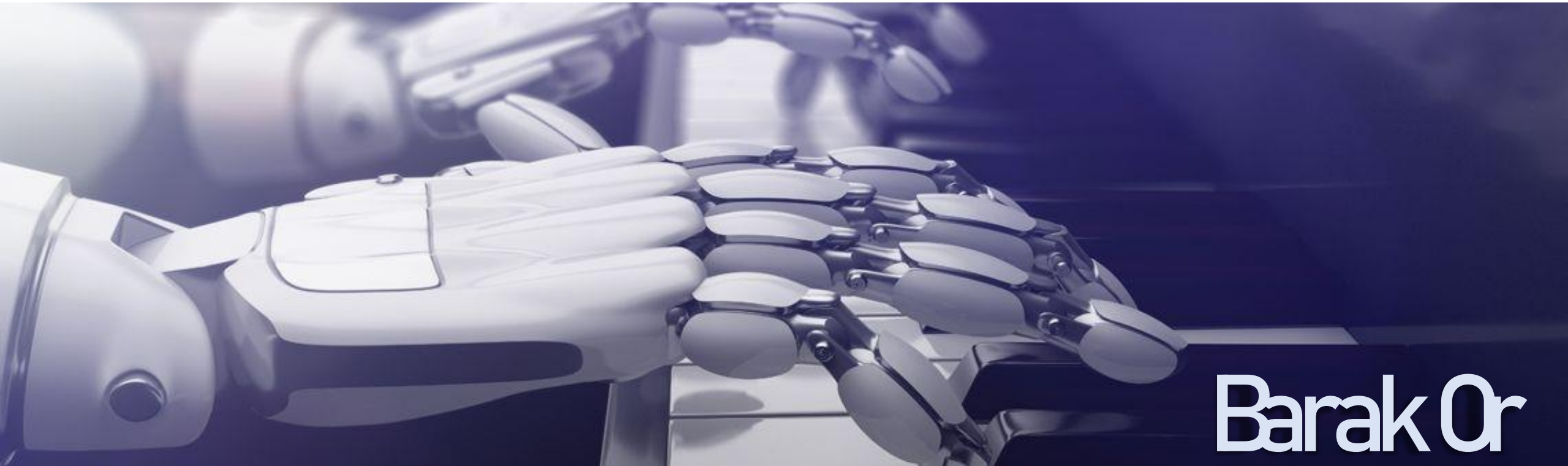
Value Based Methods in Deep Reinforcement Learning



Barak Or

Who am I ?

Algorithm Engineer

Algorithm Engineer

Lead of Navigation Project.

Physics Teacher

Teaching Assistant

Control, Navigation, System Engineering,

Intelligent structure, and more.

Center for Security
Technion - Israel

Gemunder Prize Winner, 2018

TECHNION
Israel Institute
of Technology

M.Sc. , B.Sc. Aerospace Engineering

Estimation and Optimal Control.

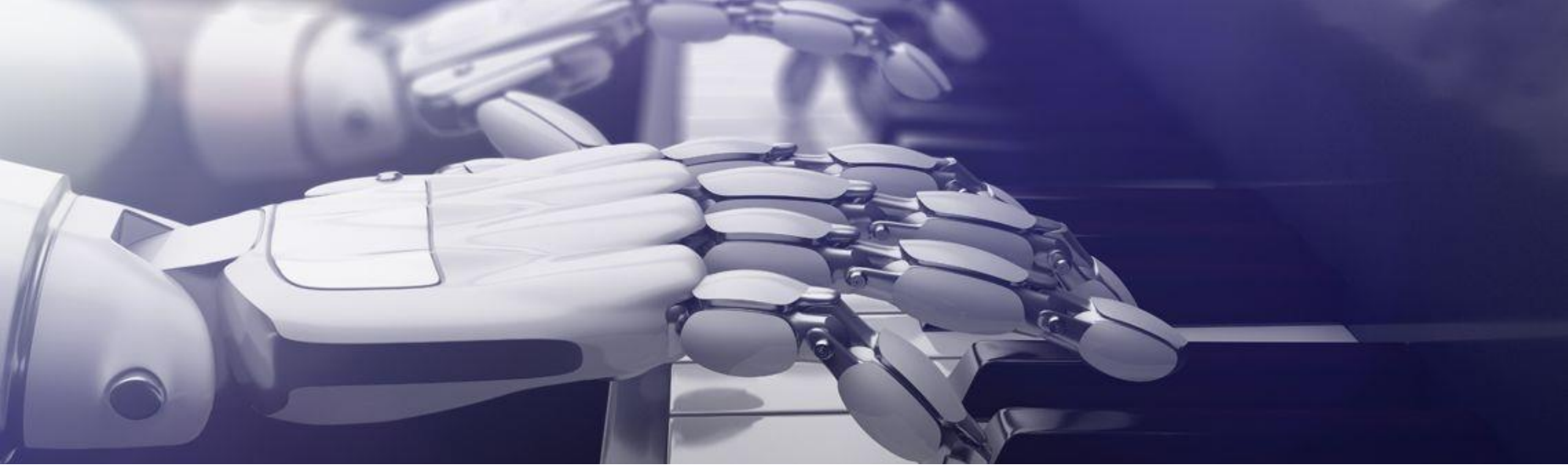
B.A Economics & Management

AskPeople
ייעוץ לכל שלב בחיים

Co-Founder

With my Brother Shahar Or

Contents



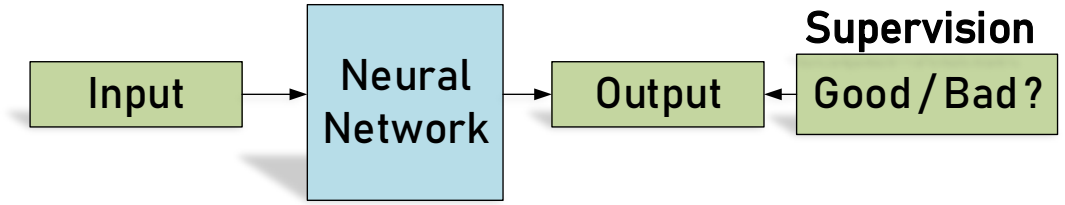
Background

Let's review the RL Framework...

-Supervised Learning: a learning system tries to learn a latent map based on labeled examples.

-Unsupervised Learning: a learning system tries to establish a model for data distribution based on unlabeled examples.

-Reinforcement Learning: a decision-making system is trained to make optimal decisions (experience).



Actually, all kind of learning are Supervised by a loss function. The sources of supervision must be defined by human.

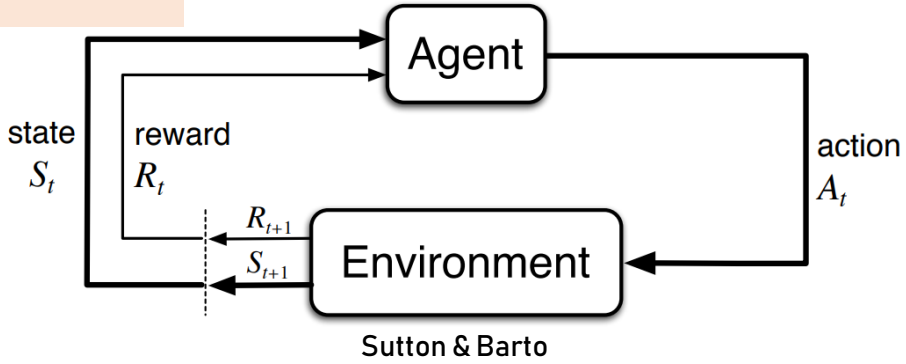
Let's review the RL Framework...

An **agent** acting in an **environment**.

At every point of time, the agent observes the **state** of the environment and decides on an **action** that changes the state.

For each such action, the agent is given a **reward** signal.

The agent's role is to maximize the total received reward.



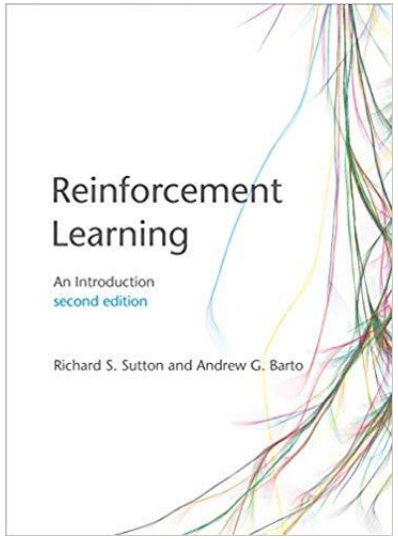
Sutton & Barto

Let's review the RL Framework...

- A framework for learning to solve **sequential decision making** problems by trial & error in a world that provides occasional rewards.

- This is the task of deciding, from experience, the sequence of actions to perform in an uncertain environment in order to achieve some goals.

- Inspired by behavioral psychology, reinforcement learning (RL) proposes a formal framework to this problem.




Richard S. Sutton


Andrew G. Barto

Let's review the RL Framework...

An artificial agent may learn by interacting with its environment.

Using the experience gathered, the artificial agent should be able to optimize some objectives given in the form of **cumulative rewards**.

-This approach applies in principle to any type of **sequential decision-making** problem relying on past experience.

-The environment may be stochastic, the agent may only observe partial information about the current state, etc.

Why Deep Reinforcement Learning ?

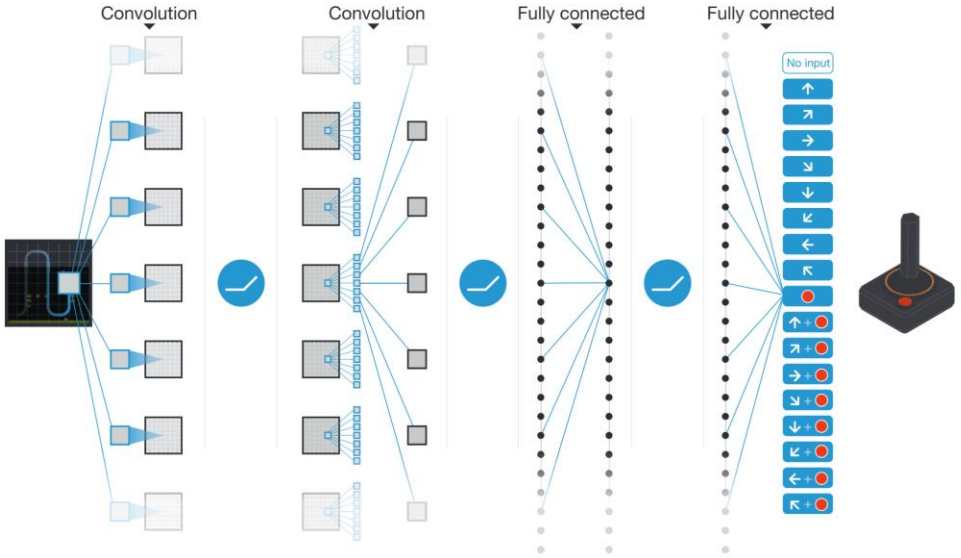
-Over the past few years, RL has become increasingly popular due to its success in addressing challenging sequential decision-making problems.

-Several of these achievements are due to the combination of RL with deep learning techniques.

-For instance, a deep RL agent can successfully learn from visual perceptual inputs made up of thousands of pixels (Mnih et al., 2015 / 2013).

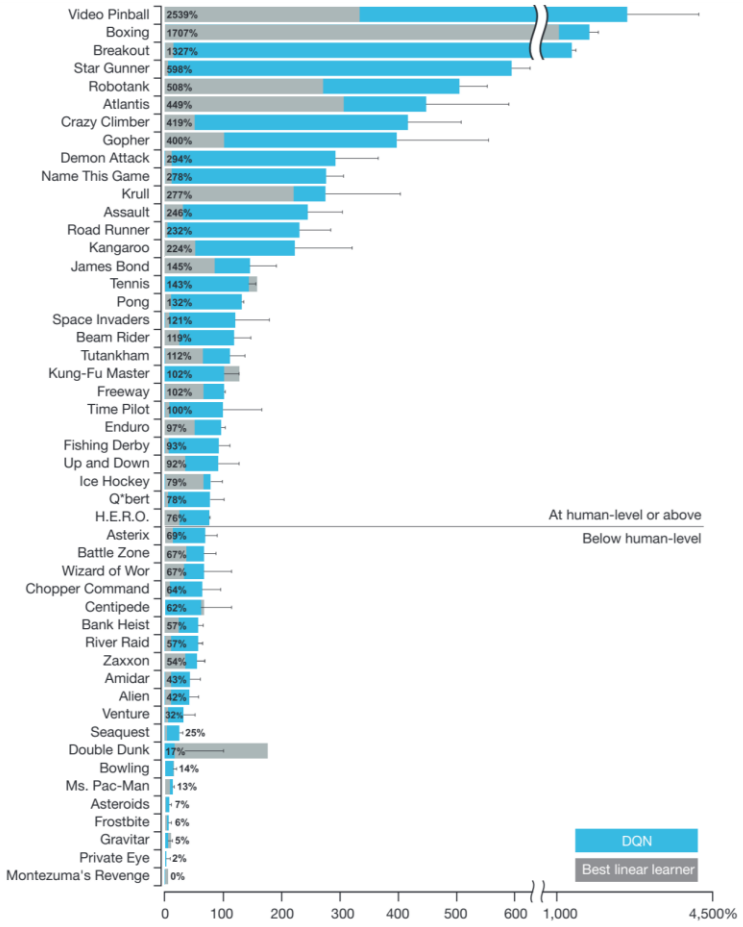
Why Deep Reinforcement Learning ?

Human-level control through deep reinforcement Learning, Volodymyr Mnih et al., 2015. on Nature.



Innovators Under 35

The first system to play Atari games as well as a human can.



Why Deep Reinforcement Learning ?

-This is all about a combination of RL and DL.

-It has been able to solve a wide range of complex decision-making tasks that were previously out of reach for a machine.

-Deep RL opens up many new applications in domains such as healthcare, robotics, finance, and more.

"One of the most existing field in AI. Its merging the power and the capabilities of deep neural networks to represent and comprehend the world with the ability to act and then understanding the world".

Lex Fridman, MIT

Why Deep Reinforcement Learning ?

"The promise of DL is to Convert raw data into meaning representation"

"The promise of DRL is going behind and building an agent that use that representation and acts to achieve success in this world"

Lex Fridman, MIT

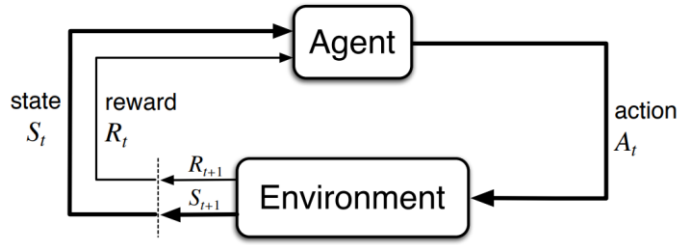
Value Based Method – Intro

Types of Reinforcement Learning

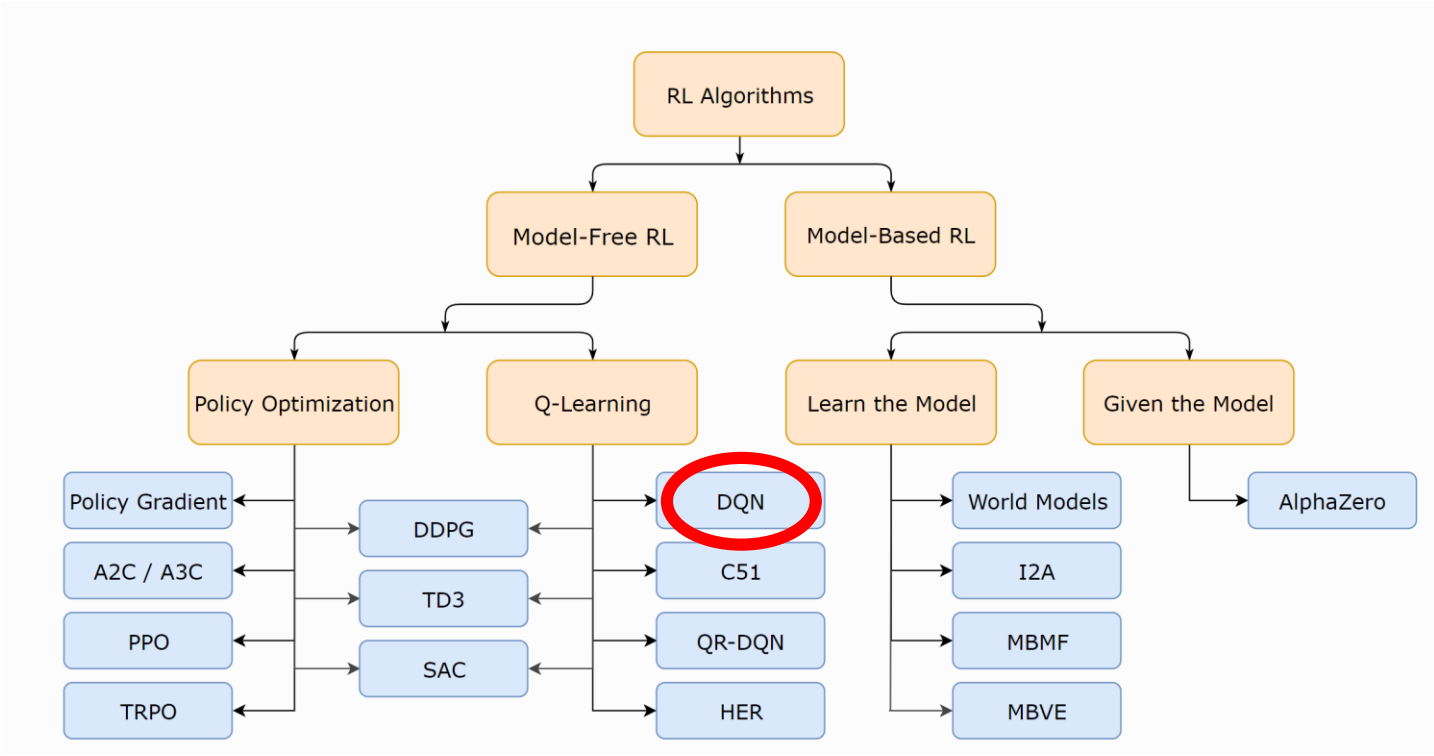
Value-Based (model free): learn the state or state-action value. Act by choosing the best action in state. Exploration is necessary. (Q learning)

Policy-Based **(model free)**: learn directly the stochastic policy function that maps state to action. Act by sampling policy.

Model-Based: learn the model of the world, then plan using the model. Update and re-plan the model often.



Types of Reinforcement Learning



OpenAI

Mathematical Background

Expected Return

An RL agent goal is to find a policy such that it optimizes the expected return (V-value function)

$$V^{\pi}(s) = E\left[\sum_{k=0}^{\infty} \gamma^{k} r_{t+k} \mid s_t = s, \pi\right]$$

The **optimal expected return** is defined as:

$$V^{*}(s) = \max_{\pi} V^{\pi}(s)$$

The optimal V-value function is the expected discounted reward when in a given state *s* the agent follows the policy π^{*} thereafter.

Mathematical Background

Q value

There are more functions of interest. One of them is the Quality Value function:

$$Q^{\pi}(s,a) = E\left[\sum_{k=0}^{\infty} \gamma^{k} r_{t+k} \mid s_t = s, a_t = a, \pi\right]$$

Similarly to the V-function, the optimal Q value is given by:

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

The optimal Q-value is the expected discounted return when in a given state s and for a given action *a* the agent follows the policy π^{*} thereafter.

Mathematical Background

The optimal policy can be obtained directly from this optimal value:

$$\pi^*(s) = \arg\max_a Q^\pi(s,a)$$

Mathematical Background

Advantage function

We can relate between the last two functions:

$$A^{\pi}\left(s,a\right)=Q^{\pi}\left(s,a\right)-V^{\pi}\left(s\right)$$

It describes “how good” the action *a* is, as compared to the expected return when following directly policy π.

Mathematical Background

Bellman Equation

-In order to learn the Q value, the Bellman equation is used. It promise a unique solution Q*:

$$Q^*(s,a) = \left(\mathcal{B}Q^*\right)(s,a)$$

-where $\mathcal{B}$ is the Bellman operator:

$$\mathcal{B}Q^*(s,a) = \sum_{s' \in \mathcal{S}} T(s,a,s')\left(R(s,a,s') + \gamma \max_{a'} Q^*(s',a')\right)$$

-In order to promise optimal value:

-state-action pairs are represented discretely

-all actions are repeatedly sampled in all states

Q Learning

-Q learning in an off-policy method, learns the value of taking an action in a state and from that learn Q value and choose how to act in the world.

-Define a state- action value function: an expected return when starting in *s*, performing *a*, and following π. Represented in a tabulated form.

According to Q learning, the agent uses any policy to estimate Q that maximizes the future reward. Q directly approximates Q*, when agent keep updating each state-action pair.

New state old state Reward estimate of Optimal future Q

$$Q^{\pi}_{t+1}(s_t, a_t) = (1-\alpha) Q^{\pi}_{t}(s_t, a_t) + \alpha \left(R_t + \gamma \max_a Q^{\pi}_{t}(s_{t+1}, a) \right)$$

learning rate discount factor

Chris Watkins

Q Learning

For non DL approaches, this Q function is just a table:

↑ *states* ↓

	a_1	a_2	a_3	a_4
s_1	10	52	15	-2
s_2	14	30	8	7
s_3	42	0	-5	-10
s_4	-3	-1	-7	-20

← *actions* →

In real-world scenario, value iteration is impractical…

In *Breakout game* the state are screen pixels:

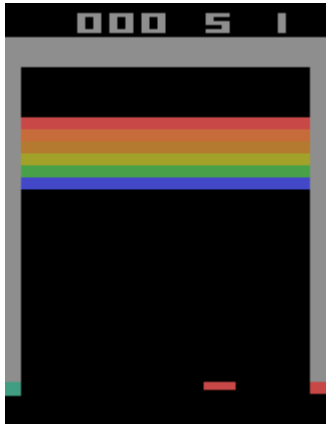
Image size: 84x84

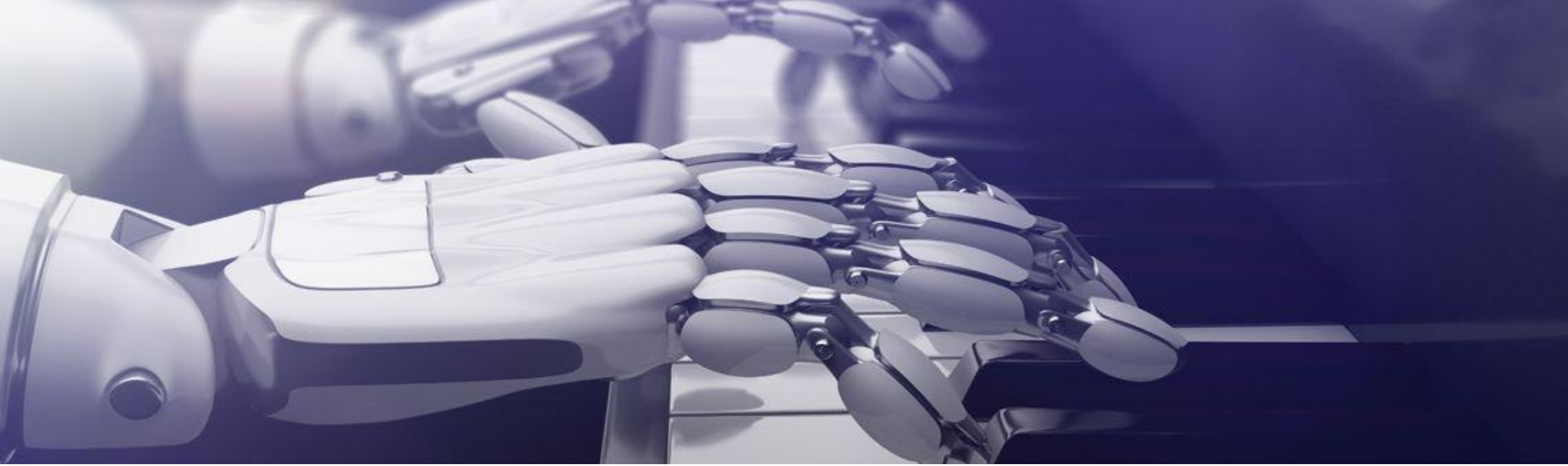
Consecutive: 4 images

Gray levels: 256

Number of rows in Q-table:

$$256^{84\times84\times4} \approx 10^{69,970}$$





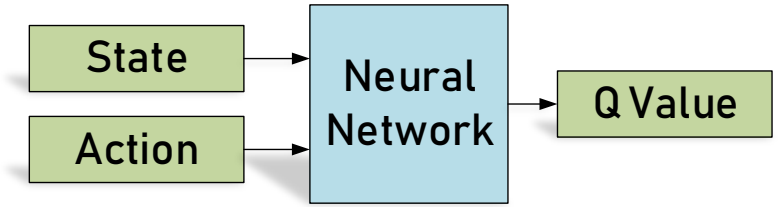
Deep RL

DQN: Deep Q-Networks

Use a neural network to approximate the Q function :

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

Neural network is good as a function approximator.



DQN: Deep Q-Networks

The loss function has two Qs function:

$$\mathcal{L} = E\left[\left(r + \gamma \max_{a'} Q(s', a'; \theta_k) - Q(s, a; \theta_k)\right)^2\right]$$

Target Prediction

Target: the predicted Q value of taking an action in particular state.

Prediction: the value that you get when actually taking that action (calculating the value on the next step and choosing the one that minimize the total loss).

DQN: Deep Q-Networks

Parameter updating:

$$\theta_{k+1} = \theta_k + \alpha\left(r + \gamma \max_{a'} Q(s', a'; \theta_k) - Q(s, a; \theta_k)\right)\nabla_{\theta_k} Q(s, a; \theta_k)$$

Target Prediction

-When updating the weights, one also changes the target.

-Due to the approximation by neural networks, large errors are built in the state-action space. Hence Bellman equation is not converges w.p. 1.

-Errors may propagate with this update rule (slow / unstable/ etc.)

DQN: Deep Q-Networks

DQN algorithm is able to obtain strong performance in an online setting for a variety of ATARI games, directly learns from **pixels**.

Two heuristics to limit the instabilities:

1. The parameters of target Q-network are updated only every N iterations. This prevents the instabilities to propagate quickly and by that it minimize the risk of divergence.

2. The experience replay memory trick is used (see next slide).

DQN Trick: Experience Replay

-in DQN a CNN architecture is used. The approximation of Q-values using non-linear functions is not (always) stable.

-according to experience replay trick: all experiences are stored in a replay memory.

-when training the network, random samples from the replay memory are used instead of the most recent action to proceed.

-In different words: the agent collects memories \ stores experience (state transitions, actions and rewards) and creates mini-batches for the training.

Lin 1992

DQN Trick: Epsilon Greedy Exploration

-As the Q function converges to Q*, it actually settles with the first effective strategy it finds. Hence, exploration is greedy.

-An effective way to explore is by choosing a random action with probability “epsilon” and other wise (1-epsilon), use the greedy action (with highest Q value).

DDQN: Double Deep Q-Networks

-The max operation in Q-learning uses the same values both to select and evaluate an action.

-It makes it more likely to select overestimated values (in case of noise or inaccuracies) , resulting in overoptimistic value estimates.

-In DDQN: Separate network for each Q, hence there are two neural networks.

-Helps reduce bias.

-The policy is still chosen according to the values obtained by the current weights.

$$\mathcal{L} = E\left[\left(r + \gamma \max_{a'} Q^{A}\left(s', a'\right) - Q^{B}\left(s, a\right)\right)^{2}\right]$$

DDQN: Double Deep Q-Networks

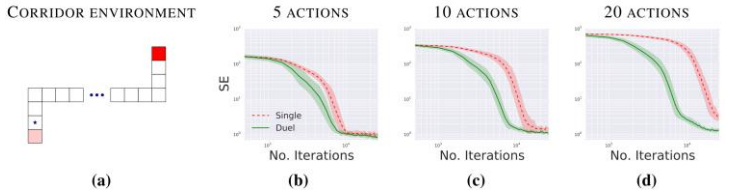
-In DDQN: Separate network for each Q, hence there are two neural networks.

$$Q_{t+1}^{A}\left(s_t,a_t\right)=\left(1-\alpha\right)Q_t^{A}\left(s_t,a_t\right)+\alpha\left(R_t+\gamma Q_t^{B}\left(s_{t+1},\arg\max_a Q_t^{A}\left(s_{t+1},a\right)\right)\right)$$

$$Q_{t+1}^{B}\left(s_t,a_t\right)=\left(1-\alpha\right)Q_t^{A}\left(s_t,a_t\right)+\alpha\left(R_t+\gamma Q_t^{B}\left(s_{t+1},\arg\max_a Q_t^{A}\left(s_{t+1},a\right)\right)\right)$$

$$\mathcal{L}=E\left[\left(r+\gamma\max_{a'}Q^{A}\left(s',a'\right)-Q^{B}\left(s,a\right)\right)^2\right]$$

Dueling Deep Q-Networks

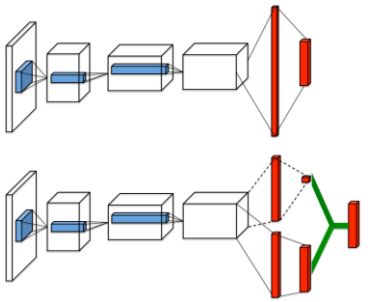


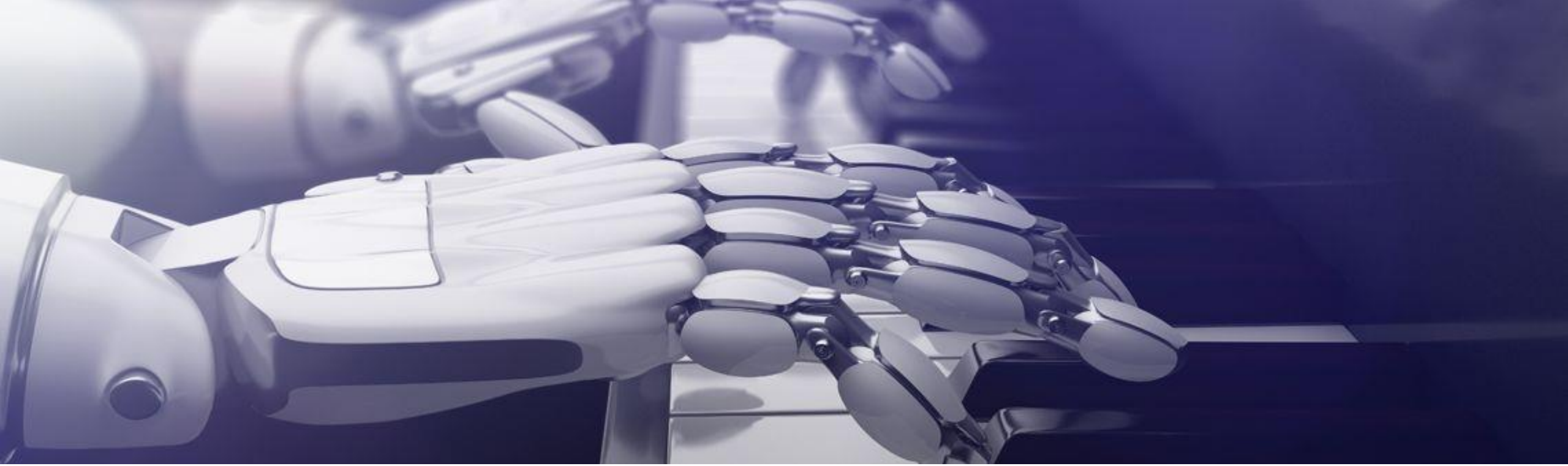
-Q contains **advantage** (A) value additional to the value (V) of being in that state. A is defined earlier as the advantage of taking action *a* in state *s* among all other possible actions and states.

If all the actions you aim to take are “pretty good”, we want to know: how better it is.

$$Q(s,a) = A(s,a) + V(s)$$

-dueling network represents two separate estimators: one for the state value function and one for the state-dependent action advantage function





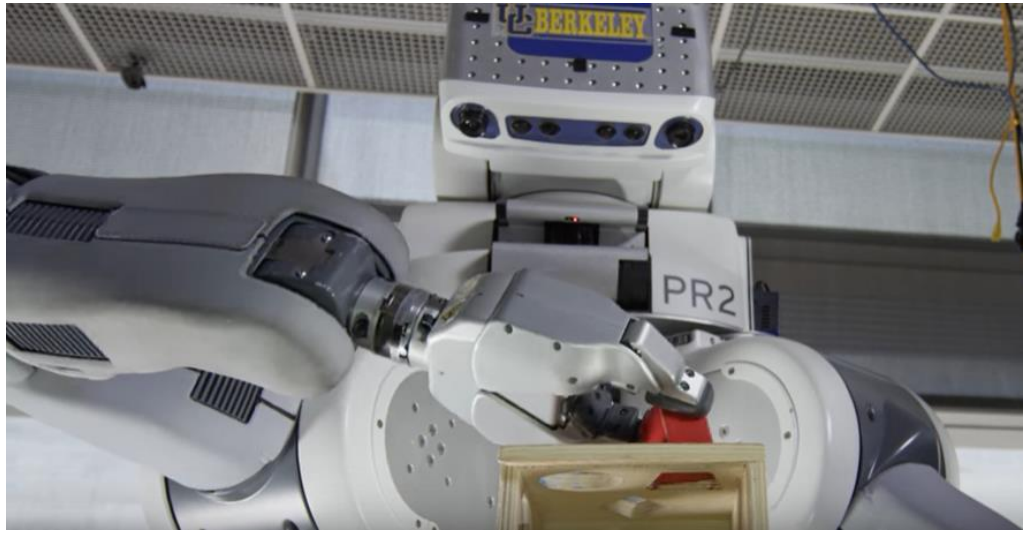
Applications

Deep Traffic – DQN for Traffic Navigation

selfdrivingcars

Applications

Pieter Abbeel: Accelerated computing.



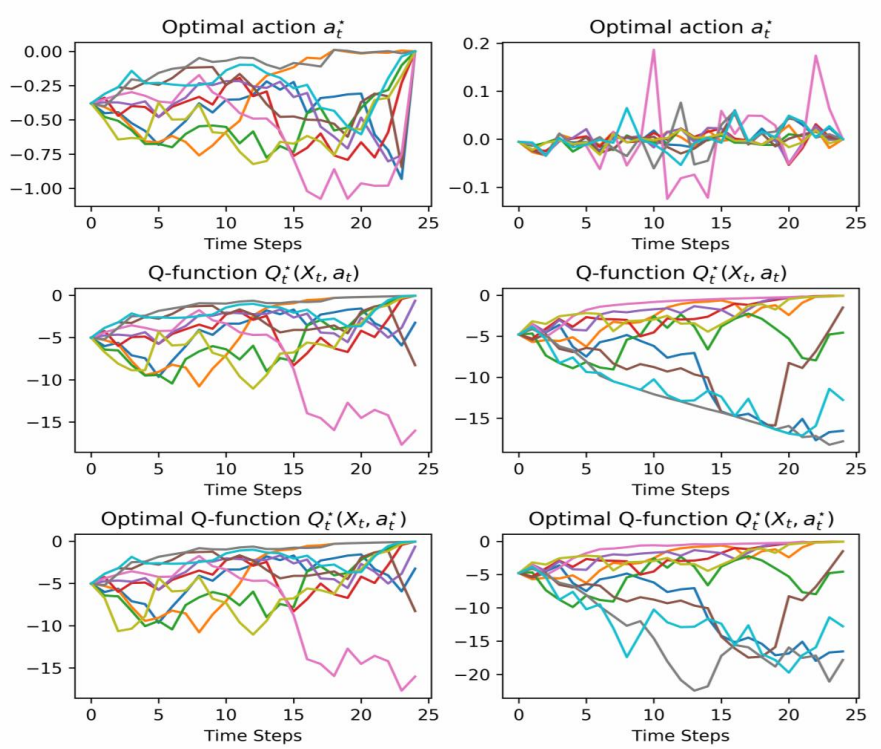
Applications

-DQN in finance:

I.Halperin from NYU lastly (September 2019, arxiv) suggested an optimal option portfolio based on Fitted Q learning.

It gets better results than the famous Black-Shoules model !

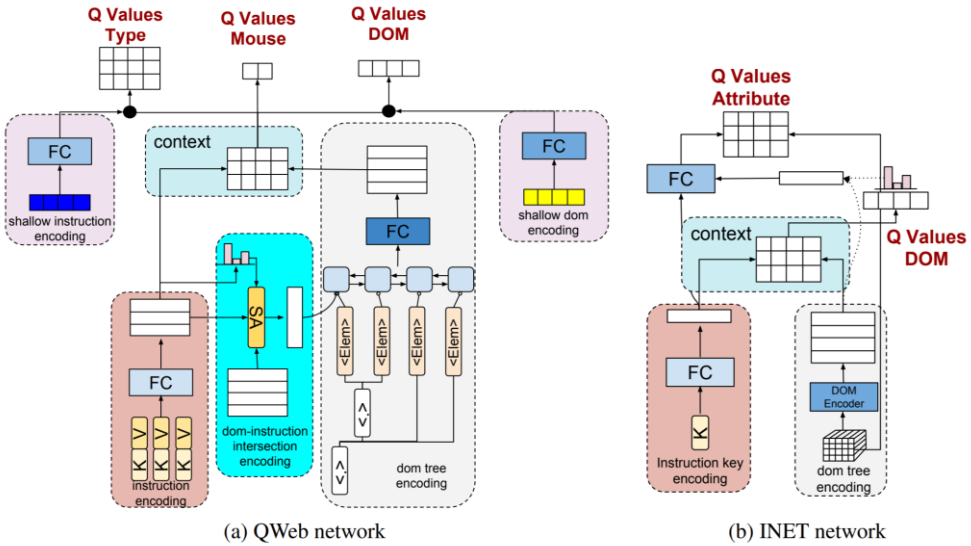
QLBS: Q-learner in the Black-Scholes (-Merton) Worlds.



Applications

-Qweb: Learning to Navigation the Web (ICLR 2019)

Training reinforcement learning agents to navigate the Web (navigator agent) by following certain instructions, such as **book a flight ticket.**



(a) QWeb network (b) INET network

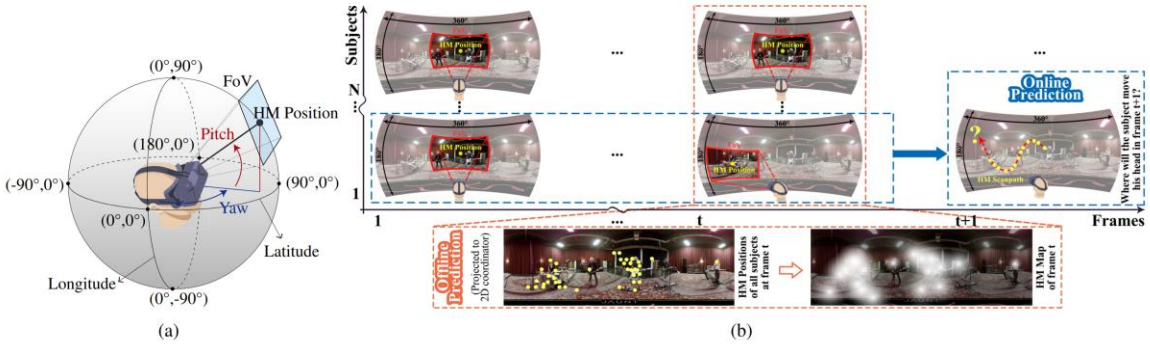
Some Applications…

-DQN in computer vision:

Predicting Head Movement in Panoramic Video: A Deep Reinforcement Learning Approach.

Xu et al. 2015, Arxiv.

Deep reinforcement learning (DRL) can be applied to predict head movement positions.



Some Applications…

-DQN in Autonomous Car

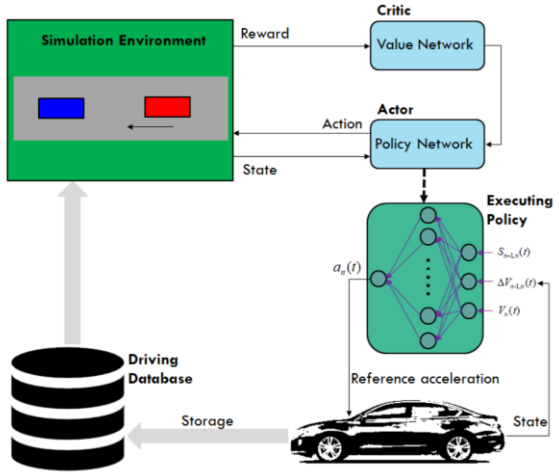
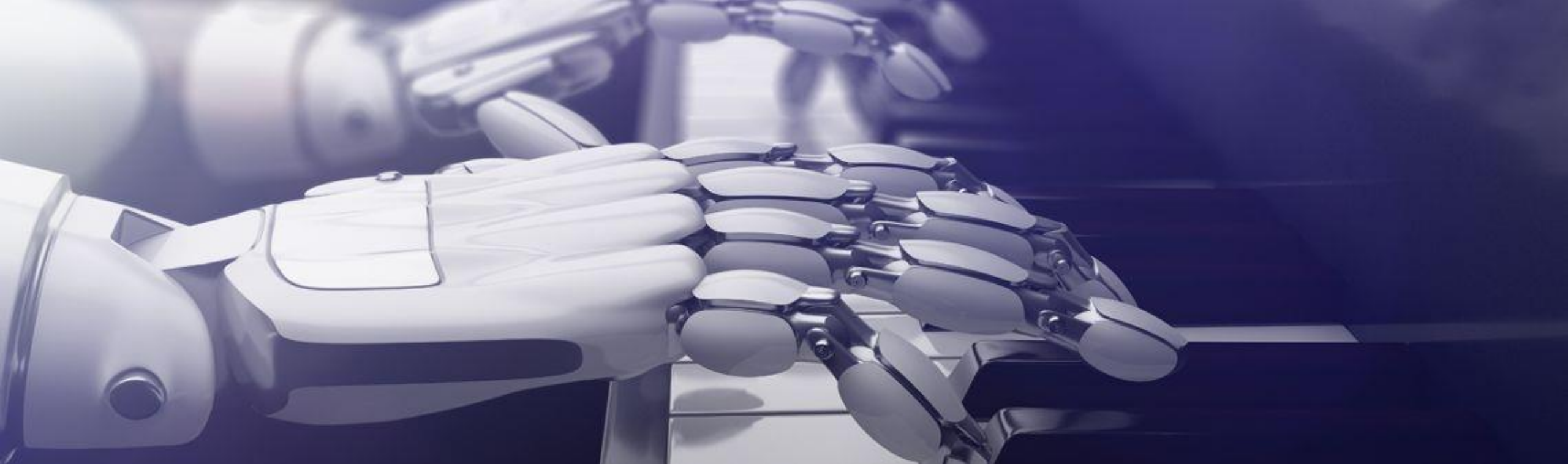


FIGURE 1 Conceptual diagram of human-like car following by deep RL.

Human-Like Autonomous Car-Following Model with Deep Reinforcement Learning. Zhu el al. 2018, Arxiv. a framework for human-like autonomous car-following planning based on DRL. Demonstrates that RL methodology can offer **insight into driver behavior and can contribute to the development of human-like** autonomous driving algorithms and **traffic-flow models.**



Summary

Keep Learning

barakorr@gmail.com
barakor
www.barakor.com

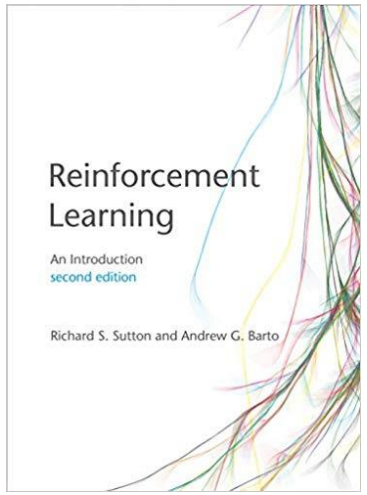
David silver

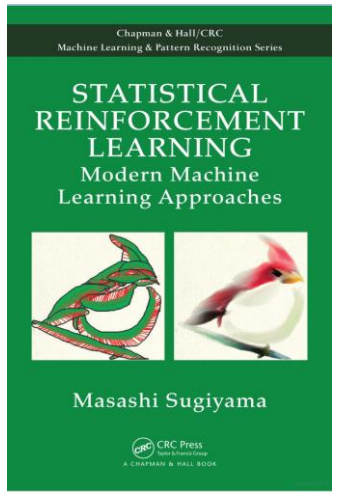
coursera

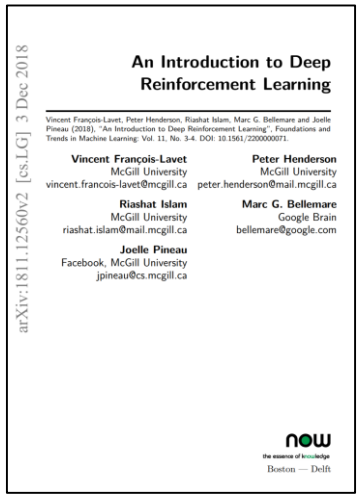
Practical Reinforcement Learning

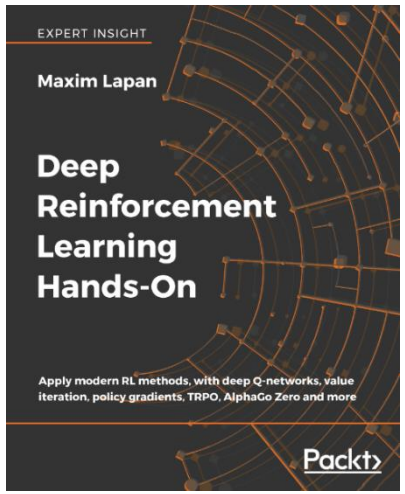
UNIVERSITY OF ALBERTA Reinforcement Learning Specialization

NYU TANDON SCHOOL OF ENGINEERING Machine Learning and Reinforcement Learning in Finance Specialization



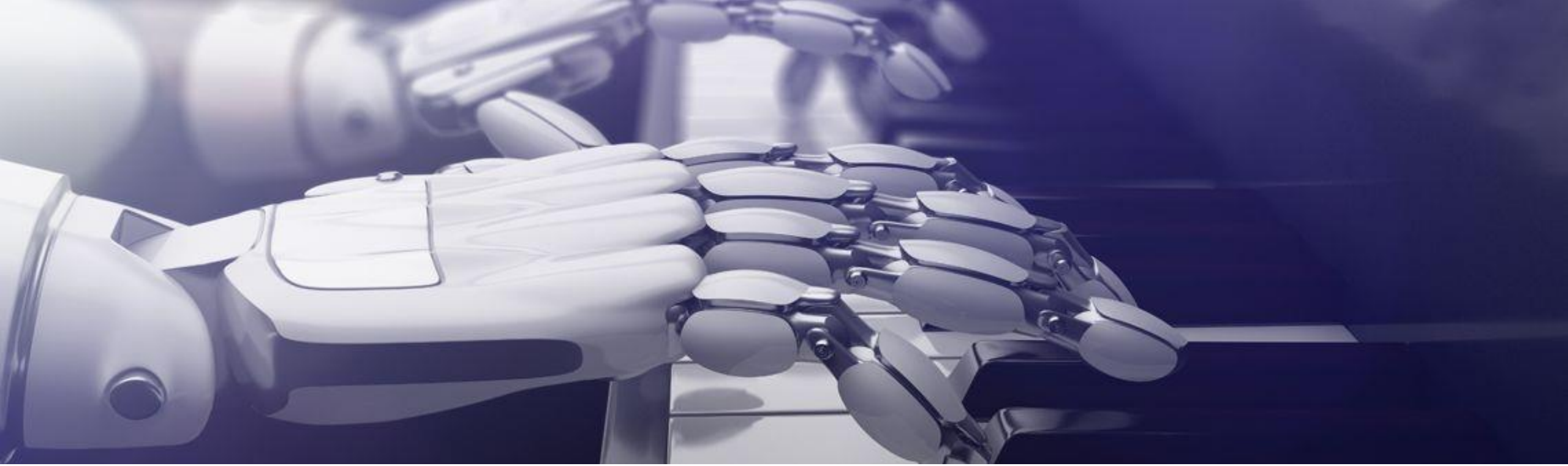






Deep Reinforcement Learning Artificial Intelligence (AI) Podcast. Dec 16, 2018.
Pieter Abbeel

Deep Reinforcement Learning in the Real World. Nov 8, 2019.
Sergey Levine



Thank you!

barakorr@gmail.com
barakor