Reinforcement Learning Revision - Things to Know

Pavlos Andreadis

April 2018

1 Generally

- What are the following?
  - action space;
  - state space;
  - transition function;
  - reward function;
  - discount factor;
  - policy;
  - return;
  - value function (state-value and action-value functions);
- Explain the Markov Property (with equations and conceptually).
- What does it mean to evaluate a policy? Equivalently, what is the goal of Policy Evaluation? Or what is th RL prediction task?
- What is the Reinforcement Learning (RL) control problem?
- Write a reward or transition function in matrix format.
- Draw a transition graph.
- Write a policy in matrix format.
- Execute a given policy on paper.
- What are the differences between episodic and continuing tasks, and what role does the discount factor play in either of them?
- What is an absorbing/terminating state and what is the return from such a state?

- Explain the formulas defining the state (V) and action (Q) value functions.
- Explain the Bellman Equations, and the Bellman Optimality Equations, for V and Q.
- Give the relation between V and Q under different conditions (stochastic versus deterministic policy, optimal versus non-optimal policy).
- Value functions are defined for a specific policy.
- There exists a unique solution to the Bellman equations.
- What do we mean by 'tabular' RL?
- What does it mean for an RL algorithm to be model-based or model-free?
- For the tabular case, and if we known the model, we can solve the Bellman Equations as a linear programming problem of $|S|$ number of equations and unknowns (where $|S|$ is the number of states).
- What is a backup operation in the context of RL?
- What is a "full backup"?
- Explain the impact of the learning rate, generally and specifically for the tabular case.
- What is "sampling"?
- What does it mean to "bootstrap"?
- A Markov Decision Process (MDP) with a defined policy defines a Markov Chain (what is a Markov Chain?).
- Explain the concept of Generalised Policy Iteration (GPI).
- Are any GPI algorithms guarantee to converge to an optimal policy? No.
- Model a described problem as an MDP/SMDP/POMDP, identify what if any information is missing or ill-specified, and assume values for any missing info such that they are consistent with the problem. Explain those assumptions in terms of how a learned policy would behave online.
- Explain the Difference between Temporal Difference (TD), TD(λ), Monte Carlo, and Dynamic Programming, with the use of backup diagrams.
- Why would you use:
  - Dynamic Programming;
  - Monte Carlo;
  - Temporal Difference?

- What algorithms are guaranteed to converge and under what conditions?
- Given the description of a problem, pick an algorithm to solve it, and argue for your choice, while still pointing out any negatives.
- Can we generally define an optimal policy if we are given the respective state-value function?

2 Dynamic Programming

- Dynamic Programming algorithms are model-based.
- Understand Iterative Policy Evaluation in detail and be able to explain it and run a few steps.
- Explain the Policy Improvement Theorem.
- Do a policy improvement step (greedy or ϵ-greedy).
- Understand Policy Iteration Iteration in detail and be able to explain it and run a few steps.
- Understand Value Iteration in detail and be able to explain it and run a few steps.
- Explain how Value Iteration relates to Policy Iteration.
- Explain Asynchronous Dynamic Programming. Recognise an example algorithm as being an instance of Asynchronous Dynamic Programming.

3 Monte Carlo

- Monte Carlo (MC) RL is model-free.
- MC RL does not bootstrap.
- MC uses samples from real-world experience or simulation.
- Though a ”simulator” is a model, it does not necessarily explicitly define a transition and reward function (which is what we are concerned with when calling a procedure ”model-free” or ”model-based”).
- Explain the difference between first and every visit MC. Given a set of trajectories, produce the samples under either method.
- Given one or more sampled trajectories, execute a requested MC algorithm for prediction or control.
- Explain the idea behind exploring starts.

- Explain MC control as an instantiation of Generalised Policy Iteration.
- Infinite Sampling and Exploring All States are not realistic assumptions in most real applications.
- Explain the concepts of on-policy and off-policy.
- Explain ϵ-greedy action selection/policies.
- Do a step of policy improvement for an ϵ-greedy policy.
- Understand On-Policy MC Control in detail and be able to explain it and run a few steps (given example trajectories).
- Understand Off-Policy MC Control in detail and be able to explain it and run a few steps (given example trajectories).
- What are the Estimation and Behaviour policies, in the context of an off-policy RL algorithm?
- Understand the incremental implementation of Off-Policy MC control.
- Different ways to achieve exploration in MC (and RL in general):
  - on-policy that still explores
  - off-policy (decoupling exploration from evaluation)
  - exploring starts
- Understand MC sample-averaging as learning rate.
- MC is more robust towards violations of the Markov assumption, since it doesn't bootstrap.

4 Temporal Difference

- TD learning does not require a model.
- TD learning bootstraps.
- Understand TD(0) in detail and be able to explain it and run a few steps.
- What is the "target" of an update procedure/backup?
- Why might TD converge faster than MC? (It moves towards a better estimate that takes transitions into account, or it bootstraps. MC better matches the training data).
- Explain on-policy and off-policy TD control differences.
- Understand SARSA in detail and be able to explain it and run a few steps (given example trajectories).

- Understand Q-Learning in detail and be able to explain it and run a few steps (given example trajectories).
- Explain the difference between the on-line performance of SARSA and Q-Learning (witrh or without an example application).

5 Eligibility Traces

- Compute an n-step return (also show in backup diagram).
- n-step return when episode ends in less than $n + 1$ steps is full return.
- Understand n-step TD evaluation (this is not TD(λ)!!) in detail and be able to explain it and run a few steps (given example trajectories).
- Explain complex backups, and give an example.
- Explain the formula for λ-return.
- Understand the λ-return algorithm in detail and be able to explain it and run a few steps (given example trajectories).
- λ-return is not typically called $TD(\lambda)$.
- What is the main problem with the λ return algorithm (and n-step TD)?
- Explain the relation between the forward and backward view of TD(λ).
- explain the concept of eligibility traces.
- Iteratively update the eligibility traces of states given a sampled trajectory.
- Explain the λ parameter, and compare it with the discount factor.
- Understand SARSA(λ) in detail and be able to explain it and run a few steps (given example trajectories).
- What is the difficulty with implementing Q(λ)?

6 Function Approximation

- Explain the concept of Function Approximation for value functions in RL.
- What are some reasons why we might choose to use function approximation in RL?
- Given the description of an RL problem, suggest a set of features to describe states with.
- Given the representation of a linear value function as weights, and a state vector, compute the value function at that state.

- Give and explain the general formula for doing backups with function approximation.
- Do an update given a target, a proposed RL algorithm, and a linear representation a the value function.
- TD and MC can be applied to RL problems with function approximation, as long as we can compute or estimate partial derivatives of the value function over the approximate value function parameters.
- The MC solution is gradient descent.
- The TD solution approximates/is like gradient descent.
- Explain the concept of Coarse Coding.

7 Semi-MDPs and Options

- Explain the concept of Reward Shaping.
- Main problem with reward shaping is that it changes the problem solved.
- What is a Semi-MDP (SMDP)?
- For SMDPs, the Markov assumption holds at states -when- a decision is made. The Markov assumption does not necessarily hold in-between time-steps.
- The time between decisions can be discrete or continuous.
- Explain the Bellman Equations for SMDPs.
- We can run DP, MC, and TD algorithms on SMDPs with few changes.
- write the Q-Learning update for a discrete time SMDP.
- Explain the role of $\epsilon^{-l\tau}$ for continuous time SMDPs.
- Explain the concept of Options.
- Define an option (the tuple).
- Given a deterministic policy over options, and their definitions, write out the state-action trajectory as if the policy was being executed.
- Give an example of a hierarchy of options.
- Given a problem description, suggest a hierarchy of options.
- We can learn optimal policies over options, but these are only optimal given the constraints the options impose on behaviour. Procedures ove SMDPs have the same guarantees as over MDPs, but with solutions constrained by the options.

- Given the definition of an option, and a set of states, answer queatios such as:
  - Can the option terminate in this state?
  - Can the option be initiated in this state?
- How do options define an SMDP?
- (Optimal) Bellman Equations for Options.
- Options are a good way of introducing hierarchies to RL problems.

8 Inverse Reinforcement Learning

- We can use supervised learning to learn a policy directly, bu using observations of an agent's behaviour (i.e. trajectories). This is "Behavioural Cloning".
- Explain the concept of Inverse RL (IRL).
- Why might we choose to do IRL instead of Behavioural Cloning?
- We can solve the IRL problem, given an observed trajectory or a policy, by formulating a linear program:
  - Explain the tabular-case formulation
  - Explain the linear function approximation case formulation.
- There are multiple reward functions that can explain a given behaviour (trajectory or policy), so the linear programs include constraints to produce something more intuitive/useful. Explain how different requirements inform our linear program objective function (maximum difference on Q, and normalisation).
- Why do we need a model for formulating the linear function approximation linear program?

9 Partially Observable MDPs

- What does partial observability mean in the context of RL?
- Explain what a belief state is.
- Give and explain the Bellman Equations for POMDPs.
- Compute the value of a belief state given the state value function.
- The main problem with POMDPs is that the value function is defined over probability distributions.

- For finite worlds (state, action, measurement spaces, observations, finite horizons), we can represent the value function as a piece-wise linear function.
- Given a POMDP model, a current belief state, and an observation, compute the posterior belief state.
- Read a piece-wise linear value function graph.
- Explain the concept of "pruning" and its necessity for POMDP Value Iteration.
- Given a small POMDP problem, and a belief state, give an optimal policy for a short horizon (possibly implicitly defined by terminating/absorbing states).

10 Multi-Agent RL

- Read and write payoff matrices for 2-player games.
- What are zero-sum and fully cooperative games?
- Understand and explain pure and mixed strategies.
- Define a Stochastic Game.
- Explain Shapley's [1953] Value Iteration algorithm.
- Given a problem description of an iterative zero-sum game, and a description of the current state, compute an optimal policy for your agent using minimax search.