Value-based RL：
用神经网络去拟合optimal action-value function Q*

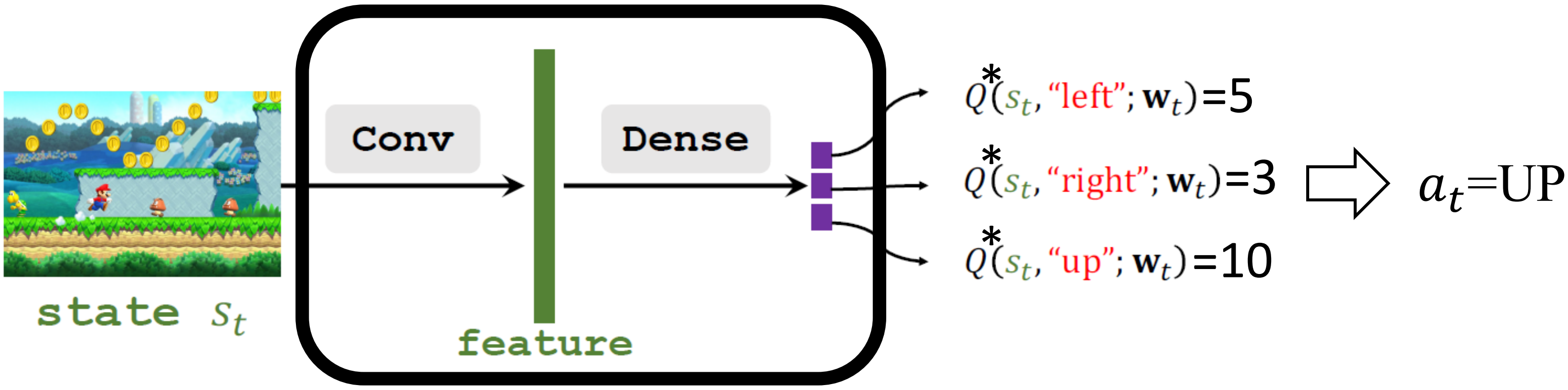
Goal: Win the game (≈ maximize the total reward.)

Question: If we know $Q^\star(s, a)$, what is the best action?

- Obviously, the best action is $a^\star = \underset{a}{\operatorname{argmax}}\, Q^\star(s, a)$.

Challenge: We do not know $Q^\star(s, a)$.

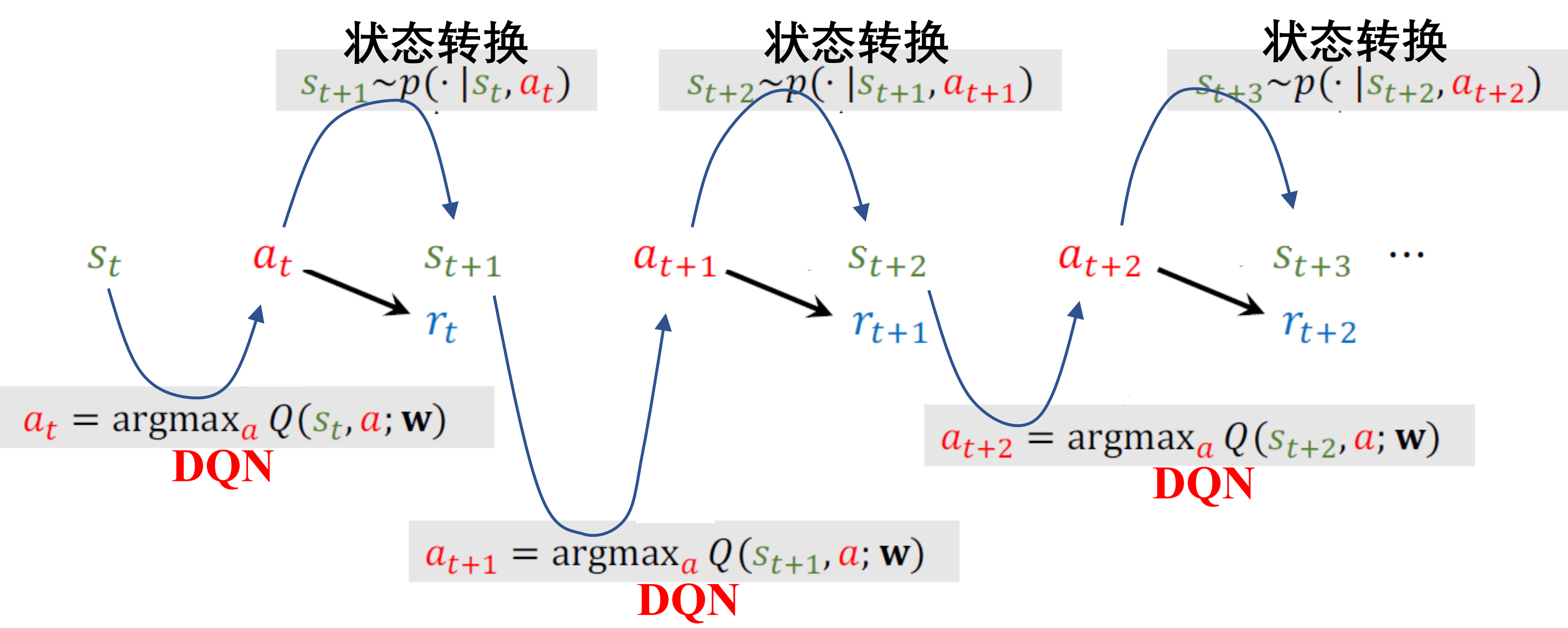
- Solution: Deep Q Network (DQN)
- Use neural network $Q(s, a; \mathbf{w})$ to $\boxed{\text{approximate } Q^\star(s, a)}$



DQN params: w_t

- Input shape: size of the screenshot.
- Output shape: dimension of action space.

trained DQN: 假如我们有一个trained DQN (Q*)，如何将其应用于agent?



问: 我们要利用什么数据，如何训练一个DQN来去拟合Q*?

Temporal Difference (TD) Learning

prediction: $q_t = Q^*(s_t^2, a_t^2; w_t)$

TD target: $r_t = 3$ | $\gamma \cdot Q^*(s_{t+1}^1, a_{t+1}^1; w_t)$

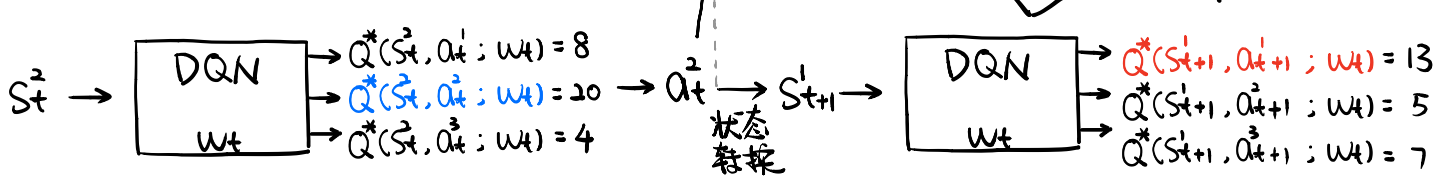
$y_t = r_t + \gamma \cdot Q^*(s_{t+1}^1, a_{t+1}^1; w_t)$ [1]

$$Loss = \frac{1}{2}\left(Q^*(s_t^2, a_t^2; w_t) - y_t\right)^2$$

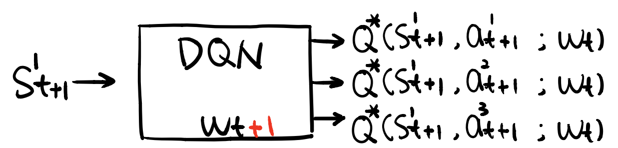
Gradient descent: $w_{t+1} = w_t - \alpha \cdot \frac{\partial Loss}{\partial w}\Big|_{w=w_t}$

$$= w_t - \alpha \cdot (q_t - y_t)\frac{\partial Q^*(s_t^2, a_t^2; w_t)}{\partial w}\Big|_{w=w_t}$$

update params



THEN:



→ 重复上述过程，可以看出，在每个时间点上，就会利用 TD learning 更新一次权重 w.

[1]:

$$Q^*(s_t^2, a_t^2; w_t) = \max_{\pi} Q_\pi(s_t^2, a_t^2; w_t)$$
$$= \max_{\pi} E[U_t | S_t = s_t^2, A_t = a_t^2; w_t]$$

$$Q^*(s_{t+1}^1, a_{t+1}^1; w_t) = \max_{\pi} Q_\pi(s_{t+1}^1, a_{t+1}^1; w_t)$$
$$= \max_{\pi} E[U_{t+1} | S_{t+1} = s_{t+1}^1, A_{t+1} = a_{t+1}^1; w_t]$$

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

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

$$\underbrace{r_t + \gamma \cdot Q^*(s_{t+1}^1, a_{t+1}^1; w_t) = y_t}_{\text{Target}} \approx \underbrace{Q^*(s_t^2, a_t^2; w_t)}_{\text{Prediction}}$$

Temporal Difference (TD) Learning

Algorithm: One iteration of TD learning.

1. Observe state $S_t = s_t$ and perform action $A_t = a_t$.
2. Predict the value: $q_t = Q(s_t, a_t; \mathbf{w}_t)$.
3. Differentiate the value network: $\mathbf{d}_t = \frac{\partial\, Q(s_t, a_t; \mathbf{w})}{\partial\, \mathbf{w}} \Big|_{\mathbf{w}=\mathbf{w}_t}$.
4. Environment provides new state s_{t+1} and reward r_t.
5. Compute TD target: $y_t = r_t + \gamma \cdot \max_a Q(s_{t+1}, a; \mathbf{w}_t)$.
6. Gradient descent: $\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \cdot (q_t - y_t) \cdot \mathbf{d}_t$.