“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

Marlos C. Machado

https://www.duneawakening.com/en/developer-update-2-sounds-of-arrakis

Class 29/35

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.

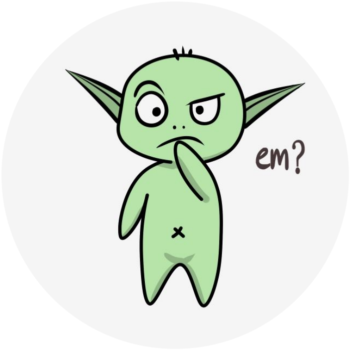
The deadlines in the public session **do not align** with the deadlines in Coursera.

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

Reminders and Notes

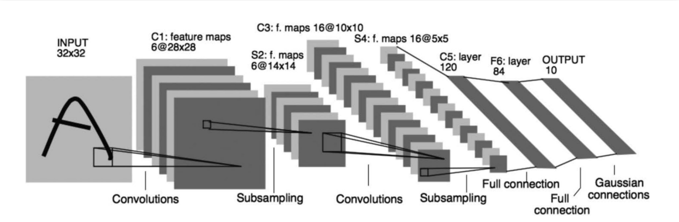
- The practice quiz for Control with FA is due today (only 4 to go!).
- The programming assignment for Control with FA is due Wednesday.
- Rich Sutton will give a guest lecture Dec 9th, Monday. Spread the word.
- A note on the final exam:
  - The required reading from the syllabus does not mean that's what will be covered in the final exam. There are some mismatches. Anything we discussed in class is fair game, including Maximization Bias and Double Learning (Section 6.7), and Nonlinear Function Approximation: Artificial Neural Networks (Section 9.7).
- The Student Perspectives of Teaching (SPOT) Survey is now available.

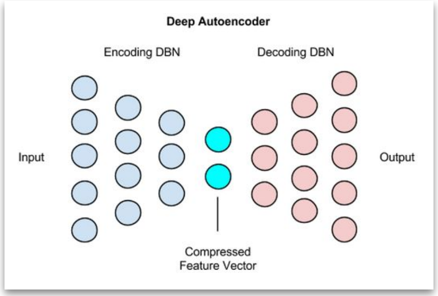
Please, interrupt me at any time!



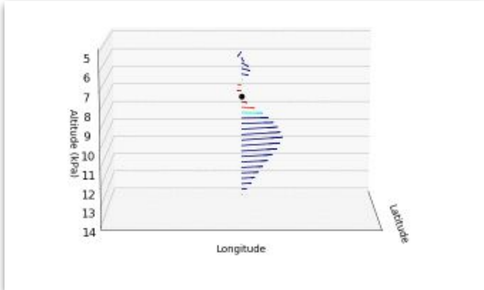
https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Last Class: Neural Networks and Feature/Input Construction





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



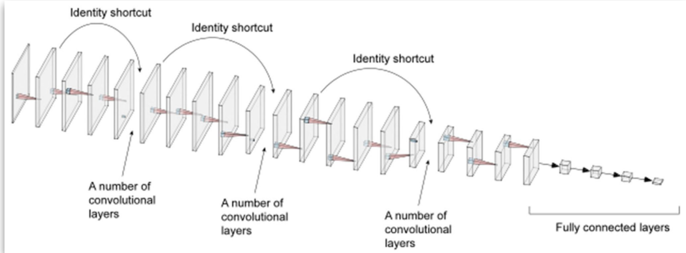
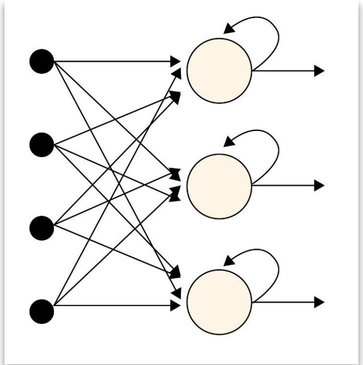


Image by Yu, Miao, and Wang (2022)



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

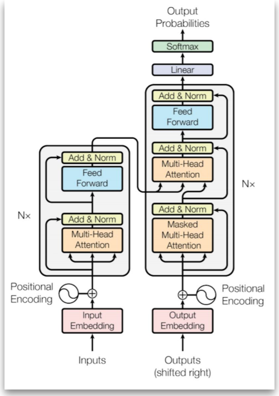
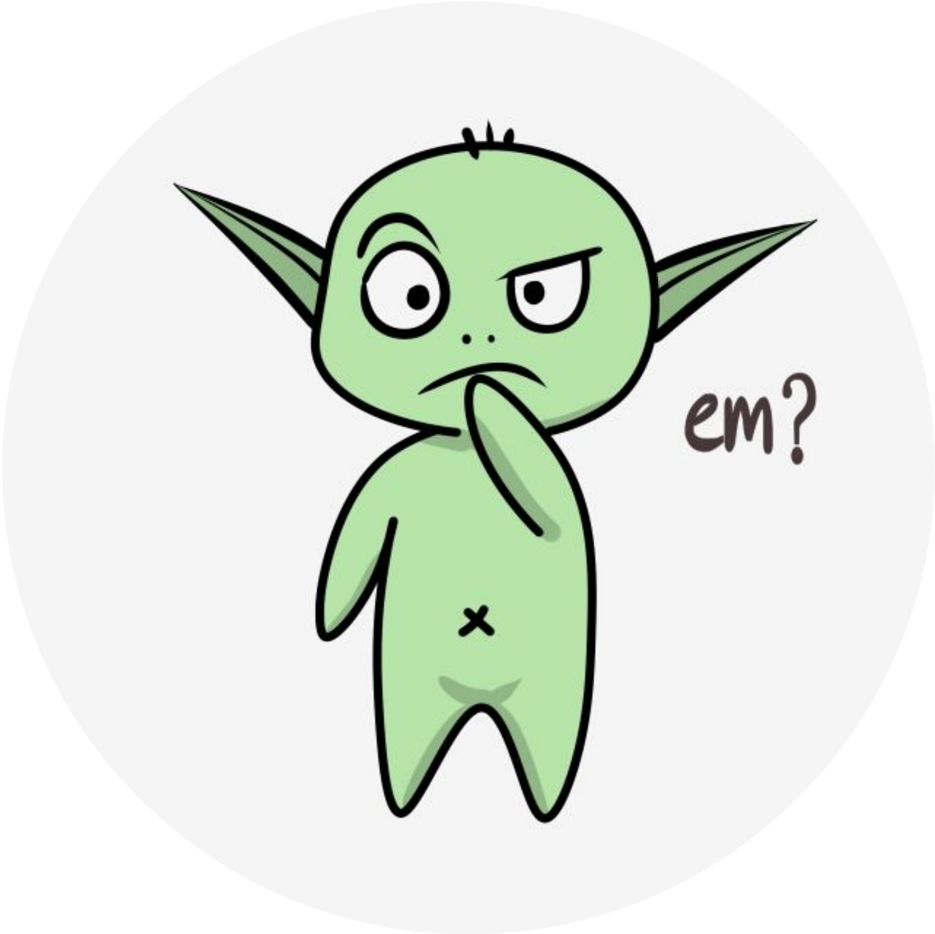


Image by Vaswani et al. (2017)



https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

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 **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 **w**.
- Before (until last class): $S_t \mapsto U_t$.
  Now: $S_t, A_t \mapsto U_t$.
- Action-value prediction:

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

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 **w**.
- Before (until last class): $S_t \mapsto U_t$.
  Now: S_t, $A_t \mapsto U_t$.
- Action-value prediction:

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

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

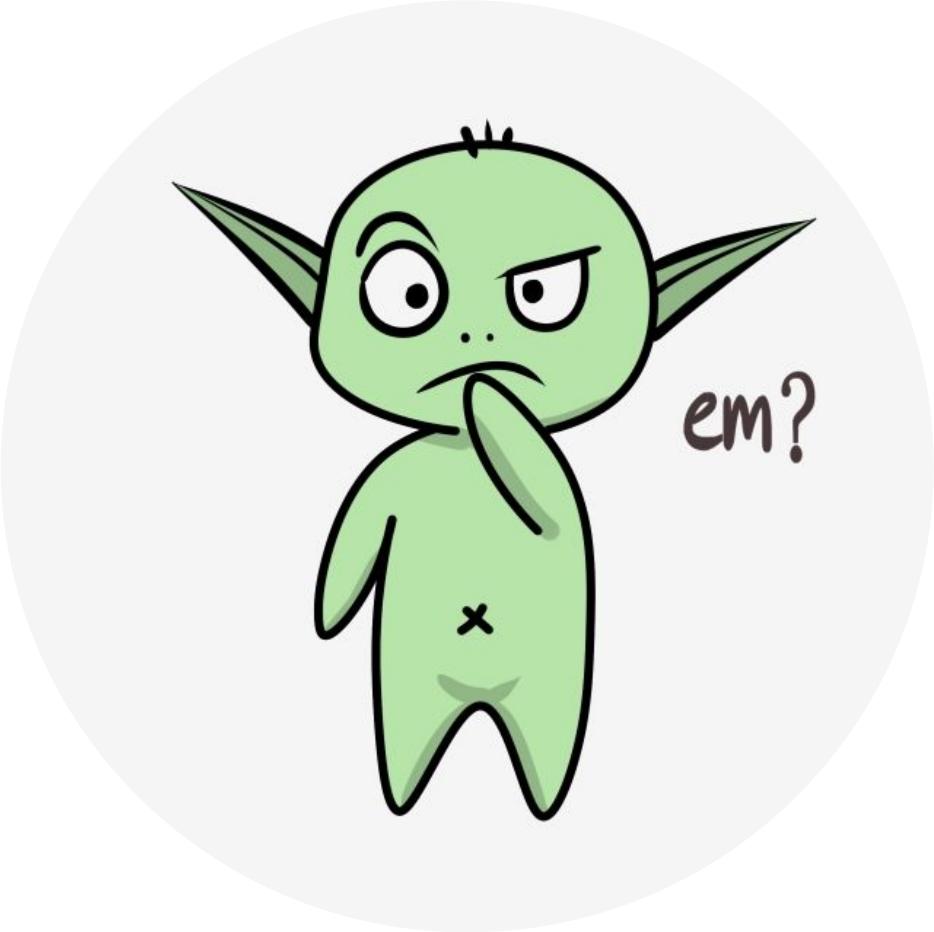
$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \Big[R_{t+1} + \gamma \hat{q}(S_{t+1}, A_{t+1}, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t) \Big] \nabla \hat{q}(S_t, A_t, \mathbf{w}_t)$$

Episodic Semi-gradient Sarsa

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

Input: a differentiable action-value function parameterization $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to \mathbb{R}$
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:
      $\mathbf{w} \leftarrow \mathbf{w} + \alpha\big[R - \hat{q}(S, A, \mathbf{w})\big]\nabla\hat{q}(S, A, \mathbf{w})$
      Go to next episode
    Choose A' as a function of $\hat{q}(S', \cdot, \mathbf{w})$ (e.g., ε-greedy)
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha\big[R + \gamma\hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w})\big]\nabla\hat{q}(S, A, \mathbf{w})$
    $S \leftarrow S'$
    $A \leftarrow A'$



https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

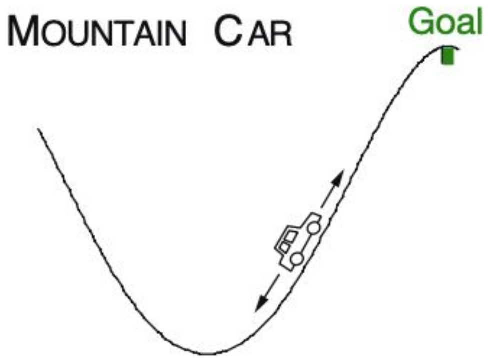
Example: Mountain Car Task

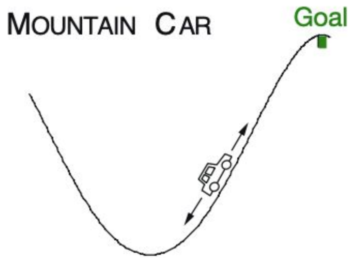
- Observations: (x, ẋ)
- Actions:
  - Full throttle forward: +1
  - Full throttle reverse: -1
  - Zero throttle: 0
- Rewards: -1 at every time step, until end of episode.
- Dynamics:

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

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

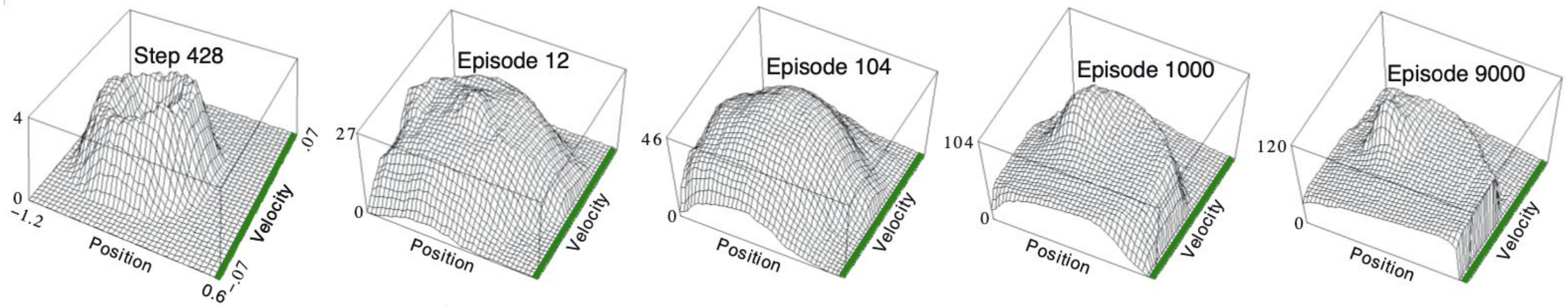
$$-1.2 \leq x_{t+1} \leq 0.5 \text{ and } -0.07 \leq \dot{x}_{t+1} \leq 0.07$$

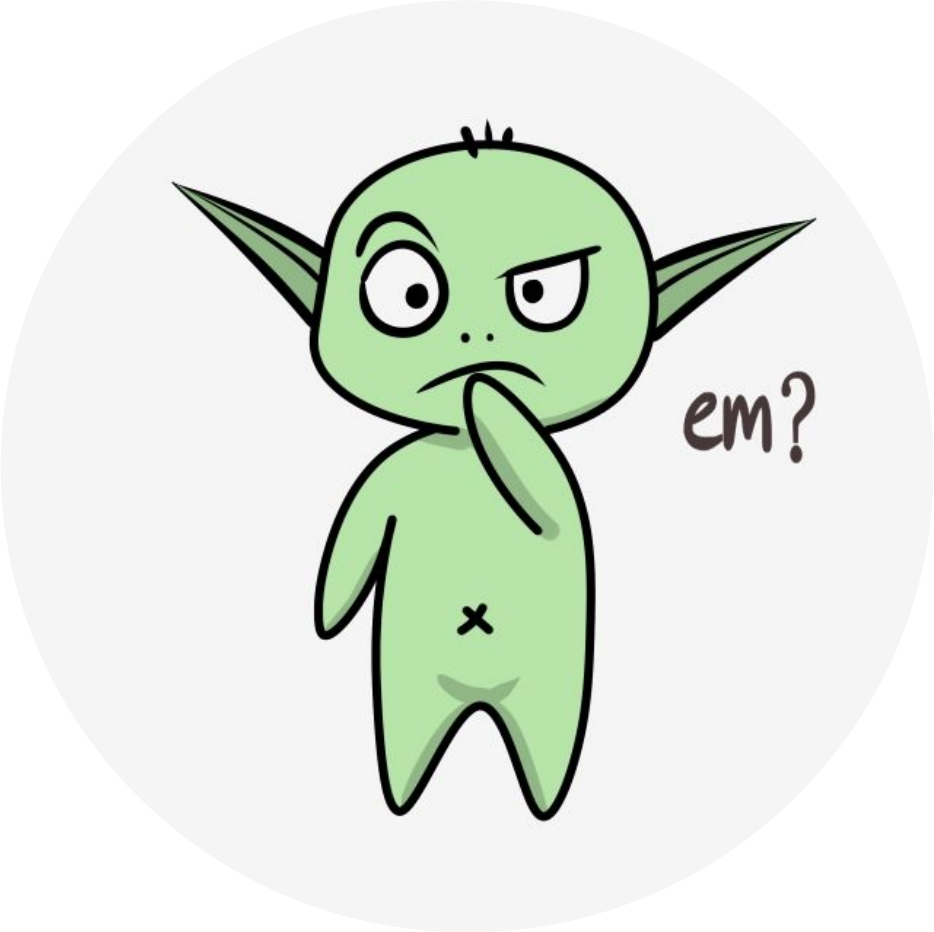




"Solution": Mountain Car Task

- 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.





https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

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(π):

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

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.

Avg. Reward: A Problem Setting for Continuing Tasks

- (Differential) Return:

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

Avg. Reward: A Problem Setting for Continuing Tasks

- (Differential) Return:

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

- Differential value functions:

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

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

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

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

Avg. Reward: A Problem Setting for Continuing Tasks

- Differential value functions:

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

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

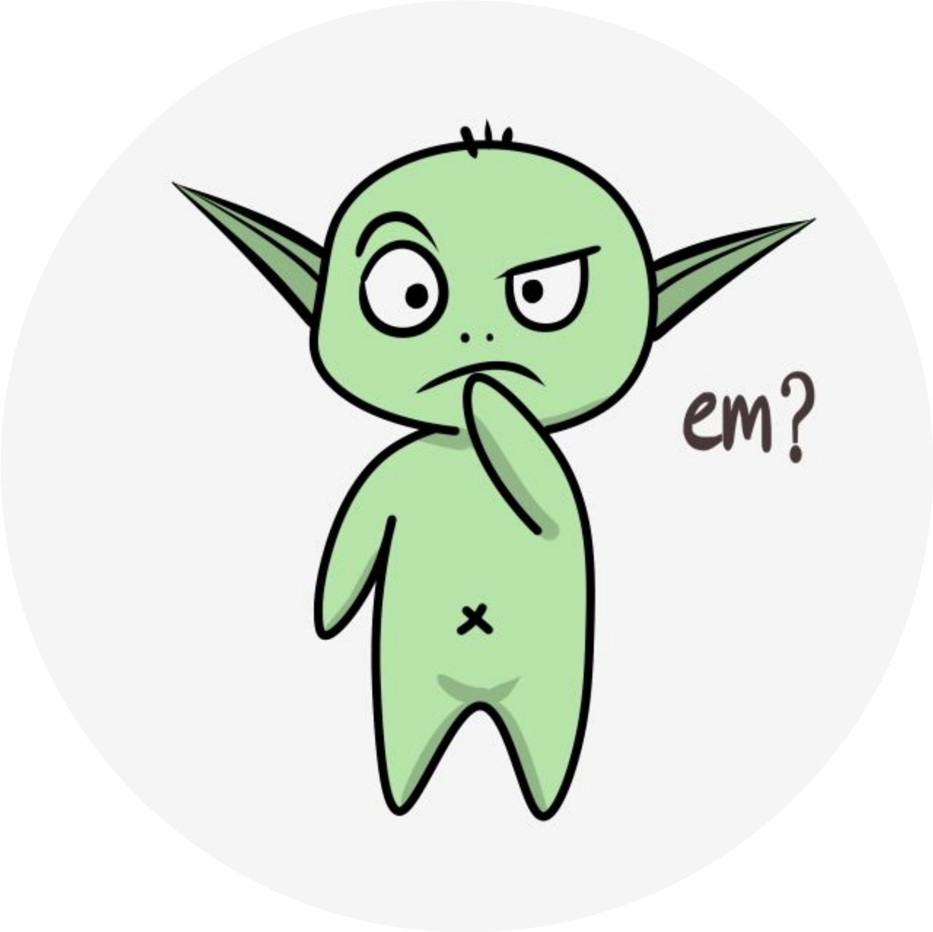
$$v_*(s) = \max_a \sum_{r,s'} p(s', r | s, a)\Big[r - \max_\pi r(\pi) + v_*(s')\Big], \text{ and}$$

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

- Differential TD error:

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

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



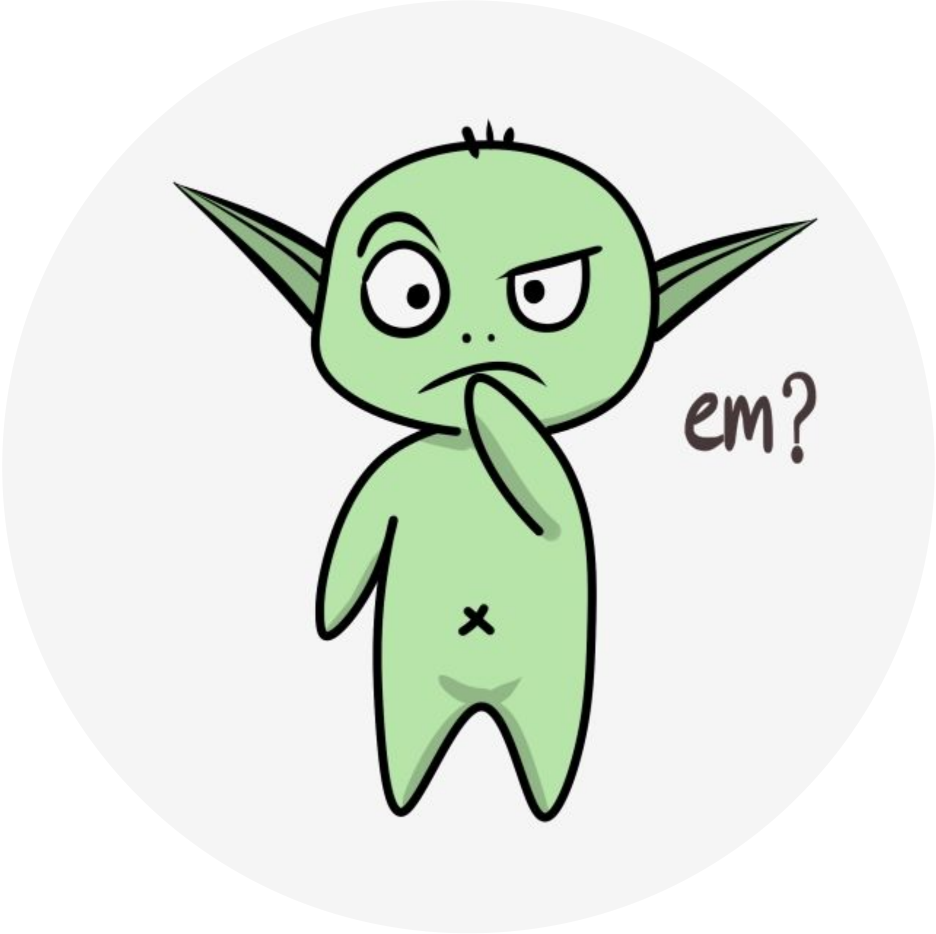
https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Differential semi-gradient Sarsa

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

Input: a differentiable action-value function parameterization $\hat{q} : \mathcal{S} \times \mathcal{A} \times \mathbb{R}^d \to \mathbb{R}$
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)
    $\delta \leftarrow R - \bar{R} + \hat{q}(S', A', \mathbf{w}) - \hat{q}(S, A, \mathbf{w})$
    $\bar{R} \leftarrow \bar{R} + \beta\delta$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\nabla\hat{q}(S, A, \mathbf{w})$
    $S \leftarrow S'$
    $A \leftarrow A'$



https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht