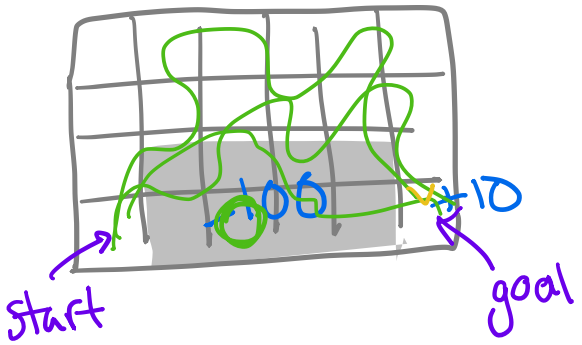
Goal find $\pi^*(a|s)$

How? learn $Q^*(s,a) \rightarrow \pi(a|s) = \underset{a}{\text{argmax}}\ Q(s,a)$

1. Q-Learning
2. Deep Q-Networks



deterministic $p(s'|s,a)$

ϵ-greedy policy

$$a = \begin{cases} \underset{a}{\text{argmax}}\ Q(s,a) & 1-\epsilon \\ \text{random} & \epsilon \end{cases}$$

Q-Learning Algorithm

initialize Q-values $Q(s,a) = 0 \quad \forall (s,a)$

for num episodes

initialize state s

while not done

select action $a = \begin{cases} \underset{a}{\text{argmax}}\ Q(s,a) & 1-\epsilon \\ \text{random} & \epsilon \end{cases}$

take action a and observe reward r and next state s'

update Q-value

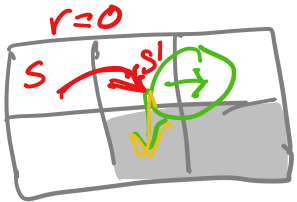
SARSA $Q(s',a')$

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$$

target — curr. est

$$Q^*(s,a) = r + \gamma \max_{a'} Q^*(s,a)$$

error (TD-error, RPE) → drives learning



(s,a,r,s',a')

$\underset{a'}{\text{argmax}}\ Q(s',a')$

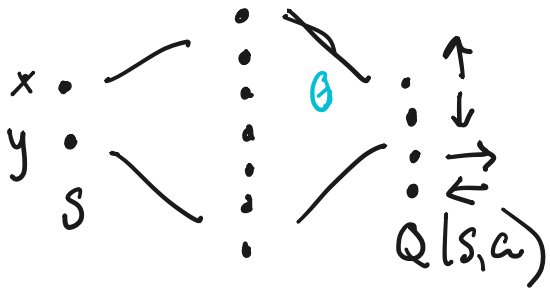
$\arg\max_{a'} Q(s,a')$

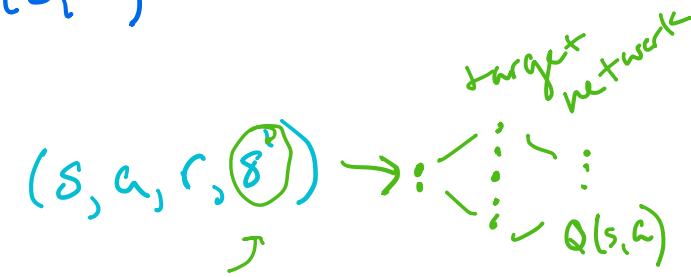
Problem w/ Q-Learning : TABULAR

Deep RL : use neural networks for function approximation

Deep Q-Networks :

- use NN to approximate $Q(s,a)$





$$\mathcal{L}(\theta) = \left\langle \left(\underbrace{r + \gamma \max_{a'} Q(s', a'; \theta^-)}_{\text{target}} - \underbrace{Q(s,a;\theta)}_{\text{curr. est.}} \right)^2 \right\rangle_{(s,a,r,s') \sim \text{Unif. (Mem)}}$$

δ

1. correlated inputs → experience replay
2. high variance gradient → target network

periodically set $\theta^- = \theta$

$$\theta \leftarrow \theta + \alpha \delta \nabla_\theta Q(s,a;\theta)$$

DQN Algorithm

initialize Q-network with random weights θ
initialize target network with weights $\theta^- = \theta$
for num episodes
  initializing state s
  while not done:
    select action $a = \begin{cases} \arg\max_a Q(s,a;\theta) & 1-\epsilon \\ \text{random} & \epsilon \end{cases}$
    take action a and observe reward r and next state s'

store experience (s, a, r, s') in replay memory

sample random batch of (s, a, r, s') from memory

calculate target

$$y = \begin{cases} r + \gamma \max_{a'} Q(s', a'; \theta^-) & \text{if } s \text{ not terminal} \\ r & \text{if } s \text{ terminal} \end{cases}$$

perform gradient descent step on $(y - Q(s, a; \theta))^2$ wrt. θ

every C steps update target network $\theta^- = \theta$

Main (Deep) RL Algorithm Classes

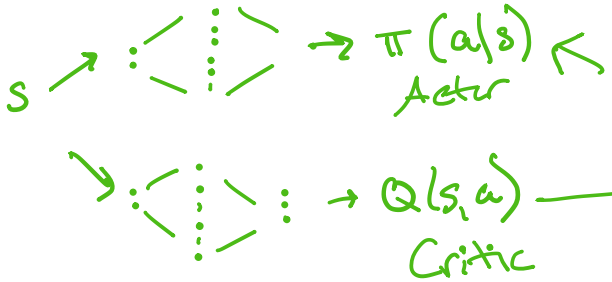
1. Value-based (e.g. DQN)

2. Policy-based (e.g. Policy Gradient) $J(\theta) = \left\langle \sum_{t=0}^{T} \gamma^t r_t \right\rangle_{\pi_\theta}$

$$\nabla_\theta J(\theta) = \left\langle \sum_{t=0}^{T} \nabla_\theta \pi(a_t|s_t) G_t \right\rangle_{\pi_\theta}$$

$$G_t = \sum_{t'=t}^{T} \gamma^{t'} r_{t'}$$

3. Actor-Critic



4. Combinations!