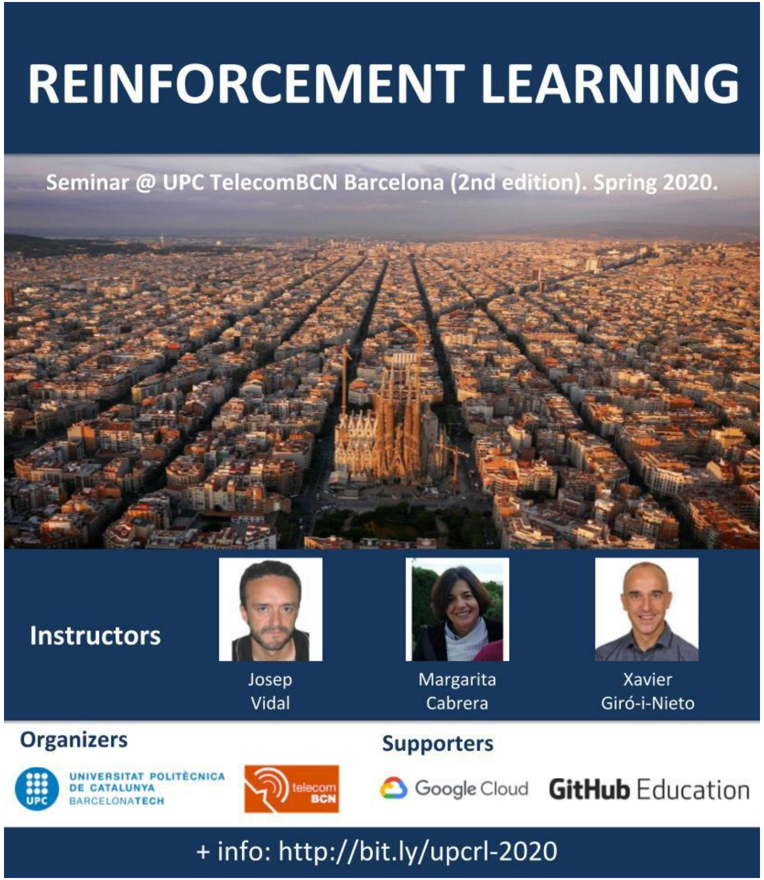


https://telecombcn-dl.github.io/mrl-2020/

telecos BCN

MRL 2020 - Day 10 - Part 1

REINFORCE

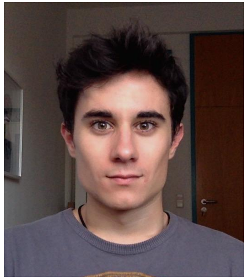
Xavier Giro-i-Nieto
[@DocXavi](https://twitter.com/DocXavi)
xavier.giro@upc.edu

Associate Professor
Universitat Politècnica de Catalunya
Barcelona Supercomputing Center

UPC

BSC

Acknowledgements



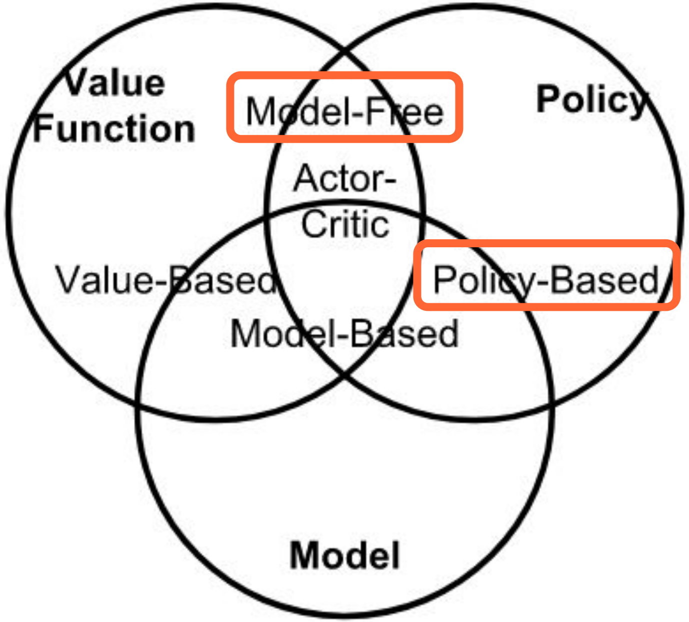
[Víctor Campos]
victor.campos@bsc.es

PhD Candidate

Barcelona Supercomputing Center

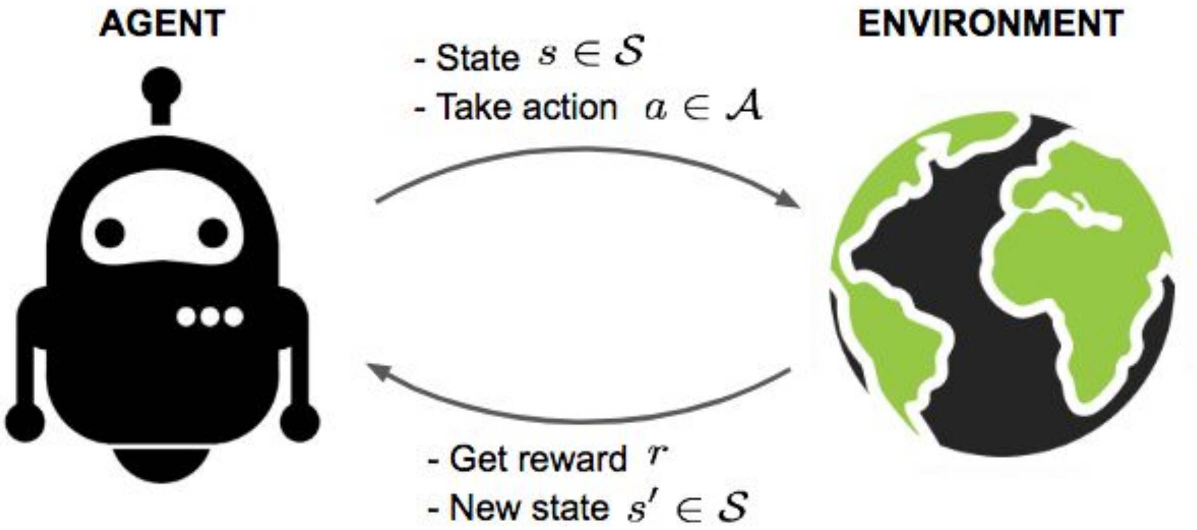
Policy-based RL

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)

Policy-based RL



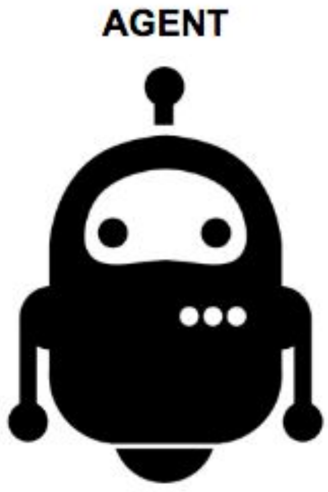
Policy π

Value function

Model
(of the Environment)

Goals of Reinforcement Learning

Policy-based RL



Policy π

Value function

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

$$\boxed{\pi_\theta(a|s)}$$

Our goal:

estimate the probability of taking action a given state s"

Policy Gradient

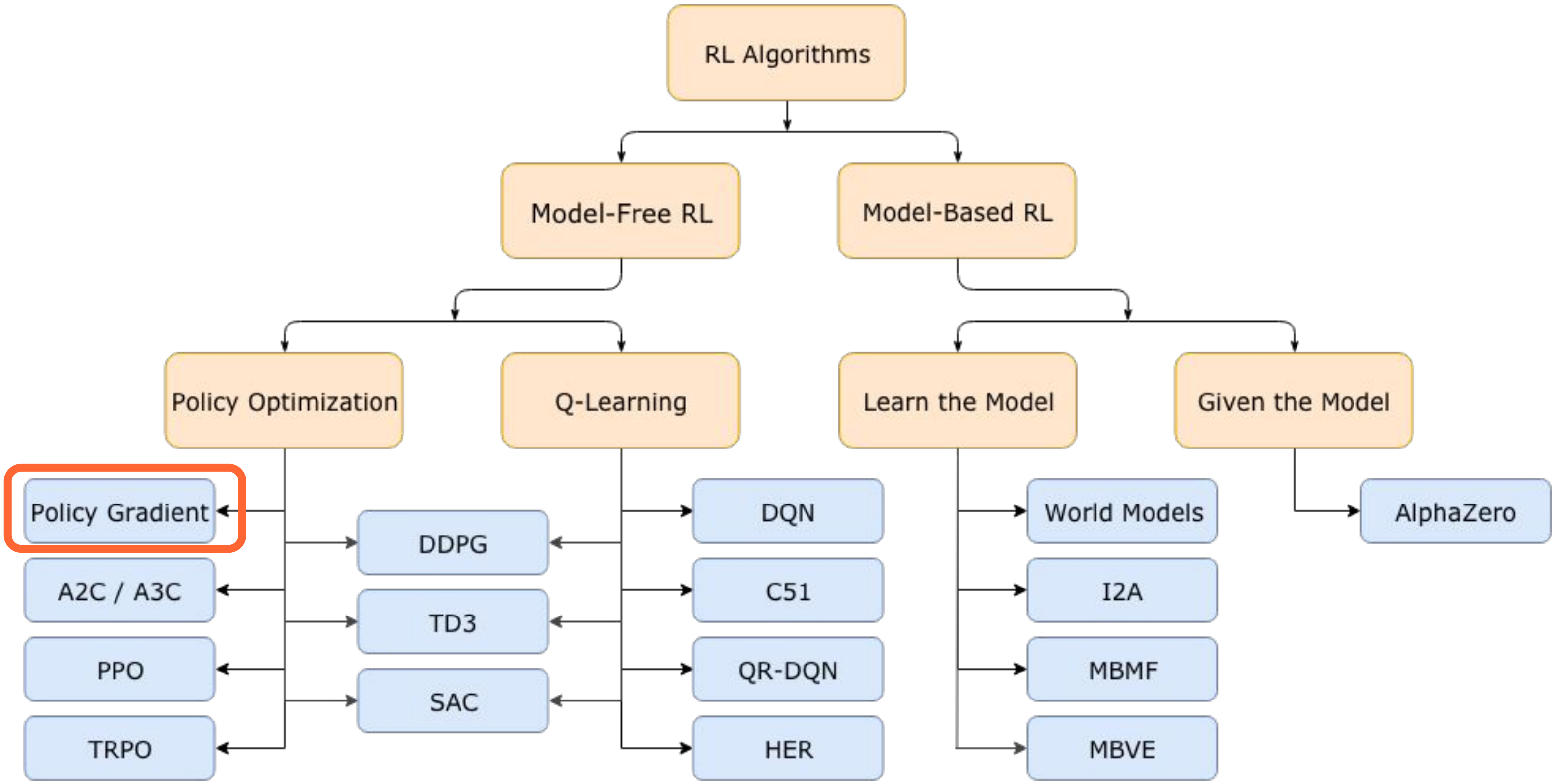


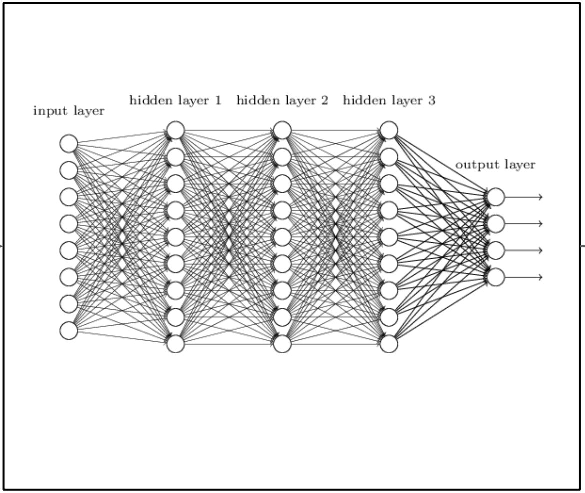
Figure: OpenAI Spinning Up

Previously: Loss function to compute gradients

Neural Network

input

x



output

$f_\theta(x)$

labels (ground truth)

input

$$\mathcal{L}(w) = distance(f_\theta(x), y)$$

error

parameters (weights, biases)

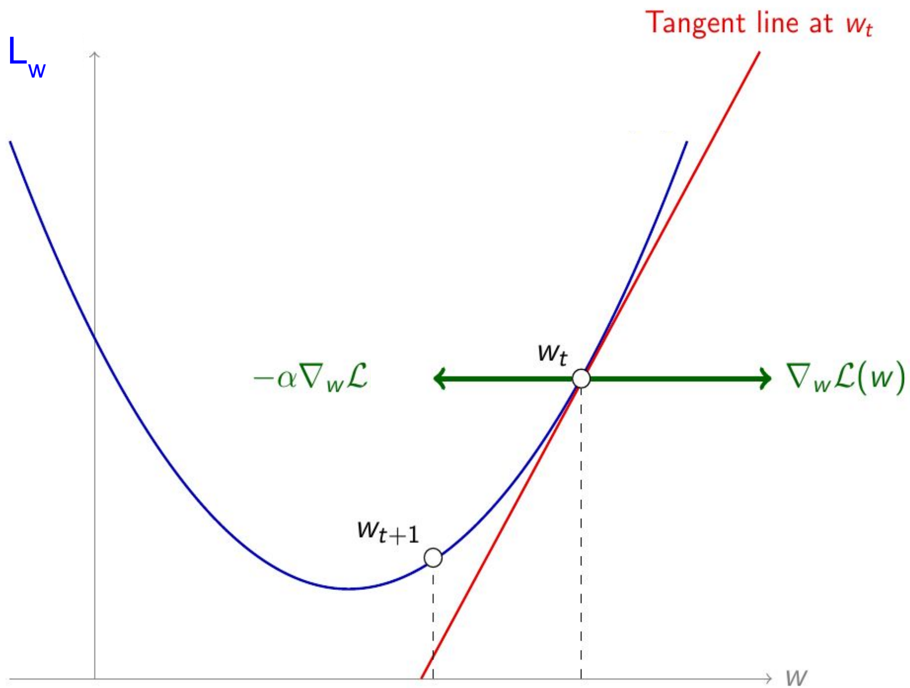
Previously: Gradient Descent (GD)

By estimating the gradient of the Loss (∇L) with respect to each parameter in the NN, we use (Stochastic) Gradient Descent and backpropagation to iteratively update them.

Descend
(minus sign)

$$w_{t+1} \leftarrow w_t - \alpha \nabla \mathcal{L}_w(w_t)$$

Learning rate (LR)



Training Neural Networks for RL

Question: What target function should we use to optimize a policy $\pi_\theta(a|s)$?

Reminder:

The **optimal policy** is that one capable of achieving the optimal value functions $V_*(s)$ and $Q_*(s,a)$

Optimal policy π_*

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

Value functions for policy π

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

Training Neural Networks for RL

Question: What target function should we use to optimize a policy $\pi_\theta(a|s)$?

Reminder:

The **optimal policy** is that one capable of achieving the optimal value functions $V_*(s)$ and $Q_*(s,a)$

Optimal policy π_*

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

Value functions for policy π

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

Training Neural Networks for RL

Question: How can we estimate the expected return of a policy $\pi_\theta(a|s)$?

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

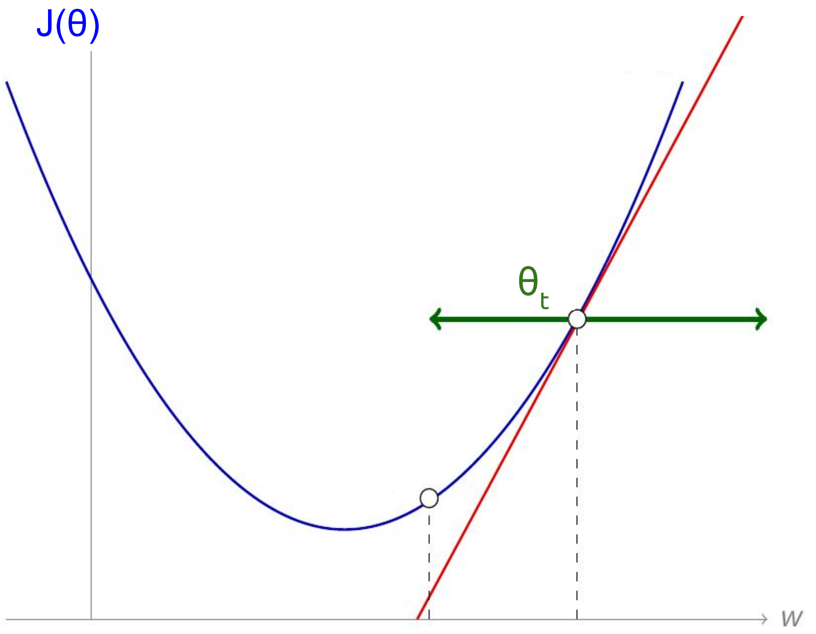
New notation J instead of G !!

$$J(\theta) = E_{\tau \sim p_\theta(\tau)}\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t)\right] \approx \frac{1}{N}\sum_i \sum_t r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t})$$

sum over samples from π_θ

Training Neural Networks for RL

Question: Which direction should the update of parameter θ take in RL ?



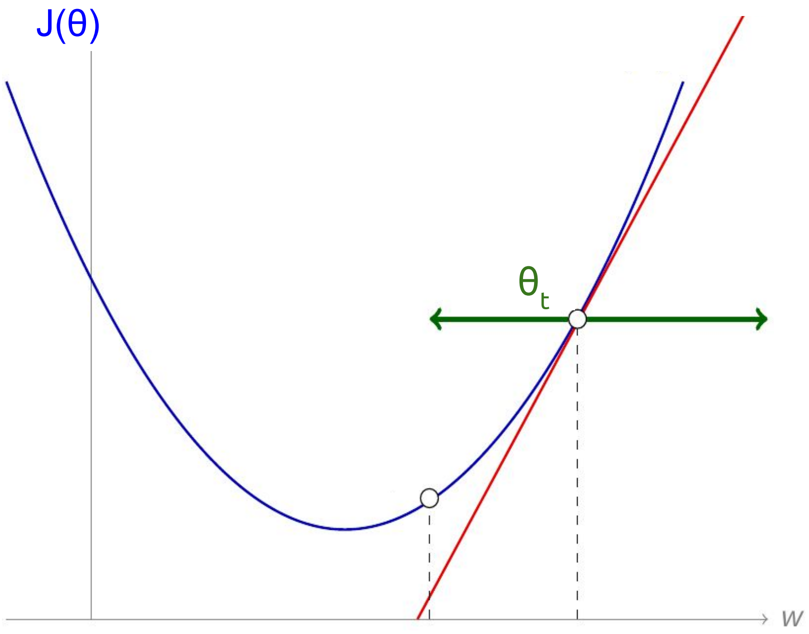
Gradient Ascent

By estimating the gradient of the Expected Return (∇J) with respect to each parameter in the NN, we use (Stochastic) Gradient **Ascent** and backpropagation to iteratively update them.

Ascend
(plus sign)

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$$

Learning rate (LR)



REINFORCE (Vanilla Policy Gradients - VPN)

$$\nabla_\theta J(\theta) =$$

Expected return of the policy

REINFORCE (Vanilla Policy Gradients - VPN)

Parameters of the policy $\pi\theta(a|s)$

$$\nabla_\theta J(\theta) =$$

Expected return of the policy $\pi_\theta(a|s)$

REINFORCE (Vanilla Policy Gradients - VPN)

Parameters of the policy $\pi\theta(a|s)$

$$\nabla_\theta J(\theta) =$$

Expected return of the policy $\pi_\theta(a|s)$

Opposite goals in:

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

REINFORCE (Vanilla Policy Gradients - VPN)

Parameters of the policy $\pi\theta(a|s)$

$$\nabla_\theta J(\theta) =$$

How to estimate the gradient ?

Expected return of the policy $\pi_\theta(a|s)$

REINFORCE (Vanilla Policy Gradients - VPN)

Parameters of the policy $\pi\theta(a|s)$

$$\nabla_\theta J(\theta) =$$

Expected return of the policy $\pi_\theta(a|s)$

REINFORCE (Vanilla Policy Gradients - VPN)

Expected return of the policy $\pi_\theta(a|s)$

$$\nabla_\theta J(\theta) =$$

Parameters of the policy $\pi\theta(a|s)$

Supervised learning: minimize a loss L(θ) function by gradient descent.

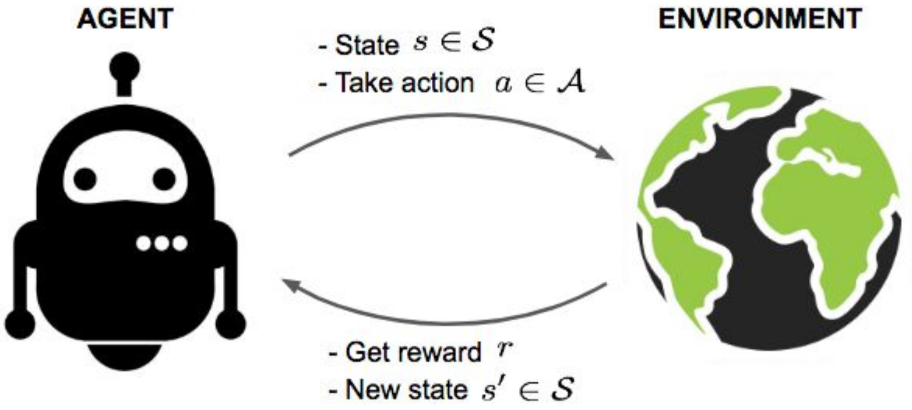
$$\nabla \mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \nabla L(\mathbf{y}_i, \hat{\mathbf{y}}_i)$$

The parameters of the NN, θ, are iteratively updated by assessing the loss function between N pairs of predicted (ŷ) and ground truth (y) labels.

Question: What are the equivalent of N pairs (ŷ,y) in reinforcement learning ?

REINFORCE (Vanilla Policy Gradients - VPN)

Question: What are the equivalent of N pairs (ŷ,y) in reinforcement learning ?



N complete episodes of our policy $\pi\theta(a|s)$ with the environment.

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

REINFORCE (Vanilla Policy Gradients - VPN)

Expected return of the policy $\pi_\theta(a|s)$

$$\nabla_\theta J(\theta) = \boxed{\frac{1}{N}\sum_{i=1}^{N}}$$

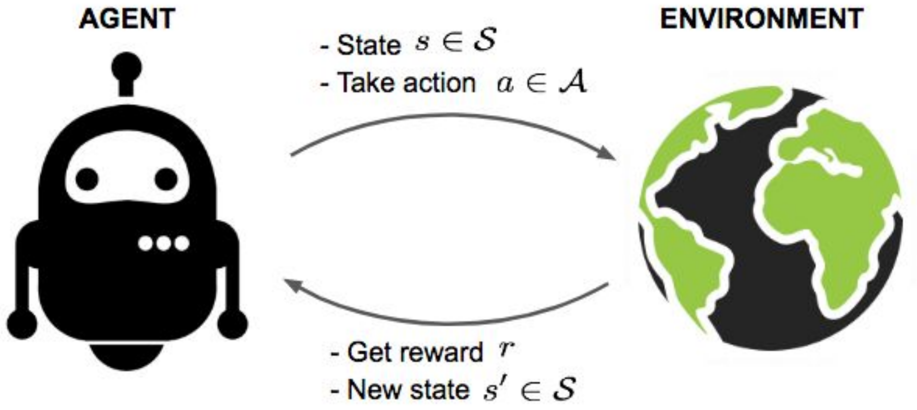
Estimation of the expectation over all possible trajectories, using N sampled trajectories (Monte Carlo)

N complete episodes of our policy $\pi\theta(a|s)$ with the environment.

Slide: Víctor Campos, Barcelona TensorFlow Meetup (2018)

REINFORCE (Vanilla Policy Gradients - VPN)

Question: What are the equivalent of N pairs (ŷ,y) in reinforcement learning ?



N complete episodes of our policy $\pi\theta(a|s)$ with the environment, of T interactions:

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

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

REINFORCE (Vanilla Policy Gradients - VPN)

Expected return of the policy $\pi_\theta(a|s)$

$$\nabla_\theta J(\theta) = \boxed{\frac{1}{N}\sum_{i=1}^{N}} \left[\sum_{t=1}^{T} \boxed{\quad ? \quad} \right]$$

Estimation of the expectation over all possible trajectories, using N sampled trajectories (Monte Carlo)

REINFORCE (Vanilla Policy Gradients - VPN)

Remembering the definition of the return... $G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$

Expected return of the policy $\pi_\theta(a|s)$

$$\nabla_\theta J(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left[\sum_{t=1}^{T} R_{i,t} \boxed{\quad ? \quad} \right]$$

(discounted) reward

REINFORCE (Vanilla Policy Gradients - VPN)

...but also the (log)-probability of following a specific trajectory...

Expected return of the policy $\pi_\theta(a|s)$

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

Slide: Víctor Campos, Barcelona TensorFlow Meetup (2018)

REINFORCE (Vanilla Policy Gradients - VPN)

...and the derivative !

The policy gradient. If we follow it, the action $\mathbf{a}_{i,t}$ will be more likely if the agent ever finds itself again in state $\mathbf{s}_{i,t}$

Expected return of the policy $\pi_\theta(a|s)$

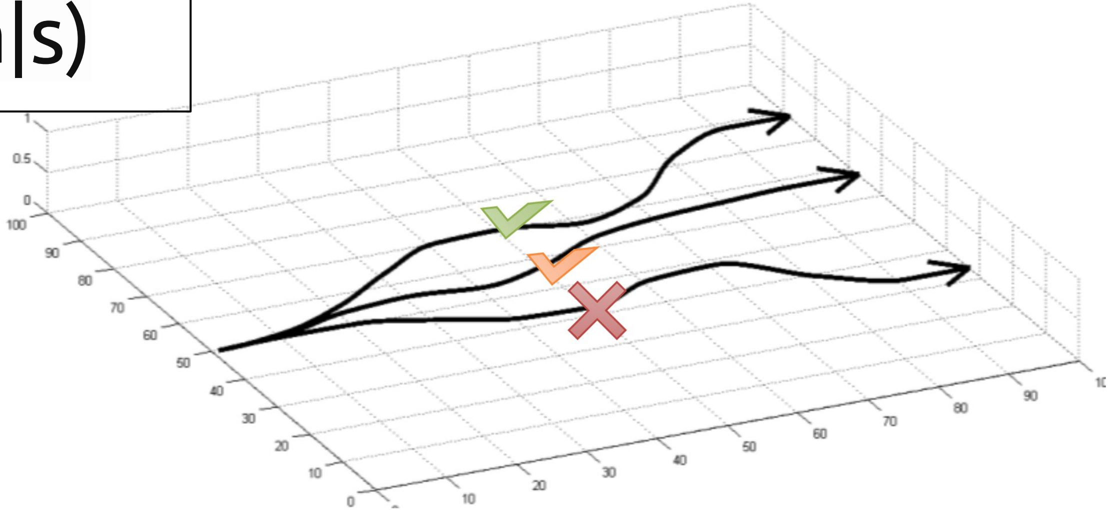
$$\nabla_\theta J(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{t=1}^{T} R_{i,t} \boxed{\nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}, \mathbf{s}_{i,t})}\right]$$

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.

$$\pi_\theta(a|s)$$



Source: Lilian Weng, [“A long peek into reinforcement learning”](#) (2018)

REINFORCE (Vanilla Policy Gradients - VPN)

The policy gradient. If we follow it, the action **a**$_{i,t}$ will be more likely if the agent ever finds itself again in state **s**$_{i,t}$

Policy parameters (weights of the NN)

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

Estimation of the expectation over all possible trajectories, using N sampled trajectories (Monte Carlo)

(discounted) reward

Slide: Víctor Campos, Barcelona TensorFlow Meetup (2018)

REINFORCE (Vanilla Policy Gradients - VPN)

1. Initialize θ at random
2. Generate one episode $S_1,A_1,R_2,S_2,A_2,\ldots,S_T$
3. For t=1, 2, … , T:
    - Estimate the the return G_t since the time step t.
    - Compute the gradient $\nabla_\theta J(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{t=1}^{T} R_{i,t}\nabla_\theta \log \pi_\theta(\mathbf{a}_{i,t}, \mathbf{s}_{i,t})\right]$
    - $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

Learn more

Hado van Hasselt, [“Policy Gradients and Actor Critics”](). UCL / Deepmind 2018.

Learn more

OpenAI

Lilian Weng, “Policy Gradient Algorithms” (2018)

Final Questions