

# Advanced Topics in Reinforcement Learning

Lecture 15: Deep Reinforcement Learning II

Josiah Hanna

University of Wisconsin — Madison

# Announcements

- Homework due October 21 at 9:30AM (minute class starts)
- Read Chapter 13 for next week. Policy-based RL!
- Upcoming dates:
  - Literature survey due next week: October 30
  - Exam: November 6

# Learning Outcomes

After this week, you will be able to:

1. Describe key benefits and challenges of using neural networks as function approximators in RL.
2. Describe how deep learning techniques are used within RL environments.
3. Implement key deep RL techniques such as target networks and experience replay.

# Neural Network Training in RL

- Semi-Gradient Q-learning:

- $$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \alpha(R_{t+1} + \gamma \max_{a'} \hat{q}(S_{t+1}, a', \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t)) \nabla \hat{q}(S_t, A_t, \mathbf{w}_t)$$

- The parameter,  $\mathbf{w}_t$ , is all weights and biases of the neural network.
- Backpropagation algorithm: use chain rule of calculus to derive gradient of network outputs with respect to each weights or bias of the network.
- Adjust each weight in proportion to gradient of output times TD-error.

# Darsh's Presentation

- Slides

# DQN Overview

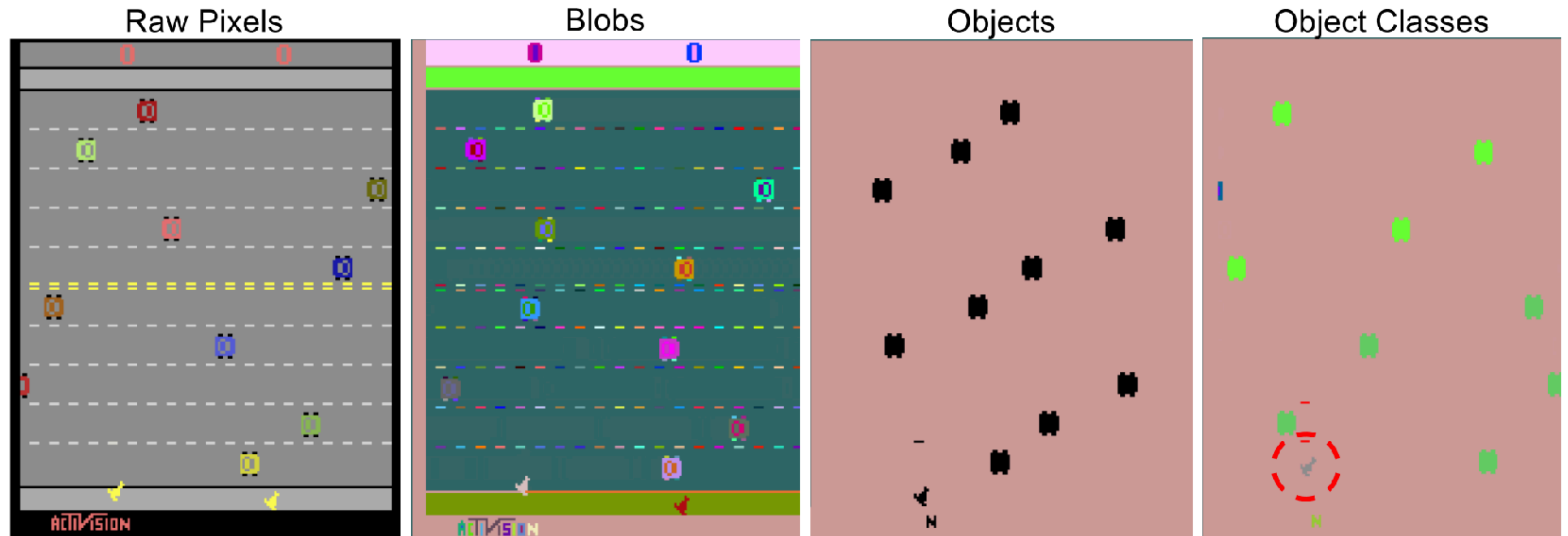
- Single algorithm, neural network architecture, and hyper-parameter setting that played 49 Atari video games at human-level.
- Training and evaluation is independent for each game.
  - The final neural network from training on “Breakout” cannot play “Pong.”
- Landmark result for deep reinforcement learning.

# The Atari 57 Benchmark

- 57 Atari video games turned into RL benchmarks
- Why were Atari games hard for reinforcement learning algorithms?
  - Representation learning; hyper-parameter robustness
  - Prior state-of-the-art: neuroevolution and then Deepmind's predecessor to deep Q-learning.
- With a suitable representation, some games are simpler than others.

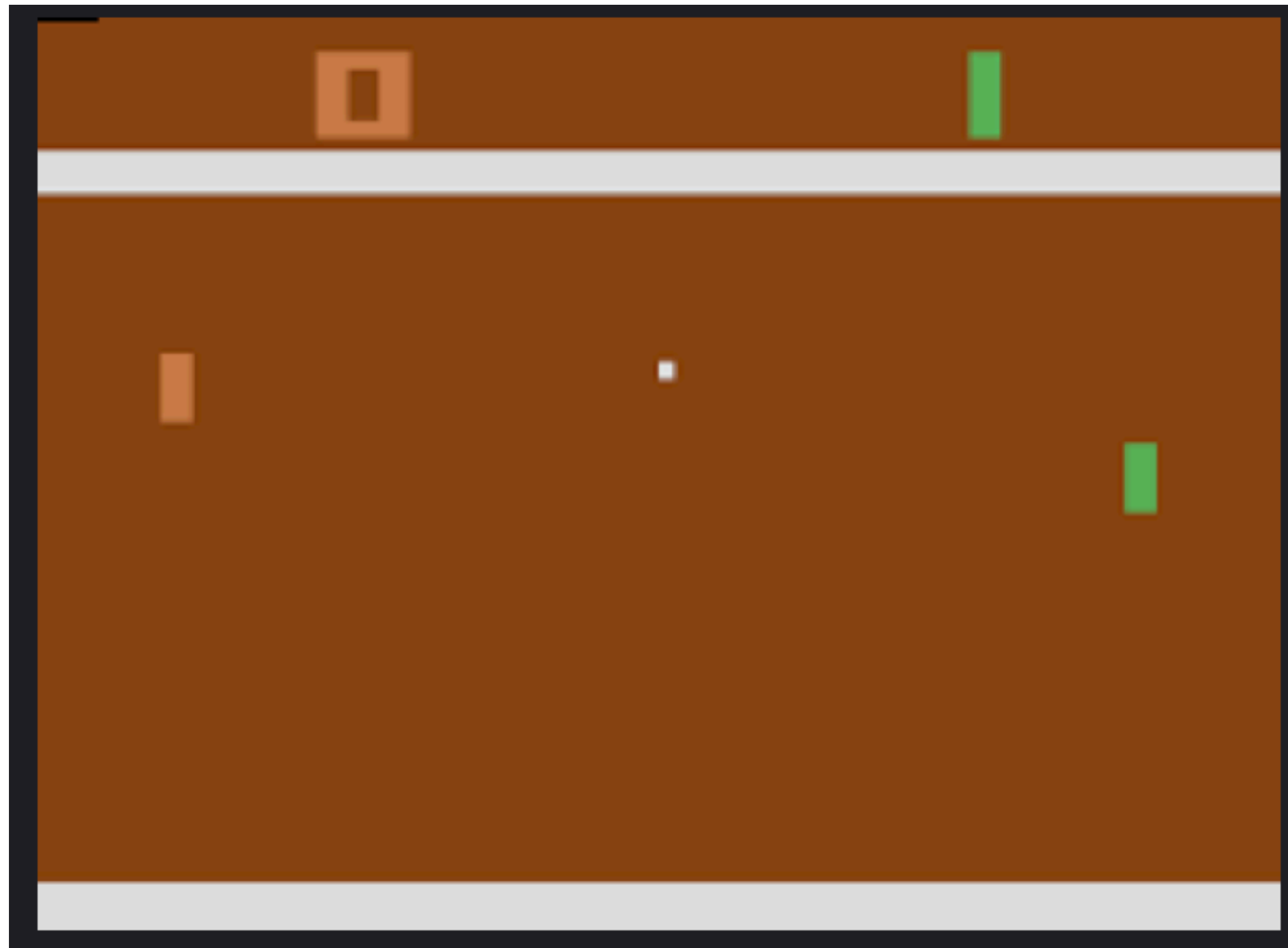
“HyperNEAT-GGP: A HyperNEAT-based Atari General Game Player.” Hausknecht, Khandelwal, Miikkulainen, and Stone. 2012.

# Feature Engineering



“HyperNEAT-GGP: A HyperNEAT-based Atari General Game Player.” Hausknecht, Khandelwal, Miikkulainen, and Stone. 2012.

# Easy and Hard Games in Atari



**Pong**

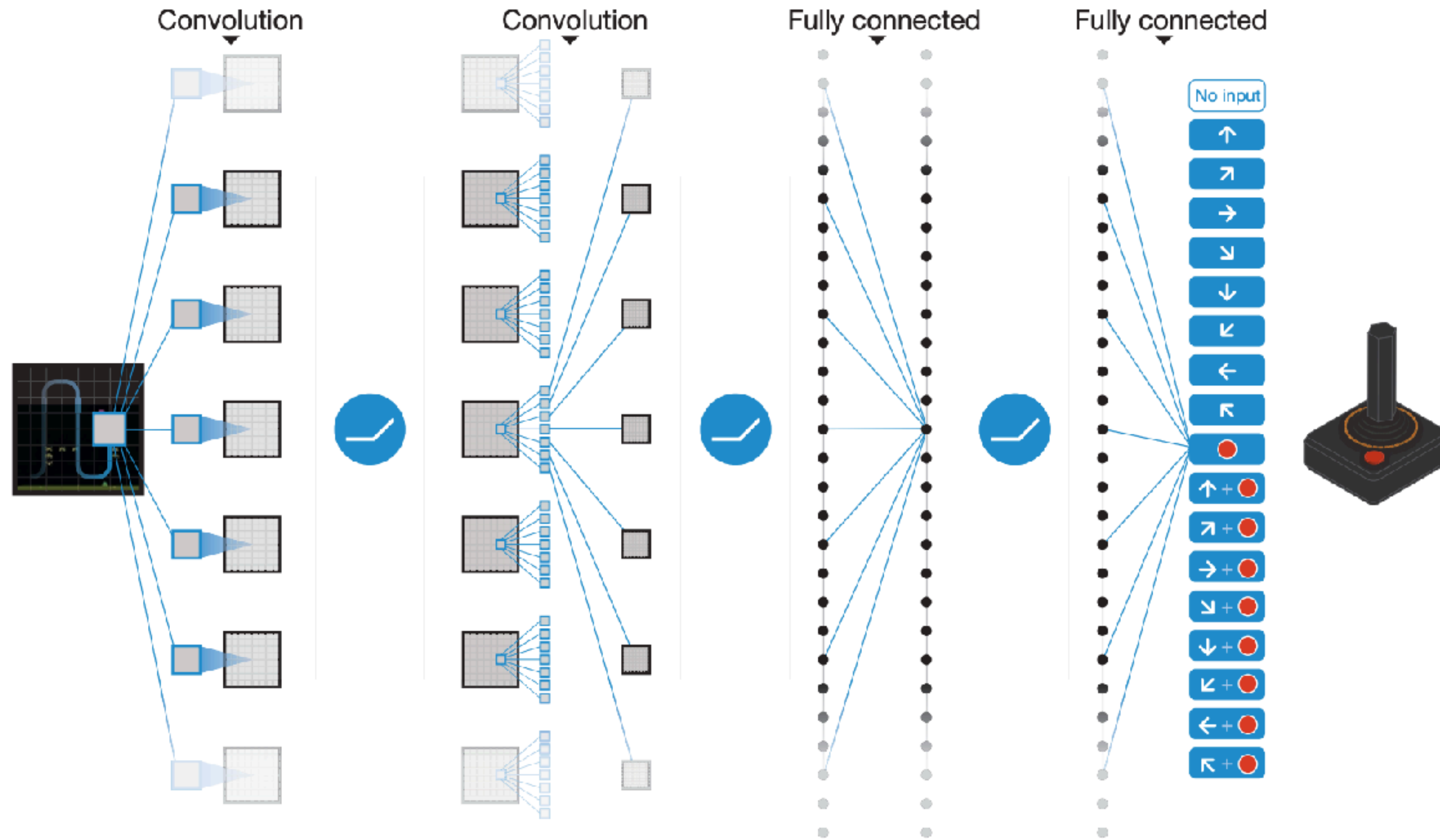


**Montezuma's Revenge**

# DQN Architecture

- Core algorithm is semi-gradient Q-learning with a convolutional neural network as the function approximator.
- Key techniques for effective training across tasks:
  - Pre-processing
  - Convolutional neural network
  - Experience replay
  - Target networks
  - Reward clipping

# DQN Architecture

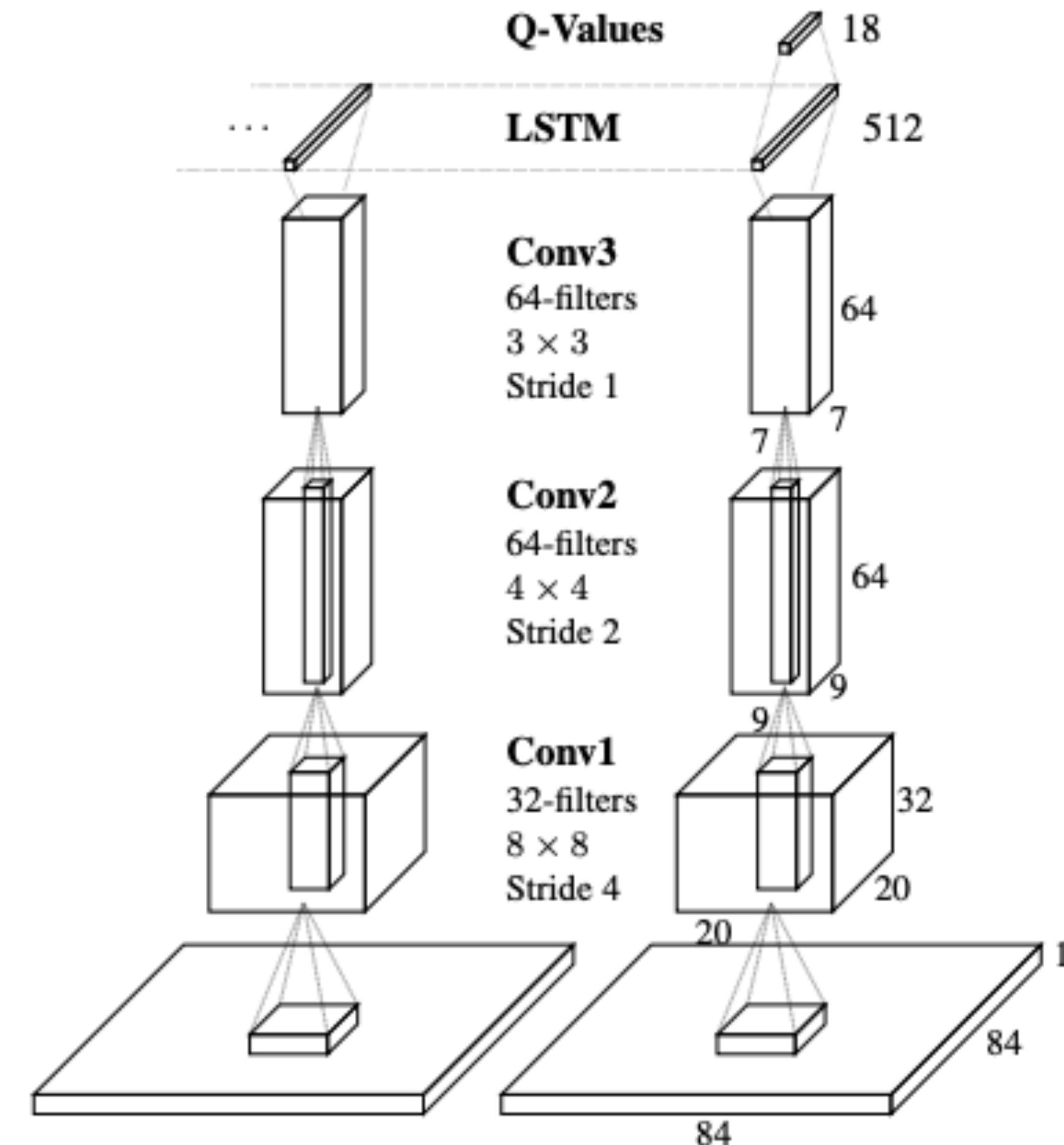


# Pre-processing

- Large RGB images take a lot of memory.
  - Solution: downsample and turn the image to greyscale.
- Images are non-Markovian observations of state.
  - Solution: frame-stacking, i.e., concatenate past four frames together.
  - The agent repeats the same action for four consecutive frames and then can choose a new action.

# Recurrent Neural Networks

- Recurrent neural networks update a hidden state that is an input to computations at the next time-step.
- Allows networks to remember previously seen inputs when computing future outputs.
- In RL, provides a method to learn a Markov state.
- Alternative to frame-stacking.



# Experience Replay

- The basic semi-gradient Q-learning algorithm processes  $(s, a, s', r)$  transitions as they are experienced and then discards them.
- Experience replay: keep around the most recent transitions (in DQN, the past 1 million) and use a random subset to update the action-value function.
  - Uses samples multiple times.
  - Reduces correlation between samples.
- Other choices besides random subset can improve performance.

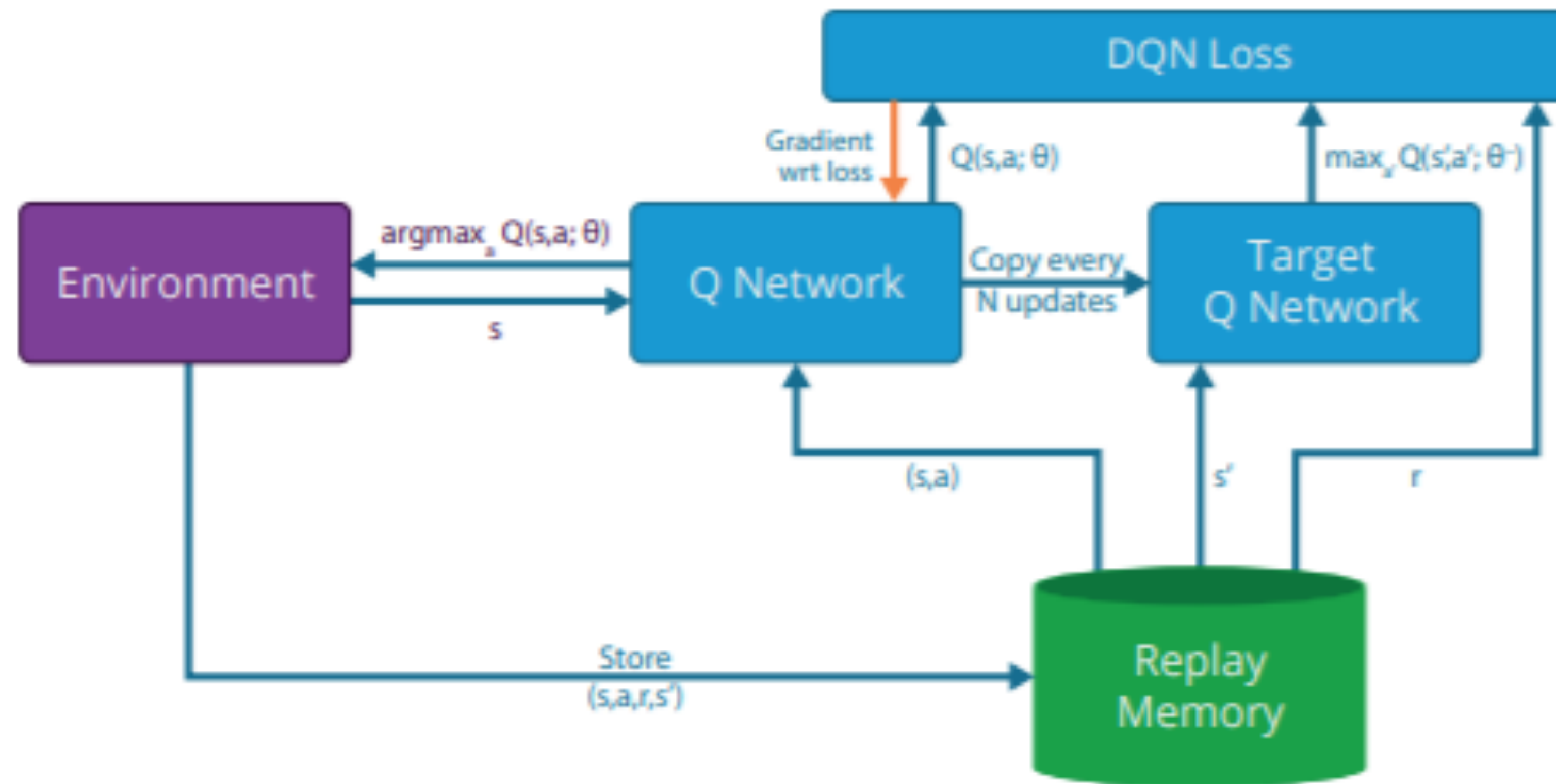
# Target Networks

- The basic semi-gradient Q-learning algorithm always uses the most recent parameters to form the training target  $R_{t+1} + \gamma \max_{a'} \hat{q}(S_{t+1}, a', \mathbf{w})$
- DQN uses a separate **target network** to compute  $\gamma \max_{a'} \hat{q}(S_{t+1}, a', \tilde{\mathbf{w}})$ .
  - The target network is infrequently updated by setting the target network parameters to be the same as the main network's parameters.
  - Makes the learning target more stable as in supervised learning.

# Reward clipping

- Different Atari games have different reward magnitudes.
- Why is this a challenge?
  - Hard to tune a step-size that works across all games.
- Solution: clip rewards to be between -1 and 1.

# DQN Architecture



# Overall Algorithm

**Algorithm 1: deep Q-learning with experience replay.**

Initialize replay memory  $D$  to capacity  $N$

Initialize action-value function  $Q$  with random weights  $\theta$

Initialize target action-value function  $\hat{Q}$  with weights  $\theta^- = \theta$

**For** episode = 1,  $M$  **do**

    Initialize sequence  $s_1 = \{x_1\}$  and preprocessed sequence  $\phi_1 = \phi(s_1)$

**For**  $t = 1, T$  **do**

        With probability  $\varepsilon$  select a random action  $a_t$

        otherwise select  $a_t = \operatorname{argmax}_a Q(\phi(s_t), a; \theta)$

        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  $D$

        Sample random minibatch of transitions  $(\phi_j, a_j, r_j, \phi_{j+1})$  from  $D$

        Set  $y_j = \begin{cases} r_j & \text{if episode terminates at step } j+1 \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^-) & \text{otherwise} \end{cases}$

        Perform a gradient descent step on  $(y_j - Q(\phi_j, a_j; \theta))^2$  with respect to the network parameters  $\theta$

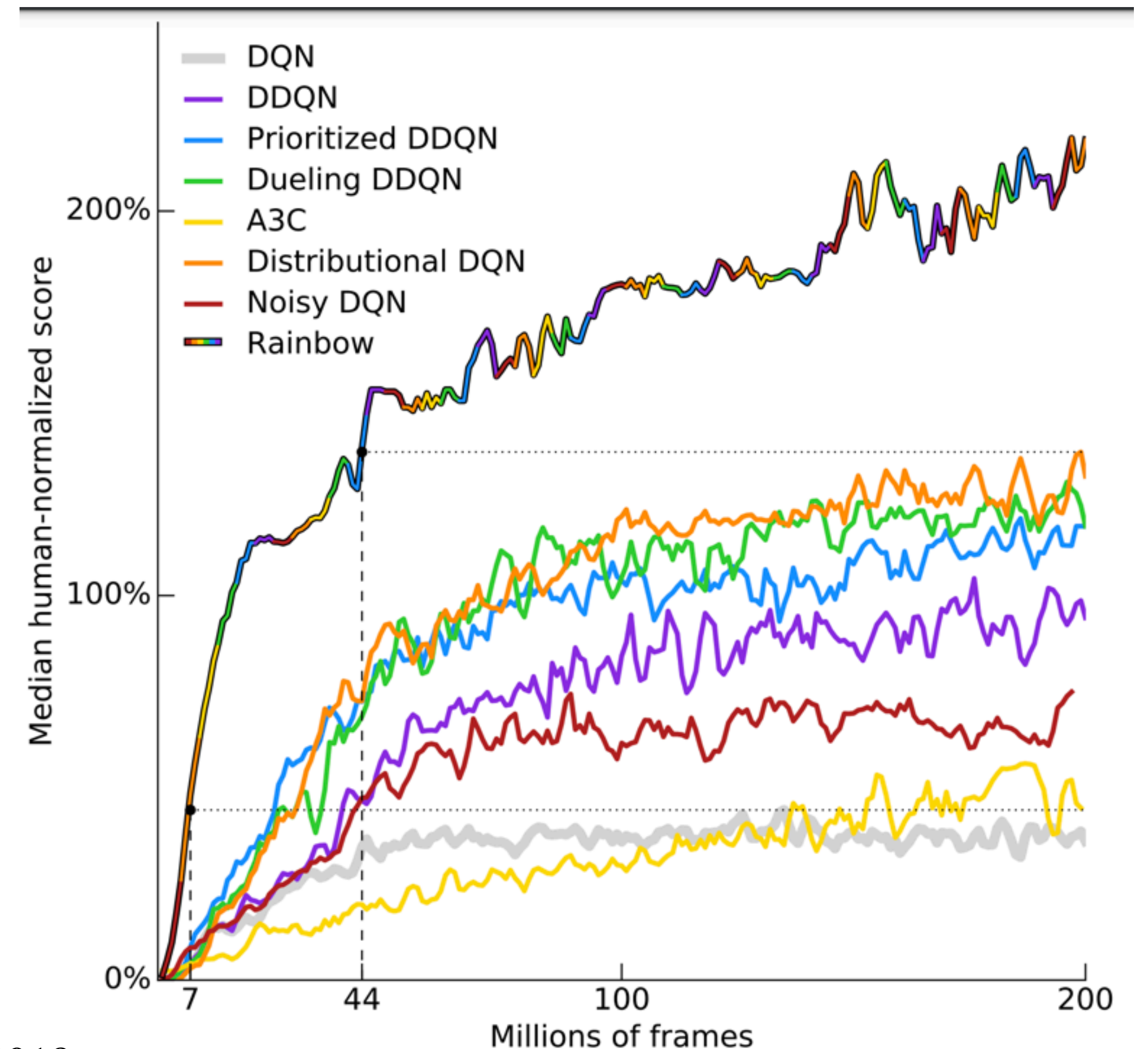
        Every  $C$  steps reset  $\hat{Q} = Q$

**End For**

**End For**

# Looking Forward

- DQN launched a surge of interest in deep reinforcement learning that has led to many exciting new applications and RL developments.
- DQN is widely used in practice though many improvements have been made.



Rainbow: Combining Improvements in Deep Reinforcement Learning. Hessel et al. 2018.

<https://www.deepmind.com/blog/agent57-outperforming-the-human-atari-benchmark>

# Yidong's Presentation

- Slides

# Summary

- Deep reinforcement learning is not just deep learning + RL.
  - It's often deep learning + RL + new techniques and tricks for stability.
- This week focused on deep value-based RL.
  - Deep networks can be used for model-based RL.
  - Deep networks can represent policies in policy gradient RL.

# Action Items

- Complete literature review.
- Read Chapter 13 (skip 13.5).