Reinforcement Learning

Lecture 6: Temporal-difference learning

Robert Lieck

Durham University

Lecture overview

Lecture covers chapter 6 and chapter 12 in Sutton & Barto [1]

1. **Temporal-difference learning**
   - dopamine and reward predictor error
   - definition
   - behaviour example
2. **SARSA (on-policy TD control)**
3. **Off-policy learning**
4. **Q-learning (off-policy TD control)**
5. **TD(λ)**

Definition: dopamine and RPE

In the early '90s, scientists were struggling to understand the role of dopamine [2]. Dopamine neurons are clustered in the midbrain and send out signals somehow related to 'reward' especially to the frontal lobe (planning and problem solving).

In mid '90s, scientists [3] connected dopamine to reward prediction errors (RPE) proposing the brain uses a TD learning algorithm. Since then, RPE theory has been tested and validated thousands of times.

Dopamine pathway

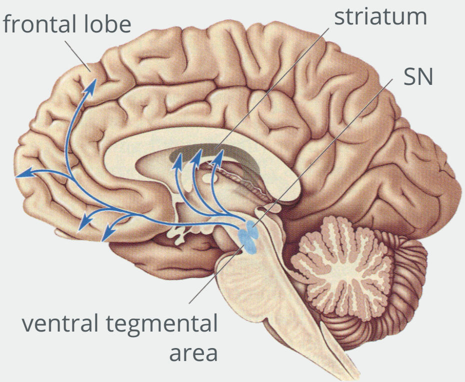


Figure from Wolters Kluwer Health.

Temporal-difference learning

Overview: TD learning

Temporal-difference learning learns from **episodes of experience**:

1. It's also **model-free** (requires no knowledge of MDP transitions/rewards)
2. Learns from *incomplete* episodes by bootstrapping
3. Updates an estimate towards another estimate

Quote:

Sutton & Barto [1] write:

"If one had to identify one idea as central and novel to reinforcement learning, it would undoubtedly be temporal-difference (TD) learning."

Follow along in Colab:

Recap: MC prediction, incremental updates

Putting this together, we sample episodes from experience under policy π

$$S_1, A_1, R_2, S_2, A_2, ..., S_T \sim \pi,$$

and every time we visit a state, we're going to increase a visit counter, then we will use our running mean:

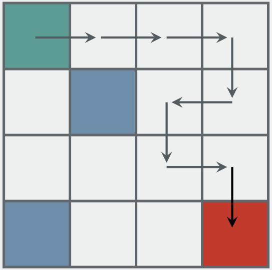
$$N(S_t) \leftarrow N(S_t) + 1$$

$$V(S_t) \leftarrow V(S_t) + \frac{1}{N(S_t)}(G_t - V(S_t))$$

It's common to also just track a running mean and forget about old episodes:

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t))$$

Example: episode



Temporal-difference learning definition

Definition: temporal-difference learning

Incremental every-visit Monte Carlo:

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t))$$

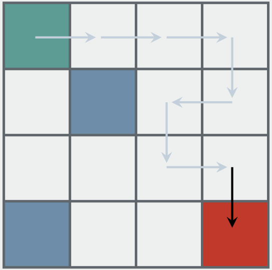
In TD(0), we update our value function $V(S_t)$ towards an estimate of the return:

$$V(S_t) \leftarrow V(S_t) + \alpha(\underbrace{R_{t+1} + \gamma V(S_{t+1})}_{\text{TD target}} - V(S_t))$$

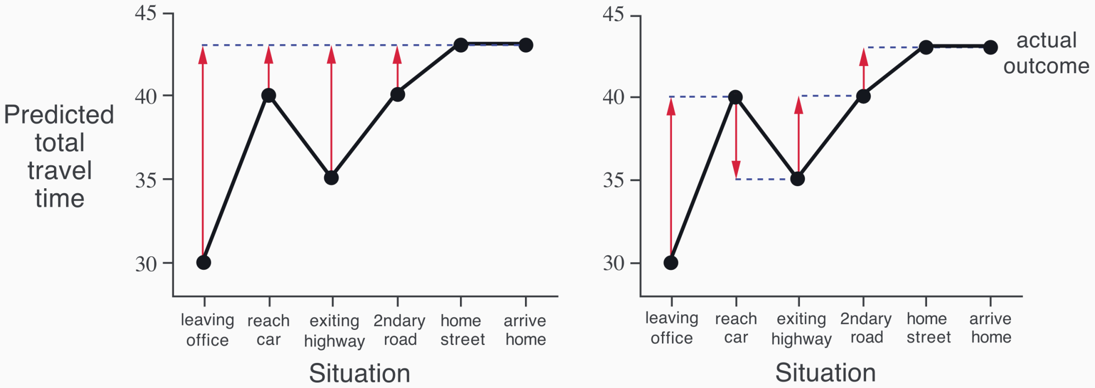
the part inside the brackets is called the TD error δ_t:

$$\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$$

Example: TD(0) learning



Temporal-difference learning behaviour example



Graphs from Sutton & Barto [1]

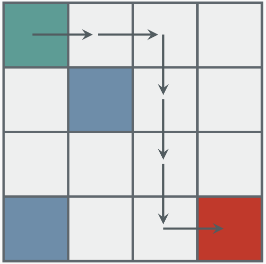
Temporal-difference learning SARSA (on-policy TD control)

The SARSA update pattern is: $Q(S,A) \leftarrow Q(S,A) + \alpha(R + \gamma Q(S',A') - Q(S,A))$

Algorithm: SARSA for on-policy TD control

```
Q = np.zeros([n_states, n_actions])

for episode in range(num_episodes):
  s = env.reset()
  pa = random_ϵ_greedy_policy(Q, ϵ, s, n_actions)
  a = np.random.choice(np.arange(len(pa)), p=pa)
  for t in itertools.count():
    s', reward, done, _ = env.step(a)
    pa' = random_ϵ_greedy_policy(Q,ϵ, s', n_actions)
    a'  = random.choice(arange(len(pa')), p=pa')
    Q[s][a] += α * (reward + γ*Q[s'][a'] − Q[s][a])
    if done:
      break
    s = s'
    a = a'
```



Off-policy learning definition

Definition: off-policy learning

In contrast to on-policy 'learning on the job', off-policy learning is where you can evaluate policies $\pi(a|s)$ different to the one currently being followed $\mu(a|s)$:

- learn from observing other agents
- learn about multiple policies while following one policy
- reuse experience from the past
- learn the optimal policy while exploring other policies

Example: relaxing for dopamine

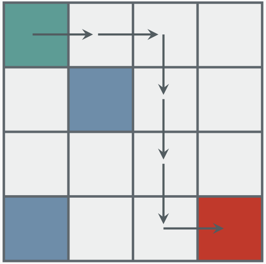
Figure from Callaway Resort & Gardens.

Q-learning update: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$

Algorithm: Q-learning (off-policy TD control)

```
Q = np.zeros([n_states, n_actions])

for episode in range(num_episodes):
  s = env.reset()
  for t in itertools.count():
    pa = random_ϵ_greedy_policy(Q, ϵ, s, n_actions)
    a  = random.choice(arange(len(pa)), p=pa)
    s', reward, done, _ = env.step(a)
    Q[s][a] += α*(reward + γ*max(Q[s'][:])−Q[s][a])
    if done:
      break
    s = s'
```



Temporal-difference learning TD(λ)

Definition: TD(λ)

In TD(0), we update our value function $V(S_t)$ towards an estimate of the return:

$$V(S_t) \leftarrow V(S_t) + \alpha(\underbrace{R_{t+1} + \gamma V(S_{t+1})}_{\text{TD target}} - V(S_t))$$

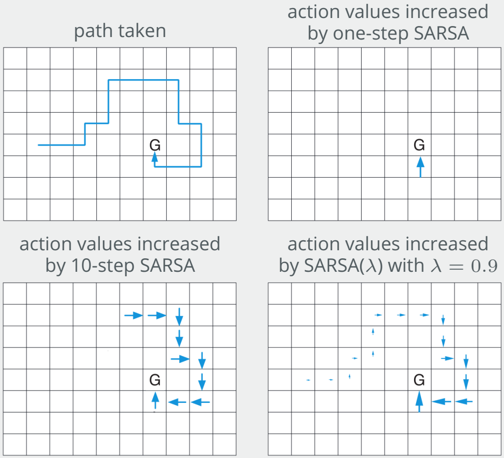
In TD(λ) we use a trace-decay parameter λ that averages n-step updates to a return:

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t^\lambda - V(S_t)),$$

where:

$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_t^n$$

Example: TD(λ) grid world [1]



Take Away Points

Summary

In summary, TD learning:

- is model-free
- behaves like dopamine in the brain
- learns from experiences rather than complete episodes
- is computationally efficient
- Q-learning converges to the optimal action-value function
- there is a spectrum of approaches between TD and MC learning

References I

[1] Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction (second edition). Available online. MIT press, 2018.

[2] Tomas Ljungberg, Paul Apicella, and Wolfram Schultz. “Responses of monkey dopamine neurons during learning of behavioral reactions”. In: Journal of neurophysiology 67.1 (1992), pp. 145–163.

[3] P Read Montague, Peter Dayan, and Terrence J Sejnowski. “A framework for mesencephalic dopamine systems based on predictive Hebbian learning”. In: Journal of neuroscience 16.5 (1996), pp. 1936–1947.

[4] David Silver. Reinforcement Learning lectures. https://www.davidsilver.uk/teaching/. 2015.