Reinforcement Learning

An Overview

Agenda

- Types of learning
- What is Reinforcement Learning?
- How humans learn using Reinforcement Learning?
- Reinforcement learning framework
- Challenge for RL in real-world applications
- Component of an RL agent
- Examples of RL systems
- 3 types of RL: model-based, value-based, policy-based
- Q-learning

Agenda

- Deep Q-Networks (DQN)
- Policy Optimization (TRPO and PPO)
- AlphaZero

greatlearning
Learning for Life

What you already know?

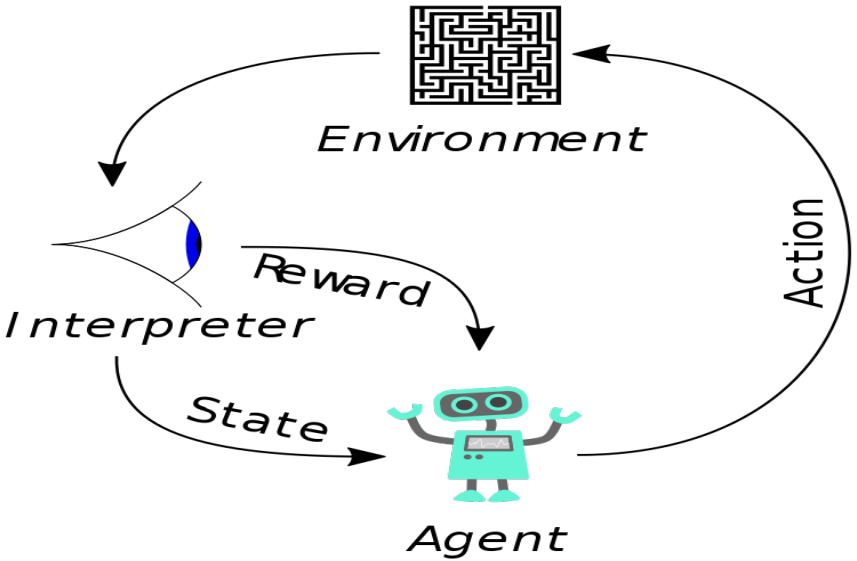
Supervised Learning

Unsupervised Learning

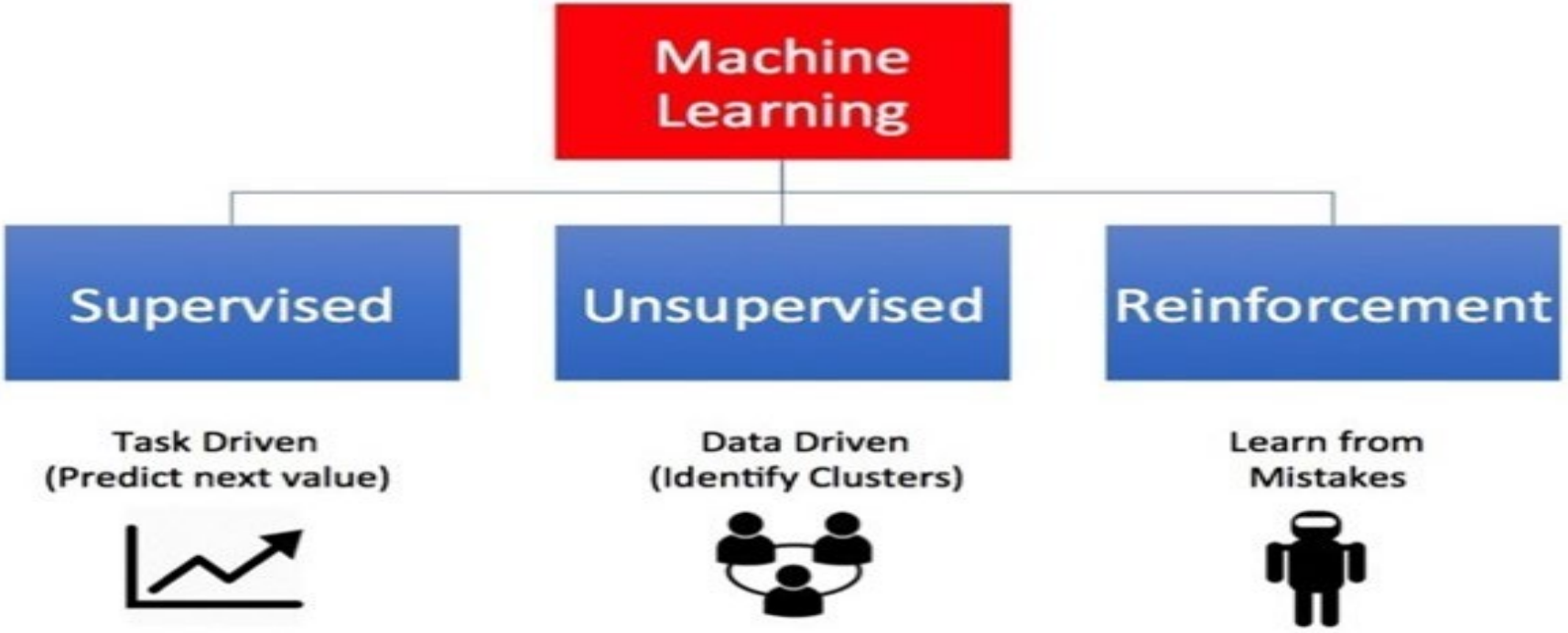
Semi-Supervised Learning

What is Reinforcement Learning?

Learning by Experience!!



Machine Learning



Patents analysis: DL Vs. Kmeans Vs. RL

https://www.lens.org/
https://patents.google.com/

Patent search:	(deep learning)	97164
Patent search:	(kmeans clustering)	620
Patent search:	(reinforcement learning)	65048

How Reinforcement Learning is different?

Supervised Learning vs Reinforcement Learning

Supervised Learning

- Training based on supervisor/label/annotation
- Feedback is instantaneous
- Time does not matter

Reinforcement Learning

- Training only based on reward signal ◦ Feedback is delayed
- Time matters
- Agent actions affect subsequent data

Deep Learning

Deep Learning

DL is a general purpose framework for representation learning

- Given an objective
- Learn representation that is required to achieve objective
- Directly from raw inputs
- Use minimal domain knowledge

Deep Reinforcement Learning

Deep reinforcement learning is the combination of reinforcement learning (RL) and deep learning. This field of research has been able to solve a wide range of complex decision-making tasks that were previously out of reach for a machine.

- AI is an agent that can solve human-level task RL defines the objective
- DL gives the mechanism
- RL + DL = general intelligence

Reinforcement Learning Framework

Agent and Environment

Image source -https://www.freecodecamp.org/news/a-brief-introduction-to-reinforcement-learning-7799af5840db/

Examples: RL

Teach a Baby, How to walk?

Teach a Dog, New trick.

Teach a Cat to get food.

Example: Teach dog new tricks

Consider the scenario of teaching a dog new tricks. The dog doesn't understand our language, so we can't tell him what to do. Instead, we follow a different strategy. We emulate a situation (or a cue), and the dog tries to respond in many different ways. If the dog's response is the desired one, we reward them with snacks. Now the next time the dog is exposed to the same situation, the dog executes a similar action with even more enthusiasm in expectation of more food. That's like learning "what to do" from positive experiences. Similarly, dogs will tend to learn what not to do when faced with negative experiences.

Teach dog new tricks -How Reinforcement Learning works in a broader sense:

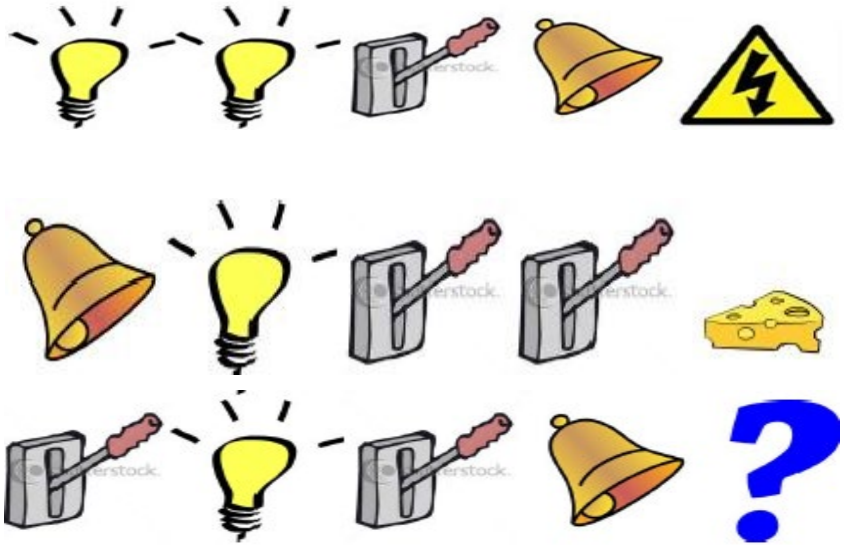
- Your dog is an "agent" that is exposed to the environment. The environment could be in your house, with you.
- The situations they encounter are analogous to a state. An example of a state could be your dog standing and you use a specific word in a certain tone in your living room
- Our agents react by performing an action to transition from one "state" to another "state," your dog goes from standing to sitting, for example.
- After the transition, they may receive a reward or penalty in return. You give them a treat! Or a "No" as a penalty.
- The policy is the strategy of choosing an action given a state in expectation of better outcomes.

Example: Teach cat new tricks

What if agent state = last 3 items in sequence?

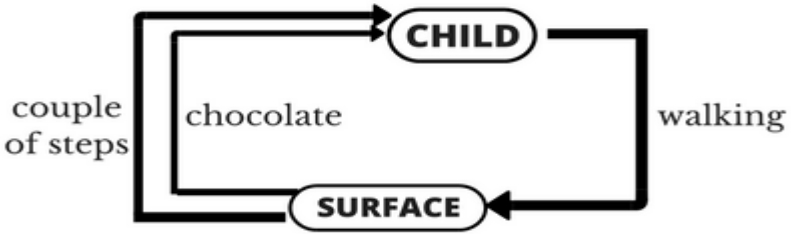
What if agent state = counts for lights, bells and livers?

What if agent state = complete sequence?



Example: Baby learning how to walk

The “problem statement” of how to walk, where the child is an agent trying to manipulate the environment (which is the surface on which it walks) by taking actions (viz walking) and he/she tries to go from one state (viz each step he/she takes) to another. The child gets a reward (let’s say chocolate) when he/she accomplishes a sub module of the task (viz taking couple of steps) and will not receive any chocolate (a.k.a negative reward)when he/she is not able to walk.



Reinforcement Learning Framework

At each step agent -

- Take **Action**
- Observe **New State**
- Receive **Reward**

RL is a general purpose framework for decision making

- RL is for an agent with the capacity to act
- Each action influences the agent's future state
- Success is measured by a scalar reward signal
- Goal: select actions to maximize future reward

Reinforcement Learning Process

Reinforcement Learning Process

In a way, Reinforcement Learning is the science of making optimal decisions using experiences. Breaking it down, the process of Reinforcement Learning involves the following simple steps:

- Observation of the environment
- Deciding how to act using some strategy
- Acting accordingly
- Receiving a reward or penalty
- Learning from the experiences and refining our strategy
- Iterate until an optimal strategy is found

Main components in RL process: Rewards

It is a scalar feedback signal(single number) which tells us that how well the agent is doing and the agent tries to maximise
the total reward
accumulated over time

Rewards: Examples

- A dog receives a reward from a trainer for completing a certain task and no reward for failing to do so
- Fly stunt manoeuvres in a helicopter (https://www.youtube.com/watch?v=0JL04JJjocc)
  - +ve reward for following desired trajectory
  - -ve reward for crashing
- Defeat the world champion at Backgammon
  - +/- ve reward for winning/losing a game
- Manage an investment portfolio
  - +ve reward for each $ in bank
- Control a power station
  - +ve reward for producing power
  - -ve reward for exceeding safety thresholds
- Make a humanoid robot walk
  - +ve reward for forward motion
  - -ve reward for falling over
- Play many different Atari games better than humans
  - +/- ve reward for increasing/decreasing score

Main components in RL process: Goals and Actions

- The goal is to maximise the total reward by taking certain actions with respect to policy.
- It is time dependent.
- The goal could be to get an immediate reward or sacrificing the immediate rewards to reach a long-term goal.

Example: Agent working on a long-term strategy in the game of chess where it sacrifices some pieces to do a checkmate on the opponent.

Main components in RL process: State

It is the summary of the information that is used to determine what happens next that is, the current situation in an environment.

Like a frame in a Pacman game is a state which describes the opponents location, ones own location,
the location of the rewards etc.

Main components in RL Agent: Policy

- The policy is the behaviour of the agent. It is a set of rules that the agent follows to get the most reward.
- It maps the state to an action.
- It can be deterministic or it can be stochastic (for a given state, the probability of taking some action).

Main components in RL Agent: Model

Its agent's representation of the environment. It can be divided into two states:

Transition: It predicts the next state. e.g., given the dynamics such as the velocity, position of an object what will the environment do next.

This equation tell $\mathcal{P}^a_{ss'} = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$ t state, s' given the current state, s and action, a.

Reward: It predicts the immediate reward. Or how much reward will the object get following an action.

This equation tel $\mathcal{R}^a_s = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$ et given the current state, s and action, a.

OpenAI Gym:

OpenAI Gym is a Python package comprising a selection of RL environments, ranging from simple “toy” environments to more challenging environments, including simulated robotics environments and Atari video game environments.

It was developed with the aim of becoming a standardized environment and benchmark for RL research.

https://gym.openai.com/

http://gym.openai.com/docs/

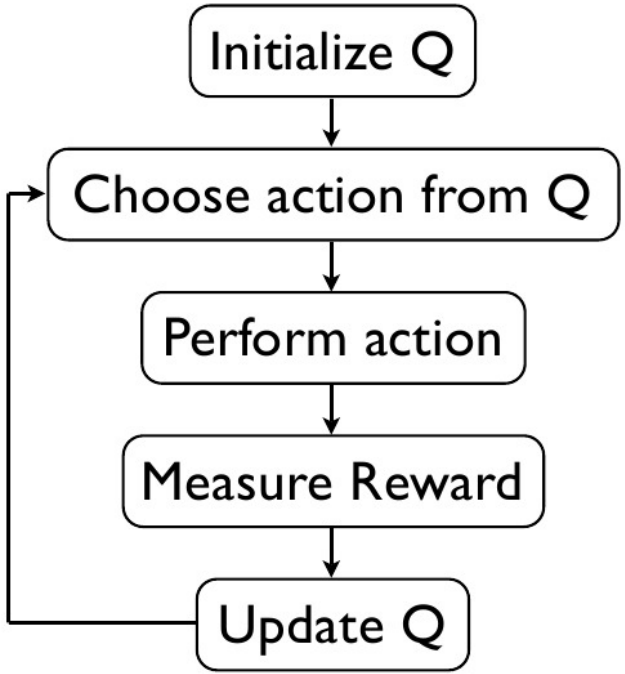
Q-Learning

Q- Learning is an off-policy, model-free learning method. Off-policy means, it does not require to follow a specific policy, the agent's actions could be random and despite this, it can find an optimal policy.

What's 'Q'?

The 'q' in q-learning stands for quality. Quality in this case represents how useful a given action is in gaining some future reward.

Q-learning: Algorithm



Q-learning: Algorithm

For an environment, we build a table called Q-table which has the dimensions SxA where S and A are the number of states and actions respectively. For every state, there are actions and the likeliness of choosing a particular action depends on the values in Q-table called as state-action value.

Initially the values of the Q-table are 0. An action is chosen for a state. If that action gives a good reward for the next state then the Q-value for the state-action is increased otherwise, the Q-value is decreased.

$$R = \begin{array}{c} \\ \text{State} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \text{Action} \\ \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} -1 & -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & 0 & -1 & 100 \\ -1 & -1 & -1 & 0 & -1 & -1 \\ -1 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & -1 & 0 & -1 & 100 \\ -1 & 0 & -1 & -1 & 0 & 100 \end{bmatrix} \end{array}$$

Q-learning: Algorithm

The updation of the Q-values are done using the following equation:

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

Here s(t) and a(t) are the previous state and action and s(t+1) and r(t+1)are the current state and reward.The learned value is the target and the old value is the prediction and the difference between them is the error. We then fix the old value using this error with some amount called as learning rate.

Smart Taxi

Develop an agent to learn Taxi Driver to pick-up passenger and drop passenger automatically

Example: Smart Taxi

The Smart Taxi's task is to pick up the passenger at one location and drop them off in another. Here are a few things that we like our Smart Taxi to take care of:

- Drop off the passenger to the right location.
- Save passenger's time by taking minimum time possible to drop off
- Take care of passenger's safety and traffic rules

Smart Taxi - Rewards

The agent (the imaginary driver) is going to learn how to control the cab by trial experiences in the environment, we need to decide the rewards and/or penalties and their magnitude accordingly. Here a few points to consider:

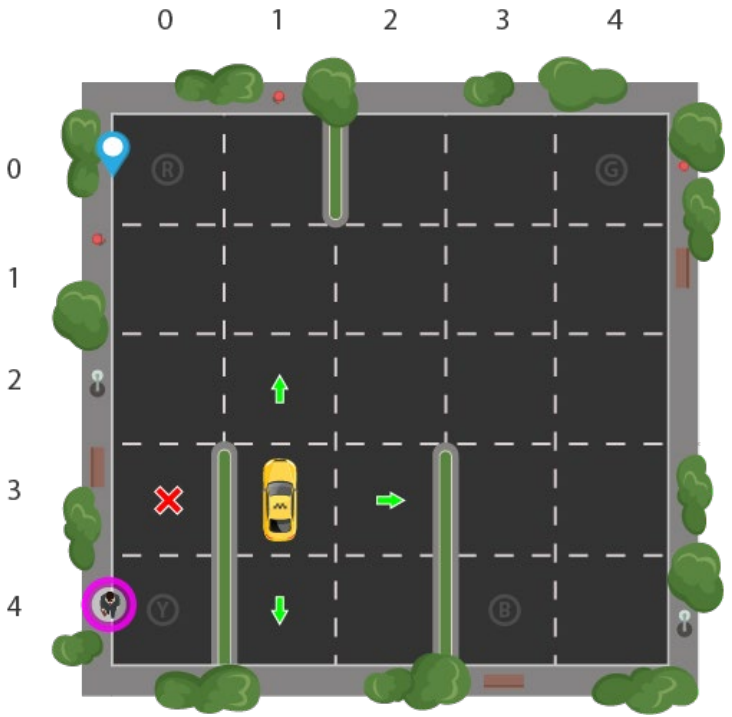
- receive a high positive reward for a successful dropoff because this behavior is highly desired
- be penalized if it tries to drop off a passenger in wrong locations
- get a slight negative reward for not making it to the destination after every time-step. "Slight" negative because we would prefer our agent to reach late instead of making wrong moves trying to reach to the destination as fast as possible

Smart Taxi - States

In Reinforcement Learning, the agent encounters a state, and then takes action according to the state it's in.

The State Space is the set of all possible situations our taxi could inhabit. The state should contain useful information the agent needs to make the right action.

Let's say we have a training area for our Smart Taxi where we are teaching it to transport people in a parking lot to four different locations (R, G, Y, B):



Smart Taxi - Actions

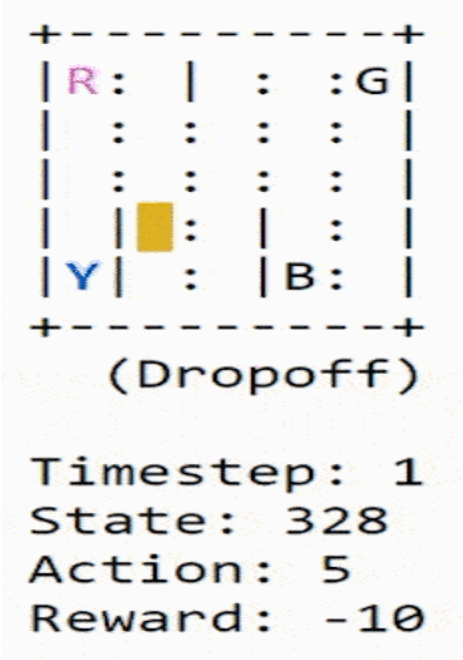
The agent encounters one of the 500 states and it takes an action. The action in our case can be to move in a direction or decide to pickup/dropoff a passenger.

In other words, we have six possible actions: **south, north, east, west, pickup, dropoff**

This is the action space: the set of all the actions that our agent can take in a given state.

We observe that the taxi cannot perform certain actions in certain s**tates due to walls.** In environment's code, we will simply provide a -1 penalty for every wall hit and the taxi won't move anywhere. This will just rack up penalties causing the taxi to consider going around the wall.

Smart Taxi



Q-table

Initialized

Q-Table		Actions					
		South (0)	North (1)	East (2)	West (3)	Pickup (4)	Dropoff (5)
States	0	0	0	0	0	0	0
	. . .	. . .	. . .	. . .	. . .	. . .	. . .
	327	0	0	0	0	0	0
	. . .	. . .	. . .	. . .	. . .	. . .	. . .
	499	0	0	0	0	0	0

Training

Q-Table		Actions					
		South (0)	North (1)	East (2)	West (3)	Pickup (4)	Dropoff (5)
States	0	0	0	0	0	0	0
	. . .	. . .	. . .	. . .	. . .	. . .	. . .
	328	-2.30108105	-1.97092096	-2.30357004	-2.20591839	-10.3607344	-8.5583017
	. . .	. . .	. . .	. . .	. . .	. . .	. . .
	499	9.96984239	4.02706992	12.96022777	29	3.32877873	3.38230603

Hands-on: Smart Taxi

Utilize the following functions:

- env.reset: resets the environment and returns a random initial state.
- env.step(action): advances the environment by one timestep.
- env.render: renders one frame of environment (helpful in visualization)

The environment returns:

- observation: an environment-specific object representing your observation of the environment
- reward: the amount of reward/score achieved by the previous action
- done: indicates whether it is time to reset the environment again, i.e., the agent has achieved the goal
- info: diagnostic info such as performance and latency useful for debugging purposes

greatlearning
Learning for Life

Develop an agent to learn to walk in a frozen lake automatically on its own

Hands-on: Frozen Lake

Types of Reinforcement Learning Approach

- Value-Based
- Policy-Based
- Model-Based

Challenges within RL

- Learning and Planning
- Exploration and Exploitation

More Applications

- Flying Helicopter - https://www.youtube.com/watch?v=0JL04JJjocc
- Driving - https://www.youtube.com/watch?v=0xo1Ldx3L5Q
- Robot - https://www.youtube.com/watch?v=370cT -OAzzM
- Google Cuts Its Giant Electricity Bill With DeepMind -Powered AI- http://www.bloomberg.com/news/articles/2016 -07-19/google-cuts-its-giantelectricity-bill-with-deepmind-powered-ai
- Text Generation- https://www.youtube.com/watch?v=pbQ4qe8EwLo

References

https://www.csie.ntu.edu.tw/~yvchen/f106-adl/doc/171123+171127_DeepRL.pdf

https://skymind.ai/wiki/deep-reinforcement-learning

https://www.freecodecamp.org/news/a-brief-introduction-to-reinforcement-learning-7799af5840db/