UNIVERSITY OF LIVERPOOL

COMP 532
Machine Learning and BioInspired Optimization

Lecture 14: Deep Learning

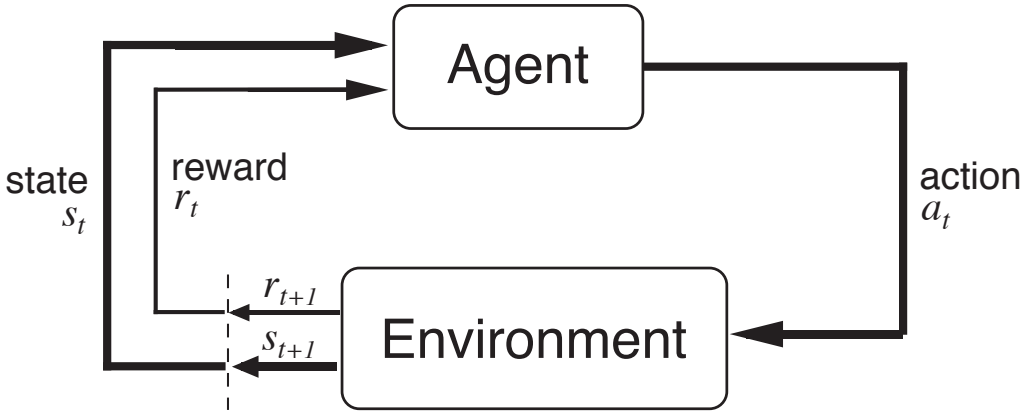
Dr. Shan Luo

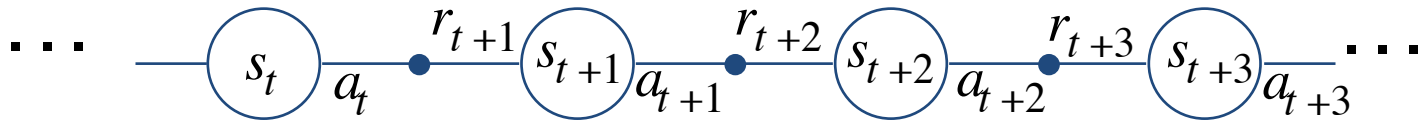
Department of Computer Science

shan.luo@liverpool.ac.uk

Recap

- Reinforcement learning
  - Agent-Environment Interface





Recap

- Reinforcement learning
  - Action-value function

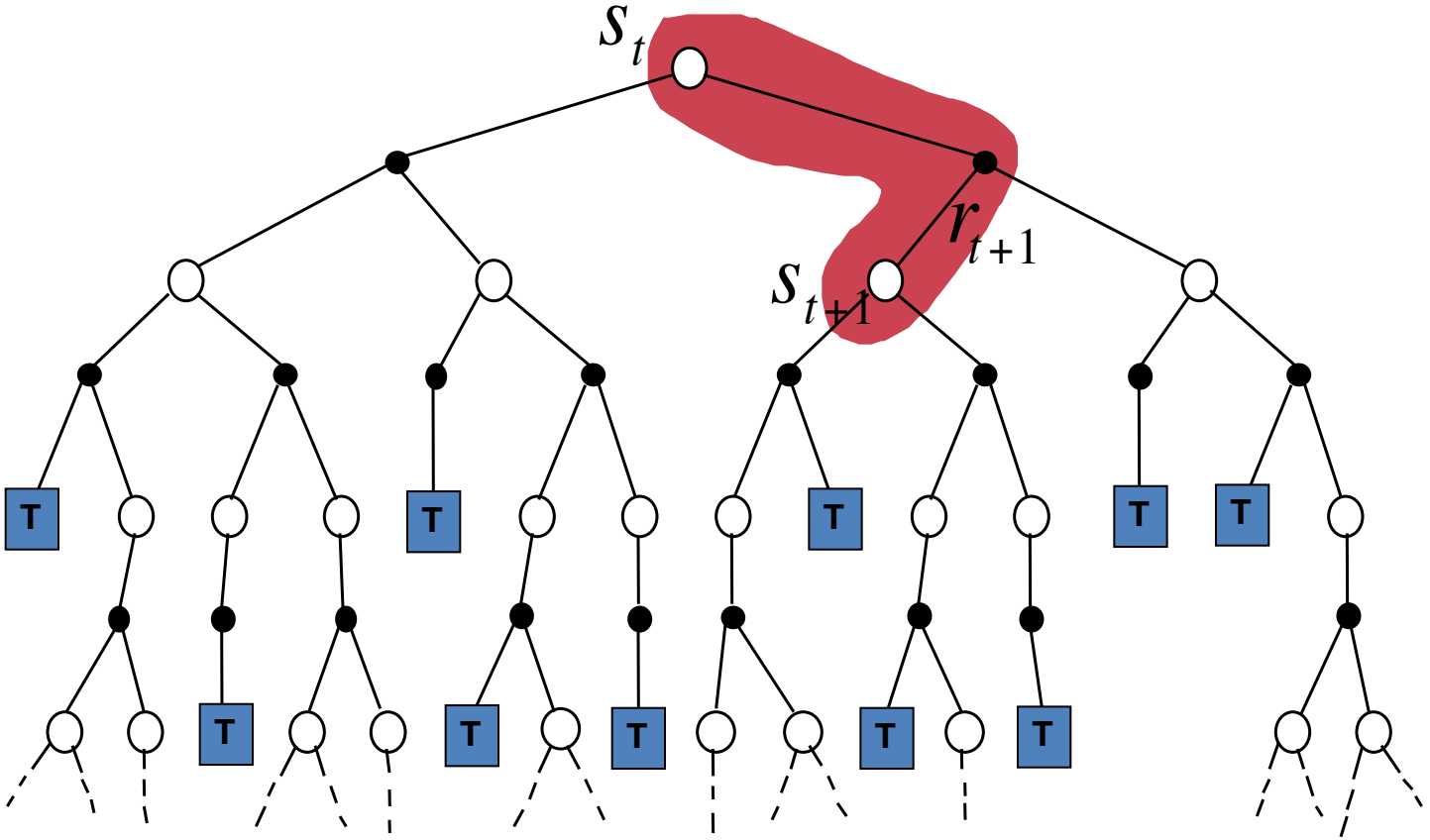
The **value of taking an action in a state under policy π** is the expected return starting from that state, taking that action, and thereafter following π:

Action-value function for policy π:

$$Q^{\pi}(s,a) = E_{\pi}\left\{R_t \mid s_t = s, a_t = a\right\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^{k} r_{t+k+1} \mid s_t = s, a_t = a\right\}$$

Recap

- Reinforcement learning
  - Temporal Difference (TD) method



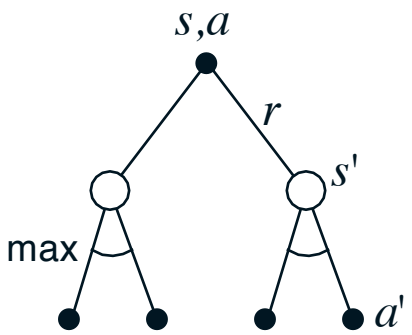
Recap

- Reinforcement learning
  - Bellman Optimality Equation for Q^*

$$Q^*(s,a) = E\left\{ r_{t+1} + \gamma \max_{a'} Q^*(s_{t+1},a') \,\middle|\, s_t = s, a_t = a \right\}$$

$$= \sum_{s'} \mathbf{P}^a_{ss'} \left[\mathbf{R}^a_{ss'} + \gamma \max_{a'} Q^*(s',a') \right]$$

The relevant backup diagram:



Q^* is the unique solution of this system of nonlinear equations.

Recap

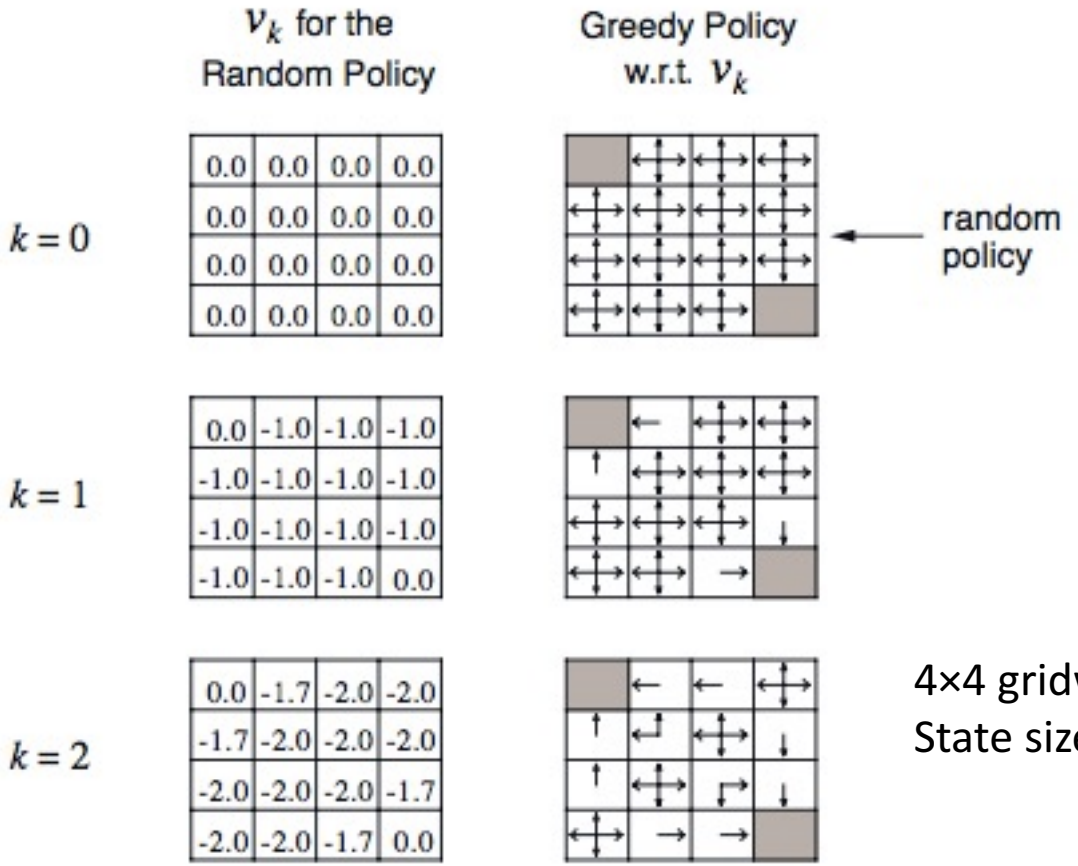
- Reinforcement learning
  - Q-learning: off-policy TD control

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

Initialize $Q(s, a)$ arbitrarily
Repeat (for each episode):
  Initialize s
  Repeat (for each step of episode):
    Choose a from s using policy derived from Q (e.g., ε-greedy)
    Take action a, observe r, s'
    $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$
    $s \leftarrow s'$;
  until s is terminal

Recap

- Reinforcement learning
  - Gridworld



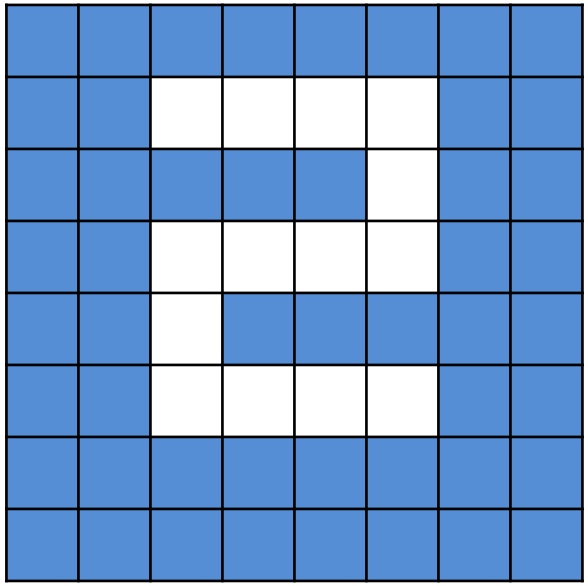
4×4 gridworld
State size: 4x4=16

Overview

- Why apply DL in RL?
  - State size explosion
  - Function approximator
- Deep Q-Learning
  - Architecture
  - Atari case study
- Gym-OpenAI

Recap + Inspiration

- Reinforcement learning
  - What if an 8x8 image? State size?



256^{100}!

We can't get a table for such a large state space.

Recap + Inspiration

- Reinforcement learning
  - What about larger images?

[210, 160, 3] Atari RGB image

Recap + Inspiration

- Reinforcement learning
  - What about larger images?

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

What's the problem with this?
Not scalable. Must compute Q(s,a) for every state-action pair. If state is e.g., current game state (image) pixels, computationally infeasible to compute for entire state space!

Solution: use a function approximator to estimate Q(s,a), e.g., a neural network!

Deep Q-learning

- Q-learning: Solving for the optimal policy
- Use a function approximator to estimate the action-value function

$$\mathrm{Q}(s_t, a_t; \theta) \approx Q^*(s, a)$$

θ: function parameters (weights)

If the function approximator is a dep neural network
=> Deep Q-learning

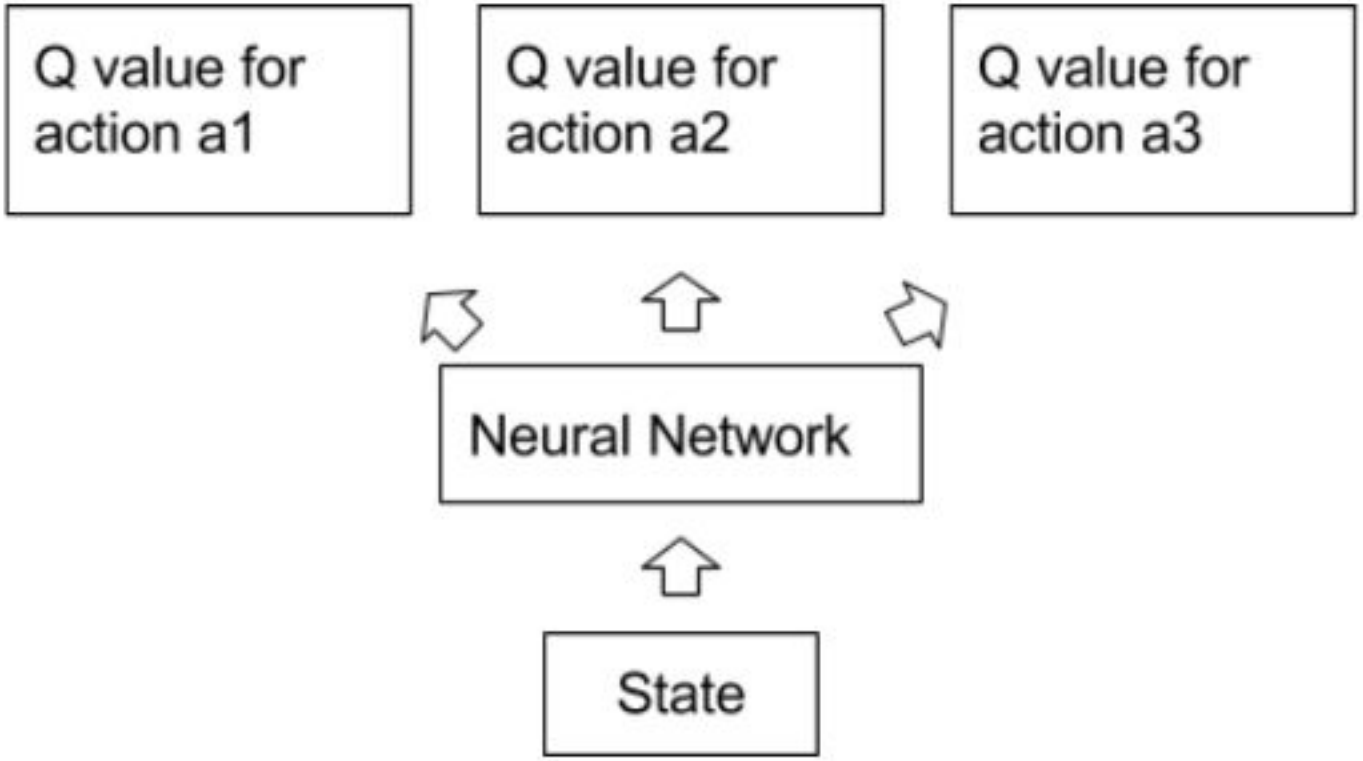
Deep Q-learning

$$Q^*(s,a) = E\left\{ r_{t+1} + \gamma \max_{a'} Q^*(s_{t+1},a') \,\middle|\, s_t = s, a_t = a \right\}$$

- Q-learning: Solving for the optimal policy
- Forward pass:
  - Loss function: $L_i(\theta_i) = E\left[\left(y_i - Q(s,a;\theta_i)\right)^2\right]$
    where $y_i = E\left[r_{t+1} + \gamma \max_{a'} Q(s',a';\theta_{i-1}) | s,a\right]$
- Backward pass:
  - Gradient update (with respect to Q-function parameters θ)

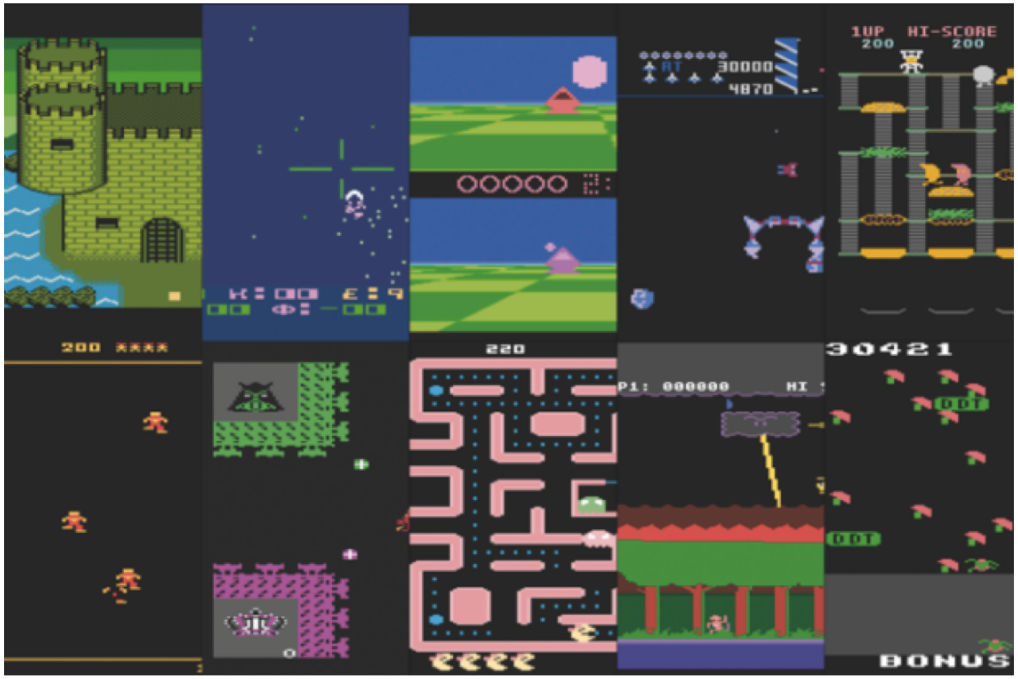
$$\nabla_{\theta_i} L_i(\theta_i) = E\left[\left(r_{t+1} + \gamma \max_{a'} Q(s',a';\theta_{i-1}) - Q(s,a;\theta_i)\right) \nabla_{\theta_i} Q(s,a;\theta_i)\right]$$

Deep Q-learning

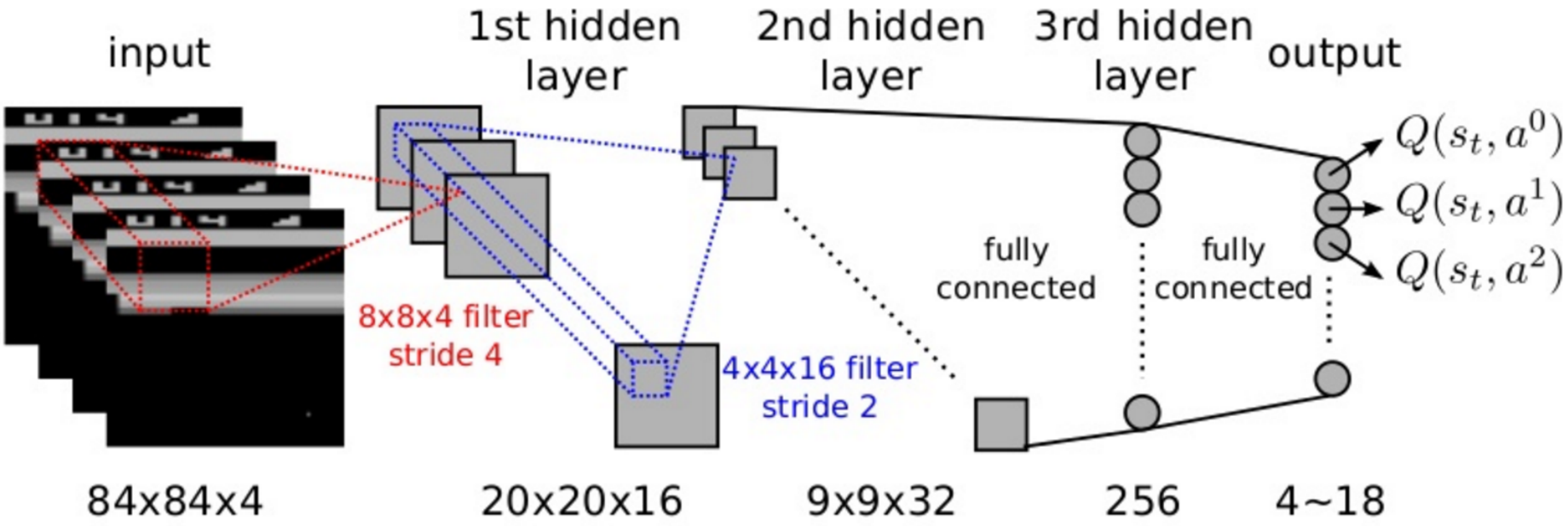


Case Study: Playing Atari Games

- **Objective:** Complete the game with the highest score
- **State:** Raw pixel inputs of the game state
- **Action:** Game controls e.g., Left, Right, Up, Down
- **Reward:** Score increase/decrease at each time step



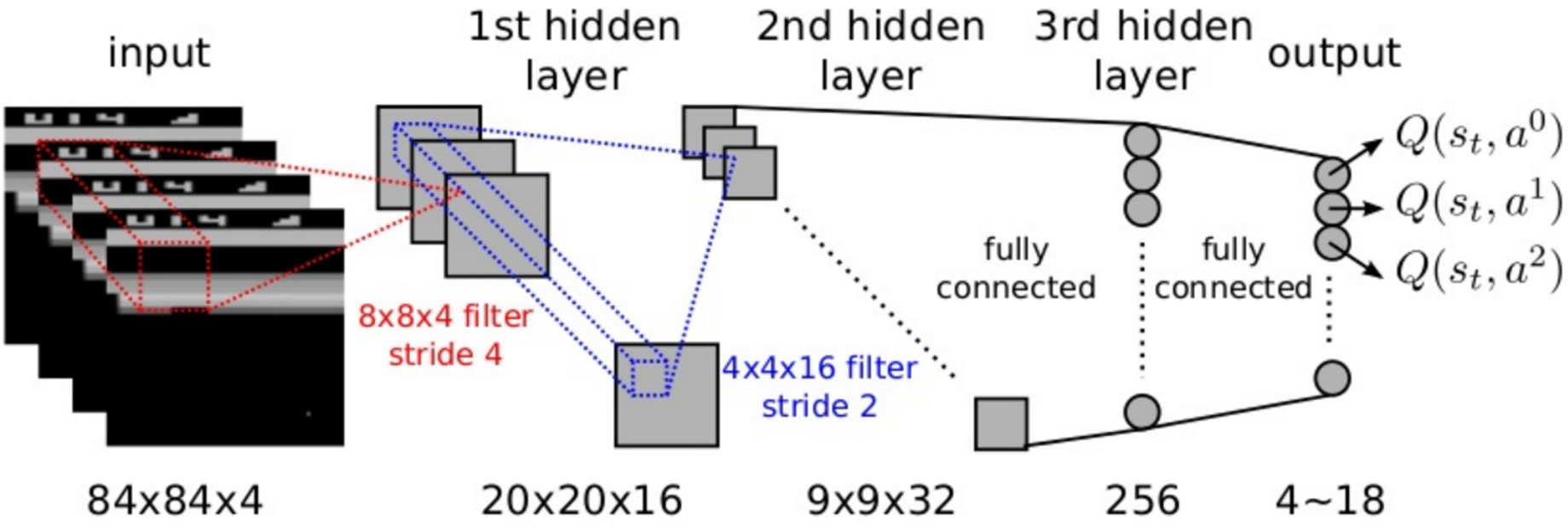
Case Study: Playing Atari Games



Input: 84x84x4 stack of last 4 frames

(After RGB->grayscale conversion, down-sampling and cropping)

Case Study: Playing Atari Games



Input: 84x84x4
stack of last 4 frame

2xConv layers

2xFC layers

Output

Case Study: Playing Atari Games

https://www.youtube.com/watch?v=V1eYniJ0Rnk

Volodymyr *et al.* “Human-level control through deep reinforcement learning." *Nature* 2015.

Case Study: AlphaGo

- Ground breaking result in 2016
  - For the first time, a human professional Go player is beaten by an AI: Google DeepMind's AlphaGo

- Based on

  convolutional networks..

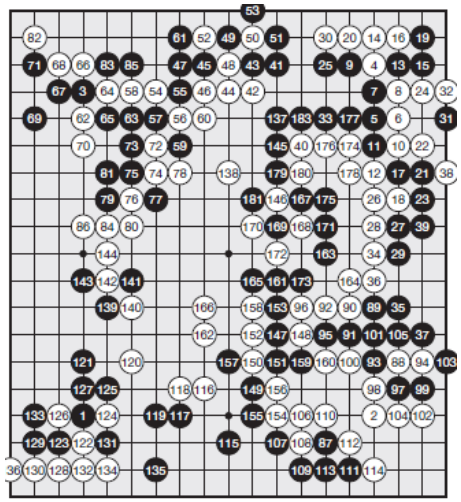
  ..reinforcement learning..

  ..and Monte Carlo search

From Chess to Go

1997, an AI called “Deep Blue” beat the chess world champion Kasparov.

Search space: b^d: $b = 35, d = 80$



Search space: b^d: $b = 250, d = 150$

Case Study: AlphaGo

AI: AlphaGo

AlphaGo

European champion: Fan Hui

Silver, et al. "Mastering the game of Go with deep neural networks and tree search." *nature* 2016.

Key Points

- How to evaluate all possible positions and moves?
  - **Convolutional networks**!
  - Reduce state space to hierarchical set of features

- Combination of **supervised learning**, based on historical data from human game play..

  ..and **reinforcement learning**, in self-play, to improve the learnt policy

  (and a bit of **Monte Carlo** search)

Gym-OpenAI

- A toolkit for developing and comparing reinforcement learning algorithms.
- Open source interface to reinforcement learning tasks.
- https://gym.openai.com

The Takeaway Message

- Convolutional Networks are not only suited for image processing!

- A combination of techniques can outperform the individual parts
  - Current hot topic: Deep Reinforcement Learning

- But: *"robots will never understand the beauty of the game the same way that we humans do"*
  – Lee Sedol, Go world champion