Introduction to Reinforcement Learning

DAVIDE BACCIU - BACCIU@DI.UNIPI.IT



Preliminaries

The Course (3 CFU - ~24h)

- ✓ RL Fundamentals
 - ✓ Markov Decision Processes
 - ✓ Planning by Dynamic Programming
 - ✓ Model-Free Prediction & Control
- ✓ Value Function Methods
- ✓ Policy Gradient Methods
- Exploration and Exploitation
- ✓ Deep reinforcement learning

Lectures by me Guest lectures by Maurizio Parton Student seminars

Advanced topics and applications: continual learning, meta-learning, inverse RL, RL frameworks, some case studies

Course Completion

✓ M.Sc. students

✓ Prepare a 15 minutes presentation to be given in one of the 2 student lecture dates

✓ 5 minutes Q&A on the content of the presentation

✓ Seminar content

- Read 3 relevant papers on a topic of interest for the course; summarize their content and confront the methods
- Implement 1 RL method from literature and attempt a validation on a simple application; describe the model, its implementation and the validation results

✓ Ph.D. Students

- Read 3 (or more) relevant papers on a topic of interest for the course and summarize their content in a report (6-10 pages single column, NeurIPS format)
- ✓ Sketch/propose a novel RL method/application: report your idea in sufficient detail in a short paper (6-10 pages single column, NeurIPS format)
- ✓ Implement a RL-based application and validate it: prepare a short presentation to report the results (10-15 slides describing the model, the implementation and the results)
- Contact me and agree on alternative ways (e.g. using RL in your Ph.D. project, ...)

Fill in your contact details here

<u>https://docs.google.com/spreadsheets/d/10fcU3SVYb5wOYd-</u> <u>c6uag9kk5spq6O1Q86m0fv2aSWsA/edit?usp=sharing</u>

Resources

The course webpage (elearning.di.unipi.it/course/view.php?id=227)

- ✓Course calendar & news
- ✓ Slides, video lectures, additional materials
- ✓Course assignment upload

Reference Book



Richard S. Sutton and Andrew G. Barto, Reinforcement Learning: An Introduction, Second Edition, MIT Press (available online)

A Note

Much of the content of this course and its slides are heavily based on the classical course by David Silver

https://www.davidsilver.uk/teaching/

Introduction



Positioning Reinforcement Learning

What characterizes Reinforcement Learning (vs other ML tasks)?

- ✓ No supervisor: only a *reward* signal
- Delayed asynchronous feedback
- Time matters (sequential data, continual learning)
- Agent's actions affect the subsequent data it receives (inherent non-stationarity)

Using Reinforcement Learning

- Learning to maneuver vehicles
- Learn to control robots (walking, navigation, manipulation)
- ✓ Manage portfolios
- ✓ Play games
- ✓ Discover new molecules
- End-to-end learning with discrete structures

Game Playing









Manipulation



https://www.youtube.com/watch?v=jwSbzNHGflM

Formalizing Reinforcement Learning

Rewards

 \checkmark A reward R_t is a scalar feedback signal

✓ Indicates how well agent is doing at step t

The agent's job is to maximise cumulative reward

Reinforcement learning is based on the reward hypothesis

 All goals can be described by the maximisation of expected cumulative reward

What is a Reward?

- Learning to drive a car (+ve reward for getting places safely -ve reward for crashing)
- Make a humanoid robot walk (+ve reward for forward motion, -ve reward for tripping over)
- Make a robot arm manipulate objects (+ve reward for goal achievement, -ve reward for object falling)
- ✓ Manage an investment portfolio (+ve reward for each \$ in bank)
- Play games (+=-ve reward for increasing/decreasing score)
- Discover new molecules (+ve reward for synthesizable molecule, -ve reward for toxic molecule)

Sequential Decision Making

✓ Goal: select actions to maximise total future reward

Actions may have long term consequences

Reward may be delayed

✓ It may be better to sacrifice immediate reward to gain more long-term reward

✓ Examples:

- \checkmark A financial investment (may take months to mature)
- ✓ Refuelling a helicopter (might prevent a crash in several hours)
- ✓ Blocking opponent moves (might help winning chances many moves from now)



Agent and Environment

- \checkmark At each step *t* the agent:
 - ✓ Executes action **A**_t
 - \checkmark Receives observation \boldsymbol{O}_t
 - \checkmark Receives scalar reward R_t
- ✓ The Environment:
 - \checkmark Receives action A_t
 - \checkmark Emits observation \boldsymbol{O}_{t+1}
 - \checkmark Emits scalar reward R_{t+1}

✓t increments at environment step

History and State

The history is the sequence of observations, actions, rewards

$$H_t = O_1; R_1; A_1 \dots A_{t-1}; O_r; R_t$$

✓i.e. all observable variables up to time t

✓ i.e. the sensorimotor stream of a robot or embodied agent

What happens next depends on the history:

✓ The agent selects actions

✓The environment selects observations/rewards

State S_t is the information used to determine what happens next and is a function of history

$$S_t = f(H_t)$$

Environment State

✓ The environment state S_t^e is the environment *e* private representation at time *t*

 ✓ Whatever information the environment uses to generate the next observation/reward

✓ The environment state is not usually visible to the agent (unobservable environment)

✓ Even if S_t^e is visible, it may contain irrelevant information



Agent State

- ✓ The agent state S_t^a the internal representation owned by agent a
 - ✓ Whatever information the agent uses to select next action
- The agent state is the information used by reinforcement learning algorithms
- Generally speaking a function of history

$$S_t^a = f(H_t)$$



Information (Markov) State

An information state (Markov state) contains all useful information from the history

Definition (Markov State)

A state S_t is Markov if and only if $P(S_{t+1}|S_1, ..., S_t) = P(S_{t+1}|S_t)$

✓ The future is independent of the past given present (d-separation) $H_{1:t} \rightarrow S_t \rightarrow H_{t+1:\infty}$

✓ The state is a sufficient statistics for the future

 \checkmark The environment state S_t^e is Markov

The history H_t is Markov

What's the best (agent) state model?



Fully Observable Environment

 ✓ Full observability ⇒ Agent directly observes the environment state

 $O_t = S_t^a = S_t^e$

Formally this is a Markov Decision Process (MDP)

✓ Next lecture (and much of the RL literature)



Partially Observable Environment

- \checkmark Partial observability \Rightarrow Agent indirectly observes the environment
 - A robot with camera vision only may not know absolute location
 - ✓ A trading agent only observes current prices
 - ✓ A poker player only observes public cards

✓ Formally $S_t^a \neq S_t^e$ and the problem is a Partially Observable Markov Decision Process (POMDP)

\checkmark The agent needs to build its own state representation S_t^a

• History: $S_t^a = H_t$

✓ Beliefs on environment state: $S_t^a = [P(S_t^e = s^1) ... P(S_t^e = s^N)]$

✓ A dynamic memory (RNN): $S_t^a = \sigma(W_s S_{t-1}^a + W_o O^t)$

Components of a Reinforcement Learning Agent

Key Components of an RL Agent

Policy: agent's behaviour function

- ✓ Value function: how good is each state and/or action
- ✓ Model: agent's representation of the environment

An RL agent may include one or more of the above

Policy

A policy π is the agent's behaviour
 It is a map from state s to action a
 Deterministic policy: a = π(s)
 Stochastic policy: π(a|s) = P(A_t = a|S_t = s)

Value Function



How "good" is a specific state/action for an agent?

DAVIDE BACCIU - UNIVERSITÀ DI PISA

Value Function

✓ The value function v is a predictor of future reward ✓ Used to evaluate the goodness/badness of states ✓ And therefore to select between actions, e.g $v_{\pi}(s) = \mathbb{E}_{\pi}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+2} + \cdots | S_t = s]$

Expected (discounted) future reward following policy π from state s

Model

A model predicts what the environment will do next

✓ Predict next state s' following an action a $\mathcal{P}_{ss'}^a = P(S_{t+1} = s' | S_t = s, A_t = a)$ ✓ Predict next reward

$$\mathcal{R}_s^a = \mathbb{E}[R_{t+1}|S_t = s, A_t = a]$$

A Forever Classic - The Maze Example



 Rewards: -1 per time-step
 Actions: N, E, S, W
 States: Agent location



DAVIDE BACCIU - UNIVERSITÀ DI PISA

									-
		-14	-13	-12	-11	-10	-9		
Start	-16	-15			-12		-8		
		-16	-17			-6	-7		
			-18	-19		-5			
		-24		-20		-4	-3		
		-23	-22	-21	-22		-2	-1	Goal

Maze Example (Value Function)

Numbers denote the value $v_{\pi}(s)$ for each s

Expected time to reach the goal

DAVIDE BACCIU - UNIVERSITÀ DI PISA



Maze Example (Model)

 Agent may have an internal (imperfect) model of the environment
 How actions change the state
 How much reward from each state

✓ Grid Layout: transition model $\mathcal{P}^{a}_{ss'}$

✓ Numbers: immediate reward model \mathcal{R}_s^a

Characterizing RL Agents (I)

✓Value Based

- ✓Policy (Implicit)
- ✓Value Function
- Policy Based
 Policy
 Value Function
- Actor Critic
 Policy
 Value Function
 -

Characterizing RL Agents (II)

✓ Model Free

✓Model

Policy and/or Value Function

✓Model Based

✓Model

Policy and/or Value Function



A Taxonomy

Problems within Reinforcement Learning

Learning and Planning

Two fundamental problems in sequential decision making

- ✓ Reinforcement Learning
 - The environment is initially unknown
 - The agent interacts with the environment
 - The agent improves its policy
- Planning (reasoning, introspection, search,...)
 - A model of the environment is known
 - The agent performs computations with its model (no external interaction)
 - The agent improves its policy

Atari Example – Reinforcement learning



✓ Rules of the game are unknown

 Learn directly from interactive game-play

 Pick actions on joystick, see pixels and scores

Atari Example – Planning



Rules of the game are known
Agent contains emulator (model)
If I take action a from state s:
what would the next state be?
what would the score be?

Plan ahead to find optimal policy
 e.g. tree search

Exploration Vs Exploitation

Reinforcement Learning follows a trial-and-error process
 The agent should discover a good policy
 From its experiences of the environment
 Without losing too much reward along the way

Exploration finds more information about the environment
 Exploitation exploits known information to maximise reward

Effective reinforcement learning requires to trade between exploration and exploitation

Examples

✓ Restaurant Selection

Exploitation - Go to your favourite restaurant

Exploration - Try a new restaurant

✓ Holiday planning

Exploitation – The camping site you go to since you are born

✓ Exploration – Hitchhike and follow the flow

✓ Game Playing

Exploitation - Play the move you believe is best

Exploration - Play an experimental move

Prediction & Control

Prediction: evaluate the futureGiven a policy

Control: optimise the futureFind the best policy

Gridworld Example - Prediction



What is the value function for the uniform random policy?

Gridworld Example - Control



 \mathcal{V}_*

 π_*

What is the optimal value function over all possible policies?

What is the optimal policy?

Wrap-up

Take home messages

 Reinforcement learning is a general-purpose framework for decisionmaking

- Reinforcement learning is for an agent with the capacity to act and observe
- ✓ The state is the sufficient statistics to characterize the future
 - Depends on the history of actions and observations
 - Environment state Vs Agent state

Success is measured by a scalar reward signal
 The goal is to select actions to maximise future reward (exploit)
 In order to be effective we should not forget to explore

Next Lecture

Markov Decision Processes (MDP)

✓A formal model for (observable) environments in reinforcement learning

Foundational approach to reinforcement learning

- Markov property and processes
- Discounted rewards
- ✓ Bellman expectation

Extending to partially-observable environments

Short-term Calendar (tentative)

- ✓L2 Thursday 01 Apr h. 14-16
- ✓ L3 Wednesday 7 Apr h. 14-16
- ✓L4 Tuesday 27 Apr h. 14-16
- ✓L5 Tuesday 04 May h. 14-16
- ✓L6 Monday 10 May h. 16-18
- ✓ L7-L9 Guest lecture by Maurizio Parton