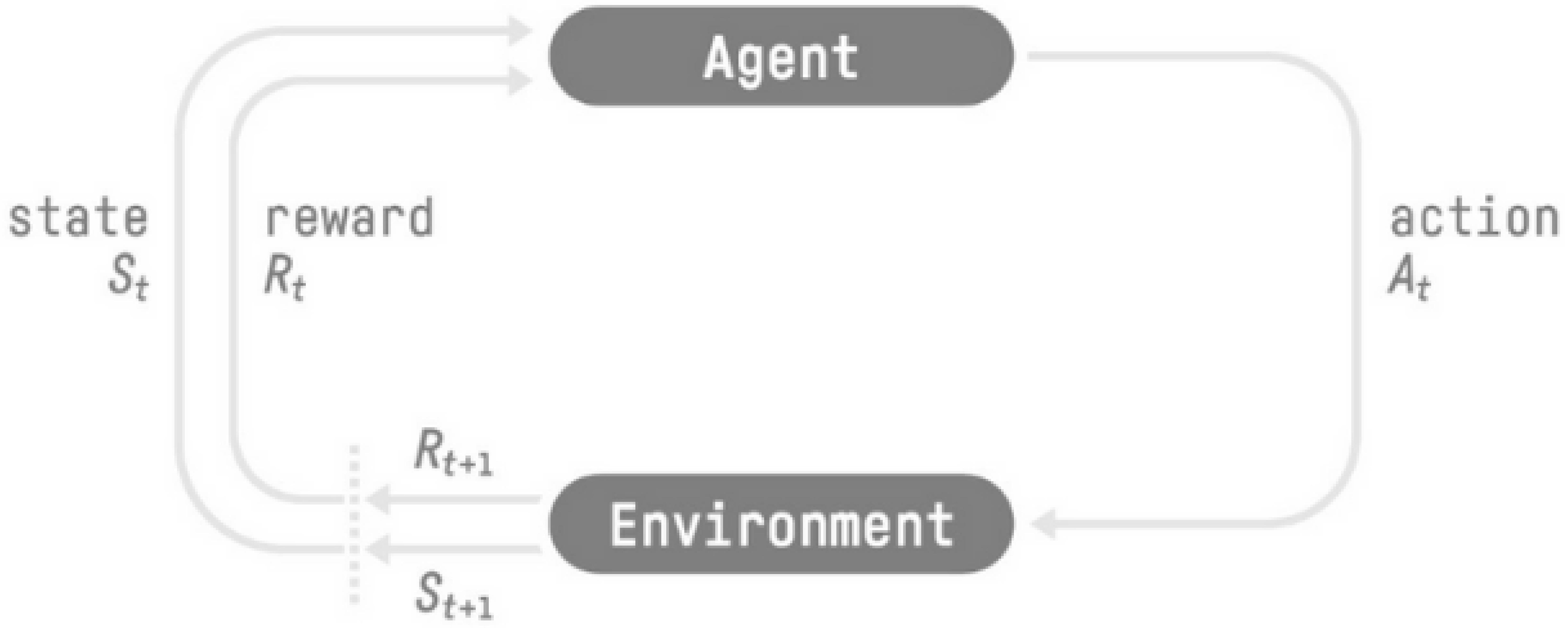
AI CAMP

school of ai Algiers

SOAI AI Camp

Solving RL: Value-Based Methods

Let’s do a recap about RL



AI CAMP

school of ai Algiers

Example

AI CAMP

school of ai Algiers

Goal

Maximizing the cumulative reward

$$R(\boxed{\tau}) = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \dots$$

Return: cumulative reward

Gamma: discount rate

Trajectory (read Tau)
Sequence of states and actions

AI CAMP

school of ai Algiers

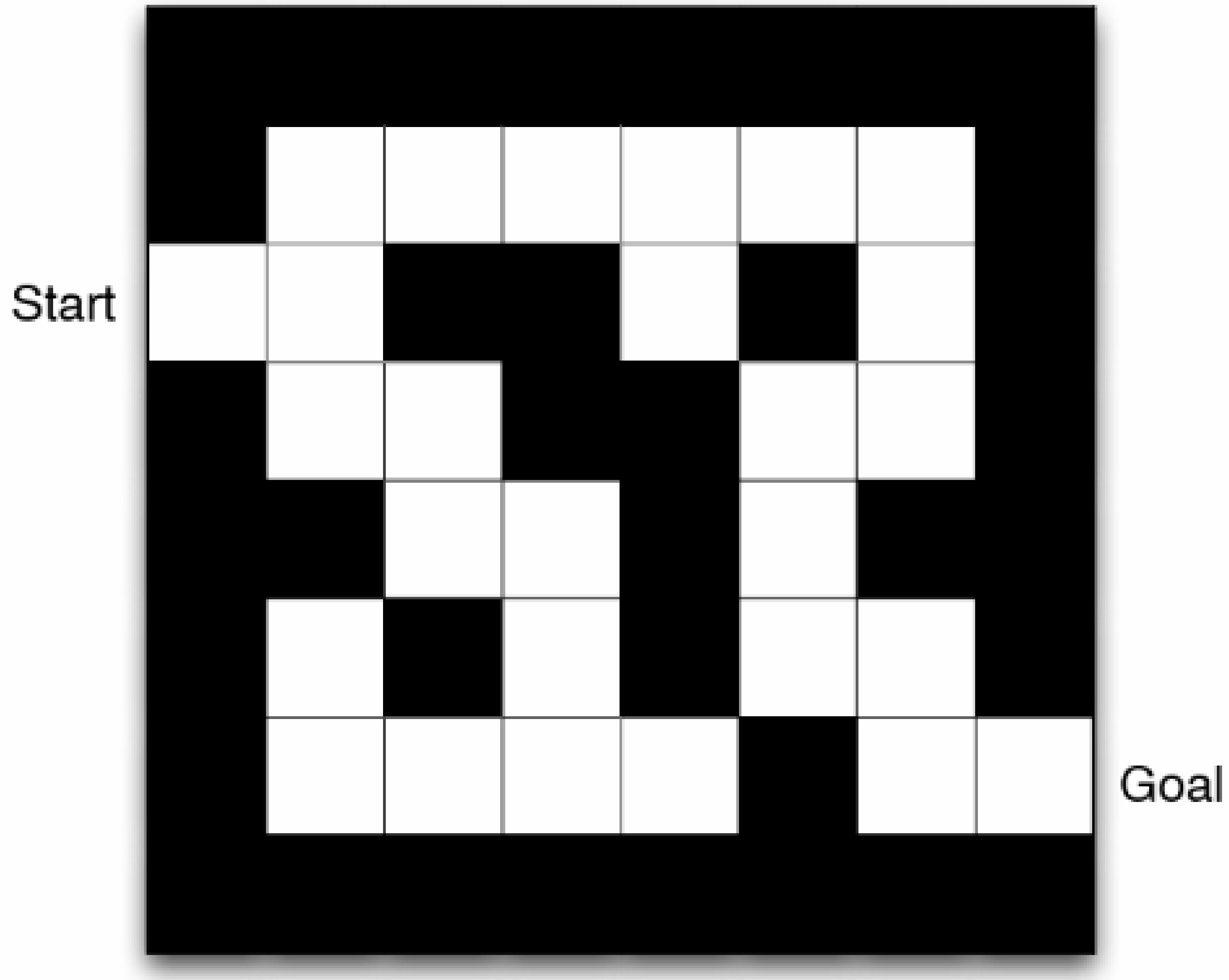
AI CAMP

school of ai Algiers

Solving Reinforcement Learning

Problem

Maze World



AI CAMP

school of ai Algiers

Problem

Maze World

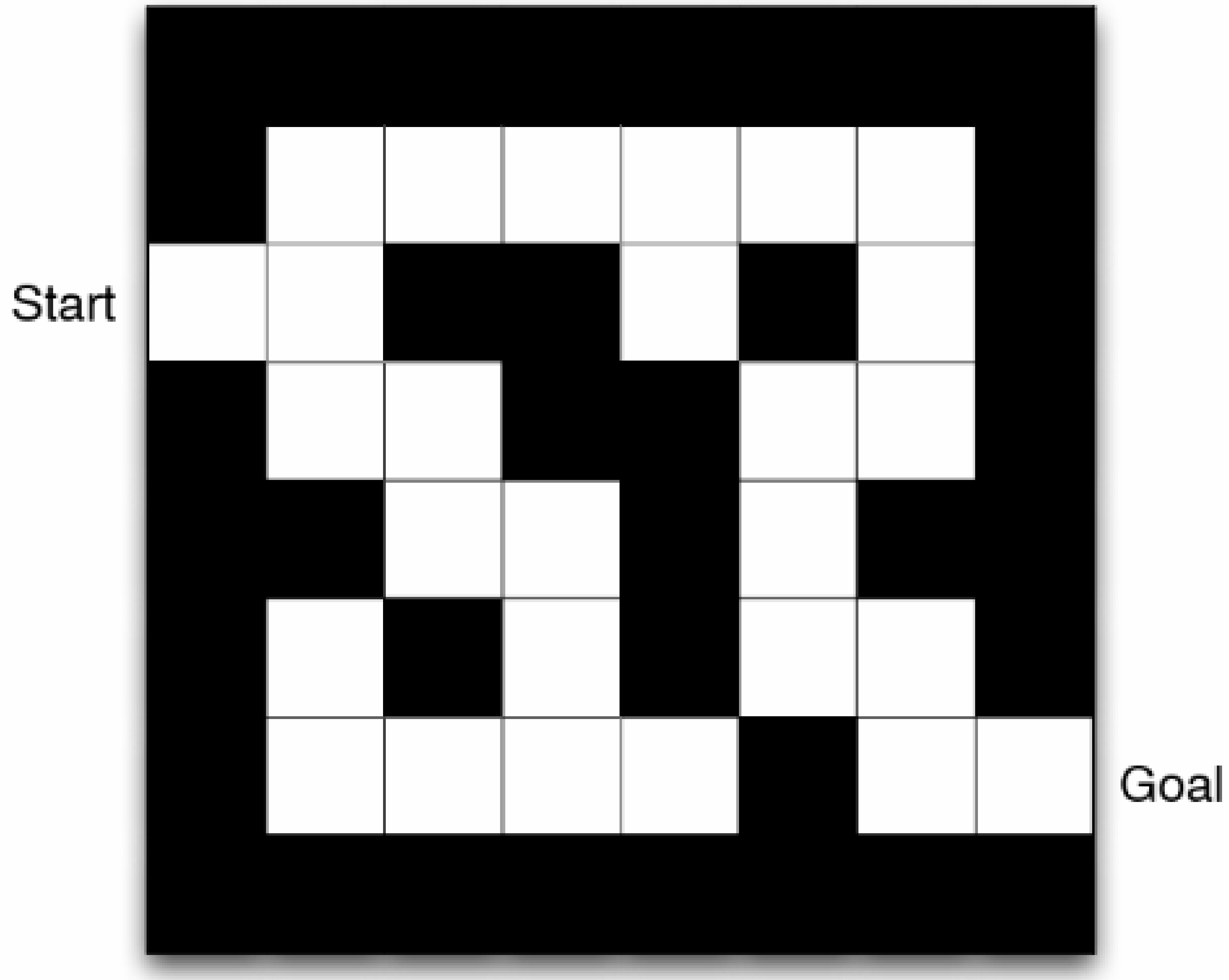
Actions

Up

Down

Left

Right



Rewards

-1 per step

-2 hitting wall

States

S: initial state

G: terminal state

Value functions

State Value Function

$$\underbrace{v_\pi(s)}_{\text{Value function}} = \underbrace{\mathbb{E}_\pi\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots\right.}_{\text{Expected discounted return}} \mid \underbrace{\left.S_t = s\right]}_{\text{Starting at state s}}$$

AI CAMP

school of ai Algiers

Value functions

State-Action Value Function

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\Big[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \Big| s_t = s, a_t = a\Big]$$

AI CAMP

school of ai Algiers

Value functions

Relation

$$v_*(s) = \max_{a(s)} q_{\pi*}(s, a)$$

How to adjust our estimations

Policy

Greedy Policy

$$\pi^*(s) = \arg\max_a Q^*(s, a)$$

AI CAMP

school of ai Algiers

Estimation Adjustment

$$\underbrace{Q(S_t, A_t)}_{\text{New Q-value estimation}} \leftarrow \underbrace{Q(S_t, A_t)}_{\text{Former Q-value estimation}} + \underbrace{\alpha}_{\text{Learning Rate}}[\underbrace{\underbrace{\underbrace{R_{t+1}}_{\text{Immediate Reward}} + \underbrace{\gamma max_a Q(S_{t+1}, a)}_{\text{Discounted Estimate optimal Q-value of next state}}}_{\text{TD Target}} - \underbrace{Q(S_t, A_t)}_{\text{Former Q-value estimation}}}_{\text{TD Error}}]$$

Math & Theory is cool. But
Let's build up intuition!

AI CAMP

school of ai Algiers

Q-Learning Algorithm

Algorithm 14: Sarsamax (Q-Learning)

Input: policy π, positive integer *num_episodes*, small positive fraction α, GLIE $\{\epsilon_i\}$
Output: value function Q ($\approx q_\pi$ if *num_episodes* is large enough)
Initialize Q arbitrarily (e.g., $Q(s,a) = 0$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$, and $Q(terminal\text{-}state, \cdot) = 0$) Step 1
for $i \leftarrow 1$ *to num_episodes* **do**
  $\epsilon \leftarrow \epsilon_i$
  Observe S_0
  $t \leftarrow 0$
  repeat
    Choose action A_t using policy derived from Q (e.g., ϵ-greedy) Step 2
    Take action A_t and observe R_{t+1}, S_{t+1} Step 3
    $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$ Step 4
    $t \leftarrow t + 1$
  until S_t *is terminal*;
end
return Q

AI CAMP

school of ai Algiers

AI CAMP

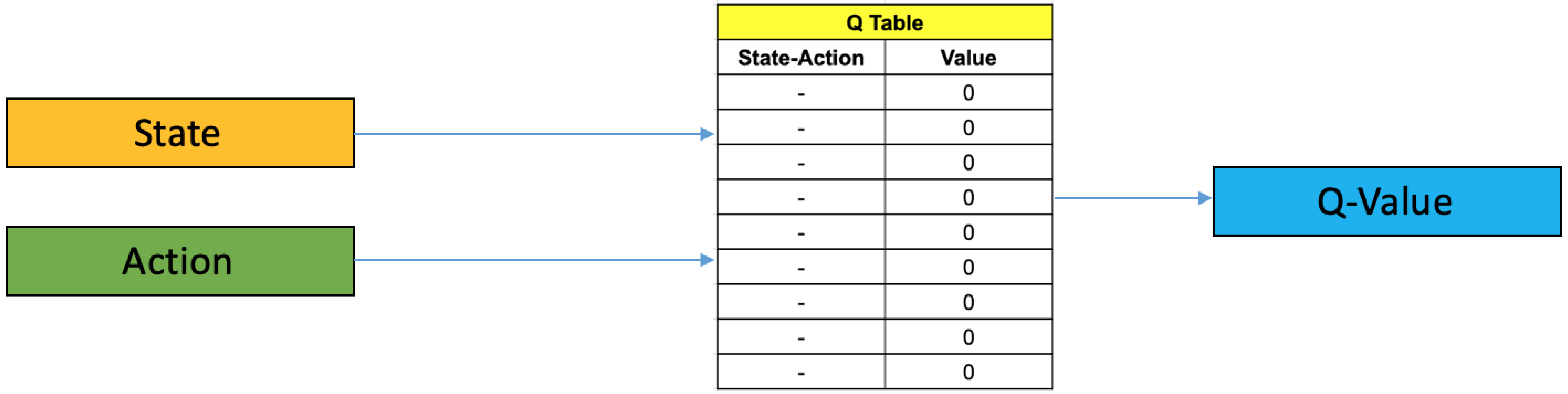
school of ai Algiers

Q-Learning is so cool!
BUT

Q-Learning Limitations

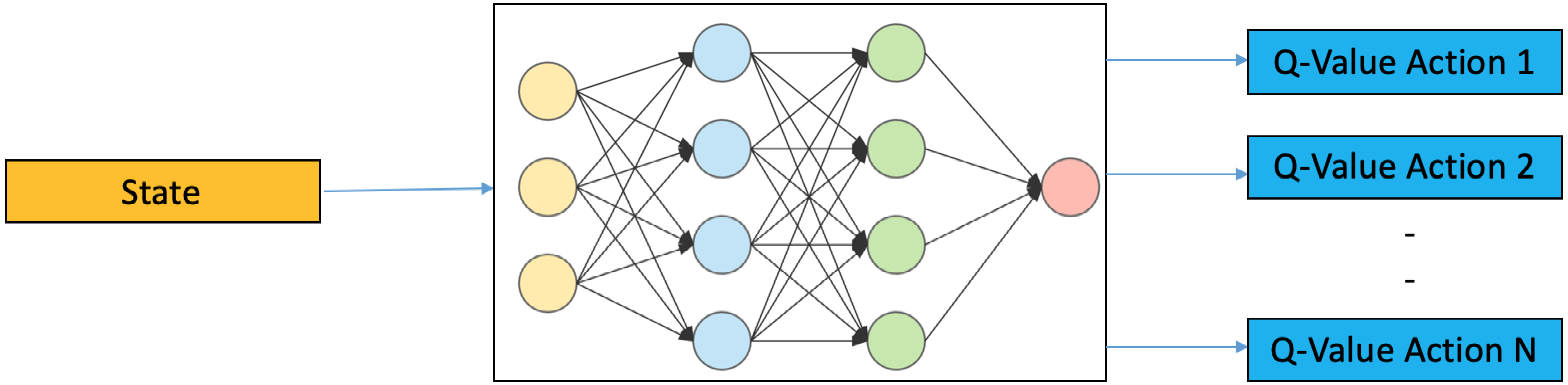
- Handling high dimensional state/action spaces
- Function approximation generalization capabilities
- Familiarity and usability only in discrete action spaces

QL vs DQL



Q Table	
State-Action	Value
-	0
-	0
-	0
-	0
-	0
-	0
-	0
-	0
-	0

Q Learning

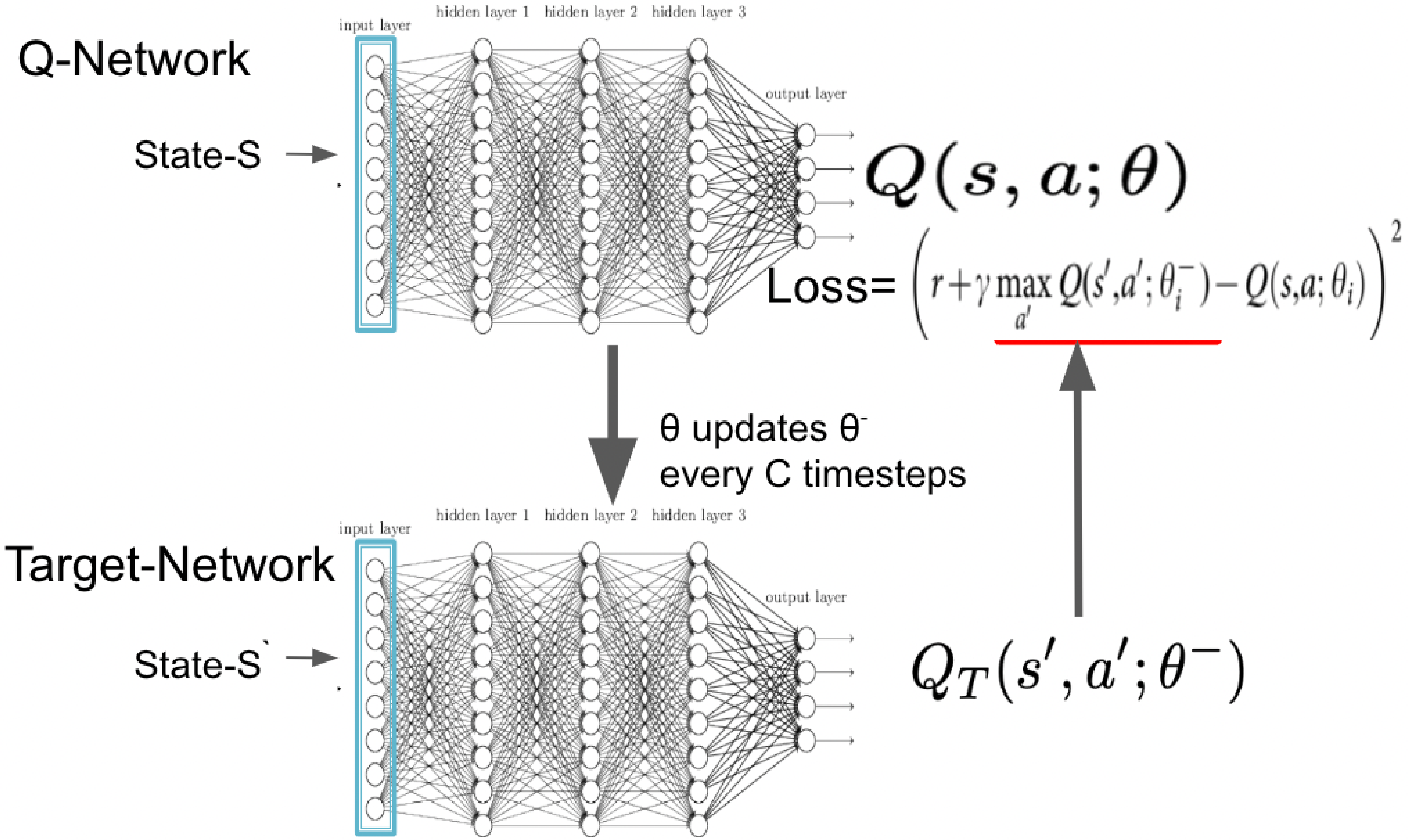


Deep Q Learning

AI CAMP

school of ai Algiers

DQN Networks



AI CAMP

school of ai Algiers

Deep Q Network Algorithm

Algorithm 1 Deep Q-learning with Experience Replay

Initialize replay memory $\mathcal{D}$ to capacity N
Initialize action-value function Q with random weights
for episode $= 1, M$ **do**
    Initialise sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(s_1)$
    for $t = 1, T$ **do**
        With probability ϵ select a random action a_t
        otherwise select $a_t = \max_a Q^*(\phi(s_t), a; \theta)$
        Execute action a_t in emulator and observe reward r_t and image x_{t+1}
        Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
        Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $\mathcal{D}$
        Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $\mathcal{D}$
        Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma \max_{a'} Q(\phi_{j+1}, a'; \theta) & \text{for non-terminal } \phi_{j+1} \end{cases}$
        Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ according to equation $\boxed{3}$
    end for
end for

AI CAMP

school of ai Algiers