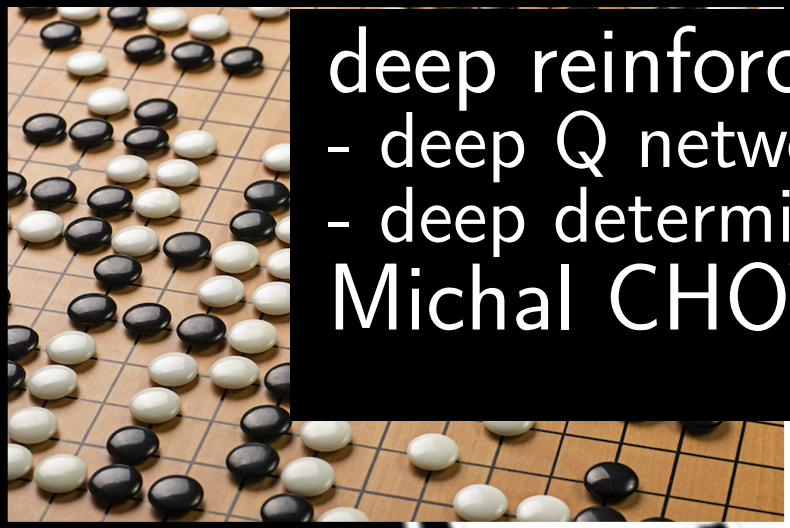
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[r_{t+1} + \lambda \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right]$$

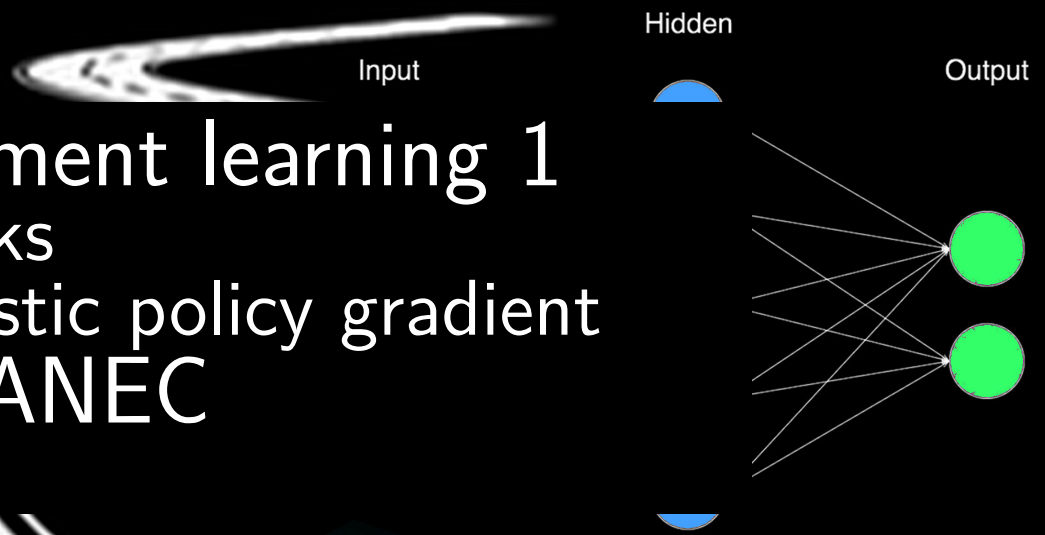
(The New Action Value = The Old Value) + The Learning Rate × (The New Information − the Old Information)

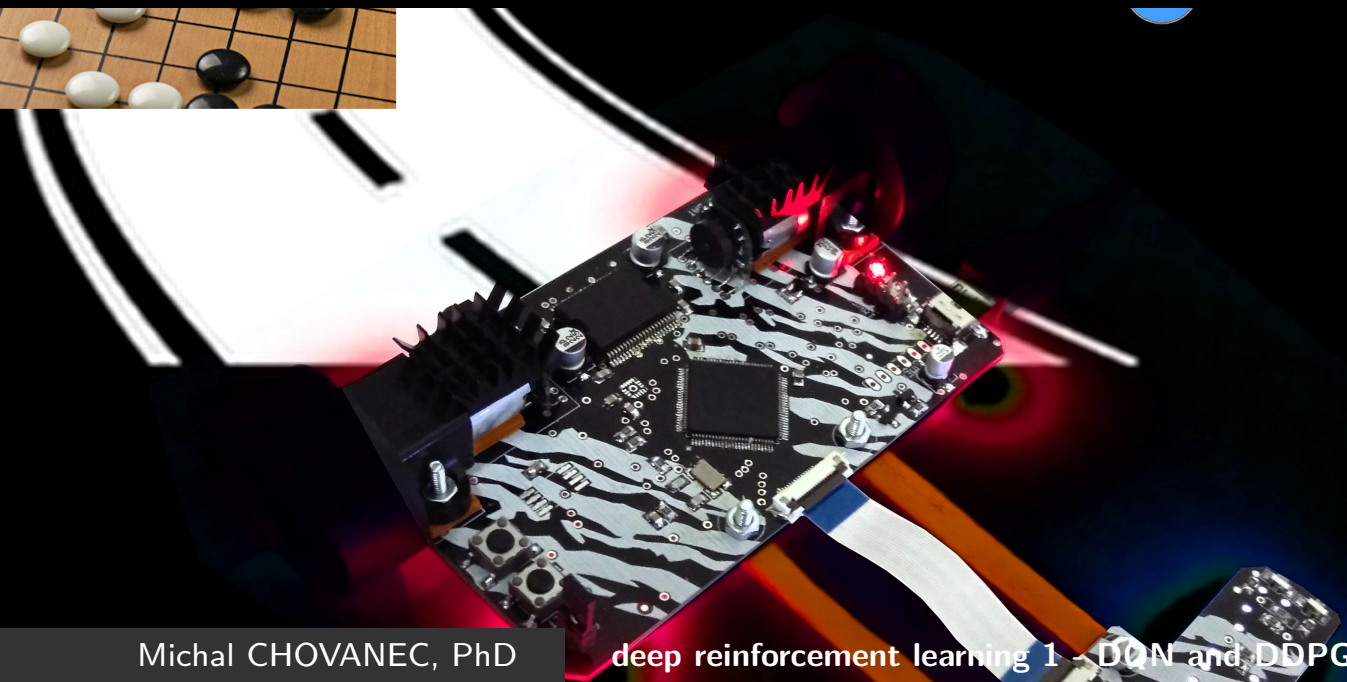


deep reinforcement learning 1

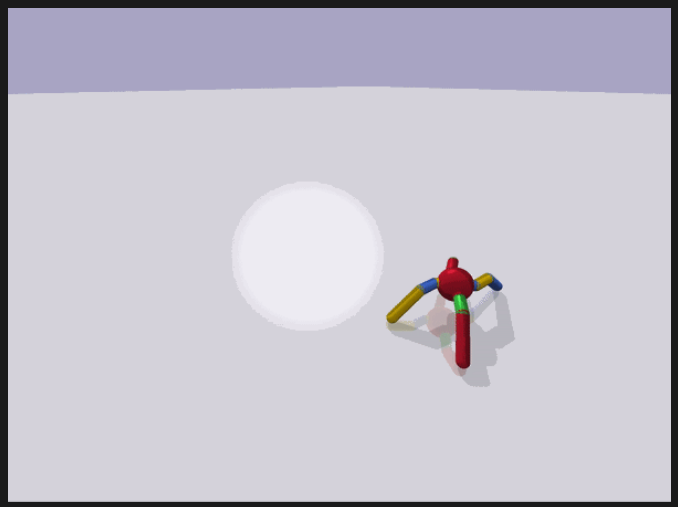
- deep Q networks
- deep deterministic policy gradient

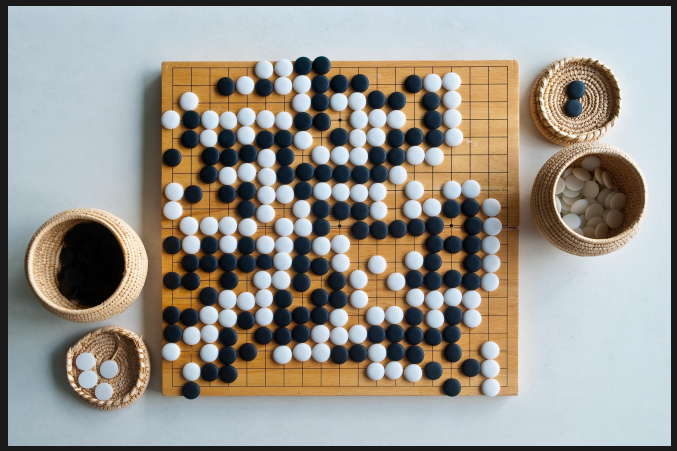
Michal CHOVANEC

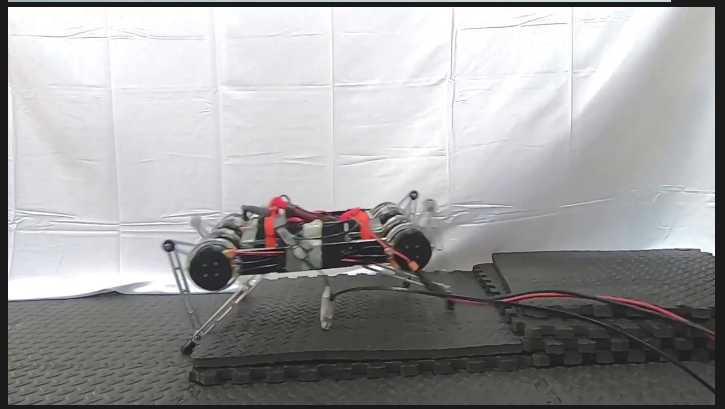




reinforcement learning

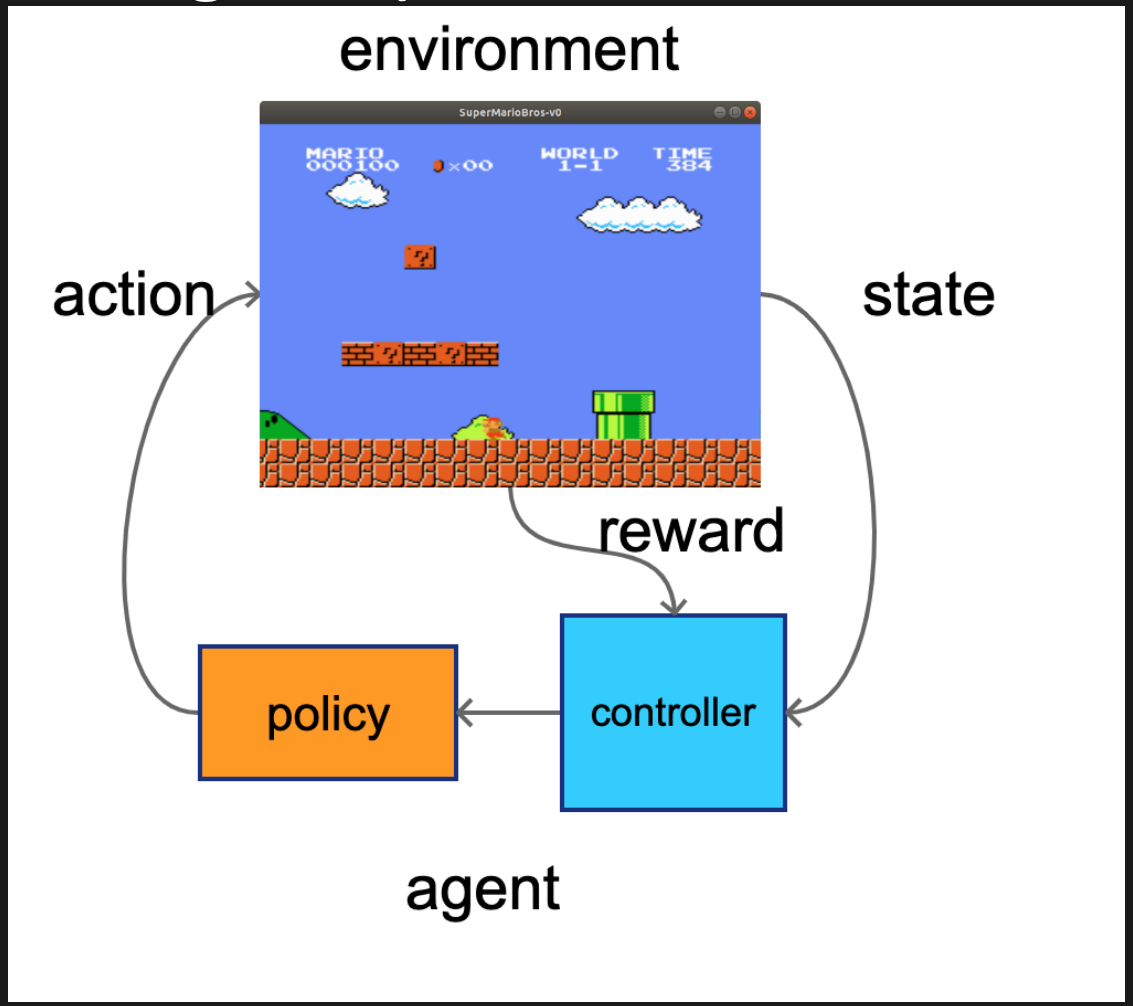




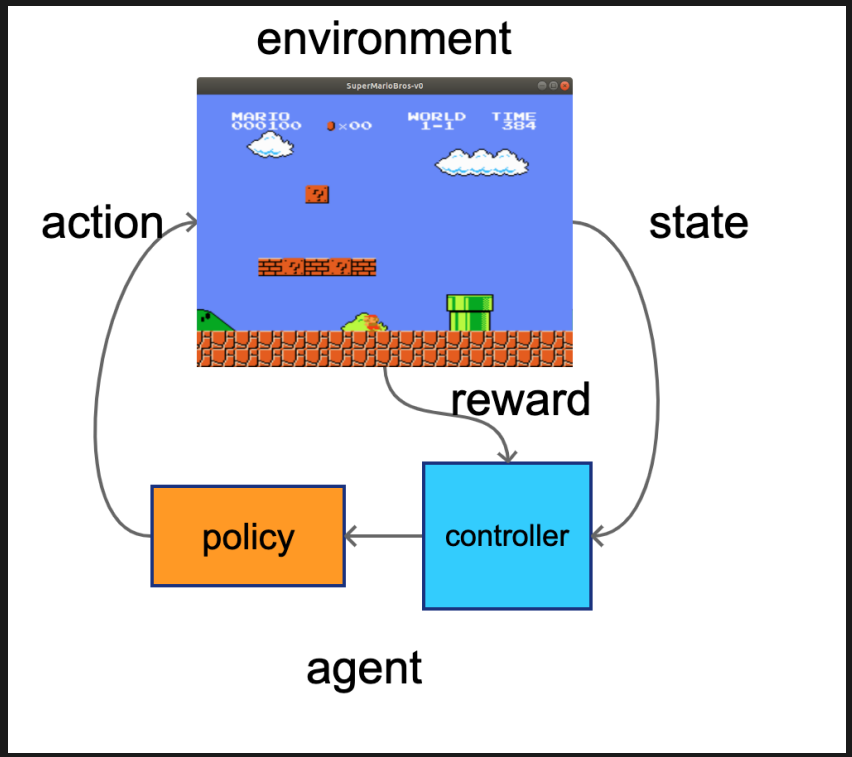


reinforcement learning

learning from punishments and rewards



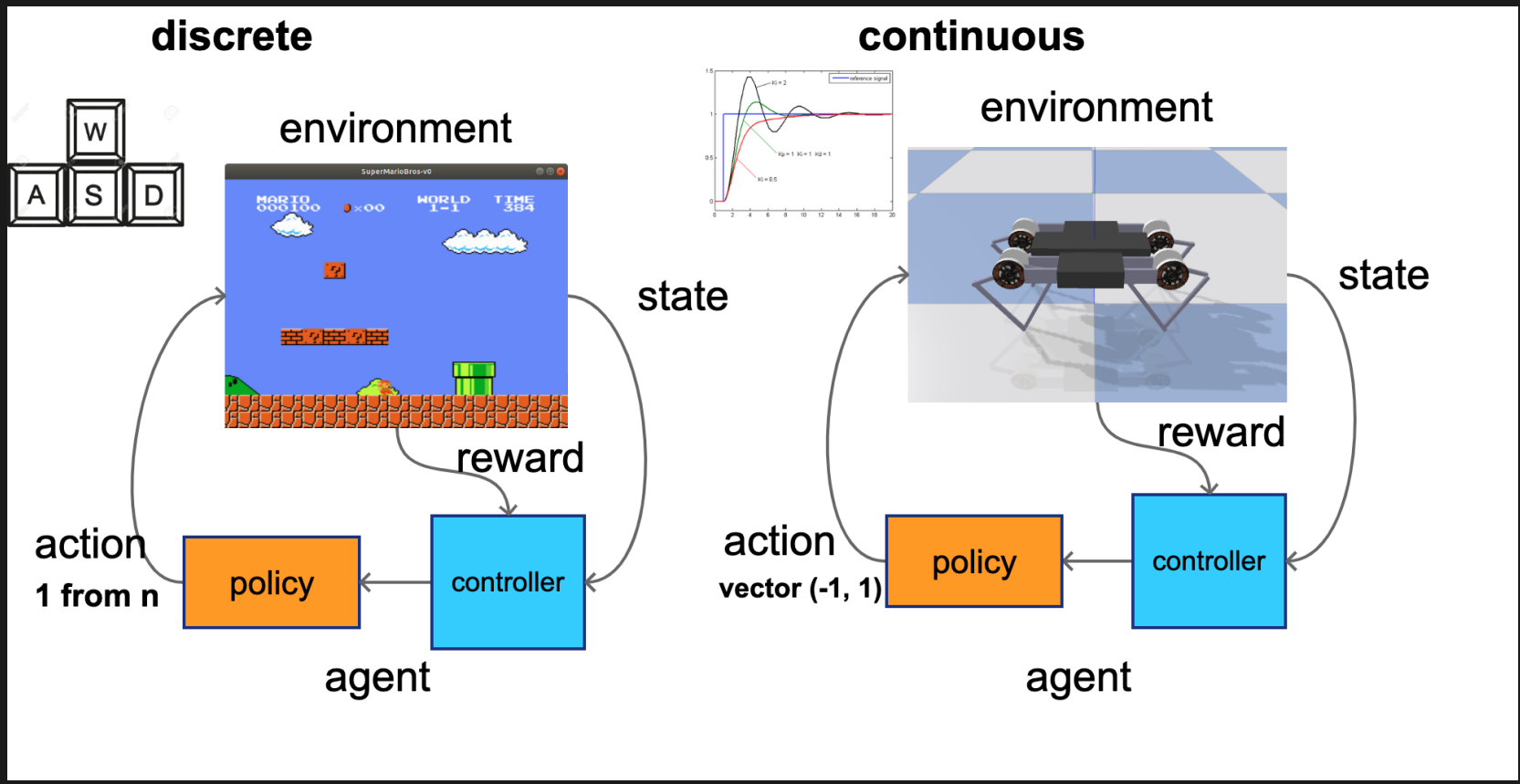
reinforcement learning



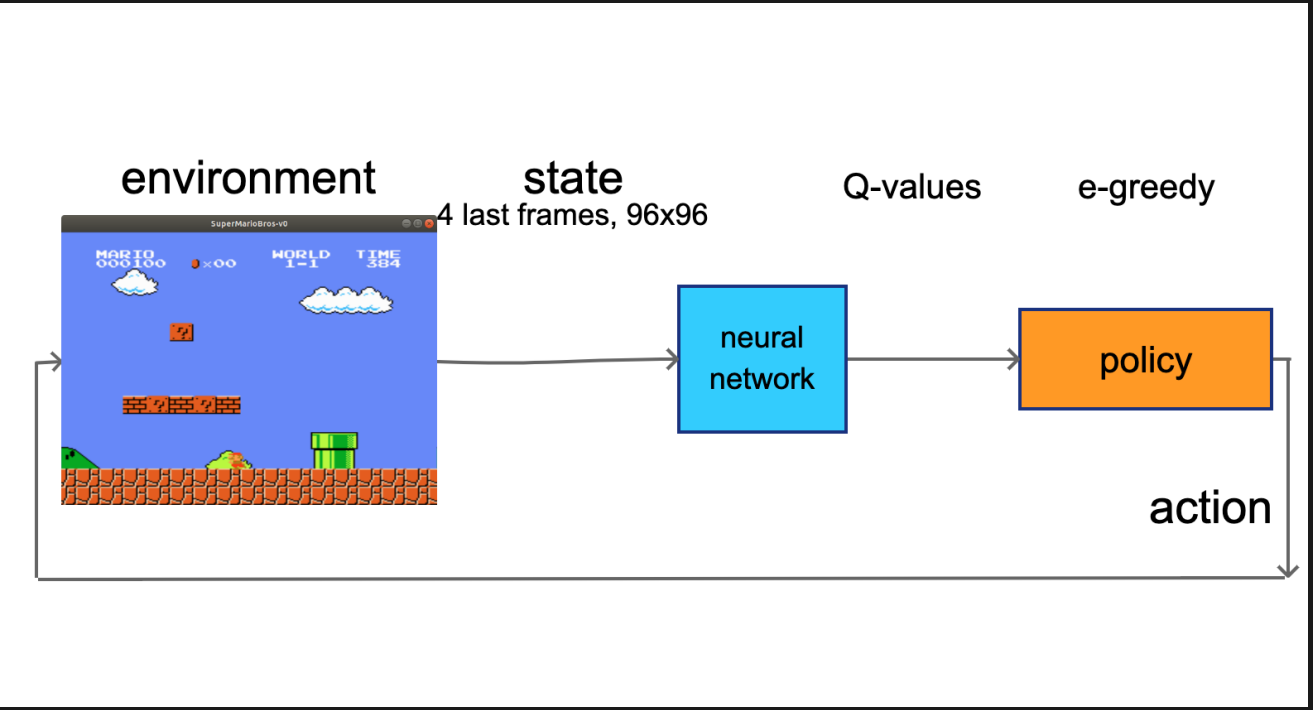
- obtain state
- select action
- exectute action
- learn from experiences

action space

- discrete action space
  - keys, keypad
- continuous action space
  - motors, PWMs, steering, force controll



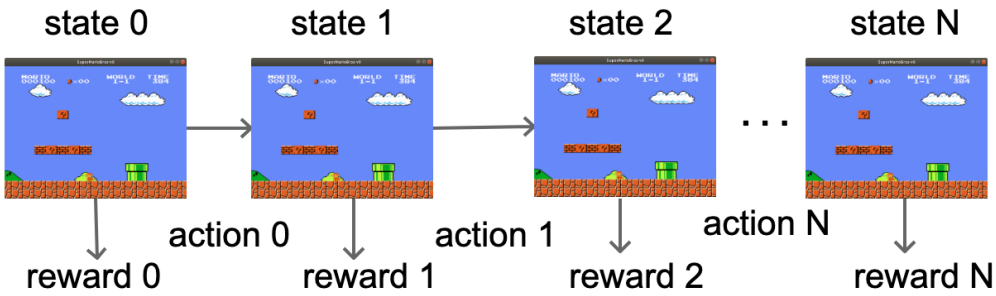
deep Q learning



1. play games
2. store transitions into buffer
   - state, action, reward, done
3. learn from buffer

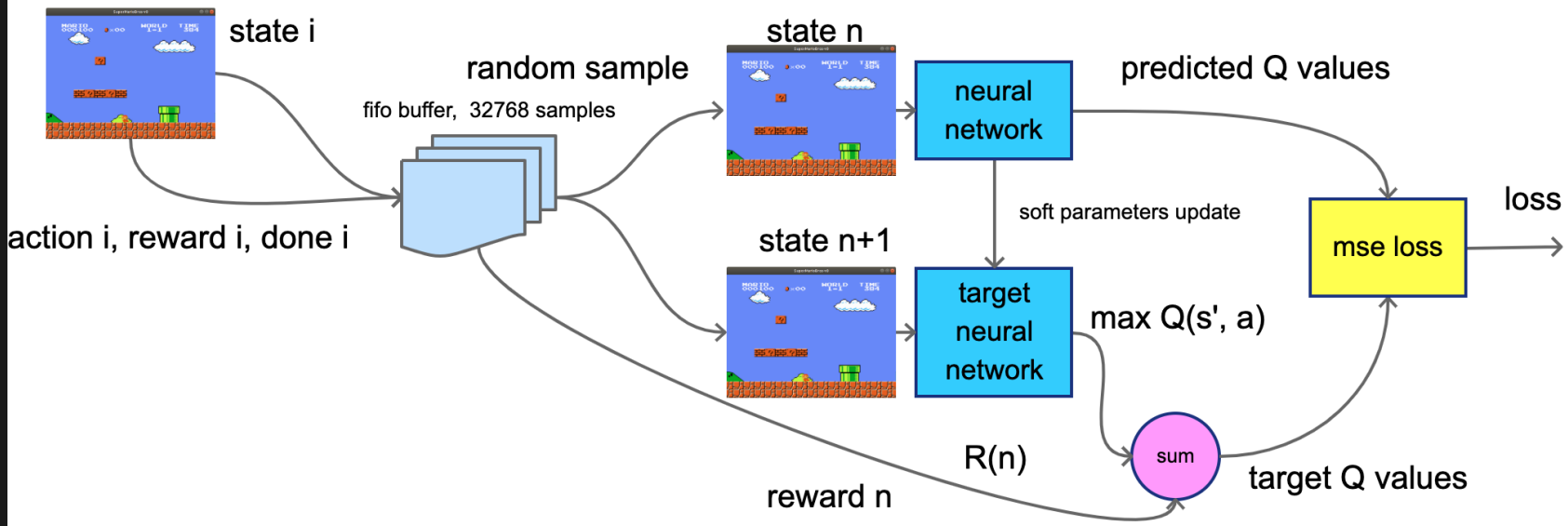
deep Q learning

1, game play

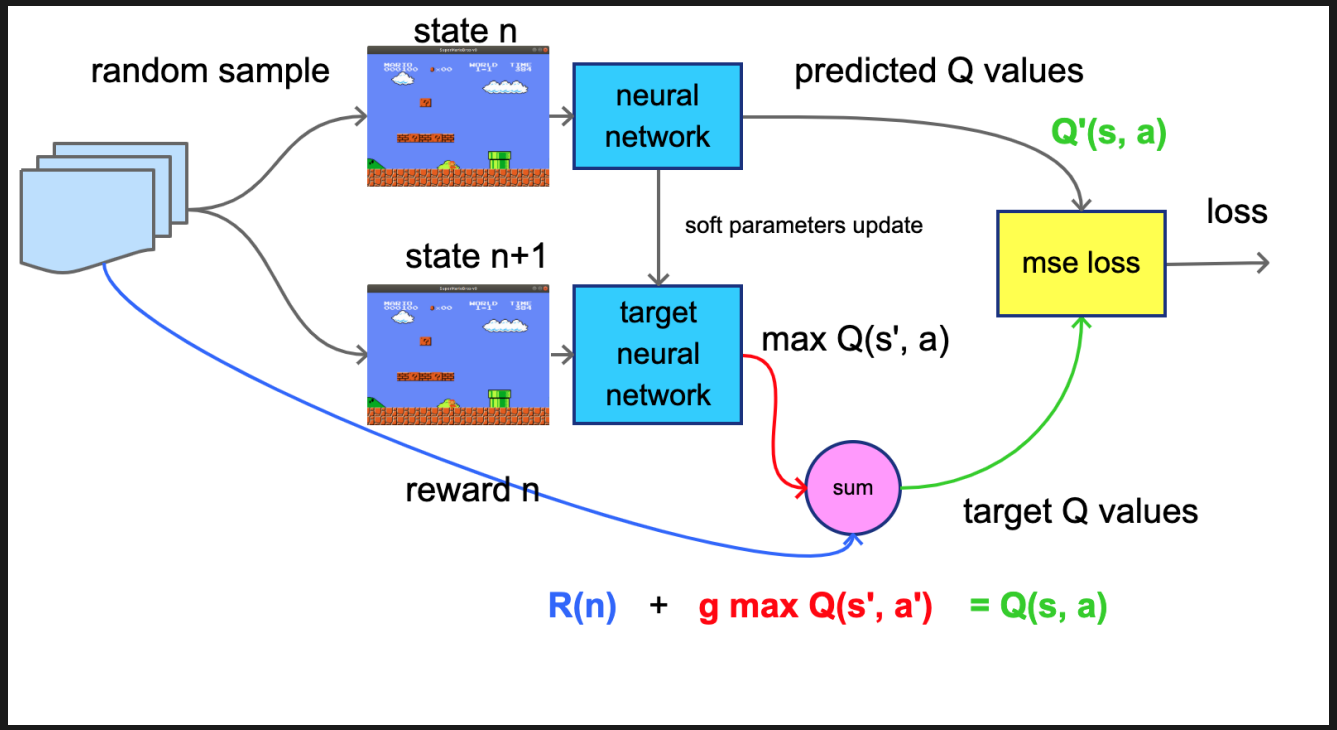


2, experience replay buffer

3, train network



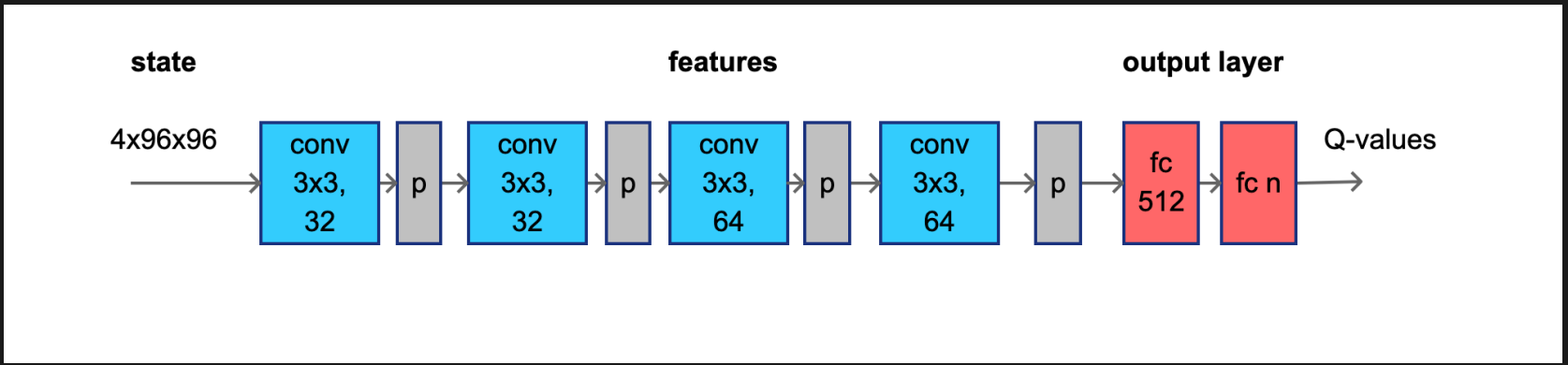
deep Q learning



$$Q(s,a;\theta) = \underbrace{R}_{\textit{reward}} + \underbrace{\gamma \max_{a'} Q(s',a';\theta^{-})}_{\textit{discounted future reward}}$$

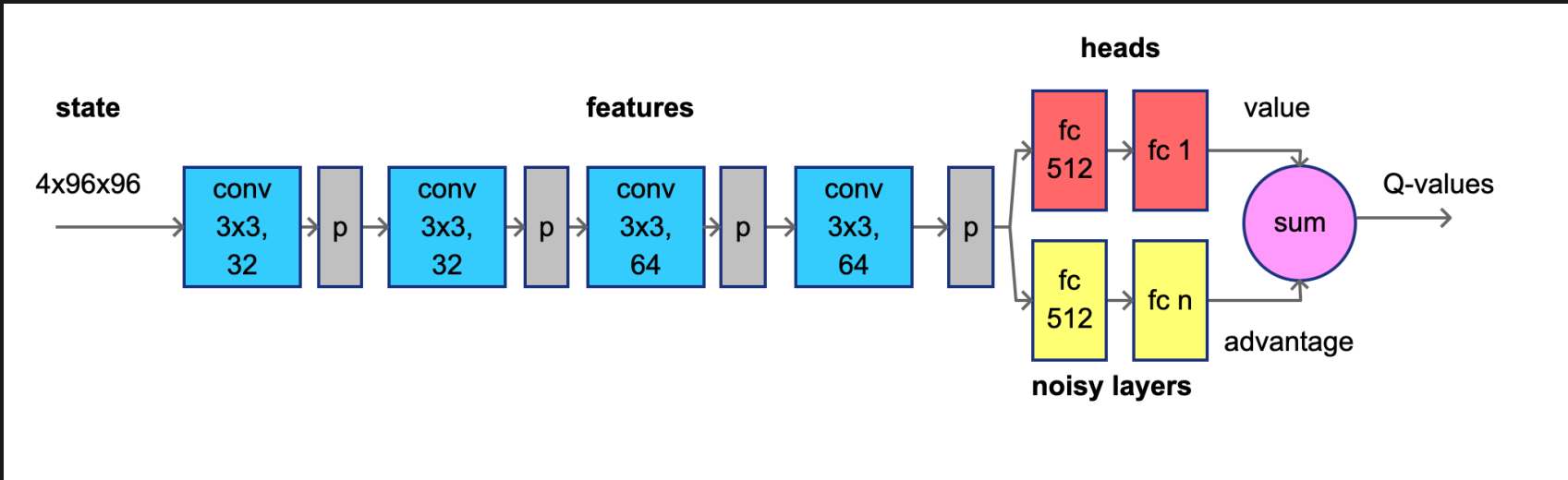
$$\mathcal{L}(\theta) = \left(R + \gamma \max_{a'} Q(s',a';\theta^{-}) - Q(s,a;\theta)\right)^2$$

model architecture



- input 96x96 grayscale, 4 stacked frames
- 3x3 convs + pooling
- two fully connected layers
- small learning rate $\eta = 0.0001$, batch size $= 32$
- $\gamma = 0.99$
- exploration ϵ-greedy, 1M samples linear decay from 1 to 0.05
- total training 10M samples

dueling DQN, model architecture



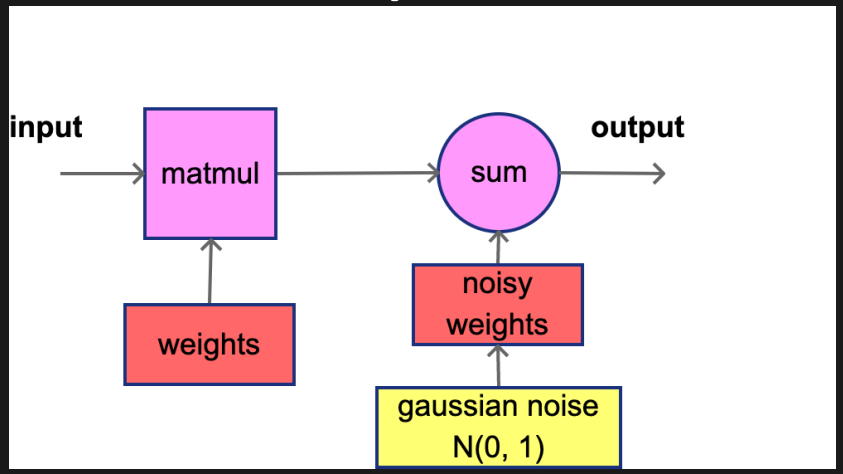
$$Q(s,a) = V(s) + A(s,a)$$

$$Q(s,a) = V(s) + A(s,a) - \frac{1}{|\mathcal{A}|} \sum_{a' \in \mathcal{A}} A(s,a')$$

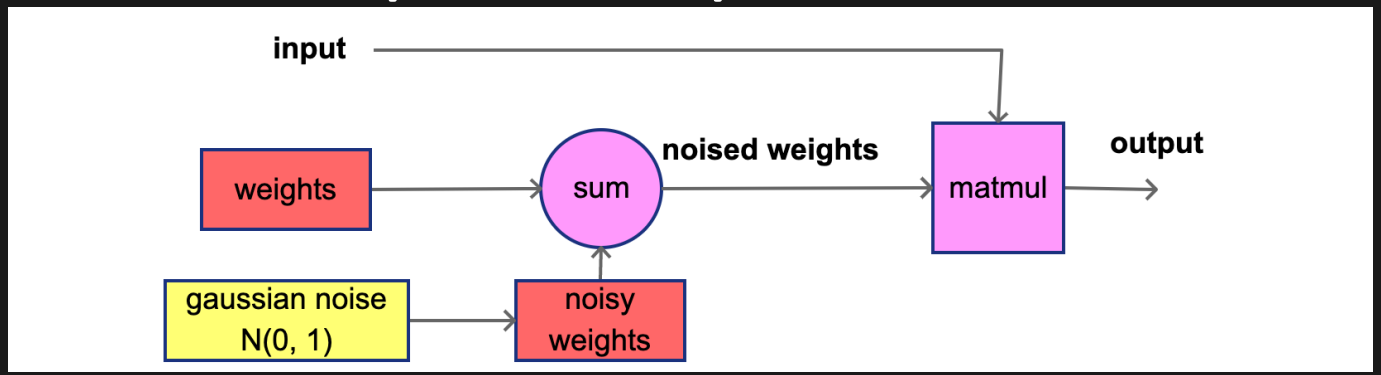
```
WRONG   : q = value + advantage - advantage.mean()
CORRECT : a = value + advantage - advantage.mean(dim=1, keepdim=True)
```

noisy layers for exploration

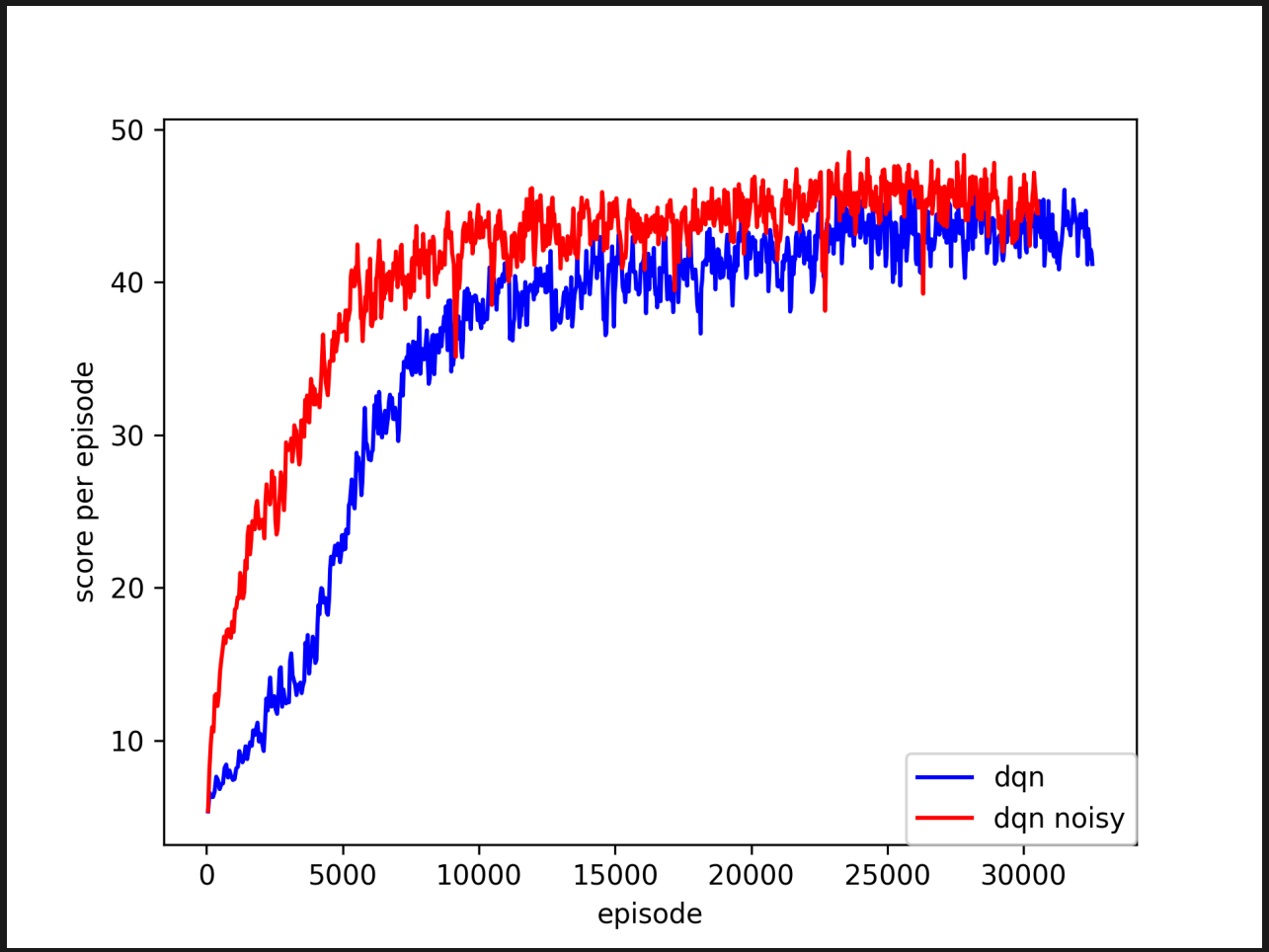
action space noise



parameter space noise



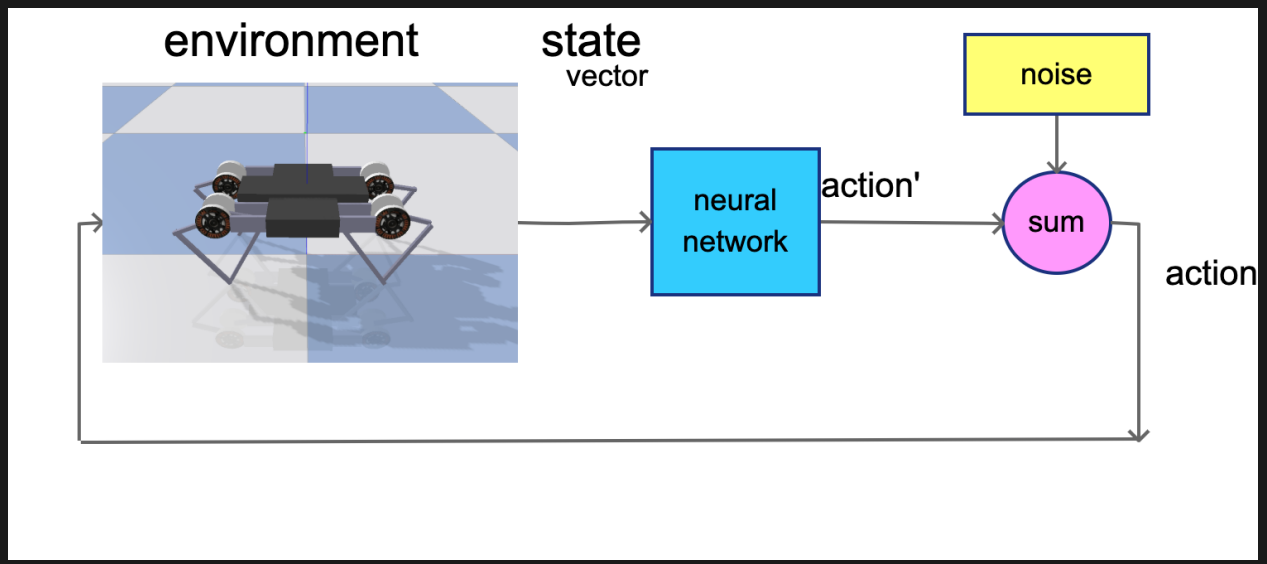
results on MsPacman



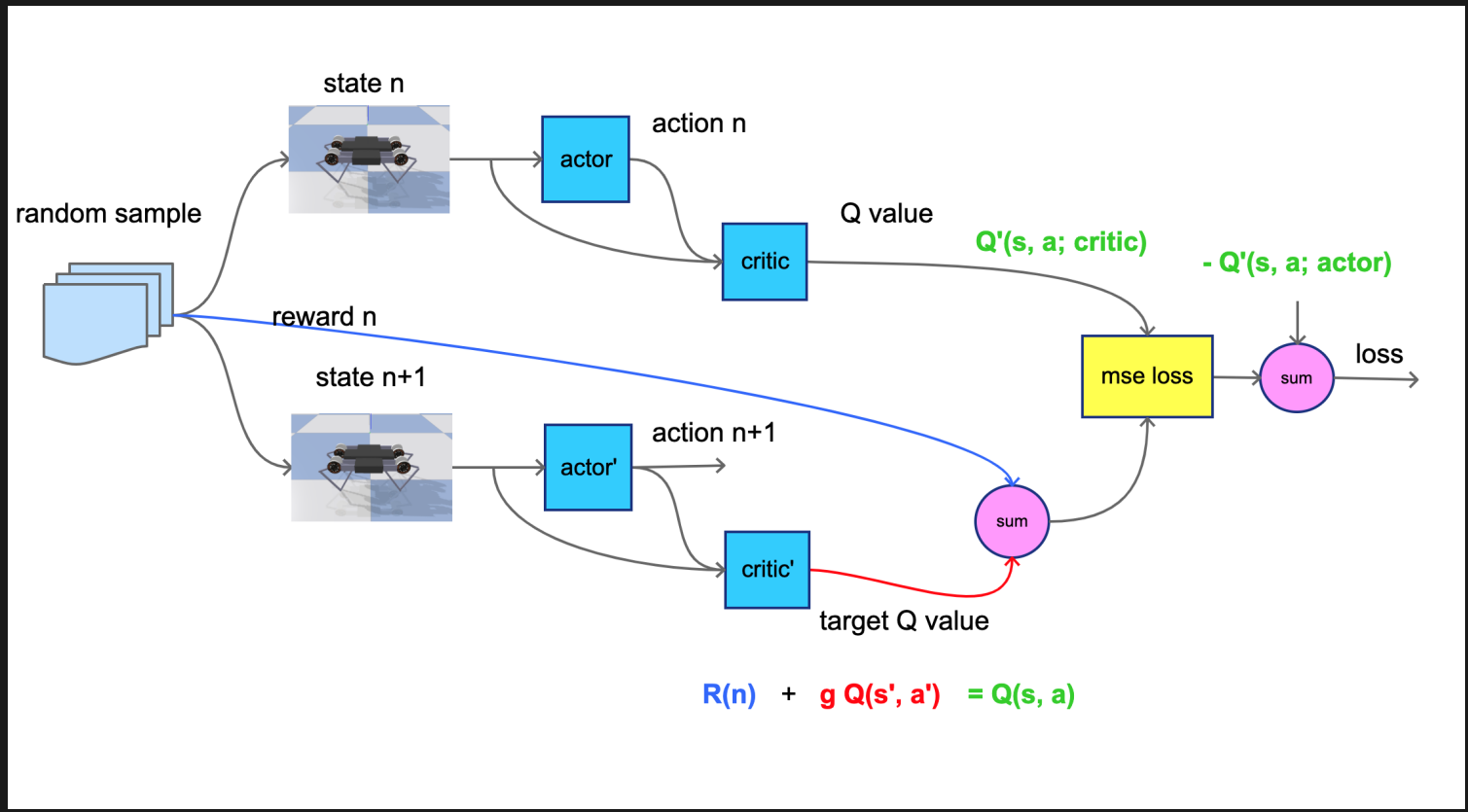
DDPG

deep deterministic policy gradietnt

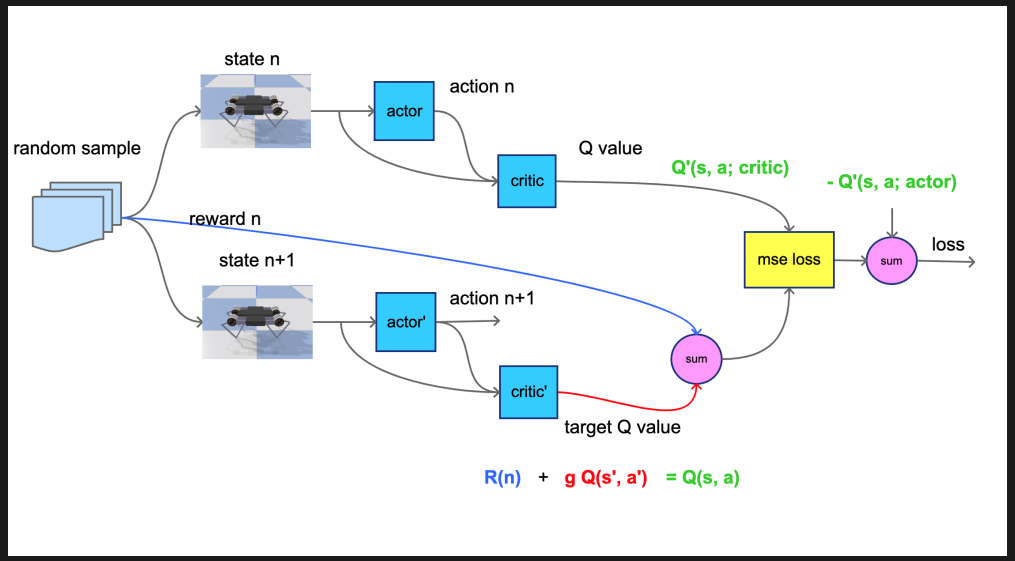
- continuous action space
- natural extension of DQN
- actor-critic structure



DDPG


state n
action n
actor
random sample
Q value
critic
Q'(s, a; critic)
- Q'(s, a; actor)
reward n
mse loss
sum
loss
state n+1
action n+1
actor'
sum
critic'
target Q value
R(n) + g Q(s', a') = Q(s, a)

DDPG

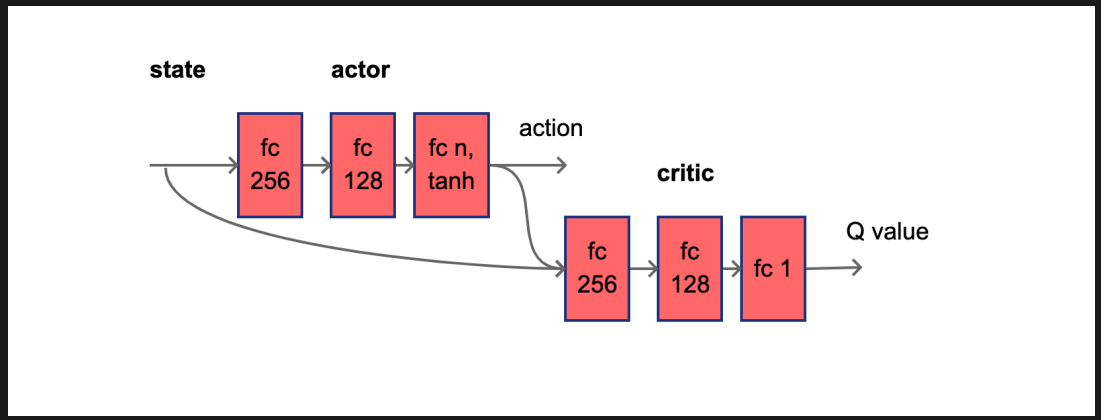


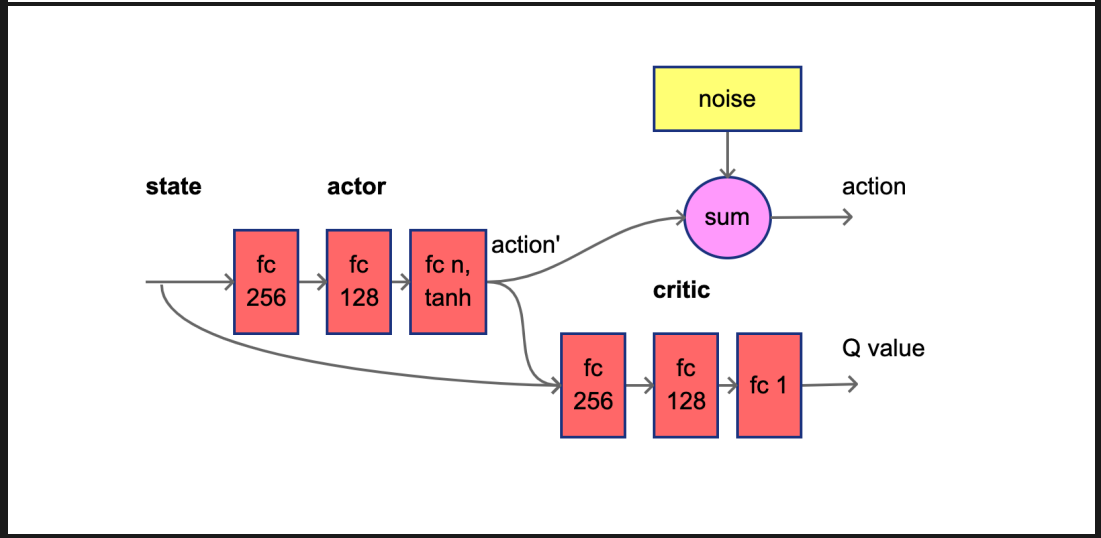
$$\mathcal{L}(\theta) = \left(R + \gamma Q(s', A(s'; \phi^-); \theta^-) - Q(s, A(s; \phi); \theta)\right)^2$$
$$\mathcal{L}(\phi) = -Q(s, A(s; \phi); \theta)$$

where

- Q is critic network with parameters θ
- A is actor network with parameters ϕ

model architecture





books to read

- Maxim Lapan, 2020, Deep Reinforcement Learning Hands-On second edition
- Maxim Lapan, 2018, Deep Reinforcement Learning Hands-On
- Praveen Palanisamy, 2018, Hands-On Intelligent Agents with OpenAI Gym
- Andrea Lonza, 2019, Reinforcement Learning Algorithms with Python
- Rajalingappaa Shanmugamani, 2019, Python Reinforcement Learning
- Micheal Lanham, 2019, Hands-On Deep Learning for Games

Q&A

michal.nand@gmail.com

https://github.com/michalnand/imagination_reinforcement_learning