YOUNG && YANDEX

Value-based подходы в RL; Q-learning

Радослав Нейчев

Выпускник и преподаватель ШАД и МФТИ, руководитель группы ML-разработки Лаборатории ИИ Яндекса, основатель girafe-ai, к.ф.-м.н.

Содержание

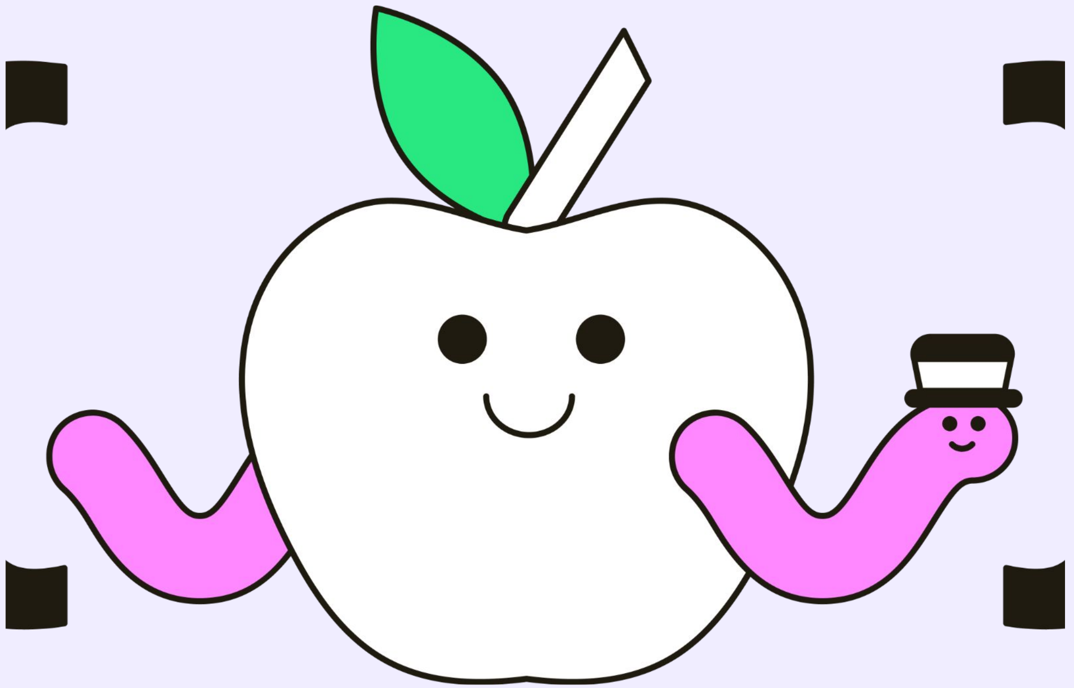
References

These slides are almost the exact copy of Practical RL course week 2 slides by Shvechikov Pavel.
Special thanks to YSDA team for making them publicly available.

Original slides link: https://github.com/yandexdataschool/Practical_RL

Reinforcement Learning problem recap

01



Reinforcement learning

- Given:

  Usually no reference answers

  - Objects ~~and reference answers~~ $x \in \mathcal{X}$ E.g. want the robot to walk
  - Loss/objective function $\boxed{L(\hat{y}, y)}$ Usually even hard to formulate, non-differentiable
  - Model family $f \in \mathcal{F}, f : \mathcal{X} \longrightarrow \mathcal{Y}$
- Goal:
  - Find optimal mapping $f^* = \arg\min_f L(f(x), y)$

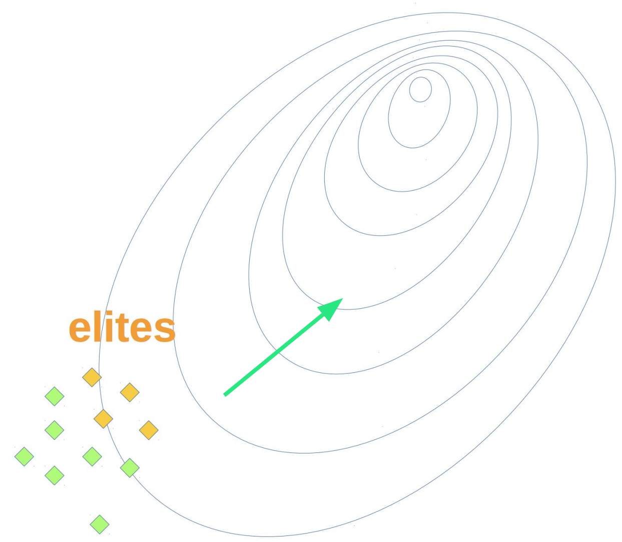
Previously

The MDP formalism

- State, Action, Reward, next State

Cross-Entropy Method (CEM)

- easy to implement, good results
- rich theoretical background
- black box
  - no knowledge of environment
  - no knowledge of intermediate rewards



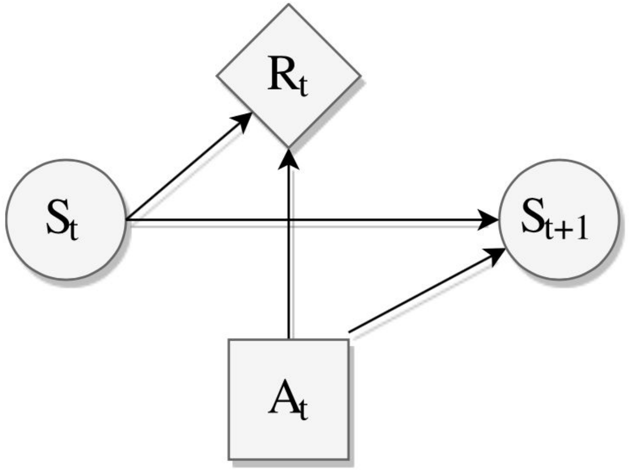
Improve on the CEM → dive into the black box

Given dynamics, how to find an optimal policy?

Definition of Markov Decision Process

MDP is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$, where

1. $\mathcal{S}$ – set of states of the world
2. $\mathcal{A}$ – set of actions
3. $\mathcal{P} : \mathcal{S} \times \mathcal{A} \mapsto \Delta(\mathcal{S})$ – state-transition function, giving us $p(s_{t+1} \mid s_t, a_t)$
4. $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ – reward function, giving us $\mathbb{E}_R \left[R(s_t, a_t) \mid s_t, a_t \right]$.



Markov property

$$p(r_t, s_{t+1} \mid s_0, a_0, r_0, \ldots, s_t, a_t) = p(r_t, s_{t+1} \mid s_t, a_t)$$

(next state, expected reward) depend on (previous state, action)

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Goal: solve an MDP by finding **an** optimal policy

1. What is the objective?
   a. Reward: discounting and design
   b. Expected objective: state- and action-value function
2. How to evaluate the objective?
   a. Bellman expectation equations
3. How to improve the objective?
   a. Bellman optimality equations

Slide source: Lecture 02 from [Practical RL course] by Shvechikov Pavel

Explaining goals to agent through reward

Reward hypothesis (R.Sutton)

Goals and purposes can be thought of as the maximization of the expected value of the cumulative sum of a received scalar signal

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Explaining goals to agent through reward

Reward hypothesis (R.Sutton)

Goals and purposes can be thought of as the maximization of the expected value of the cumulative sum of a received scalar signal

Cumulative reward is called a return:

$$G_t \triangleq R_t + R_{t+1} + R_{t+2} + \ldots + R_T$$

E.g.: reward in **chess** – value of taken opponent's piece

Explaining goals to agent through reward

Reward hypothesis (R.Sutton)

Goals and purposes can be thought of as the maximization of the expected value of the cumulative sum of a received scalar signal

Cumulative reward is called a return:

end of an episode

$$\boxed{G_t} \triangleq \boxed{R_t} + R_{t+1} + R_{t+2} + \ldots + R_{\boxed{T}}$$

immediate reward

E.g.: reward in **chess** – value of taken opponent's piece

E.g.: data center non-stop cooling system

States – temperature measurements
Actions – different fans speed
R = 0 for exceeding temperature thresholds
R = +1 for each second system is cool

What could go wrong with such a design?

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

E.g.: data center non-stop cooling system

States – temperature measurements
Actions – different fans speed
R = 0 for exceeding temperature thresholds
R = +1 for each second system is cool

What could go wrong with such a design?

Infinite return for non optimal behaviour!

$$G_t = 1 + 1 + 0 + 1 + 1 + 0 + \ldots = \sum_{t=1}^{\infty} R_t = \infty$$

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

E.g.: moving to destination

State – position, velocities of joints

Actions – actuator forces to joints

R = max(0,   d(x, B) - d(x', B))

What could go wrong with such a design?

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

E.g.: moving to destination

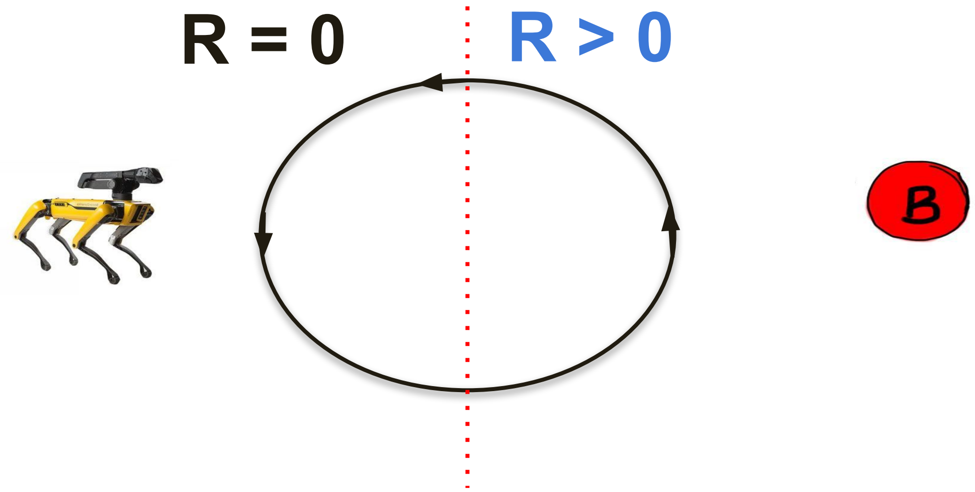
State – position, velocities of joints

Actions – actuator forces to joints

R = max(0, d(x, B) - d(x’, B))

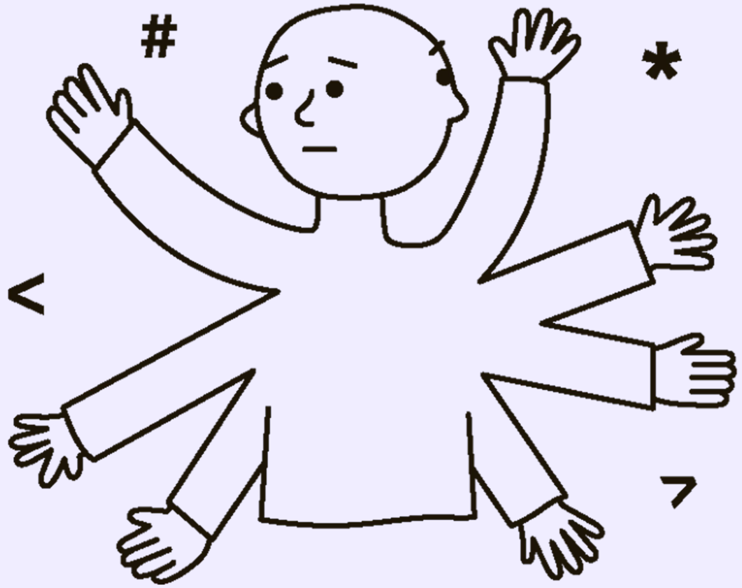
What could go wrong with such a design?

Positive
feedback loop!



Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Reward discounting



02

Reward discounting

Get rid of infinite sum by discounting

$$0 \leq \gamma < 1$$

$$G_t \triangleq R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

discount factor

The same cake compared to today's one worth

- γ times less tomorrow
- γ^2 times less the day after tomorrow

γ will eat it day by day

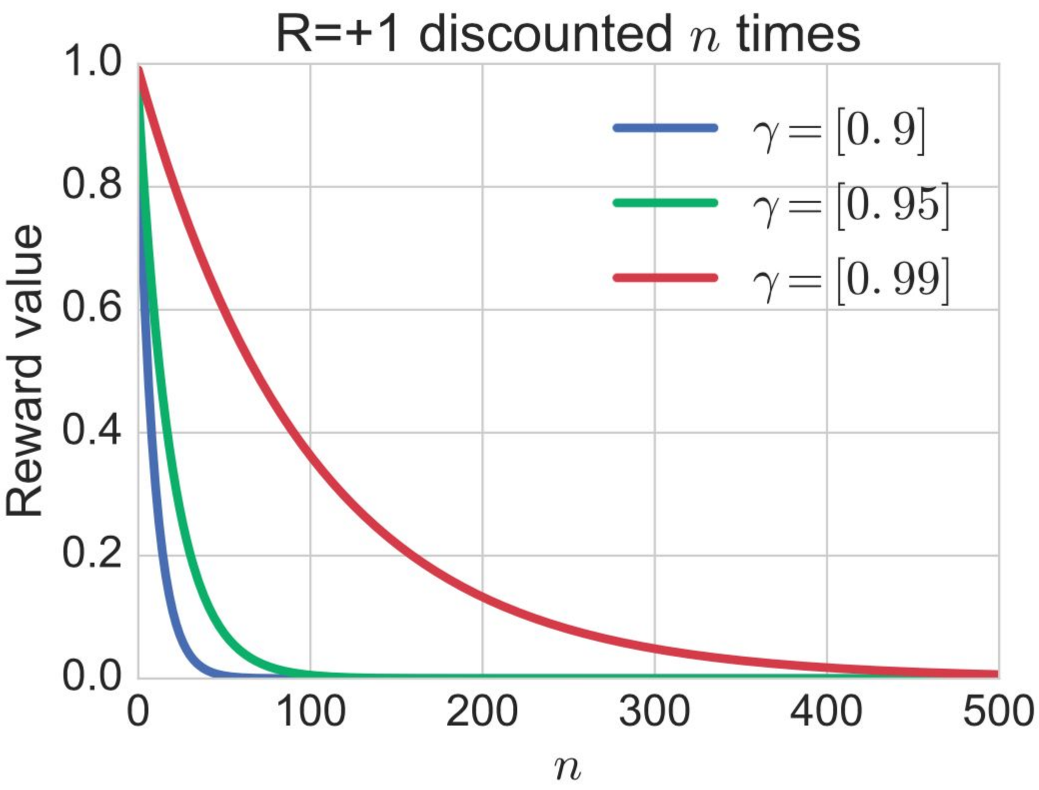
Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Discounting makes sums finite

Maximal return for **R = +1**

$$G_0 = \sum_{k=0}^{\infty} \gamma^k = \frac{1}{1-\gamma}$$

γ	0.9	0.95	0.99
$\frac{1}{1-\gamma}$	10	20	100



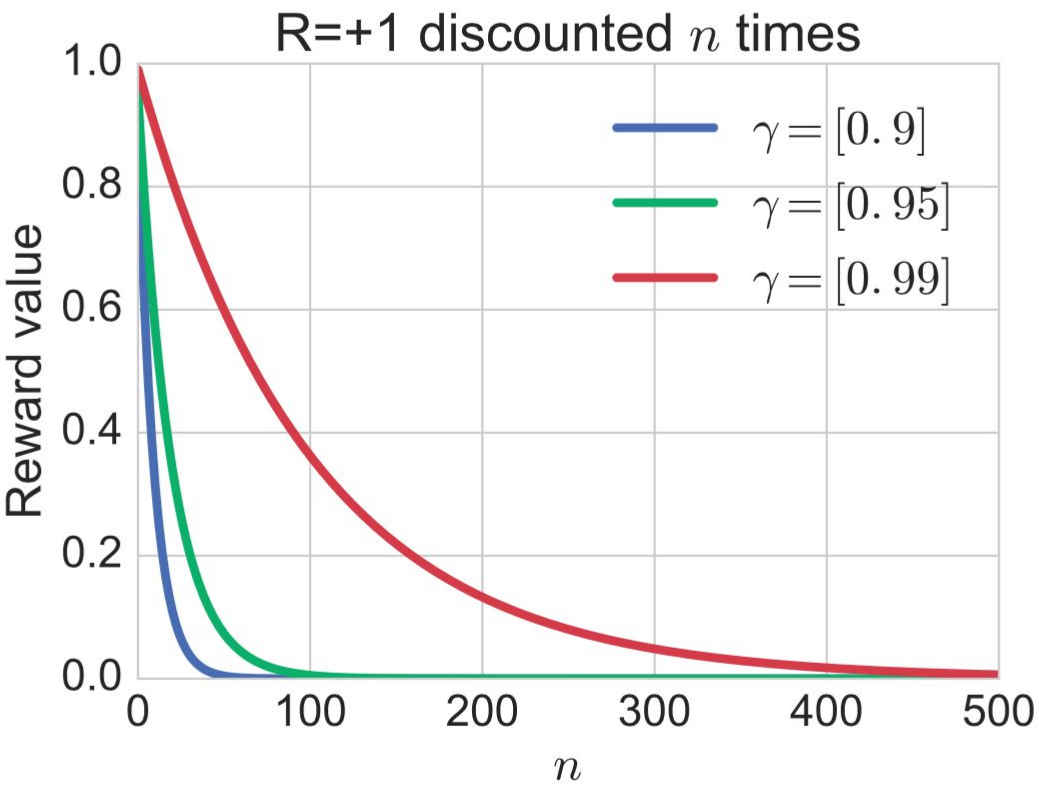
Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Discounting makes sums finite

Maximal return for R = +1

$$G_0 = \sum_{k=0}^{\infty} \gamma^k = \frac{1}{1-\gamma}$$

γ	0.9	0.95	0.99
$\frac{1}{1-\gamma}$	10	20	100



Any discounting changes optimisation task and its solution!

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Discounting is inherent to humans

Quasi-hyperbolic $$f(t) = \beta\gamma^t$$

Hyperbolic discounting $$f(t) = \frac{1}{1 + \beta t}$$

Laibson, D. (1997). Golden eggs and hyperbolic discounting. The Quarterly Journal of Economics, 112(2), 443-478.

Discounting is inherent to humans

Quasi-hyperbolic $f(t) = \beta\gamma^t$

Hyperbolic discounting $f(t) = \dfrac{1}{1 + \beta t}$

Mathematical convenience

$$G_t = R_t + \gamma(R_{t+1} + \gamma R_{t+2} + \ldots)$$
$$= \boxed{R_t + \gamma\, G_{t+1}}$$

Remember this one!
We will need it later

Laibson, D. (1997). Golden eggs and hyperbolic discounting. The Quarterly Journal of Economics, 112(2), 443-478.

Slide source: Lecture 02 from [Practical RL course] by Shvechikov Pavel

Discounting is a stationary end-of-effect model

Any action affects (1) immediate reward (2) next state

Discounting is a stationary end-of-effect model

Any action affects (1) immediate reward (2) $\boxed{\text{next state}}$

Action indirectly affects future rewards

But how long does this effect lasts?

$$
\begin{aligned}
G_0 &= R_0 + \gamma R_1 + \gamma^2 R_2 + \ldots + \gamma^T R_T \\
&= (1-\gamma) R_0 \\
&\quad + (1-\gamma)\gamma(R_0 + R_1) \\
&\quad + (1-\gamma)\gamma^2(R_0 + R_1 + R_2) \\
&\quad \cdots \\
&\quad + \gamma^T \cdot \sum_{t=0}^{T} R_t
\end{aligned}
$$

G is expected return under stationary end-of-effect model

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Discounting is a stationary end-of-effect model

Any action affects (1) immediate reward (2) $\boxed{\text{next state}}$

Action indirectly affects future rewards

But how long does this effect lasts?

“End of effect” probability

$$
\begin{aligned}
G_0 &= R_0 + \gamma R_1 + \gamma^2 R_2 + \ldots + \gamma^T R_T \\
&= \boxed{(1-\gamma)} R_0 \\
&+ \boxed{(1-\gamma)} \boxed{\gamma} (R_0 + R_1) \\
&+ \boxed{(1-\gamma)} \gamma^2 (R_0 + R_1 + R_2) \\
&\cdots \\
&+ \gamma^T \cdot \sum_{t=0}^{T} R_t
\end{aligned}
$$

“Effect continuation” probability

G is expected return under stationary end-of-effect model

Reward design – don’t shift, reward for WHAT

E.g.: chess – value of taken opponent's piece

- Problem: agent will not have a desire to win!

E.g.: moving to destination

- Problem: agent will not bother about the goal!

Reward design – don't shift, reward for WHAT

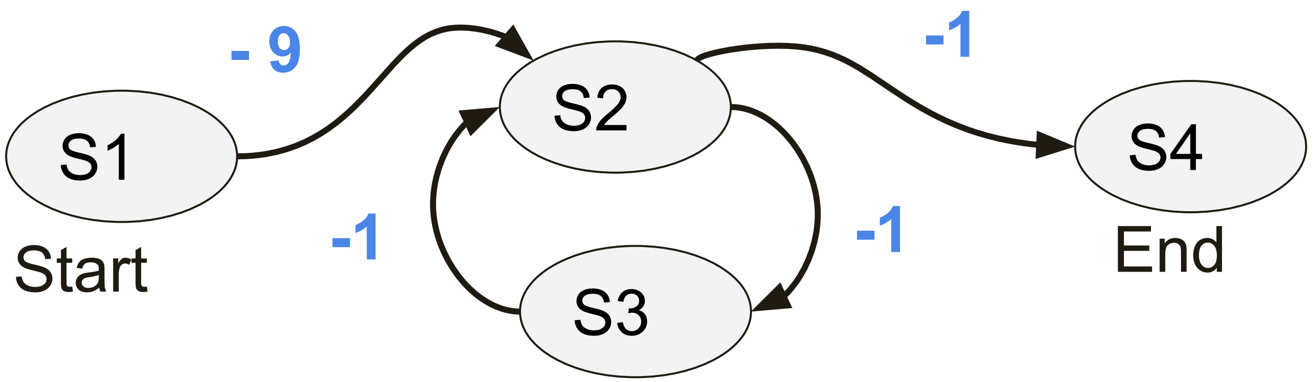
E.g.: chess – value of taken opponent's piece

- Problem: agent will not have a desire to win!

E.g.: moving to destination

- Problem: agent will not bother about the goal!

Take away: reward only for WHAT, but never for HOW



Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Reward design – don’t shift, reward for WHAT

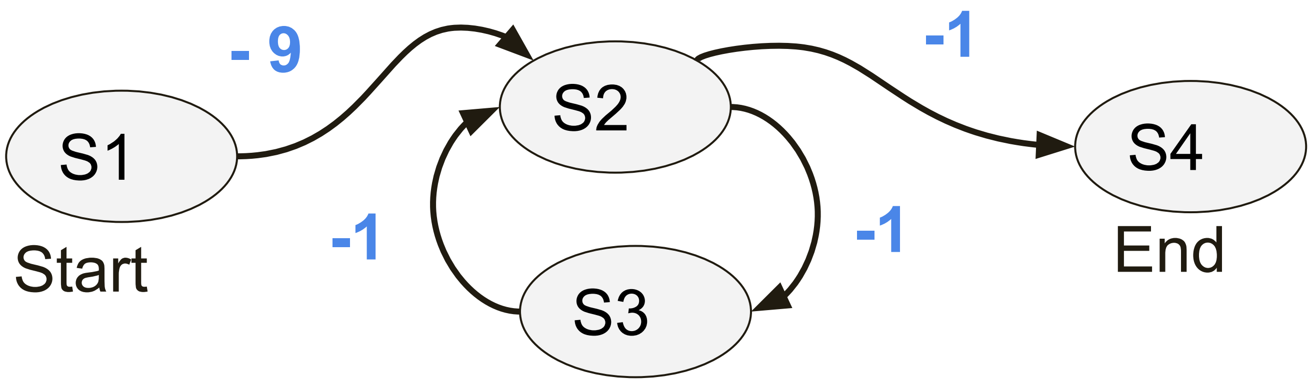
E.g.: chess – value of taken opponent's piece

- Problem: agent will not have a desire to win!

E.g.: moving to destination

- Problem: agent will not bother about the goal!

Take away: reward only for WHAT, but never for HOW



Take away: do not subtract mean from rewards

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Faulty reward functions

- Reward for ball possession in soccer
  - Vibrating near the ball
- Cyclic behaviours

Slide source: Lecture 02 from [Practical RL course]() by Shvechikov Pavel

Reward design – scaling, shaping

What transformations do not change optimal policy?

Reward **scaling** – division by positive constant

- May be useful in practise for approximate methods

Ng, A. Y., Harada, D., & Russell, S. (1999, June). Policy invariance under reward transformations: Theory and application to reward shaping. In ICML (Vol. 99, pp. 278-287).

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Reward design – scaling, shaping

What transformations do not change optimal policy?

Reward **scaling** – division by positive constant

- May be useful in practise for approximate methods

Reward **shaping** – add a potential-based shaping function F(s, a, s'):

$$R'(s, a, s') = R(s, a, s') + F(s, a, s')$$

Intuition: when no discounting F adds as much as it subtracts from the total return

$$F(s, a, s') = \gamma\Phi(s') - \Phi(s)$$

Ng, A. Y., Harada, D., & Russell, S. (1999, June). Policy invariance under reward transformations: Theory and application to reward shaping. In ICML (Vol. 99, pp. 278-287).

State-value (V) and action-value (Q) functions

03

Optimal policy maximizes expected return

$$
\begin{aligned}
\mathbb{E}\left[G_0\right] &= \mathbb{E}\left[R_0 + \gamma R_1 + \ldots + \gamma^T R_T\right] \\
&= \mathbb{E}_{E,\pi_\theta}\left[\,G_0\,\right] \\
&= \mathbb{E}_{\pi_\theta}\left[\,G_0\,\right] \\
&= \mathbb{E}\left[G_0 \mid \pi_\theta\right] \\
&= \mathbb{E}_{\substack{s_{0:T} \\ a_{0:T}}}\left[\,G_0\,\right] \\
&= \mathbb{E}_{s_0}\left[\,\mathbb{E}_{a_0|s_0}\left[\,R_0 + \mathbb{E}_{s_1|s_0,a_0}\left[\,\mathbb{E}_{a_1|s_1}\left[\,\gamma R_1 + \ldots\,\right]\,\right]\,\right]\,\right] \\
&= \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim p_\theta}\left[\,\gamma^t R_t\,\right] \\
&= \mathbb{E}_{\tau\sim p_\theta(\tau)}\left[\,G(\tau)\,\right]
\end{aligned}
$$

$$\tau \triangleq (s_0, a_0, s_1, \ldots, a_{T-1}, s_T)$$

$$p_\theta(\tau) = p(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t|s_t)\; p(s_{t+1}|s_t, a_t)$$

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

State-value function v(s)

v(s) is expected return conditional on state:

$$
\begin{aligned}
v_\pi(s) &\doteq \mathbb{E}_\pi [\, G_t \,|\, S_t = s \,] \\
&= \mathbb{E}_\pi [\, R_t + \gamma G_{t+1} \,|\, S_t = s \,] \\
&= \sum_a \pi(a \,|\, s) \sum_{r,\, s'} p(r, s' \,|\, s, a) \Big[r + \gamma \mathbb{E}_\pi [\, G_{t+1} | S_{t+1} = s' \,] \Big] \\
&= \sum_a \pi(a \,|\, s) \sum_{r,\, s'} p(r, s' \,|\, s, a) \left[r + \gamma v_\pi(s') \right]
\end{aligned}
$$

Intuition: value of following policy π from state s

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

State-value function v(s)

v(s) is expected return conditional on state:

stochasticity in policy & environment

$$
\begin{aligned}
v_\pi(s) &\triangleq \boxed{\mathbb{E}_\pi}[G_t \mid S_t = s] \\
&= \mathbb{E}_\pi[R_t + \gamma G_{t+1} \mid S_t = s] \\
&= \sum_a \pi(a \mid s) \sum_{r,s'} p(r, s' \mid s, a)\Big[r + \gamma \mathbb{E}_\pi[G_{t+1} \mid S_{t+1} = s']\Big] \\
&= \sum_a \pi(a \mid s) \sum_{r,s'} p(r, s' \mid s, a)[r + \gamma v_\pi(s')]
\end{aligned}
$$

Intuition: value of following policy π from state s

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

State-value function v(s)

v(s) is expected return conditional on state:

stochasticity in policy & environment

Environment stochasticity

$$v_\pi(s) \doteq \boxed{\mathbb{E}_\pi}[G_t \mid S_t = s]$$

$$= \mathbb{E}_\pi[R_t + \gamma G_{t+1} \mid S_t = s]$$

$$= \sum_a \boxed{\pi(a \mid s)} \sum_{r,s'} \boxed{p(r, s' \mid s, a)} \Big[r + \gamma \mathbb{E}_\pi[G_{t+1} | S_{t+1} = s']\Big]$$

Policy stochasticity

$$= \sum_a \pi(a \mid s) \sum_{r,s'} p(r, s' \mid s, a) [r + \gamma v_\pi(s')]$$

Intuition: value of following policy π from state s

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

State-value function v(s)

v(s) is expected return conditional on state:

stochasticity in policy & environment

Environment stochasticity

$$
\begin{aligned}
v_\pi(s) &\doteq \boxed{\mathbb{E}_\pi}\left[G_t \mid S_t = s\right] \\
&= \mathbb{E}_\pi\left[R_t + \gamma G_{t+1} \mid S_t = s\right] \\
&= \sum_a \boxed{\pi(a \mid s)} \sum_{r,s'} \boxed{p(r, s' \mid s, a)} \Big[r + \gamma \underline{\mathbb{E}_\pi\left[G_{t+1} \mid S_{t+1} = s'\right]}\Big] \\
&= \sum_a \pi(a \mid s) \sum_{r,s'} p(r, s' \mid s, a) \left[r + \gamma \underline{v_\pi(s')}\right]
\end{aligned}
$$

Policy stochasticity

By definition

Intuition: value of following policy π from state s

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Action-value function q(s, a)

Is <u>expected</u> return conditional on <u>state **and action**</u>:

Intuition: value of following policy π <u>after</u> committing action **a** in state **s**

$$
\begin{aligned}
q_\pi(s,a) &= \mathbb{E}_\pi\left[\, G_t \,|\, S_t = s,\, A_t = a \,\right] \\
&= \mathbb{E}_\pi\left[\, R_t + \gamma G_{t+1} \,|\, S_t = s,\, A_t = a \,\right] \\
&= \sum_{r,\,s'} p(r, s' \,|\, s, a)\Big[r + \gamma \mathbb{E}_\pi\left[\, G_{t+1} | S_{t+1} = s' \,\right]\Big] \\
&= \sum_{r,\,s'} p(r, s' \,|\, s, a)\left[r + \gamma v_\pi(s') \right]
\end{aligned}
$$

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Action-value function q(s, a)

Is expected return conditional on state **and action**:

Intuition: value of following policy π after committing action **a** in state **s**

No policy stochasticity at first step

$$
\begin{aligned}
q_\pi(s,a) &= \mathbb{E}_\pi\left[\, G_t \mid S_t = s, \boxed{A_t = a\,}\right] \\
&= \mathbb{E}_\pi\left[\, R_t + \gamma G_{t+1} \mid S_t = s, A_t = a \,\right] \\
&= \sum_{r,\,s'} p(r, s' \mid s, a)\Big[r + \gamma \mathbb{E}_\pi\left[\, G_{t+1} | S_{t+1} = s' \,\right]\Big] \\
&= \sum_{r,\,s'} p(r, s' \mid s, a)\left[r + \gamma v_\pi(s')\right]
\end{aligned}
$$

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

Relations between v(s) and q(s,a)

We already know how to write q(s,a) in terms of v(s)

$$q_\pi(s,a) = \sum_{r,s'} p(r,s' \mid s,a)\left[r + \gamma v_\pi(s')\right]$$

What about v(s) in terms of q(s,a)?

Relations between v(s) and q(s,a)

We already know how to write q(s,a) in terms of v(s)

$$q_\pi(s,a) = \sum_{r,s'} p(r,s' \mid s,a)\left[r + \gamma v_\pi(s')\right]$$

What about v(s) in terms of q(s,a)?

$$\begin{aligned} v_\pi(s) &= \sum_a \pi(a \mid s) \sum_{r,s'} p(r,s' \mid s,a)\left[r + \gamma v_\pi(s')\right] \\ &= \sum_a \pi(a \mid s) q_\pi(s,a) \end{aligned}$$

So, we could now write q(s, a) in terms of q(s,a)!

$$q_\pi(s,a) = \sum_{r,s'} p(r,s' \mid s,a)\left[r + \gamma \sum_{a'} \pi(a' \mid s') q_\pi(s',a')\right]$$

Slide source: Lecture 02 from Practical RL course by Shvechikov Pavel

All in one slide

$$G_t = \sum_{t'=t}^{T} \gamma^{(t'-t)} r_{t'}$$

$$Q^\pi(s,a) = E_\pi[G_t | s_t = s, a_t = a]$$

$$V^\pi(s) = E_\pi[G_t | s_t = s]$$

Recurrent relations

$$Q^\pi(s,a) = E_{s_{t+1}}[r_t + \gamma V^\pi(s_{t+1})]$$

$$Q^\pi(s,a) = E_{s_{t+1}, a_{t+1} \sim \pi}[r_t + \gamma Q^\pi(s_{t+1}, a_{t+1})]$$

Slide source: Lecture 04 from [Practical RL course]

Optimal policy

For all π, s, a: $Q^{\pi^*}(s,a) \geq Q^{\pi}(s,a)$

$$\pi^*(s) = argmax_a Q^{\pi^*}(s,a)$$

Bellman optimality equation

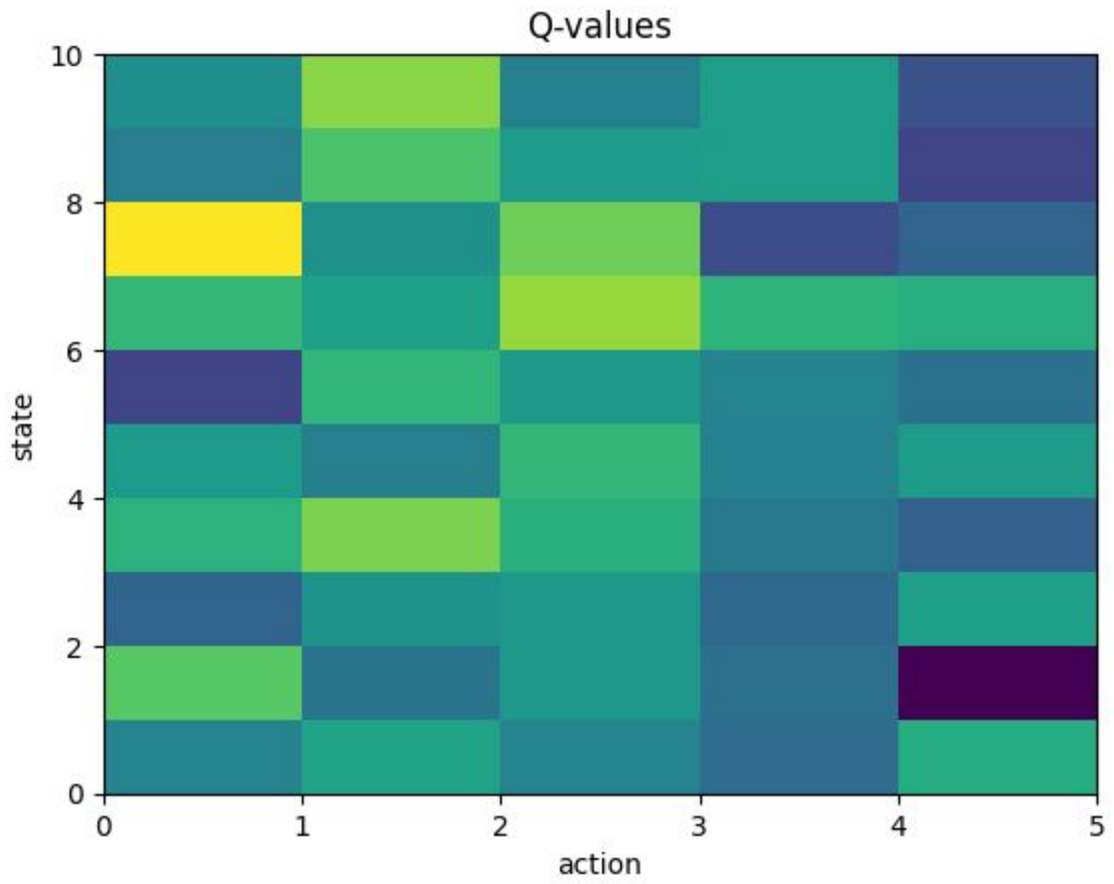
$$Q^*(s_t,a) = E_{s_{t+1}}[r_t + \max_{a'} Q^*(s_{t+1},a')]$$

Slide source: Lecture 04 from [Practical RL course]

Optimal policy (for tabular case)

For all π, s, a: $\quad Q^{\pi^*}(s,a) \geq Q^{\pi}(s,a)$

$$\pi^*(s) = argmax_a Q^{\pi^*}(s,a)$$



Bellman optimality equation

$$Q^*(s_t, a) = E_{s_{t+1}}[r_t + \max_{a'} Q^*(s_{t+1}, a')]$$

Slide source: Lecture 04 from [Practical RL course]

Q-learning

Training step

$$Q(s_t, a_t) \longleftarrow Q(s_t, a_t) + \alpha \boxed{(r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t))}$$

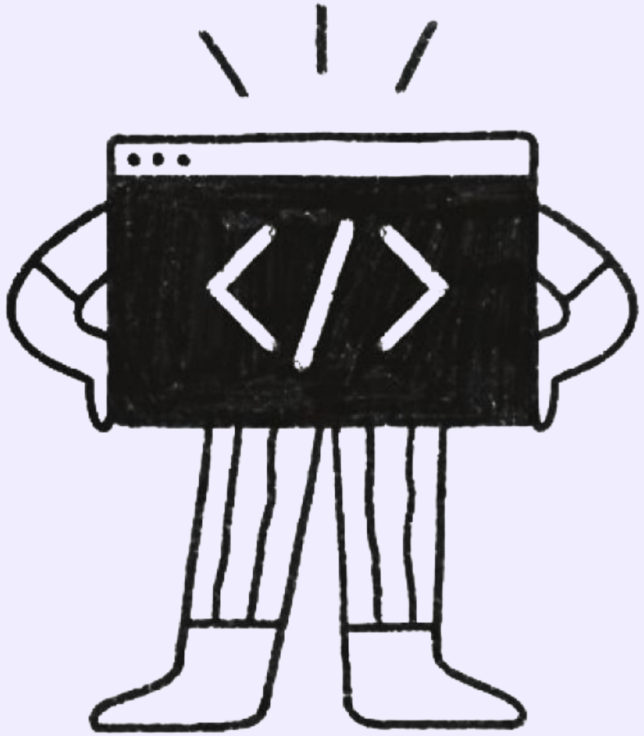
Q-learning as MSE minimization

$$L = (r_t + \gamma \max_{a'} \underbrace{\boxed{Q(s_{t+1}, a')}}_{\text{Const}} - Q(s_t, a_t))^2$$

$$\nabla L = 2 \cdot \boxed{(r_t + \gamma \max_{a'} \boxed{Q(s_{t+1}, a')} - Q(s_t, a_t))}$$

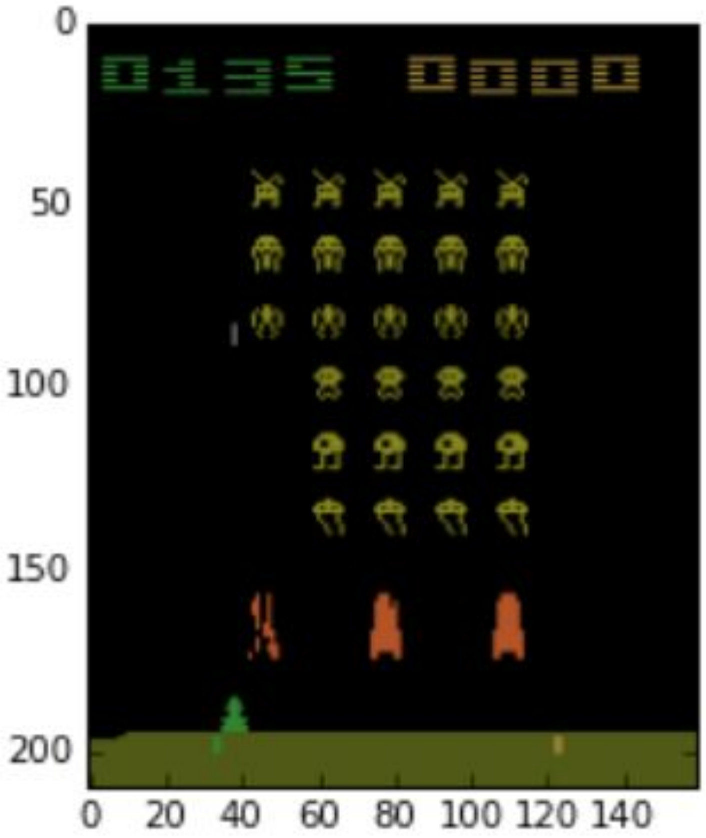
Slide source: Lecture 04 from [Practical RL course]

Approximate Q-learning



04

Real world



How many states are there? Approximately

$$|S| = 2^{210 \cdot 160 \cdot 8 \cdot 3}$$

Slide source: Lecture 04 from [Practical RL course]

Problem

State space is usually large, sometimes continuous.

Two solutions:

- Binarize state space *(obvious)*
- Approximate agent with a function *(crossentropy method)*

Which one would you prefer for atari?

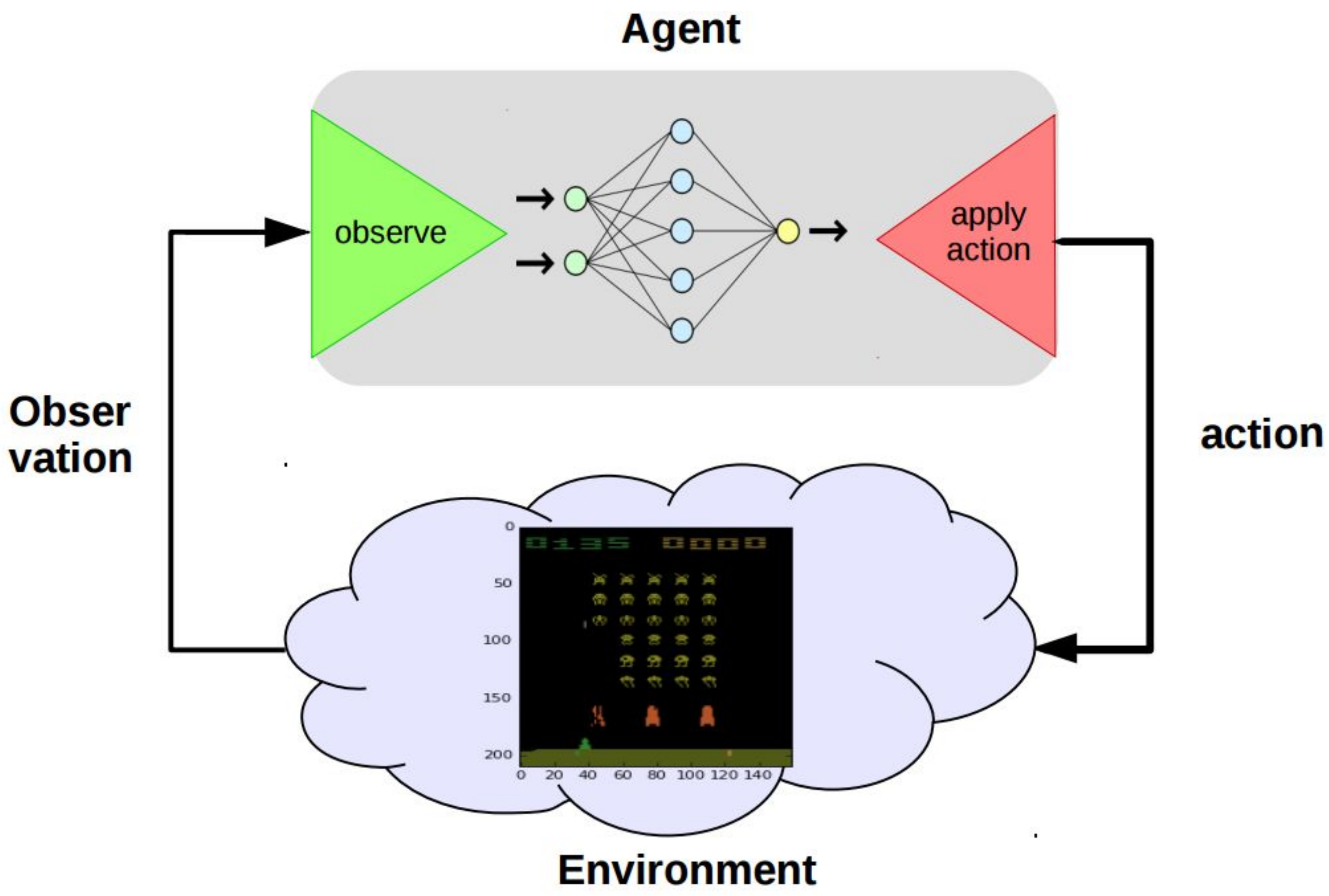
Problem

State space is usually large, sometimes continuous.

And so is action space;

Two solutions:

- Binarize state space ⬅ Too many bins or handcrafted features
- Approximate agent with a function ⬅ Let's pick this one



Slide source: Lecture 04 from [Practical RL course]

From tables to approximations

- Before:
  - For all states, for all actions, remember Q(s,a)

- Now:
  - Approximate Q(s,a) with some function
  - e.g. linear model over state features

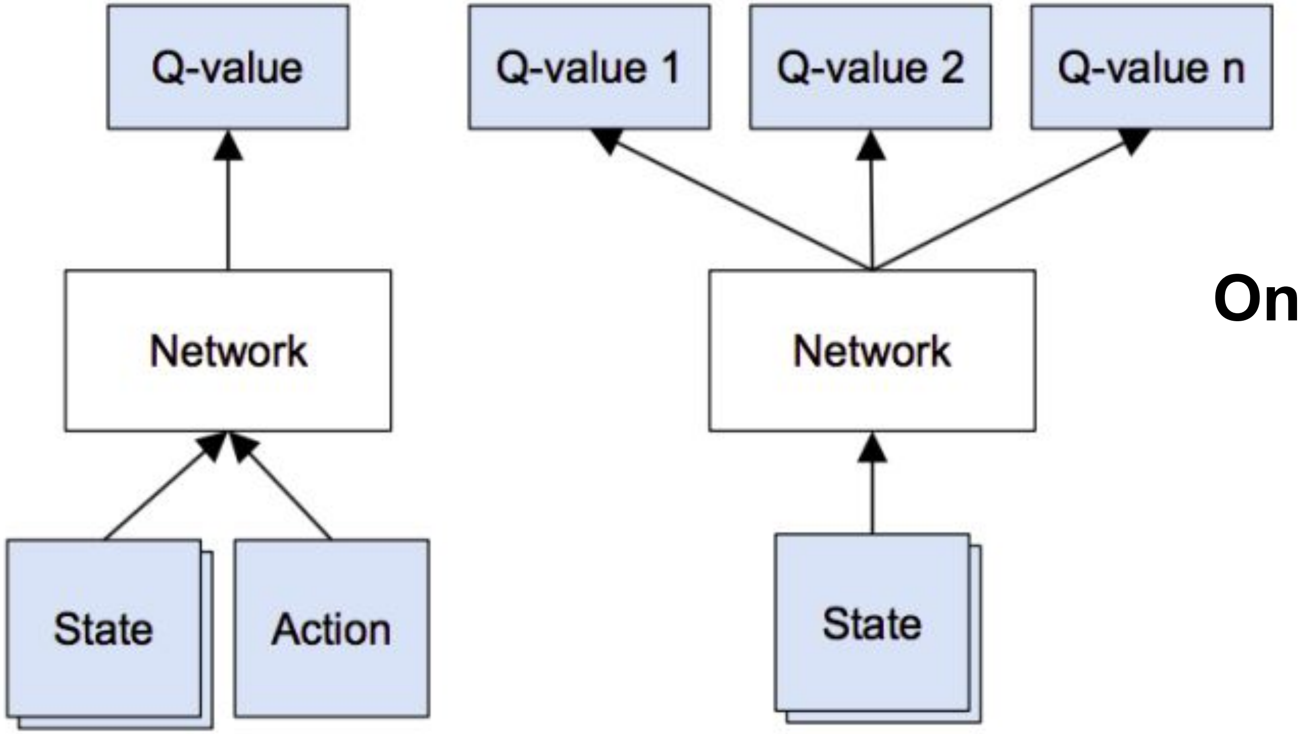
$$argmin_{w,b}(Q(s_t,a_t)-[r_t+\gamma \cdot max_{a'} Q(s_{t+1},a')])^2$$

Question: should we use **classification** or **regression** model?
(e.g. logistic regression Vs linear regression)

Slide source: Lecture 04 from [Practical RL course]()

Possible architectures

Continuous control or large number of actions



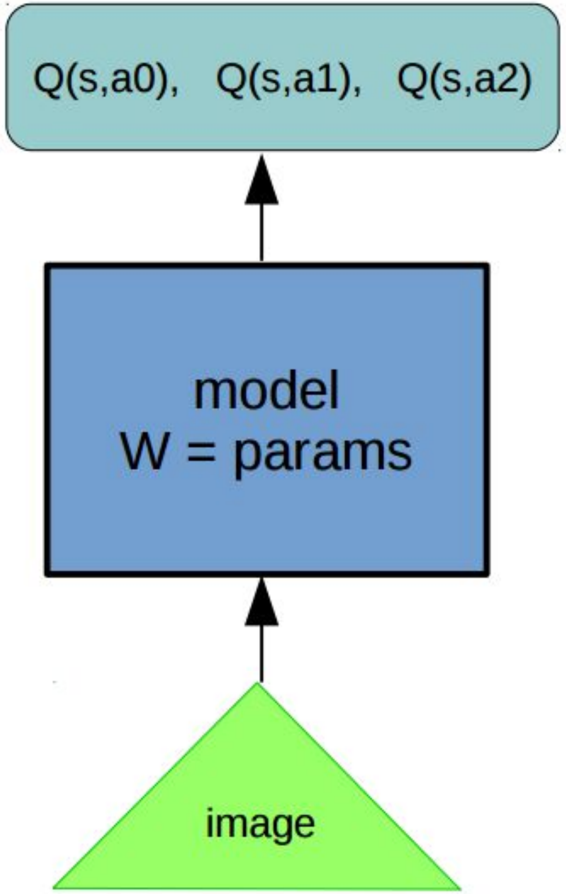
One pass for all actions

Given **(s,a)**
Predict Q(s,a)

Given **s** predict all q-values
Q(s,a0), Q(s,a1), Q(s,a2)

Slide source: Lecture 04 from [Practical RL course]

Approximate Q-learning



Q-values:

$$\hat{Q}(s_t, a_t) = r + \gamma \cdot max_{a'} Q(s_{t+1}, a')$$

Objective:

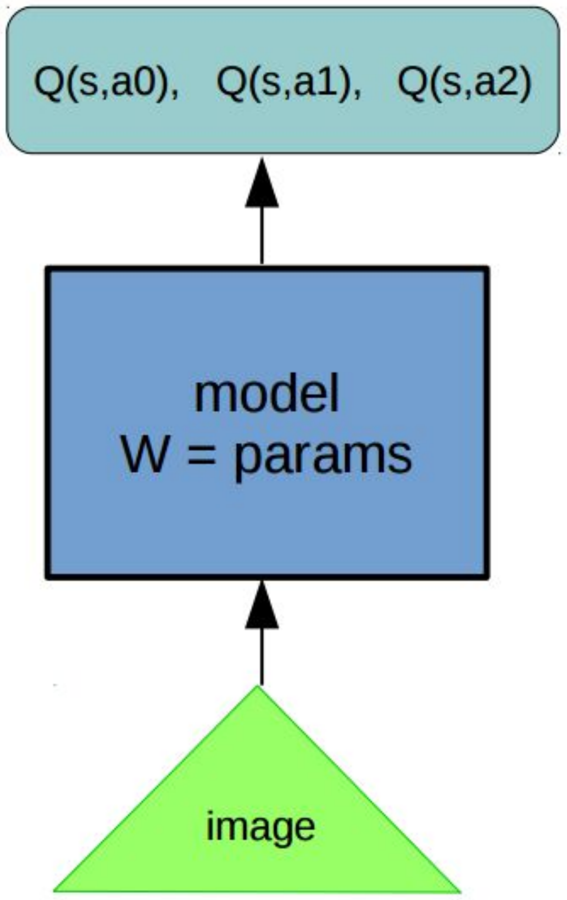
$$L = (Q(s_t, a_t) - [r + \gamma \cdot max_{a'} \underbrace{\boxed{Q(s_{t+1}, a')}}_{\text{Const}}])^2$$

Gradient step:

$$w_{t+1} = w_t - \alpha \cdot \frac{\delta L}{\delta w}$$

Slide source: Lecture 04 from [Practical RL course]

Approximate Q-learning



Objective:

$$L = \left(Q(s_t, a_t) - \underbrace{\hat{Q}(s_t, a_t)}_{\text{consider const}}\right)^2$$

Q-learning:

$$\hat{Q}(s_t, a_t) = r + \gamma \cdot max_{a'} Q(s_{t+1}, a')$$

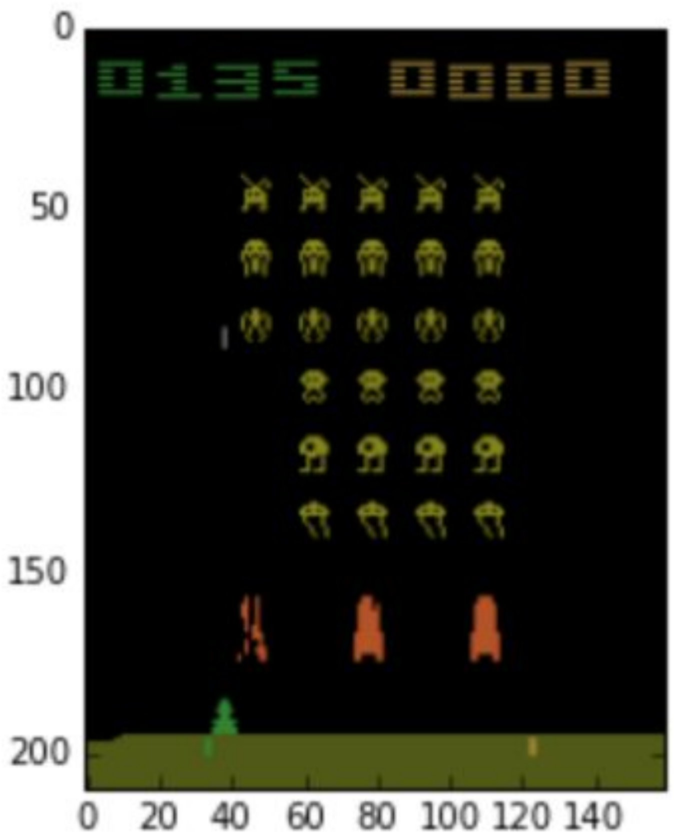
SARSA:

$$\hat{Q}(s_t, a_t) = r + \gamma \cdot Q(s_{t+1}, a_{t+1})$$

Expected Value SARSA:

$$\hat{Q}(s_t, a_t) = r + \gamma \cdot \underset{a' \sim \pi(a|s)}{E} Q(s_{t+1}, a')$$

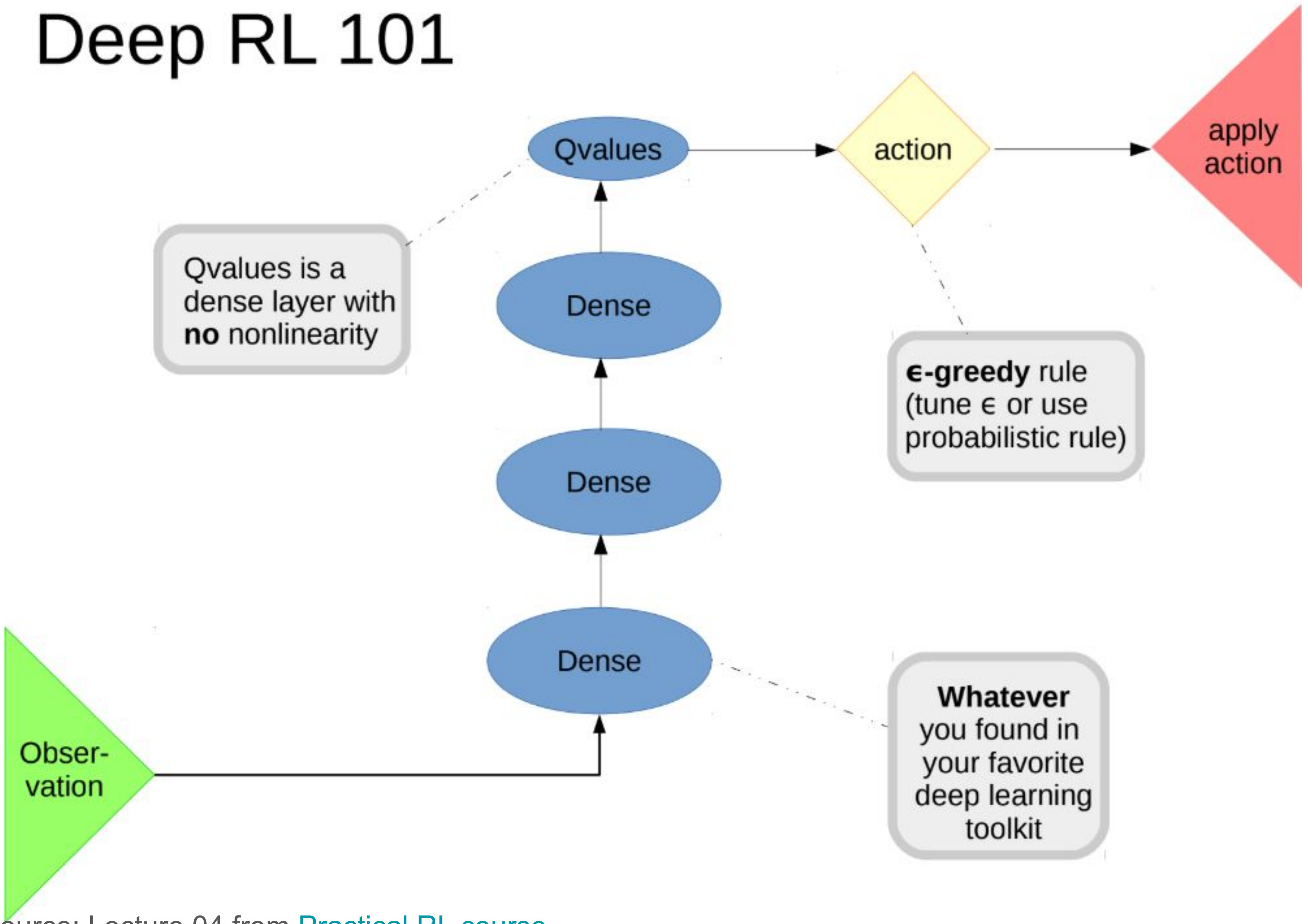
Slide source: Lecture 04 from [Practical RL course]



What kind of network digests images well?

Slide source: Lecture 04 from Practical RL course

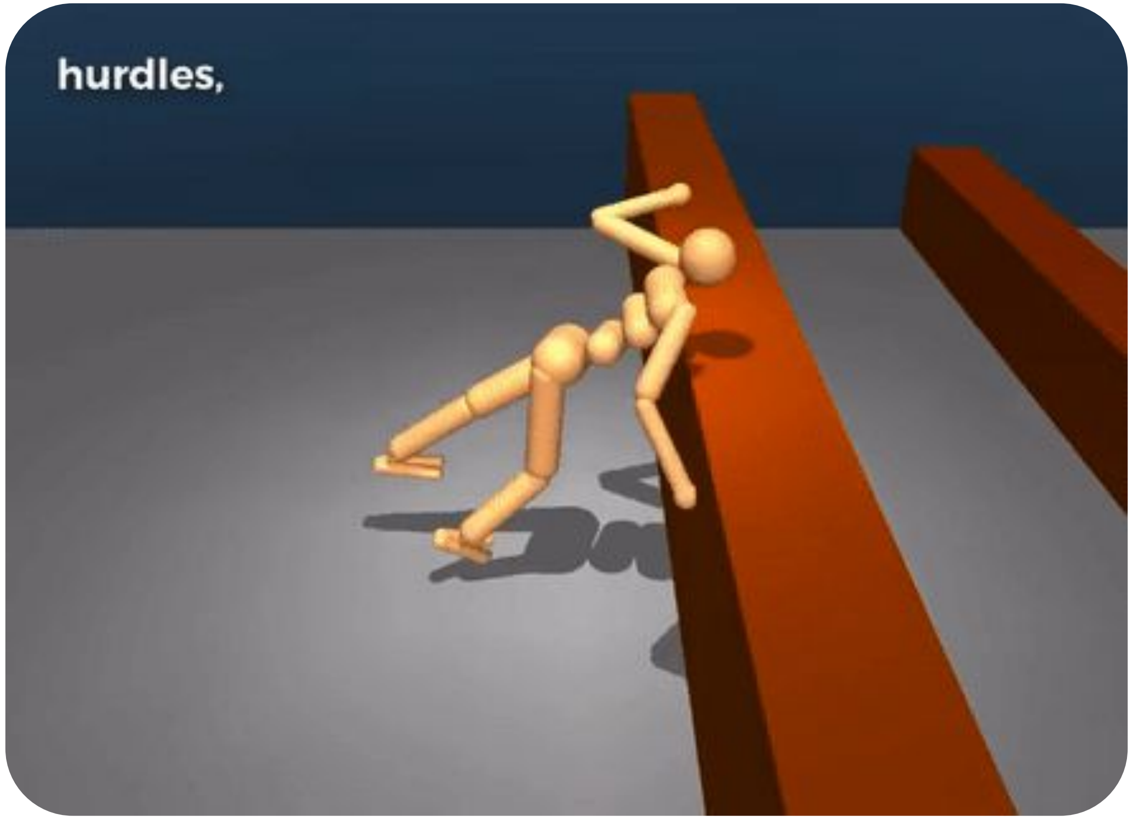
Basic deep Q-learning



Slide source: Lecture 04 from [Practical RL course]

Outro

- We can approximate the reward function and exploit it!
- We just reformulate the RL problem and still use Supervised Learning approaches
- Remember the Markov assumptions (again!)



source: Emergence of Locomotion Behaviours in Rich Environments, DeepMind

[@Rads_ai](https://t.me/Rads_ai)

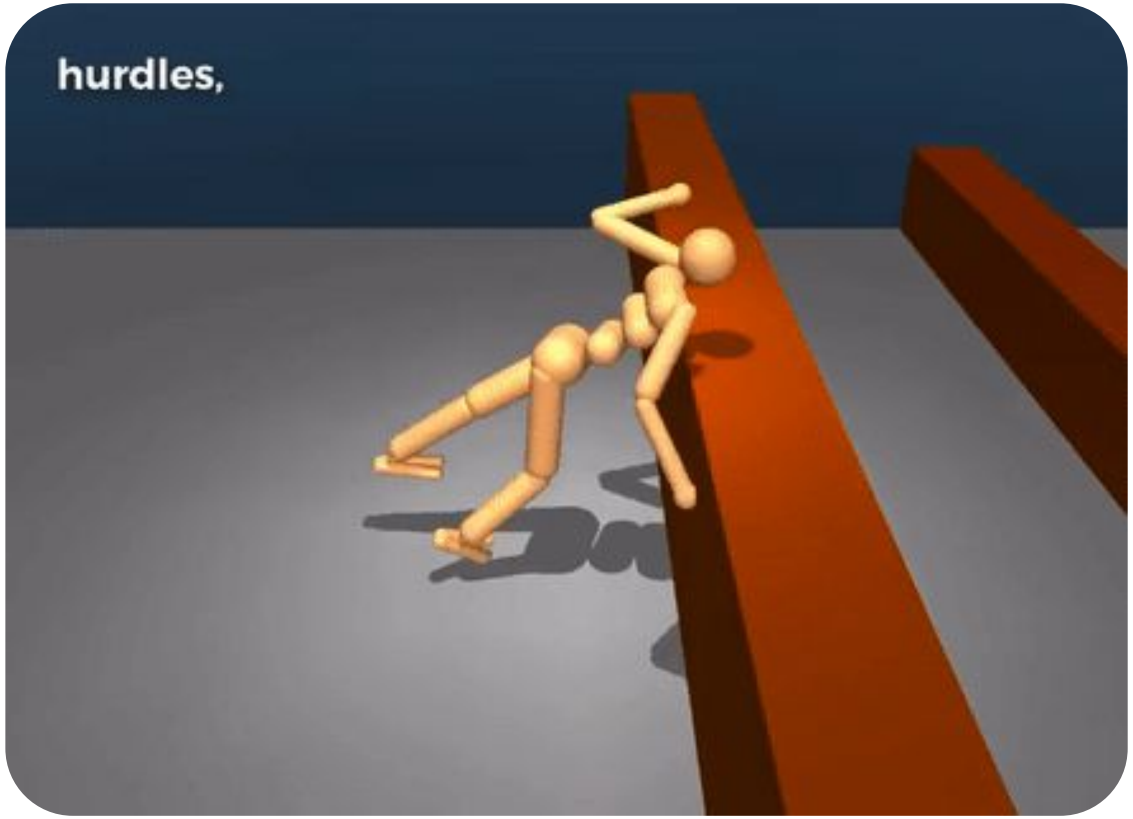
Канал Радослава
с текстовыми
разборами занятий

[@Young_and_Yandex](https://t.me/Young_and_Yandex)

Канал стажировок
Яндекса

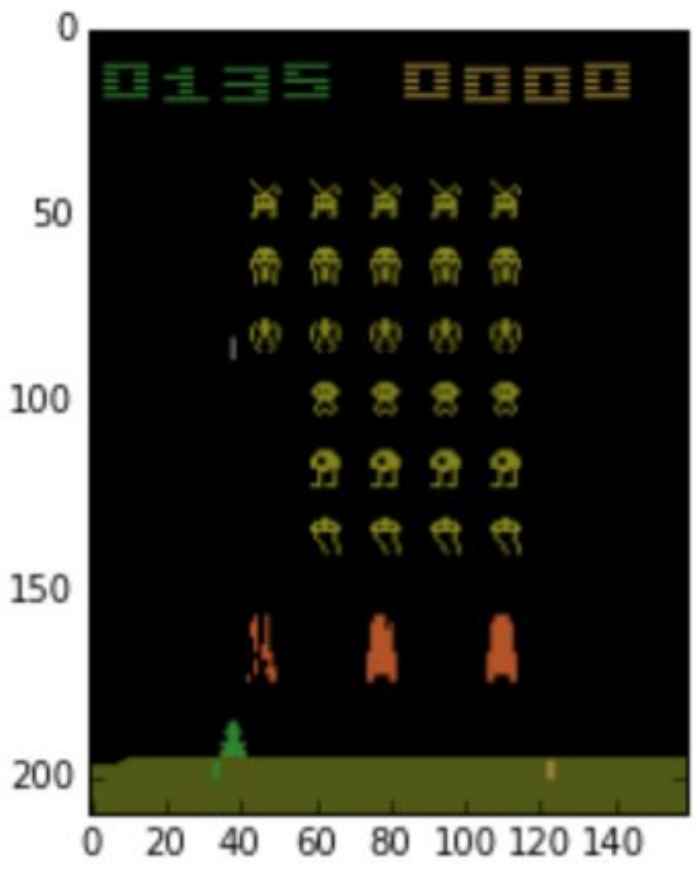
Y&Y

Outro..?



source: Emergence of Locomotion Behaviours in Rich Environments, DeepMind


WHAT IF I TOLD YOU
THAT IT'S ALL ONE GREAT BIG LIE.
imgflip.com

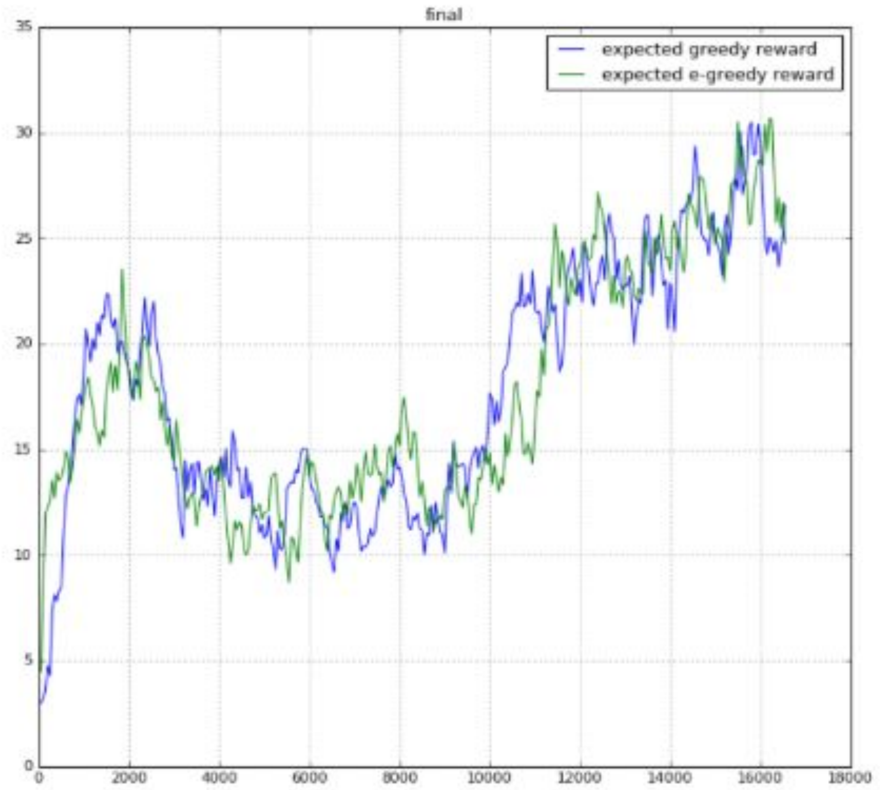


How bad it is if agent spends

next 1000 ticks under the left rock?
(while training)

Problem

- Training samples are **not “i.i.d”**,
- Model forgets parts of environment it hasn't visited for some time
- Drops on learning curve
- **Any ideas?**

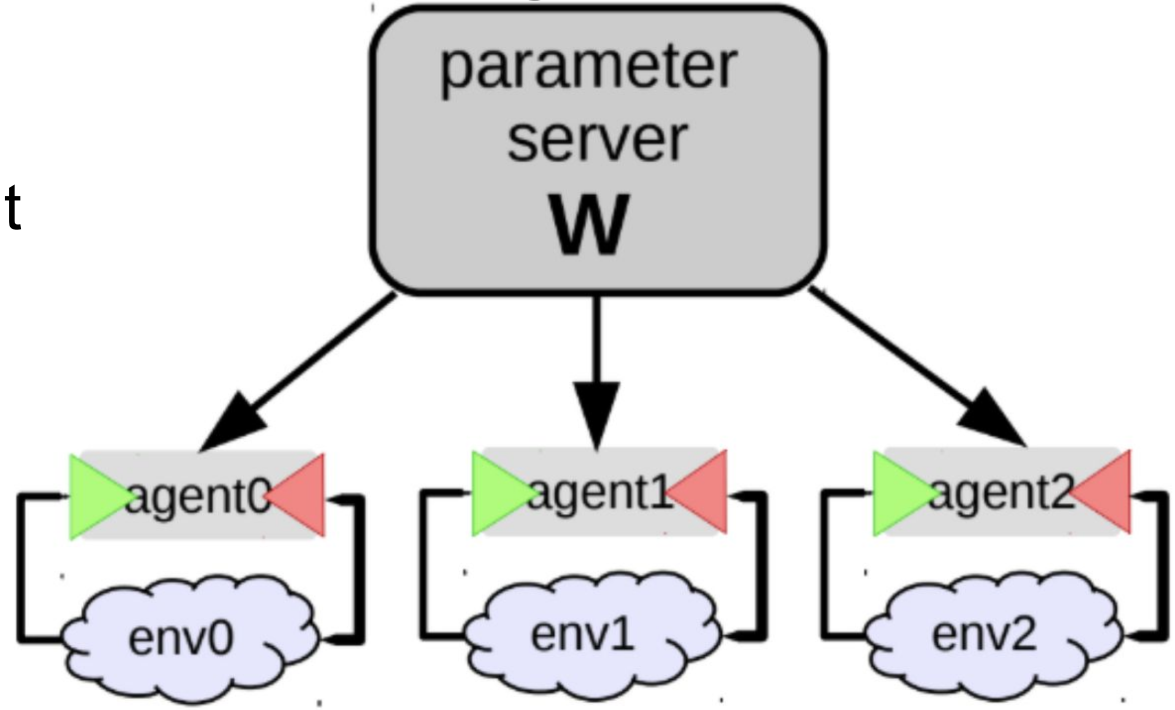


Slide source: Lecture 04 from [Practical RL course]

Multiple agents trick

Idea: Throw in several agents with shared **W**.

- Chances are, they will be exploring different parts of the environment
- More stable training
- Requires a lot of interaction



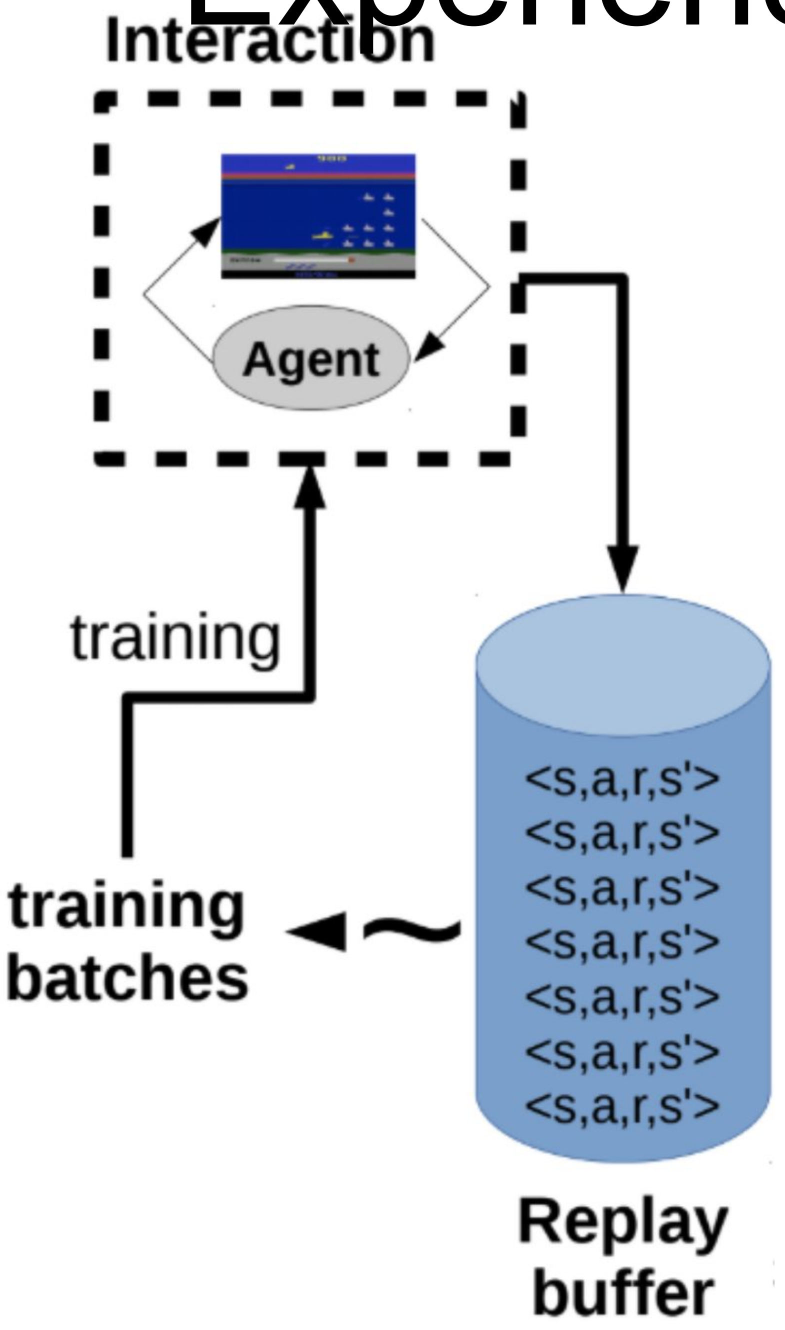
Question: your agent is a real robot car.
Will there be any problems?

Slide source: Lecture 04 from [Practical RL course]

Experience replay

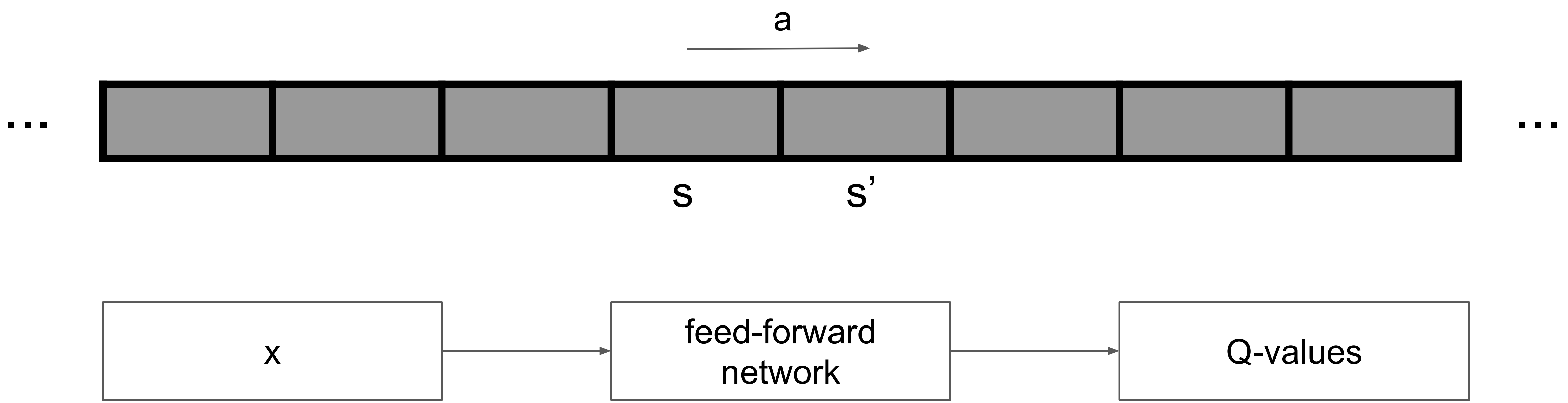
Idea: store several past interactions
<s,a,r,s'>
Train on random subsamples

- Atari DQN > 10^5 interactions
- Closer to i.i.d.
  pool contains several sessions
- Older interactions were obtained
  under weaker policy



Slide source: Lecture 04 from [Practical RL course]

Autocorrelation



Target is based on prediction

Q(s, a) correlates with Q(s’, a)

Slide source: Lecture 04 from [Practical RL course]

Target network

Idea: use network with frozen weights to compute the target

$$L(\Theta) = E_{s\sim S, a\sim A}[(Q(s,a,\Theta) - (r + \gamma \max_{a'} \underbrace{\boxed{Q(s',a',\Theta^-)}}_{\textbf{Const}}))^2]$$

where Θ^- is the frozen weights

Hard target network:

Update Θ^- every **n** steps and set its weights as Θ

Target network

Idea: use network with frozen weights to compute the target

$$L(\Theta) = E_{s\sim S,a\sim A}[(Q(s,a,\Theta) - (r + \gamma \max_{a'} \underbrace{\boxed{Q(s',a',\Theta^-)}}_{\text{Const}}))^2]$$

where Θ^- is the frozen weights

Hard target network:

Update Θ^- every **n** steps and set its weights as Θ

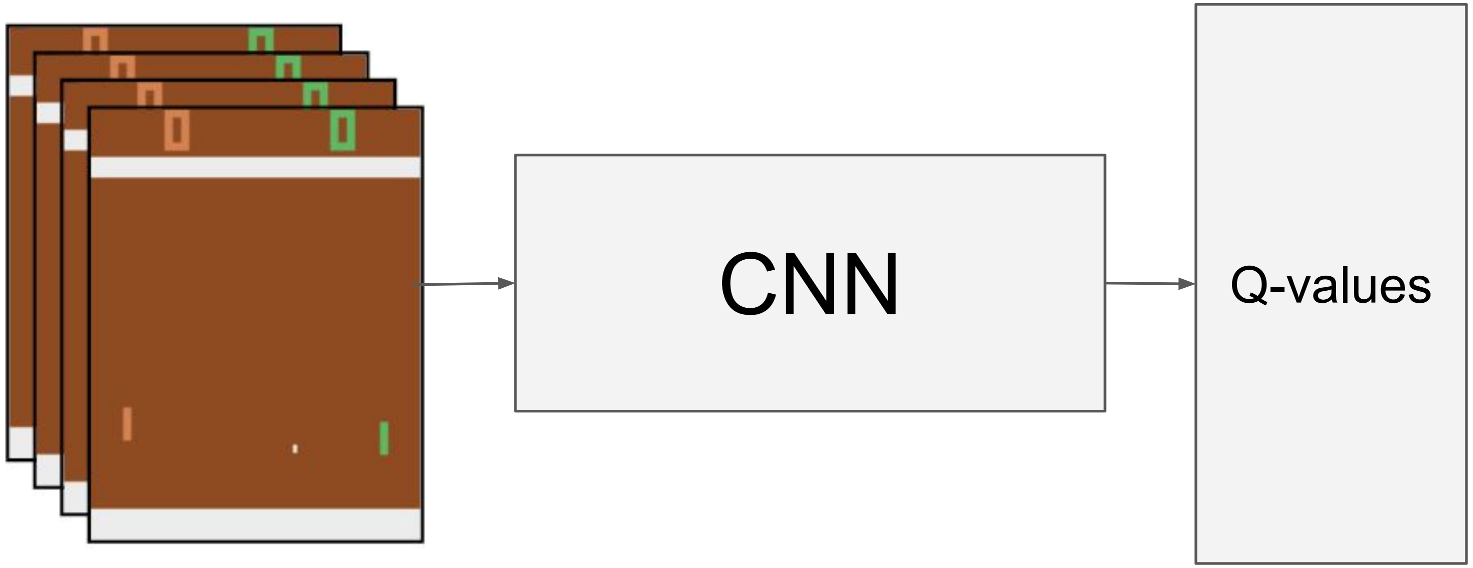
Soft target network:

Update Θ^- every step:

$$\Theta^- = (1-\alpha)\Theta^- + \alpha\Theta$$

Playing Atari with Deep Reinforcement Learning

(2013, Deepmind)



4 last frames as input

Experience replay

10^6 last transitions

Update weights using:

$$L(\Theta) = E_{s\sim S,a\sim A}[(Q(s,a,\Theta) - (r + \gamma \max_{a'} Q(s',a',\Theta^-)))^2]$$

Update Θ^- every 5000 train steps

Slide source: Lecture 04 from [Practical RL course]

Problem of overestimation

We use “max” operator to compute the target

$$L(s,a) = (Q(s,a) - (r + \gamma \max_{a'} Q(s',a')))^2$$

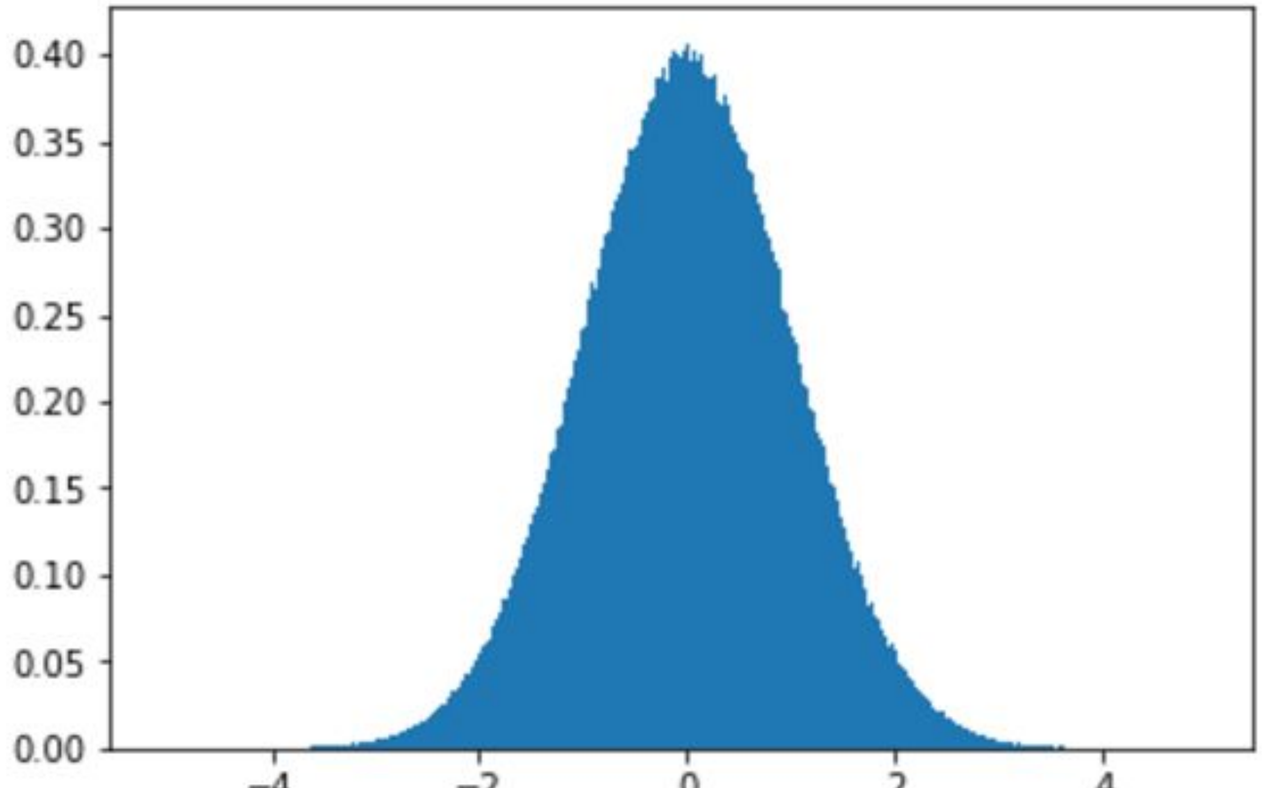
We have a problem

(although we want $\mathrm{E}_{s \sim S, a \sim A}[L(s,a)]$ to be equal zero)

Problem of overestimation

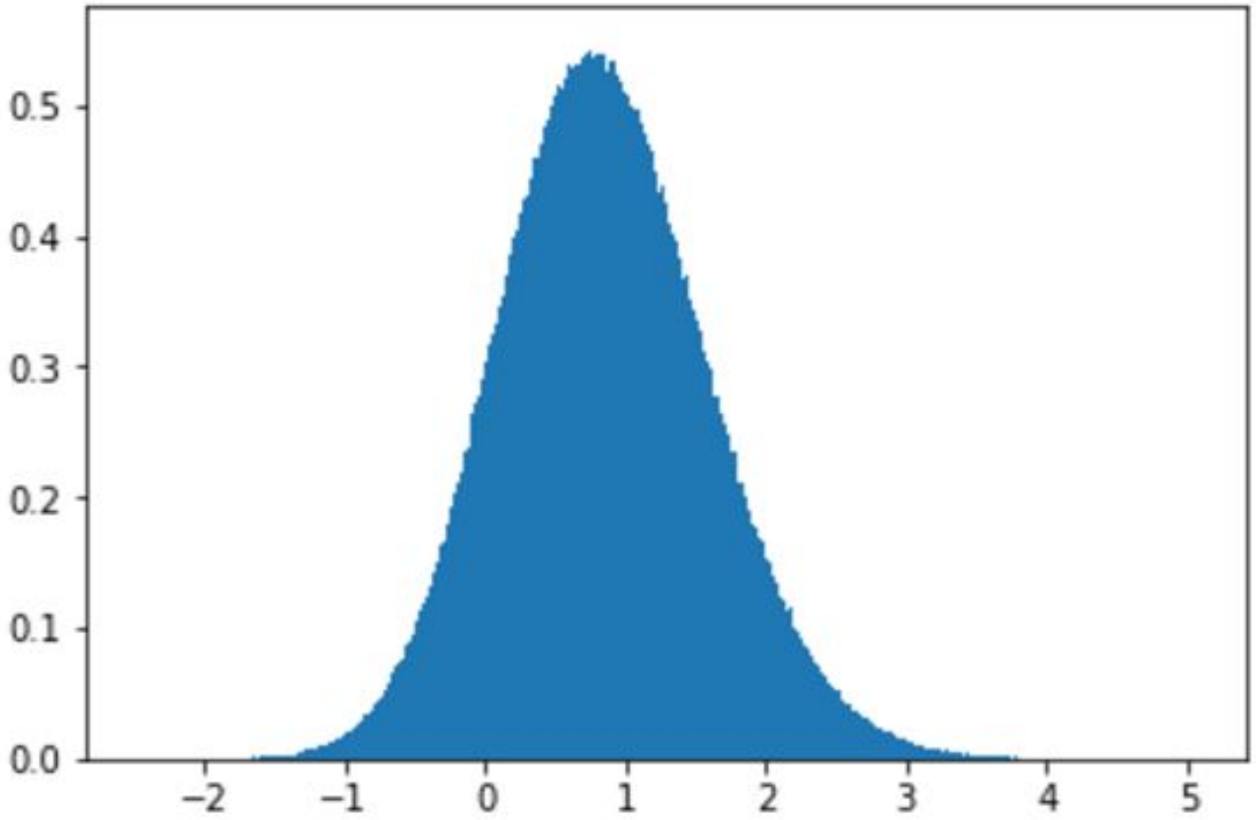
Normal distribution
$3*10^6$ samples

mean: ~0.0004



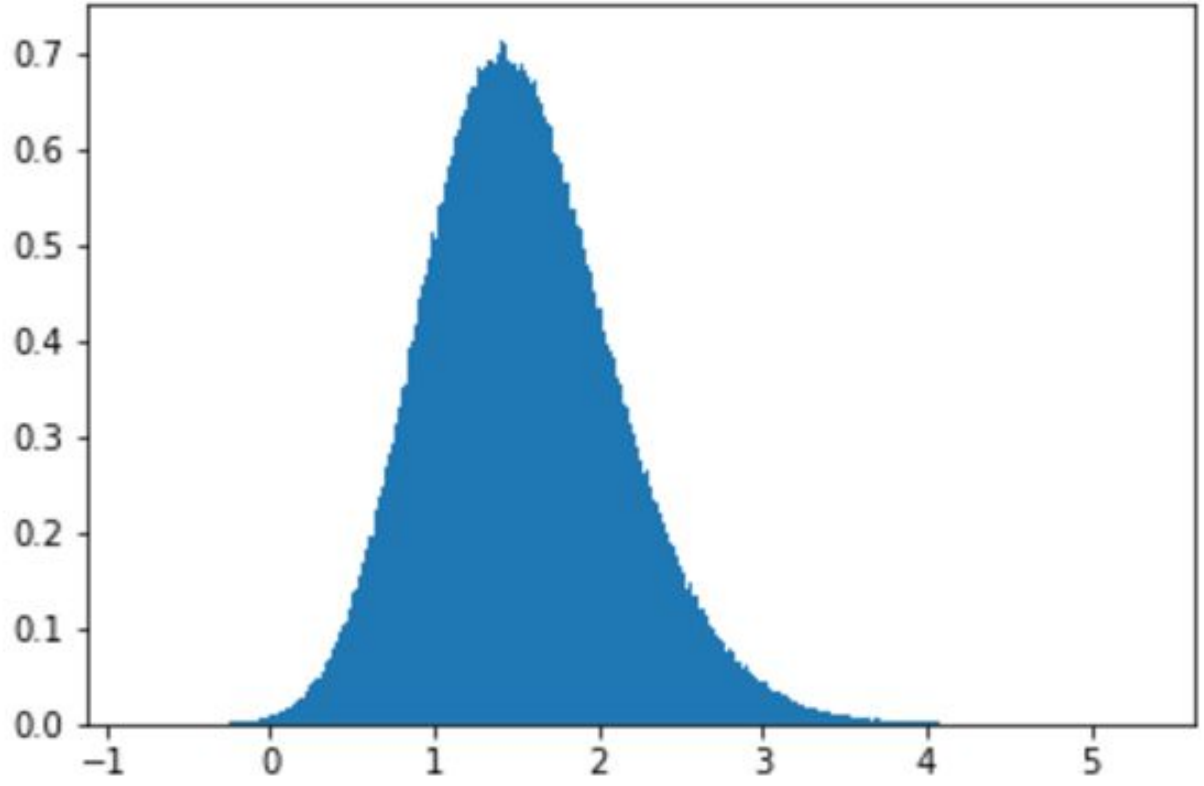
Problem of overestimation

Normal distribution
$3*10^6$ x 3 samples
Then take maximum of every tuple
mean: ~0.8467

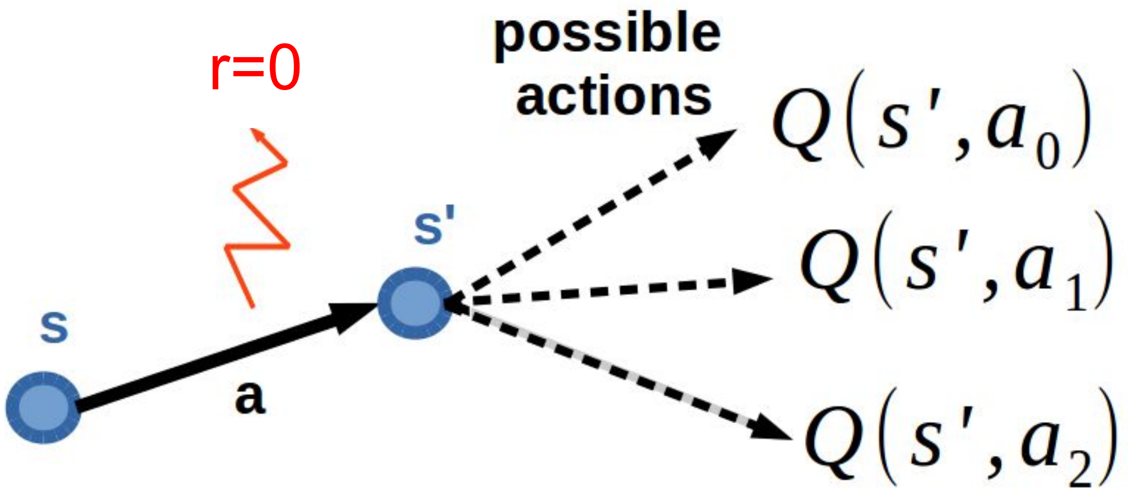


Problem of overestimation

Normal distribution
$3 * 10^6 \times 10$ samples
Then take maximum of every tuple
mean: ~1.538



Problem of overestimation

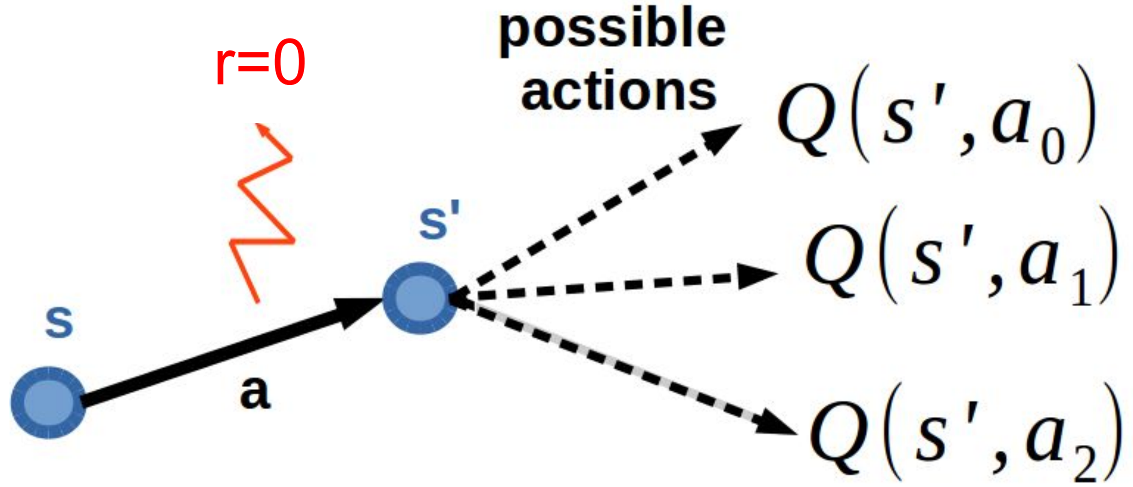


Suppose true $Q(s', a')$ are equal to **0** for all a'

But we have an approximation (or other) error $\sim N(0, \sigma^2)$

So $Q(s, a)$ should be equal to **0**

Problem of overestimation

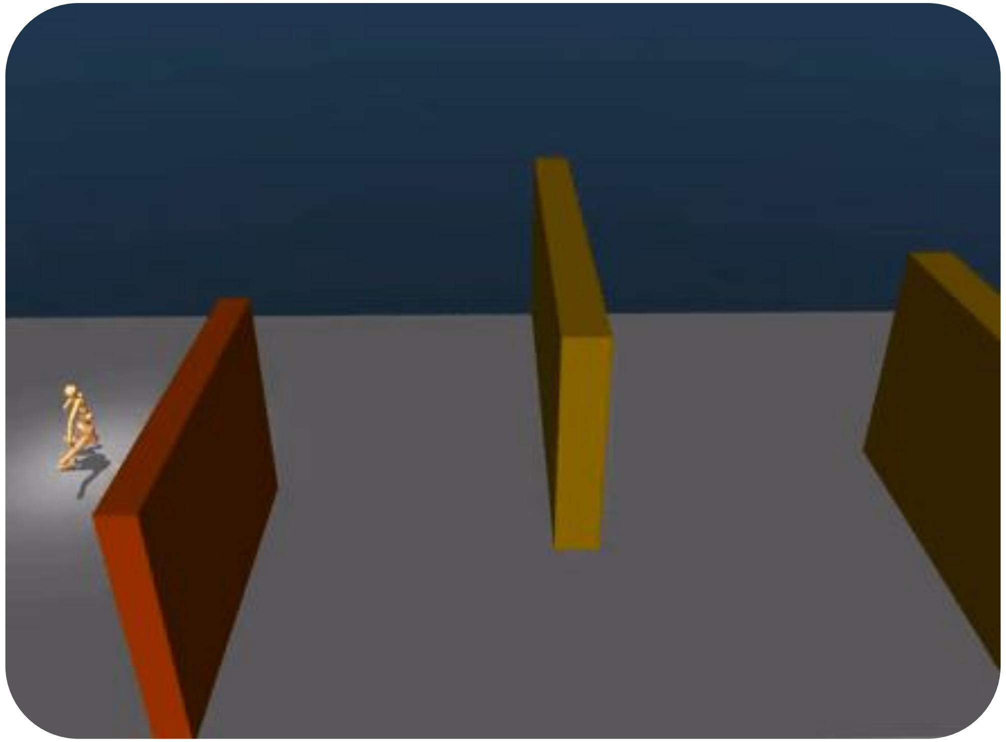


But if we update $Q(s,a)$ towards $r + \gamma \max_{a'} Q(s', a')$

we will have overestimated $Q(s,a)$ > **0** because

$$\mathrm{E}[\max_{a'} Q(s', a')] >= \max_{a'} E[Q(s', a')]$$

Вот теперь точно все



source: Emergence of Locomotion Behaviours in Rich Environments, DeepMind

[@Rads_ai](https://t.me/Rads_ai)
Канал Радослава с текстовыми разборами занятий

[@Young_and_Yandex](https://t.me/Young_and_Yandex)
Канал стажировок Яндекса