TD Learning

Shusen Wang

Outline

1. Mathematically derive the TD target.
2. TD learning algorithms:
    - Sarsa algorithm for learning Q_π.
    - Q-learning algorithm for learning $Q^\star$.
3. Multi-step TD target.

Derive TD Target

Discounted Return

Definition of discounted return:

- $U_t = R_t + \underbrace{\gamma \cdot R_{t+1} + \gamma^2 \cdot R_{t+2} + \gamma^3 \cdot R_{t+3} + \gamma^4 \cdot R_{t+4} + \cdots}_{= \gamma \cdot (R_{t+1} + \gamma \cdot R_{t+2} + \gamma^2 \cdot R_{t+3} + \gamma^3 \cdot R_{t+4} + \cdots)}$

Discounted Return

- $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot R_{t+2} + \gamma^3 \cdot R_{t+3} + \gamma^4 \cdot R_{t+4} + \cdots$

$$= R_t + \gamma \cdot (\underbrace{R_{t+1} + \gamma \cdot R_{t+2} + \gamma^2 \cdot R_{t+3} + \gamma^3 \cdot R_{t+4} + \cdots}_{= U_{t+1}})$$

Discounted Return

Identity: $U_t = R_t + \gamma \cdot U_{t+1}$.

- $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot R_{t+2} + \gamma^3 \cdot R_{t+3} + \gamma^4 \cdot R_{t+4} + \cdots$

  $= R_t + \gamma \cdot \underbrace{(R_{t+1} + \gamma \cdot R_{t+2} + \gamma^2 \cdot R_{t+3} + \gamma^3 \cdot R_{t+4} + \cdots)}_{= U_{t+1}}$

Derive TD Target

Identity: $U_t = R_t + \gamma \cdot U_{t+1}$.

- Assume R_t depends on (S_t, A_t, S_{t+1}).
- $Q_\pi(s_t, a_t) = \mathbb{E}[U_t | s_t, a_t]$

Derive TD Target

Identity: $U_t = R_t + \gamma \cdot U_{t+1}$.

- Assume R_t depends on (S_t, A_t, S_{t+1}).
- $Q_\pi(s_t, a_t) = \mathbb{E}[U_t | s_t, a_t]$

$$= \mathbb{E}[R_t + \gamma \cdot U_{t+1} | s_t, a_t]$$

$$= \mathbb{E}[R_t | s_t, a_t] + \gamma \cdot \mathbb{E}[U_{t+1} | s_t, a_t]$$

It is taken w.r.t. $(S_{t+1}, A_{t+1}), (S_{t+2}, A_{t+2}), \cdots$

Derive TD Target

Identity: $U_t = R_t + \gamma \cdot U_{t+1}$.

- Assume R_t depends on (S_t, A_t, S_{t+1}).
- $Q_\pi(s_t, a_t) = \mathbb{E}[U_t | s_t, a_t]$

$$= \mathbb{E}[R_t + \gamma \cdot U_{t+1} | s_t, a_t]$$

$$= \mathbb{E}[R_t | s_t, a_t] + \gamma \cdot \mathbb{E}[U_{t+1} | s_t, a_t]$$

$$\mathbb{E}[U_{t+1} | s_t, a_t] = \mathbb{E}[Q_\pi(S_{t+1}, A_{t+1}) | s_t, a_t]$$

Derive TD Target

Identity: $U_t = R_t + \gamma \cdot U_{t+1}$.

- Assume R_t depends on (S_t, A_t, S_{t+1}).
- $Q_\pi(s_t, a_t) = \mathbb{E}[U_t | s_t, a_t]$

$$= \mathbb{E}[R_t + \gamma \cdot U_{t+1} | s_t, a_t]$$

$$= \mathbb{E}[R_t | s_t, a_t] + \gamma \cdot \mathbb{E}[U_{t+1} | s_t, a_t]$$

$$\mathbb{E}[U_{t+1} | s_t, a_t] = \mathbb{E}[Q_\pi(S_{t+1}, A_{t+1}) | s_t, a_t]$$

Q_π eliminates all the future states and actions from time $t+1$.

Derive TD Target

Identity: $U_t = R_t + \gamma \cdot U_{t+1}$.

- Assume R_t depends on (S_t, A_t, S_{t+1}).
- $Q_\pi(s_t, a_t) = \mathbb{E}[U_t | s_t, a_t]$

$$= \mathbb{E}[R_t + \gamma \cdot U_{t+1} | s_t, a_t]$$

$$= \mathbb{E}[R_t | s_t, a_t] + \gamma \cdot \mathbb{E}[U_{t+1} | s_t, a_t]$$

$$\mathbb{E}[U_{t+1} | s_t, a_t] \quad = \quad \mathbb{E}[Q_\pi(S_{t+1}, A_{t+1}) | s_t, a_t]$$

The expectation is taken w.r.t. only S_{t+1} and A_{t+1}.

Derive TD Target

- Assume R_t depends on (S_t, A_t, S_{t+1}).

- $Q_\pi(s_t, a_t) = \mathbb{E}[U_t | s_t, a_t]$

$$= \mathbb{E}[R_t + \gamma \cdot U_{t+1} | s_t, a_t]$$

$$= \mathbb{E}[R_t | s_t, a_t] + \gamma \cdot \mathbb{E}[U_{t+1} | s_t, a_t]$$

$$= \mathbb{E}[R_t | s_t, a_t] + \gamma \cdot \mathbb{E}[\underline{Q_\pi(S_{t+1}, A_{t+1})} | s_t, a_t].$$

Derive TD Target

Identity: $Q_\pi(s_t, a_t) = \mathbb{E}[R_t + \gamma \cdot Q_\pi(S_{t+1}, A_{t+1}) | s_t, a_t]$, for all π.

- Assume R_t depends on (S_t, A_t, S_{t+1}).
- $Q_\pi(s_t, a_t) = \mathbb{E}[U_t | s_t, a_t]$

$$= \mathbb{E}[R_t + \gamma \cdot U_{t+1} | s_t, a_t]$$

$$= \mathbb{E}[R_t | s_t, a_t] + \gamma \cdot \mathbb{E}[U_{t+1} | s_t, a_t]$$

$$= \mathbb{E}[R_t | s_t, a_t] + \gamma \cdot \mathbb{E}[Q_\pi(S_{t+1}, A_{t+1}) | s_t, a_t].$$

Derive TD Target

Identity: $Q_\pi(s_t, a_t) = \mathbb{E}[R_t + \gamma \cdot Q_\pi(S_{t+1}, A_{t+1}) | s_t, a_t]$, for all π.

- We do not know the expectation.
- Approximate it using Monte Carlo (MC).

Derive TD Target

Identity: $Q_\pi(s_t, a_t) = \mathbb{E}[R_t + \gamma \cdot Q_\pi(S_{t+1}, A_{t+1}) | s_t, a_t]$, for all π.

y_t is its MC approximation.

- Let (s_{t+1}, r_t) be an observation of (S_{t+1}, R_t).
- Sample $a_{t+1} \sim \pi(\cdot | s_{t+1})$.
- TD target: $y_t = r_t + \gamma \cdot Q_\pi(s_{t+1}, a_{t+1})$.

Derive TD Target

Identity: $Q_\pi(s_t, a_t) = \mathbb{E}[R_t + \gamma \cdot Q_\pi(S_{t+1}, A_{t+1}) | s_t, a_t]$, for all π.

y_t is its MC approximation.

TD learning: Encourage $Q_\pi(s_t, a_t)$ to approach y_t.

Sarsa

Tabular Version

- We want to learn $Q_\pi(s, a)$.
- Suppose there are finite number of states and actions.
- Draw a table and learn the entries.

	Action a_1	Action a_2	Action a_3	Action a_4	$\cdots$
State s_1					
State s_2					
State s_3					
$\vdots$					

Sarsa (tabular version)

- Observe (s_t, a_t, r_t, s_{t+1}).
- Sample $a_{t+1} \sim \pi(\cdot\,|s_{t+1})$, where π is a policy function.
- TD target: $y_t = r_t + \gamma \cdot Q_\pi(s_{t+1}, a_{t+1})$.

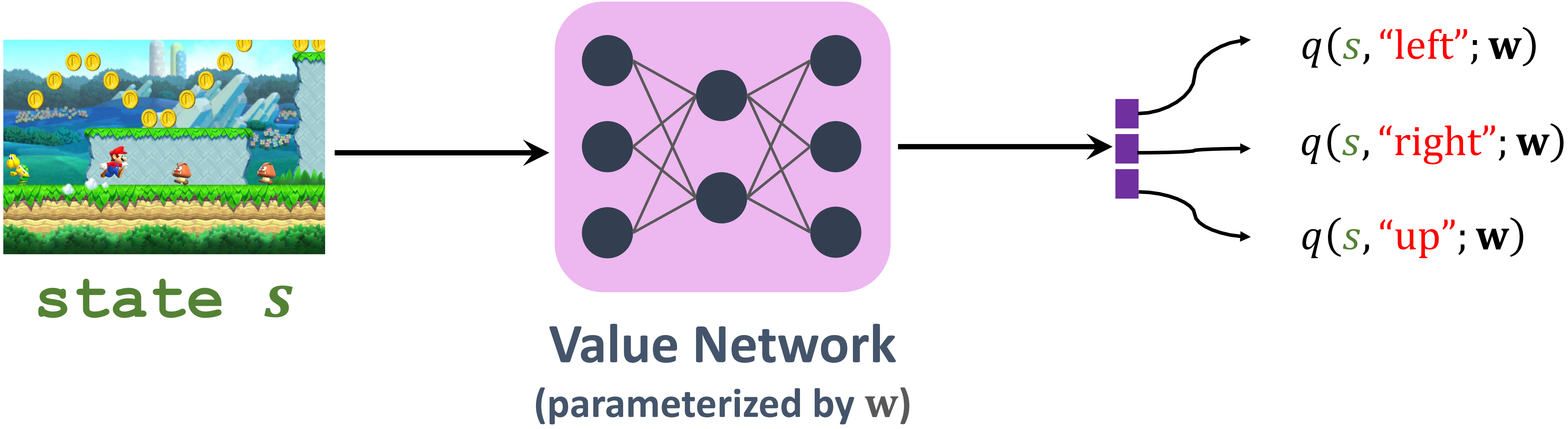
	Action a_1	Action a_2	Action a_3	Action a_4	$\cdots$
State s_1					
State s_2					
State s_3					
$\vdots$					

Sarsa (tabular version)

- Observe (s_t, a_t, r_t, s_{t+1}).
- Sample $a_{t+1} \sim \pi(\cdot \mid s_{t+1})$, where π is a policy function.
- TD target: $y_t = r_t + \gamma \cdot Q_\pi(s_{t+1}, a_{t+1})$.
- TD error: $\delta_t = Q_\pi(s_t, a_t) - y_t$.
- Update: $Q_\pi(s_t, a_t) \leftarrow \underbrace{Q_\pi(s_t, a_t) - \alpha \cdot \delta_t}_{\text{make } Q_\pi(s_t, a_t) \text{ closer to } y_t}$.

Value Network Version

- Approximate $Q_\pi(s, a)$ by a value network, $q(s, a|\mathbf{w})$.



Value Network Version

- Approximate $Q_\pi(s, a)$ by a value network, $q(s, a|\mathbf{w})$.
- Note that the $Q_\pi(s, a)$ and $q(s, a|\mathbf{w})$ depends on π.
- q is used as the critic who evaluates the actor. (Actor-Critic Method.)
- We seek to learn the parameter, $\mathbf{w}$.

Sarsa (Value Network Version)

- Observe (s_t, a_t, r_t, s_{t+1}).
- Sample $a_{t+1} \sim \pi(\cdot \mid s_{t+1})$, where π is a policy function.
- TD target: $y_t = r_t + \gamma \cdot q(s_{t+1}, a_{t+1} | \mathbf{w})$.
- TD error: $\delta_t = q(s_t, a_t | \mathbf{w}) - y_t$.
- SGD: $\mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \delta_t \cdot \frac{\partial\, q(s_t, a_t | \mathbf{w})}{\partial\, \mathbf{w}}$.

Sarsa: Recap

- **Goal:** Learn the action-value function Q_π.
- **Tabular version** (directly learn Q_π).
  - There are finite state and actions.
  - Draw a table, and update the table using Sarsa.
- **Value network version** (function approximation).
  - Approximate Q_π by the value network $q(s, a|\mathbf{w})$.
  - Update the parameter, $\mathbf{w}$, using Sarsa.
  - Application: actor-critic method.

Q-Learning

Sarsa VS Q-Learning

- Sarsa is for training action-value function, $Q_\pi(s, a)$.
- TD target: $y_t = r_t + \gamma \cdot Q_\pi(s_{t+1}, a_{t+1})$.
- We used Sarsa for value network (critic).

Sarsa VS Q-Learning

- Sarsa is for training action-value function, $Q_\pi(s,a)$.
- TD target: $y_t = r_t + \gamma \cdot Q_\pi(s_{t+1}, a_{t+1})$.
- We used Sarsa for value network (critic).

- Q-learning is for training the optimal action-value function, $Q^\star(s,a)$.
- TD target: $y_t = r_t + \gamma \cdot \max_a Q^\star(s_{t+1}, a)$.
- We used Q-learning for DQN.

Derive TD Target

- We have proved that for all π,

$$Q_\pi(s_t, a_t) = \mathbb{E}[R_t + \gamma \cdot Q_\pi(S_{t+1}, A_{t+1}) | s_t, a_t].$$

- If π is the optimal policy $\pi^\star$, then

$$Q_{\pi^\star}(s_t, a_t) = \mathbb{E}[R_t + \gamma \cdot Q_{\pi^\star}(S_{t+1}, A_{t+1}) | s_t, a_t].$$

- We denote $Q_{\pi^\star}$ by $Q^\star$.

Identity: $Q^\star(s_t, a_t) = \mathbb{E}[R_t + \gamma \cdot Q^\star(S_{t+1}, A_{t+1}) | s_t, a_t].$

Derive TD Target

Identity: $Q^{\star}(s_t, a_t) = \mathbb{E}[R_t + \gamma \cdot Q^{\star}(S_{t+1}, A_{t+1}) | s_t, a_t]$.

- The action A_{t+1} is computed by

$$A_{t+1} = \underset{a}{\operatorname{argmax}}\, Q^{\star}(S_{t+1}, a).$$

- Thus $Q^{\star}(S_{t+1}, A_{t+1}) = \max_{a} Q^{\star}(S_{t+1}, a)$.

Derive TD Target

Identity: $Q^\star(s_t, a_t) = \mathbb{E}[R_t + \gamma \cdot Q^\star(S_{t+1}, A_{t+1}) | s_t, a_t]$.

- The action A_{t+1} is computed by

$$A_{t+1} = \underset{a}{\operatorname{argmax}}\, Q^\star(S_{t+1}, a).$$

- Thus $Q^\star(S_{t+1}, A_{t+1}) = \max_{a} Q^\star(S_{t+1}, a)$.

Identity: $Q^\star(s_t, a_t) = \mathbb{E}\left[R_t + \gamma \cdot \max_{a} Q^\star(S_{t+1}, a) \,\middle|\, s_t, a_t\right]$.

Derive TD Target

Identity: $Q^{\star}(s_t, a_t) = \mathbb{E}\left[R_t + \gamma \cdot \max_{a} Q^{\star}(S_{t+1}, a) \,\middle|\, s_t, a_t\right].$

Derive TD Target

Identity: $$Q^{\star}(s_t, a_t) = \underbrace{\mathbb{E}\left[R_t + \gamma \cdot \max_a Q^{\star}(S_{t+1}, a) \,\middle|\, s_t, a_t\right]}_{\approx\, y_t}.$$

- Let (s_{t+1}, r_t) be an observation of (S_{t+1}, R_t).
- TD target: $y_t = r_t + \gamma \cdot \max_a Q^{\star}(s_{t+1}, a)$.

Q-Learning (tabular version)

- Observe (s_t, a_t, r_t, s_{t+1}).
- TD target: $y_t = r_t + \gamma \cdot \max_{a} Q^{\star}(s_{t+1}, a)$.

Q-Learning (tabular version)

- Observe (s_t, a_t, r_t, s_{t+1}).
- TD target: $y_t = r_t + \gamma \cdot \boxed{\max_{a} Q^{\star}(s_{t+1}, a).}$

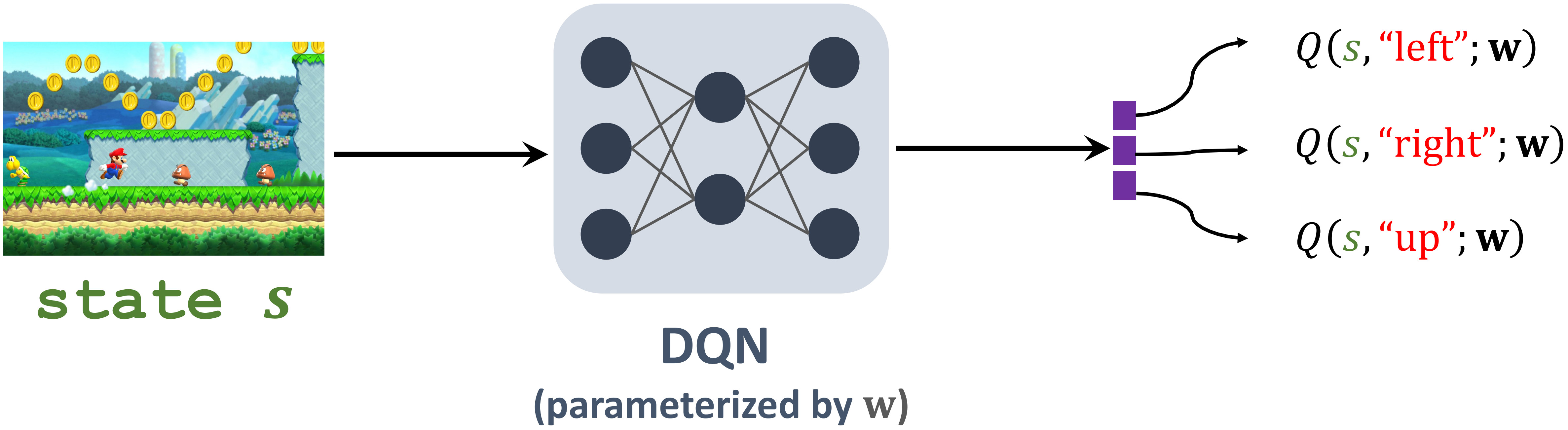
	Action a_1	Action a_2	Action a_3	Action a_4	$\cdots$
State s_1					
State s_2					
State s_3					
$\vdots$					

Q-Learning (tabular version)

- Observe (s_t, a_t, r_t, s_{t+1}).
- TD target: $y_t = r_t + \gamma \cdot \max_{a} Q^{\star}(s_{t+1}, a)$.
- TD error: $\delta_t = Q^{\star}(s_t, a_t) - y_t$.
- Update: $Q^{\star}(s_t, a_t) \leftarrow \underbrace{Q^{\star}(s_t, a_t) - \alpha \cdot \delta_t}_{\text{make } Q^{\star}(s_t, a_t) \text{ closer to } y_t}$.

DQN Version

- Approximate $Q^{\star}(s, a)$ by a value network, $Q(s, a|\mathbf{w})$.
- The network is called deep Q network (DQN).



DQN Version

- Approximate $Q^{\star}(s, a)$ by a value network, $Q(s, a|\mathbf{w})$.
- The network is called deep Q network (DQN).
- It controls the agent by: $a_t = \max_{a} Q(s_t, a|\mathbf{w})$
- We seek to learn the parameter, $\mathbf{w}$.

Q-Learning (DQN Version)

- Observe (s_t, a_t, r_t, s_{t+1}).
- TD target: $y_t = r_t + \gamma \cdot \max_{a} Q(s_{t+1}, a \mid \mathbf{w})$.
- TD error: $\delta_t = Q(s_t, a_t \mid \mathbf{w}) - y_t$.
- Update: $\mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \delta_t \cdot \frac{\partial\, q(s_t, a_t | \mathbf{w})}{\partial\, \mathbf{w}}$.

Q-Learning: Recap

- **Goal:** Learn the optimal action-value function $Q^{\star}$.
- **Tabular version** (directly learn$Q^{\star}$).
  - There are finite state and actions.
  - Draw a table, and update the table using Q-learning.
- **DQN version** (function approximation).
  - Approximate $Q^{\star}$ by the value network $Q(s, a|\mathbf{w})$.
  - Update the parameter, $\mathbf{w}$, using Q-learning.

Multi-Step Target

Multi-Step Return

Identity: $U_t = R_t + \gamma \cdot U_{t+1}.$

Replace U_{t+1} by $R_{t+1} + \gamma \cdot U_{t+2}$

- It follows that $U_{t+1} = R_{t+1} + \gamma \cdot U_{t+2}.$

Multi-Step Return

Identity: $U_t = R_t + \gamma \cdot \boxed{U_{t+1}.}$

Replace U_{t+1} by $R_{t+1} + \gamma \cdot U_{t+2}$

Identity: $U_t = R_t + \gamma \cdot (R_{t+1} + \gamma \cdot U_{t+2}).$

Multi-Step Return

Identity: $U_t = R_t + \gamma \cdot U_{t+1}.$

Replace U_{t+1} by $R_{t+1} + \gamma \cdot U_{t+2}$

Identity: $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot U_{t+2}.$

Multi-Step Return

Identity: $U_t = R_t + \gamma \cdot U_{t+1}$.

Identity: $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot U_{t+2}$.

Identity: $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot R_{t+2} + \gamma^3 \cdot U_{t+3}$.

Multi-Step Return

Identity: $U_t = R_t + \gamma \cdot U_{t+1}.$

Identity: $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot U_{t+2}.$

Identity: $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot R_{t+2} + \gamma^3 \cdot U_{t+3}.$

Identity: $U_t = \sum_{i=0}^{m-1} \gamma^i \cdot R_{t+i} + \gamma^m \cdot U_{t+m}.$

Multi-Step TD Targets

Identity: $U_t = \sum_{i=0}^{m-1} \gamma^i \cdot R_{t+i} + \gamma^m \cdot U_{t+m}$.

- m-step TD target for learning Q_π (i.e., Sarsa):

$$y_t = \sum_{i=0}^{m-1} \gamma^i \cdot r_{t+i} + \gamma^m \cdot Q_\pi(s_{t+m}, a_{t+m}).$$

Multi-Step TD Targets

Identity: $U_t = \sum_{i=0}^{m-1} \gamma^i \cdot R_{t+i} + \gamma^m \cdot U_{t+m}$.

- m-step TD target for learning Q_π (i.e., Sarsa):

$$y_t = \sum_{i=0}^{m-1} \gamma^i \cdot r_{t+i} + \gamma^m \cdot Q_\pi(s_{t+m}, a_{t+m}).$$

- m-step TD target for learning $Q^\star$ (i.e., Q-learning):

$$y_t = \sum_{i=0}^{m-1} \gamma^i \cdot r_{t+i} + \gamma^m \cdot \max_a Q^\star(s_{t+m}, a).$$

Comparison

- One-step TD target for learning $Q^\star$ (or DQN):

$$y_t = r_t + \gamma \cdot \max_a Q^\star(s_{t+1}, a).$$

- m-step TD target for learning $Q^\star$ (or DQN):

$$y_t = \sum_{i=0}^{m-1} \gamma^i \cdot r_{t+i} + \gamma^m \cdot \max_a Q^\star(s_{t+m}, a).$$

- If m is suitably tuned, m-step target works better than one-step target [1].

Reference:

1. Hossel et al. Rainbow: combining improvements in deep reinforcement learning. In *AAAI*, 2018.

Summary

Summary

1. Mathematically derived the TD target.
2. TD learning algorithms:
    - Sarsa algorithm for learning Q_π and value network (critic).
    - Q-learning algorithm for learning $Q^\star$ and DQN.
3. Multi-step TD target.

Improvements for TD Learning

1. Experience replay (ER) and its variants (e.g., prioritized ER.)
   - Reuse experience.
   - Eliminate correlation.
2. Target network and double DQN.
   - Address the overestimation issue of Q-learning.
3. Multi-step TD target.

Thank you!