A detailed digital illustration of a desert canyon, likely Arrakis from the Dune series. The scene is characterized by towering, layered rock formations in shades of brown and orange. Sunlight filters through the crevices, creating dramatic shadows and highlights. In the foreground, a lone figure in a dark, hooded robe stands on a rocky ledge, looking out over the vast, layered landscape. The overall atmosphere is one of isolation and grandeur.

“One learns from books and example only that certain things can be done. Actual learning requires that you do those things.”

Frank Herbert, *Children of Dune*

CMPUT 365

Introduction to RL

Coursera Reminder

You **should be enrolled in the private session** we created in Coursera for CMPUT 365.

I **cannot** use marks from the public repository for your course marks.

You **need to check, every time**, if you are in the private session and if you are submitting quizzes and assignments to the private section.

The deadlines in the public session **do not align** with the deadlines in Coursera.

If you have any questions or concerns, **talk with the TAs** or email us `cmput365@ualberta.ca`.

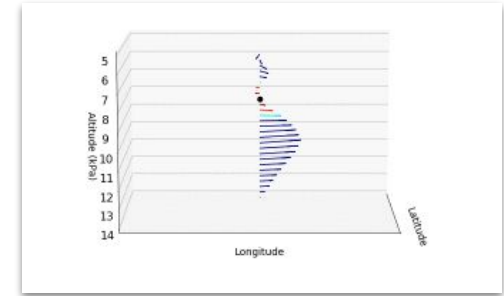
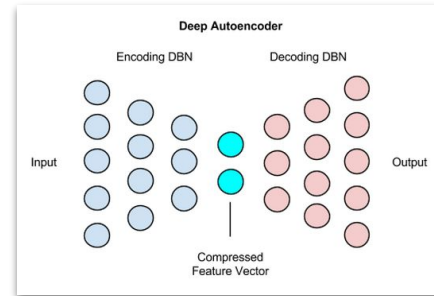
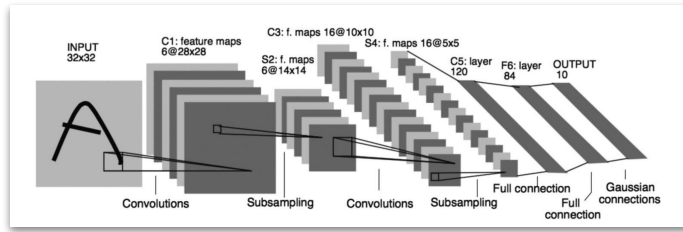
Reminders and Notes

- The practice quiz and programming assignment for “Control with FA” are due on Wednesday (only 2 more to go!).
- Exam viewing for Midterm 2:
 - ~~Tuesday (March 17): 13:00 – 15:00 @ UCOMM 3-480~~
 - ~~Wednesday (March 18): 15:00 – 17:00 @ UCOMM 3-162~~
 - Friday (March 20): 15:00 - 17:00 @ UCOMM-3-162
 - Monday (March 23): 15:00 - 17:00 @ UCOMM-3-162

Please, interrupt me at any time!



Last Class: Neural Networks and Feature/Input Construction



<https://wiki.pathmind.com/deep-autoencoder>

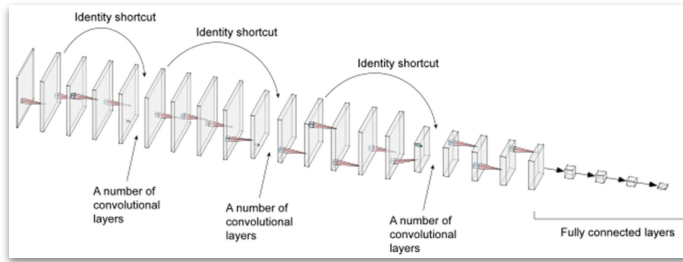
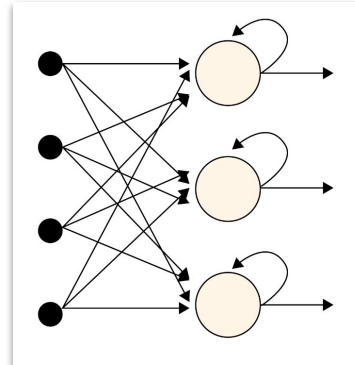


Image by Yu, Miao, and Wang (2022)



<https://www.scaler.com/topics/deep-learning/rnn/>

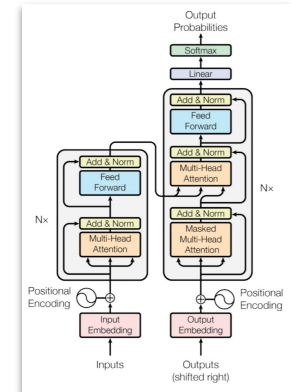
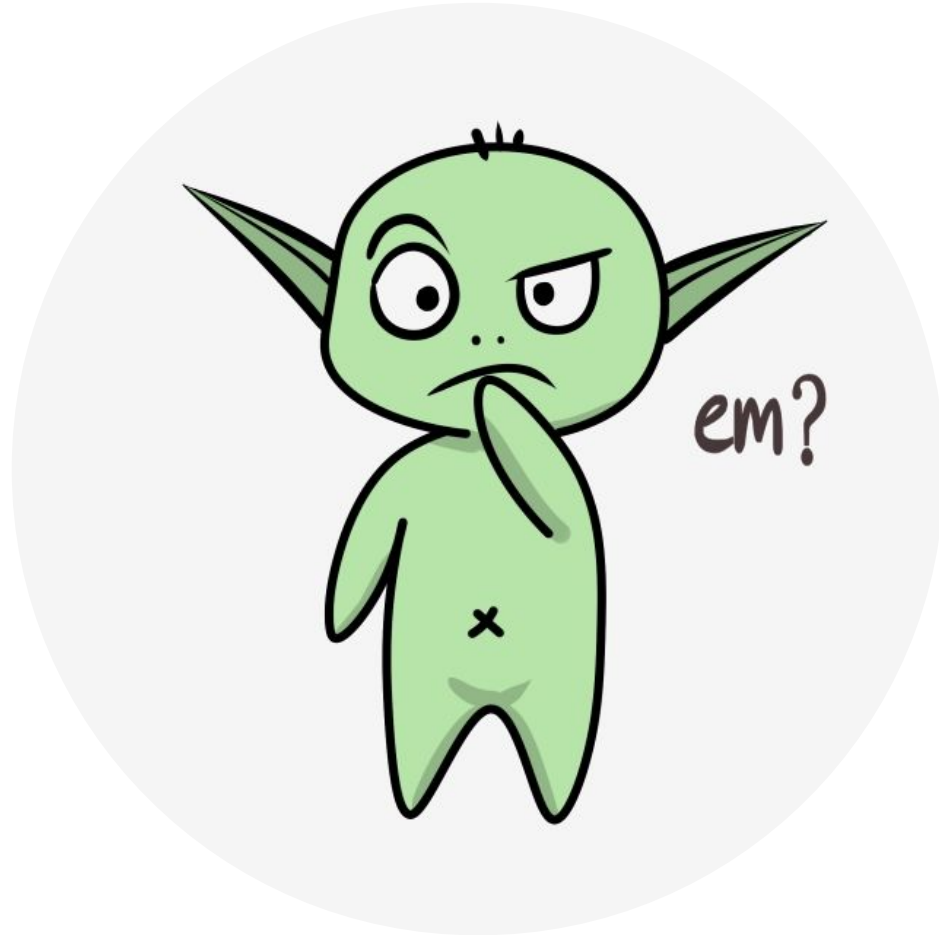


Image by Vaswani et al. (2017)



Episodic Semi-gradient Control

- We need to approximate the action-value function now, $\hat{q} \approx q_\pi$, that is represented as a parameterized function form with weight vector \mathbf{w} .
- Before (until last class): $S_t \mapsto U_t$.
Now: $S_t, A_t \mapsto U_t$.

Episodic Semi-gradient Control

- We need to approximate the action-value function now, $\hat{q} \approx q_\pi$, that is represented as a parameterized function form with weight vector \mathbf{w} .
- Before (until last class): $S_t \mapsto U_t$.
Now: $S_t, A_t \mapsto U_t$.

- Action-value prediction:

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \left[U_t - \hat{q}(S_t, A_t, \mathbf{w}_t) \right] \nabla \hat{q}(S_t, A_t, \mathbf{w}_t)$$

Episodic Semi-gradient Control

- We need to approximate the action-value function now, $\hat{q} \approx q_\pi$, that is represented as a parameterized function form with weight vector \mathbf{w} .
- Before (until last class): $S_t \mapsto U_t$.
Now: $S_t, A_t \mapsto U_t$.

- Action-value prediction:

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \left[U_t - \hat{q}(S_t, A_t, \mathbf{w}_t) \right] \nabla \hat{q}(S_t, A_t, \mathbf{w}_t)$$

- Episodic semi-gradient one-step *Sarsa*:

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \left[R_{t+1} + \gamma \hat{q}(S_{t+1}, A_{t+1}, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t) \right] \nabla \hat{q}(S_t, A_t, \mathbf{w}_t)$$

Episodic Semi-gradient Sarsa

Episodic Semi-gradient Sarsa for Estimating $\hat{q} \approx q_*$

Input: a differentiable action-value function parameterization $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \rightarrow \mathbb{R}$

Algorithm parameters: step size $\alpha > 0$, small $\varepsilon > 0$

Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)

Loop for each episode:

$S, A \leftarrow$ initial state and action of episode (e.g., ε -greedy)

Loop for each step of episode:

Take action A , observe R, S'

If S' is terminal:

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha [R - \hat{q}(S, A, \mathbf{w})] \nabla \hat{q}(S, A, \mathbf{w})$$

Go to next episode

Choose A' as a function of $\hat{q}(S', \cdot, \mathbf{w})$ (e.g., ε -greedy)

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha [R + \gamma \hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w})] \nabla \hat{q}(S, A, \mathbf{w})$$

$S \leftarrow S'$

$A \leftarrow A'$



This really works!

State of the Art Control of Atari Games Using Shallow Reinforcement Learning

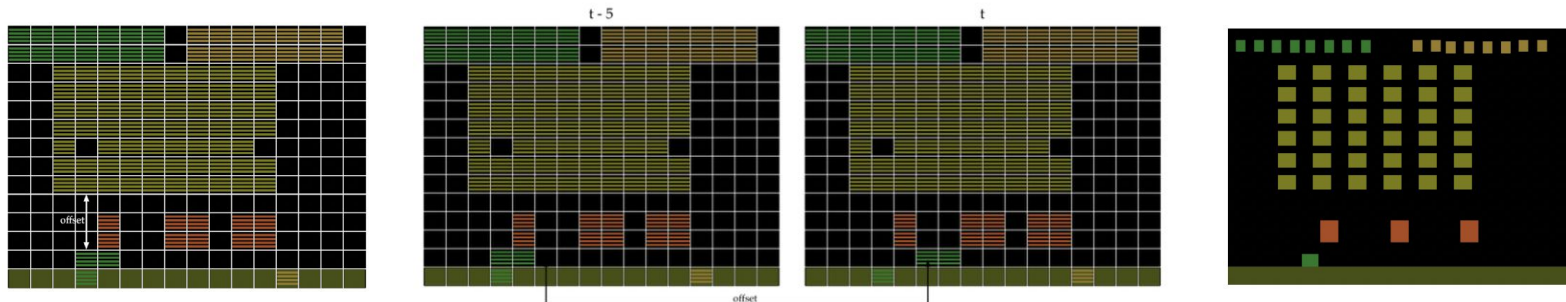
Yitao Liang[†], Marlos C. Machado[‡], Erik Talvitie[†], and Michael Bowling[‡]

[†]Franklin & Marshall College
Lancaster, PA, USA

[‡]University of Alberta
Edmonton, AB, Canada

{yliang, erik.talvitie}@fandm.edu

{machado, mbowling}@ualberta.ca





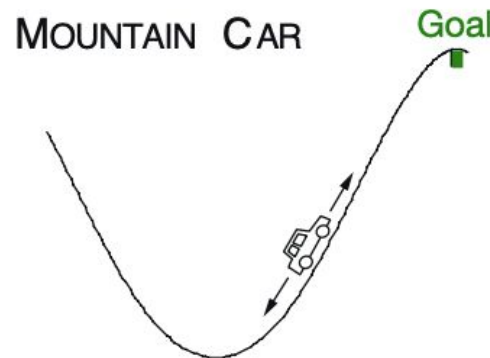
Example: Mountain Car Task

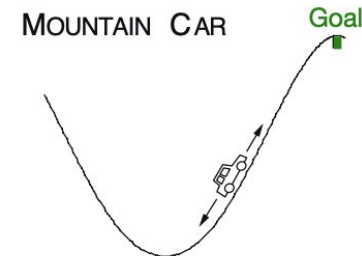
- Observations: (x, \dot{x})
- Actions:
 - Full throttle forward: +1
 - Full throttle reverse: -1
 - Zero throttle: 0
- Rewards: -1 at every time step, until end of episode.
- Dynamics:

$$x_{t+1} \doteq \text{bound}[x_t + \dot{x}_{t+1}]$$

$$\dot{x}_{t+1} \doteq \text{bound}[\dot{x}_t + 0.001A_t - 0.0025 \cos(3x_t)]$$

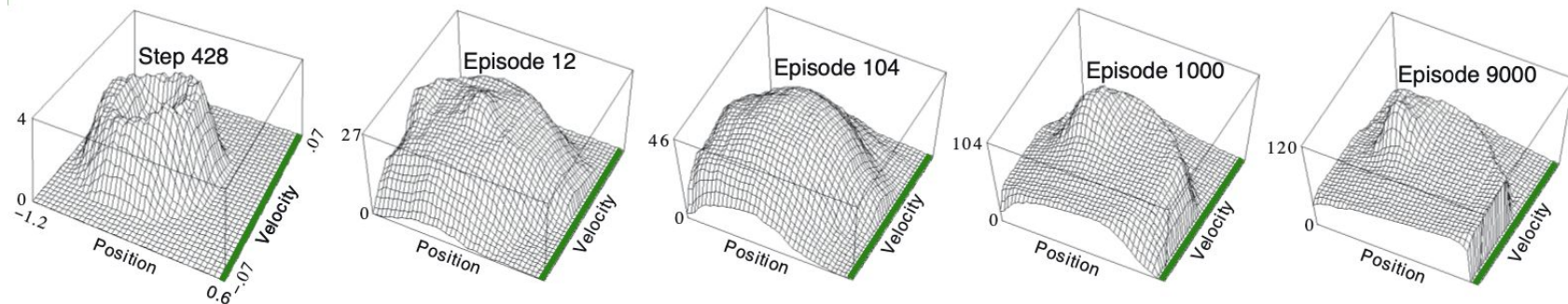
$$-1.2 \leq x_{t+1} \leq 0.5 \text{ and } -0.07 \leq \dot{x}_{t+1} \leq 0.07$$

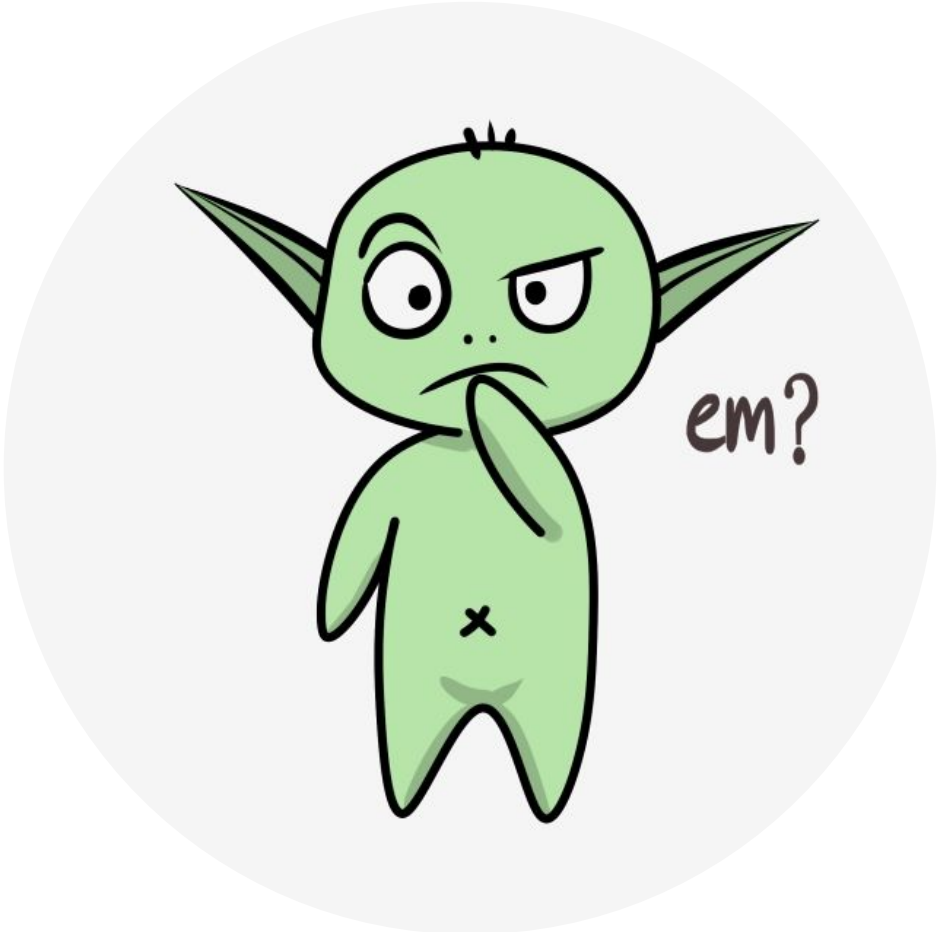




“Solution”: Mountain Car Task

- Feature representation:
 - Grid-tilings with 8 tilings and asymmetrical offsets.
 - $\hat{q}(s, a, \mathbf{w}) \doteq \mathbf{w}^\top \mathbf{x}(s, a) = \sum_{i=1}^d w_i \cdot x_i(s, a)$
- Sarsa
 - Weights initialized at zero. Effectively optimistic initialization.





Journal of Artificial Intelligence Research 47 (2013) 253–279

Submitted 02/13; published 06/13

The Arcade Learning Environment: An Evaluation Platform for General Agents

Marc G. Bellemare

University of Alberta, Edmonton, Alberta, Canada

MG17@CS.UALBERTA.CA

Yavar Naddaf

*Empirical Results Inc., Vancouver,
British Columbia, Canada*

YAVAR@EMPIRICALRESULTS.CA

Joel Veness

VENESS@CS.UALBERTA.CA

Michael Bowling

University of Alberta, Edmonton, Alberta, Canada

BOWLING@CS.UALBERTA.CA

Abstract

In this article we introduce the Arcade Learning Environment (ALE): both a challenge problem and a platform and methodology for evaluating the development of general, domain-independent AI technology. ALE provides an interface to hundreds of Atari 2600 game environments, each one different, interesting, and designed to be a challenge for human players. ALE presents significant research challenges for reinforcement learning, model learning, model-based planning, imitation learning, transfer learning, and intrinsic motivation. Most importantly, it provides a rigorous testbed for evaluating and comparing approaches to these problems. We illustrate the promise of ALE by developing and benchmarking domain-independent agents designed using well-established AI techniques for both reinforcement learning and planning. In doing so, we also propose an evaluation

Arcade Learning Environment

Over 50 Domains in 8 Minutes 23 Seconds

Deep Q-Network (and Deep RL)

[Mnih et al., 2013, 2015]

Playing Atari with Deep Reinforcement Learning

Volodymyr Mnih Koray Kavukcuoglu David Silver Alex Graves Ioannis Antonoglou

Daan Wierstra Martin Riedmiller

DeepMind Technologies

{vlad,koray,david,alex.graves,ioannis,daan,martin.riedmiller} @ deepmind.com

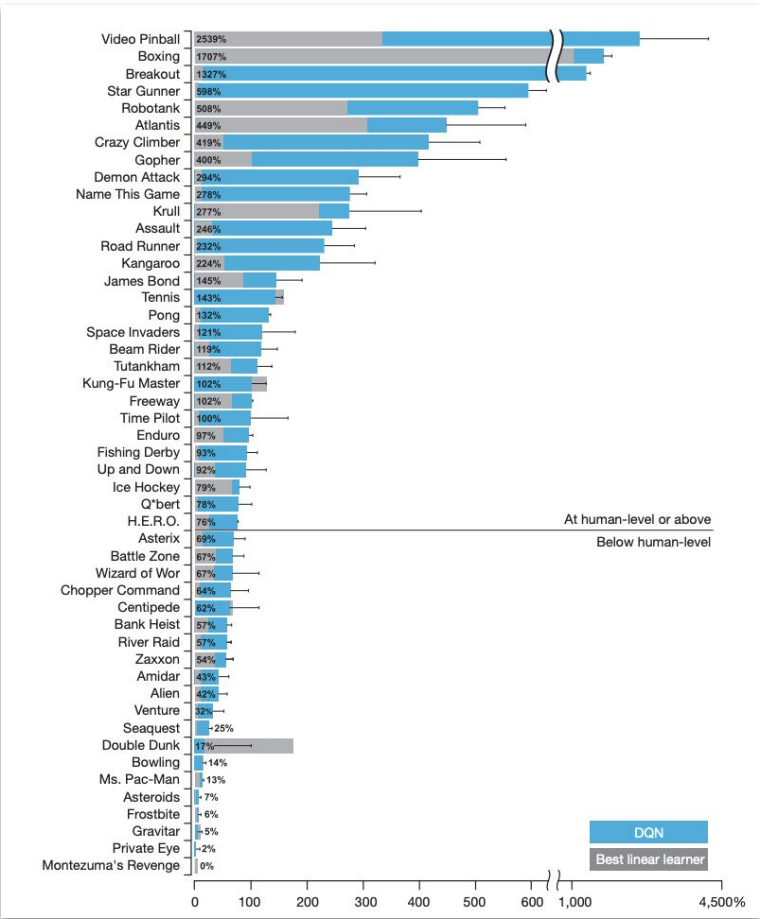
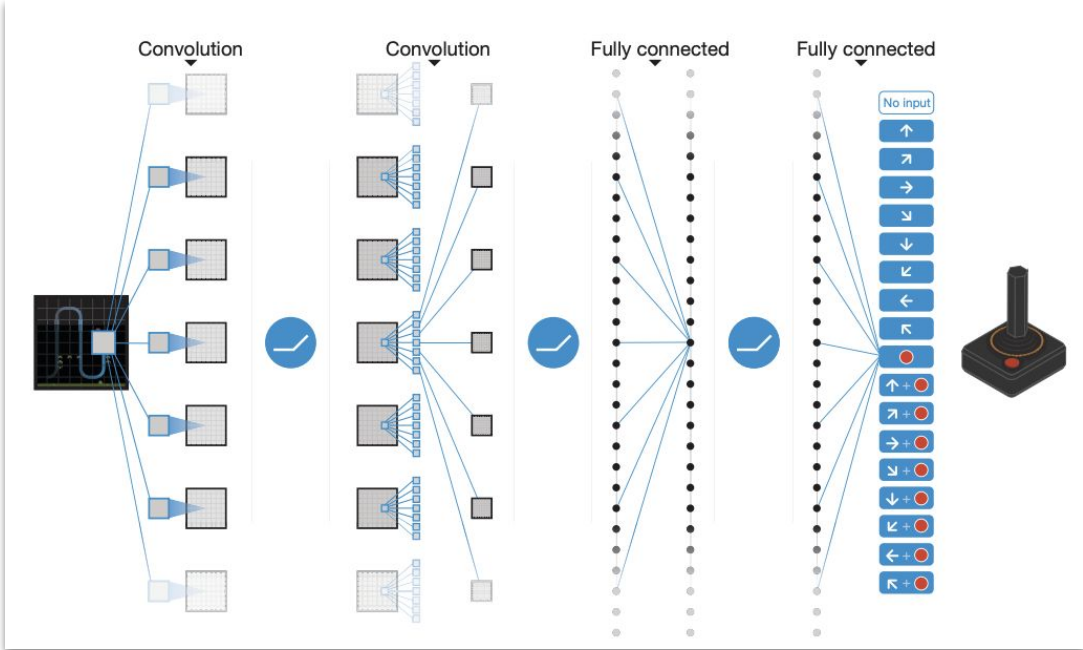
Abstract

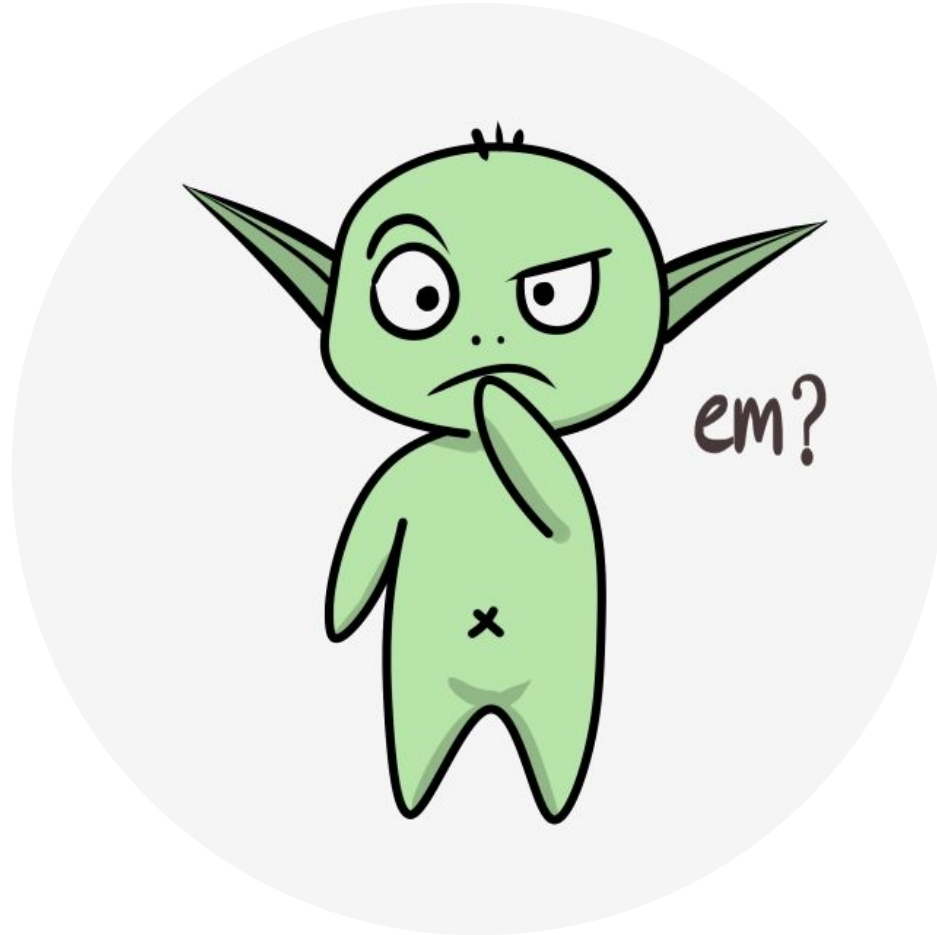
Dec 2013



Deep Q-Network (and Deep RL)

[Mnih et al., 2013, 2015]





Deep Q-Network (DQN)

[Mnih et al., 2013, 2015]

$$\mathcal{L}^{\text{DQN}} = \mathbb{E}_{(s,a,r,s') \sim U(\mathcal{D})} \left[\left(R_{t+1} + \gamma \max_{a' \in \mathcal{A}} Q(S_{t+1}, a'; \boldsymbol{\theta}^-) - Q(S_t, A_t, \boldsymbol{\theta}_t) \right)^2 \right]$$

Stacked frames

-1, 0, +1 rewards

Clipped error term

Experience replay buffer (Lin, 1993)

Original size: 1M frames

Target network

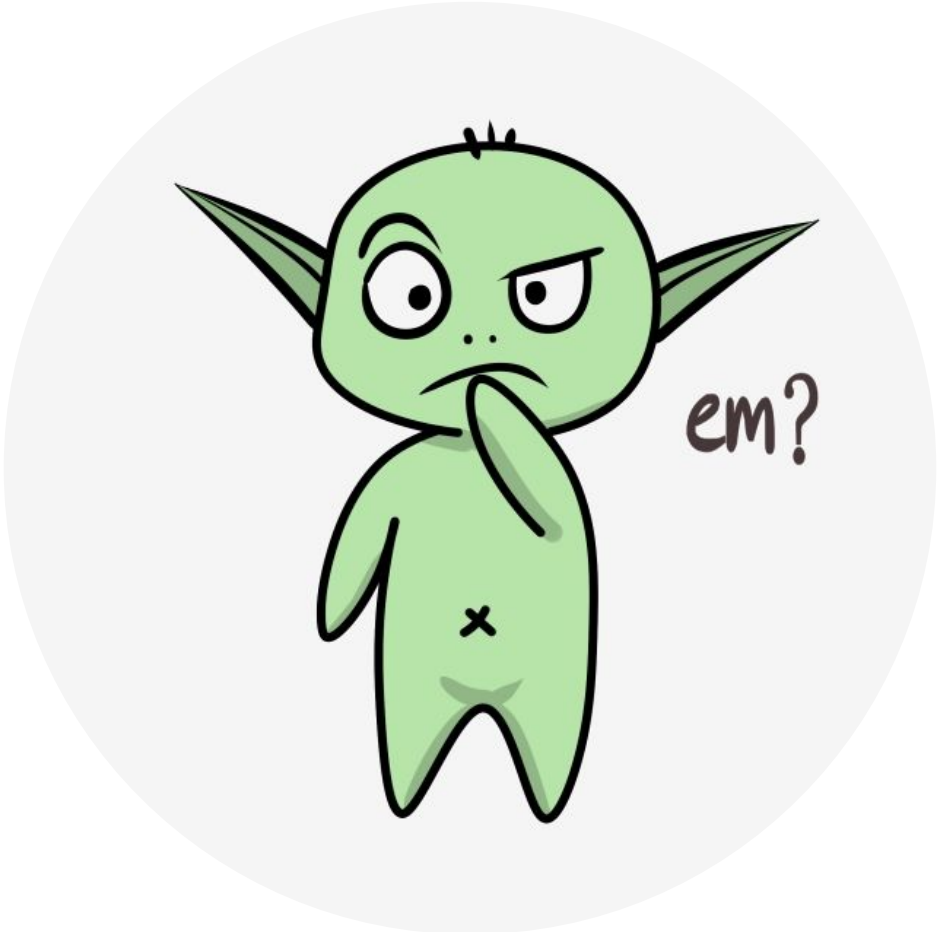
Original update frequency: 10k

ϵ decay

Originally, from 1.0 to 0.1 over 1M frames

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha \left[R_{t+1} + \gamma \max_{a' \in \mathcal{A}} Q(S_{t+1}, a'; \boldsymbol{\theta}^-) - Q(S_t, A_t, \boldsymbol{\theta}_t) \right] \nabla_{\boldsymbol{\theta}_t} Q(S_t, A_t; \boldsymbol{\theta}_t)$$

RMSProp



Deep Q-Network (DQN)

[Mnih et al., 2013, 2015]

| Game | Random Play | Best Linear Learner | Contingency (SARSA) | Human | DQN (\pm std) | Normalized DQN (% Human) |
|-------------|-------------|---------------------|---------------------|-------|---------------------|--------------------------|
| Alien | 227.8 | 939.2 | 103.2 | 6875 | 3069 (\pm 1093) | 42.7% |
| Amidar | 5.8 | 103.4 | 183.6 | 1676 | 739.5 (\pm 3024) | 43.9% |
| Assault | 222.4 | 628 | 537 | 1496 | 3359(\pm 775) | 246.2% |
| Asterix | 210 | 987.3 | 1332 | 8503 | 6012 (\pm 1744) | 70.0% |
| Asteroids | 719.1 | 907.3 | 89 | 13157 | 1629 (\pm 542) | 7.3% |
| Atlantis | 12850 | 62687 | 852.9 | 29028 | 85641(\pm 17600) | 449.9% |
| Bank Heist | 14.2 | 190.8 | 67.4 | 734.4 | 429.7 (\pm 650) | 57.7% |
| Battle Zone | 2360 | 15820 | 16.2 | 37800 | 26300 (\pm 7725) | 67.6% |
| Beam Rider | 363.9 | 929.4 | 1743 | 5775 | 6846 (\pm 1619) | 119.8% |
| Bowling | 23.1 | 43.9 | 36.4 | 154.8 | 42.4 (\pm 88) | 14.7% |

Deep Q-Network (DQN)

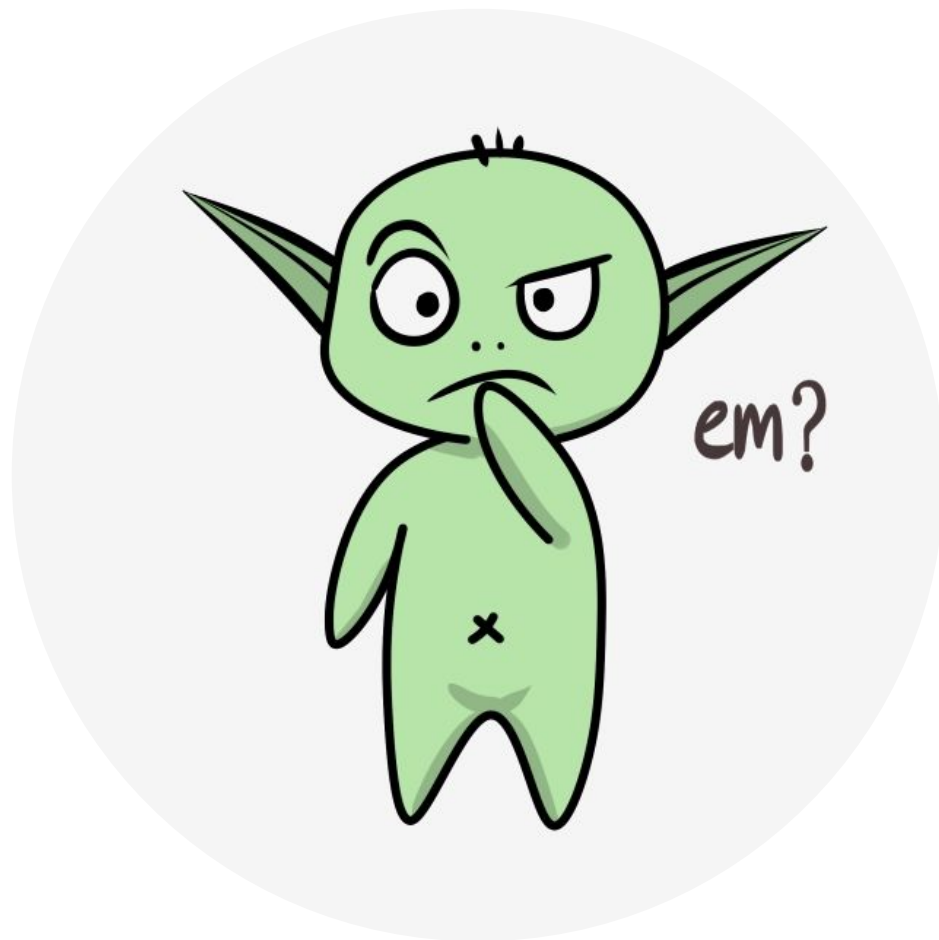
[Mnih et al., 2013, 2015]

| Game | Random Play | Best Linear Learner | Contingency (SARSA) | Human | DQN (\pm std) | Normalized DQN (% Human) |
|-------------|-------------|---------------------|---------------------|-------|----------------------|--------------------------|
| Alien | 227.8 | 939.2 | 103.2 | 6875 | 3069 (\pm 1093) | 42.7% |
| Amidar | 5.8 | 103.4 | 183.6 | 1676 | 739.5 (\pm 3024) | 43.9% |
| Assault | 222.4 | 628 | 537 | 1496 | 3359 (\pm 775) | 246.2% |
| Asterix | 210 | 987.3 | 1332 | 8503 | 6012 (\pm 1744) | 70.0% |
| Asteroids | 719.1 | 907.3 | 89 | 13157 | 1629 (\pm 542) | 7.3% |
| Atlantis | 12850 | 62687 | 852.9 | 29028 | 85641 (\pm 17600) | 449.9% |
| Bank Heist | 14.2 | 190.8 | 67.4 | 734.4 | 429.7 (\pm 650) | 57.7% |
| Battle Zone | 2360 | 15820 | 16.2 | 37800 | 26300 (\pm 7725) | 67.6% |
| Beam Rider | 363.9 | 929.4 | 1743 | 5775 | 6846 (\pm 1619) | 119.8% |
| Bowling | 23.1 | 43.9 | 36.4 | 154.8 | 42.4 (\pm 88) | 14.7% |

Tables can be misleading

[Machado et al., 2018]

- Tables imply an apples-to-apples comparison, even when they are not:
- DQN saw much more data than the baselines.
- DQN measured its performance differently than the baselines.
- DQN used domain knowledge other baselines didn't:
 - Lives signal
 - Action set



Avg. Reward: A Problem Setting for Continuing Tasks

- Continuing problems without discounting.
 - The agent cares about all rewards equally.

Avg. Reward: A Problem Setting for Continuing Tasks

- Continuing problems without discounting.
 - The agent cares about all rewards equally.
- Quality of a policy is defined by the average rate of reward, $r(\pi)$:

$$\begin{aligned}
 r(\pi) &\doteq \lim_{h \rightarrow \infty} \frac{1}{h} \sum_{t=1}^h \mathbb{E}[R_t \mid S_0, A_{0:t-1} \sim \pi] \\
 &= \lim_{t \rightarrow \infty} \mathbb{E}[R_t \mid S_0, A_{0:t-1} \sim \pi], \\
 &= \sum_s \mu_\pi(s) \sum_a \pi(a|s) \sum_{s', r} p(s', r | s, a) r
 \end{aligned}$$

If the MDP is *ergodic*: the starting state and any early decision made by the agent can have only a temporary effect; in the long run the expectation of being in a state depends only on the policy and the MDP transition probabilities.

Avg. Reward: A Problem Setting for Continuing Tasks

- (Differential) Return:

$$G_t \doteq R_{t+1} - r(\pi) + R_{t+2} - r(\pi) + R_{t+3} - r(\pi) + \dots$$

Avg. Reward: A Problem Setting for Continuing Tasks

- (Differential) Return:

$$G_t \doteq R_{t+1} - r(\pi) + R_{t+2} - r(\pi) + R_{t+3} - r(\pi) + \dots$$

- Differential value functions:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{r,s'} p(s', r | s, a) \left[r - r(\pi) + v_\pi(s') \right],$$

$$q_\pi(s, a) = \sum_{r,s'} p(s', r | s, a) \left[r - r(\pi) + \sum_{a'} \pi(a'|s') q_\pi(s', a') \right],$$

$$v_*(s) = \max_a \sum_{r,s'} p(s', r | s, a) \left[r - \max_\pi r(\pi) + v_*(s') \right], \text{ and}$$

$$q_*(s, a) = \sum_{r,s'} p(s', r | s, a) \left[r - \max_\pi r(\pi) + \max_{a'} q_*(s', a') \right]$$

Avg. Reward: A Problem Setting for Continuing Tasks

- Differential value functions:

$$v_{\pi}(s) = \sum_a \pi(a|s) \sum_{r,s'} p(s', r | s, a) \left[r - r(\pi) + v_{\pi}(s') \right],$$

$$q_{\pi}(s, a) = \sum_{r,s'} p(s', r | s, a) \left[r - r(\pi) + \sum_{a'} \pi(a'|s') q_{\pi}(s', a') \right],$$

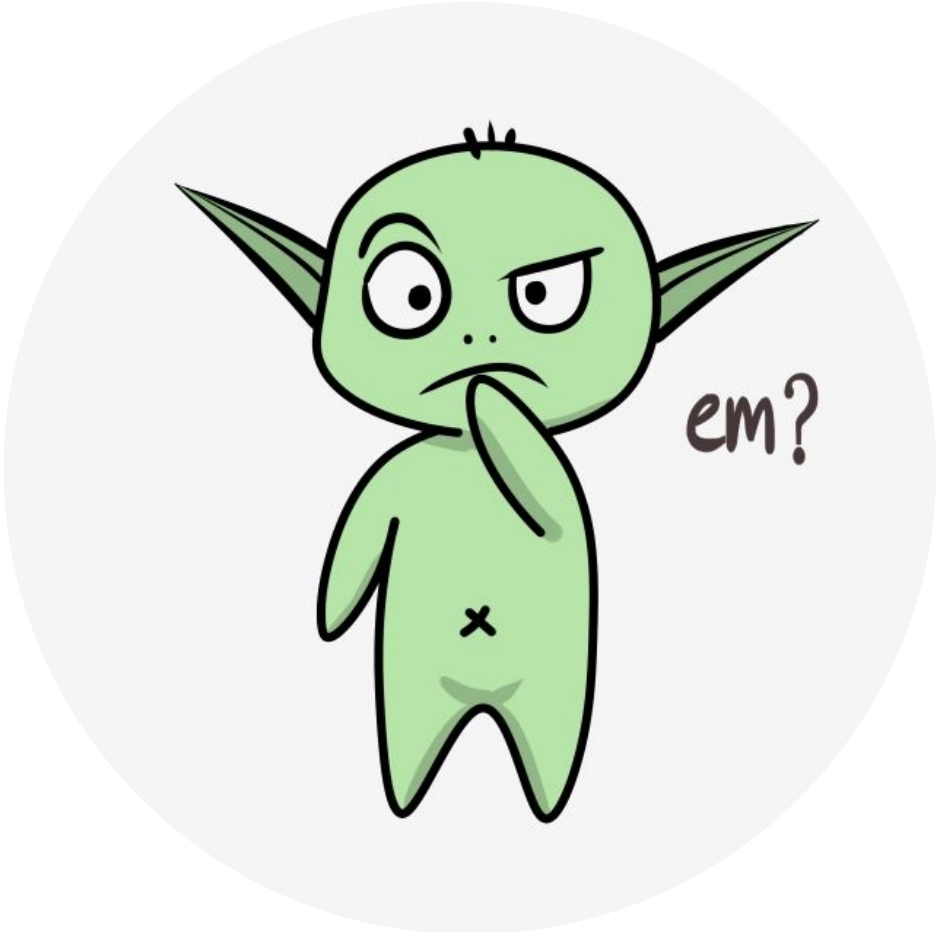
$$v_{*}(s) = \max_a \sum_{r,s'} p(s', r | s, a) \left[r - \max_{\pi} r(\pi) + v_{*}(s') \right], \text{ and}$$

$$q_{*}(s, a) = \sum_{r,s'} p(s', r | s, a) \left[r - \max_{\pi} r(\pi) + \max_{a'} q_{*}(s', a') \right]$$

- Differential TD error:

$$\delta_t \doteq R_{t+1} - \bar{R}_t + \hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t),$$

$$\delta_t \doteq R_{t+1} - \bar{R}_t + \hat{q}(S_{t+1}, A_{t+1}, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t).$$



Differential semi-gradient Sarsa

Differential semi-gradient Sarsa for estimating $\hat{q} \approx q_*$

Input: a differentiable action-value function parameterization $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \rightarrow \mathbb{R}$

Algorithm parameters: step sizes $\alpha, \beta > 0$, small $\varepsilon > 0$

Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)

Initialize average reward estimate $\bar{R} \in \mathbb{R}$ arbitrarily (e.g., $\bar{R} = 0$)

Initialize state S , and action A

Loop for each step:

 Take action A , observe R, S'

 Choose A' as a function of $\hat{q}(S', \cdot, \mathbf{w})$ (e.g., ε -greedy)

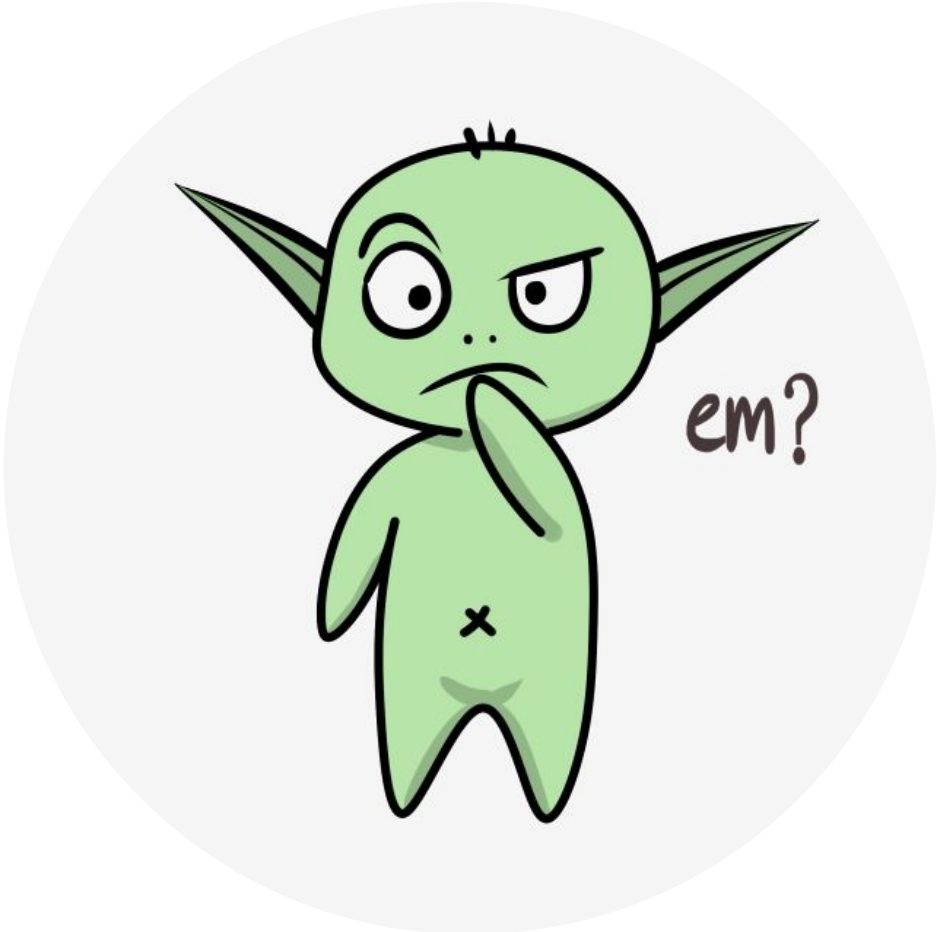
$\delta \leftarrow R - \bar{R} + \hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w})$

$\bar{R} \leftarrow \bar{R} + \beta\delta$

$\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\nabla\hat{q}(S, A, \mathbf{w})$

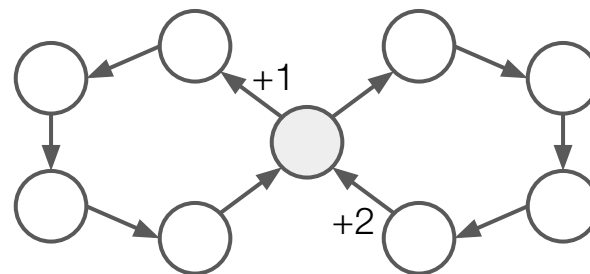
$S \leftarrow S'$

$A \leftarrow A'$



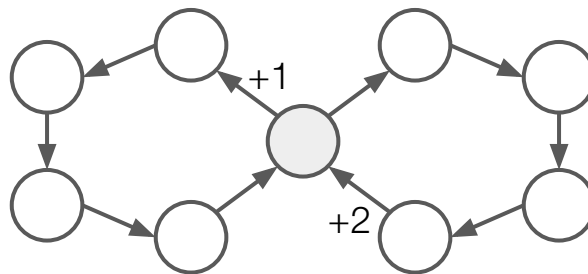
Avg. Reward: A Problem Setting for Continuing Tasks

- Continuing problems without discounting.
 - The agent cares about all rewards equally.



Avg. Reward: A Problem Setting for Continuing Tasks

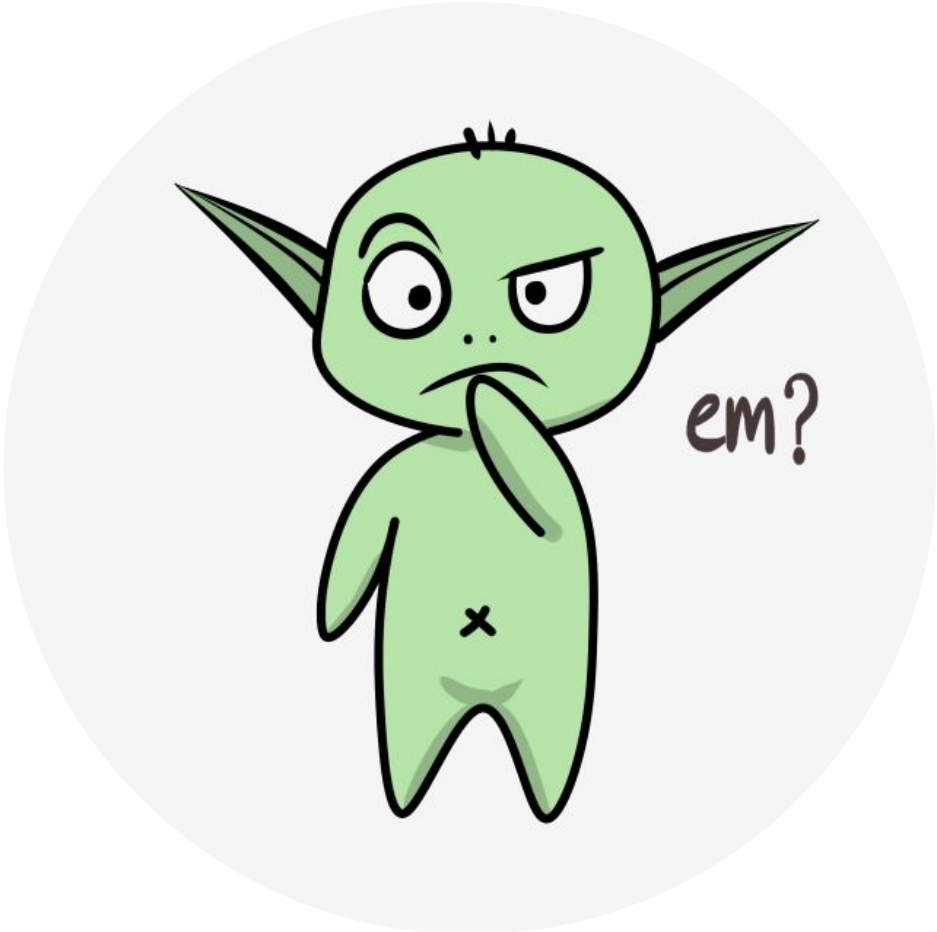
- Continuing problems without discounting.
 - The agent cares about all rewards equally.



- Quality of a policy is defined by the average rate of reward, $r(\pi)$:

$$\begin{aligned}
 r(\pi) &\doteq \lim_{h \rightarrow \infty} \frac{1}{h} \sum_{t=1}^h \mathbb{E}[R_t \mid S_0, A_{0:t-1} \sim \pi] \\
 &= \lim_{t \rightarrow \infty} \mathbb{E}[R_t \mid S_0, A_{0:t-1} \sim \pi], \\
 &= \sum_s \mu_\pi(s) \sum_a \pi(a|s) \sum_{s', r} p(s', r \mid s, a) r
 \end{aligned}$$

If the MDP is ergodic: the starting state and any early decision made by the agent can have only a temporary effect; in the long run the expectation of being in a state depends only on the policy and the MDP transition probabilities.



Exercise

In the context of control algorithms *with function approximation*, please provide:

(a) The general form of the update rule for semi-gradient one-step Q-Learning.

(b) The specific update, with no generic gradient terms, for semi-gradient one-step Q-learning *with linear function approximation*.

Exercise

In the context of control algorithms *with function approximation*, please provide:

(a) The general form of the update rule for semi-gradient one-step Q-Learning.

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha \left[R + \gamma \max_{a'} \hat{q}(S', a', \mathbf{w}) - \hat{q}(S, A, \mathbf{w}) \right] \nabla_{\mathbf{w}} \hat{q}(S, A, \mathbf{w})$$

(b) The specific update, with no generic gradient terms, for semi-gradient one-step Q-learning *with linear function approximation*.

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha \left[R + \gamma \max_{a'} \mathbf{w}^\top \mathbf{x}(S', a') - \mathbf{w}^\top \mathbf{x}(S, A) \right] \mathbf{x}(S, A)$$

Exercise

Exercise 10.7 Consider a Markov reward process consisting of a ring of three states A, B, and C, with state transitions going deterministically around the ring. A reward of +1 is received upon arrival in A and otherwise the reward is 0. What are the differential values of the three states, using (10.13)? \square

$$v_{\pi}(s) \doteq \lim_{\gamma \rightarrow 1} \lim_{h \rightarrow \infty} \sum_{t=0}^h \gamma^t \left(\mathbb{E}_{\pi}[R_{t+1} | S_0 = s] - r(\pi) \right). \quad (10.13)$$

Exercise

Exercise 10.6 Suppose there is an MDP that under any policy produces the deterministic sequence of rewards $+1, 0, +1, 0, +1, 0, \dots$ going on forever. Technically, this violates ergodicity; there is no stationary limiting distribution μ_π and the limit (10.7) does not exist. Nevertheless, the average reward (10.6) is well defined. What is it? Now consider two states in this MDP. From **A**, the reward sequence is exactly as described above, starting with a $+1$, whereas, from **B**, the reward sequence starts with a 0 and then continues with $+1, 0, +1, 0, \dots$. We would like to compute the differential values of **A** and **B**. Unfortunately, the differential return (10.9) is not well defined when starting from these states as the implicit limit does not exist. To repair this, one could alternatively define the differential value of a state as

$$v_\pi(s) \doteq \lim_{\gamma \rightarrow 1} \lim_{h \rightarrow \infty} \sum_{t=0}^h \gamma^t \left(\mathbb{E}_\pi[R_{t+1} | S_0 = s] - r(\pi) \right). \quad (10.13)$$

Under this definition, what are the differential values of states **A** and **B**? □