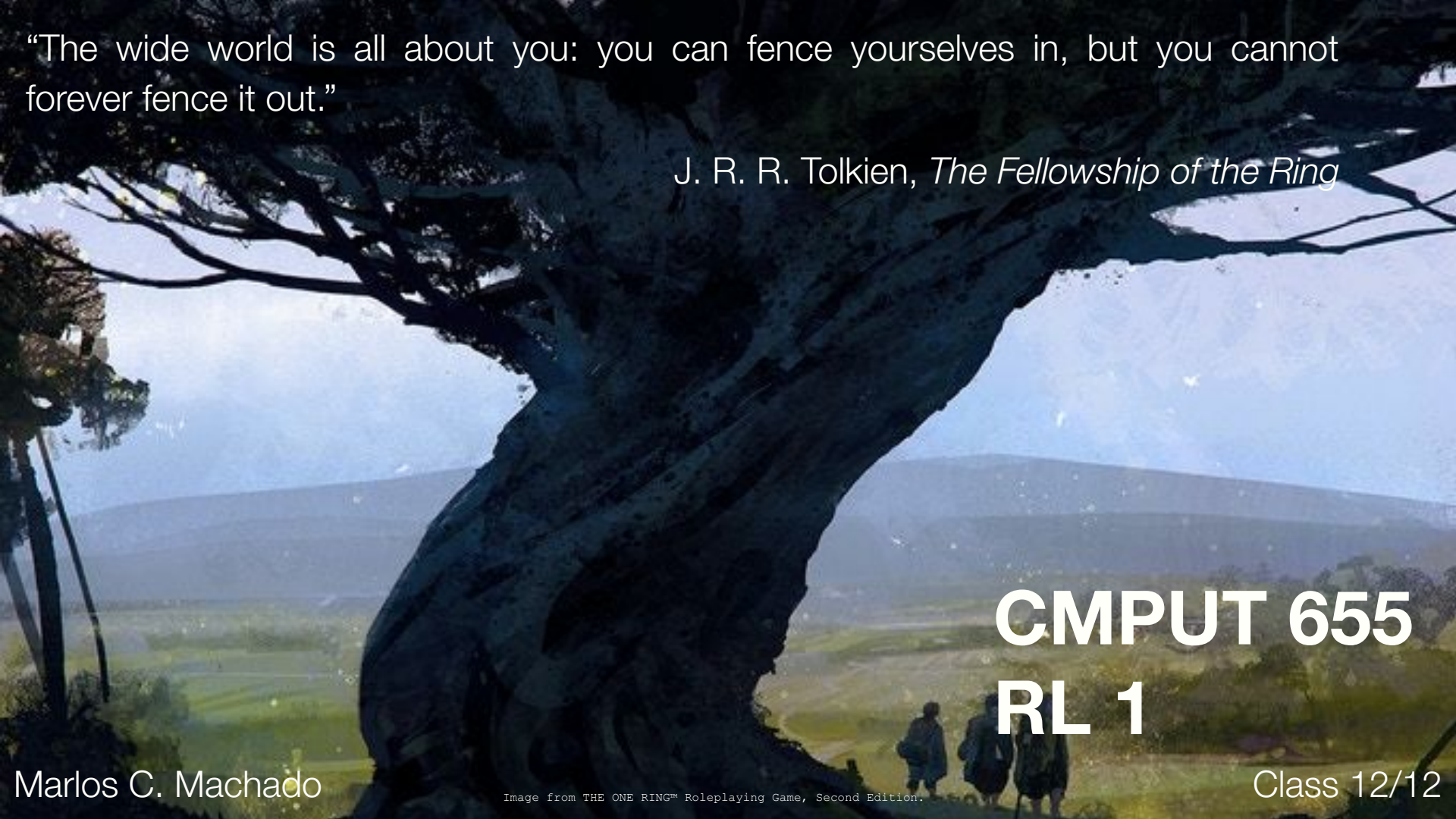


"The wide world is all about you: you can fence yourselves in, but you cannot forever fence it out."

J. R. R. Tolkien, *The Fellowship of the Ring*

CMPUT 655
RL 1

Marlos C. Machado

Image from THE ONE RING™ Roleplaying Game, Second Edition.

Class 12/12

Reminders

- The final Project Report is due on December 15th.
  - The link on eClass is already open.
  - I cannot accept late submissions.
- 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: Deep Q-Network (DQN)

[Mnih et al. 2013, 2015]

Stacked frames

-1, 0, +1 rewards

Clipped error term

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

Experience replay buffer (Lin, 1993)
Original size: 1M frames

Target network
Original update frequency: 10k

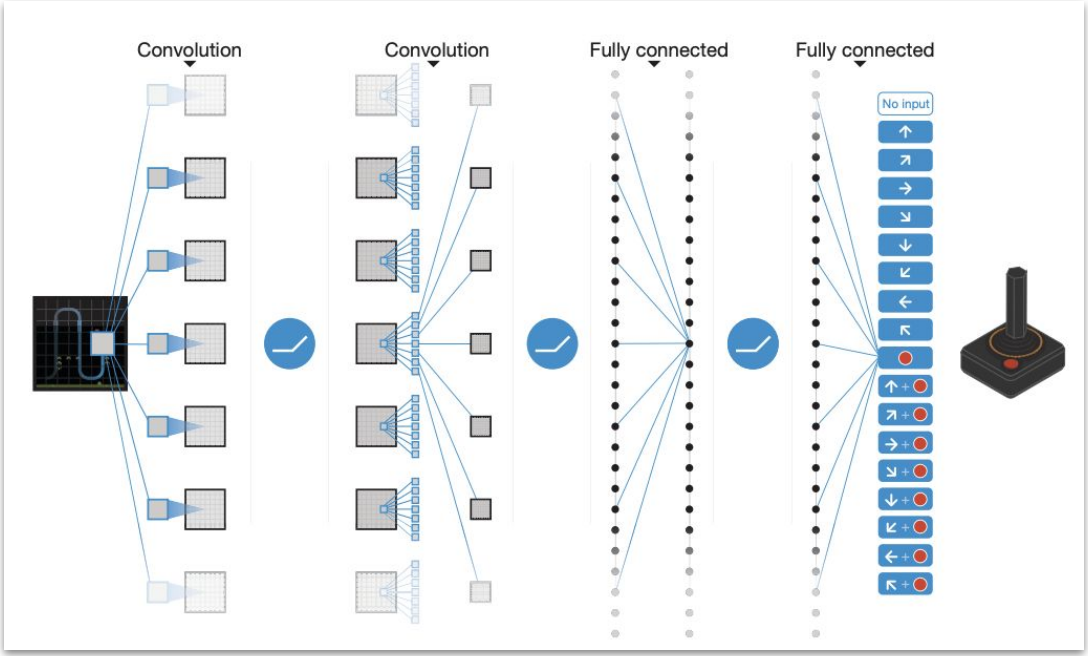
ε decay
Originally, from 1.0 to 0.1 over 1M frames

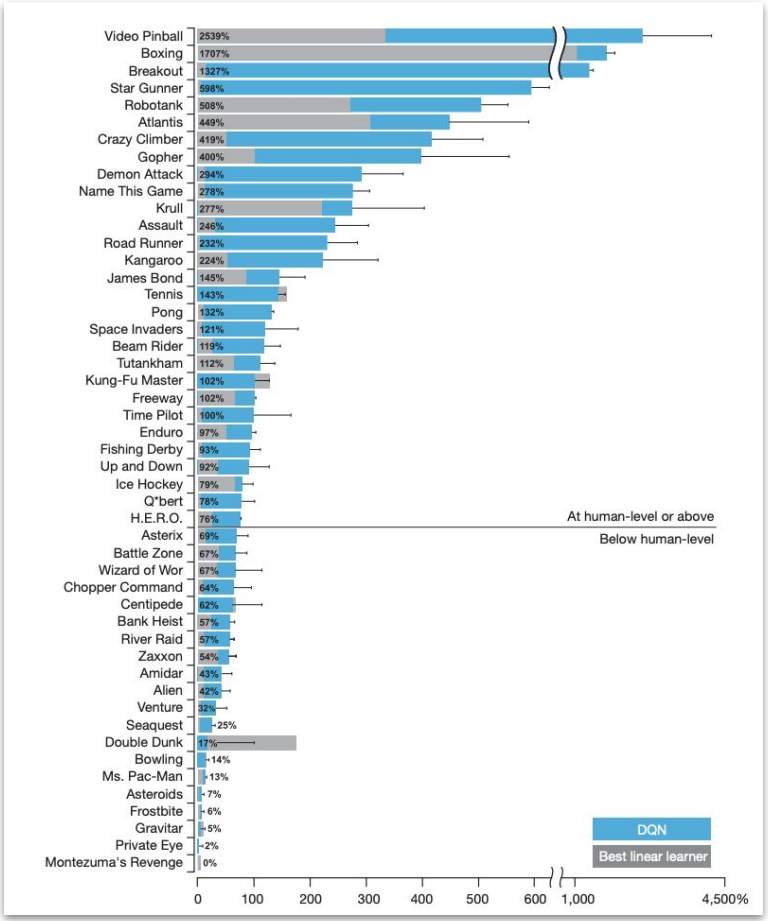
$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha\left[R_{t+1} + \gamma \max_{a' \in \mathcal{A}} Q(S_{t+1}, a'; \boldsymbol{\theta}^-) - Q(S_t, A_t, \boldsymbol{\theta}_t)\right]\nabla_{\boldsymbol{\theta}_t} Q(S_t, A_t; \boldsymbol{\theta}_t)$$

RMSProp

Last Class: Deep Q-Network

[Mnih et al. 2013, 2015]

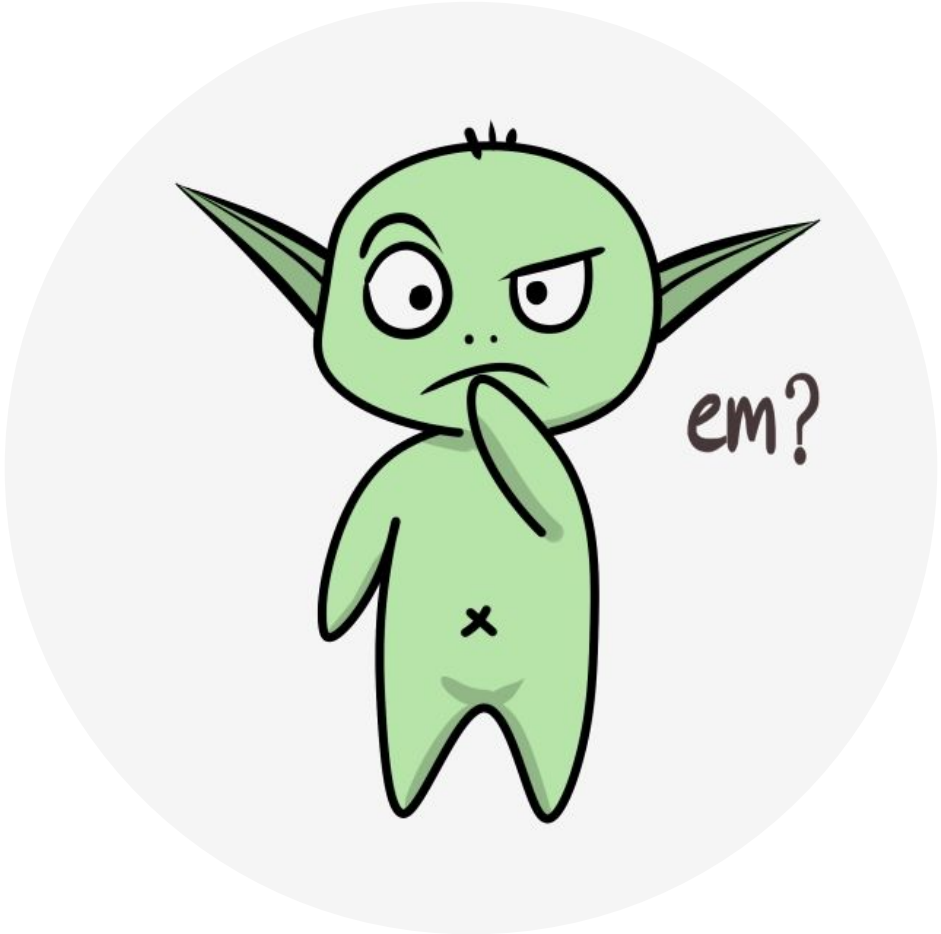




Deep Q-Network (DQN)

[Mnih et al. 2013, 2015]

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



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

Deep Q-Network (DQN)

[Mnih et al. 2013, 2015]

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

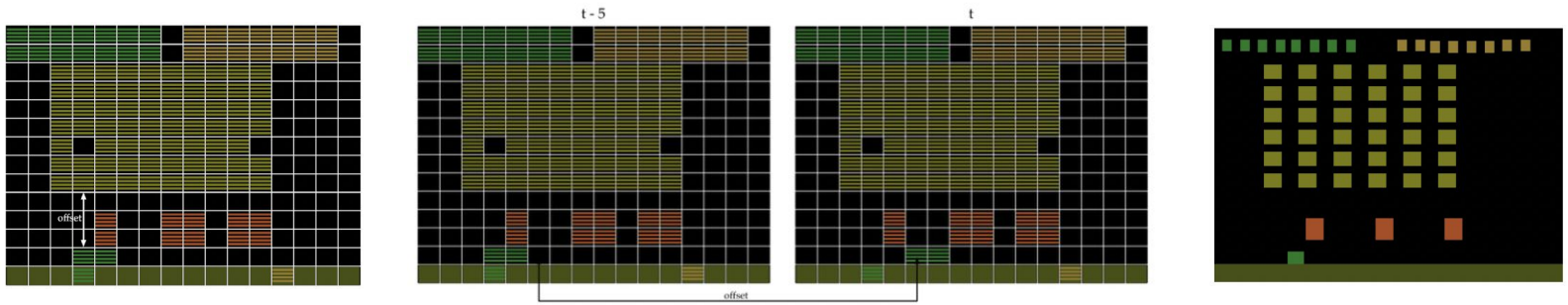
This can be a big deal

[Liang et al. 2016]

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
{yliang, erik.talvitie}@fandm.edu

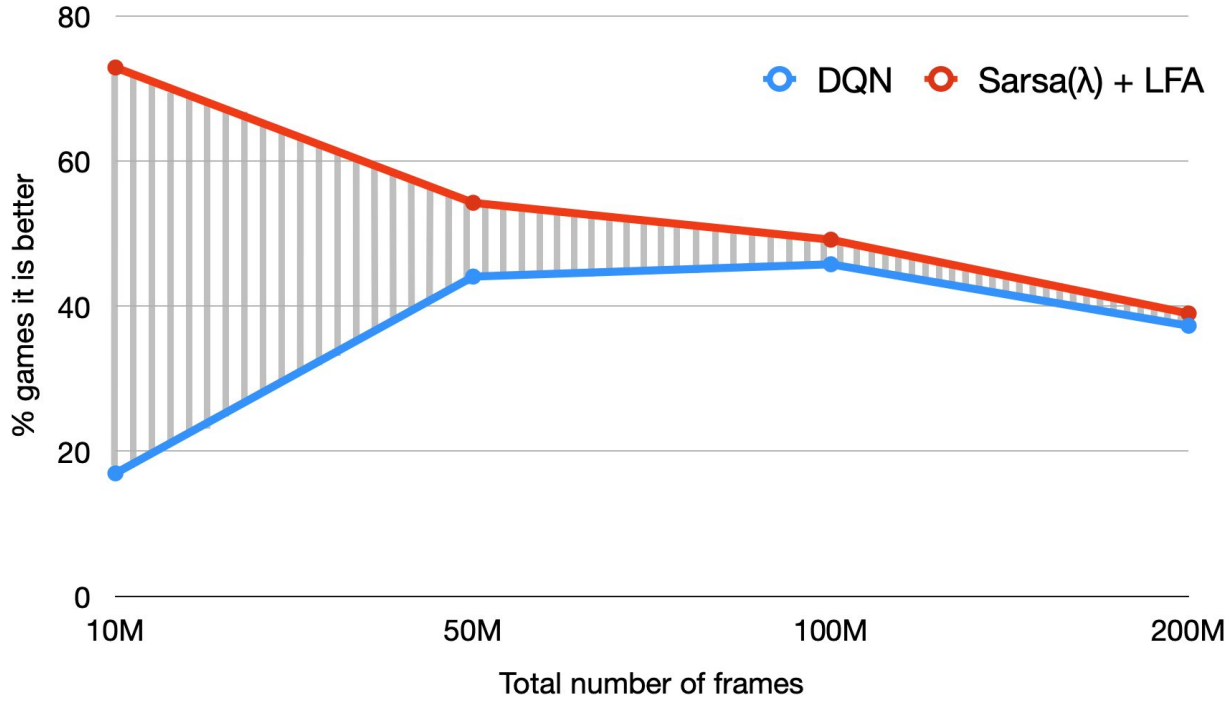
[‡]University of Alberta
Edmonton, AB, Canada
{machado, mbowling}@ualberta.ca

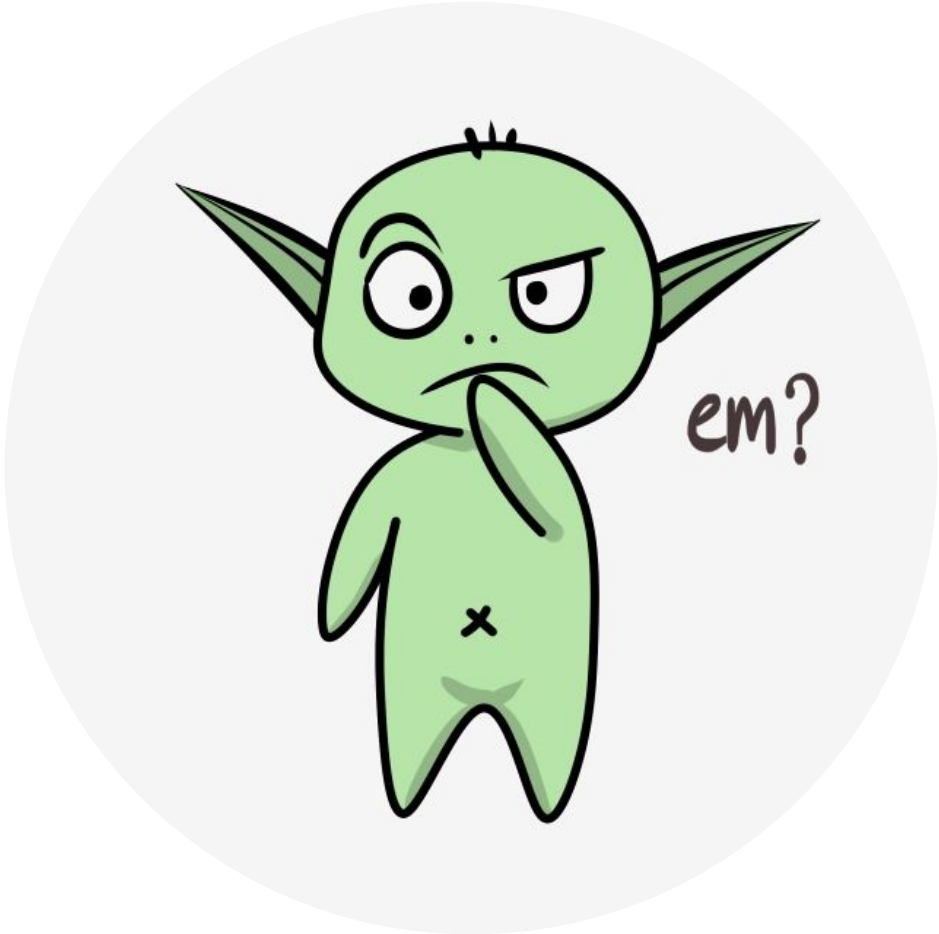


It is not that we should be using linear function approxim.

[Liang et al. 2016; Machado et al. 2018]

but we should understand and be careful with our claims





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

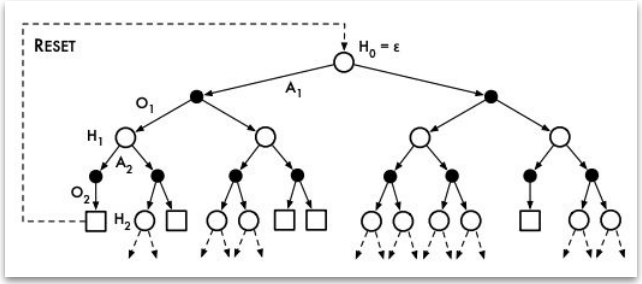
Do we need clever data processing?

Understanding your environment

[Machado et al. 2018]

- The Arcade Learning Environment (was) deterministic, episodic, and guaranteed a unique starting state, and
- in most Atari 2600 games, purpose matters more than individual actions, i.e., most Atari 2600 games have important high-level goals, but individual actions have little impact.

Can we exploit that?



The Brute

[Bellemare et al. 2015, Machado et al. 2018]

- A history is a sequence of actions and observations ht = a_1, o_1, a_2, o_2, …, o_t, with the reward r_t included in the observation o_t.
- Given a full history tree of finite depth, estimating the value for any history-action pair is simply a matter of bottom-up dynamic programming.
- In fact, we can leverage an important property of history trees: Consider a partially known history tree for a deterministic environment and define $\hat{q}(h, a) = -\infty$ for any unknown history-action pair. Then the equation

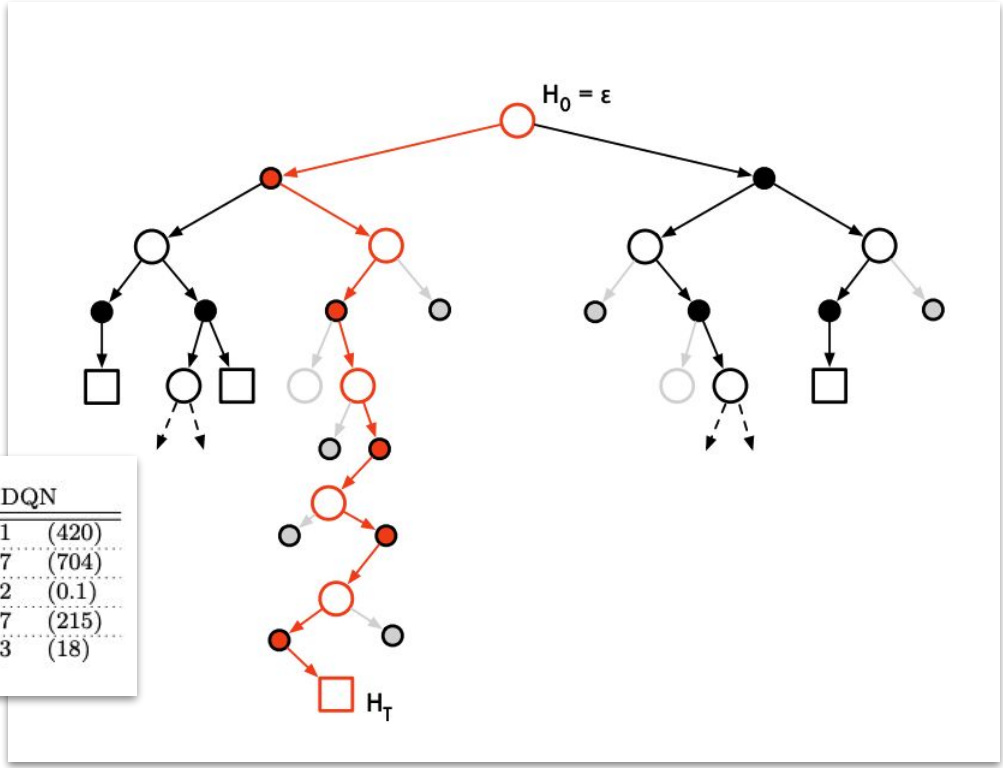
$$\hat{q}(h,a) = \sum_{h'} p(h' \mid h, a) \left(r(h,a,h') + \gamma \max_{b \in \mathcal{A}} \hat{q}(h', b) \right)$$

defines a lower bound on $q_*(h, a)$.

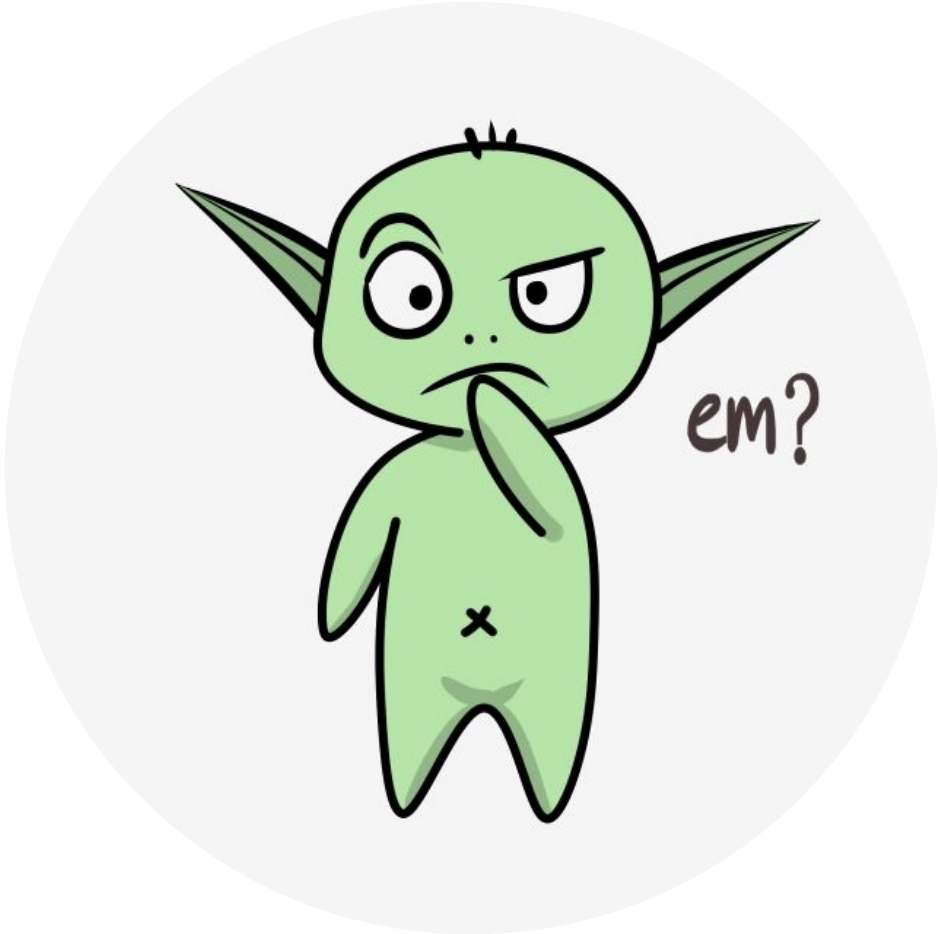
The Brute

[Bellemare et al. 2015, Machado et al. 2018]

- When learning proceeds in episodes, we can update the lower bound $\hat{q}(h, a)$ iteratively.
- Exploration: ε-greedy policy, with a decreasing ε.



Game	The Brute		Sarsa(λ) + Blob-PROST		DQN	
Asterix	6,909	(1,018)	4,173	(872)	3,501	(420)
Beam Rider	1,132	(178)	2,098	(508)	4,687	(704)
Freeway	1.1	(0.4)	32.1	(0.4)	32.2	(0.1)
Seaquest	621	(192)	1,340	(245)	1,397	(215)
Space Invaders	1,432	(226)	723	(86)	673	(18)

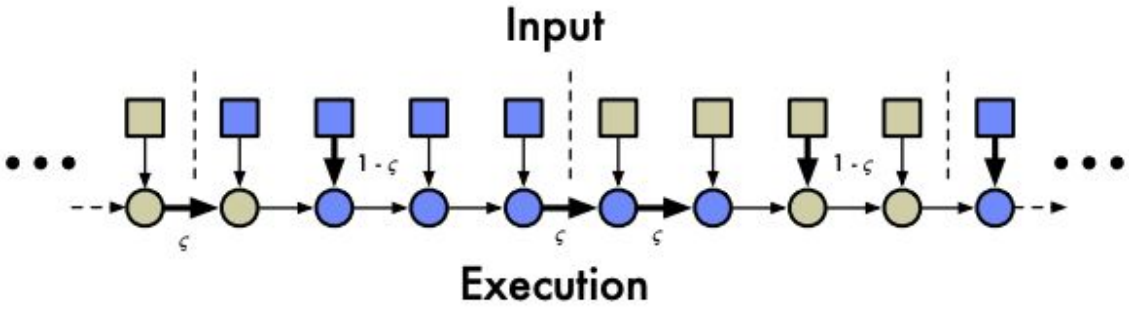


https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

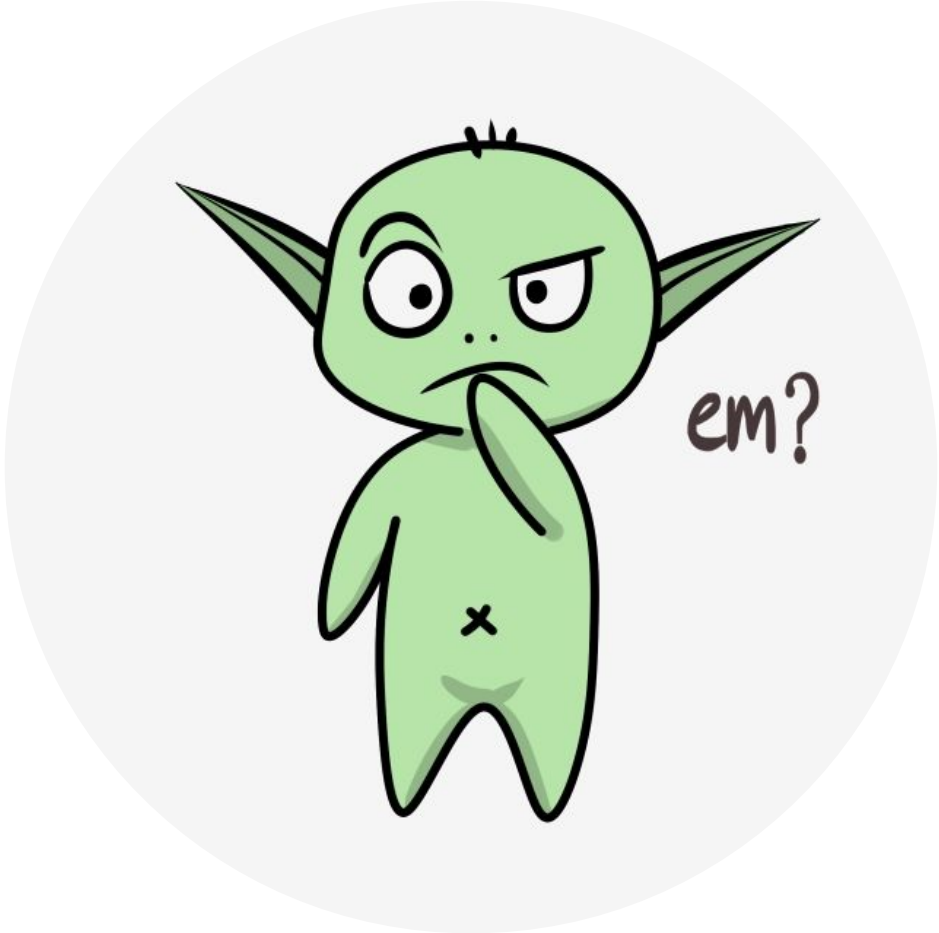
Sticky Actions: Introducing Stochasticity to the ALE

[Machado et al. 2018]

$$A_t = \begin{cases} a, & \text{with prob.} \quad 1-\varsigma, \\ a_{t-1}, & \text{with prob.} \quad \varsigma. \end{cases}$$



Game	The Brute				Sarsa(λ) + Blob-PROST				DQN			
	Determin.		Stochast.		Determin.		Stochast.		Determin.		Stochast.	
Asterix	6909	(1018)	308	(31)	4173	(872)	3411	(414)	3501	(420)	3123	(96)
Beam Rider	1132	(178)	428	(18)	2098	(508)	1851	(407)	4687	(704)	4552	(849)
Freeway	1.1	(0.4)	0.0	(0.0)	32.1	(0.4)	31.8	(0.3)	32.2	(0.1)	31.6	(0.7)
Seaquest	621	(192)	81	(7)	1340	(245)	1204	(190)	1397	(215)	1431	(162)
Space Invaders	1432	(226)	148	(11)	723	(86)	583	(31)	673	(18)	687	(37)



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

Last Class: Deep Q-Network (DQN)

[Mnih et al. 2013, 2015]

Stacked frames

-1, 0, +1 rewards

Clipped error term

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

Experience replay buffer (Lin, 1993)
Original size: 1M frames

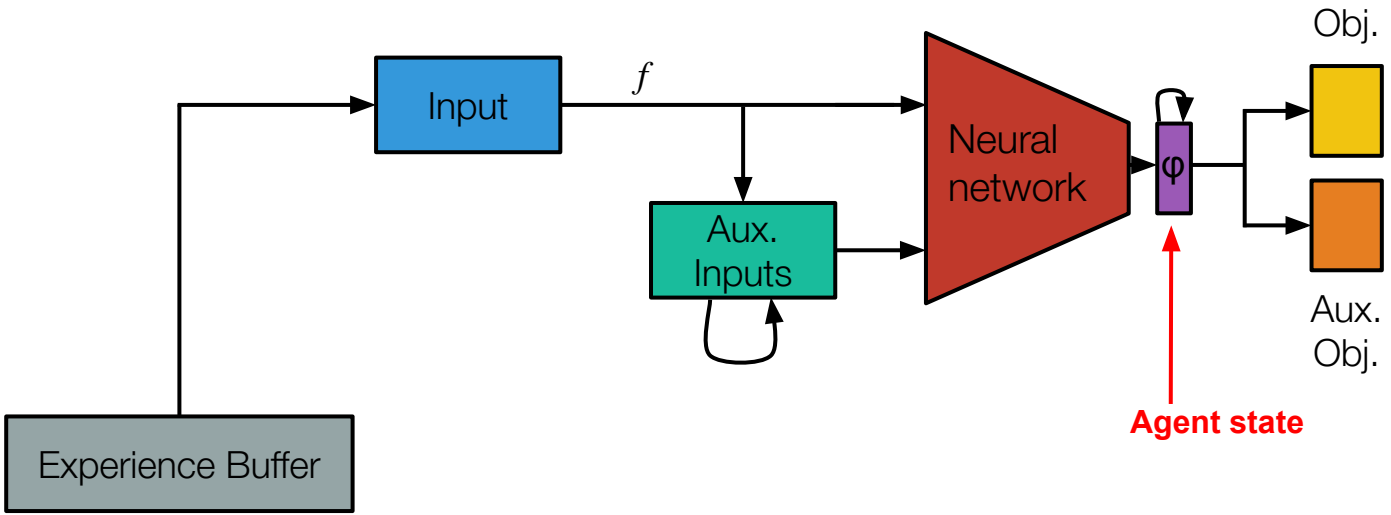
Target network
Original update frequency: 10k

ε decay
Originally, from 1.0 to 0.1 over 1M frames

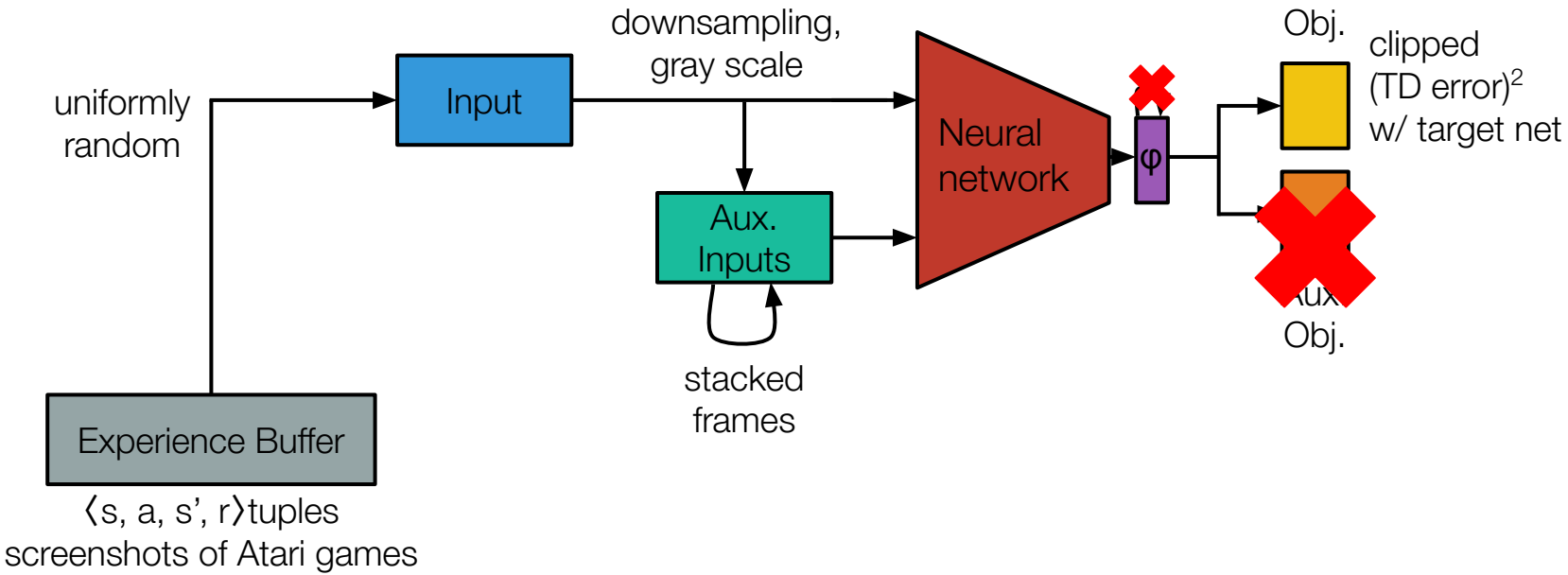
$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha\left[R_{t+1} + \gamma \max_{a' \in \mathcal{A}} Q(S_{t+1}, a'; \boldsymbol{\theta}^-) - Q(S_t, A_t, \boldsymbol{\theta}_t)\right]\nabla_{\boldsymbol{\theta}_t} Q(S_t, A_t; \boldsymbol{\theta}_t)$$

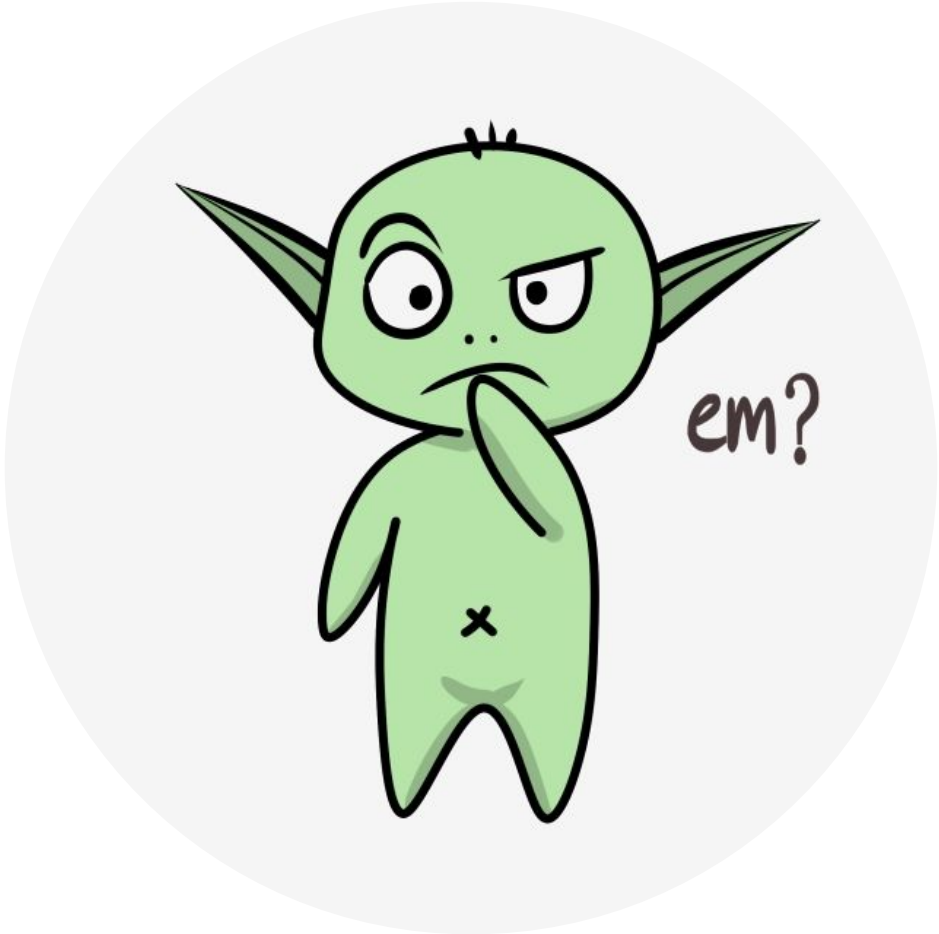
RMSProp

An overview of Deep RL algorithms



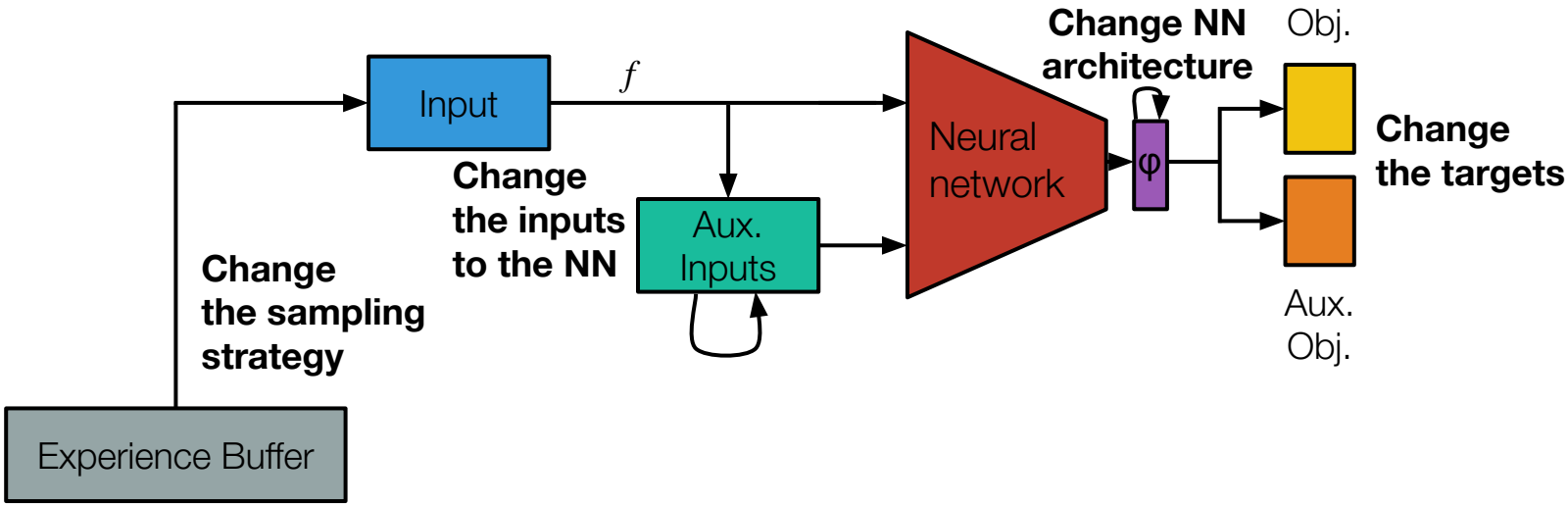
DQN


uniformly
random
Input
downsampling,
gray scale
Neural
network
φ
Obj.
clipped
(TD error)^2
w/ target net
Aux.
Inputs
stacked
frames
Obj.
Experience Buffer
⟨s, a, s', r⟩tuples
screenshots of Atari games

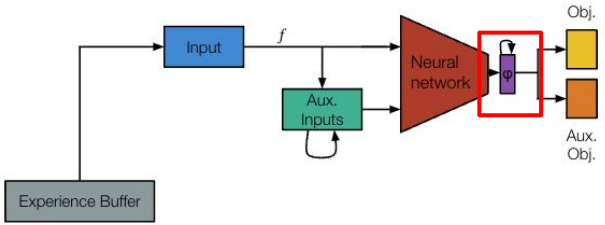


https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

An overview of Deep RL algorithms



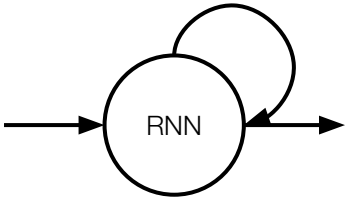
Changing the neural network architecture

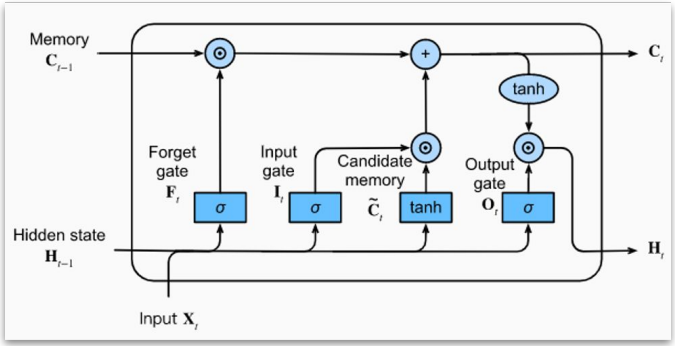


Deep Recurrent Q-Learning (DRQN)

[Hausknecht, et al. 2015]

- DQN's memory is limited to 4 timesteps and framestacking is an extra parameter.
  - The problem becomes a POMDP
- What if instead we *learned* what to remember?
  - LSTM: Long Short-Term Memory (Hochreiter and Schmidhuber 1997)

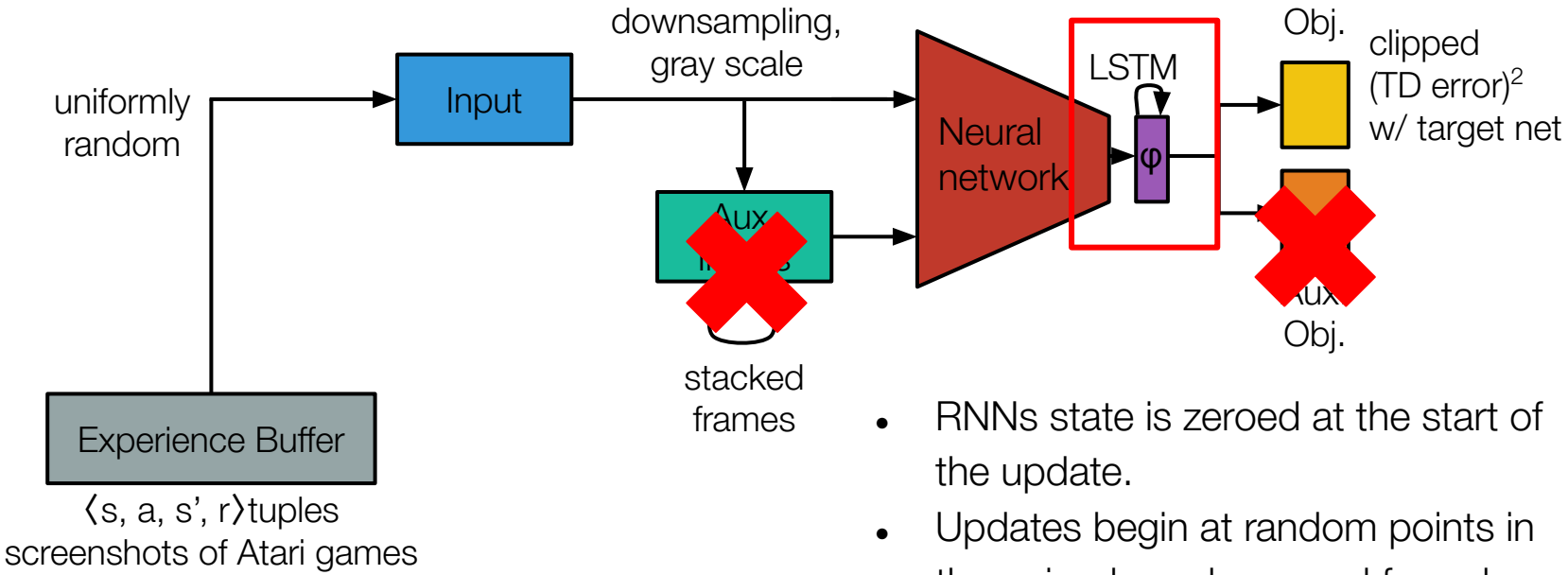




https://medium.com/@ottaviocalzone/an-intuitive-explanation-of-lstm-a035eb6ab42

Deep Recurrent Q-Learning (DRQN)

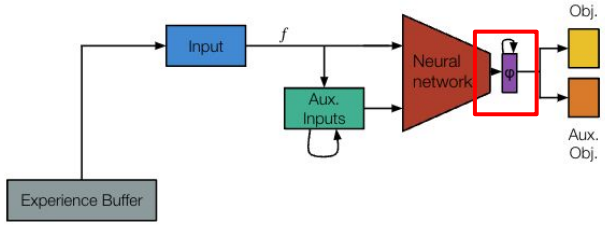
[Hausknecht, et al. 2015]



- RNNs state is zeroed at the start of the update.
- Updates begin at random points in the episode and proceed for only *x* timesteps (i.e., sequential updates).

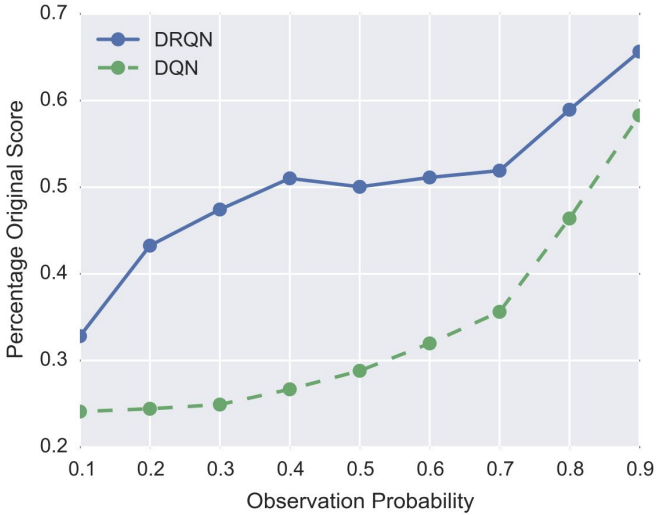
Deep Recurrent Q-Learning (DRQN)

[Hausknecht, et al. 2015]

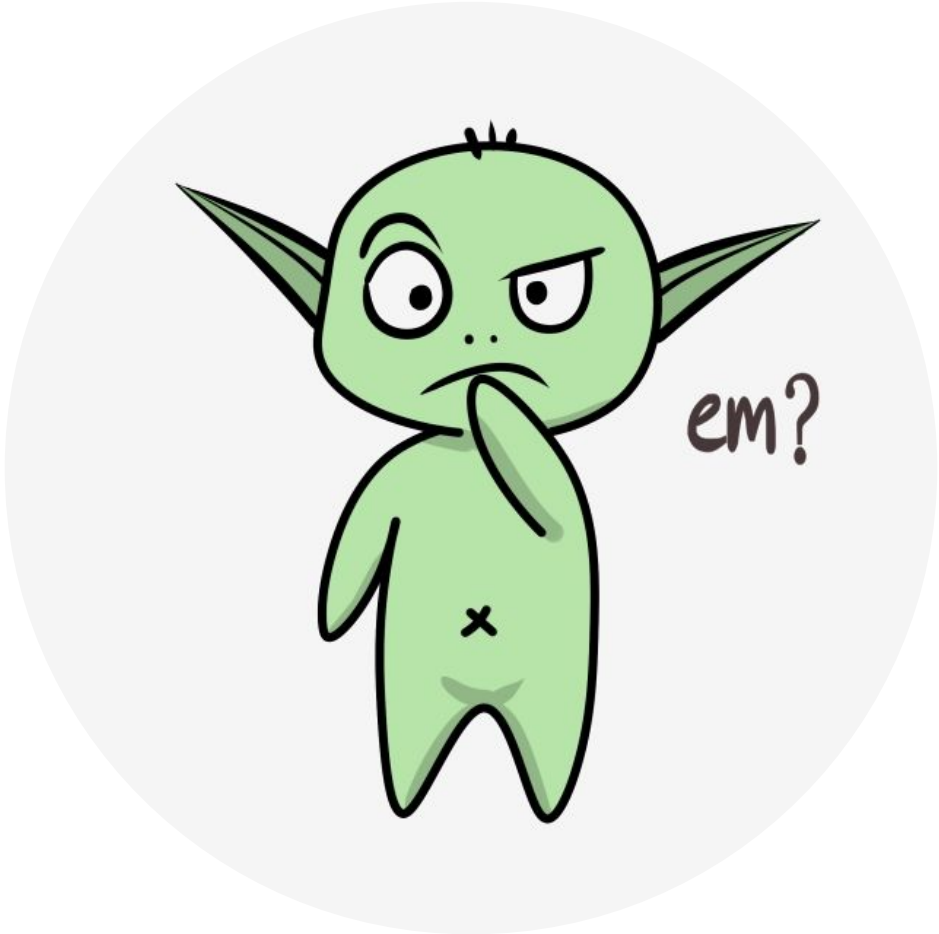


Game	DRQN $\pm std$	DQN $\pm std$ Ours	DQN $\pm std$ Mnih et al.
Asteroids	1020 (±312)	1070 (±345)	1629 (±542)
Beam Rider	3269 (±1167)	**6923** (±1027)	6846 (±1619)
Bowling	62 (±5.9)	72 (±11)	42 (±88)
Centipede	3534 (±1601)	3653 (±1903)	8309 (±5237)
Chopper Cmd	2070 (±875)	1460 (±976)	6687 (±2916)
Double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

seeds?
max over learning curve?



DRQN is said to be more robust to partial observability. Agent trained on fully-observable setting and evaluated in a flickering setting.

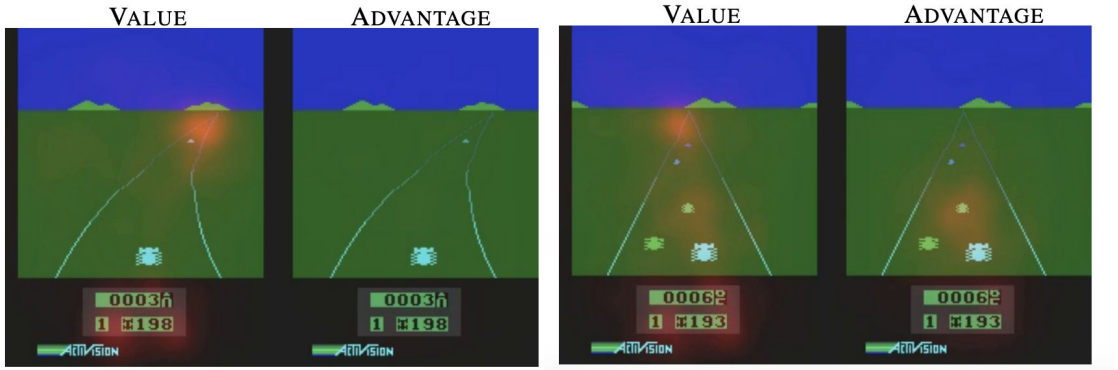


https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Duelling DQN

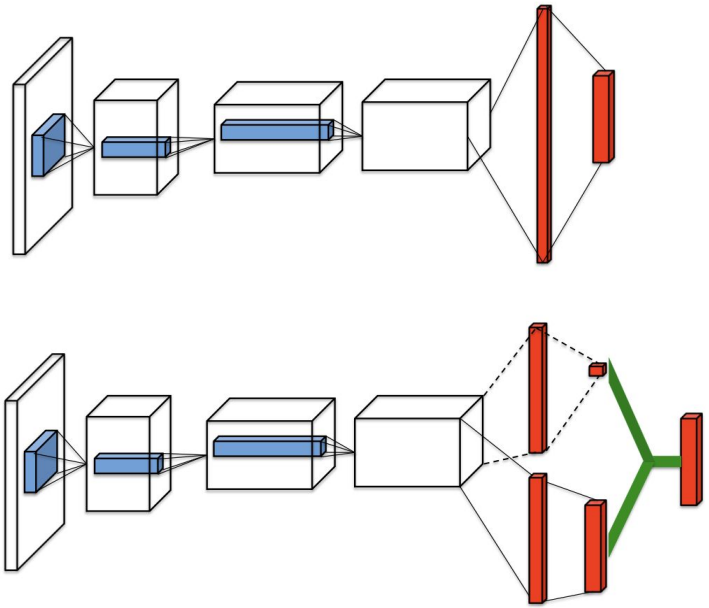
[Wang et al. 2016]

- An architecture designed with reinforcement learning in mind.
- The dueling architecture explicitly separates the representation of state values and (state-dependent) action advantages.
  - $a_\pi(s, a) = q_\pi(s, a) - v_\pi(s) \rightarrow$ Note $\mathbb{E}_\pi[a_\pi(s, a)] = 0$ because $v_\pi(s) = \mathbb{E}_\pi[q_\pi(s, a)]$.
- Key claim: for many states, it is unnecessary to estimate the value of each action choice.



Duelling DQN

[Wang et al. 2016]

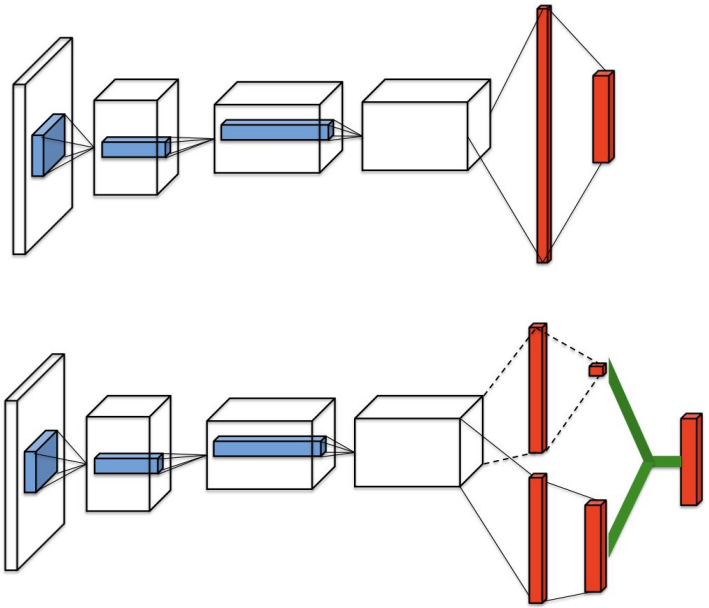


- Key innovation is how to combine the two layers.
- Let $V(s; \boldsymbol{\theta}, \boldsymbol{\beta}) \approx v_\pi(s)$, and $A(s, a; \boldsymbol{\theta}, \boldsymbol{\alpha}) \approx a_\pi(s, a)$.
- We might be tempted to do: $Q(s, a; \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = V(s; \boldsymbol{\theta}, \boldsymbol{\beta}) + A(s, a; \boldsymbol{\theta}, \boldsymbol{\alpha})$.

  But why V and A will have any of the semantics we expect them to have? This equation is actually **unidentifiable**!

Duelling DQN

[Wang et al. 2016]



- Instead, we force the adv. function to have zero adv. at the chosen action:

  $Q(s, a; \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = V(s; \boldsymbol{\theta}, \boldsymbol{\beta}) + (A(s, a; \boldsymbol{\theta}, \boldsymbol{\alpha}) - \max_{a'} A(s', a; \boldsymbol{\theta}, \boldsymbol{\alpha}))$.

- For the optimal action we end up with

  $Q(s, a^*; \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = V(s; \boldsymbol{\theta}, \boldsymbol{\beta})$.

  Now $V(s; \boldsymbol{\theta}, \boldsymbol{\beta})$ provides an estimate of the VF, while the other stream produces an estimate of the adv. function.

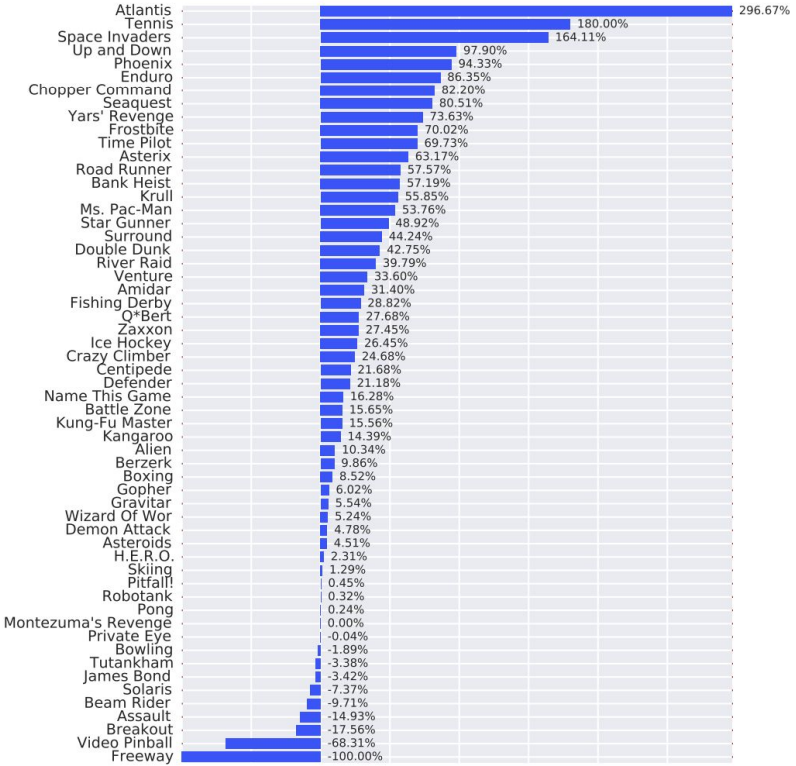
Duelling DQN

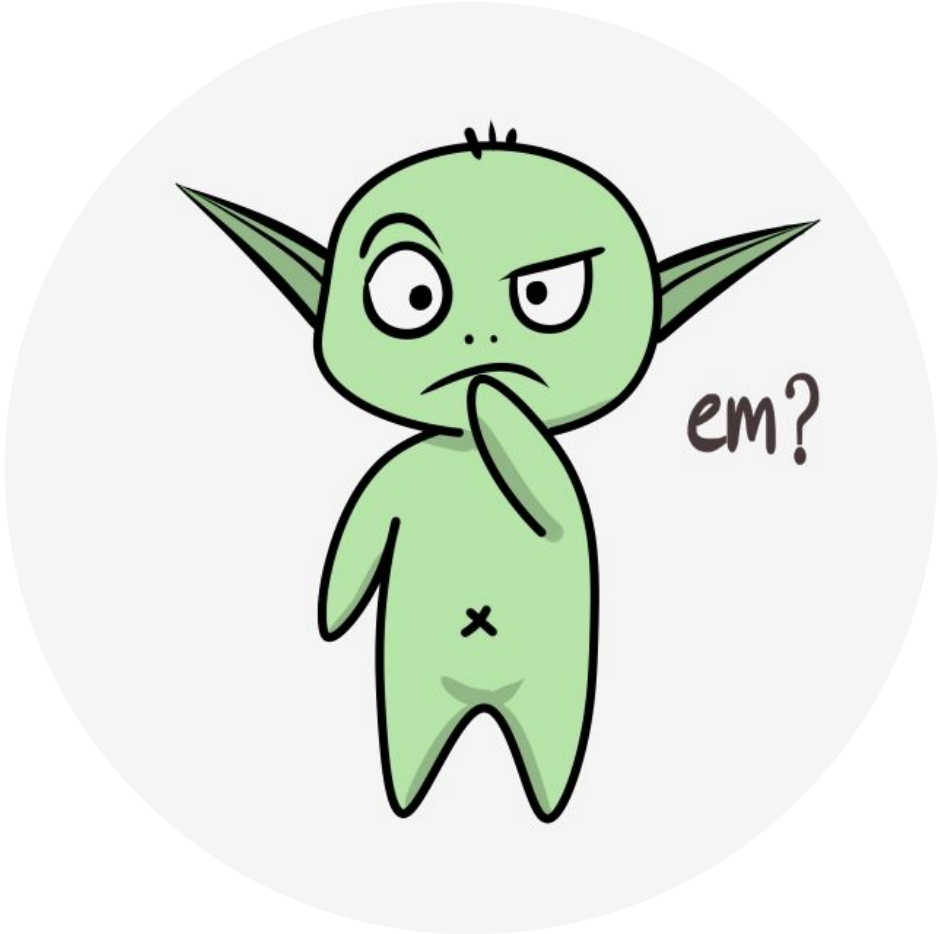
[Wang et al. 2016]

- Alternatively, one can use the average instead of the max, you lose semantics but you gain stability.

$$Q(s, a; \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = V(s; \boldsymbol{\theta}, \boldsymbol{\beta}) + \left(A(s, a; \boldsymbol{\theta}, \boldsymbol{\alpha}) - 1/|A| \sum_{a'} A(s', a; \boldsymbol{\theta}, \boldsymbol{\alpha})\right).$$

- “We follow closely the setup of van Hasselt et al. (2015) and compare to their results using single-stream Q-networks.”

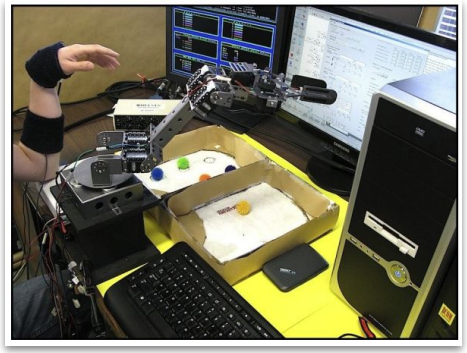




https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

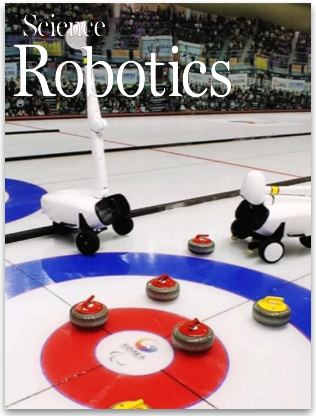
Thinking carefully about your inputs

A common thread across many successes in the “real-world”



[*Pilarski et al. 2012]

[Mnih et al. 2020]

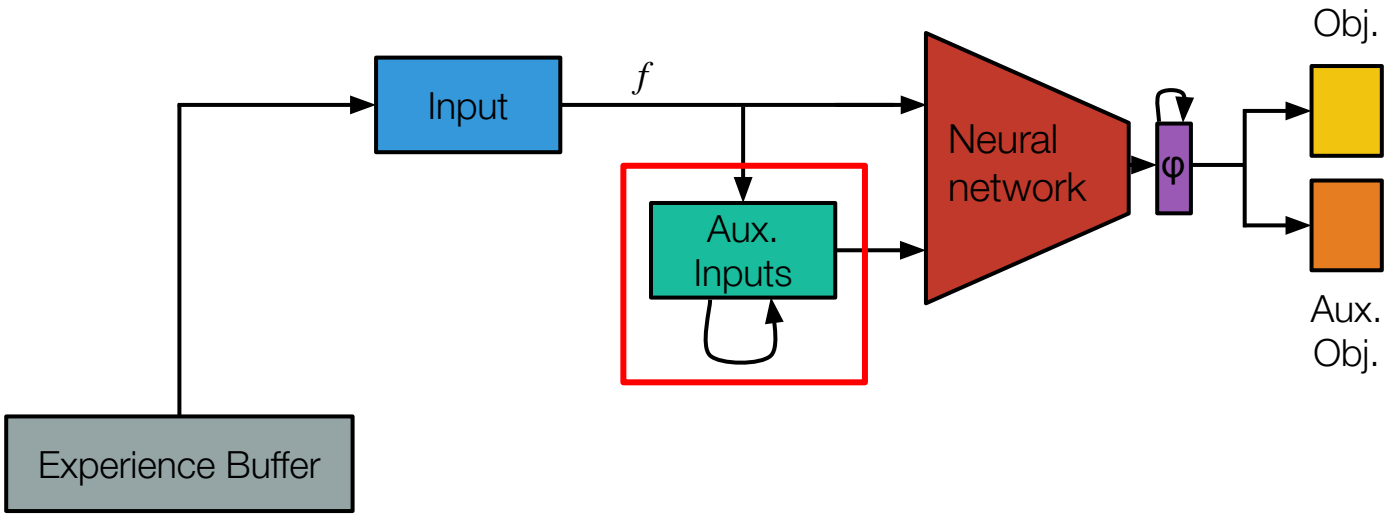


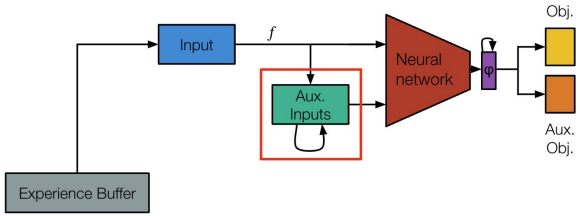
[Won et al. 2020]

[Bellemare et al. 2020]

Auxiliary Inputs

[Tao et al. 2023]





Auxiliary Inputs

[Tao et al. 2023]

- Traditional agent-state function:

$$\boldsymbol{x}_{t+1} \doteq u_\phi(\boldsymbol{x}_t, a_t, \boldsymbol{o}_{t+1})$$

- Auxiliary input: $\boldsymbol{M} : \mathcal{T} \to \mathbb{R}^N$, which acts as a summarizing function.

$$\boldsymbol{x}_{t+1} \doteq u_{\boldsymbol{\phi}}(\boldsymbol{x}_t, a_t, \boldsymbol{o}_{t+1}, \boldsymbol{M}_{t+1}) \in \mathbb{R}^k$$

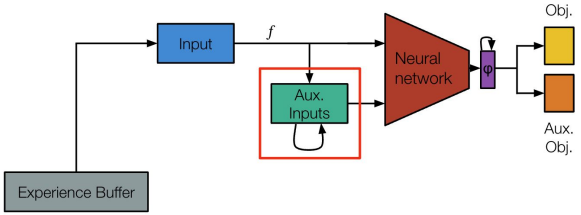
- Auxiliary inputs can summarize the past and the future:

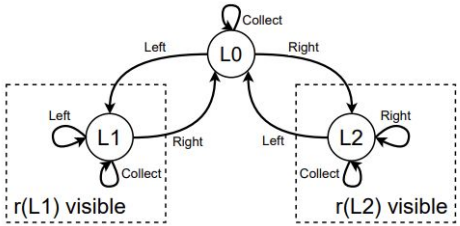
$$\begin{aligned}\boldsymbol{M}_t &\doteq \boldsymbol{M}(\{\boldsymbol{o}_0, a_0, ..., \boldsymbol{o}_t, a_t, \boldsymbol{O}_{t+1}, \ldots, \boldsymbol{O}_L\}) \\ &= \boldsymbol{M}_h(\{\boldsymbol{o}_0, a_0, ..., \boldsymbol{o}_t, a_t\}) + \mathbb{E}_{\boldsymbol{M}_f}\left[\{\boldsymbol{O}_{t+1}, \ldots, \boldsymbol{O}_L\}\right].\end{aligned}$$

- Auxiliary inputs can be used to discriminate between observations that would otherwise be aliased, leading to more expressive features that smoothly interpolate between different states.

Auxiliary Inputs

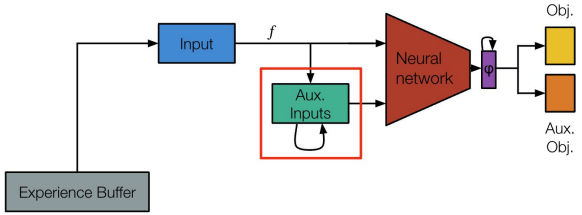
[Tao et al. 2023]

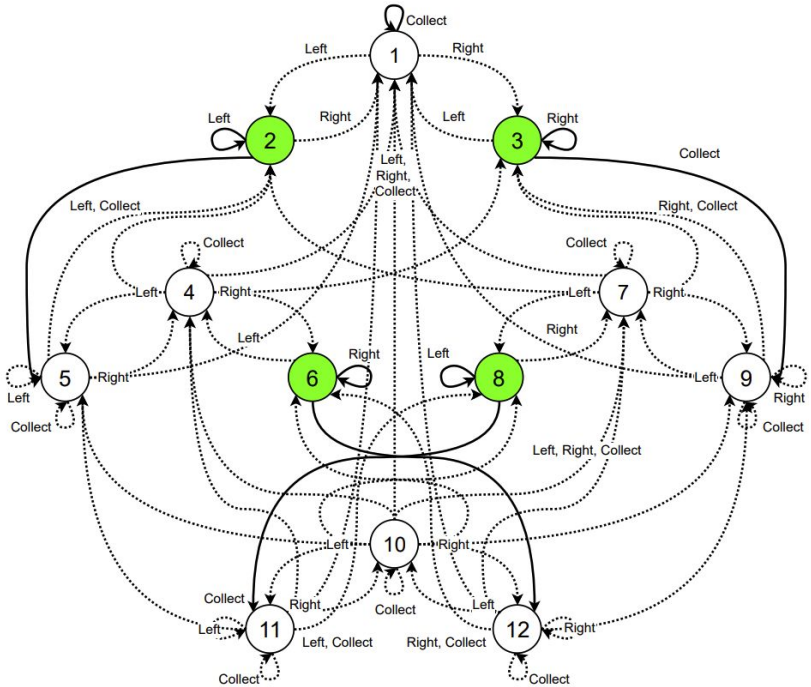




Auxiliary Inputs

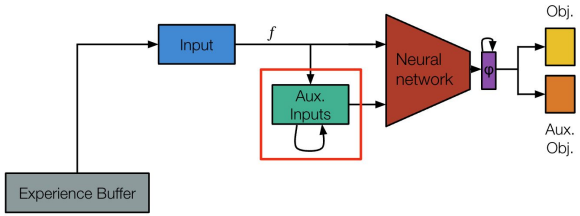
[Tao et al. 2023]

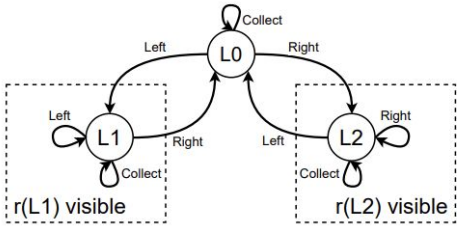




Auxiliary Inputs

[Tao et al. 2023]

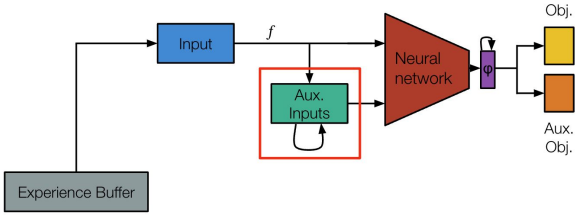


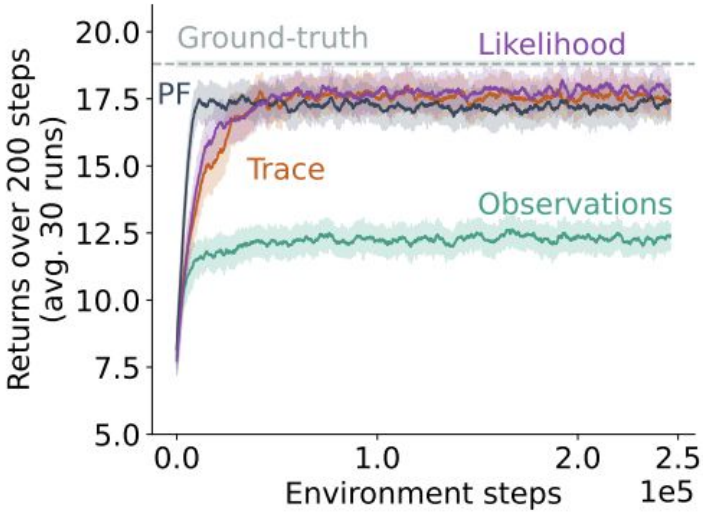


$$\boldsymbol{o}_t \doteq \begin{bmatrix} 0.\ \text{Is the agent in location 0?} \\ 1.\ \text{Is the agent in location 1?} \\ 2.\ \text{Is the agent in location 2?} \\ 3.\ \text{Is the reward in location 1 observable and missing?} \\ 4.\ \text{Is the reward in location 1 observable and present?} \\ 5.\ \text{Is the reward in location 1 unobservable?} \\ 6.\ \text{Is the reward in location 2 observable and missing?} \\ 7.\ \text{Is the reward in location 2 observable and present?} \\ 8.\ \text{Is the reward in location 2 unobservable?} \end{bmatrix}$$

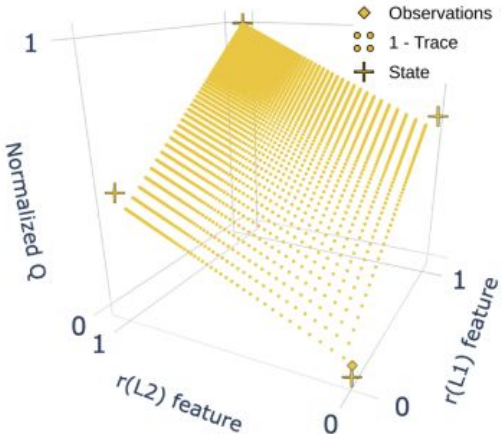
Auxiliary Inputs

[Tao et al. 2023]

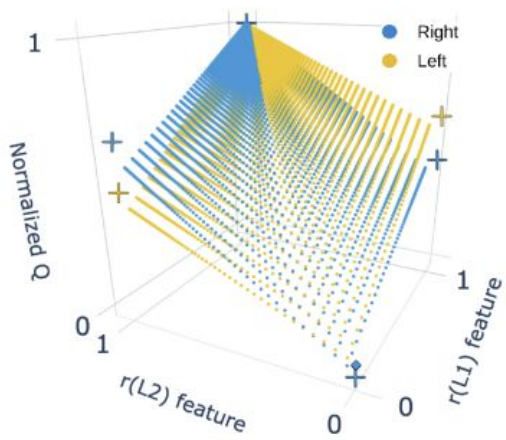




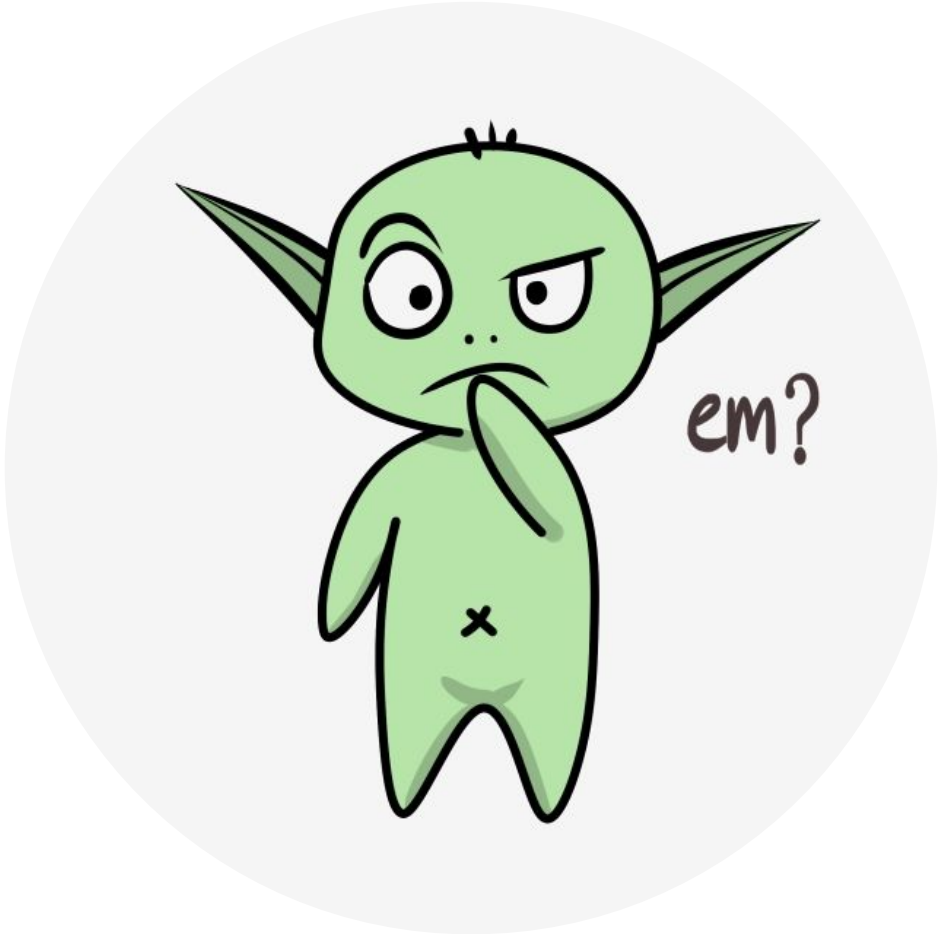
(a)



(b)



(c)

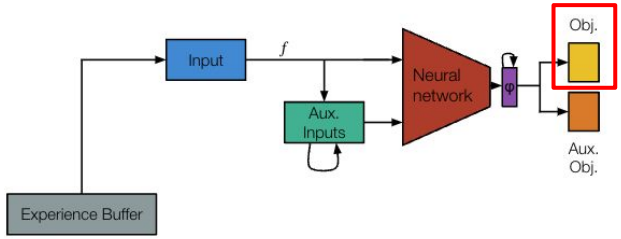


https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Changing the target function

Double DQN

[van Hasselt et al. 2016]

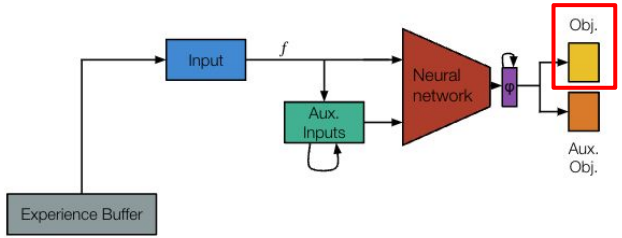


Double Learning

- The issue is that we use the same samples to determine the maximizing action and to estimate its value.
- In Bandits:
  - Split the data, learn $Q_1(a)$ and $Q_2(a)$ to estimate $q(a)$.
  - Choose actions according to one estimate and get estimate from the other:
    $A^* = \text{argmax}_a\, Q_1(a)$ $\qquad$ $Q_2(A^*) = Q_2(\text{argmax}_a\, Q_1(a))$
  - This leads to unbiased estimates, that is: $\mathbb{E}[Q_2(A^*)] = q(A^*)$

$$Q_1(S_t, A_t) \leftarrow Q_1(S_t, A_t) + \alpha \left[R_{t+1} + \gamma Q_2\left(S_{t+1}, \arg\max_a Q_1(S_{t+1}, a)\right) - Q_1(S_t, A_t) \right]$$

Marlos C. Machado



Double DQN

[van Hasselt et al. 2016]

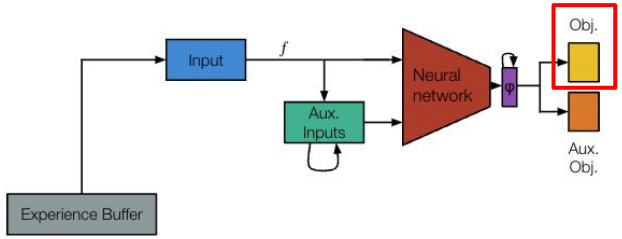
- Instead of minimizing

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

Double DQN instead minimizes

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

Instead of explicitly maintaining a second value function, DDQN proposes to evaluate the greedy policy according to the online network, but using the *target* network to estimate its value. No need for an additional network.



Double DQN

[van Hasselt et al. 2016]

- Instead of minimizing

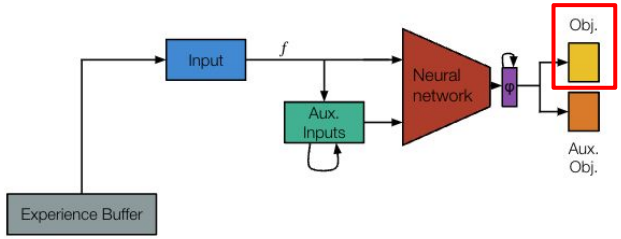
$$\mathcal{L}^{\text{DQN}} = \mathbb{E}_{(s,a,r,s')\sim U(\mathcal{D})}\left[\left(R_{t+1} + \gamma \max_{a'\in\mathcal{A}} Q(S_{t+1}, a'; \boldsymbol{\theta}^-) - Q(S_t, A_t, \boldsymbol{\theta}_t)\right)^2\right]$$

  Double DQN instead minimizes

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

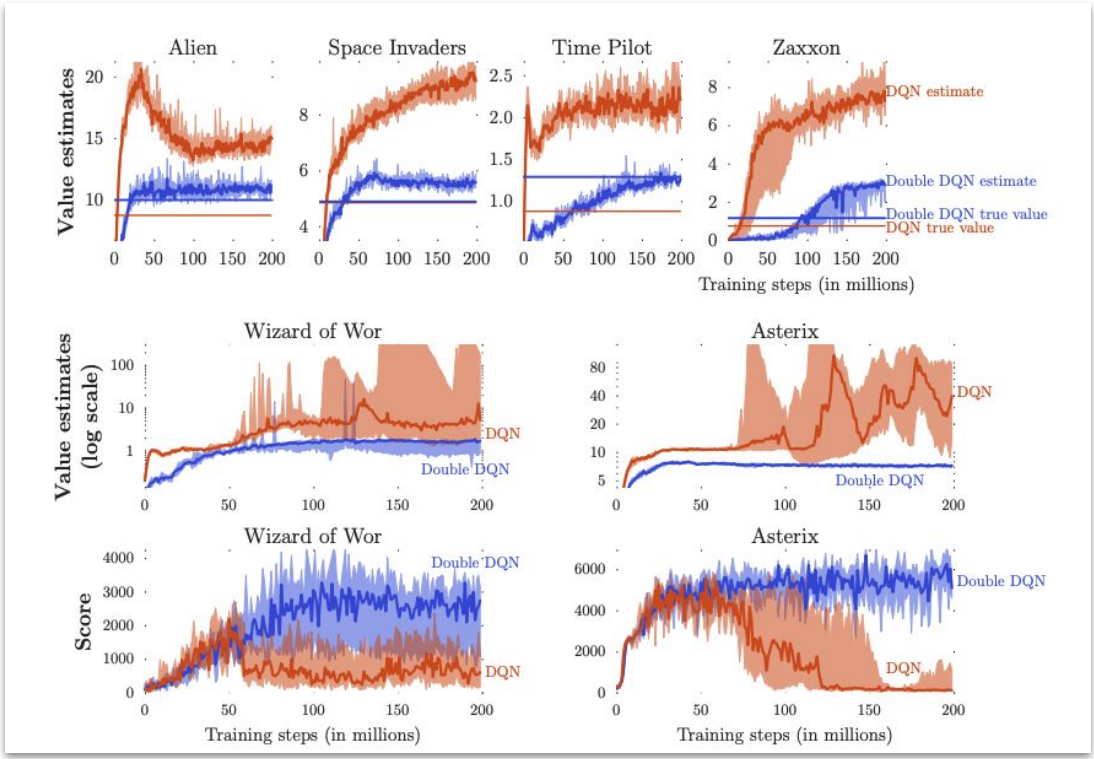
- Notice, there's no coinflip.

Instead of explicitly maintaining a second value function, DDQN proposes to evaluate the greedy policy according to the online network, but using the *target* network to estimate its value. No need for an additional network.



Double DQN

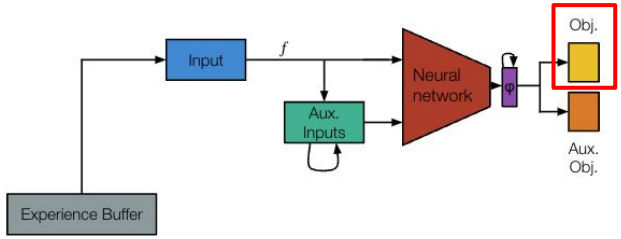
[van Hasselt et al. 2016]

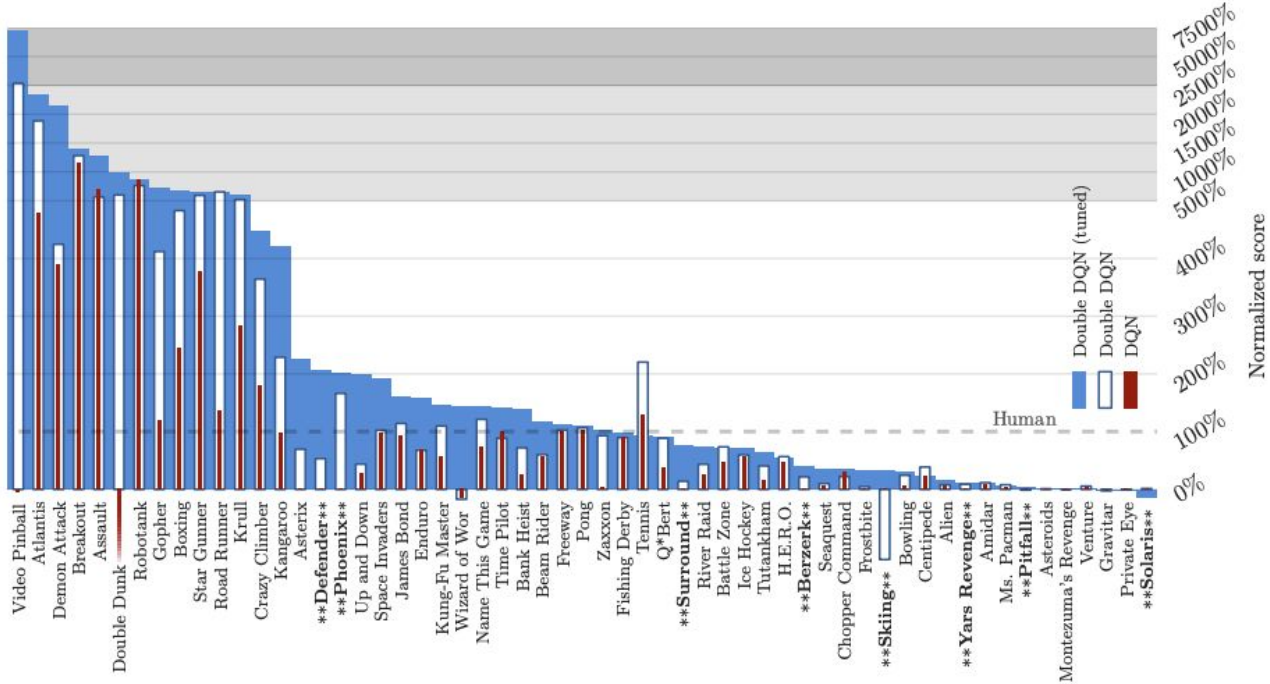


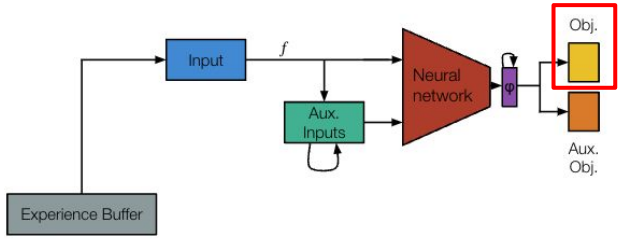
- 6 random seeds
- Darker line: median
  Shaded area: 10% and 90% quantiles with linear interp.
- Straight lines are the MC estimates of the returns after learning.

Double DQN

[van Hasselt et al. 2016]







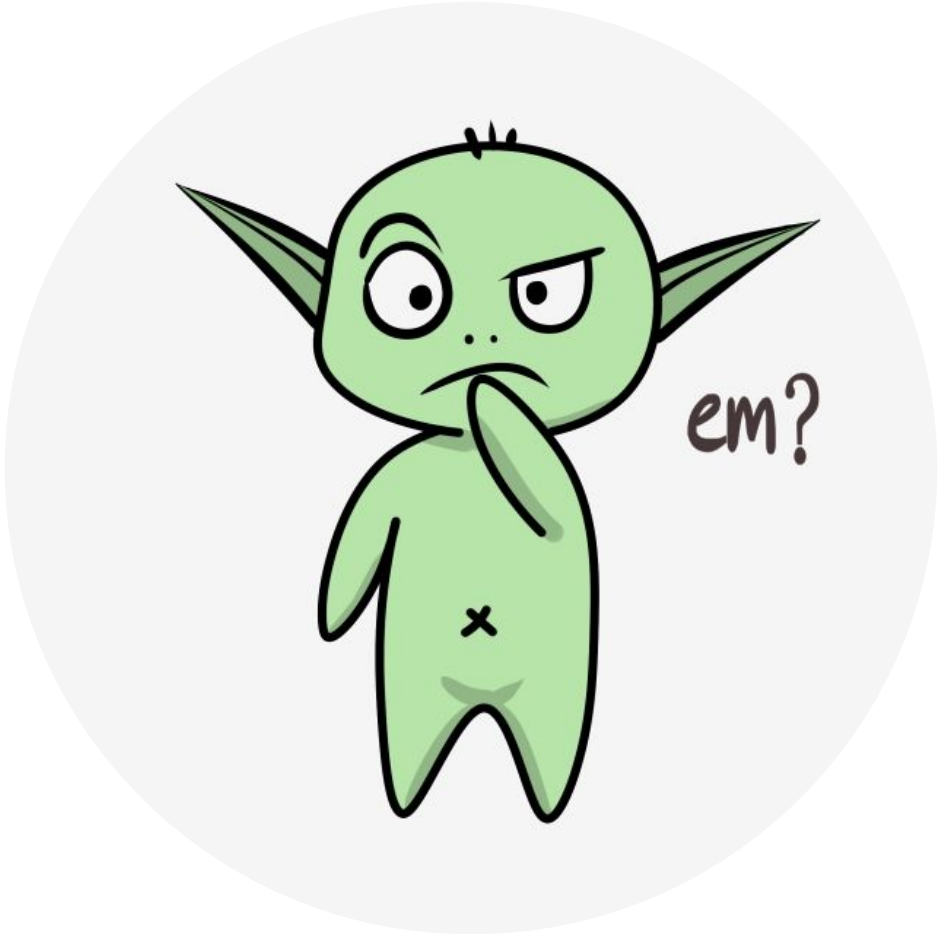
Double DQN

[van Hasselt et al. 2016]

seeds?
some measure of spread?
max over learning curve?

Game	Random	Human	DQN	Double DQN
Alien	227.80	6875.40	3069.33	2907.30
Amidar	5.80	1675.80	739.50	702.10
Assault	222.40	1496.40	3358.63	5022.90
Asterix	210.00	8503.30	6011.67	15150.00
Asteroids	719.10	13156.70	1629.33	930.60
Atlantis	12850.00	29028.10	85950.00	64758.00
Bank Heist	14.20	734.40	429.67	728.30
Battle Zone	2360.00	37800.00	26300.00	25730.00
Beam Rider	363.90	5774.70	6845.93	7654.00
Bowling	23.10	154.80	42.40	70.50
Boxing	0.10	4.30	71.83	81.70
Breakout	1.70	31.80	401.20	375.00

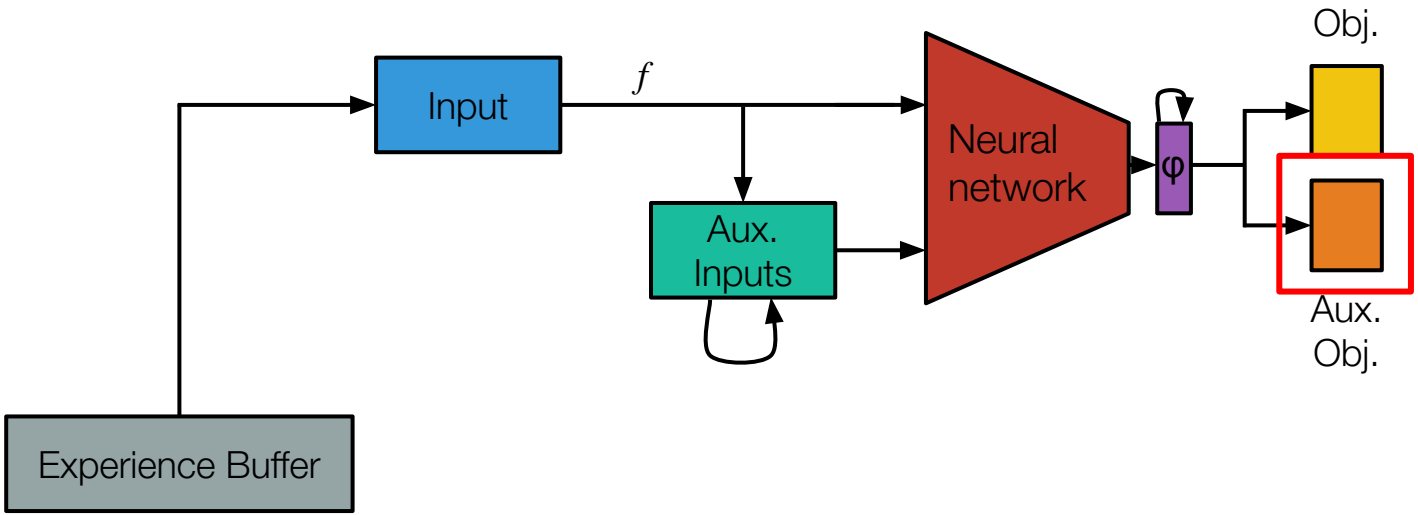
Table 3: Raw scores for the no-op evaluation condition (5 minutes emulator time). DQN as given by Mnih et al. (2015).



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

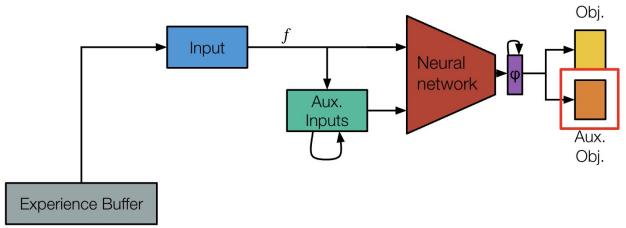
Auxiliary Tasks

[Jaderberg et al. 2016]



Auxiliary Tasks

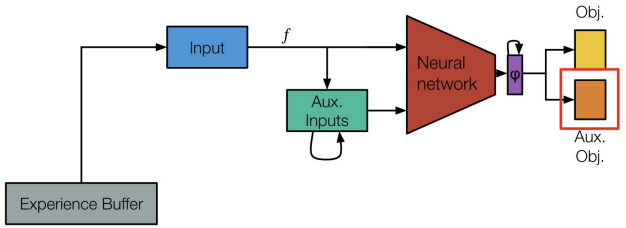
[Jaderberg et al. 2016]



- UNsupervised REinforcement and Auxiliary Learning (UNREAL).
- Idea: To improve the learned representation by taking into consideration other aspects of the env. that, maybe, are not directly associated with rewards.
  - Inspired by unsupervised learning.
- What should one learn about in the “absence” of rewards?
- Very related to ideas like predictive state representations (Littman et al. 2001) and general value functions (Sutton et al. 2011).
- Agent has a convolutional neural network and LSTM as backbone.

Auxiliary Tasks

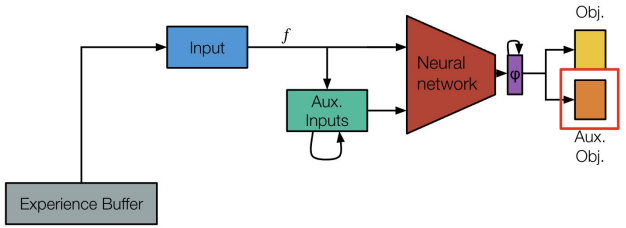
[Jaderberg et al. 2016]



$$\mathcal{L}^{\text{UNREAL}} = \mathcal{L}^{\text{DQN}} + \beta_c \sum_c \mathcal{L}_c^{\text{UNREAL}}$$

Auxiliary Tasks

[Jaderberg et al. 2016]

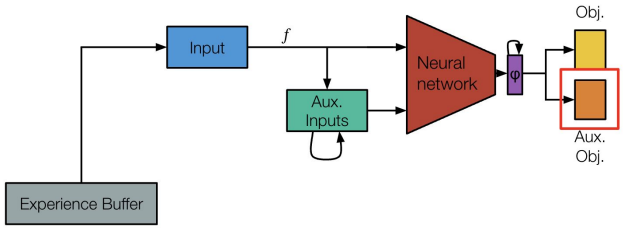


$$\mathcal{L}^{\text{UNREAL}} = \mathcal{L}^{\text{DQN}} + \beta_c \sum_c \mathcal{L}_c^{\text{UNREAL}}$$

- What auxiliary tasks should one consider?

Auxiliary Tasks

[Jaderberg et al. 2016]

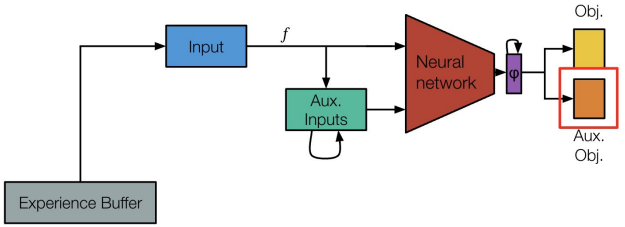


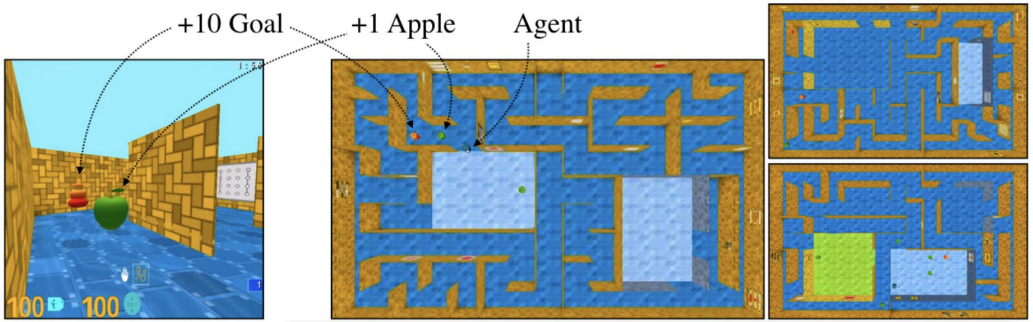
$$\mathcal{L}^{\text{UNREAL}} = \mathcal{L}^{\text{DQN}} + \beta_c \sum_c \mathcal{L}_c^{\text{UNREAL}}$$

- What auxiliary tasks should one consider?
  - Pixel changes
  - Network features
  - Reward prediction

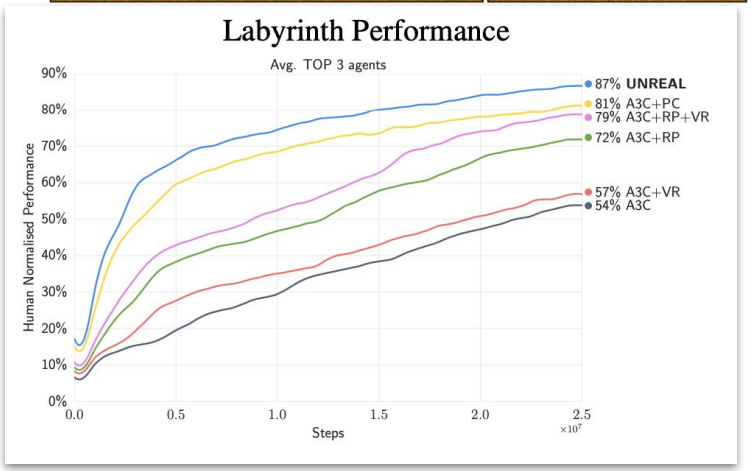
Auxiliary Tasks

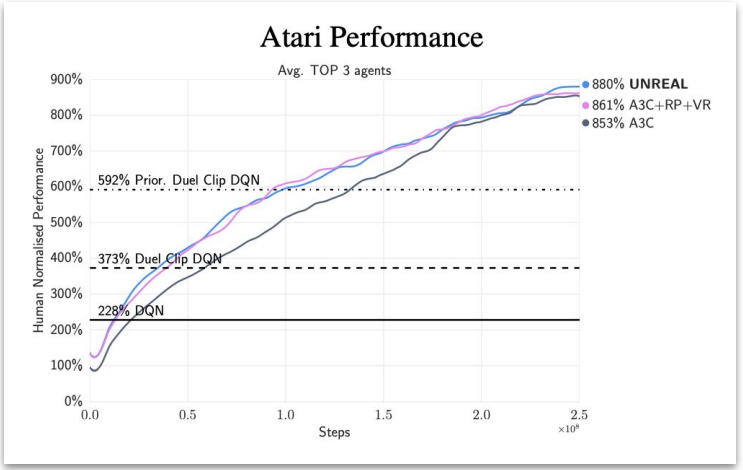
[Jaderberg et al. 2016]





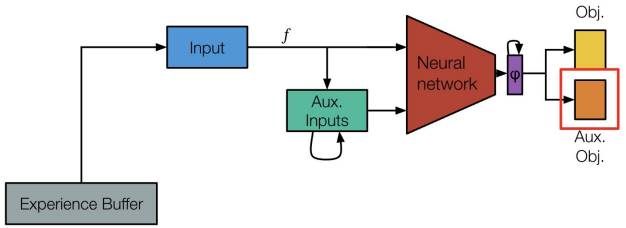
Agent Input



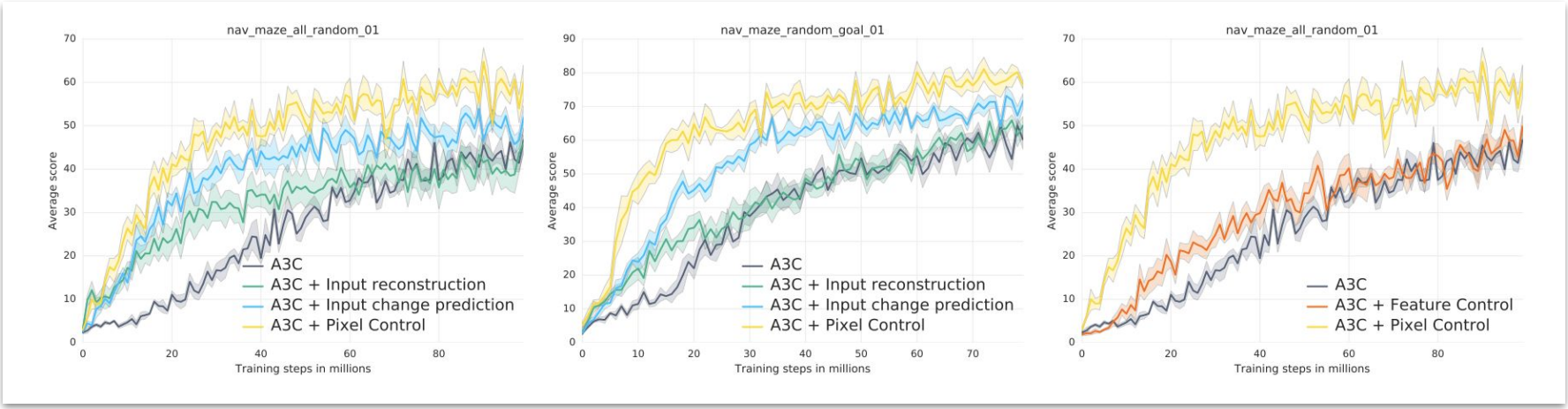


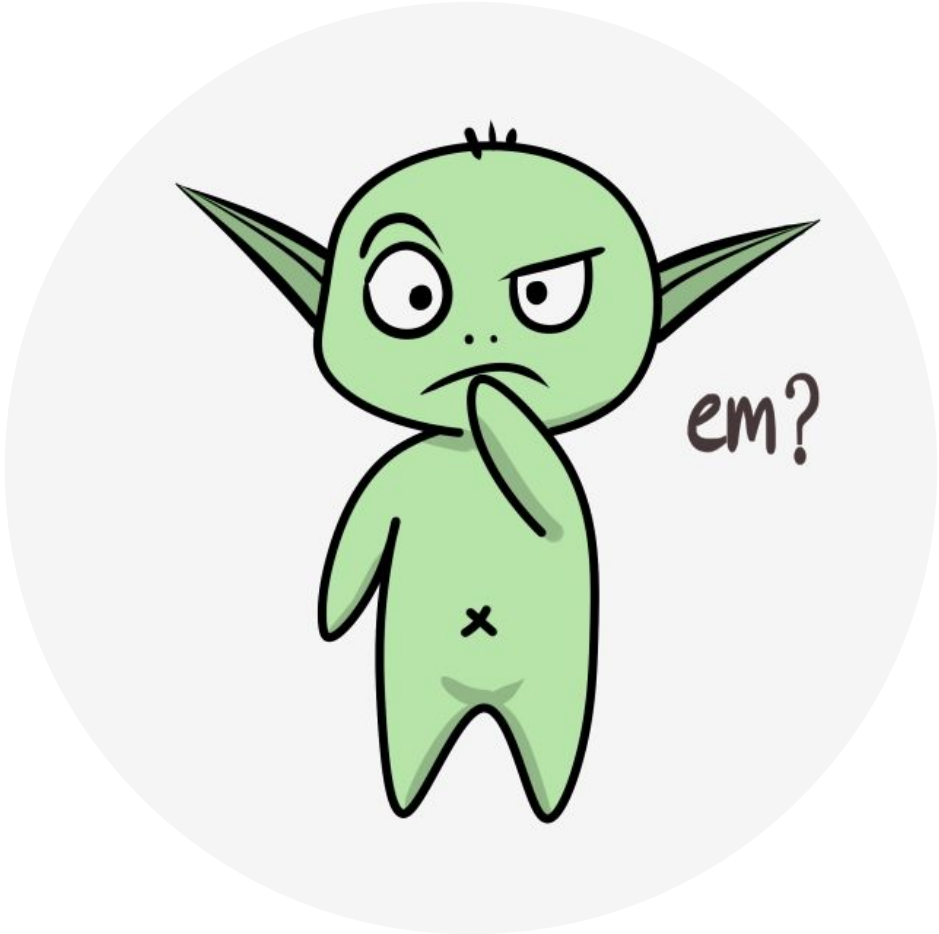
Auxiliary Tasks

[Jaderberg et al. 2016]

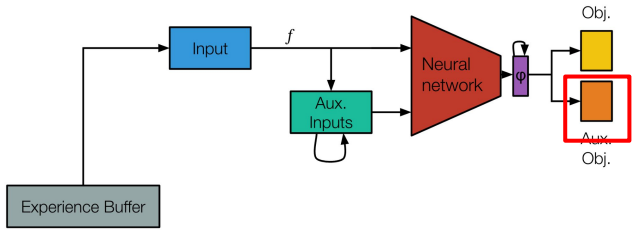


- An ablation:





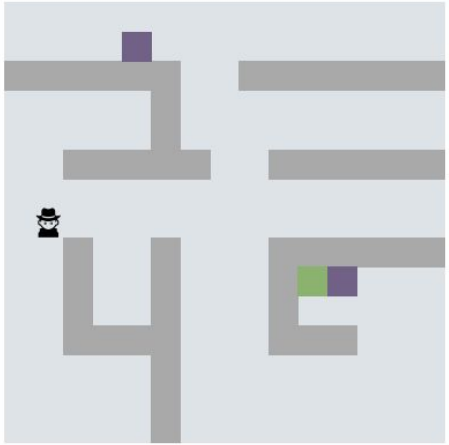
https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht



What are the best auxiliary tasks?

[Wang et al. 2022]

- What are the properties of good representations?

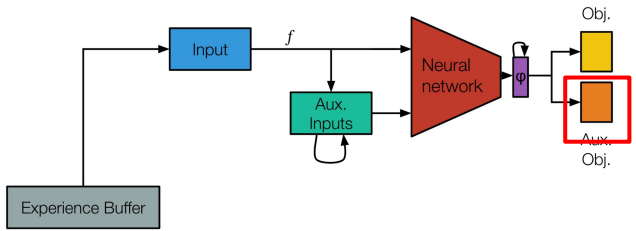


Setting:

Transfer Learning

Auxiliary tasks:

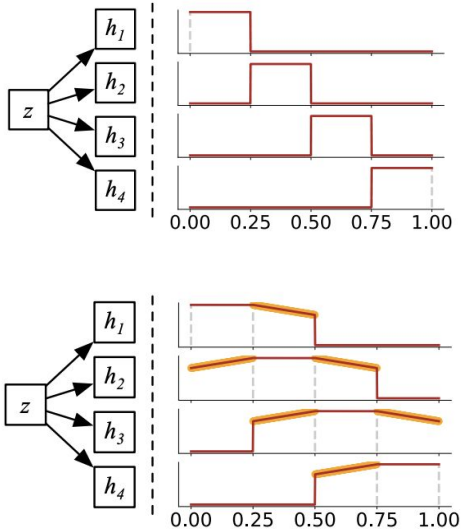
- Input Reconstruction
- Next Agent State Prediction (NAS)
- Successor Feature Prediction
- Reward Prediction
- Expert Target Prediction (XY)
- Virtual Value Function Learning
- Augmented Temporal Contrast

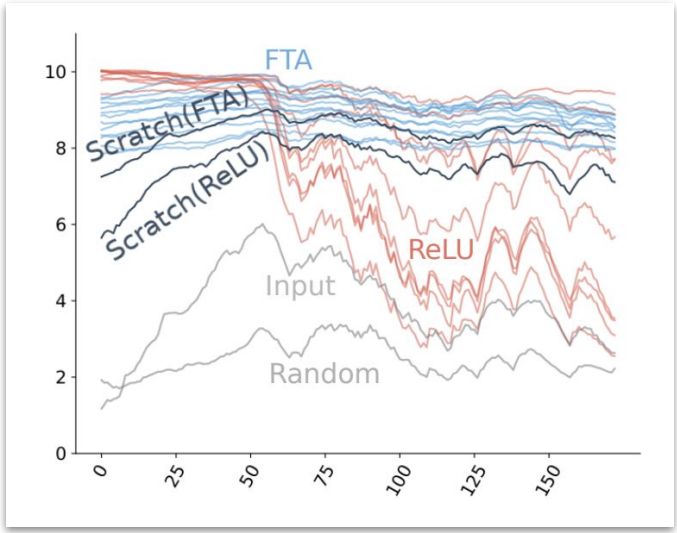


What are the best auxiliary tasks?

[Wang et al. 2022]

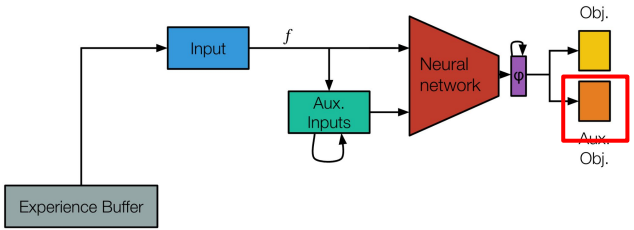
- Base agent: DQN with Adam
- Two types of activation function: ReLU and FTA (Pan et al. 2021).

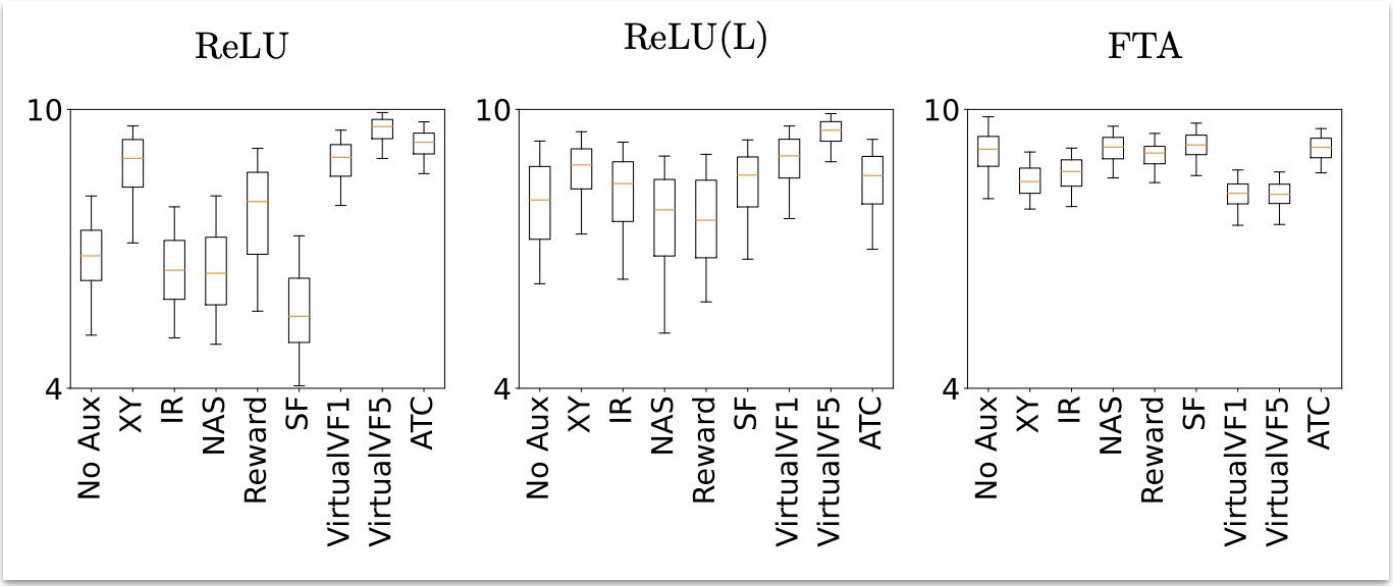




What are the best auxiliary tasks?

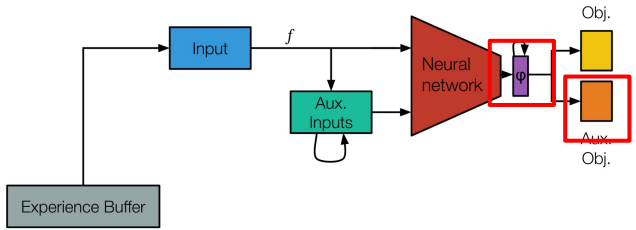
[Wang et al. 2022]

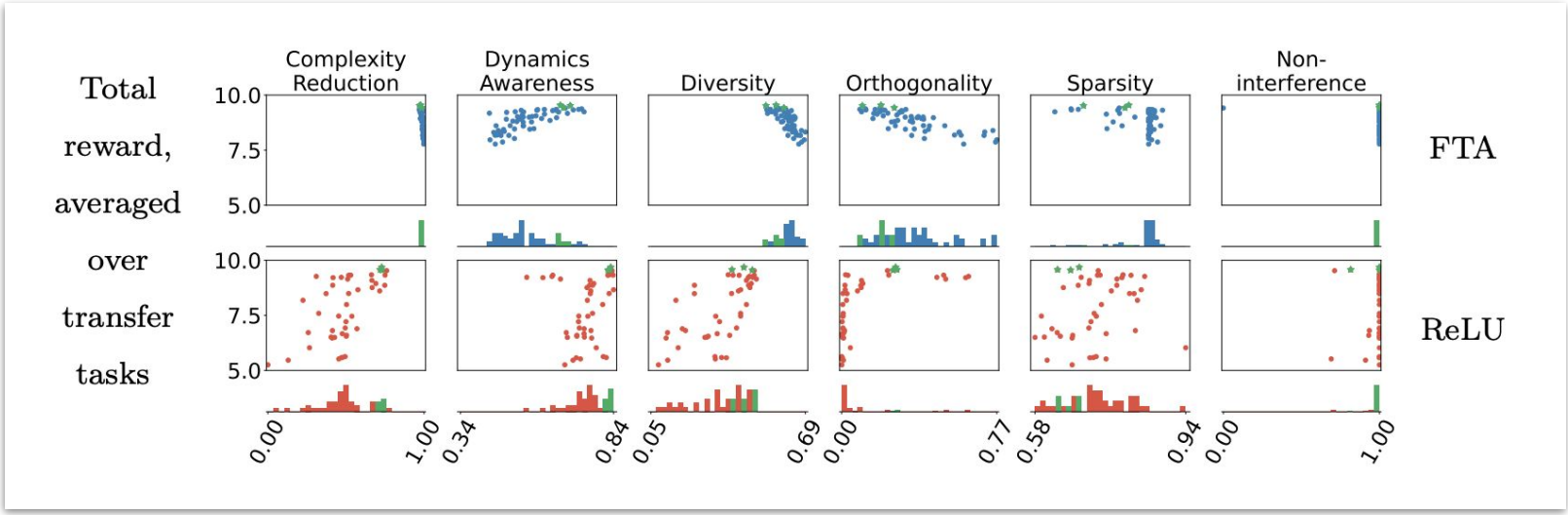


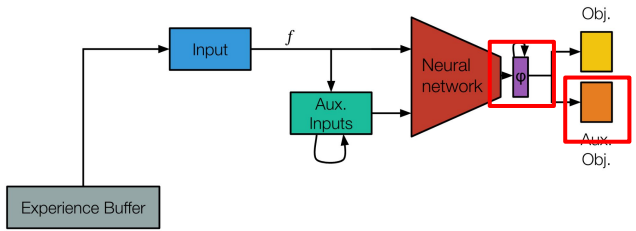


What are the the properties of good representations?

[Wang et al. 2022]

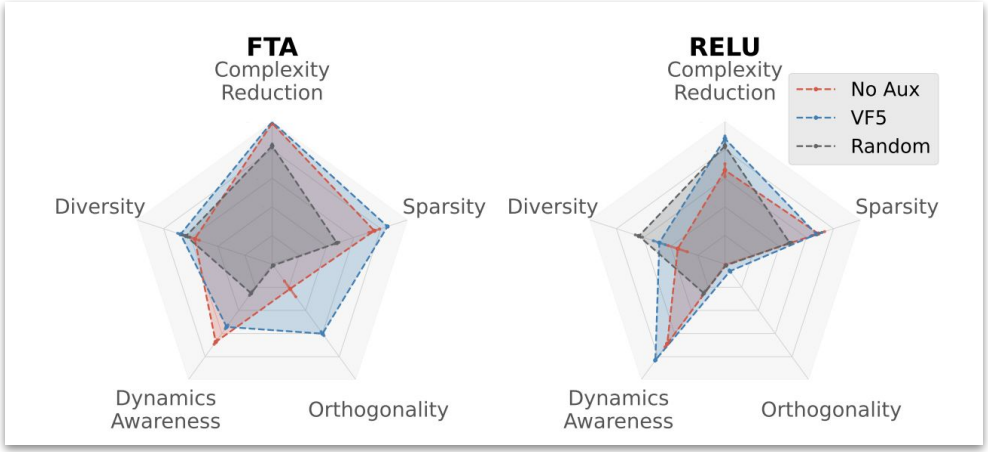






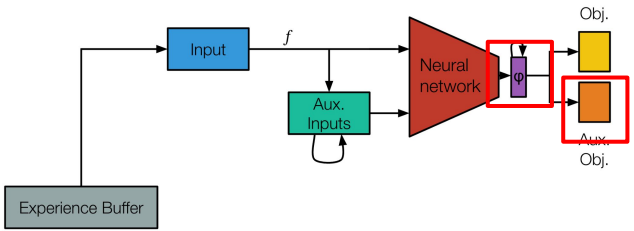
What are the the properties of good representations?

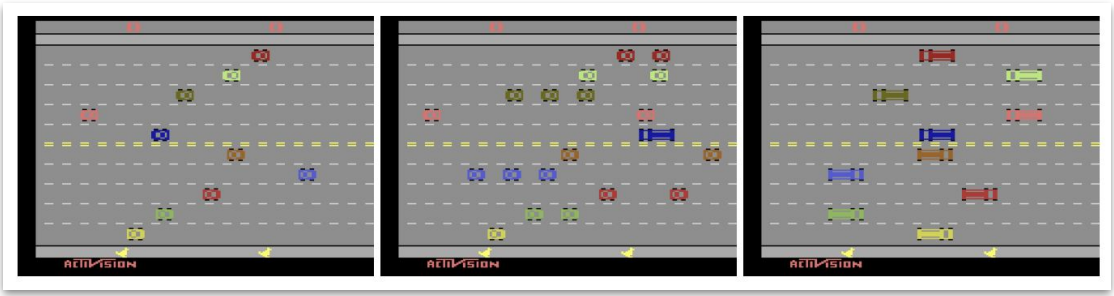
[Wang et al. 2022]

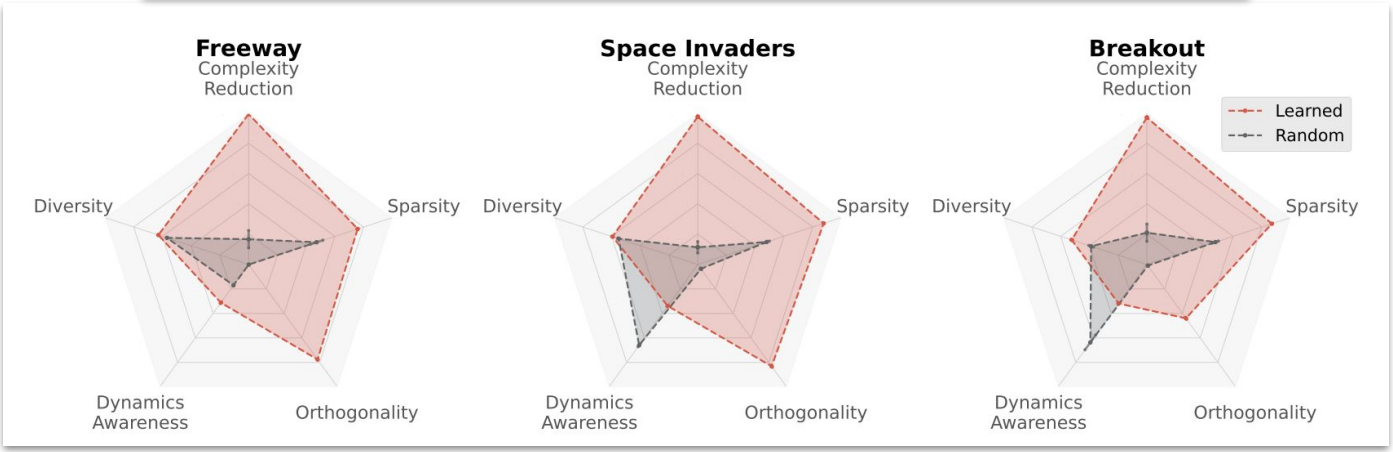


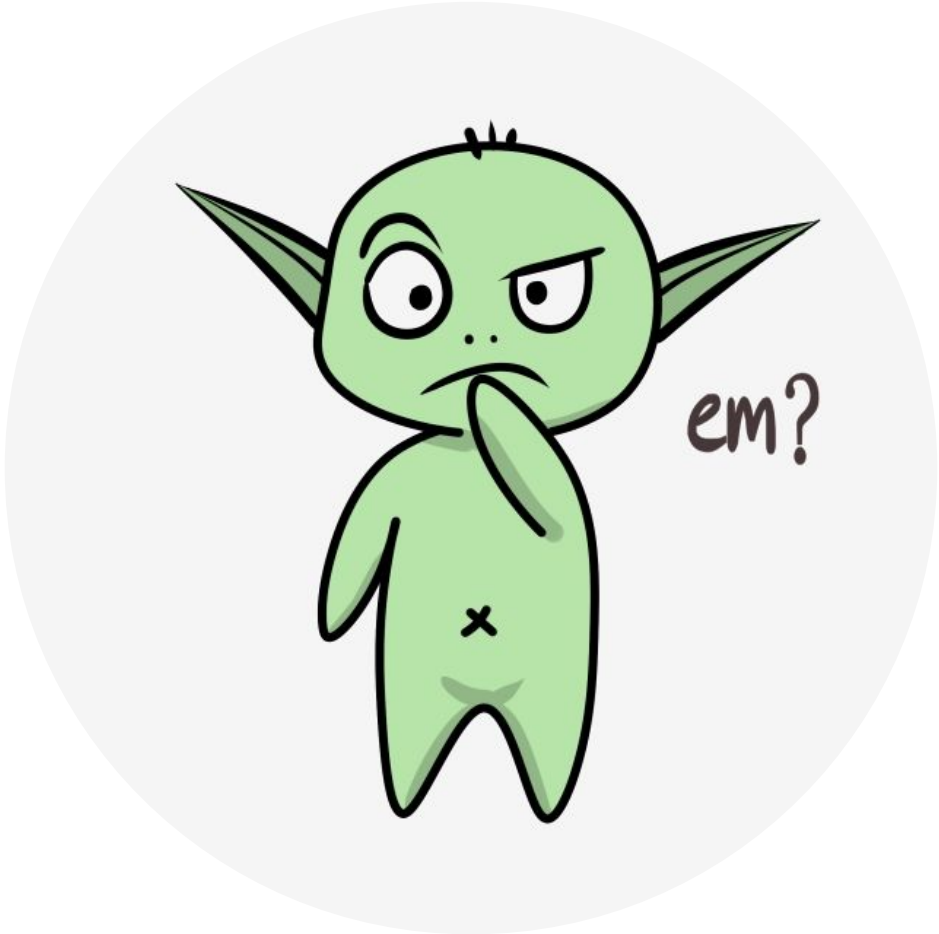
What are the the properties of good representations?

[Wang et al. 2022]









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

Changing the experience replay buffer

Prioritized Experience Replay

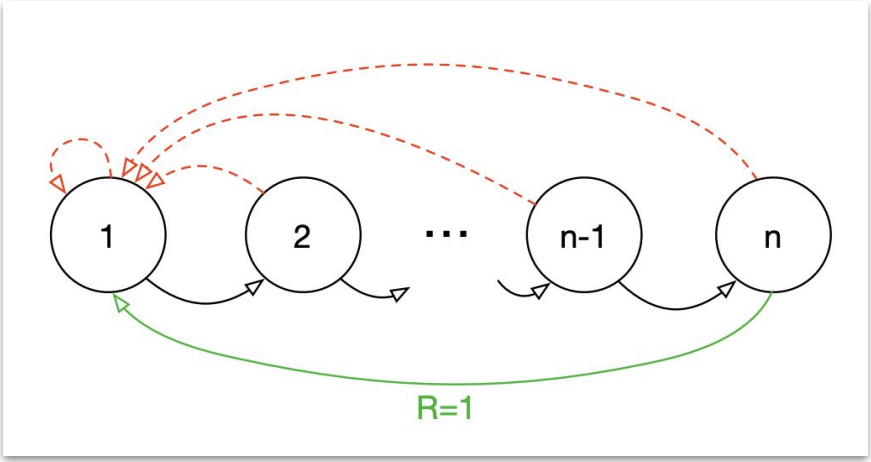
[Schaul et al. 2015]

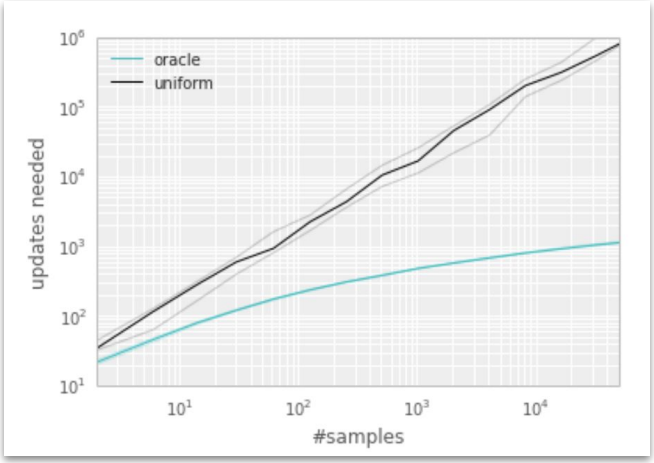
- What if we didn't sample from the experience replay buffer uniformly? What if instead we *prioritized* transitions according to their importance?

Prioritized Experience Replay

[Schaul et al. 2015]

- What if we didn't sample from the experience replay buffer uniformly?
  What if instead we *prioritized* transitions according to their importance?
- Example:

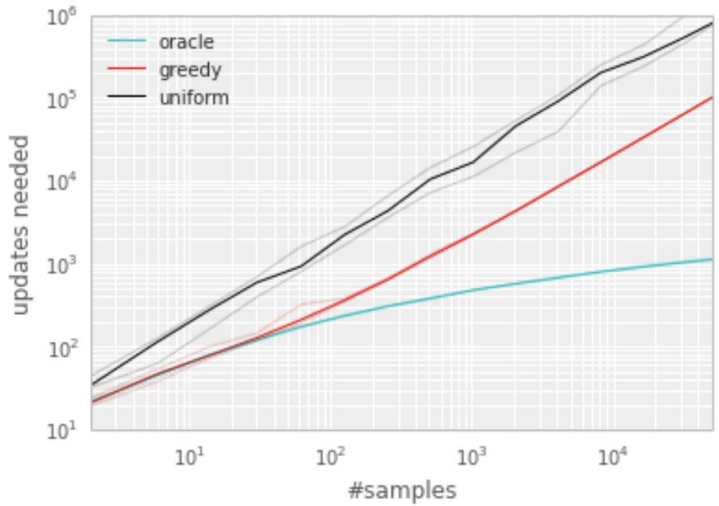




Prioritized Experience Replay

[Schaul et al. 2015]

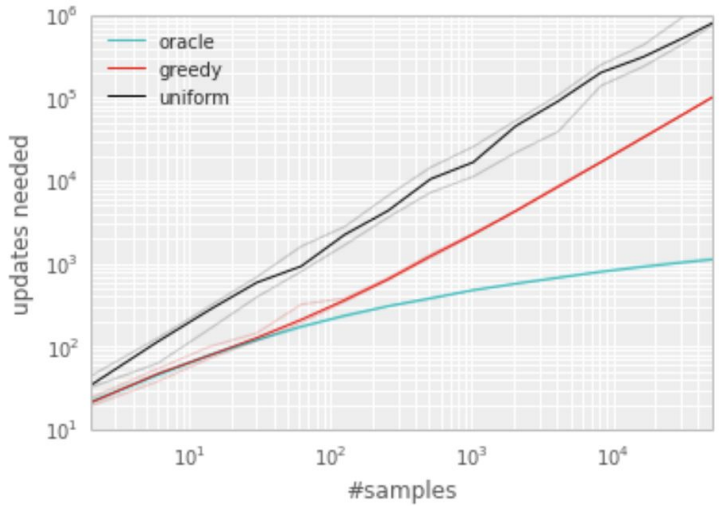
- What about the magnitude of a transition's TD error?

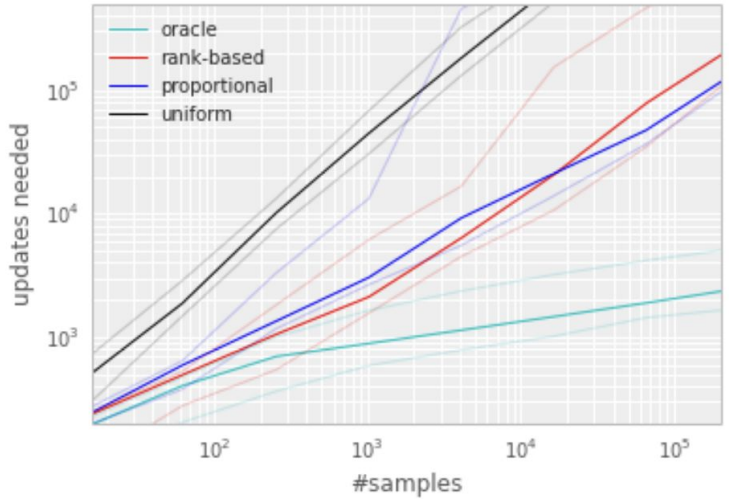


Prioritized Experience Replay

[Schaul et al. 2015]

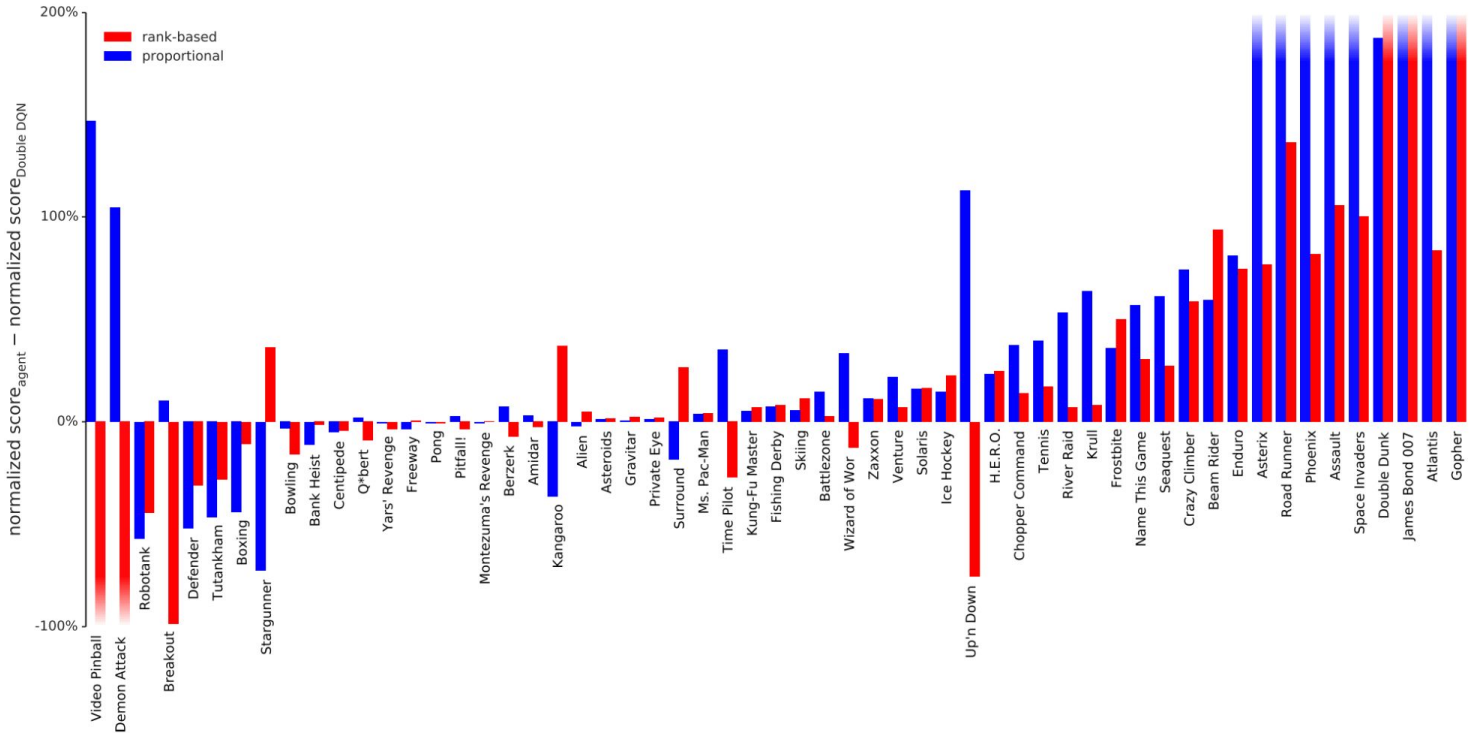
- What about the magnitude of a transition's TD error?
- That's probably too aggressive, let's interpolate between random and this one.

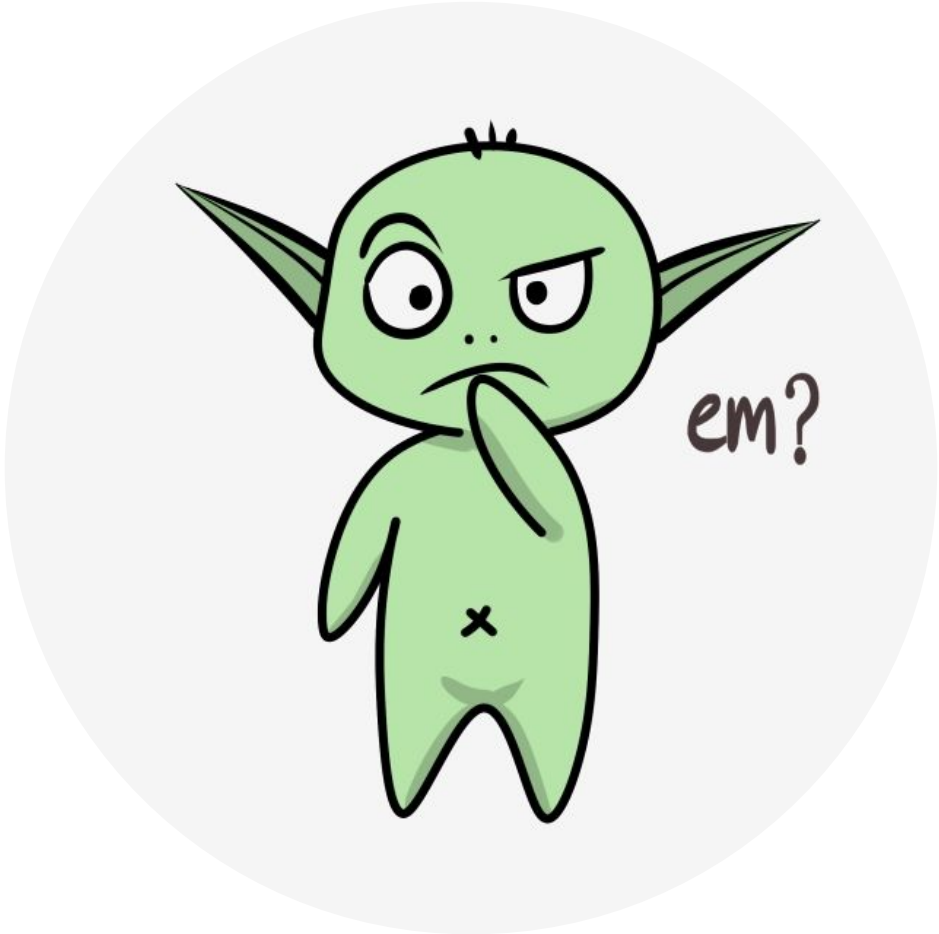




Prioritized Experience Replay

[Schaul et al. 2015]



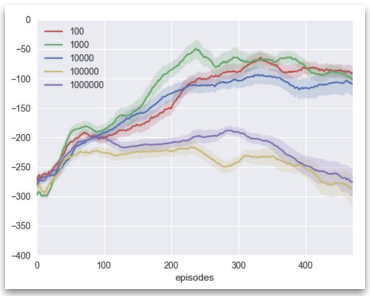


https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Understanding the Experience Replay Buffer

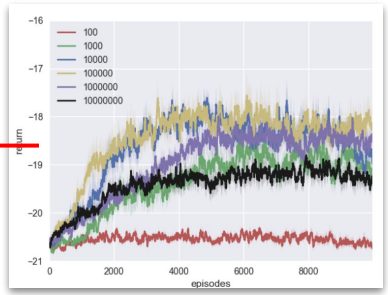
[Zhang & Sutton 2017]

- Should the replay buffer size always be 1M?
  - Obviously, it is task-dependent.
- How can we ensure the latest observation is not ignored?
  - Combined experience replay (CER).
  - It makes learning algorithms more robust to different sizes of the experience replay buffer (in smaller domains, not so much in an Atari game).
- Evaluated only on the RAM state of an Atari game.



Lunar Lander

These scores are really bad overall



Pong

Understanding the Experience Replay Buffer

[Fedus et al. 2020]

- We need to disentangle different aspects of experience replay buffers.
  - Capacity: Size of experience replay buffer.
  - Age of a transition: num. of gradient steps taken by the alg. since the transition was generated.
  - Replay ratio: number of gradient updates per environment transition.

		Replay Capacity				
		100,000	316,228	**1,000,000**	3,162,278	10,000,000
Oldest Policy	25,000,000	250.000	79.057	25.000	7.906	2.500
	2,500,000	25.000	7.906	2.500	0.791	**0.250**
	250,000	2.500	0.791	**0.250**	0.079	0.025
	25,000	**0.250**	0.079	0.025	0.008	0.003

Understanding the Experience Replay Buffer

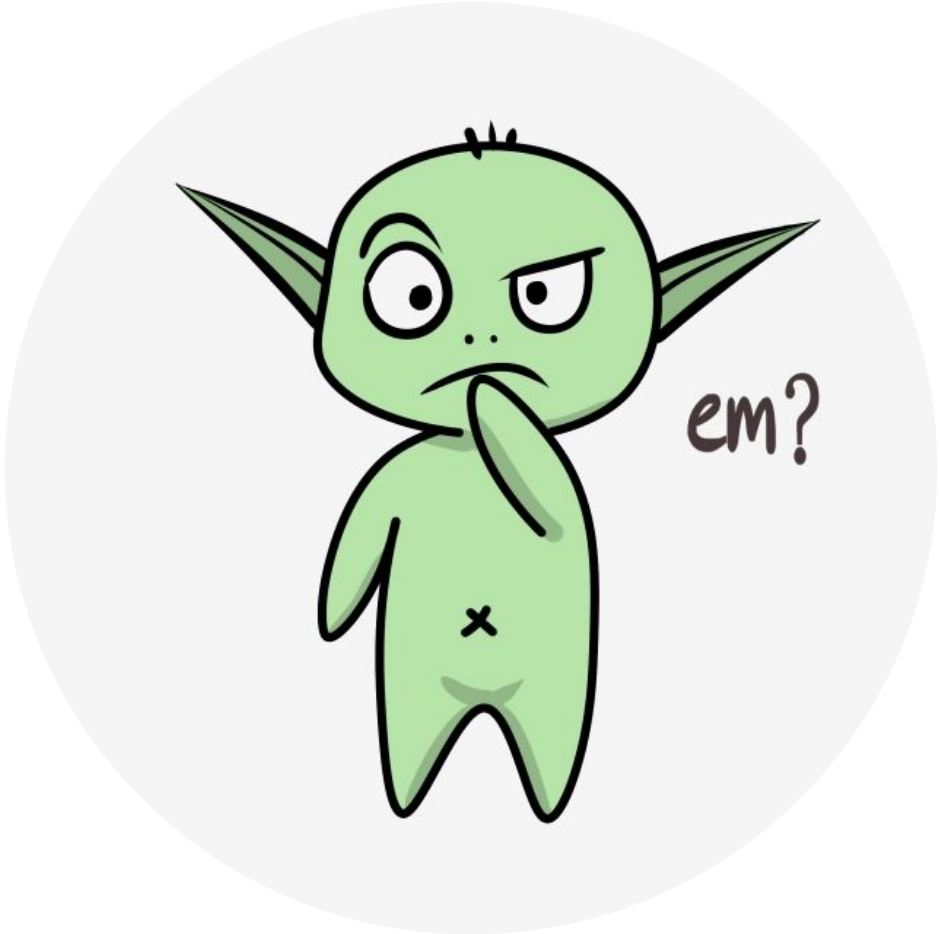
[Fedus et al. 2020]

- Some trends in experiments with Rainbow, in 14 Atari games (albeit 3 seeds):
  - Increasing replay capacity improves performance.
  - Reducing the oldest policy improves performance.
  - Increasing buffer size with a fixed replay ratio has varying improvements.

These are algorithm dependent

		Replay Capacity				
		100,000	316,228	**1,000,000**	3,162,278	10,000,000
Oldest Policy	25,000,000	-74.9	-76.6	-77.4	-72.1	-54.6
	2,500,000	-78.1	-73.9	-56.8	-16.7	**28.7**
	250,000	-70.0	-57.4	**0.0**	13.0	18.3
	25,000	**-31.9**	12.4	16.9	--	--

- *n*-step returns seem to benefit from large replay capacity.



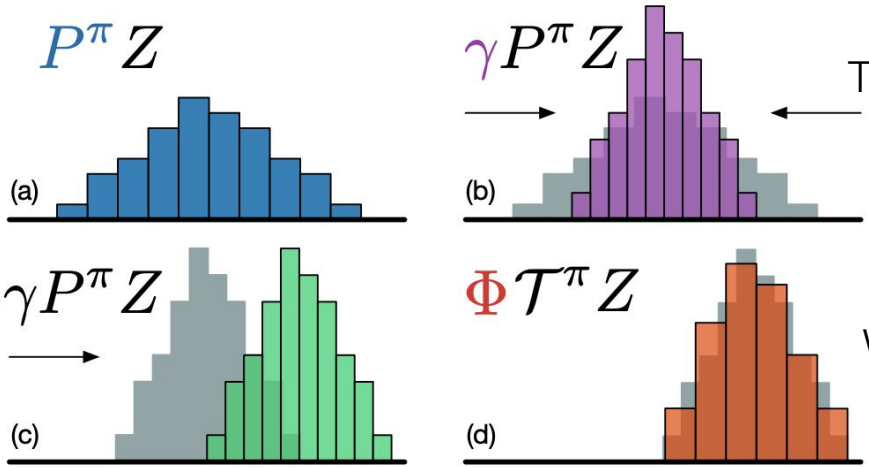
https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Distributional Q-Learning

[Bellemare et al. 2017]

- What if we computed the distribution of the return (instead of the expectation)?
- This is out of the scope of this introductory course, but here's a high-level overview. A distributional Bellman operator with a deterministic reward function:

Next state distribution under policy π



The discount factor shrinks the distribution towards 0

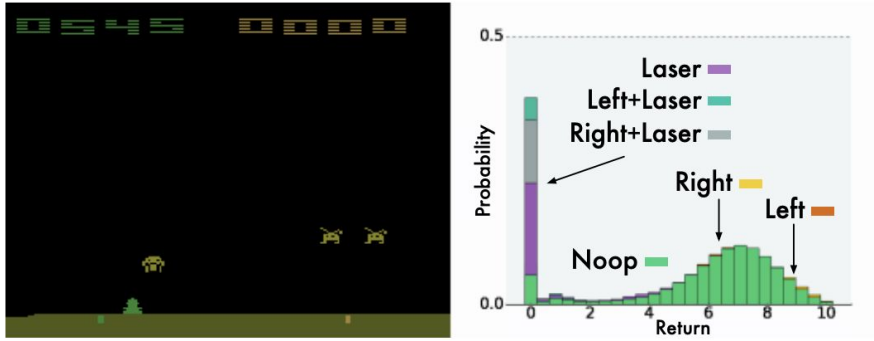
The reward shifts the distribution

We need a projection step

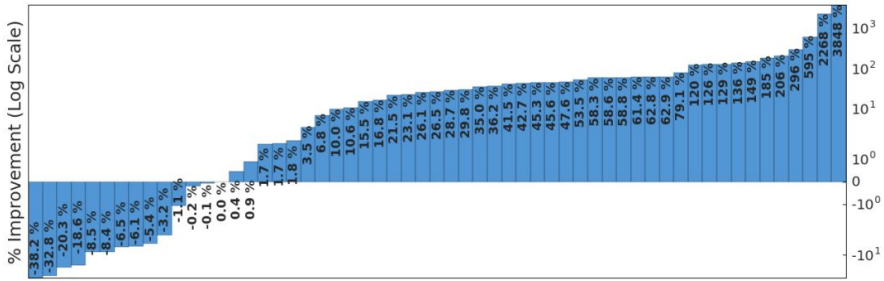
Distributional Q-Learning

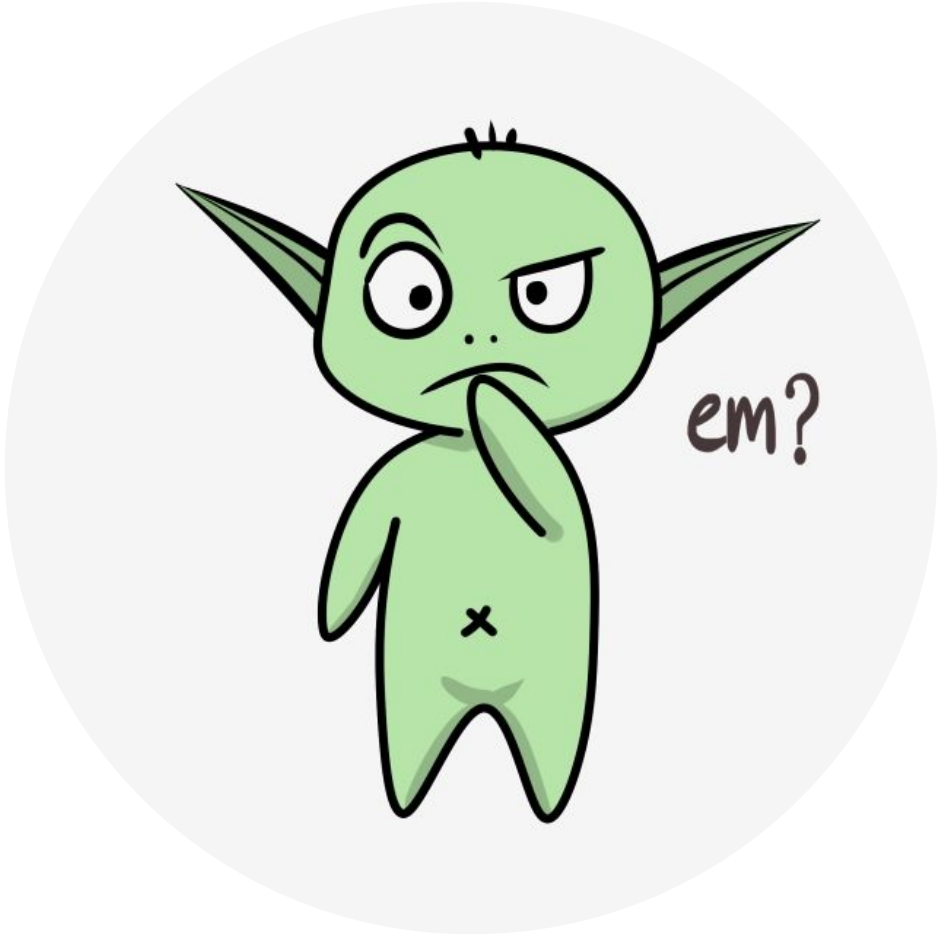
[Bellemare et al. 2017]

- Why does it work?
  - It makes no difference in the tabular or linear function approximation case (Lyle et al. 2019).
  - Auxiliary tasks? ¯_(ツ)_/¯
- There's always more nuance
  - "The performance profiles for DQN (Adam) and C51 overlap significantly" (Agarwal et al., 2021).



Learned value distribution during an episode.



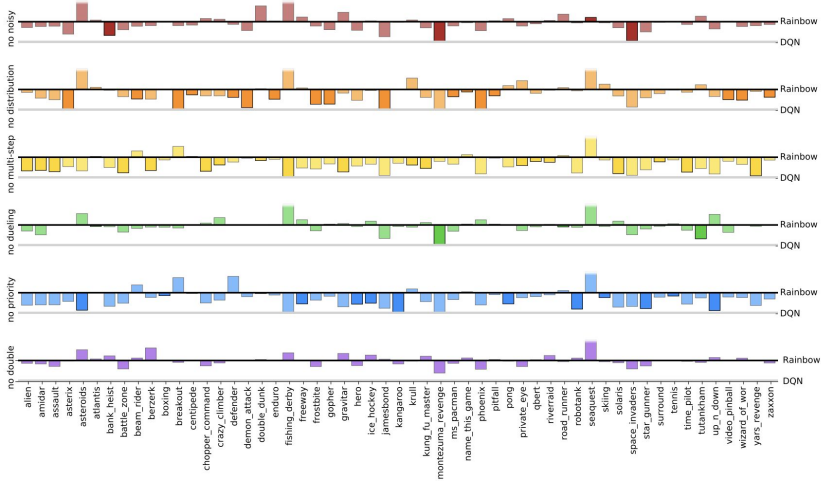


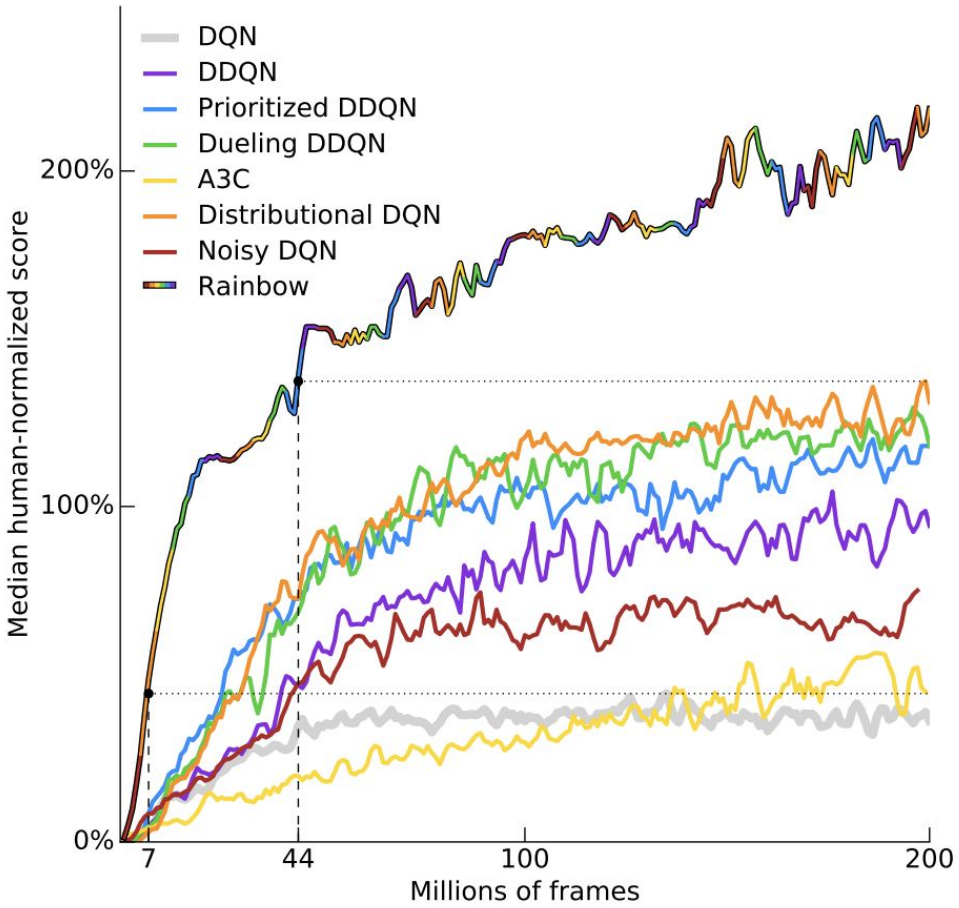
https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

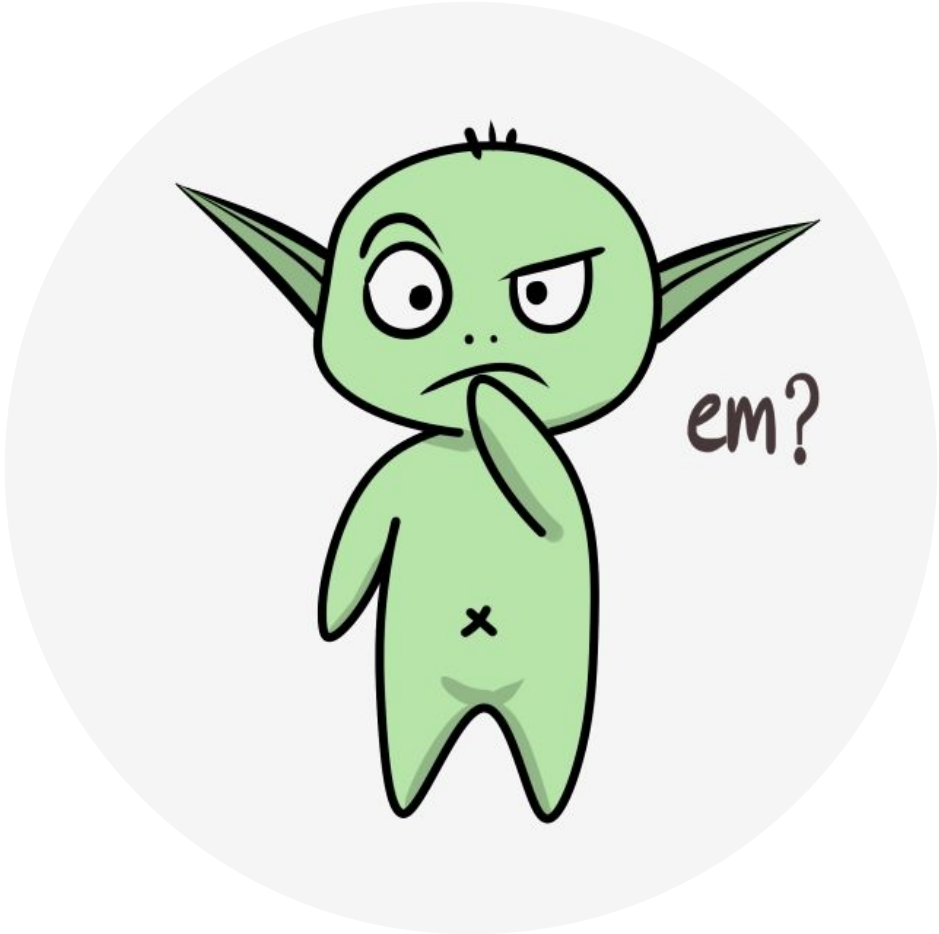
Rainbow

[Hessel et al. 2018]

- There are many ideas out there. Are they additive?







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