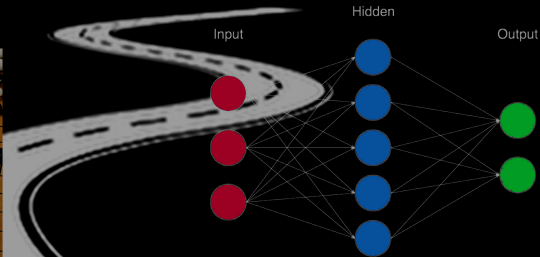
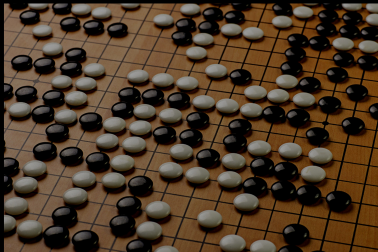


$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [r_{t+1} + \lambda \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]$$

(The New Action Value = The Old Value) + The Learning Rate × (The New Information − the Old Information)



Reinforcement learning

Michal CHOVANEC, PhD

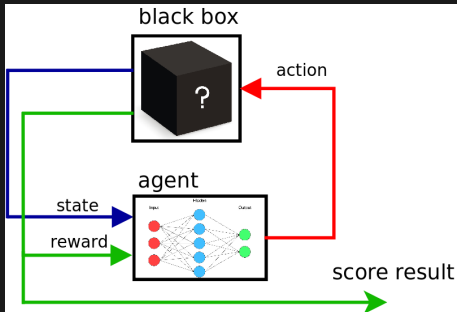
Reinforcement learning

- learn from punishment and rewards
- learn to play a game with unknown rules



Reinforcement learning

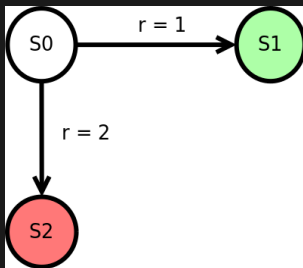
- obtain **state**
- choose **action**
- **execute** action
- obtain **reward**
- learn from **experiences**



Making decisions

two possible strategies

- strategy 1 : $S_0 \rightarrow S_1$, score = 1.0
- strategy 2 : $S_0 \rightarrow S_2$, score = 2.0



$$Q(s, a) = R(s, a)$$

where

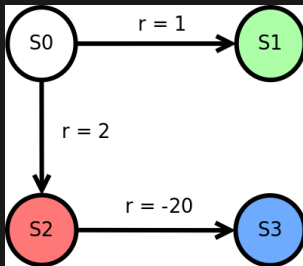
s is state

a is action

Making decisions

two possible strategies, **greedy = trap**

- strategy 1 : $S_0 \rightarrow S_1$, score = 1.0
- strategy 2 : $S_0 \rightarrow S_2 \rightarrow S_3$, score = $2.0 + (-20.0) = -18$

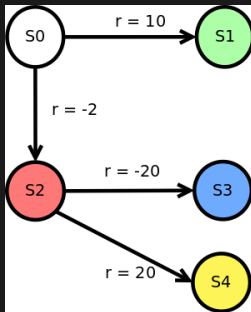


$$Q(s, a) = R(s, a) + \textit{FutureReward}$$

Making decisions

three possible strategies

- strategy 1 : $S_0 \rightarrow S_1$, score = 10.0
- strategy 2 : $S_0 \rightarrow S_2 \rightarrow S_3$, score = $-2.0 + (-20.0) = -22$
- strategy 3 : $S_0 \rightarrow S_2 \rightarrow S_4$, score = $-2.0 + (20.0) = 18$



$$Q(s, a) = R(s, a) + \max(FutureRewards)$$

Q learning

$$Q(s, a) = R(s, a) + \gamma \max_{a'} Q(s', a')$$

where

s is state

a is action

s' is next state

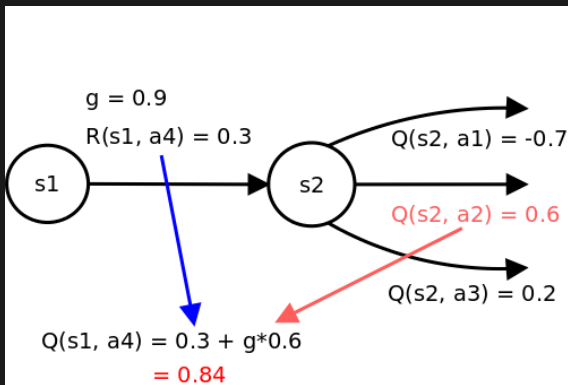
a' is best action in next state

$R(s, a)$ is reward

$\gamma \in \langle 0, 1 \rangle$ is discount factor

Q learning

$$Q(s, a) = R(s, a) + \gamma \max_{a'} Q(s', a')$$



Q learning - stochastic

$$Q(s, a) = R(s, a) + \gamma \max_{a'} Q(s', a')$$

$$\Delta Q(s, a) = R(s, a) + \gamma \max_{a'} Q(s', a') - Q(s, a)$$

$$\Delta Q(s, a) = \alpha(R(s, a) + \gamma \max_{a'} Q(s', a') - Q(s, a))$$

$$Q(s, a) = (1 - \alpha)Q(s, a) + \alpha(R(s, a) + \gamma \max_{a'} Q(s', a'))$$

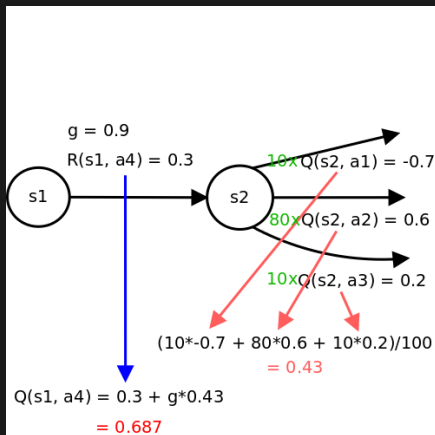
Q learning

- **obtain reward**
`reward = env.get_reward();`
- **update state**
`state_old = state;`
`state = env.get_observation();`
- **select action**
`action_old = action;`
`action = select_action(Q(state));`
- **process learning**
$$Q(\text{state_old}, \text{action_old}) += \alpha(\text{reward} + \gamma \max_{a'} Q(\text{state}, a') - Q(\text{state_old}, \text{action_old}))$$
- **execute action** `env.action(action);`

SARSA learning

$$\Delta Q(s, a) = \alpha(R(s, a) + \gamma Q(s', a') - Q(s, a))$$

$$Q(s, a) = (1 - \alpha)Q(s, a) + \alpha(R(s, a) + \gamma Q(s', a'))$$



SARSA learning

- **obtain reward**

```
reward = env.get_reward();
```

- **update state**

```
state_old = state;
```

```
state = env.get_observation();
```

- **select action**

```
action_old = action;
```

```
action = select_action(Q(state));
```

- **process learning**

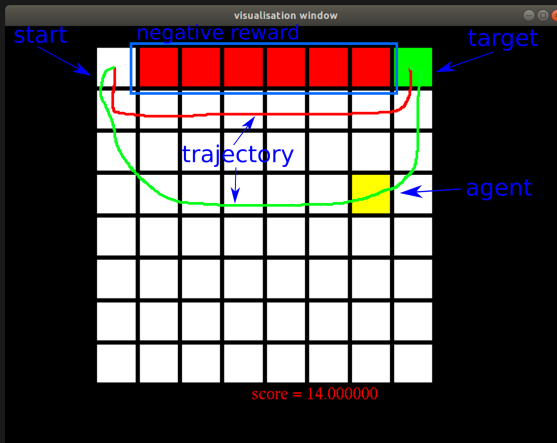
```
Q(state_old, action_old) +=  $\alpha$ (reward +  $\gamma$ Q(state, action) -  
Q(state_old, action_old))
```

- **execute action** env.action(action);

Q-learning vs SARSA - cliff example

$$Q(s, a) = (1 - \alpha)Q(s, a) + \alpha(R(s, a) + \gamma \max_{a'} Q(s', a'))$$

$$Q(s, a) = (1 - \alpha)Q(s, a) + \alpha(R(s, a) + \gamma Q(s', a'))$$



Usefull links



ImageNet Classification with Deep Convolutional Neural Networks <https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>



Alex Krizhevsky web, <https://www.cs.toronto.edu/~kriz/>



Deep Belief Nets in C++ and CUDA C: Volume III
<https://www.amazon.com/Deep-Belief-Nets-CUDA-Convolutional/dp/1530895189>



Deep Learning (Adaptive Computation and Machine Learning
<https://www.amazon.com/Deep-Learning-Adaptive-Computation-Machine/dp/0262035618>



Densely Connected Convolutional Networks <https://arxiv.org/pdf/1608.06993.pdf>



MNIST dataset <http://yann.lecun.com/exdb/mnist/>



Digital signal processing for STM32 microcontrollers using CMSIS
https://www.st.com/resource/en/application_note/dm00273990.pdf



CMSIS-NN: Efficient Neural Network Kernels for Arm Cortex-M CPUs
<https://arxiv.org/pdf/1801.06601.pdf>

Q&A



michal chovanec (michal.nand@gmail.com)
www.youtube.com/channel/UCzVvP2ou8v3afNiVrPAHQGg