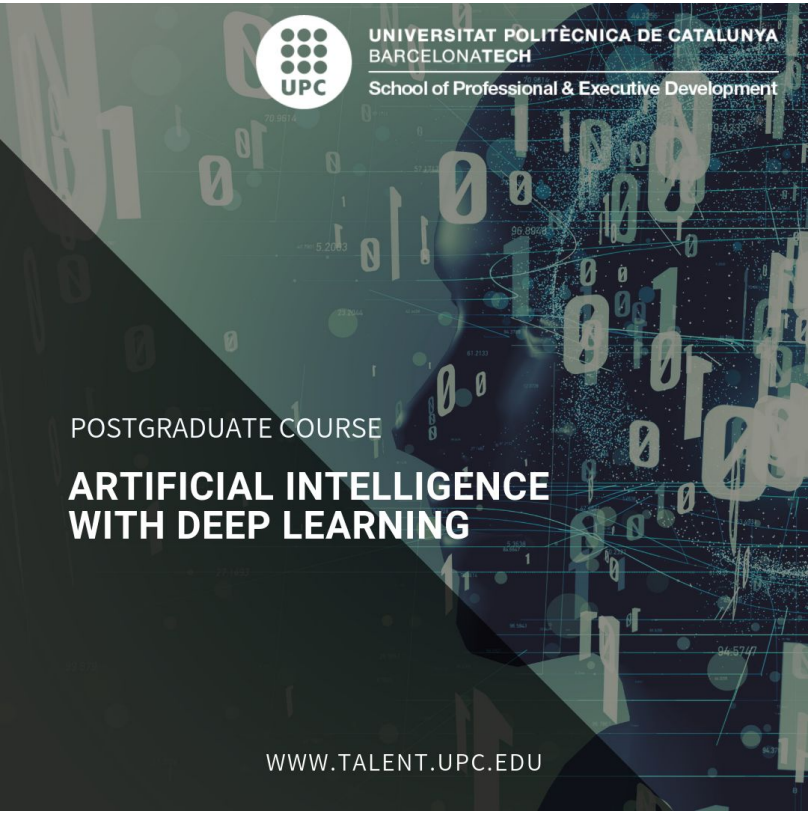


#DLUPC

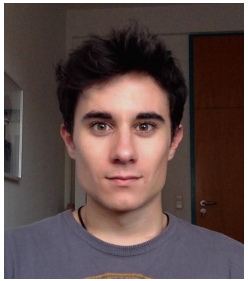
Applications 2

Reinforcement Learning

Xavier Giro-i-Nieto
xavier.giro@upc.edu

Associate Professor
Universitat Politècnica de Catalunya
Barcelona Supercomputing Center

Acknowledgements



Víctor Campos

victor.campos@bsc.es

PhD Candidate

Barcelona Supercomputing Center

Míriam Bellver

miriam.bellver@bsc.edu

PhD Candidate

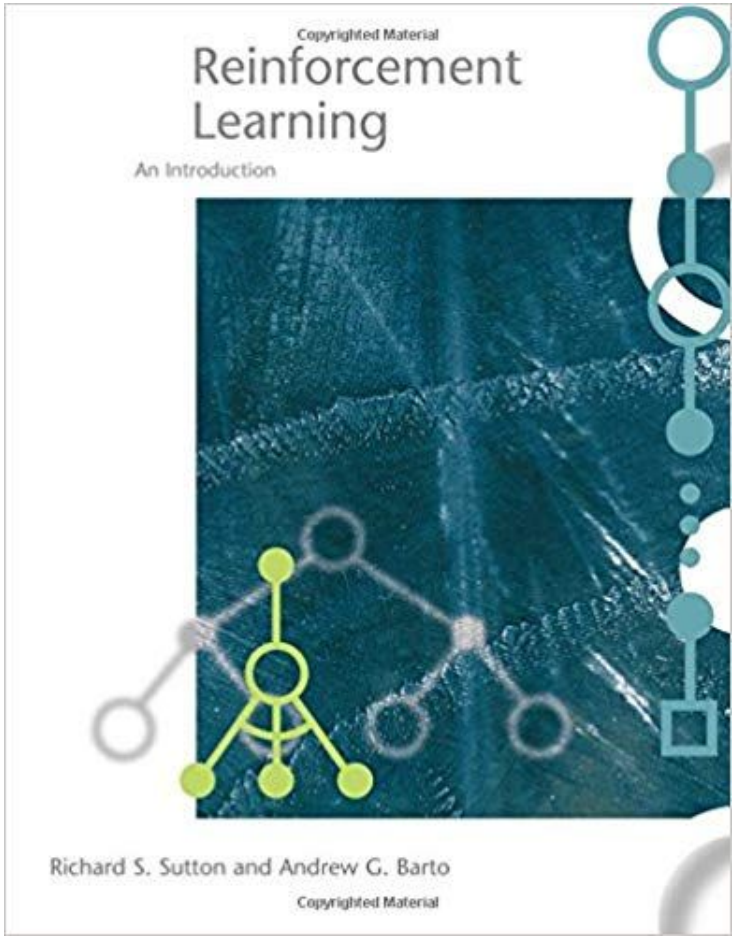
Barcelona Supercomputing Center

Acknowledgements

OpenAI

Lilian Weng, “A (Long) Peek into Reinforcement Learning” (2018)

Bibliography



Temporal-Difference Learning

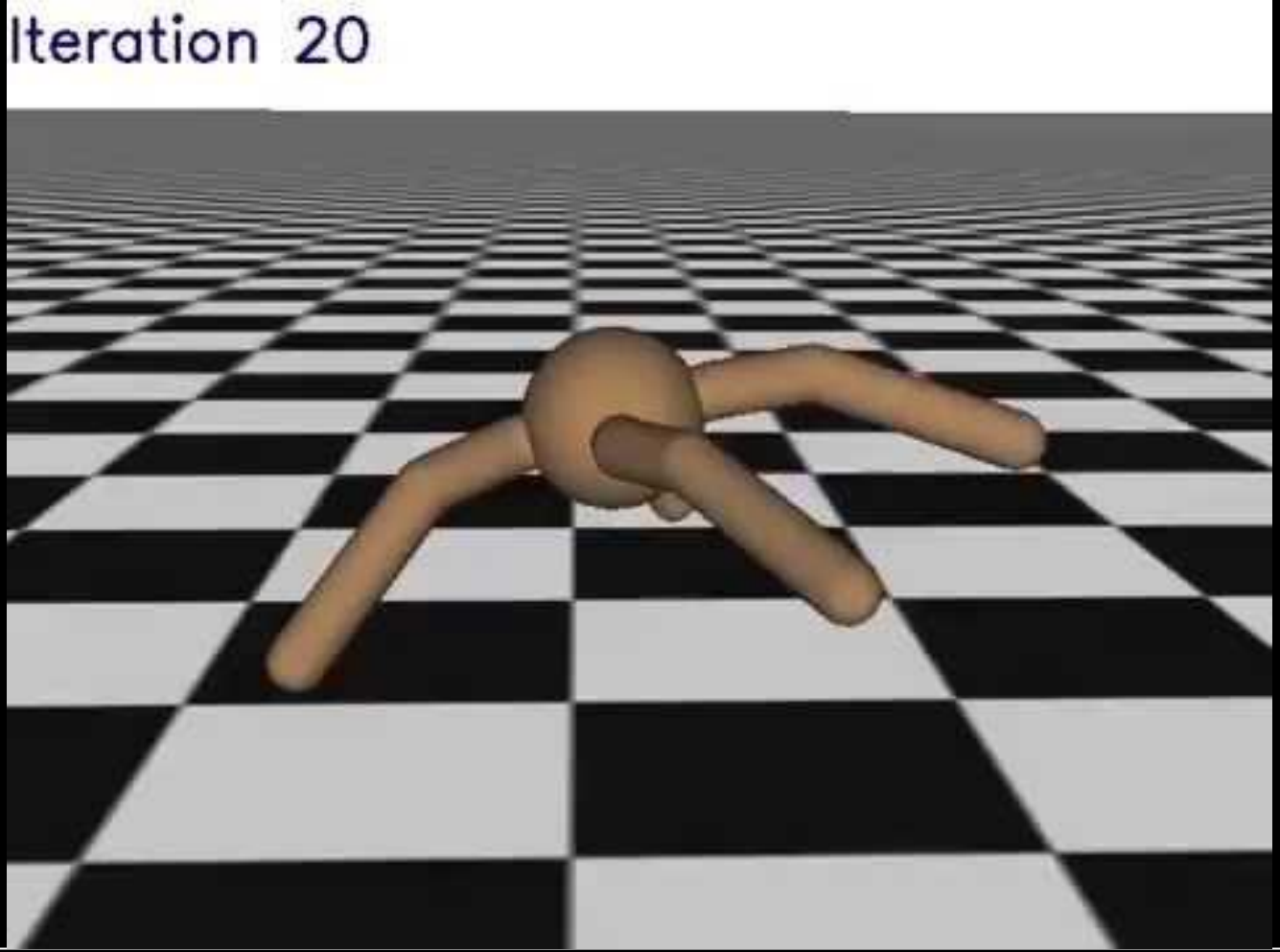
Rich Sutton
Reinforcement Learning & Artificial Intelligence Laboratory
Alberta Machine Intelligence Institute
Dept. of Computing Science, University of Alberta
Canada

Richard Sutton, "Temporal-Difference Learning" University of Alberta (2017). [YouTube]

Outline

1. **Control tasks with reinforcement learning**
2. Markov Decision Processes
3. Learning action-value functions Q
4. Learning policies

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. "Playing atari with deep reinforcement learning." NIPS Deep Learning Workshop (2013).



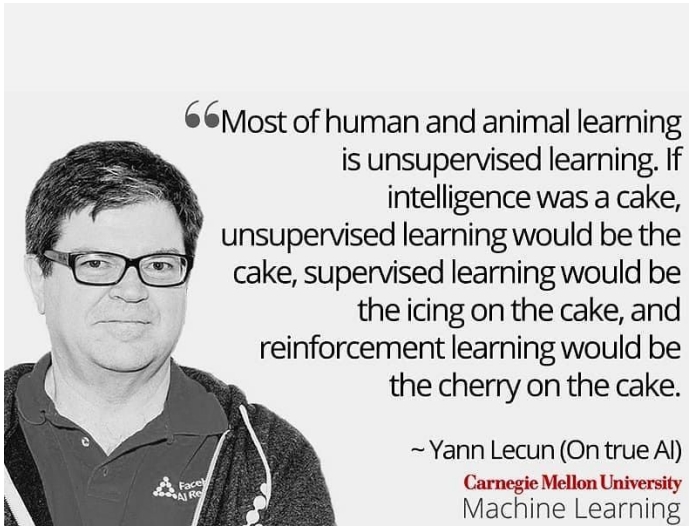
Schulman, John, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. "High-dimensional continuous control using generalized advantage estimation." ICLR 2016 [project page]

Wayve, “The first example of deep reinforcement learning on-board an autonomous car” (2018)

Types of machine learning

Yann Lecun's Black Forest cake

- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Unsupervised/Predictive Learning (cake)**
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**



- **(Yes, I know, this picture is slightly offensive to RL folks. But I'll make it up)**

Official ACM
@TheOfficialACM

Segueix

Yoshua Bengio, Geoffrey Hinton and Yann LeCun, the fathers of #DeepLearning, receive the 2018 #ACMTuringAward for conceptual and engineering breakthroughs that have made deep neural networks a critical component of computing today.
bit.ly/2HVJtdV

Tradueix el tuit

11:01 - 27 de març de 2019

Le Cake


EXIT
CILVR


Supervised Learning
TURING AWARD
CILVR
Congrats from

Our vision

Le Cake

A Pieter Abbeel i 7 més els agrada

Yann LeCun
@ylecun

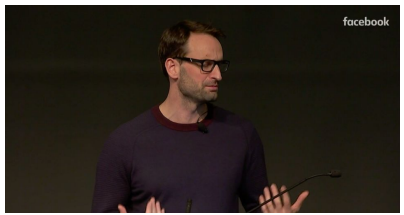
Pierre Pieter Abbeel gave a seminar at NYU today.
He showed up at the Turing reception and was given the RL piece of "Le Cake" (and lots of cherries).

Tradueix el tuit

1:10 · 13/4/19 · Facebook

A broader picture of types of learning…

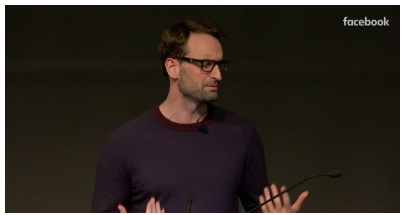
	…with a teacher	…without a teacher
Active agent…	Reinforcement learning (with extrinsic reward)	Intrinsic motivation / Exploration.
Passive agent…	Supervised learning	Unsupervised learning



Slide inspired by Alex Graves (Deepmind) at "Unsupervised Learning Tutorial" @ NeurIPS 2018.

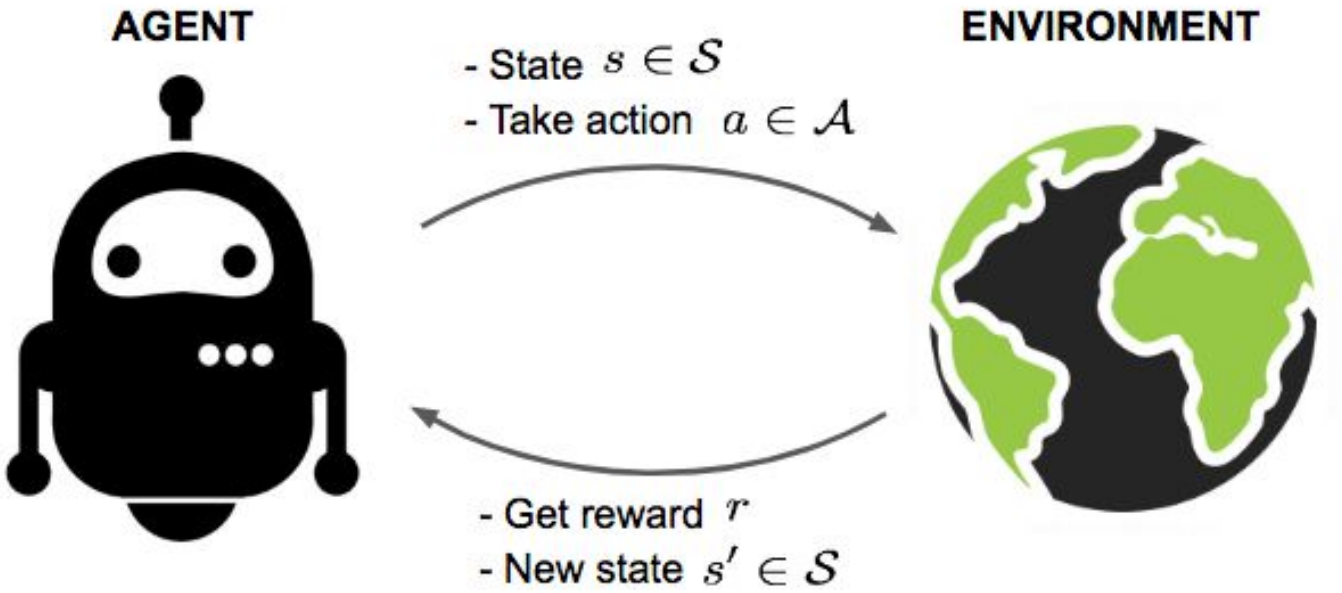
A broader picture of types of learning...

	...with a teacher	...without a teacher
Active agent...	Reinforcement learning (with extrinsic reward)	Intrinsic motivation / Exploration.
Passive agent...	Supervised learning	Unsupervised learning



Slide inspired by Alex Graves (Deepmind) at "Unsupervised Learning Tutorial" @ NeurIPS 2018.

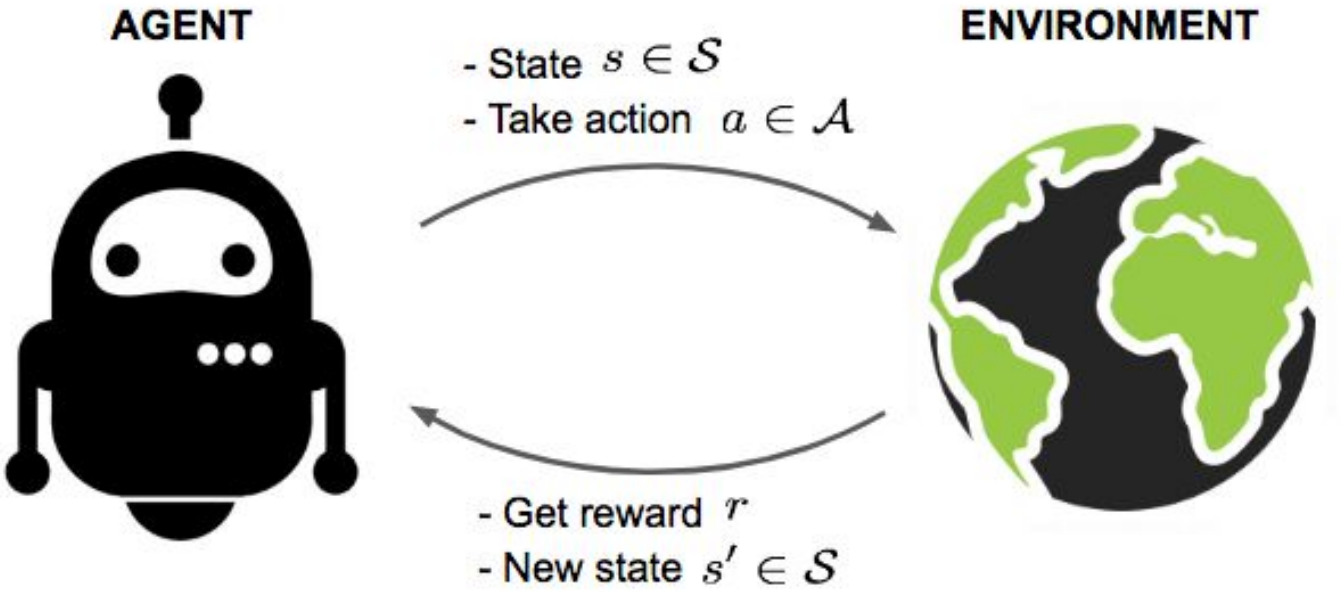
Reinforcement Learning (RL)



Lilian Weng, "A (Long) Peek into Reinforcement Learning" (2018)

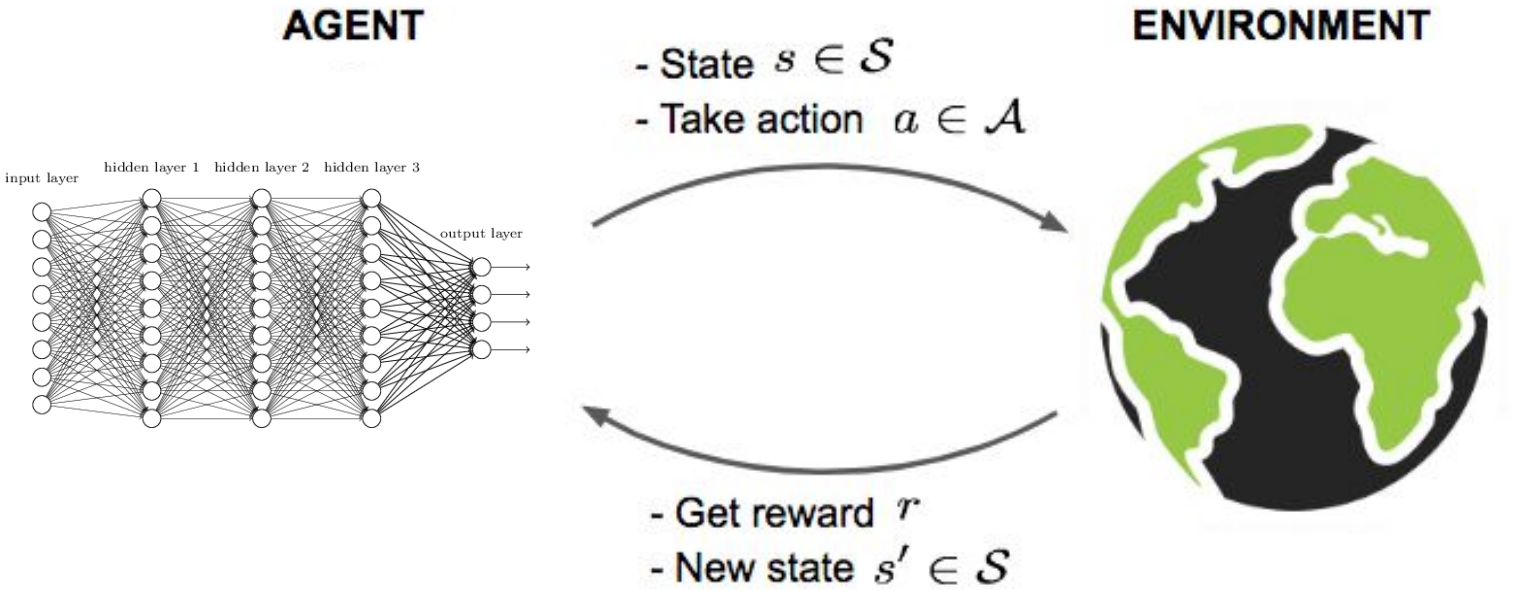
Reinforcement Learning (RL)

The **Policy** π is a function S → A that specifies which action the agent will take given a state.



Deep Reinforcement Learning (RL)

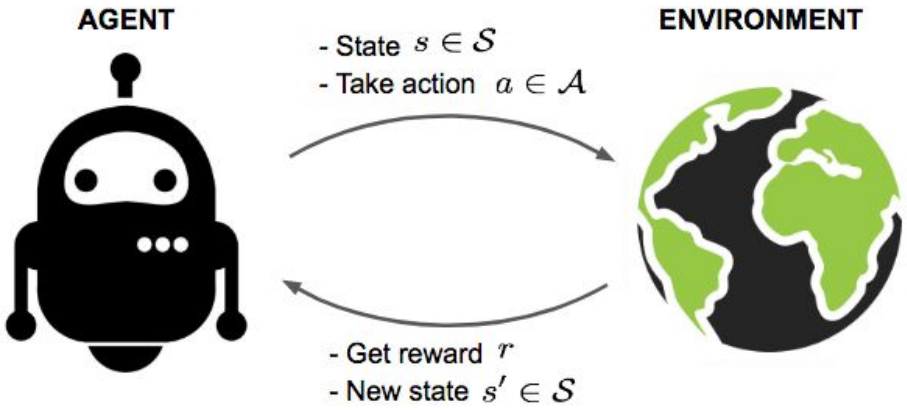
The **Policy** π can be governed through a deep neural network.



Training data by computation

In RL, training data is obtained by computing interaction sequences of:

State , Action , Reward



A complete episode consists of T interactions:

$S_1, A_1, R_2, S_2, A_2, \ldots, S_T$

One experience corresponds to a single tuple within an episode:

(s,a,r,s')

Lilian Weng, "A (Long) Peek into Reinforcement Learning" (2018)

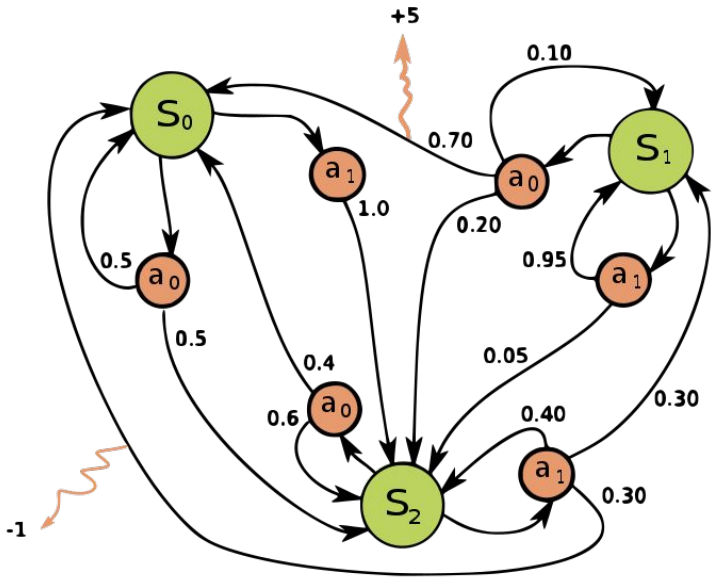
Outline

1. Control tasks with reinforcement learning
2. **Markov Decision Processes**
3. Learning action-value functions Q
4. Learning policies

Markov Decision Processes (MDP)

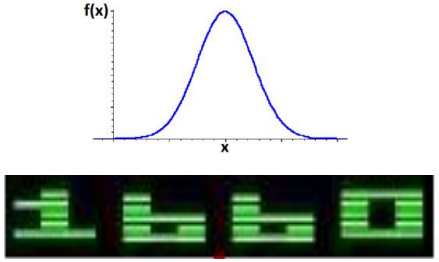
Markov Decision Processes provide a formalism for reinforcement learning problems.

Markov property:
Current state completely characterises the state of the world.

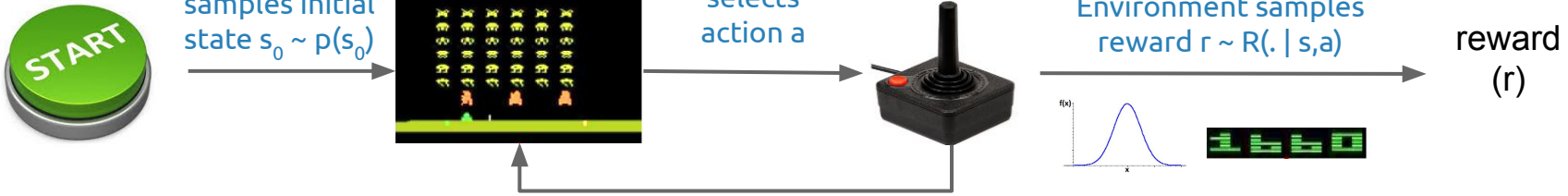


Slide concept: Serena Yeung, "Deep Reinforcement Learning". Stanford University CS231n, 2017.

Markov Decision Process (MDP)



S A R P γ

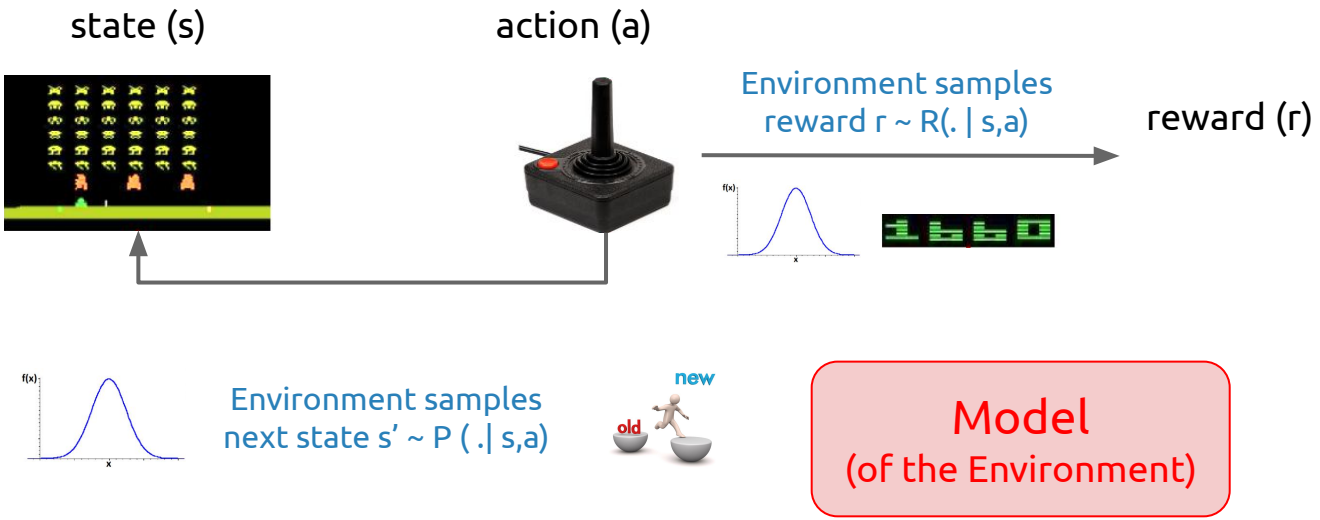


Slide concept: Serena Yeung, "Deep Reinforcement Learning". Stanford University CS231n, 2017.

MDP: Model (of the Environment)

The **Model (of the Environment)** is defined by:

- P(s'|s,a): State-transition function
- R(s,a): Reward function.



Model (of the Environment)

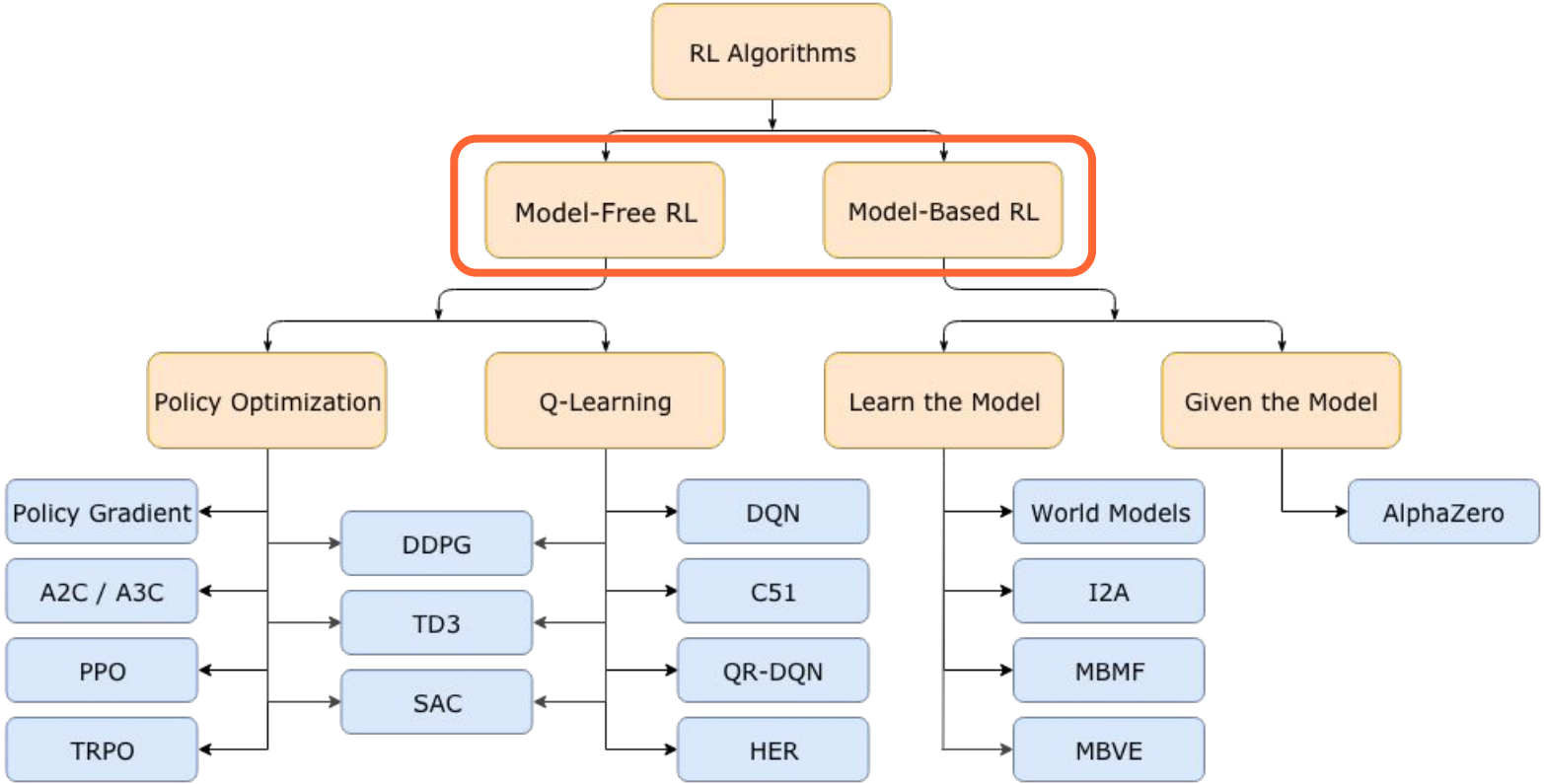


Figure: OpenAI Spinning Up

Model (of the Environment)

- **Model-based RL:** R(·| s,a) and P(·| s,a) are known, so an optimal solution can be found with Dynamic Programming.

  More details at Sergey Levine's slides @ Berkeley.

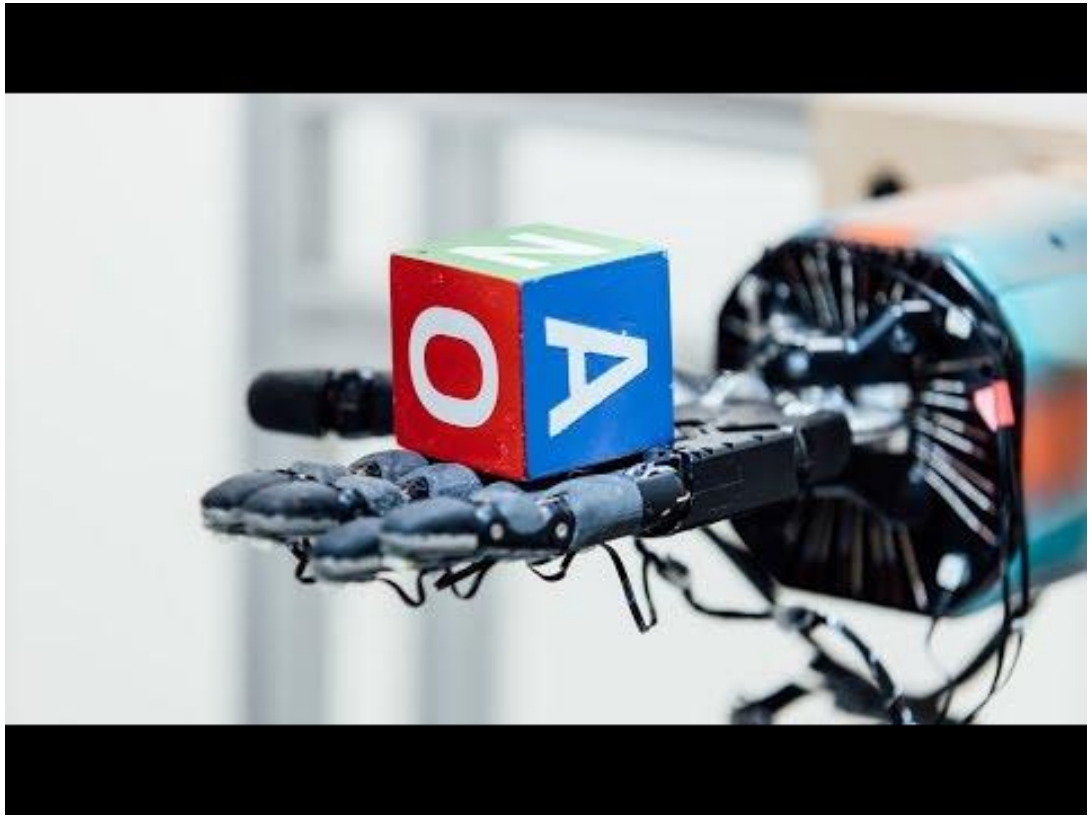
- **Model-free RL:** there is no prior knowledge of the world. The agent needs to learn all dynamics from scratch, resulting in poor data efficiency which limits its application to real-world agents.

  Requires sim2real transfer for real deployment.

Wayve, “Sim2Real: Learning to Drive from Simulation without Real World Labels” (2018)

Model (of the Environment)

Example of sim2real: Dexterity (OpenAI 2018)

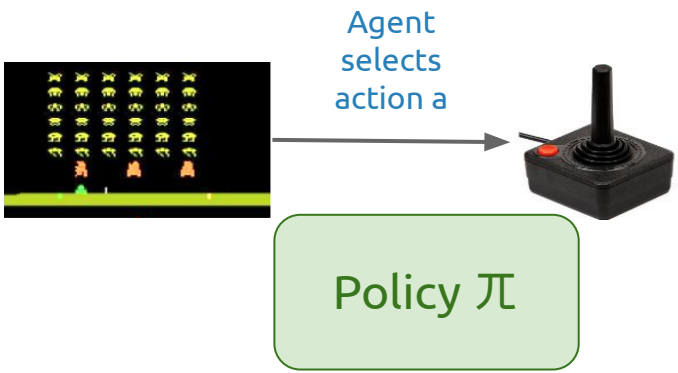


OpenAI

On- vs Off-Policy Learning

The **Policy** π is a function S → A that specifies which action to take given a state.

- **On-policy learning:** the agent is trained with a sequence of interactions of the *target policy*.
- **Off-policy learning:** the agent is trained with a sequence of interactions obtained from a *behaviour policy* different from the *target policy*.



Slide concept: Serena Yeung, "Deep Reinforcement Learning". Stanford University CS231n, 2017.

Policy: Deterministic vs Stochastic

The **Policy** π is a function $S \rightarrow A$ that specifies which action to take in each state.

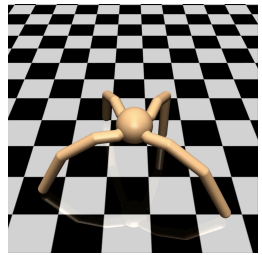
- **Deterministic:** $\pi(s)=a$
- **Stochastic:** $\pi(a|s)=\mathbb{P}_{\pi}[A=a|S=s]$

Policy: Discrete vs Continuous

The **Policy** π is a function S → A that specifies which action to take in each state.

- **Discrete actions:** categorical distribution over actions.
  - In the deterministic case: take the argmax action.
  - In the stochastic case: sample from the categorical distribution.

- **Continuous actions:** Gaussian distribution over actions; i.e. the policy generally outputs the mean and std per dimension.
  - In the deterministic case: take the mean.
  - In the stochastic case: sample from a normal distribution.



Slide concept: Víctor Campos

Return G_t

The future reward, also known as **return G_t**, is a total sum of discounted rewards from *t* and onwards in time...

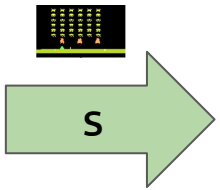
Return G_t

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

...where **γ** is the discount factor between [0,1].

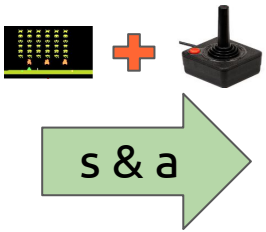
Value Functions $V_\pi(s)$ & $Q_\pi(s,a)$

The **value function** of a policy π is the expected return G_t given the state (V) or the state & action (Q):



State-value function V

$$V_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$



Action-value function Q

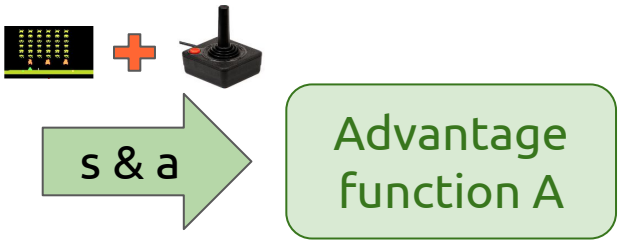
$$Q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

"Expected return…

…by following policy π starting from state s & action a"

Advantage Function $A_\pi(s,a)$

The **Advantage function (A-value)** of a policy π is the defined as:



$$A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$$

$A_\pi(s,a)$ expresses how good/bad an action is with respect to the expectation of returns over all possible actions for a given state.

Optimal Value Functions & Optimal Policy (*)

The optimal value functions produce the maximum return...

Optimal value functions

$$V_*(s) = \max_\pi V_\pi(s) \qquad Q_*(s,a) = \max_\pi Q_\pi(s,a)$$

The optimal policy achieves optimal value functions $V_*(s)$ and $Q_*(s,a)$

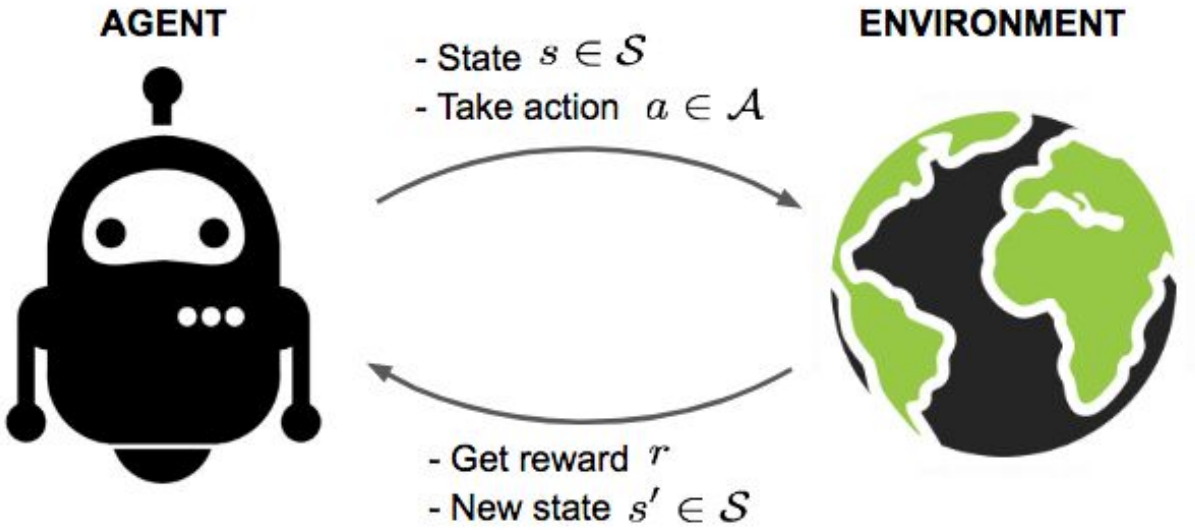
Optimal policy π_*

$$\pi_* = \arg\max_\pi V_\pi(s) \qquad \pi_* = \arg\max_\pi Q_\pi(s,a)$$

Outline

1. Control tasks with reinforcement learning
2. Markov Decision Processes
3. **Learning action-value function Q**
4. Learning policies

Value vs Policy vs Model

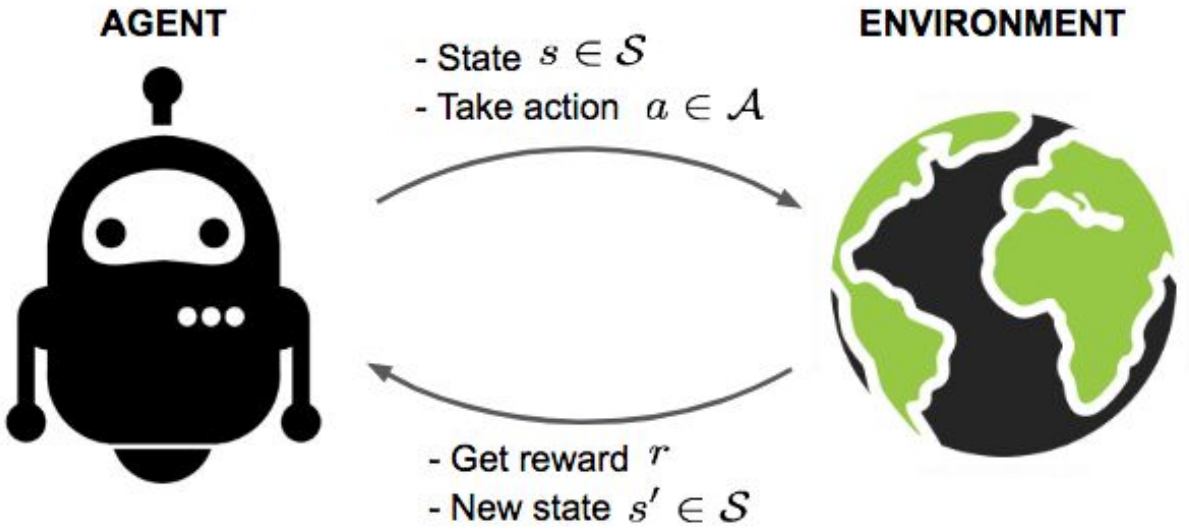


Policy π

Value function

Model
(of the Environment)

Value vs Policy vs Model



Policy π

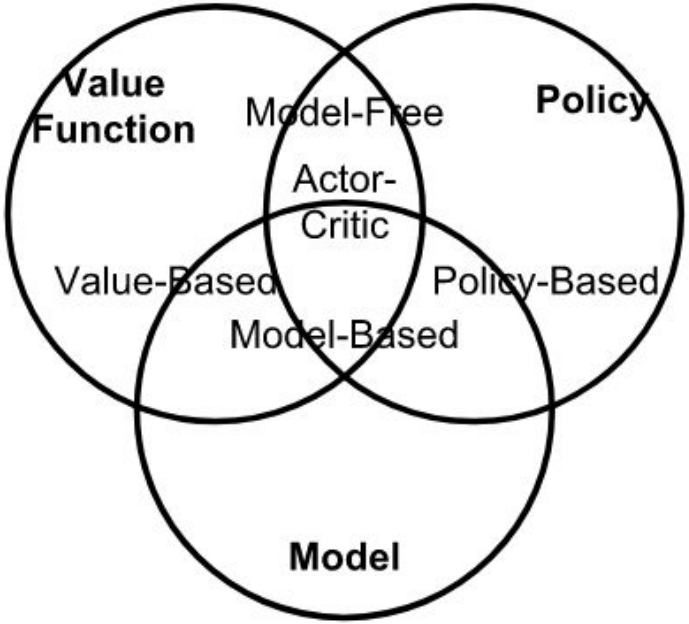
Value function

Model (of the Environment)

Goals of Reinforcement Learning

Value vs Policy vs Model

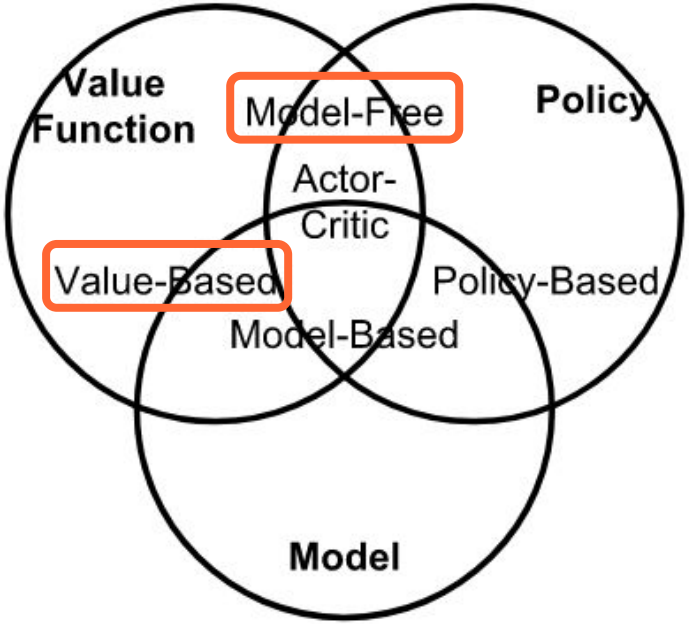
Summary of approaches in RL based on whether we want to learn the value, policy, or the model of the environment.



David Silver (Deepmind), "Introduction to Deep Learning" (2015)

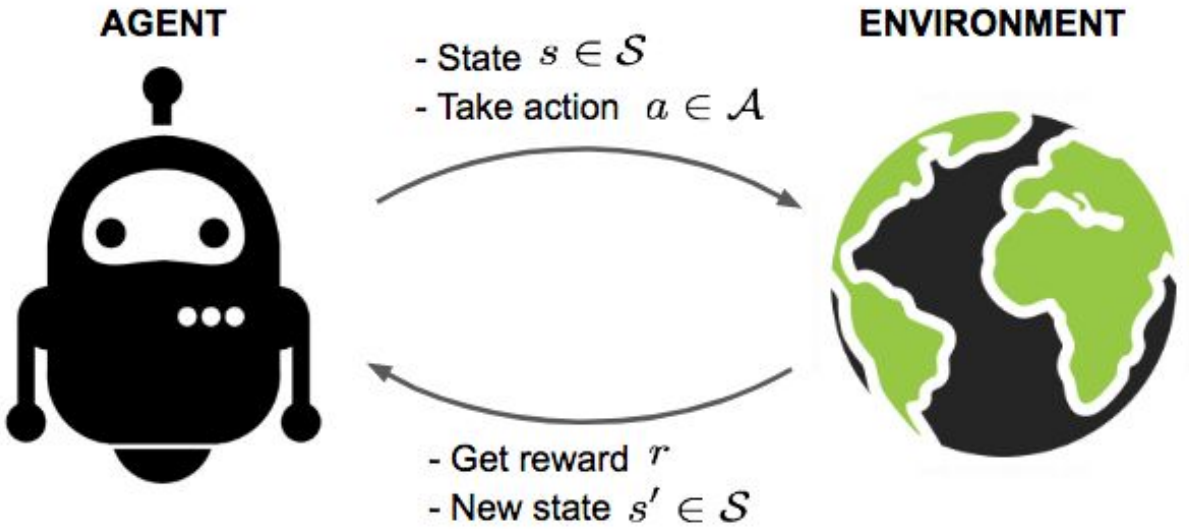
Value vs Policy vs Model

Summary of approaches in RL based on whether we want to learn the value, policy, or the model of the environment.



David Silver (Deepmind), "Introduction to Deep Learning" (2015)

Value vs Policy vs Model



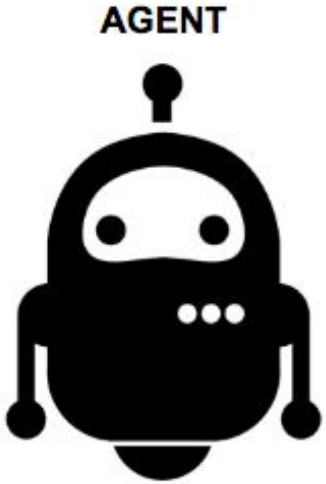
Policy π

Value function

Model (of the Environment)

Goals of Reinforcement Learning

Inferring a Policy from the Value



Given a value function, a policy can be easily defined. For example:

Greedy policy

$\pi(s) = \arg\max_{a \in A} Q(s,a)$

Policy π ← Value function

Deep Q-learning (DQN)

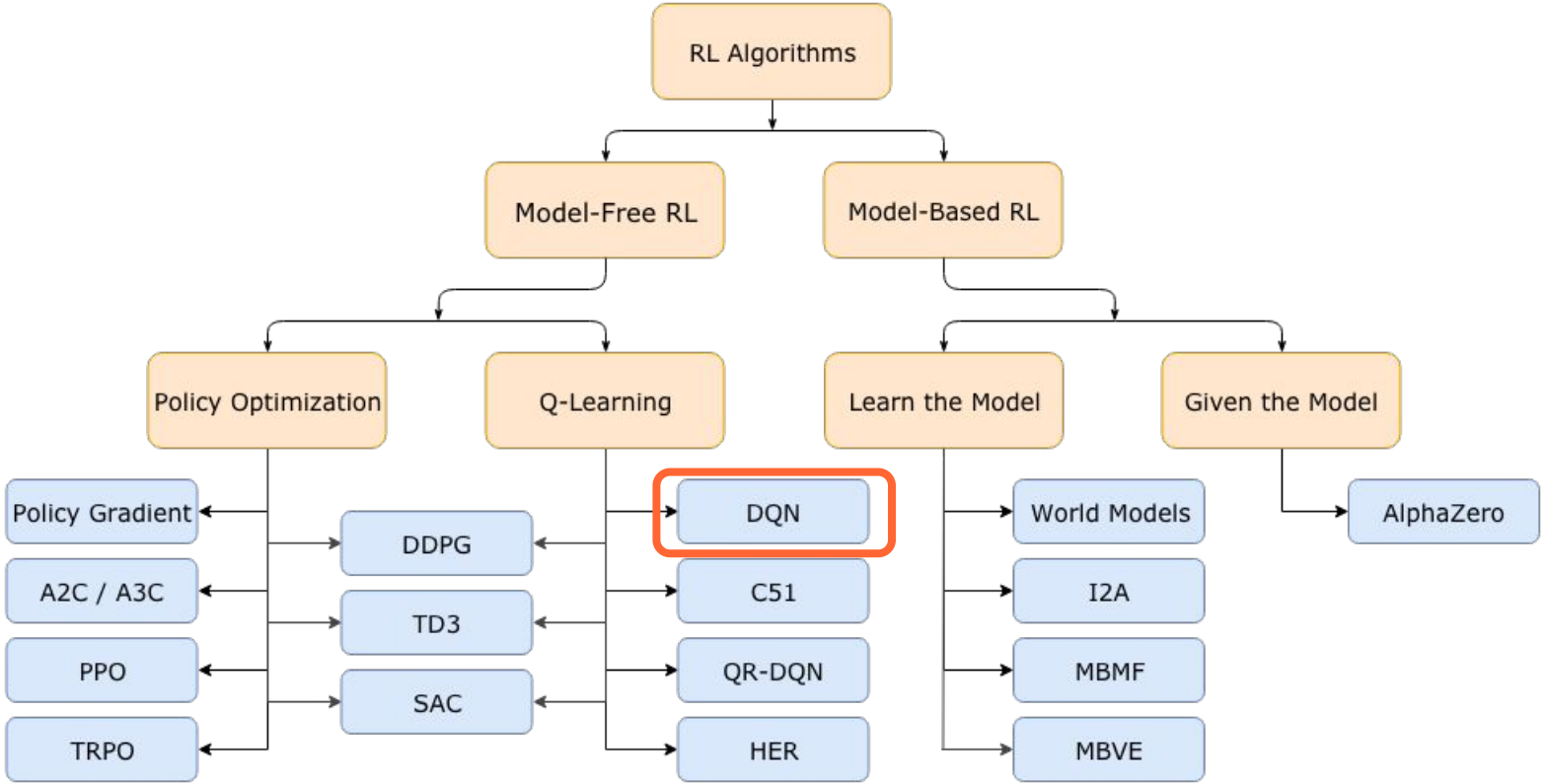


Figure: OpenAI Spinning Up

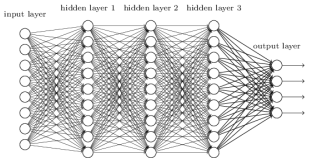
Deep Q-learning (DQN)

Problem: Estimating the optimal action-value function $Q_*(s,a)$ might be feasible with exhaustive search if few state-action pairs are possible, but impossible for large spaces.

Solution: Learn an approximation of the optimal action-value function $Q(s,a,\theta)$ with a NN:

$Q(s,a,\theta)$ → $Q_*(s) = \arg\max_{a \in A} Q(s,a,\theta)$

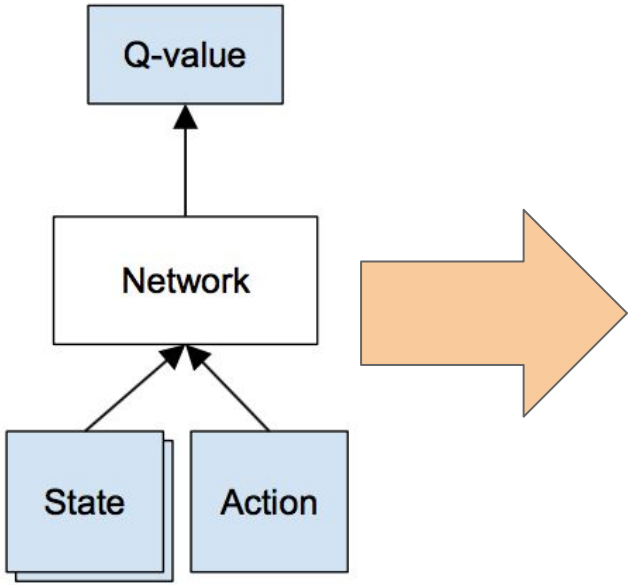
Eg. Neural Network parameters



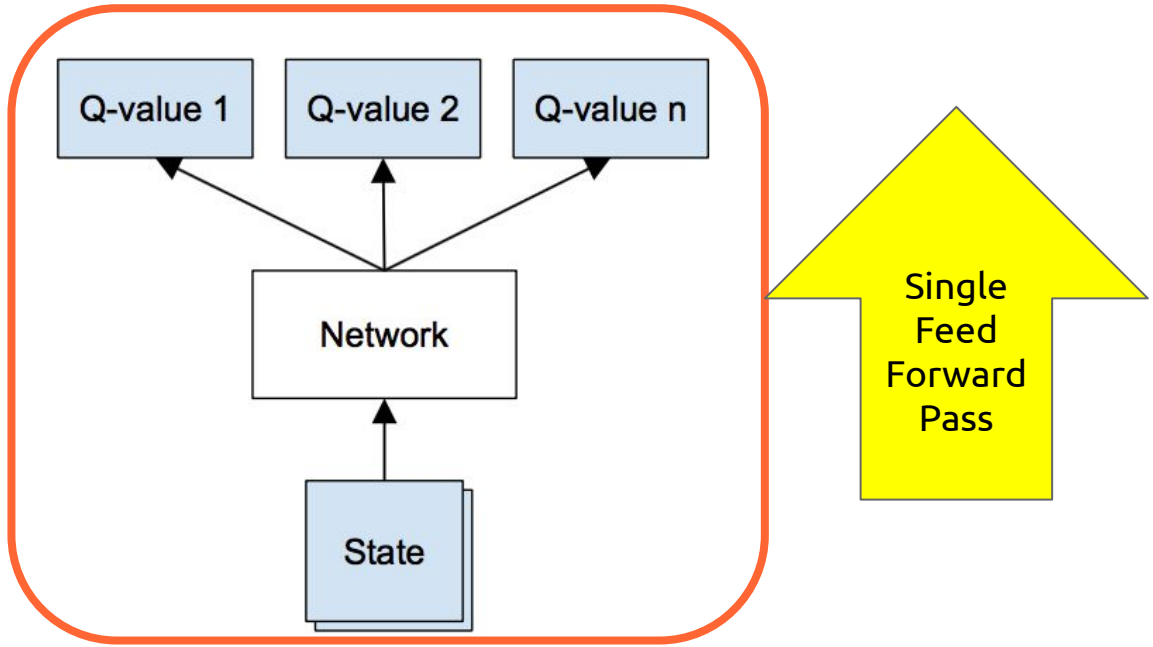
Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves et al. "Human-level control through deep reinforcement learning." *Nature* 518, no. 7540 (2015): 529-533.

Deep Q-learning (DQN)

Q(s,a, θ)

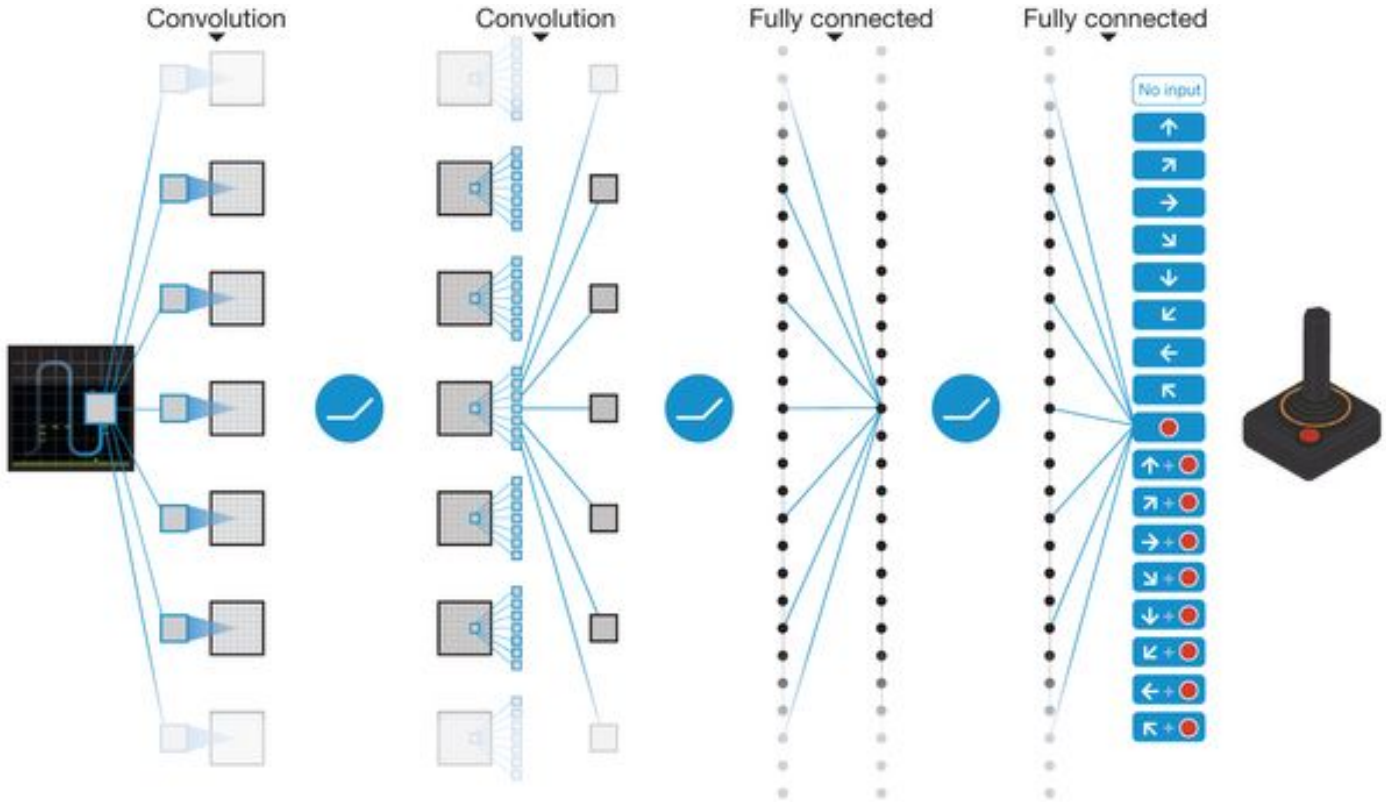


A **single feedforward** pass to compute the Q-values for all actions from the current state (efficient)



Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves et al. "Human-level control through deep reinforcement learning." *Nature* 518, no. 7540 (2015): 529-533.

Deep Q-learning (DQN)



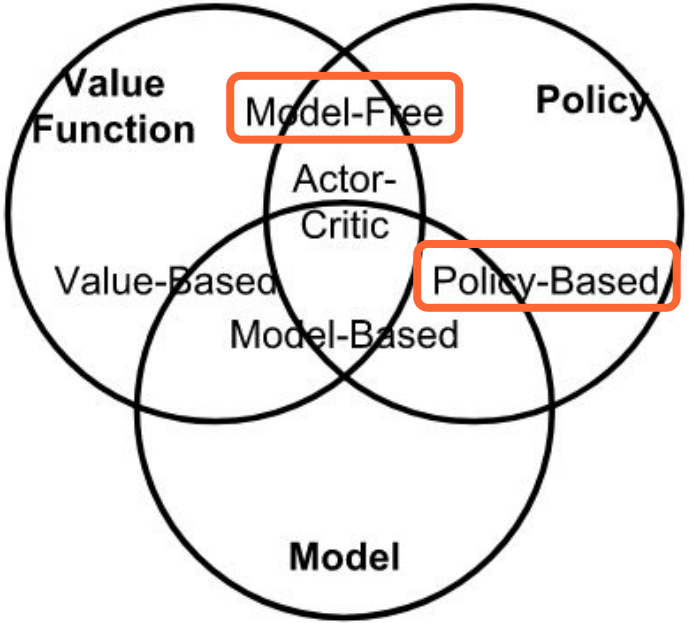
Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves et al. "Human-level control through deep reinforcement learning." *Nature* 518, no. 7540 (2015): 529-533.

Outline

1. Control tasks with reinforcement learning
2. Markov Decision Processes
3. Learning action-value function Q
4. **Learning policies**

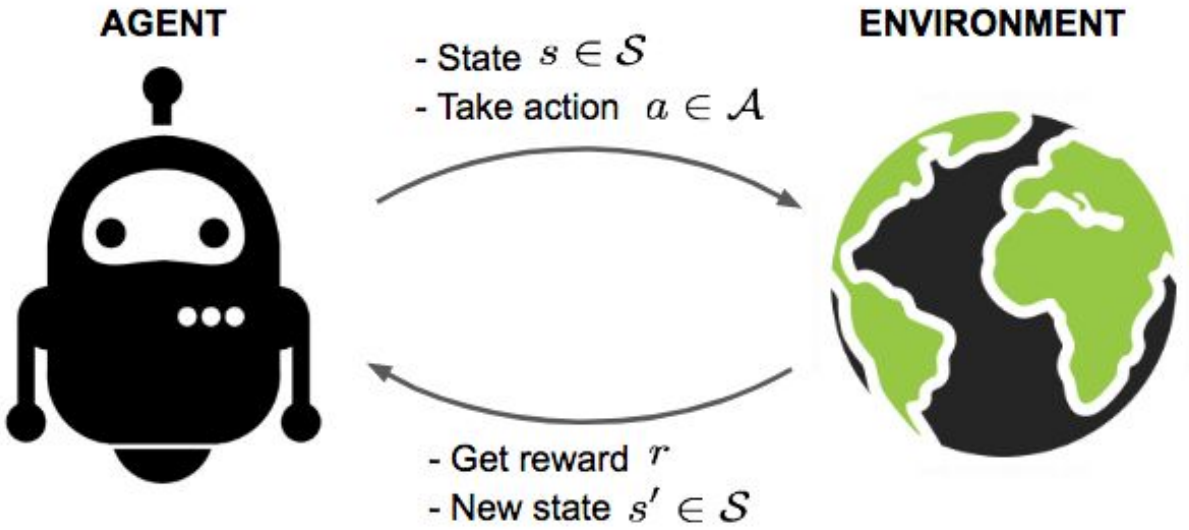
Value vs Policy vs Model

Summary of approaches in RL based on whether we want to learn the value, policy, or the model of the environment.



David Silver (Deepmind), "Introduction to Deep Learning" (2015)

Value vs Policy vs Model



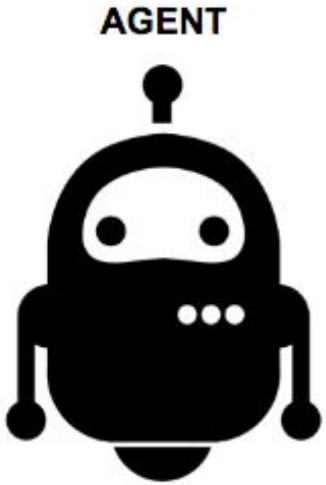
Policy π

Value function

Model (of the Environment)

Goals of Reinforcement Learning

Value vs Policy vs Model



Directly learn the policy by estimating the parameters θ of a stochastic policy function:

$$\pi(a|s;\theta)$$

Policy π

Value function

REINFORCE (Vanilla Policy Gradients - VPN)

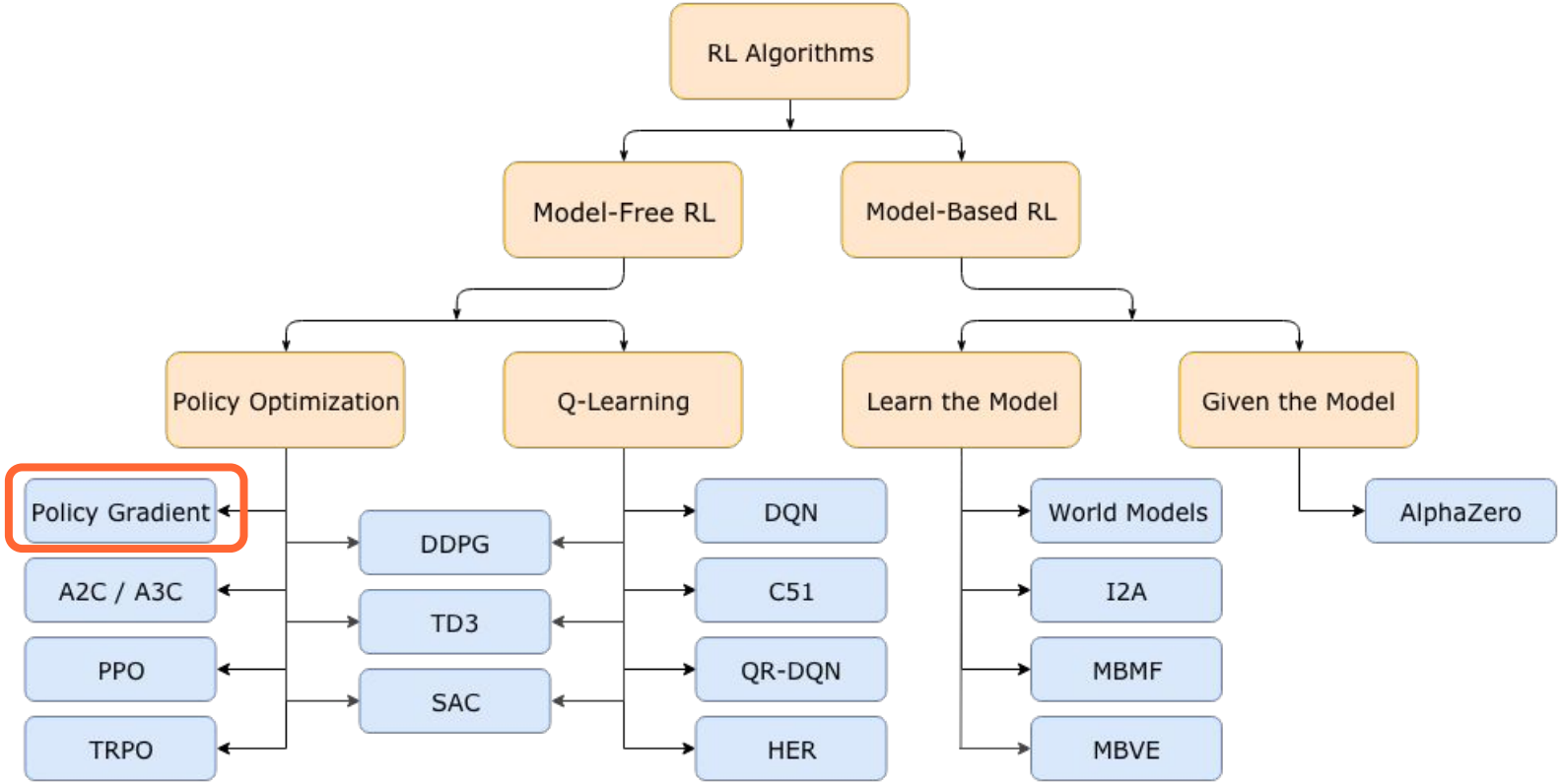


Figure: OpenAI Spinning Up

REINFORCE (Vanilla Policy Gradients - VPN)

REINFORCE (Vanilla Policy Gradients - VPN)

the mathematical formulation of 'trial-and-error': try and action, and make it more likely if it resulted in positive reward; otherwise, make it less likely.

Opposite goals in:

- **Supervised learning:** minimize a loss function by gradient descent.
- **Reinforcement learning:** maximize J(θ), the expected return of the policy, by gradient ascent.

$$\nabla_\theta J(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left[\sum_{t=1}^{T} R_{i,t} \nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}, \mathbf{s}_{i,t}) \right]$$

Sutton, Richard S., David A. McAllester, Satinder P. Singh, and Yishay Mansour. "Policy gradient methods for reinforcement learning with function approximation." NIPS 2000.

A taxonomy of RL Algorithms

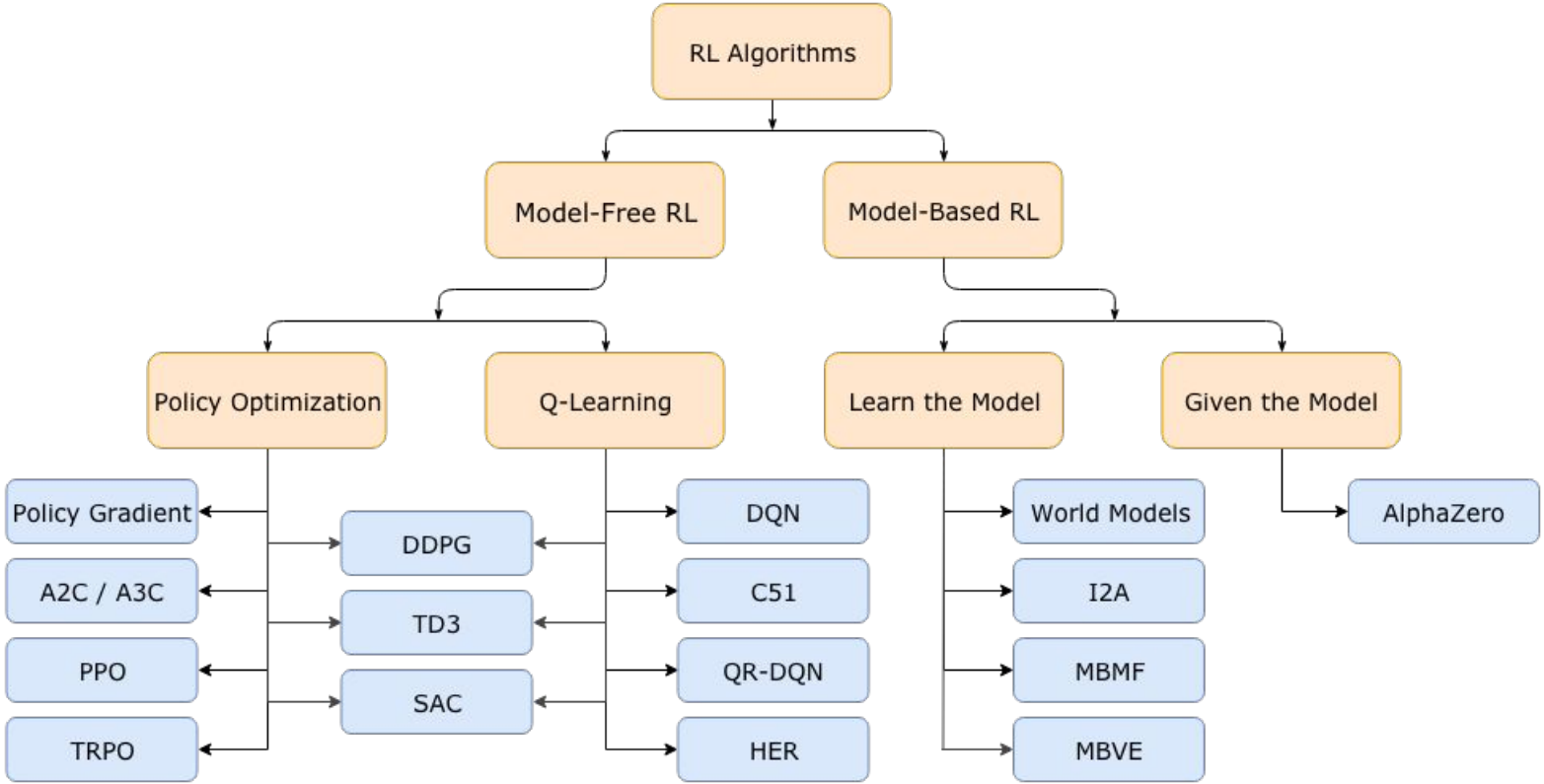


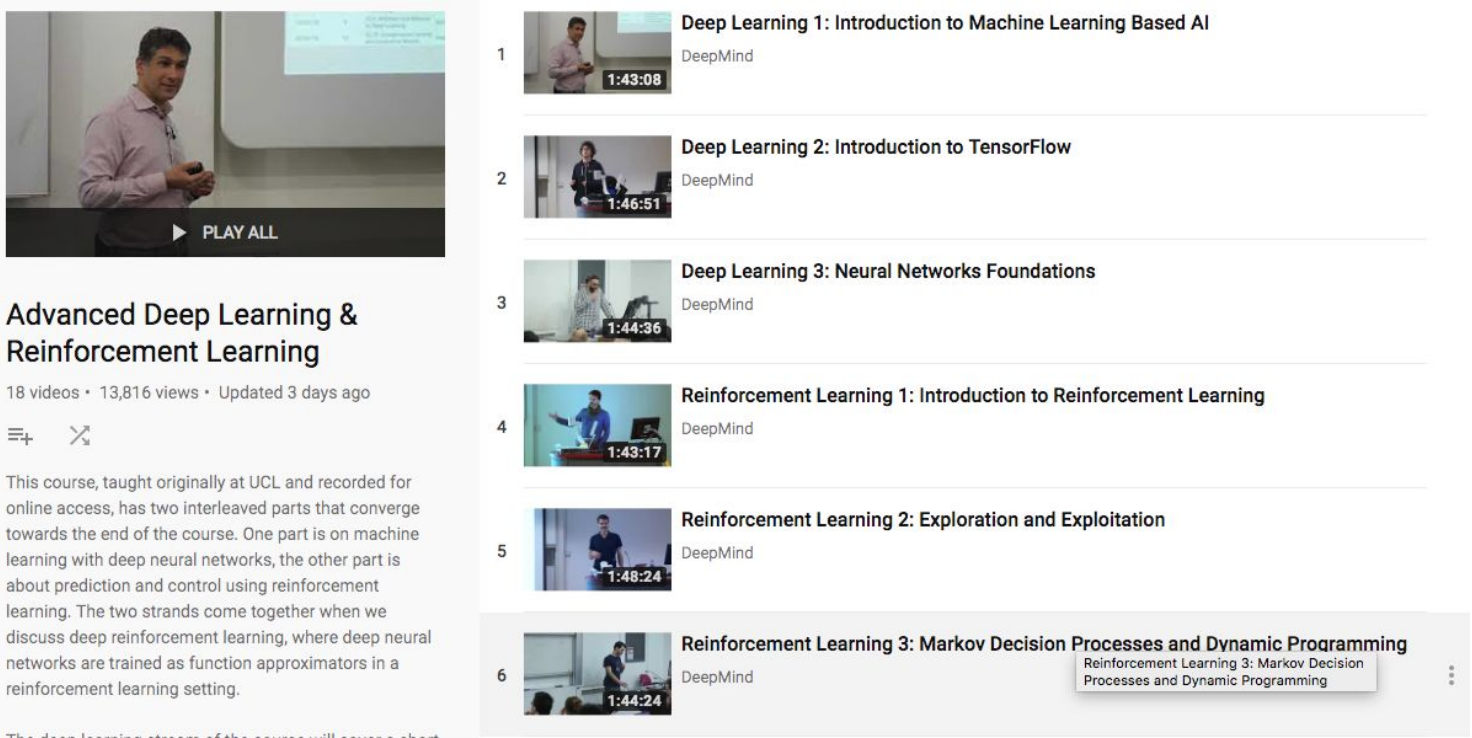
Figure: OpenAI Spinning Up

Outline

1. Control tasks with reinforcement learning
2. Markov Decision Processes
3. Learning action-value functions Q
4. Learning policies

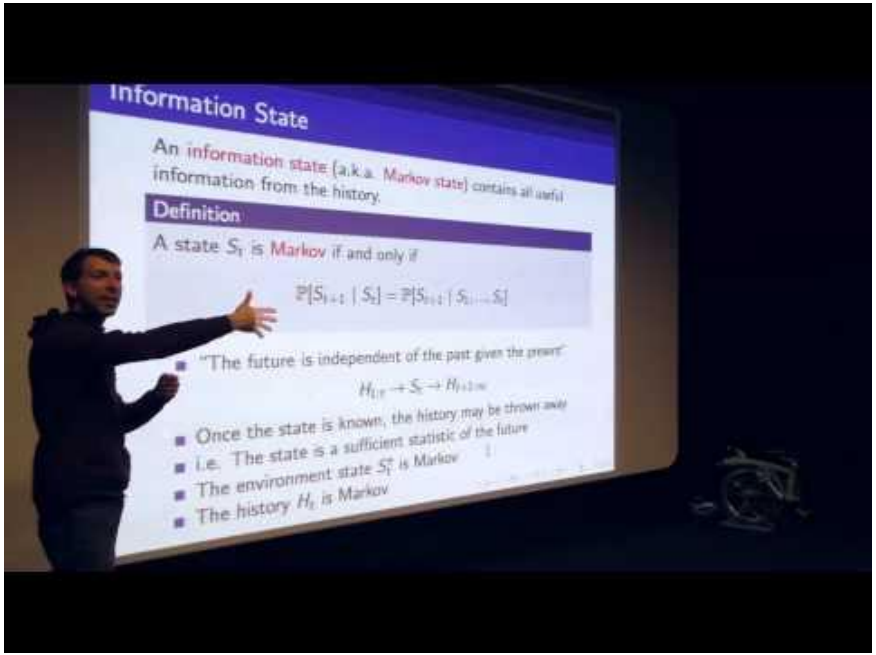
Learn more

UCL & Deepmind: [“Advanced Deep Learning & Reinforcement Learning”](#) (2018) [slides]



Learn more

David Silver, UCL COMP050, [Reinforcement Learning]



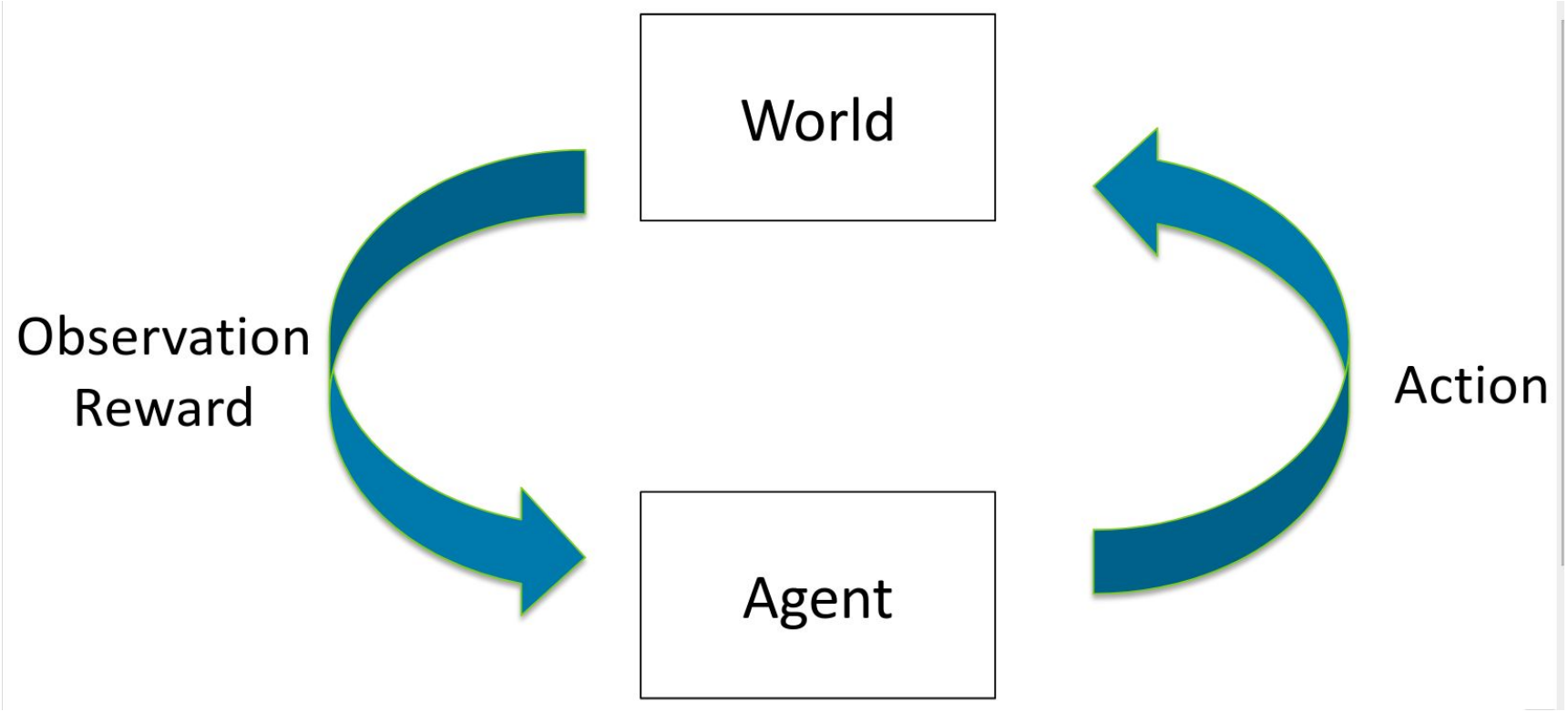
Learn more

Pieter Abbeel and John Schulman, CS 294-112 Deep Reinforcement Learning, Berkeley.

Slides: "Reinforcement Learning - Policy Optimization" OpenAI / UC Berkeley (2017)

Learn more

Emma Brunskill, "CS234: Reinforcement Learning". Stanford University.



Learn more

[OpenAI Spinning Up in Deep RL]

OpenAI

Final Questions