

# Lecture 8b: Partially Observable RL, DRQN

## CS885 Reinforcement Learning

2022-10-07

Complementary readings:

Hausknecht, M., & Stone, P. Deep recurrent Q-learning for partially observable MDPs. In 2015 AAAI fall symposium series.

Pascal Poupart

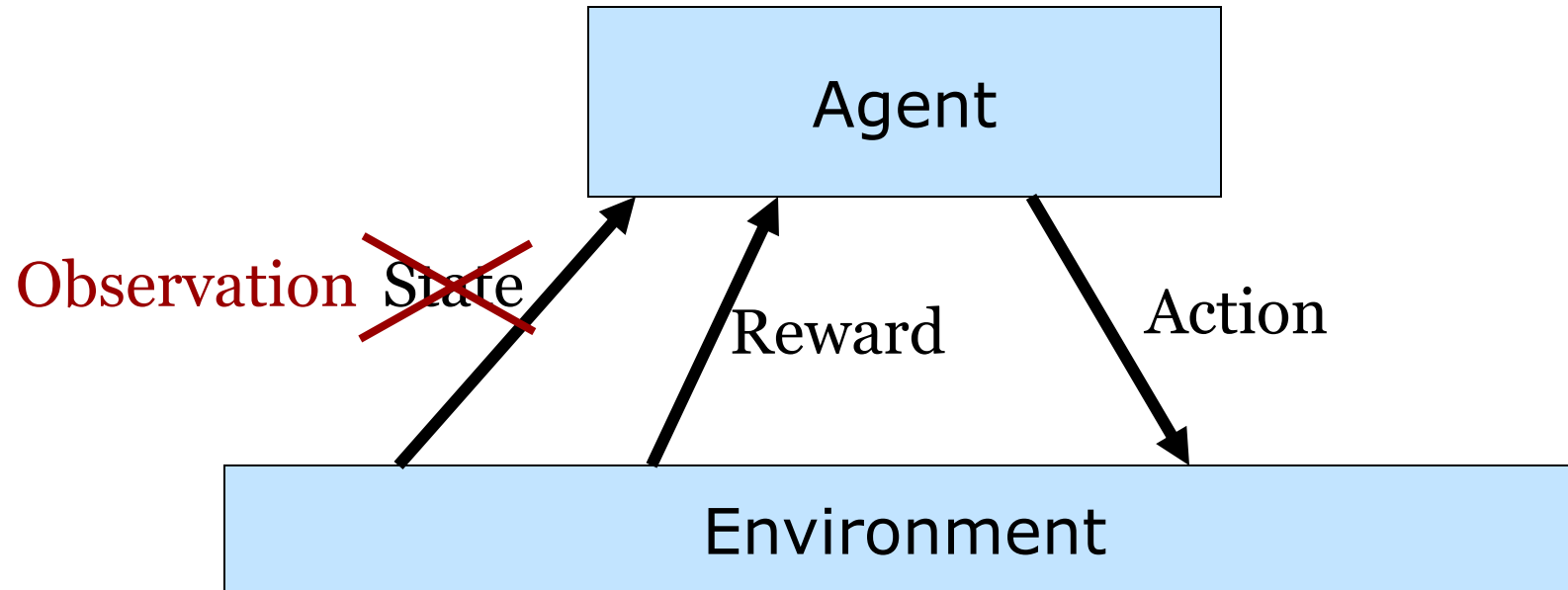
David R. Cheriton School of Computer Science



# Outline

- Partially Observable Markov Decision Processes
- Hidden Markov Models
- Recurrent neural networks
  - Long short term memory (LSTM) networks
- Deep recurrent Q-networks

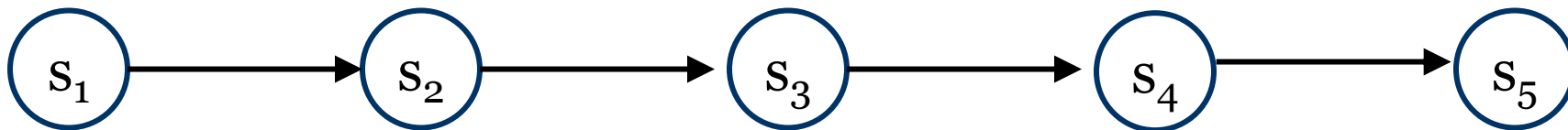
# Reinforcement Learning Problem



**Goal:** Learn to choose actions that maximize rewards

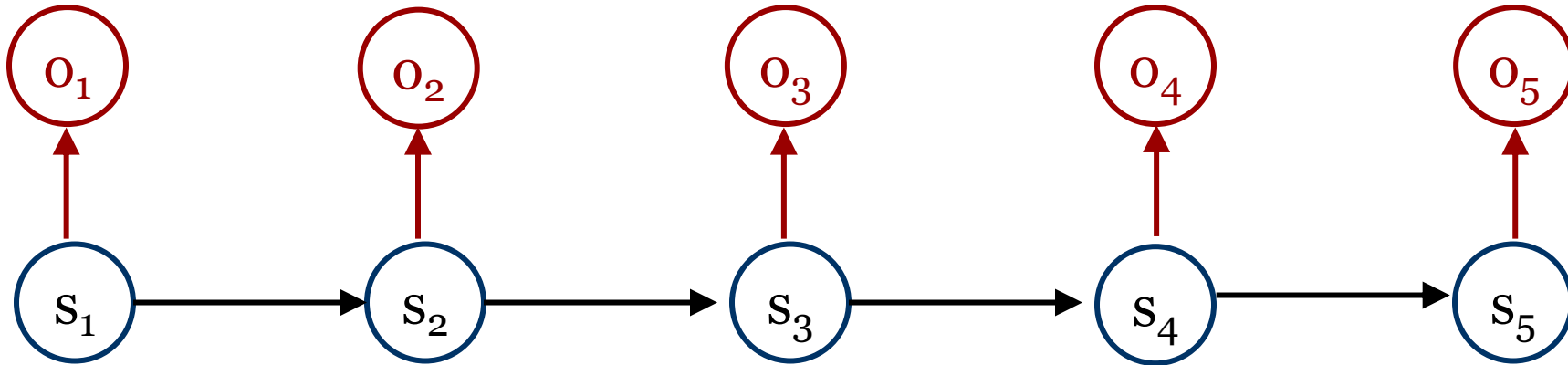
# Markov Process

- Assumptions:
  - (first-order) Markovian:  $\Pr(s_t | s_{t-1}, \dots, s_0) = \Pr(s_t | s_{t-1})$
  - Stationary:  $\Pr(s_t | s_{t-1}) = \Pr(s_{t+1} | s_t) \forall t$



# Hidden Markov Model

- Assumptions:
  - (first-order) Markovian:  $\Pr(s_t | s_{t-1}, \dots, s_0) = \Pr(s_t | s_{t-1})$
  - Stationary:  $\Pr(s_t | s_{t-1}) = \Pr(s_{t+1} | s_t) \forall t$   
 $\Pr(o_t | s_t) = \Pr(o_{t+1} | s_{t+1}) \forall t$

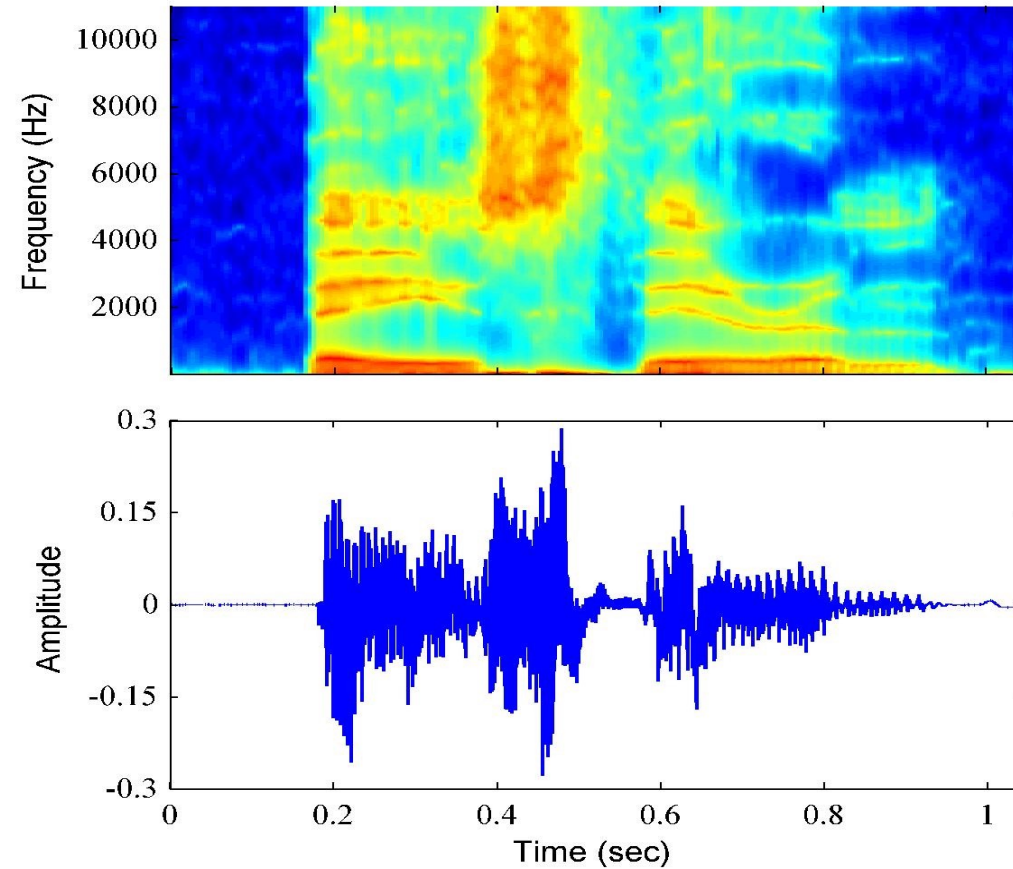


# Speech Recognition

Compute:

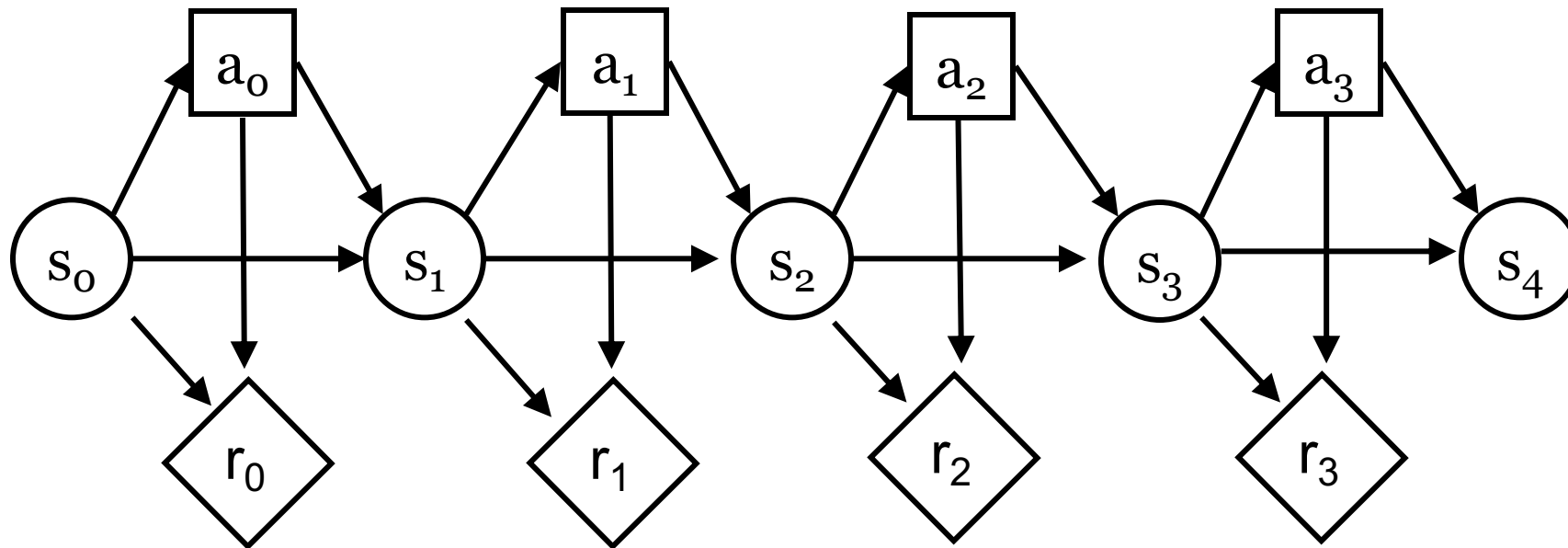
$$\Pr(\text{word}|\text{speech})$$

$$\Pr(s_t|o_t, o_{t-1}, \dots, o_1)$$



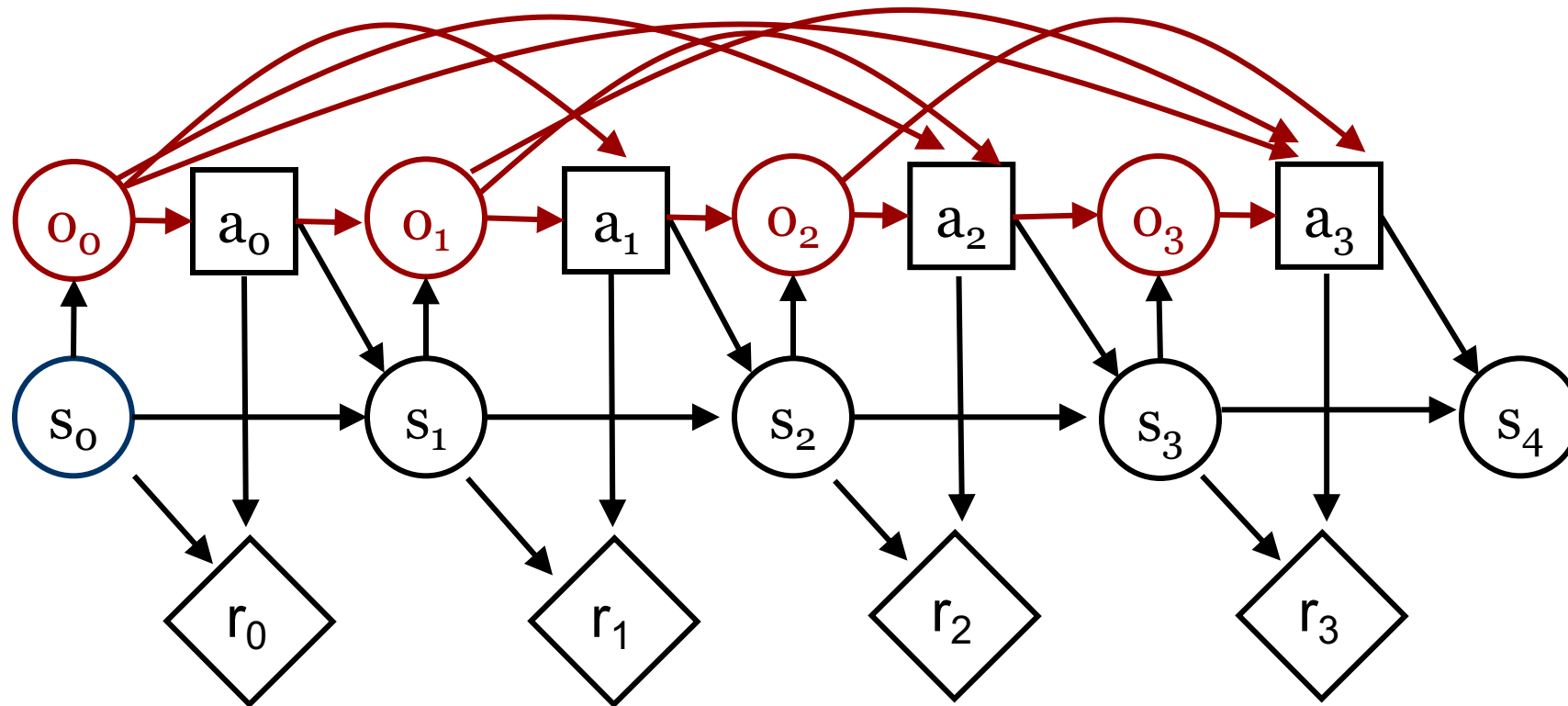
| b |    ey |    z |    th |    ih |    er |    em |  
|        Bayes' |        Theorem |

# (Fully Observable) Markov Decision Process (MDP)



# Partially Observable Markov Decision Process (POMDP)

- MDP augmented with observations





# Partially Observable RL

- Definition

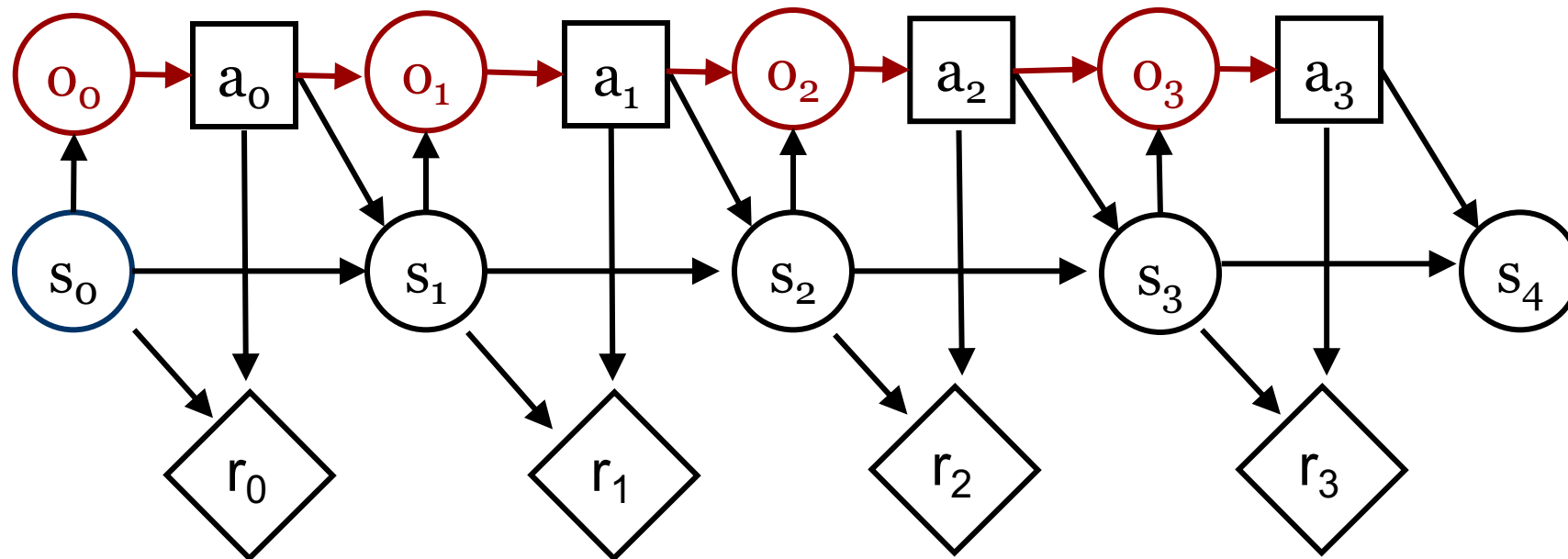
- States:  $s \in S$
- **Observations:  $o \in O$**
- Actions:  $a \in A$
- Rewards:  $r \in \mathbb{R}$
- Transition model:  $\Pr(s_t | s_{t-1}, a_{t-1})$
- **Observation model:  $\Pr(o_t | a_{t-1}, s_t)$**
- Reward model:  $\Pr(r_t | s_t, a_t)$
- Discount factor:  $0 \leq \gamma \leq 1$ . (discounted:  $\gamma < 1$ , undiscounted:  $\gamma = 1$ )
- Horizon (i.e., # of time steps):  $h$  (Finite horizon:  $h \in \mathbb{N}$ , infinite horizon:  $h = \infty$ )

} **unknown model**

- Goal: **find optimal policy  $\pi^*$  such that  $\pi^* = \operatorname{argmax}_{\pi} \sum_{t=0}^h \gamma^t E_{\pi}[r_t]$**

# Simple Heuristic

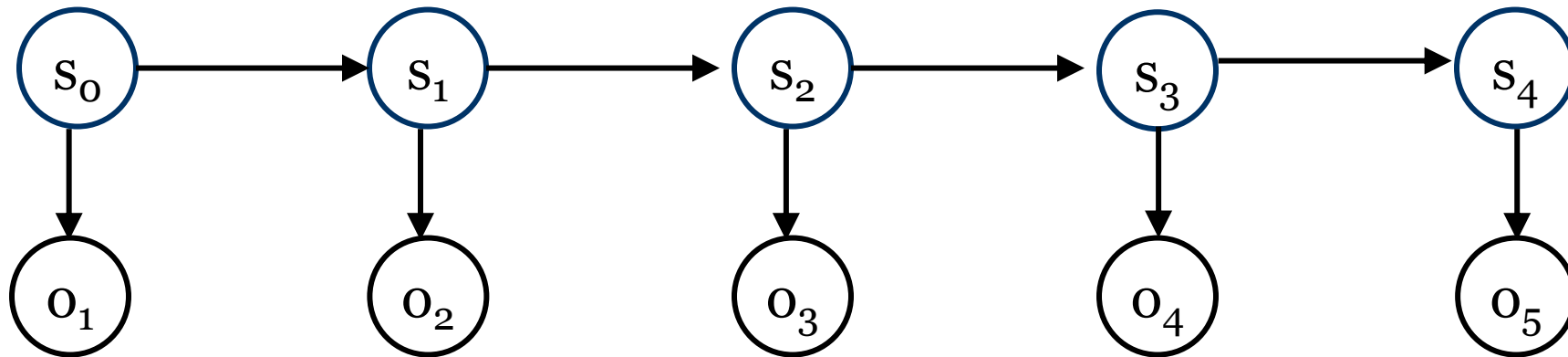
- Approximate  $s_t$  by  $o_t$  (or finite window of previous obs:  $o_{t-k}, o_{t-k+1}, \dots, o_t$ )
- Use favorite RL algo on observations instead of states



# Belief Monitoring

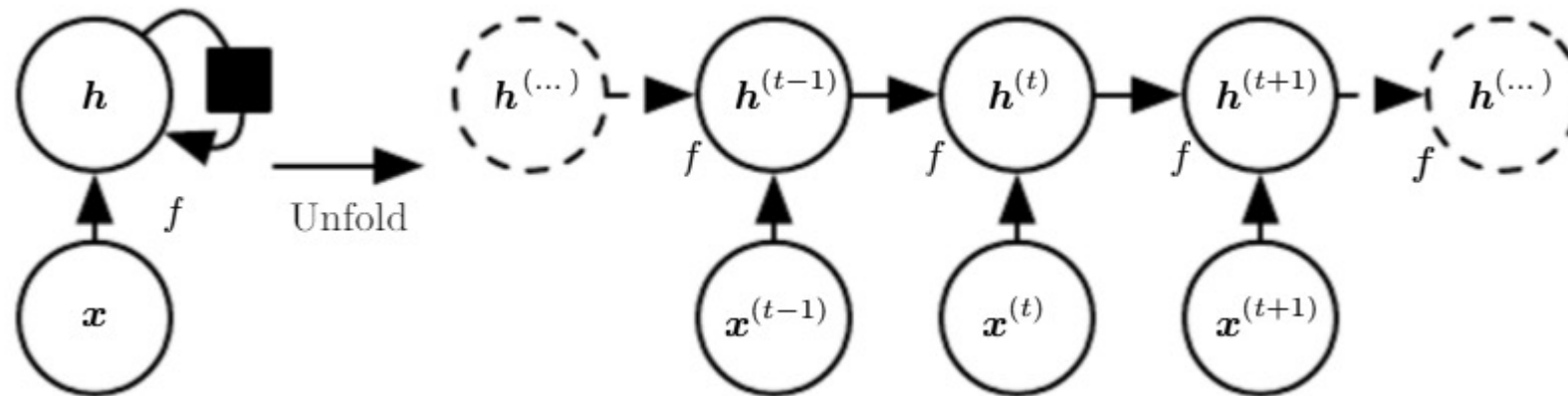
- Hidden Markov model
  - Initial state distribution:  $\Pr(s_0)$
  - Transition probabilities:  $\Pr(s_{t+1}|s_t)$
  - Observation probabilities:  $\Pr(o_t|s_t)$
- Belief monitoring

$$\Pr(s_t|o_{1..t}) \propto \Pr(o_t|s_t) \sum_{s_{t-1}} \Pr(s_t|s_{t-1}) \Pr(s_{t-1}|o_{1..t-1})$$



# Recurrent Neural Network (RNN)

- In RNNs, outputs can be fed back to the network as inputs, creating a recurrent structure
- HMMs can be simulated and generalized by RNNs
- RNNs can be used for belief monitoring  
 $\mathbf{x}_t$ : vector of observations       $\mathbf{h}_t$ : belief state

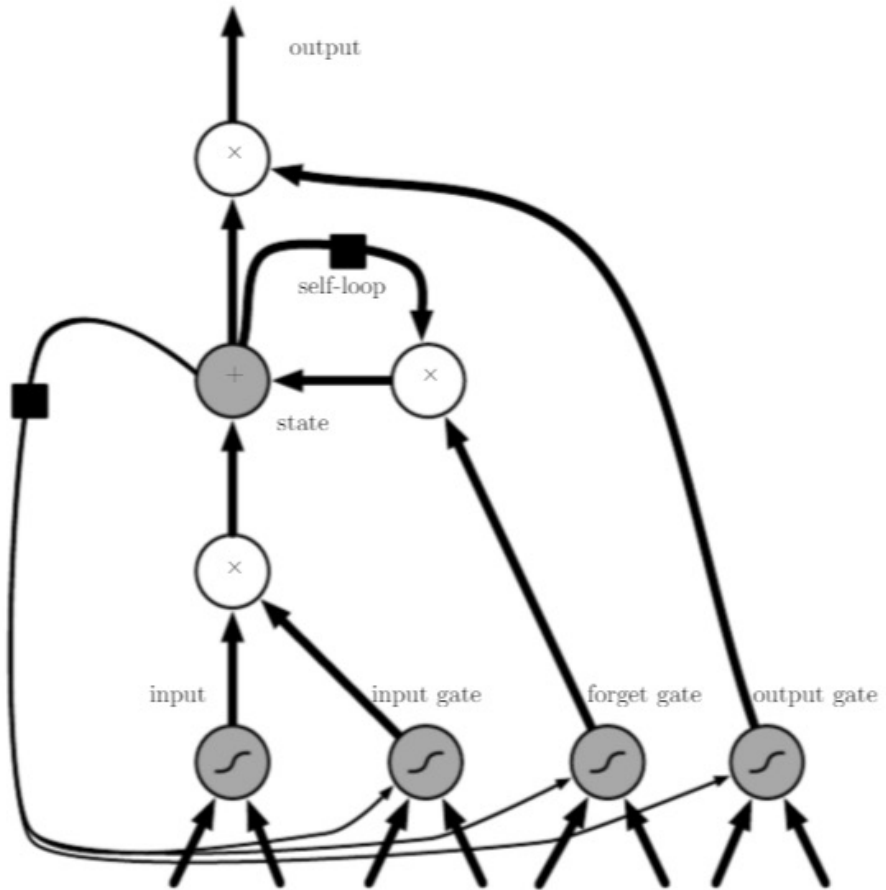


# Training

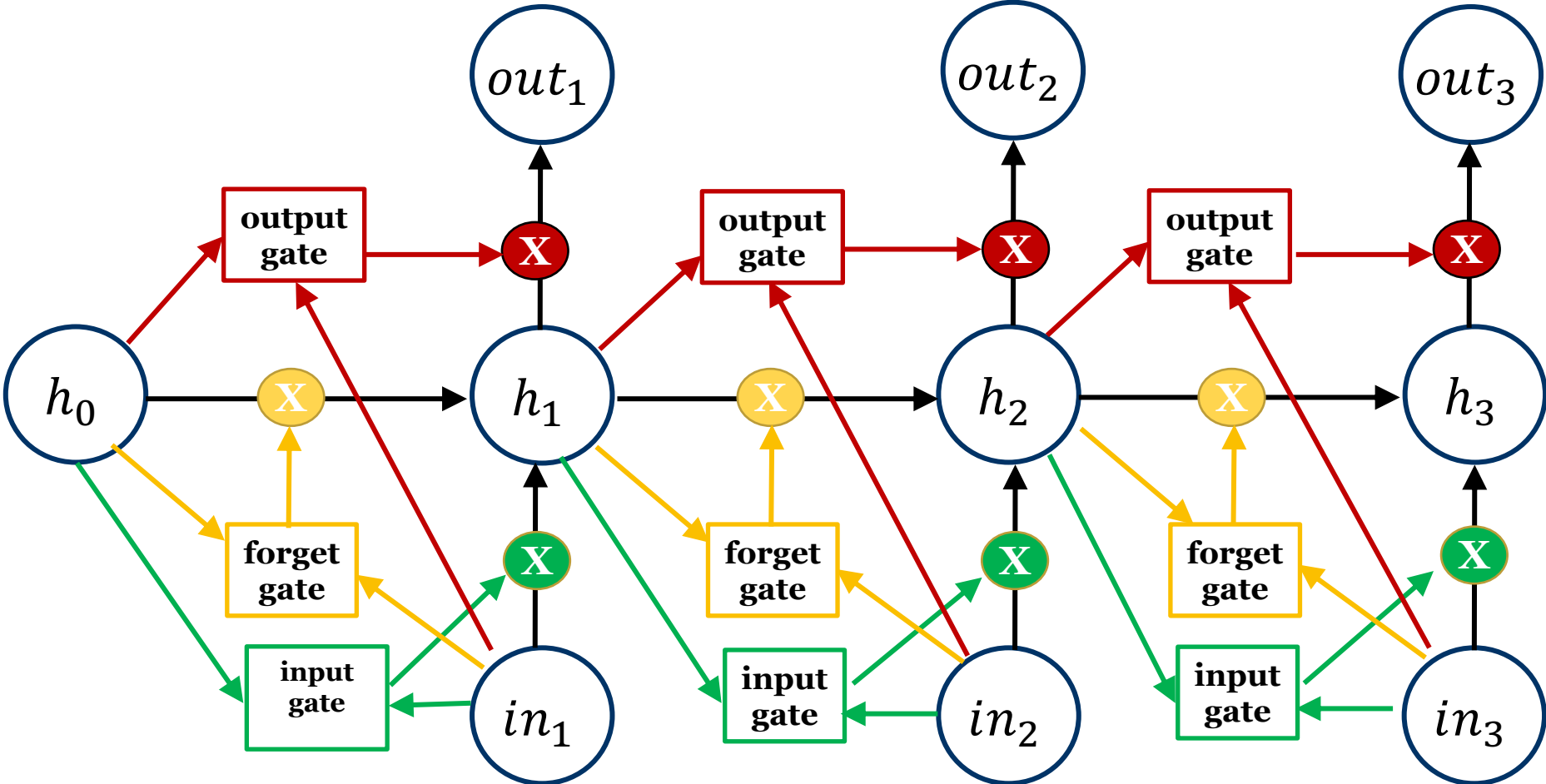
- Recurrent neural networks are trained by backpropagation on the unrolled network
  - E.g., backpropagation through time
- Weight sharing:
  - Combine gradients of shared weights into a single gradient
- Challenges:
  - Gradient vanishing (and explosion)
  - Long range memory
  - Prediction drift

# Long Short Term Memory (LSTM)

- Special gated structure to control memorization and forgetting in RNNs
- Mitigate gradient vanishing
- Facilitate long term memory

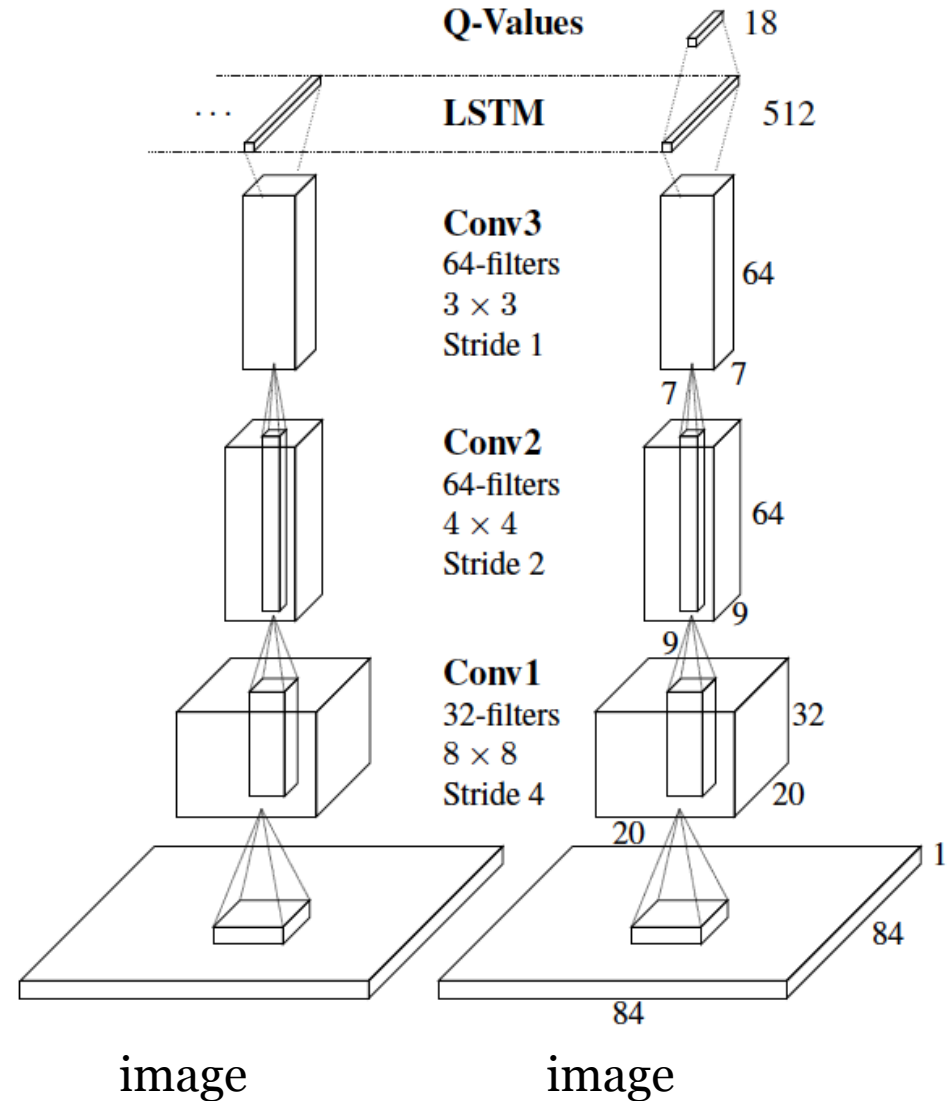
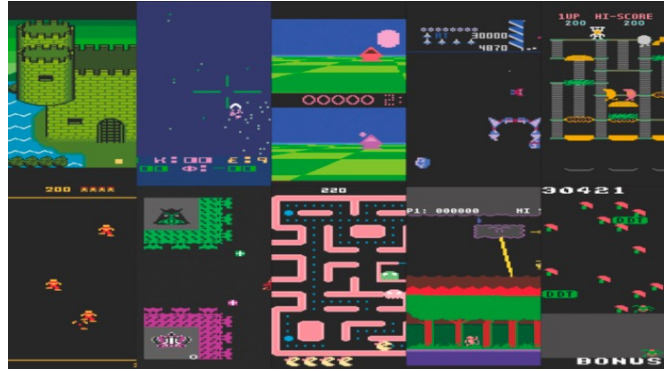


# Unrolled long short term memory



# Deep Recurrent Q-Network

- Hausknecht and Stone (2016)
  - Atari games
- Transition model
  - LSTM network
- Observation model
  - Convolutional network





# Deep Recurrent Q-Network

Initialize weights  $\mathbf{w}$  and  $\bar{\mathbf{w}}$  at random

Observe current state  $s$

Loop

Execute policy for entire episode

Add episode  $(o_1, a_1, o_2, a_2, o_3, a_3, \dots, o_T, a_T)$  to experience buffer

Sample episode from buffer

Initialize  $h_0$

For  $t = 1$  till the end of the episode do

$$\frac{\partial \text{Err}}{\partial \mathbf{w}} = \left[ Q_{\mathbf{w}}(\text{RNN}_{\mathbf{w}}(\hat{o}_{1..t}), \hat{a}_t) - \hat{r} - \gamma \max_{\hat{a}_{t+1}} Q_{\bar{\mathbf{w}}}(\text{RNN}_{\bar{\mathbf{w}}}(\hat{o}_{1..t+1}), \hat{a}_{t+1}) \right] \frac{\partial Q_{\mathbf{w}}(\text{RNN}_{\mathbf{w}}(\hat{o}_{1..t}), \hat{a}_t)}{\partial \mathbf{w}}$$

Update weights:  $\mathbf{w} \leftarrow \mathbf{w} - \alpha \frac{\partial \text{Err}}{\partial \mathbf{w}}$

Every  $c$  steps, update target:  $\bar{\mathbf{w}} \leftarrow \mathbf{w}$

# Results

Game	DRQN $\pm std$		DQN $\pm std$	
		Ours		Mnih et al.
Asteroids	1020 ( $\pm 312$ )	1070 ( $\pm 345$ )		1629 ( $\pm 542$ )
Beam Rider	3269 ( $\pm 1167$ )	<b>6923</b> ( $\pm 1027$ )		6846 ( $\pm 1619$ )
Bowling	62 ( $\pm 5.9$ )	72 ( $\pm 11$ )		42 ( $\pm 88$ )
Centipede	3534 ( $\pm 1601$ )	3653 ( $\pm 1903$ )		8309 ( $\pm 5237$ )
Chopper Cmd	2070 ( $\pm 875$ )	1460 ( $\pm 976$ )		6687 ( $\pm 2916$ )
Double Dunk	<b>-2</b> ( $\pm 7.8$ )	-10 ( $\pm 3.5$ )		-18.1 ( $\pm 2.6$ )
Frostbite	<b>2875</b> ( $\pm 535$ )	519 ( $\pm 363$ )		328.3 ( $\pm 250.5$ )
Ice Hockey	-4.4 ( $\pm 1.6$ )	-3.5 ( $\pm 3.5$ )		-1.6 ( $\pm 2.5$ )
Ms. Pacman	2048 ( $\pm 653$ )	2363 ( $\pm 735$ )		2311 ( $\pm 525$ )

Table 1: On standard Atari games, DRQN performance parallels DQN, excelling in the games of Frostbite and Double Dunk, but struggling on Beam Rider. Bolded font indicates statistical significance between DRQN and our DQN.<sup>5</sup>

Flickering games  
(missing observations)

