Policy Gradients

Shagun Sodhani

9th Data Science UA Conference

20th November, 2020

FACEBOOK AI

Agenda

Sequential Decision Making Problem

Reinforcement Learning

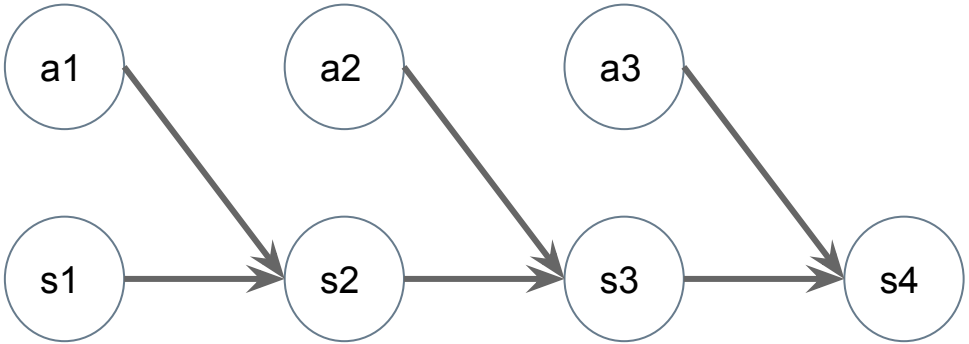
Policy Gradients

A working example

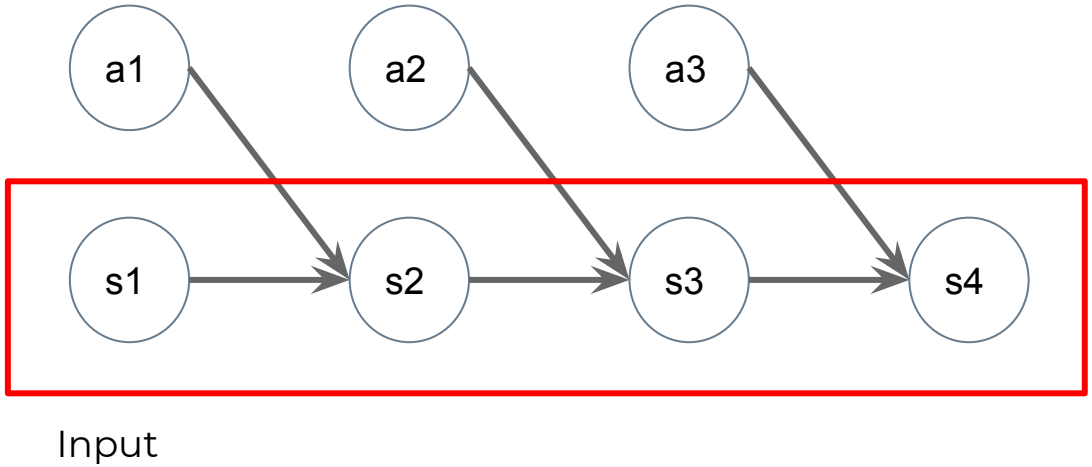
Where do I go next?

Questions are welcome at all times :)

Sequential Decision Making Problem

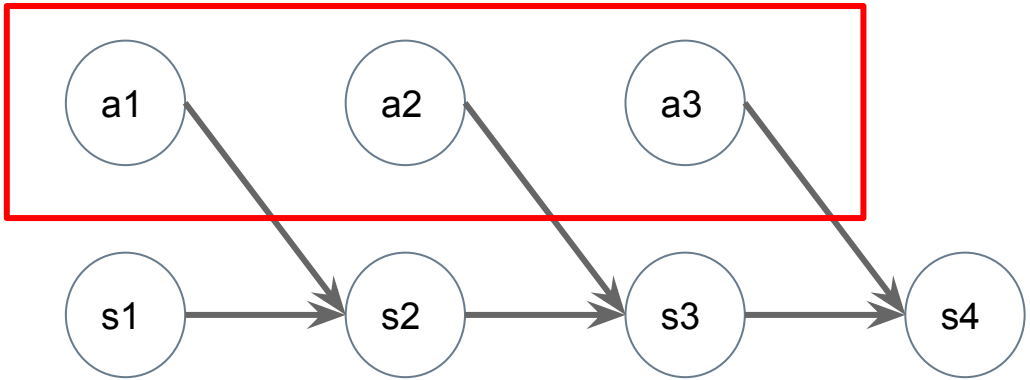


Sequential Decision Making Problem

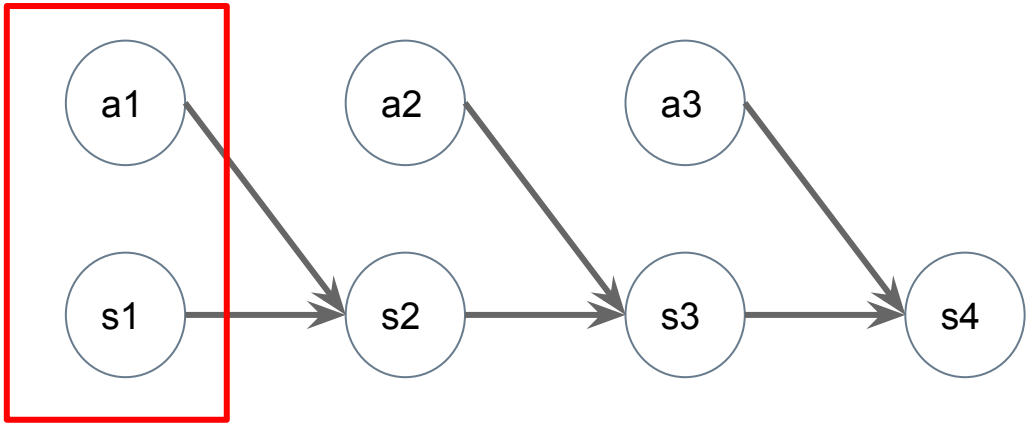


Sequential Decision Making Problem

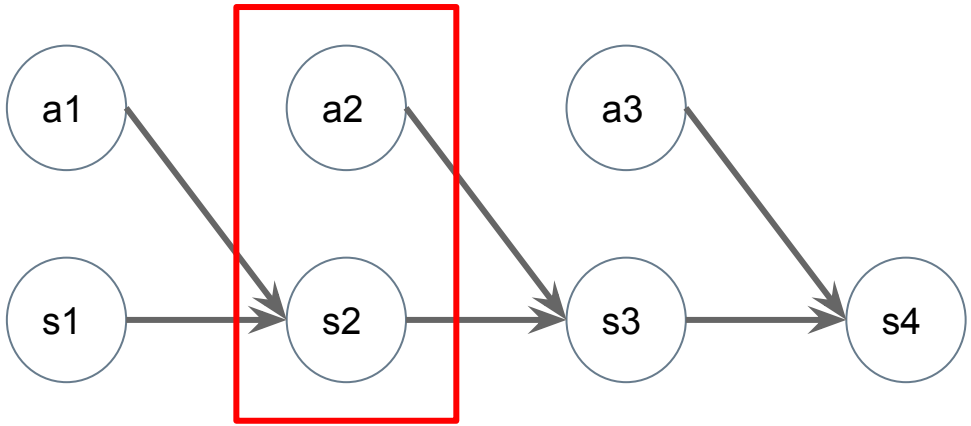
Decisions



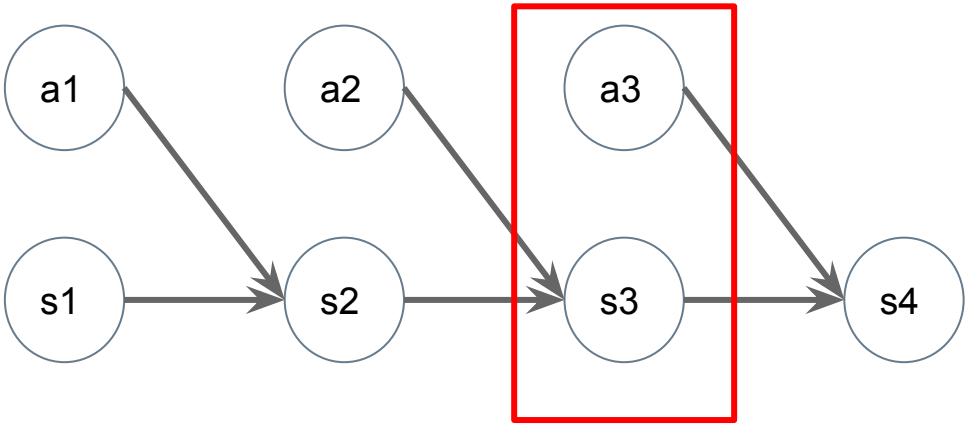
Sequential Decision Making Problem



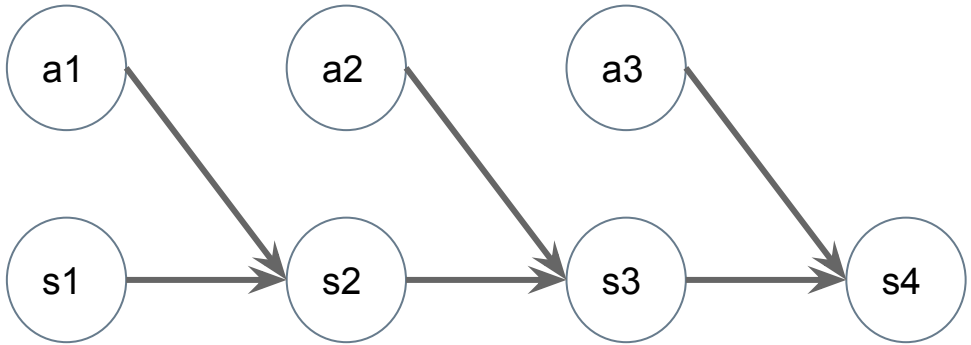
Sequential Decision Making Problem



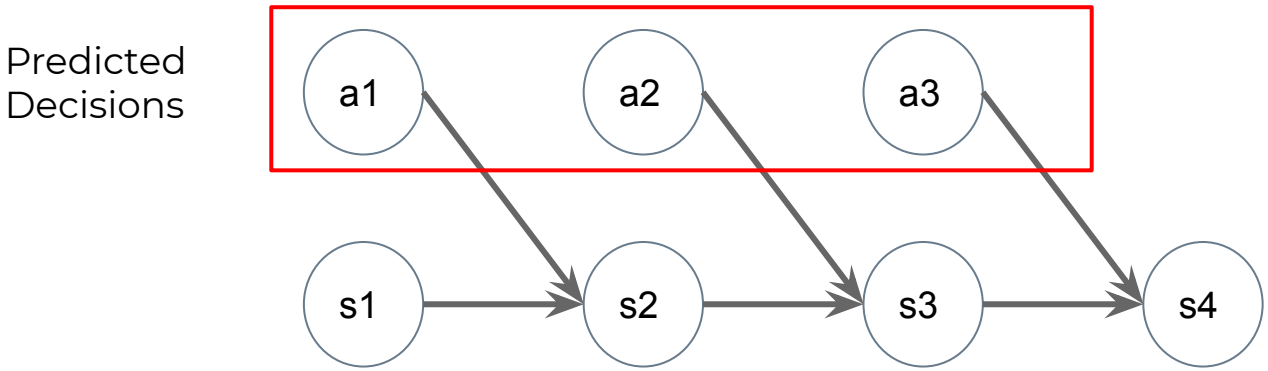
Sequential Decision Making Problem



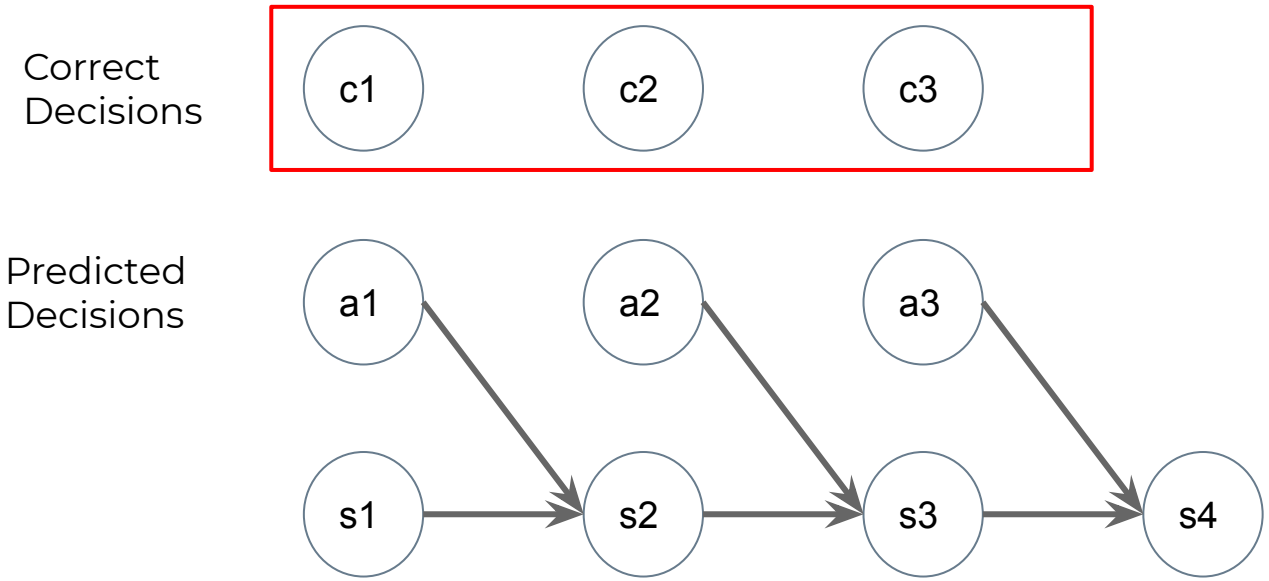
How to train a model for Sequential Decision Making Problem



How to train a model for Sequential Decision Making Problem

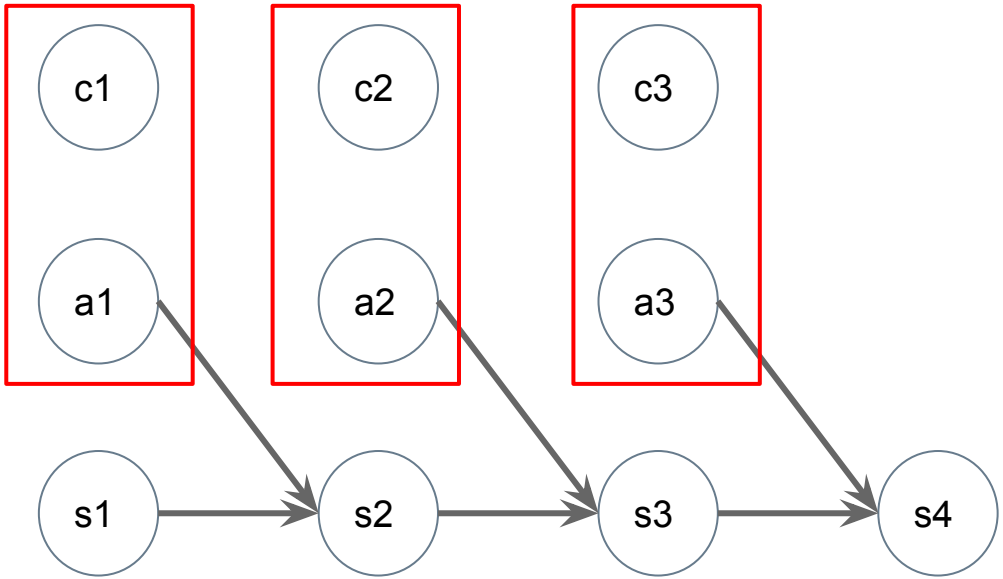


If we know the *correct* decisions, perform Supervised Learning



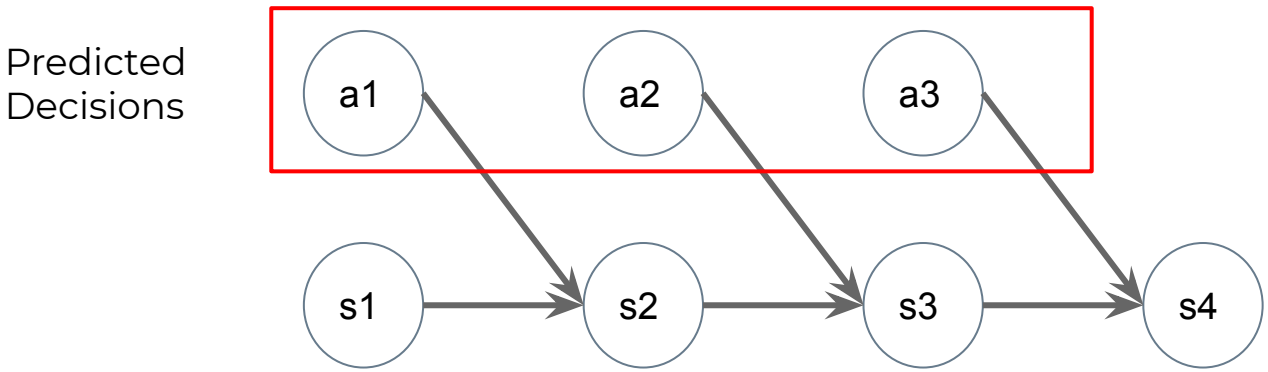
If we know the correct decisions, perform Supervised Learning

Minimize the difference between the correct and the predicted decisions



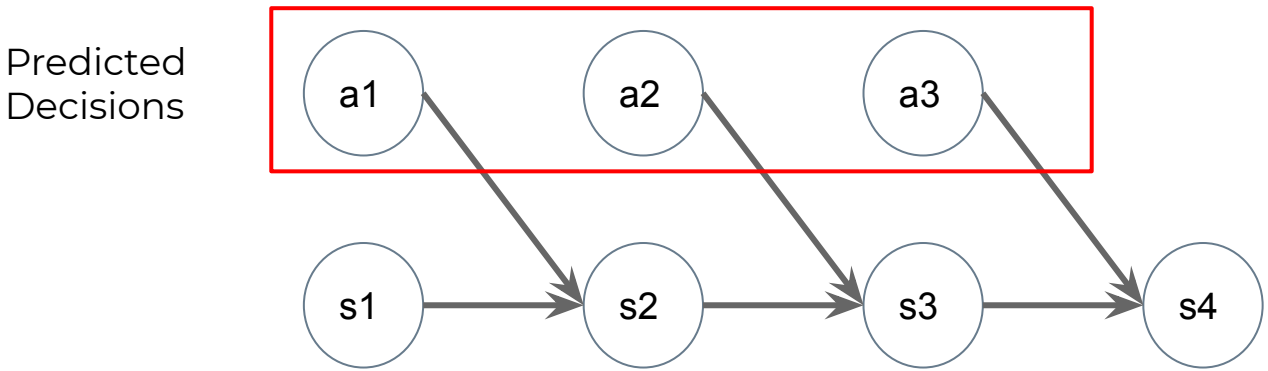
How to train a model for Sequential Decision Making Problem

We do not know the *correct* decision but we have a *sense* of how good each decision is.



Perform Reinforcement Learning

We do not know the *correct* decision but we have a *sense* of how good each decision is.



Agenda

Sequential Decision Making Problem

Reinforcement Learning

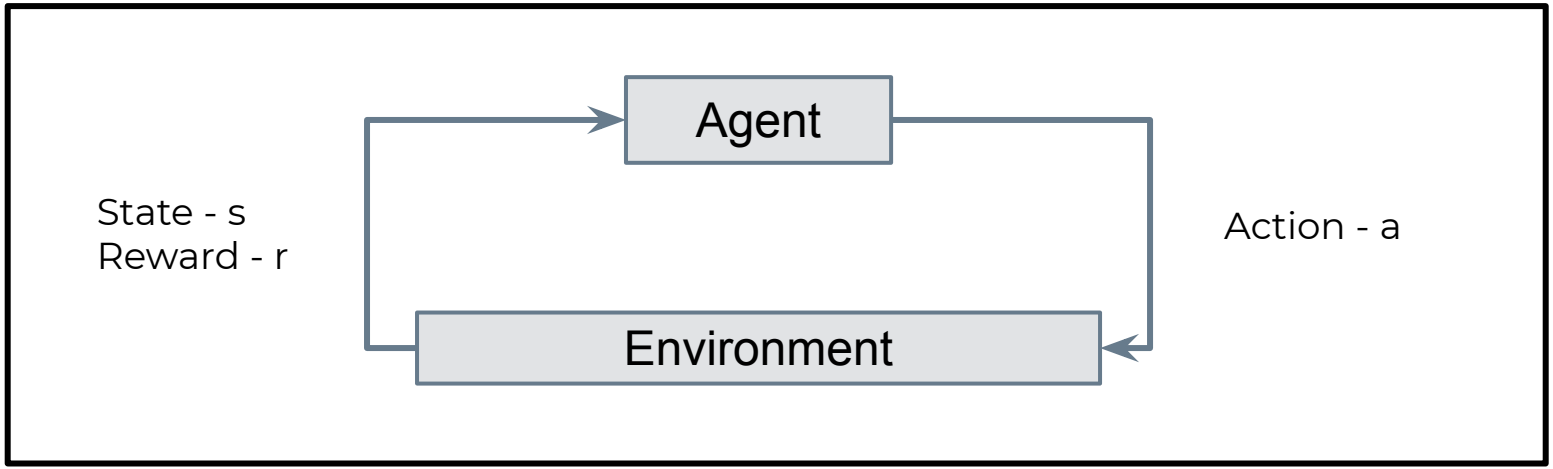
Policy Gradients

A working example

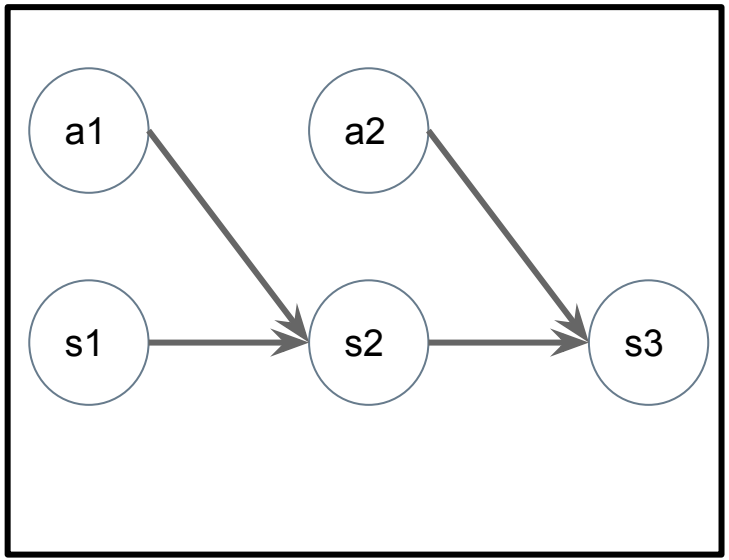
Where do I go next?

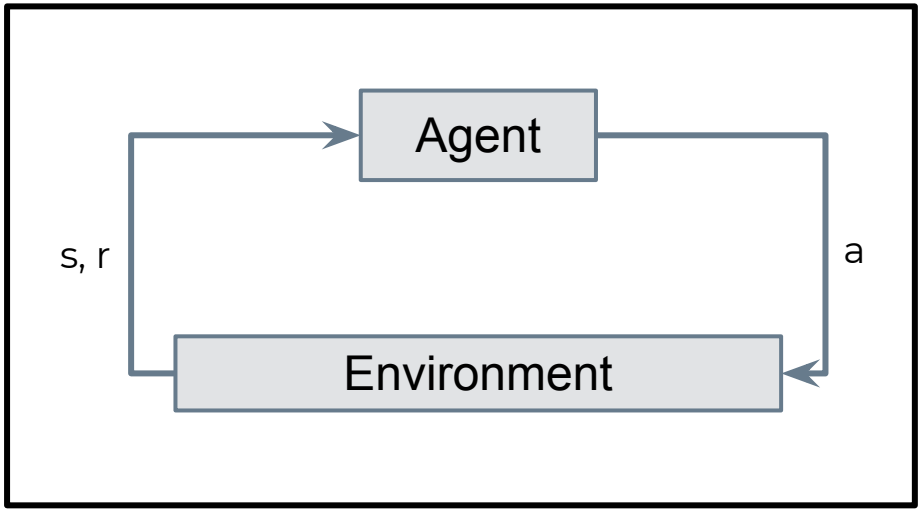
Questions are welcome at all times :)

Reinforcement Learning



Reinforcement Learning





Reinforcement Learning

1. The agent interacts with the environment and collects rewards.
2. The aim is to collect as much reward as possible.
3. Not all rewards are equal. A reward of 100$ today is probably preferred over a reward of 100$ after 1 year.
4. We *discount* the future rewards by a *discount factor*.
5. The aim is to collect as much discounted reward as possible.
6. The sum of discounted rewards is referred to as *return*

Discounted Reward

1. Let us say that the discount factor is 0.99
2. Let us say that we get a reward of 1 per step (for 10 steps).
3. Total reward = 1 +1 … + 1 (10 times) = 10
4. return = sum of discounted rewards = 1 + 0.99 *1 + 0.99*0.99 *1 + 0.99*0.99*0.99 * 1 … = 8.648
5. The agent will try to maximize the return and not the total reward.

Policy

1. A function that maps a state to an action.
2. In-practice, we will implement it using a neural network

Agenda

Sequential Decision Making Problem

Reinforcement Learning

Policy Gradients

A working example

Where do I go next?

Questions are welcome at all times :)

Policy Gradient

1. We implement the policy using a neural network.
2. We want to learn a useful policy using gradient updates.
3. We can not use the common supervised learning approaches because we do not know what the “right” action is.
4. We know that we want to maximize the returns (which depends on actions, which depend on policy).
5. But we can not directly compute the gradient because we do not know how the policy affects the state distribution.

Policy Gradient

1. Policy Gradients Theorem to the rescue.
2. We will not go over the proof of the theorem (it is a little involved).
3. Final expression:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[Q^{\pi_\theta}(s,a)\nabla_\theta \ln \pi_\theta(a \mid s)\right]$$

Let us break down the terms

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[Q^{\pi_\theta}(s,a)\nabla_\theta \ln \pi_\theta(a \mid s)\right]$$

Let us break down the terms

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[Q^{\pi_\theta}(s,a)\nabla_\theta \ln \pi_\theta(a \mid s)\right]$$

Parameters for the policy

Let us break down the terms

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[Q^{\pi_\theta}(s,a)\nabla_\theta \ln \pi_\theta(a \mid s)\right]$$

Policy

Let us break down the terms

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[Q^{\pi_\theta}(s,a)\nabla_\theta \ln \pi_\theta(a \mid s)\right]$$

Gradient for the policy

Let us break down the terms

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[Q^{\pi_\theta}(s,a)\nabla_\theta \ln \pi_\theta(a \mid s)\right]$$

How good is it to take an action a in state s.

It can be approximated by the return we observe when executing the policy.

Let us break down the terms

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[Q^{\pi_\theta}(s,a)\nabla_\theta \ln \pi_\theta(a \mid s)\right]$$

Log-likelihood of selecting action a given state s

Agenda

Sequential Decision Making Problem

Reinforcement Learning

Policy Gradients

A working example

Where do I go next?

Questions are welcome at all times :)

A working example

https://colab.research.google.com/drive/1J-s17Ppyz-8NHn7Ybe12SIIYxWo7Pjpq?usp=sharing

Agenda

Sequential Decision Making Problem

Reinforcement Learning

Policy Gradients

A working example

Where do I go next?

Questions are welcome at all times :)

Theory

1. Reinforcement Learning: An Introduction
   http://incompleteideas.net/book/the-book.html

2. https://lilianweng.github.io/lil-log/2018/02/19/a-long-peek-into-reinforcement-learning.html

3. https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.html

Code

1. Python implementation of the RL book: https://github.com/ShangtongZhang/reinforcement-learning-an-introduction
2. Spinning Up in Deep RL: https://spinningup.openai.com/en/latest/index.html
3. PyTorch-based deep reinforcement learning library https://github.com/pfnet/pfrl
4. TF-based deep reinforcement learning library https://github.com/tensorflow/agents
5. RLLib: Scalable RL https://docs.ray.io/en/latest/rllib.html

Thank you

@shagunsodhani

FACEBOOK AI