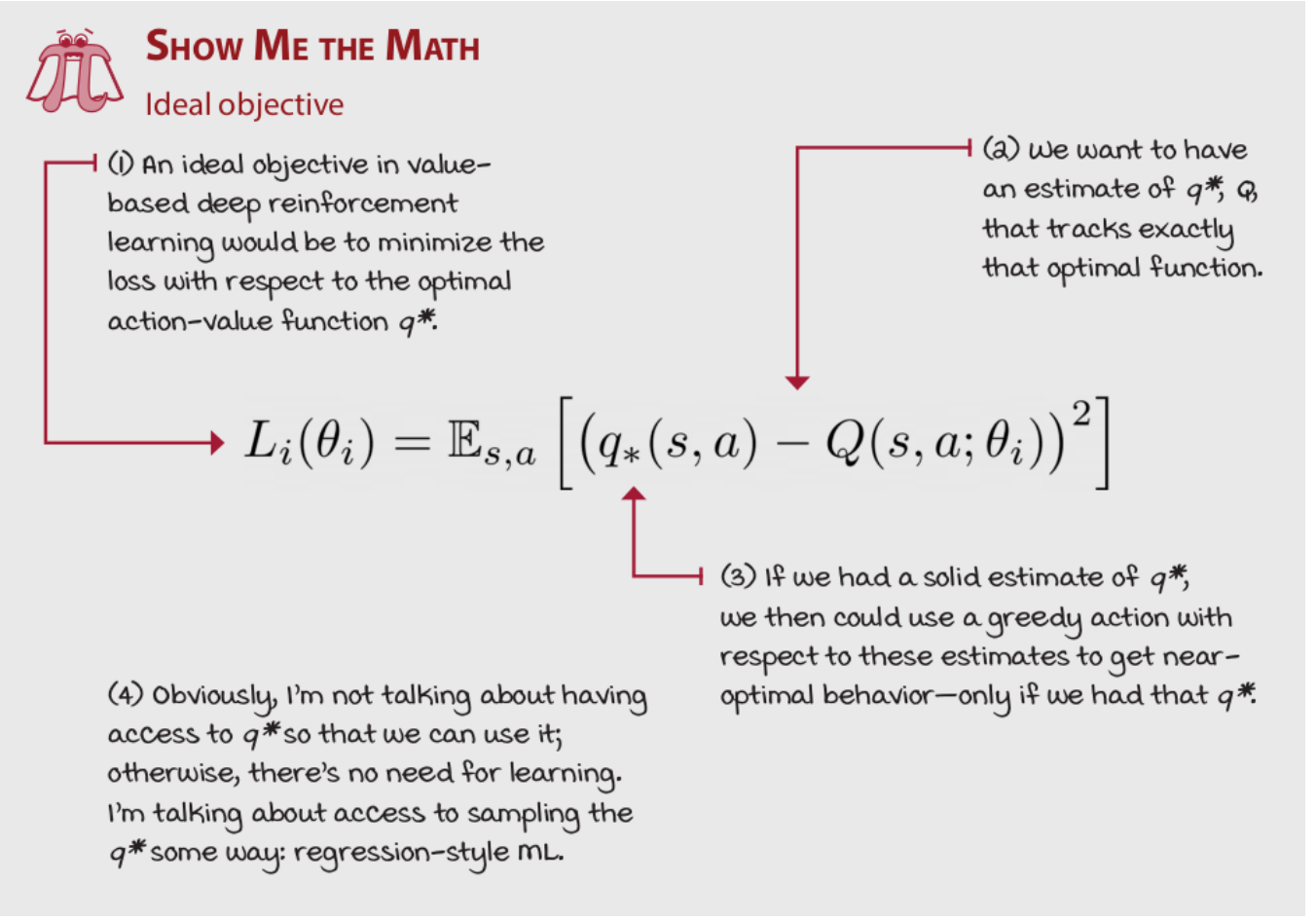
Second decision point: Selecting a neural network architecture



—

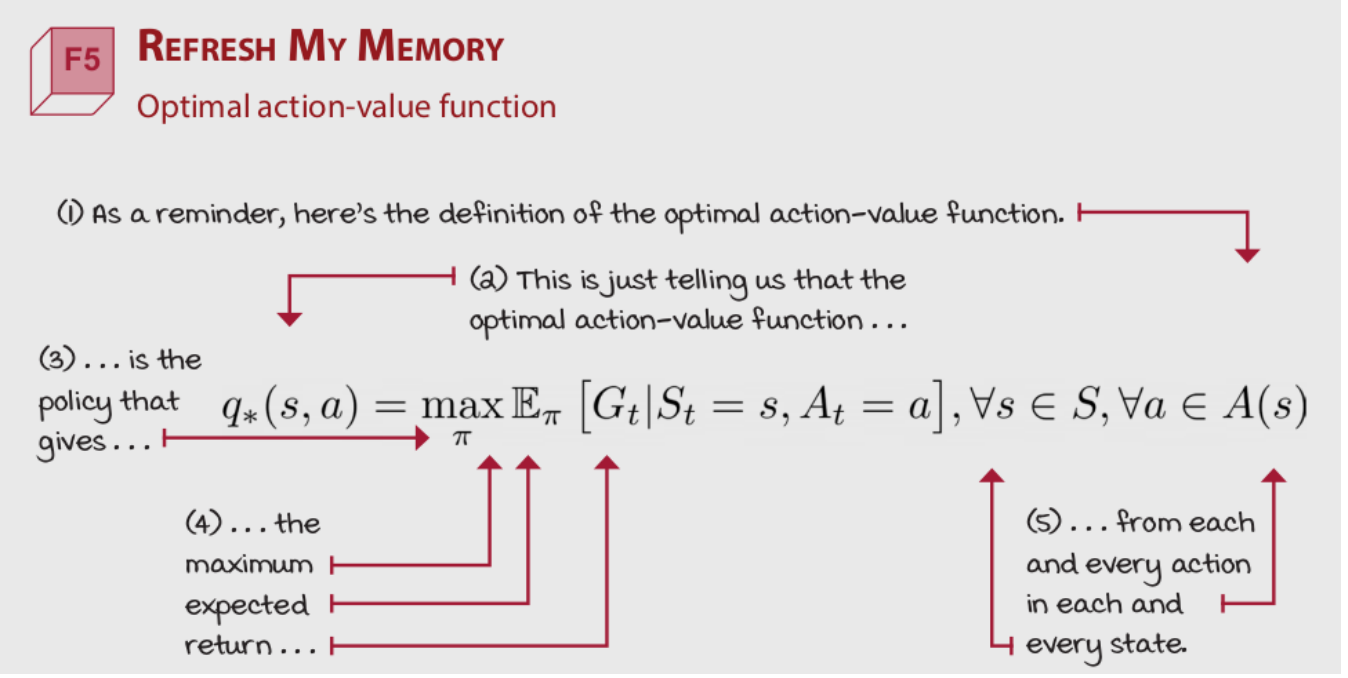
Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Third decision point: Selecting what to optimize



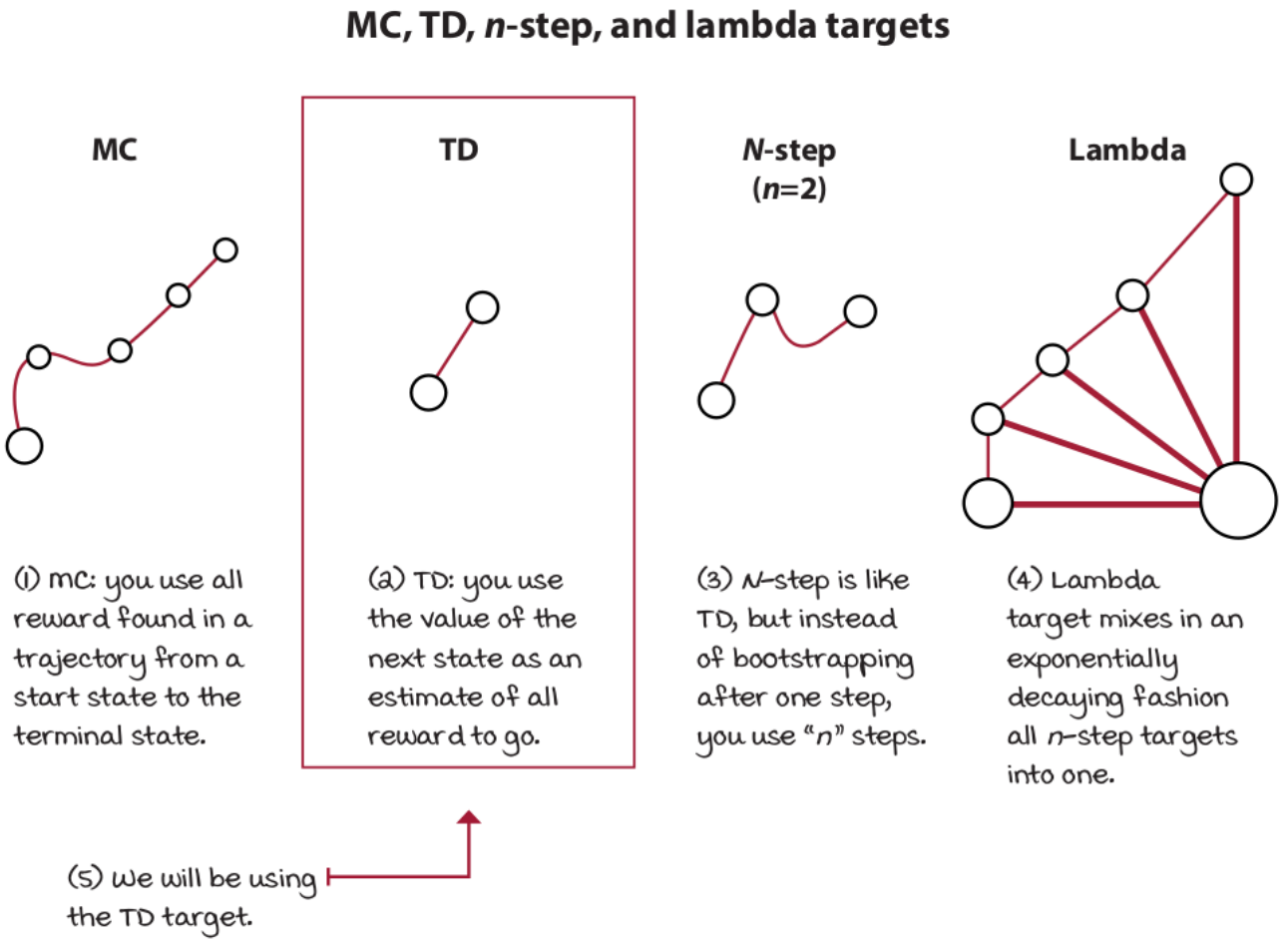
—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.



Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Fourth decision point: Selecting the targets for policy evaluation



—

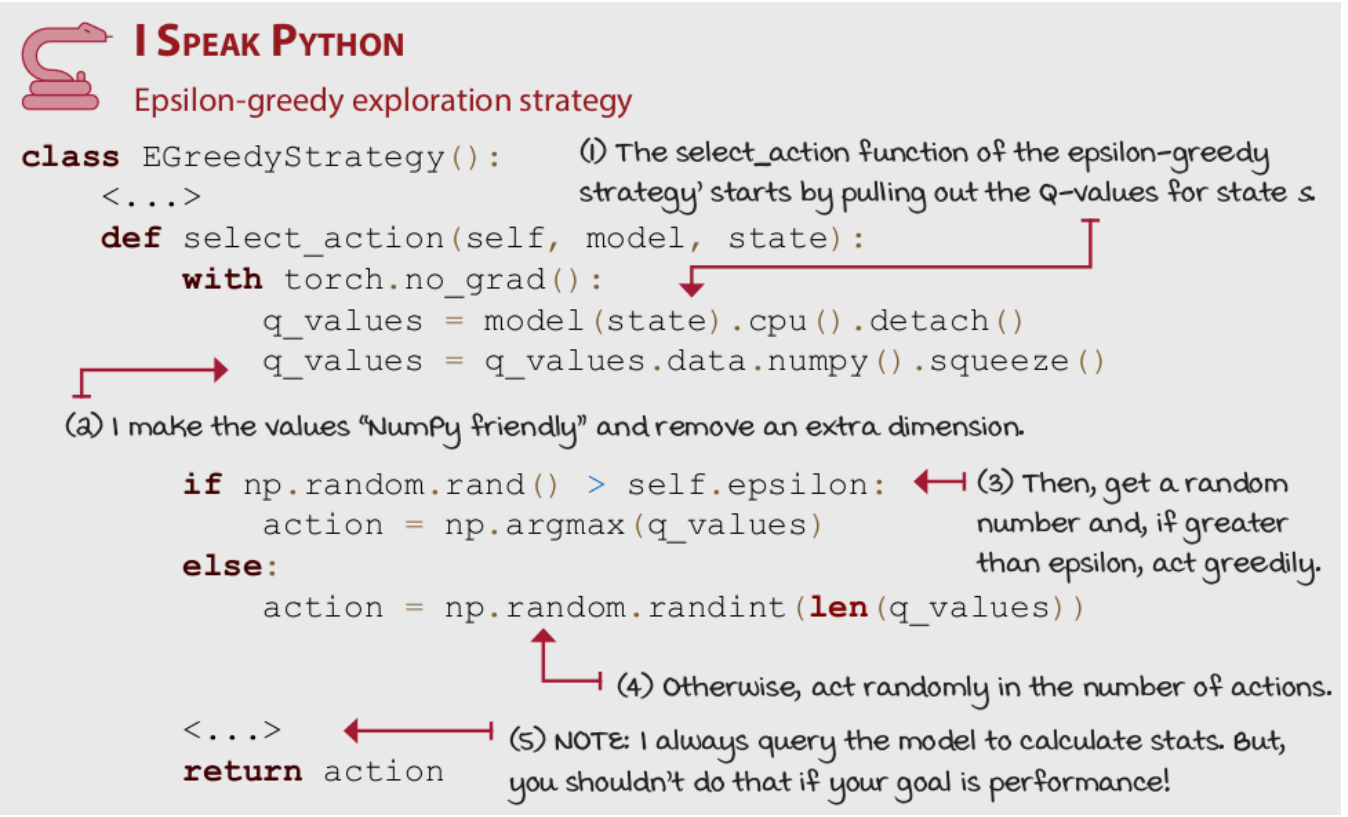
Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Fifth decision point: Selecting an exploration strategy

Another thing we need to decide is which policy improvement step to use for our generalized policy iteration needs.

—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.



Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Sixth decision point: Selecting a loss function

- L1
- L2, MSE
- ...

—
Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Seventh decision point: Selecting an optimization method

- Batch gradient descent
- Mini-batch gradient descent
- Stochastic gradient descent
- Mini-batch gradient descent vs. momentum

—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Batch gradient descent

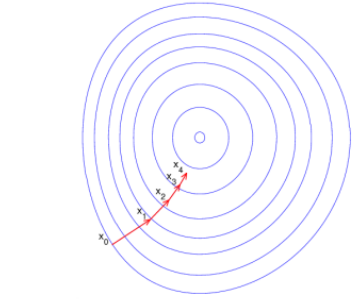
- Let $J(\mathbf{w})$ be a differentiable function of parameter vector $\mathbf{w}$
- Define the *gradient* of $J(\mathbf{w})$ to be

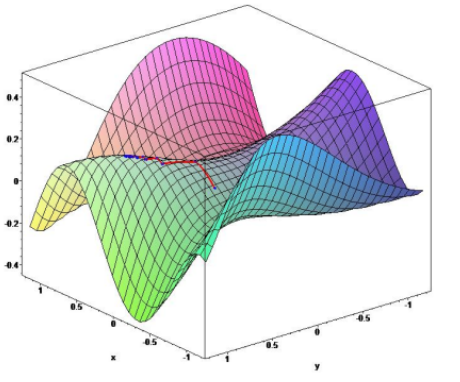
$$\nabla_{\mathbf{w}} J(\mathbf{w}) = \begin{pmatrix} \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}_1} \\ \vdots \\ \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}_n} \end{pmatrix}$$

- To find a local minimum of $J(\mathbf{w})$
- Adjust $\mathbf{w}$ in direction of -ve gradient

$$\Delta \mathbf{w} = -\frac{1}{2} \alpha \nabla_{\mathbf{w}} J(\mathbf{w})$$

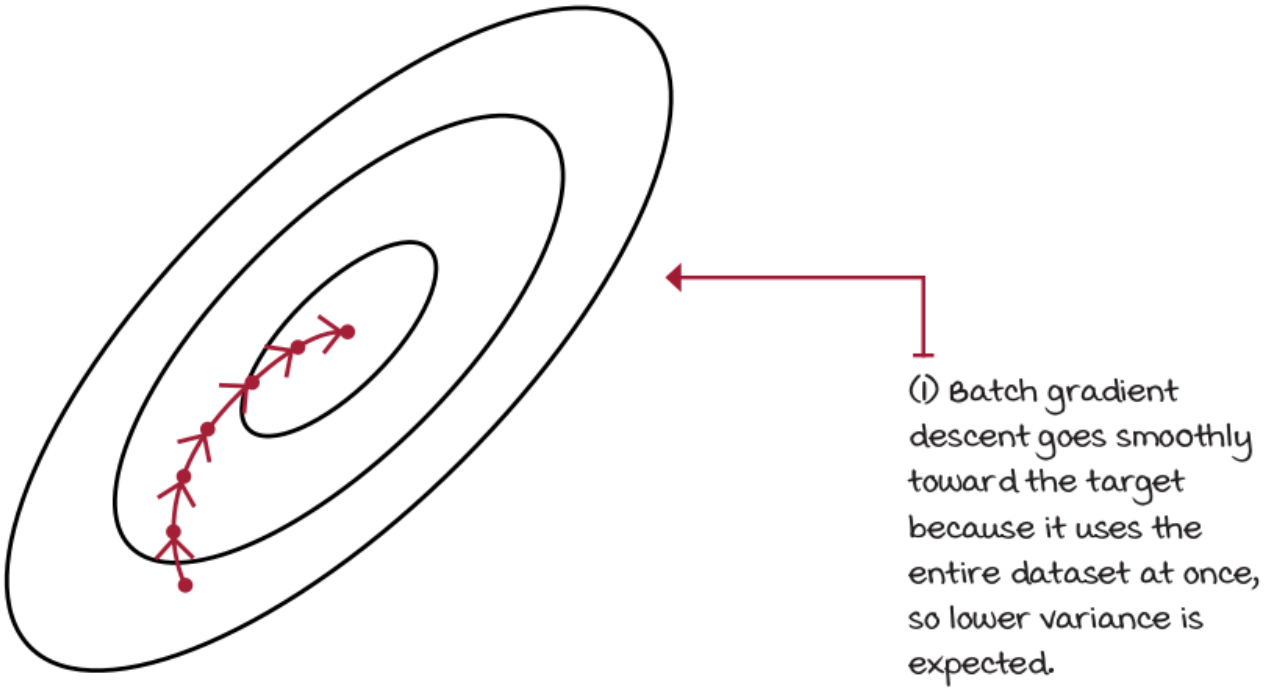
where α is a step-size parameter





Джерело: David Silver.

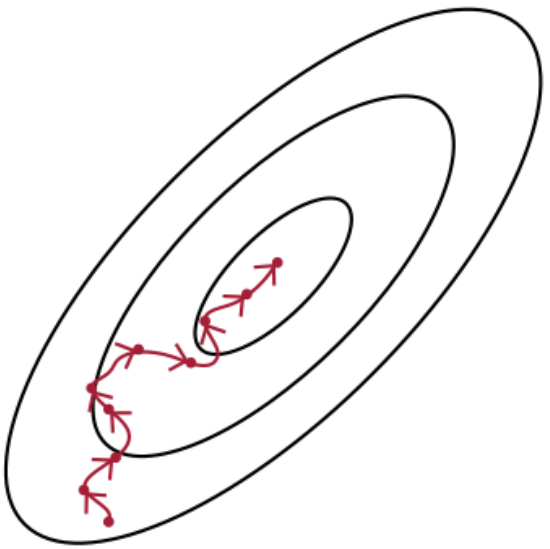
Batch gradient descent



—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Mini-batch gradient descent

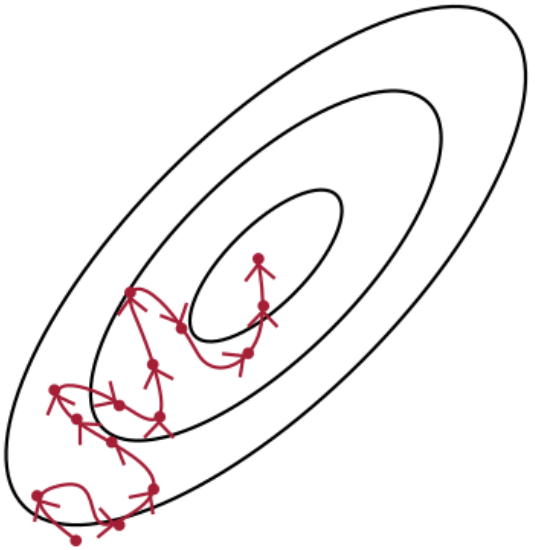


(1) In mini-batch gradient descent we use a uniformly sampled mini-batch. This results in noisier updates, but also faster processing of the data.

—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

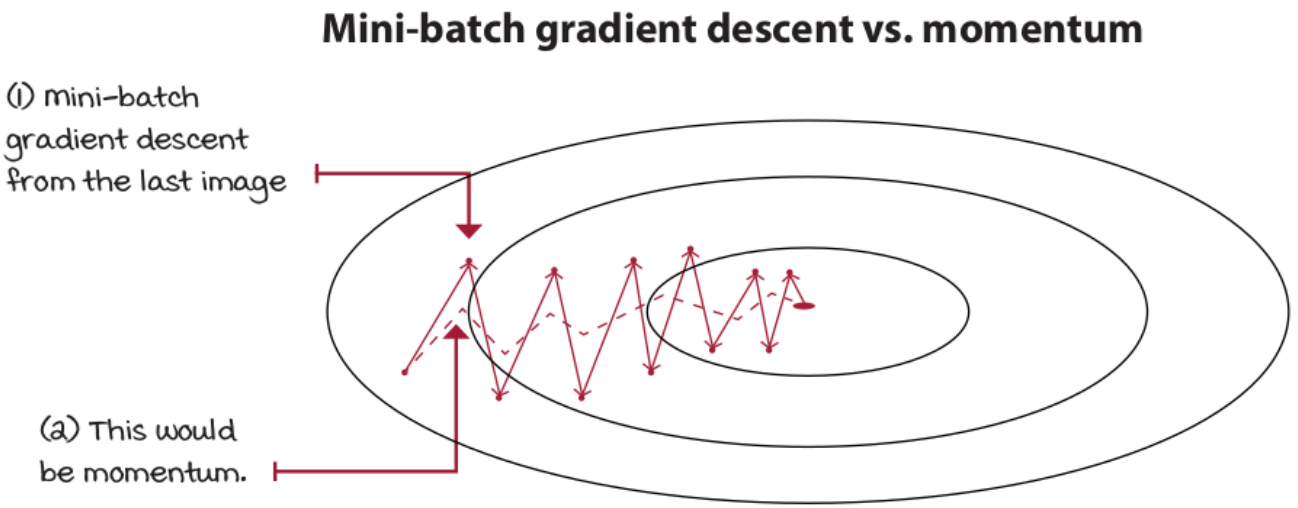
Stochastic gradient descent



(!) With stochastic gradient descent, in every iteration we step through only one sample. This makes it a noisy algorithm. It wouldn't be surprising to see several steps taking us further away from the target, and later back toward the target.

—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.



—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

It's in the Details

The full neural fitted Q-iteration (NFQ) algorithm

Currently, we've made the following selections:

- Approximate the action-value function $Q(s,a;\theta)$.
- Use a state-in-values-out architecture (nodes: 4, 512,128, 2).
- Optimize the action-value function to approximate the optimal action-value function $q^*(s,a)$.
- Use off-policy TD targets ($r + \gamma * max\_a' Q(s',a';\theta)$) to evaluate policies.
- Use an epsilon-greedy strategy (epsilon set to 0.5) to improve policies.
- Use mean squared error (MSE) for our loss function.
- Use RMSprop as our optimizer with a learning rate of 0.0005.

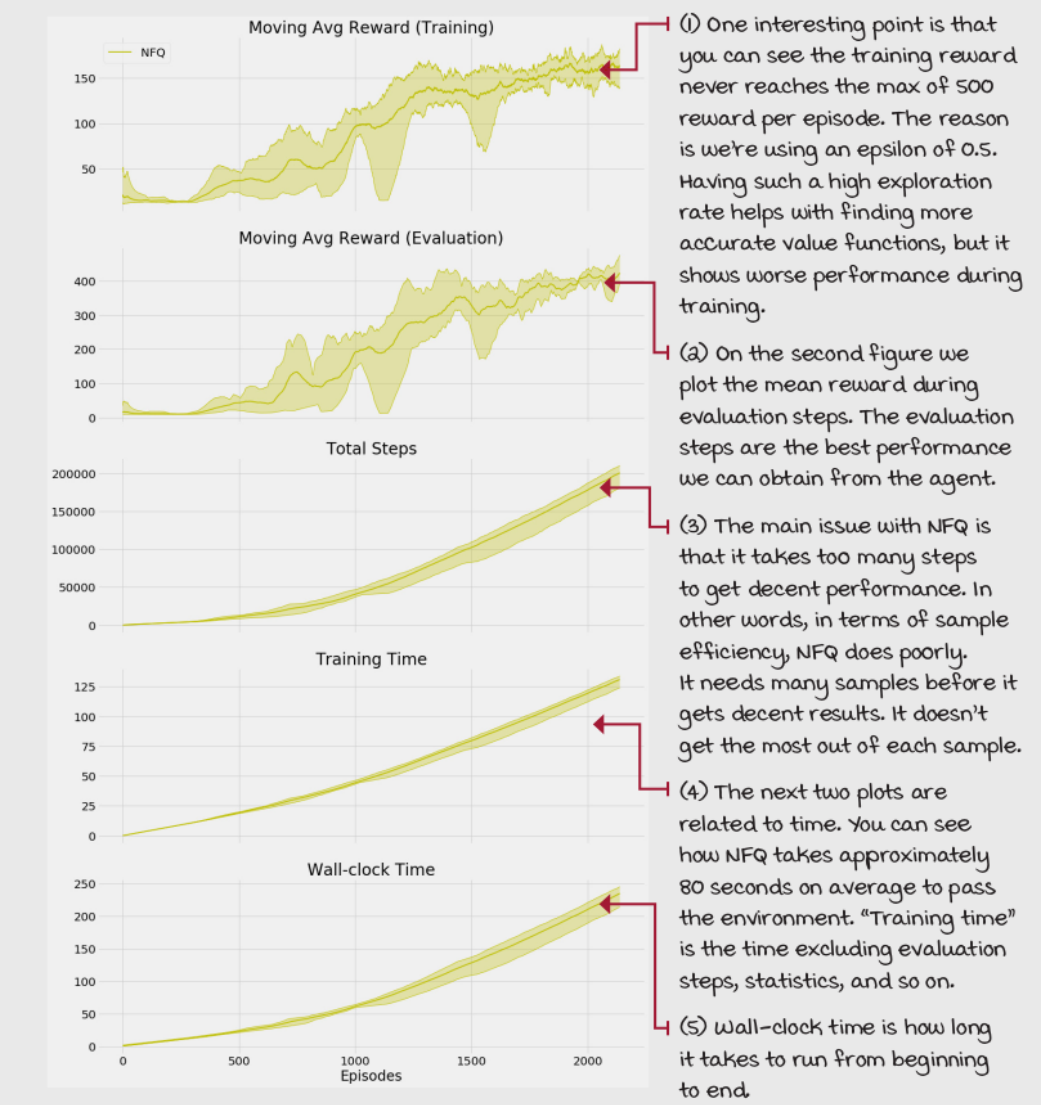
NFQ has three main steps:

1. Collect E experiences: (s, a, r, s', d) tuples. We use 1024 samples.
2. Calculate the off-policy TD targets: $r + \gamma * max\_a' Q(s',a';\theta)$.
3. Fit the action-value function $Q(s,a;\theta)$ using MSE and RMSprop.

This algorithm repeats steps 2 and 3 K number of times before going back to step 1. That's what makes it fitted: the nested loop. We'll use 40 fitting steps K.

—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.



—

Джерело: Grokking Deep Reinforcement Learning, Miguel Morales.

Демо

Література

- David Silver, Lecture 6: Value Function Approximation. [video], [slides]
- Reinforcement Learning: An Introduction - Chapter 9: On-policy Prediction with Approximation
- Reinforcement Learning: An Introduction - Chapter 10: On-policy Control with Approximation
- Tutorial: Introduction to Reinforcement Learning with Function Approximation

Кінець