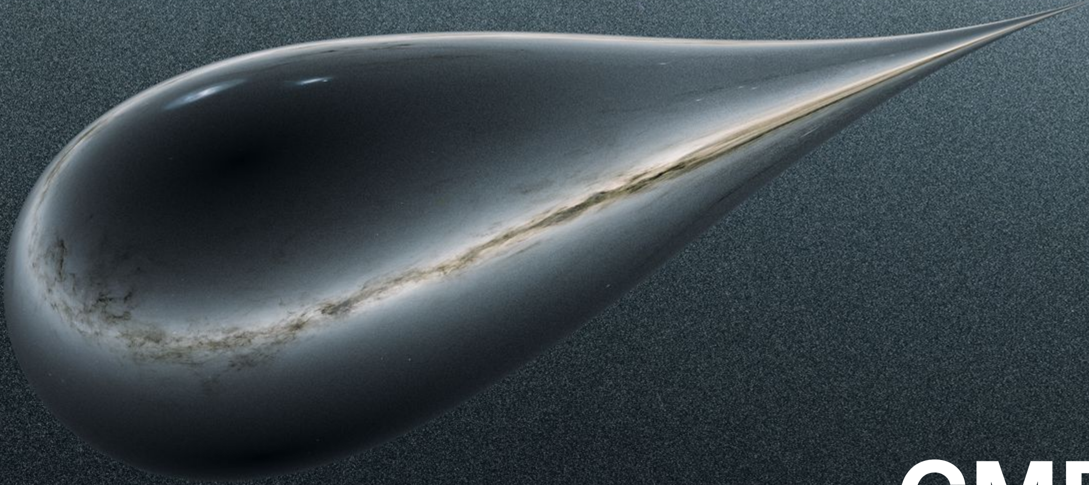
"Your lack of fear is based on your ignorance."

Liu Cixin, *The Three-Body Problem*



CMPUT 628 Deep RL

Marlos C. Machado

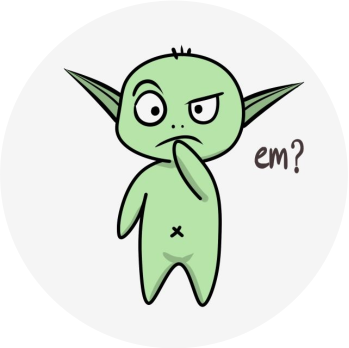
https://thomasbronzwaer.wordpress.com/2020/07/30/the-trisolaran-probe-from-liu-cixins-the-three-body-problem/

Class 10/ 25

Reminders & Notes

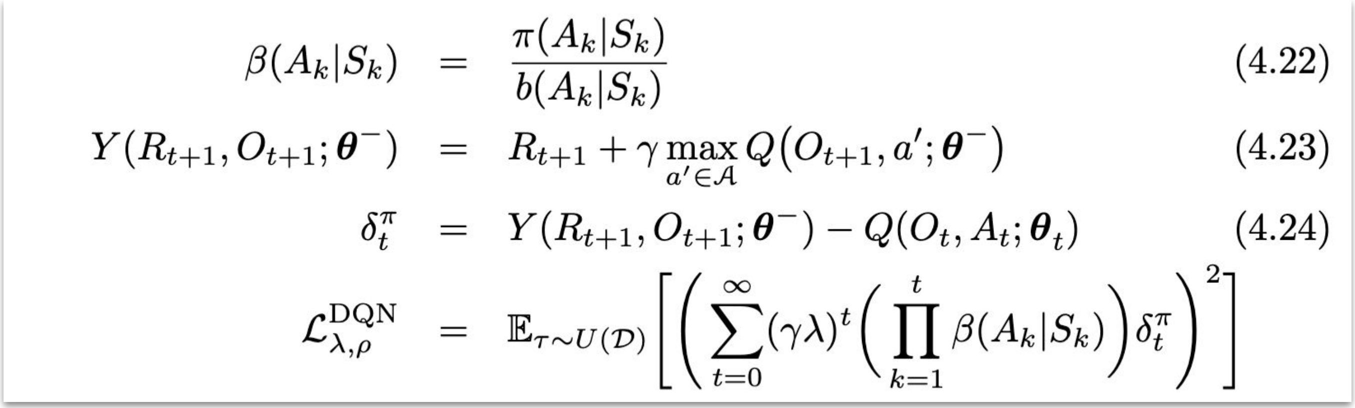
- Assignment 2 is due on Friday, February 7
- I'm doing my best to release Assignment 3; it might end up being delayed
- Lecture-notes-wise: I haven't written about distributional RL yet
- I will release instructions about seminar and paper review
  during the reading week (Feb 18 – Feb 21)
  You should send me your groups
- I will be travelling on March 3rd (Monday), 2025
  A. Rupam Mahmood will give a guest lecture on streaming deep RL

Please, interrupt me at any time!

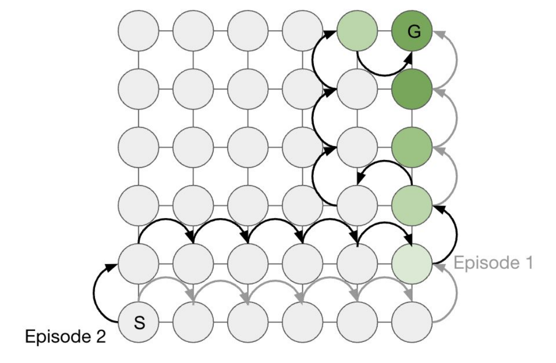


https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Last class: Multi-step Credit Assignment



$$
\begin{aligned}
\beta(A_k|S_k) &= \frac{\pi(A_k|S_k)}{b(A_k|S_k)} && (4.22)\\
Y(R_{t+1}, O_{t+1}; \boldsymbol{\theta}^-) &= R_{t+1} + \gamma \max_{a' \in \mathcal{A}} Q(O_{t+1}, a'; \boldsymbol{\theta}^-) && (4.23)\\
\delta_t^\pi &= Y(R_{t+1}, O_{t+1}; \boldsymbol{\theta}^-) - Q(O_t, A_t; \boldsymbol{\theta}_t) && (4.24)\\
\mathcal{L}_{\lambda,\rho}^{\text{DQN}} &= \mathbb{E}_{\tau \sim U(\mathcal{D})}\left[\left(\sum_{t=0}^{\infty} (\gamma\lambda)^t \left(\prod_{k=1}^{t} \beta(A_k|S_k)\right) \delta_t^\pi\right)^2\right]
\end{aligned}
$$



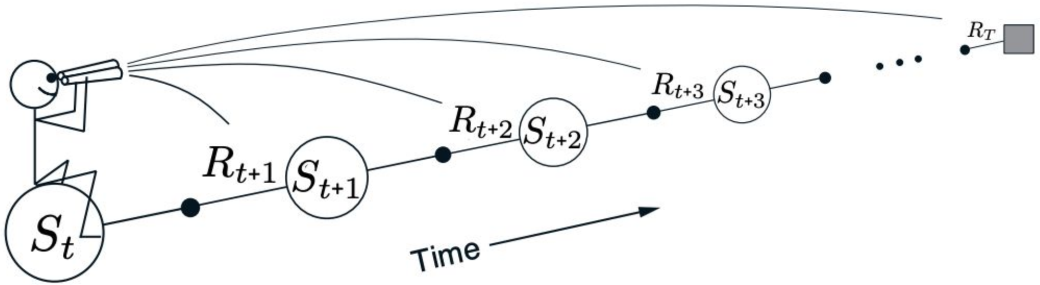
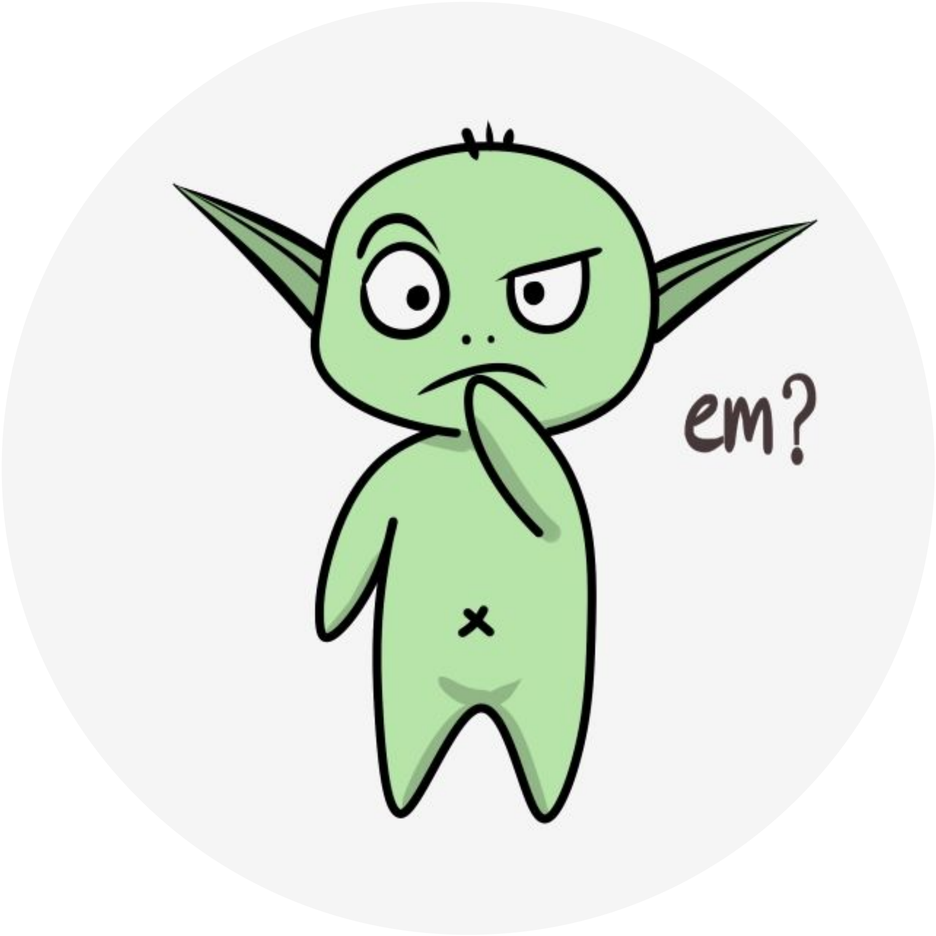
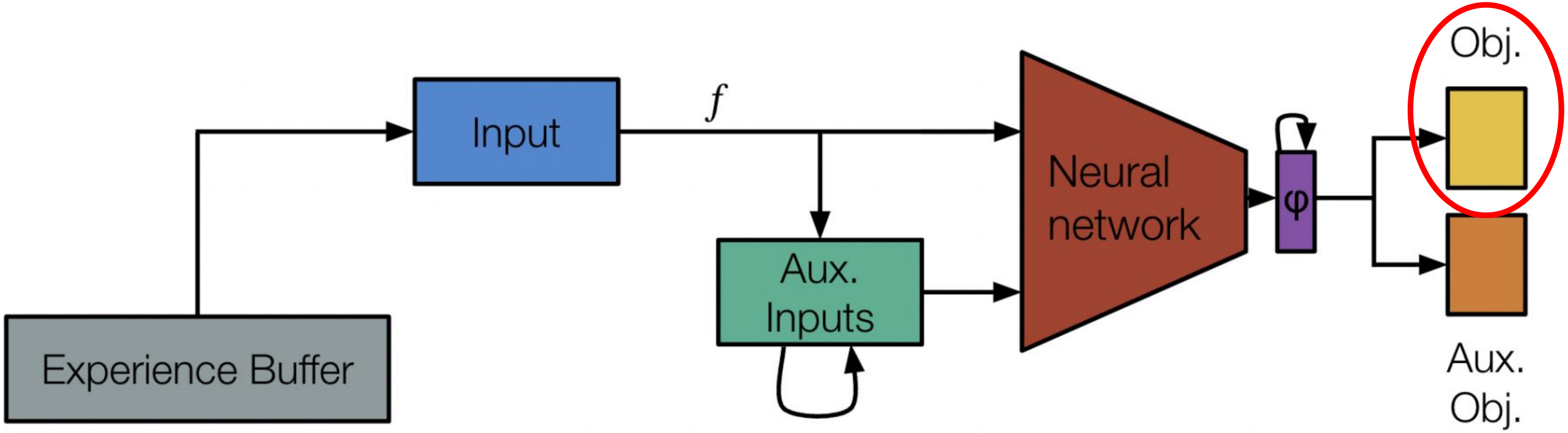


Figure 12.4: The forward view. We decide how to update each state by looking forward to future rewards and states.

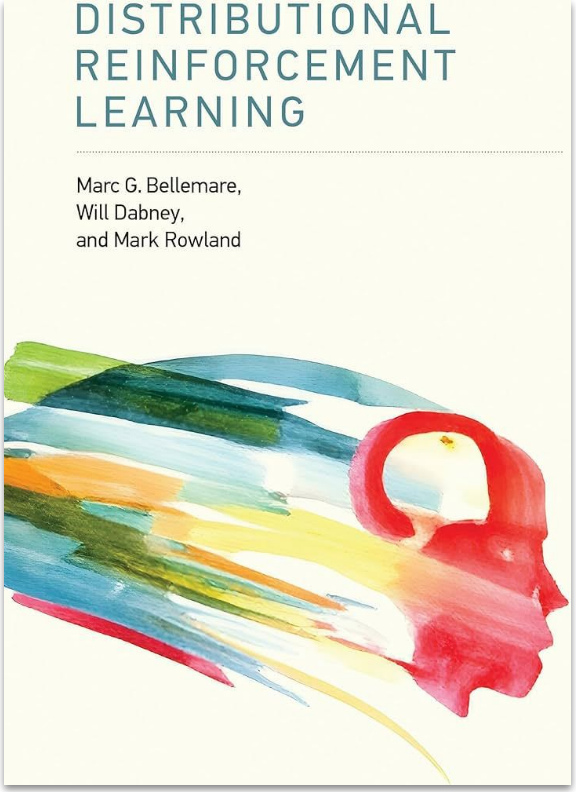


https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

We Continue to Look at Different Objective Functions



Distributional Reinforcement Learning



DISTRIBUTIONAL REINFORCEMENT LEARNING

Marc G. Bellemare,
Will Dabney,
and Mark Rowland

Traditional RL is All About the Expected Return

$$
\begin{aligned}
v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s] \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)\Big[r + \gamma v_\pi(s')\Big]
\end{aligned}
$$

$$q_{p,\pi}(o, a) = \mathbb{E}_p\Big[r(o, a)\Big] + \gamma \mathbb{E}_{p,\pi}\Big[q_\pi(O', A')\Big]$$

Distributional RL Models the Return Distribution

- Instead of modelling the expected return, they propose studying the random return, *z*, whose expectation is the value *q*. This leads to the *value distribution*:

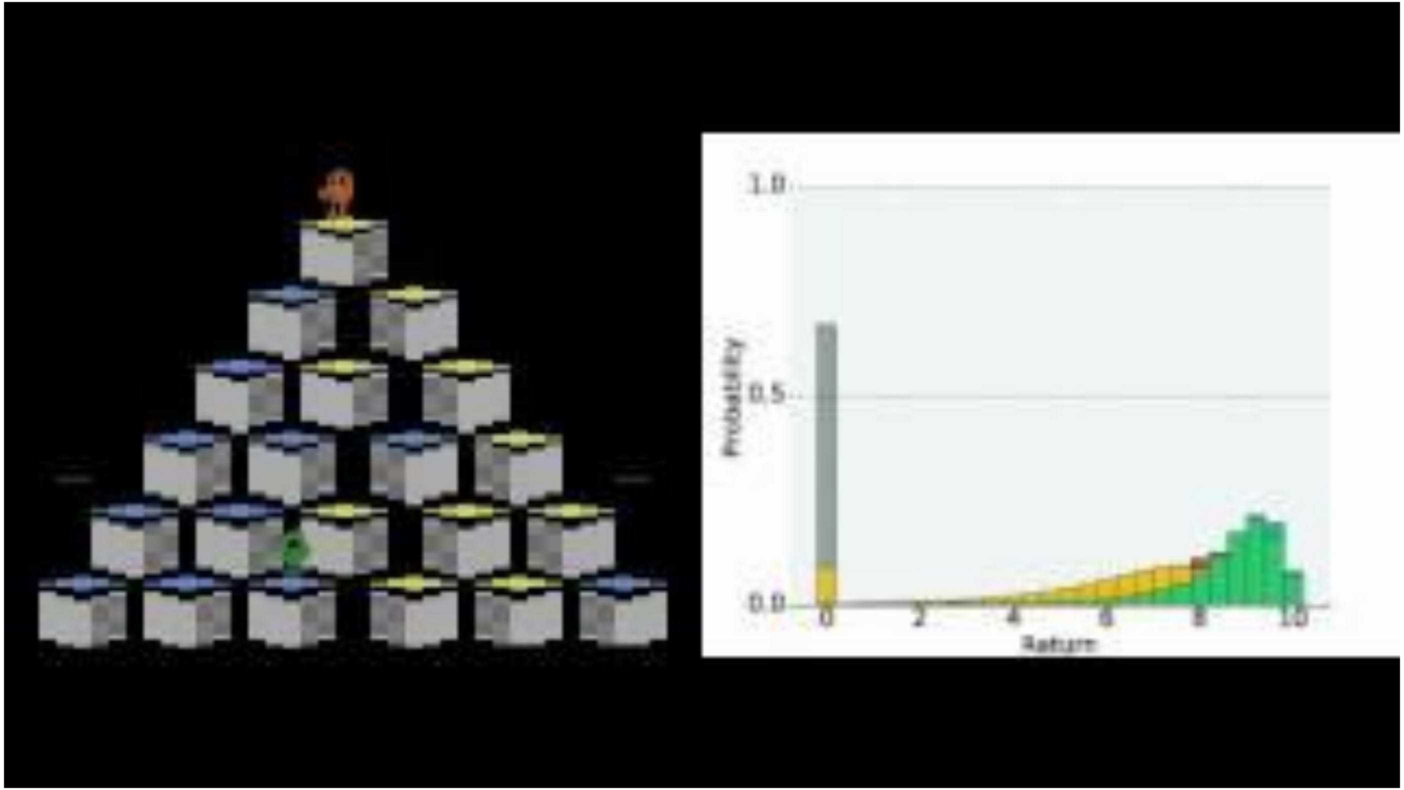
$z_{p,\pi}$ is distributed according to the same law as in the RHS

$$z_{p,\pi}(o,a) \overset{D}{=} r_p(o,a) + \gamma z_{p,\pi}(O', A')$$

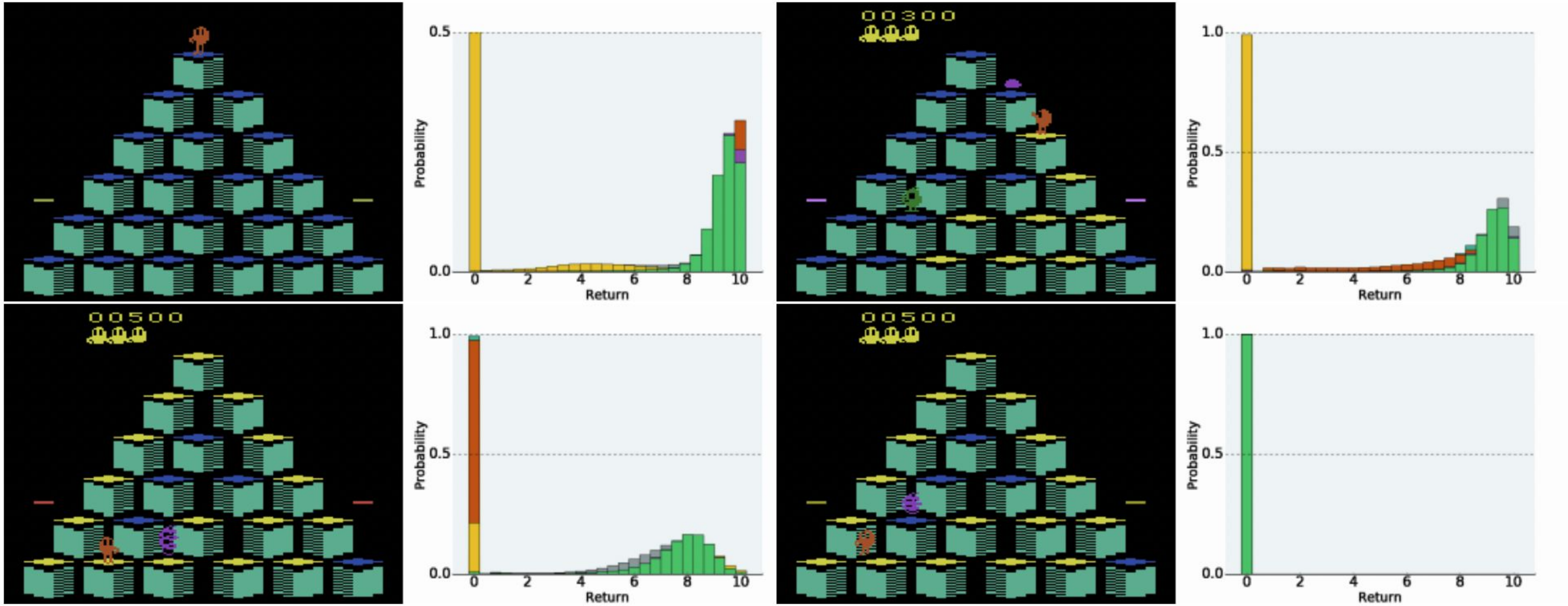
$$q_{p,\pi}(o,a) \doteq \mathbb{E}_{p,\pi}\left[z_\pi(o,a)\right]$$

- Why should we do this? Intuitions from the first paper [Bellemare et al., 2017]:
  - Preserves multimodality in value distribution
  - Mitigates the effects of learning from a nonstationary policy
  - Overall, “makes approximate reinforcement learning significantly better behaved”
  - Better deal with state aliasing
  - Auxiliary task effect (a richer set of predictions)

An example [Bellemare et al. 2017]

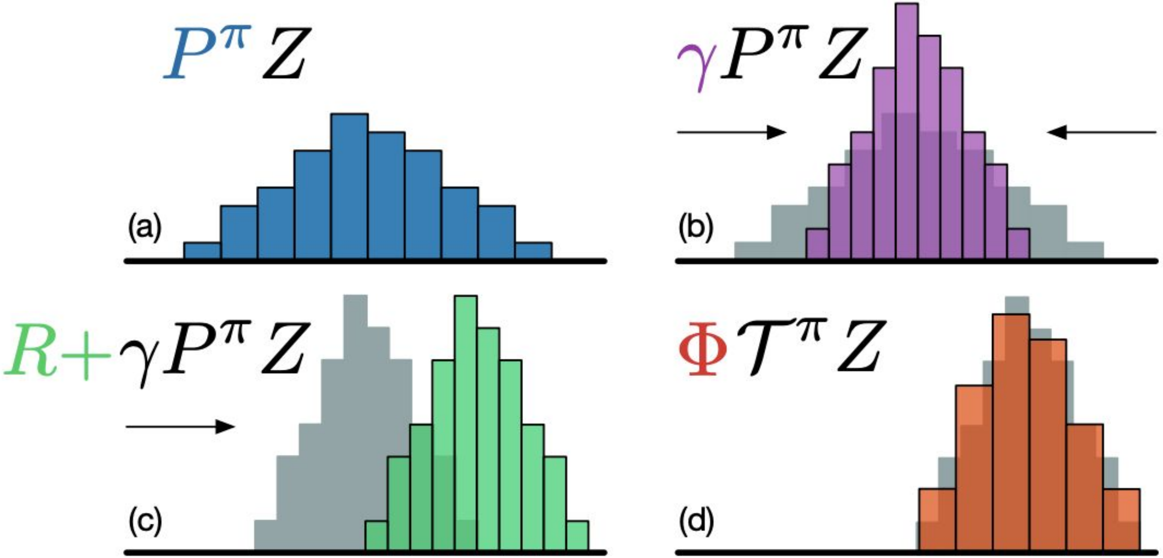


An example [Bellemare et al. 2017]



There are Equivalent Distributional Bellman Operators

$$\mathcal{T}^\pi Z(x,a) :\overset{D}{=} R(x,a) + \gamma P^\pi Z(x,a)$$



Images by Bellemare et al. (2017)

Categorial Algorithm: A First Distributional Algorithm

[Bellemare et al. 2017]

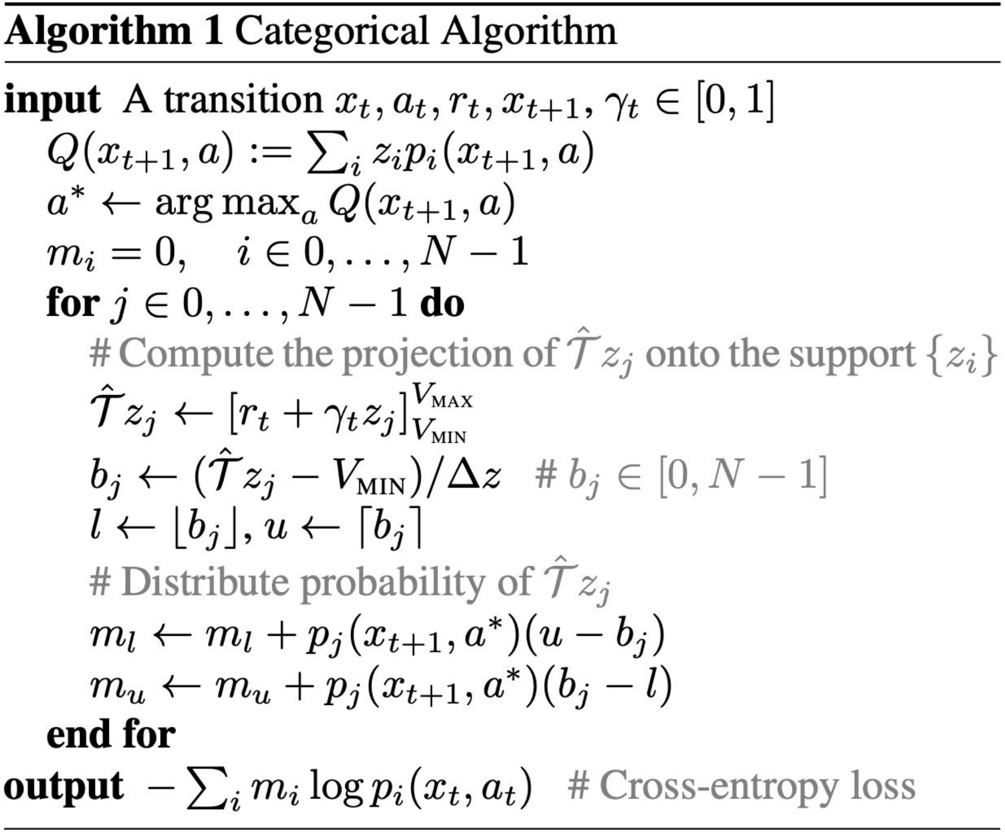
- Model value distribution using a discrete distribution with *N* atoms, ranging from V_{Min} to V_{Max} such that $\Delta z \doteq (V_{Max} - V_{Min})/(N-1)$
- The probability of each atom is given by a softmax
- *They reduce the Bellman update to multiclass classification* (more on this later)
- Sample loss is the cross-entropy term of the KL divergence

$$D_{KL}\left(\underbrace{\Phi}_{\text{Projected}}\underbrace{\hat{\mathcal{T}}}_{\text{Bellman update}} z(o, a; \boldsymbol{\theta}) || z(o, a; \boldsymbol{\theta})\right)$$

$$-\sum_{s} \Phi\hat{\mathcal{T}} z(k \mid o, a; \boldsymbol{\theta}) \log z(k \mid o, a; \boldsymbol{\theta})$$

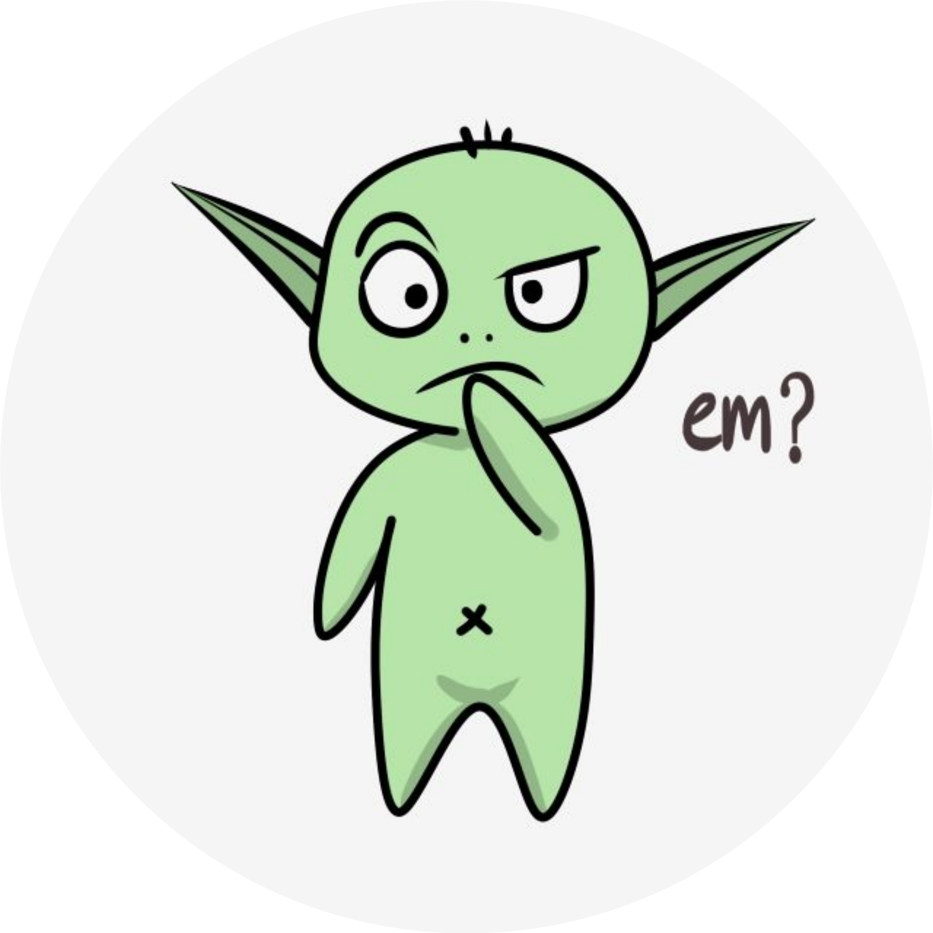
Categorial Algorithm: A First Distributional Algorithm

[Bellemare et al. 2017]



Algorithm 1 Categorical Algorithm

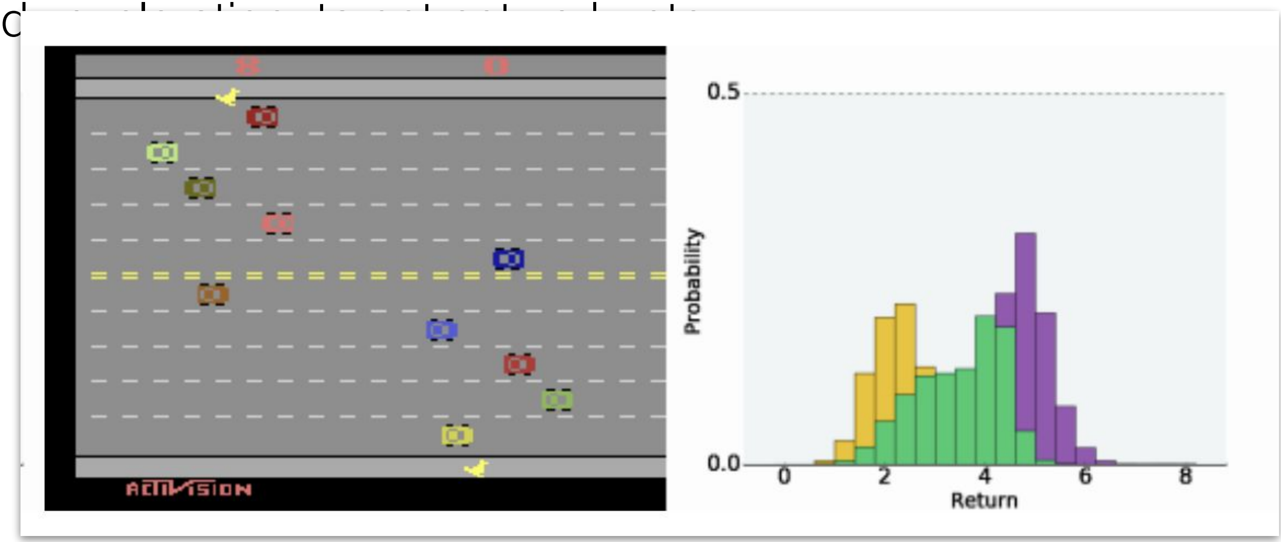
```
input A transition $x_t, a_t, r_t, x_{t+1}, \gamma_t \in [0,1]$
  $Q(x_{t+1}, a) := \sum_i z_i p_i(x_{t+1}, a)$
  $a^* \leftarrow \arg\max_a Q(x_{t+1}, a)$
  $m_i = 0, \quad i \in 0, \dots, N-1$
  for $j \in 0, \dots, N-1$ do
    # Compute the projection of $\hat{\mathcal{T}} z_j$ onto the support $\{z_i\}$
    $\hat{\mathcal{T}} z_j \leftarrow [r_t + \gamma_t z_j]_{V_{\text{MIN}}}^{V_{\text{MAX}}}$
    $b_j \leftarrow (\hat{\mathcal{T}} z_j - V_{\text{MIN}})/\Delta z$   # $b_j \in [0, N-1]$
    $l \leftarrow \lfloor b_j \rfloor, u \leftarrow \lceil b_j \rceil$
    # Distribute probability of $\hat{\mathcal{T}} z_j$
    $m_l \leftarrow m_l + p_j(x_{t+1}, a^*)(u - b_j)$
    $m_u \leftarrow m_u + p_j(x_{t+1}, a^*)(b_j - l)$
  end for
output $-\sum_i m_i \log p_i(x_t, a_t)$   # Cross-entropy loss
```



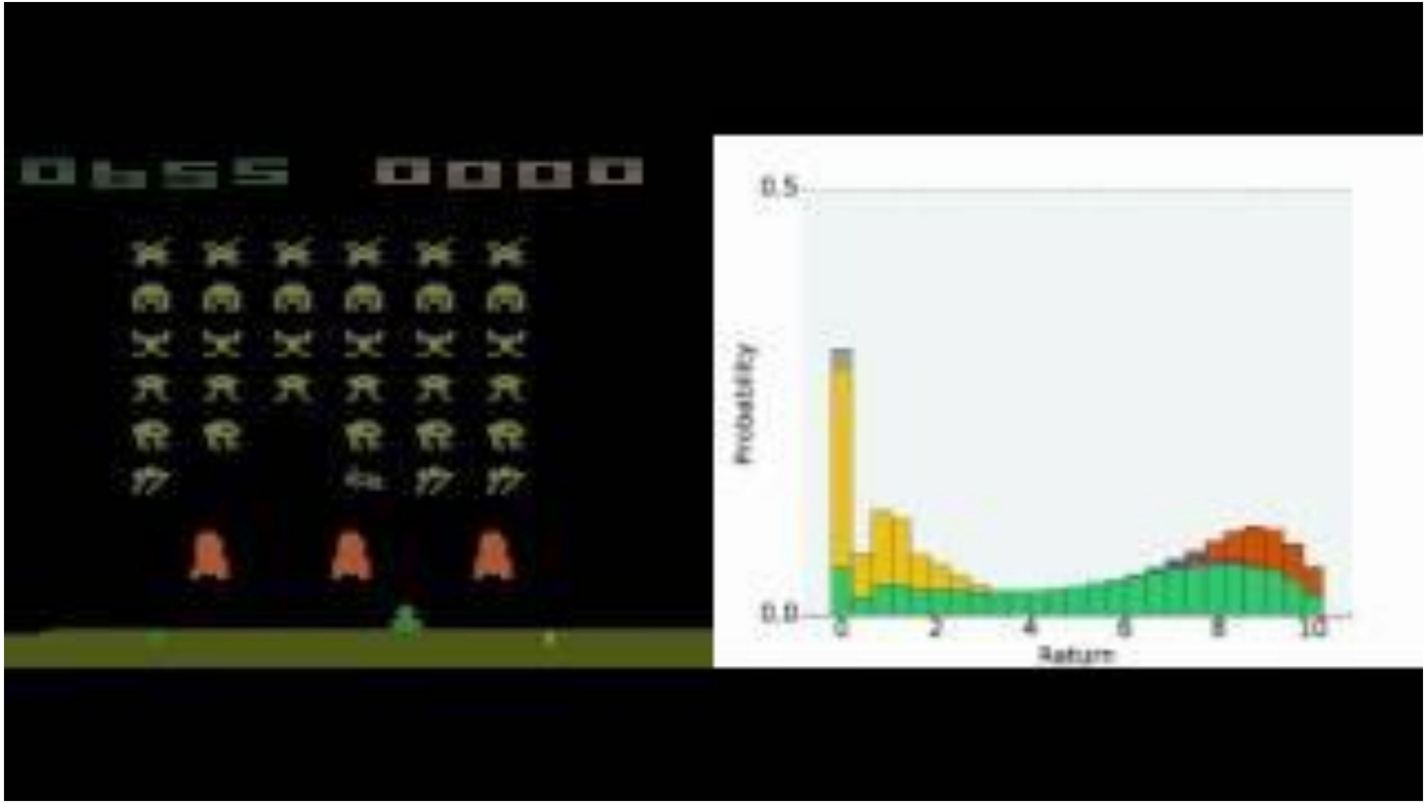
https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Categorial Algorithm in Atari 2600 Games [Bellemare et al. 2017]

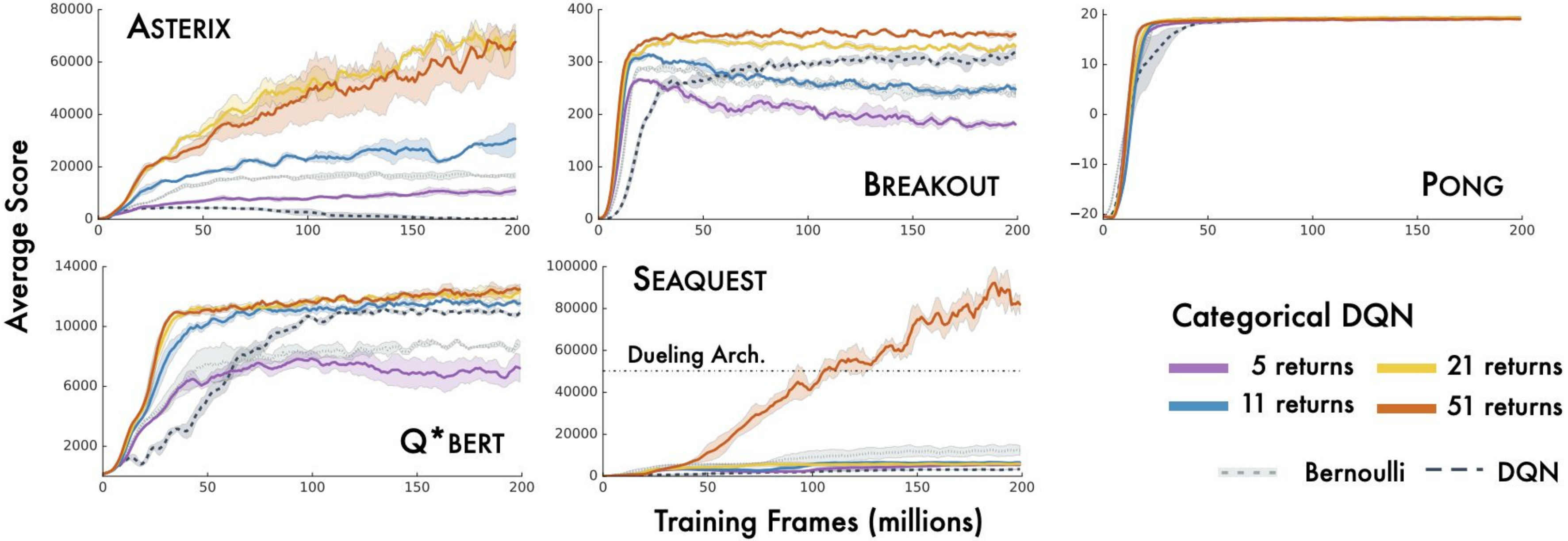
- Deterministic Arcade Learning Environment
- DQN but outputting atom prob. $p_i(o,a)$ instead of action-values ($V_{Max} = -V_{Min} = 10$)
- ε-greed



Categorial Algorithm in Atari 2600 Games [Bellemare et al. 2017]

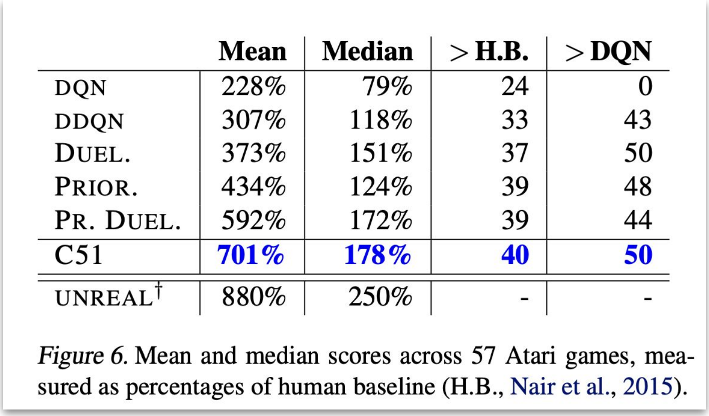


How many atoms? [Bellemare et al. 2017]



C51 on the Arcade Learning Environment [Bellemare et al. 2017]

- ε-greedy exploration with ε=0.01 and evaluation every 1M frames with ε=0.001



	Mean	Median	> H.B.	> DQN
DQN	228%	79%	24	0
DDQN	307%	118%	33	43
DUEL.	373%	151%	37	50
PRIOR.	434%	124%	39	48
PR. DUEL.	592%	172%	39	44
C51	**701%**	**178%**	**40**	**50**
UNREAL[†]	880%	250%	-	-

Figure 6. Mean and median scores across 57 Atari games, measured as percentages of human baseline (H.B., Nair et al., 2015).

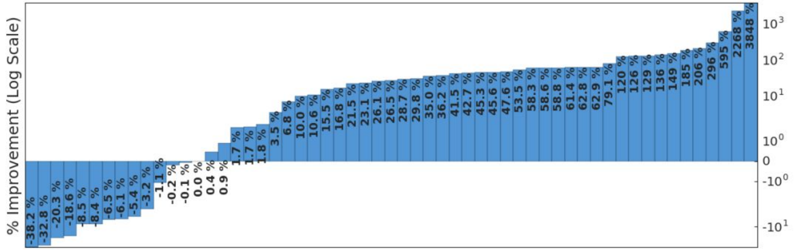
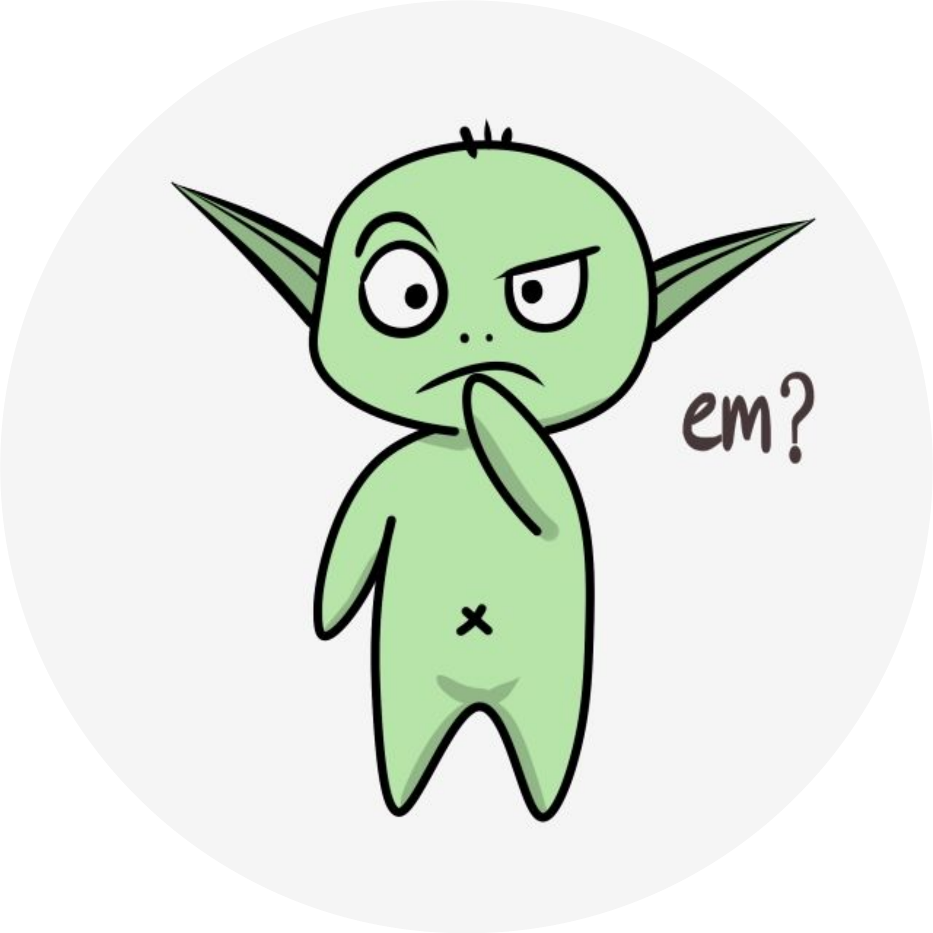


Figure 7. Percentage improvement, per-game, of C51 over Double DQN, computed using van Hasselt et al.'s method.

Number of seeds not mentioned, lower ε than the baselines…

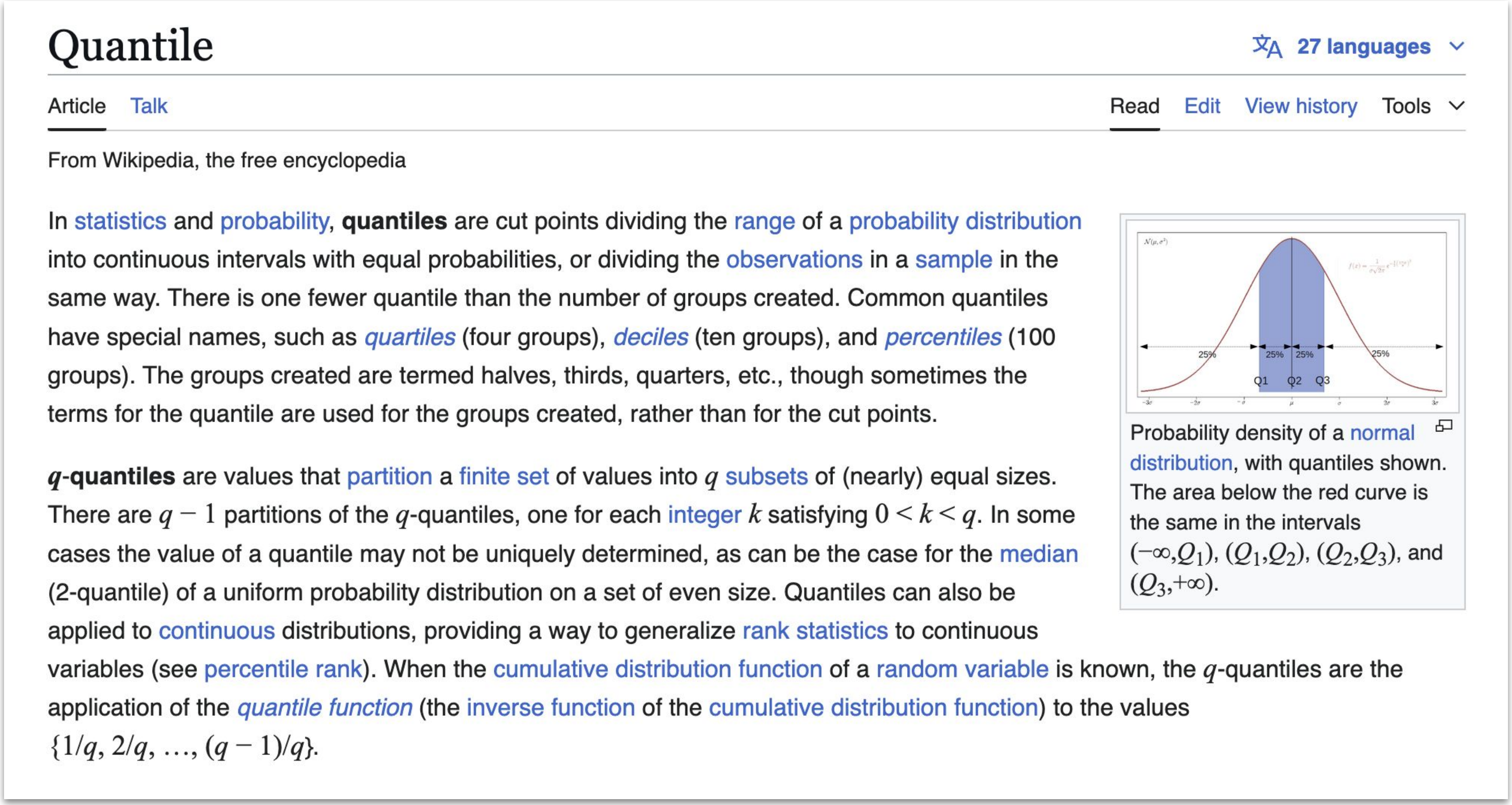
But it works!



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

C51 is but one algorithm

- C51 was not that well-aligned with the existing theoretical results at the time
    - Ideally one wants to minimize the Wasserstein distance between a distribution, z, and its Bellman update, $\mathscr{T}z$, but Bellemare et al. (2017) showed that "the Wasserstein metric, viewed as a loss, cannot generally be minimized using SGD" [Dabney et al., 2018]
- QR-DQN: using quantile regression to directly minimize the Wasserstein metric, rather than its heuristic approximation [Dabney et al., 2018]
    - "Instead of using N fixed locations for its approximation distribution and adjusting their probabilities, they assign fixed, uniform probabilities to N adjustable locations"



Quantile

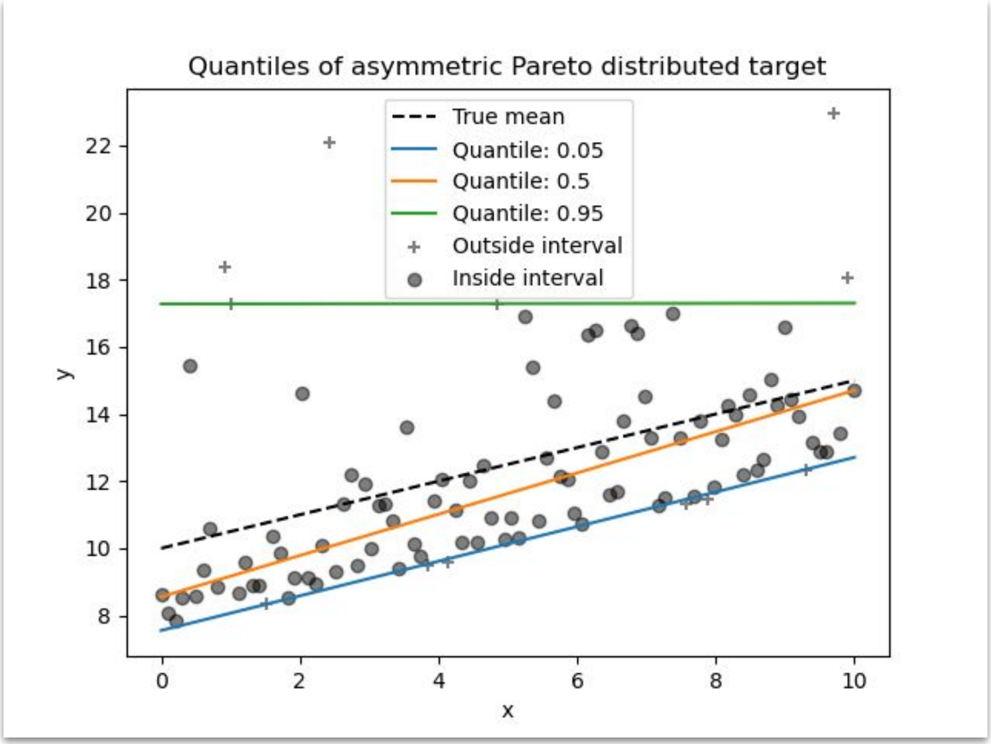
From Wikipedia, the free encyclopedia

In statistics and probability, **quantiles** are cut points dividing the range of a probability distribution into continuous intervals with equal probabilities, or dividing the observations in a sample in the same way. There is one fewer quantile than the number of groups created. Common quantiles have special names, such as *quartiles* (four groups), *deciles* (ten groups), and *percentiles* (100 groups). The groups created are termed halves, thirds, quarters, etc., though sometimes the terms for the quantile are used for the groups created, rather than for the cut points.

***q*-quantiles** are values that partition a finite set of values into q subsets of (nearly) equal sizes. There are $q - 1$ partitions of the q-quantiles, one for each integer k satisfying $0 < k < q$. In some cases the value of a quantile may not be uniquely determined, as can be the case for the median (2-quantile) of a uniform probability distribution on a set of even size. Quantiles can also be applied to continuous distributions, providing a way to generalize rank statistics to continuous variables (see percentile rank). When the cumulative distribution function of a random variable is known, the q-quantiles are the application of the *quantile function* (the inverse function of the cumulative distribution function) to the values $\{1/q, 2/q, \ldots, (q-1)/q\}$.

Probability density of a normal distribution, with quantiles shown. The area below the red curve is the same in the intervals $(-\infty, Q_1)$, (Q_1, Q_2), (Q_2, Q_3), and $(Q_3, +\infty)$.

Quantile Regression



The intuition of quantile regression

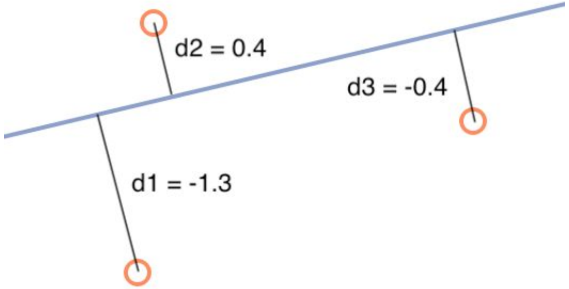
- “Traditional” regression minimizes $\mathcal{L}(\hat{y}, y) = \sum_{i}^{N} (\hat{y}_i - y_i)^2$

- Quantile regression minimizes something slightly different:

$$\mathcal{L}_\rho(\hat{y}, y) = \rho \sum_{y_i > \hat{y}_{\rho,i}} |\hat{y}_{\rho,i} - y_i| + (1 - \rho) \sum_{y_i < \hat{y}_{\rho,i}} |\hat{y}_{\rho,i} - y_i|$$

Quantile level

Example:



https://www.aptech.com/blog/the-basics-of-quantile-regression/

The intuition of quantile regression

$$\mathcal{L}(\hat{y}, y) = \sum_{i}^{N} (\hat{y}_i - y_i)^2$$

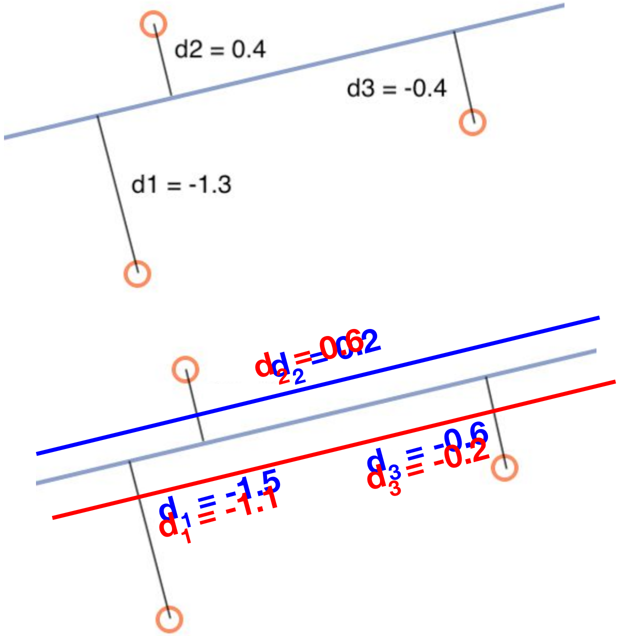
$$\mathcal{L}_\rho(\hat{y}, y) = \rho \sum_{y_i > \hat{y}_{\rho,i}} |\hat{y}_{\rho,i} - y_i| + (1-\rho) \sum_{y_i < \hat{y}_{\rho,i}} |\hat{y}_{\rho,i} - y_i|$$

- Quantile regression (ρ = 0.9):

$$\begin{aligned} & \rho \times d_2 + (1-\rho) \times |d_1 + d_3| \\ = \quad & 0.9 \times 0.4 + 0.1 \times |-1.3 - 0.4| \\ = \quad & 0.36 + 0.17 = 0.53 \quad \textit{vs. 2.01} \end{aligned}$$

$$\begin{aligned} & \rho \times d_2 + (1-\rho) \times |d_1 + d_3| \\ = \quad & 0.9 \times 0.2 + 0.1 \times |-1.5 - 0.6| \\ = \quad & 0.18 + 0.21 = 0.39 \quad \textit{vs. 2.65} \end{aligned}$$

$$\begin{aligned} & \rho \times d_2 + (1-\rho) \times |d_1 + d_3| \\ = \quad & 0.9 \times 0.6 + 0.1 \times |-1.1 - 0.2| \\ = \quad & 0.54 + 0.13 = 0.67 \quad \textit{vs. 1.61} \end{aligned}$$



https://www.aptech.com/blog/the-basics-of-quantile-regression/

The intuition of quantile regression

$$\mathcal{L}(\hat{y}, y) = \sum_{i}^{N} (\hat{y}_i - y_i)^2$$

$$\mathcal{L}_\rho(\hat{y}, y) = \rho \sum_{y_i > \hat{y}_{\rho,i}} |\hat{y}_{\rho,i} - y_i| + (1-\rho) \sum_{y_i < \hat{y}_{\rho,i}} |\hat{y}_{\rho,i} - y_i|$$

- Qua

=

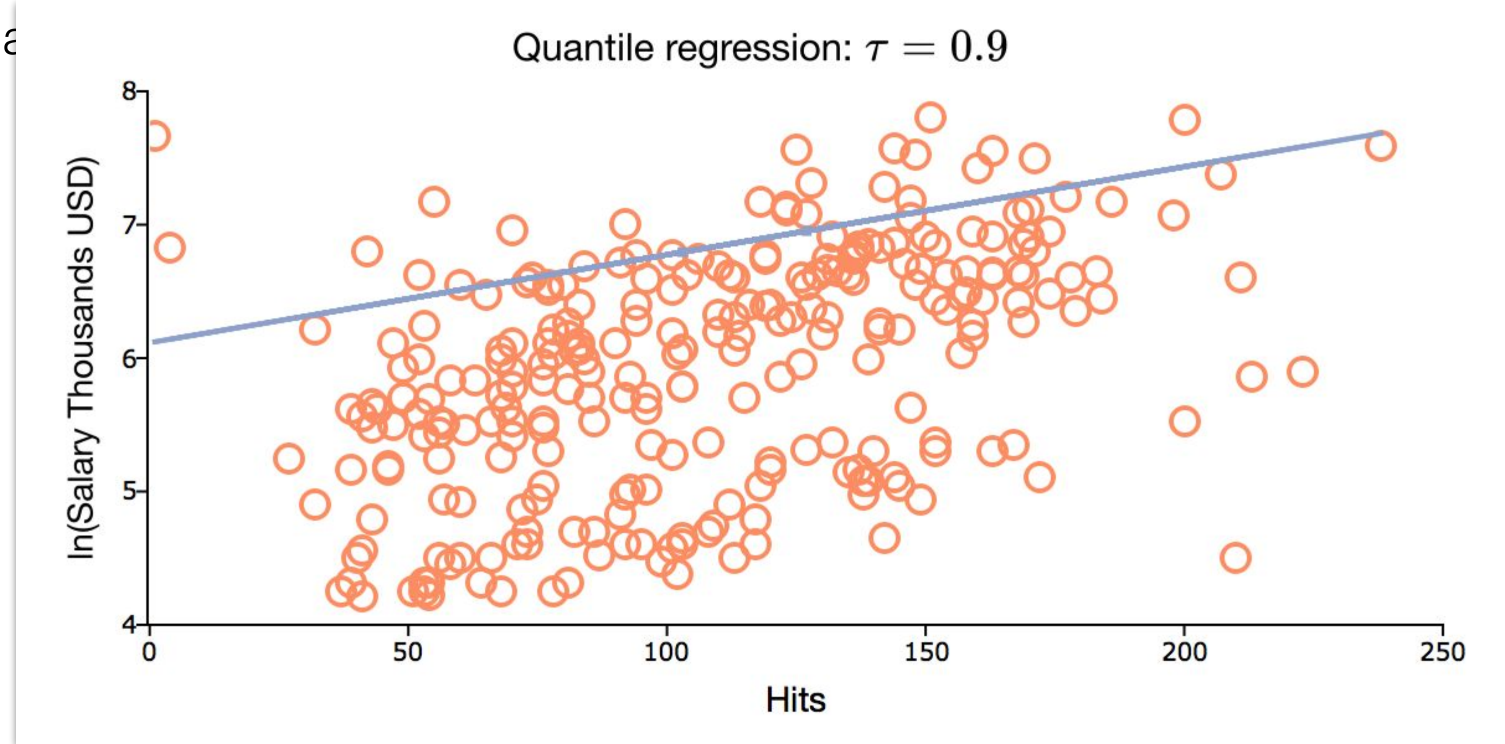
=

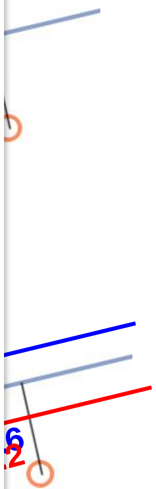
=

=

=

=





https://www.aptech.com/blog/the-basics-of-quantile-regression/

QR-DQN: Quantile Regression Q-Learning [Dabney et al., 2018]

- QR-DQN looks for the “locations” that will divide the distribution into 1/N quantiles

> *“Compared to the original parametrization, the benefits of a parameterized quantile distribution are threefold. First, (1) we are not restricted to prespecified bounds on the support, or a uniform resolution, potentially leading to significantly more accurate predictions when the range of returns vary greatly across states. This also (2) lets us do away with the unwieldy projection step present in C51, as there are no issues of disjoint supports. Together, these obviate the need for domain knowledge about the bounds of the return distribution when applying the algorithm to new tasks. Finally, (3) this reparametrization allows us to minimize the Wasserstein loss, without suffering from biased gradients, specifically, using quantile regression.”*

– Dabney et al. (2018)

QR-DQN: Quantile Regression Q-Learning [Dabney et al., 2018]

- They actually use a Quantile Huber Loss, check out the paper for that
- QRTD:

$$\theta_i(o) \leftarrow \theta_i(o) + \alpha\big(\hat{\rho}_i - \delta_{r+\gamma z' < \theta_i(o)}\big)$$

"Value" of the next observation

Estimated value of the "threshold" w/ observation *o*

Quantile

Dirac. Sort of an indicator function, but not really

QR-DQN: Quantile Regression Q-Learning [Dabney et al., 2018]

- They actually use a Quantile Huber Loss, check out the paper for that
- QRTD:

$$\theta_i(o) \leftarrow \theta_i(o) + \alpha\big(\hat{\rho}_i - \delta_{r+\gamma z' < \theta_i(o)}\big)$$

Estimated value of the “threshold” w/ observation *o*

Quantile

“Value” of the next observation, but sampled from the distrib.

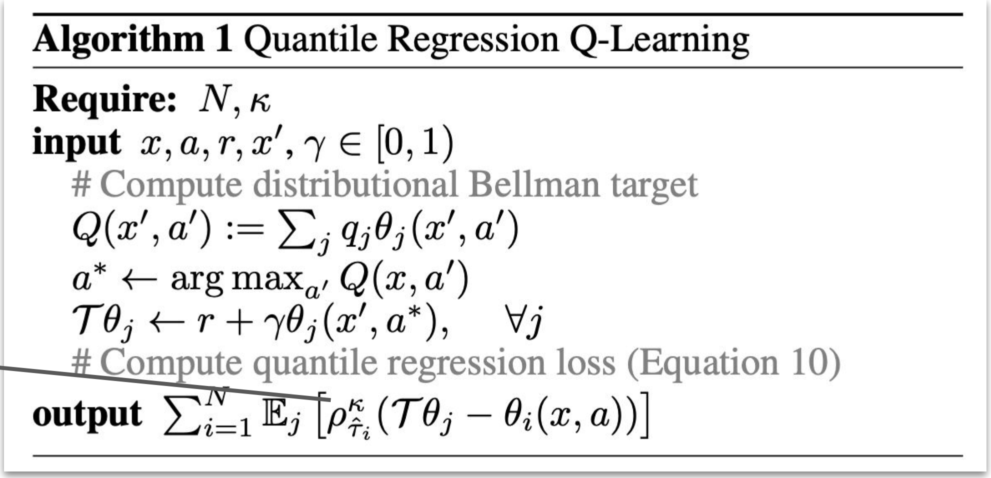
Dirac. Sort of an indicator function, but not really

QR-DQN: Quantile Regression Q-Learning [Dabney et al., 2018]

- For Deep RL there are three modifications to DQN:
  - Output layer becomes $|\mathcal{A}|$ x *N*, where *N* is a hyper-parameter giving the # of quantile targets
  - Replace Huber Loss by Quantile Huber Loss (see pseudo-code below)
  - Replace RMSProp with Adam

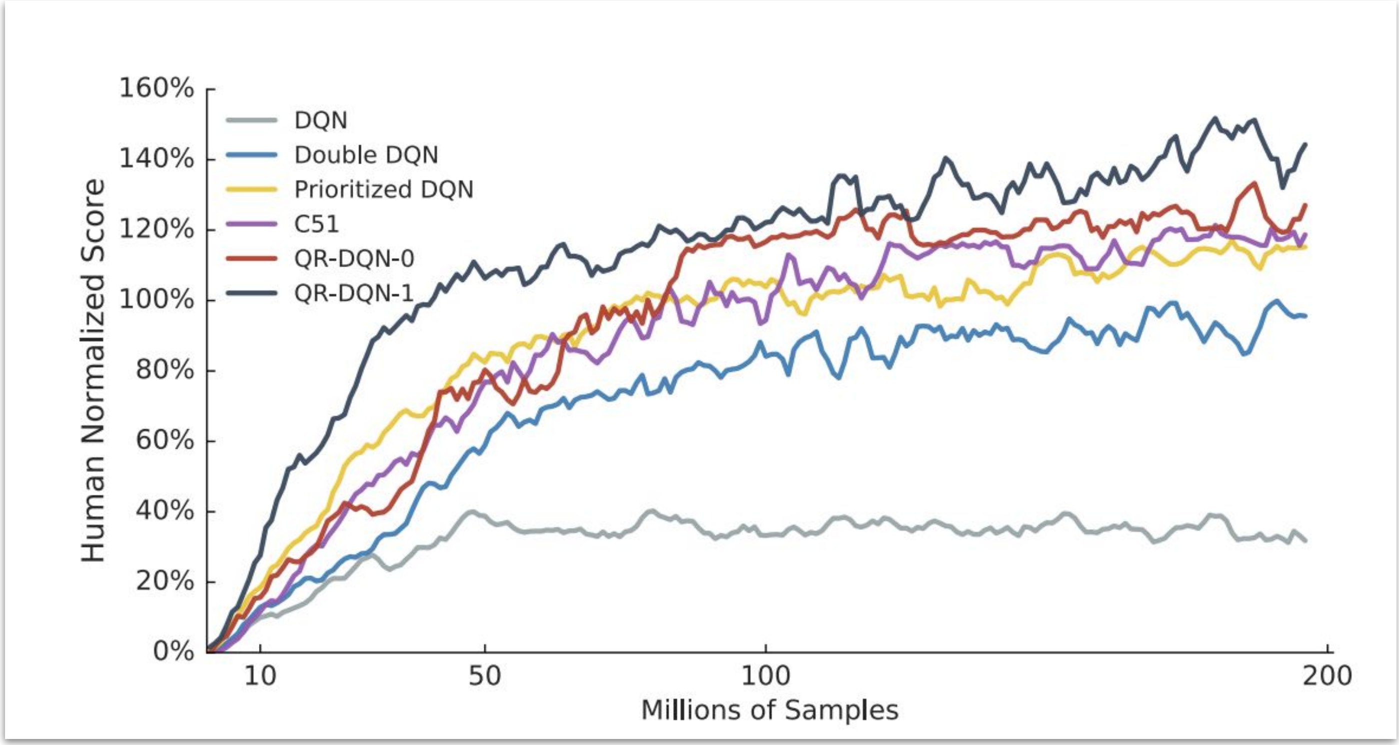
Quantile Huber Loss

(similar to the previous loss for quantile regression, but more robust)

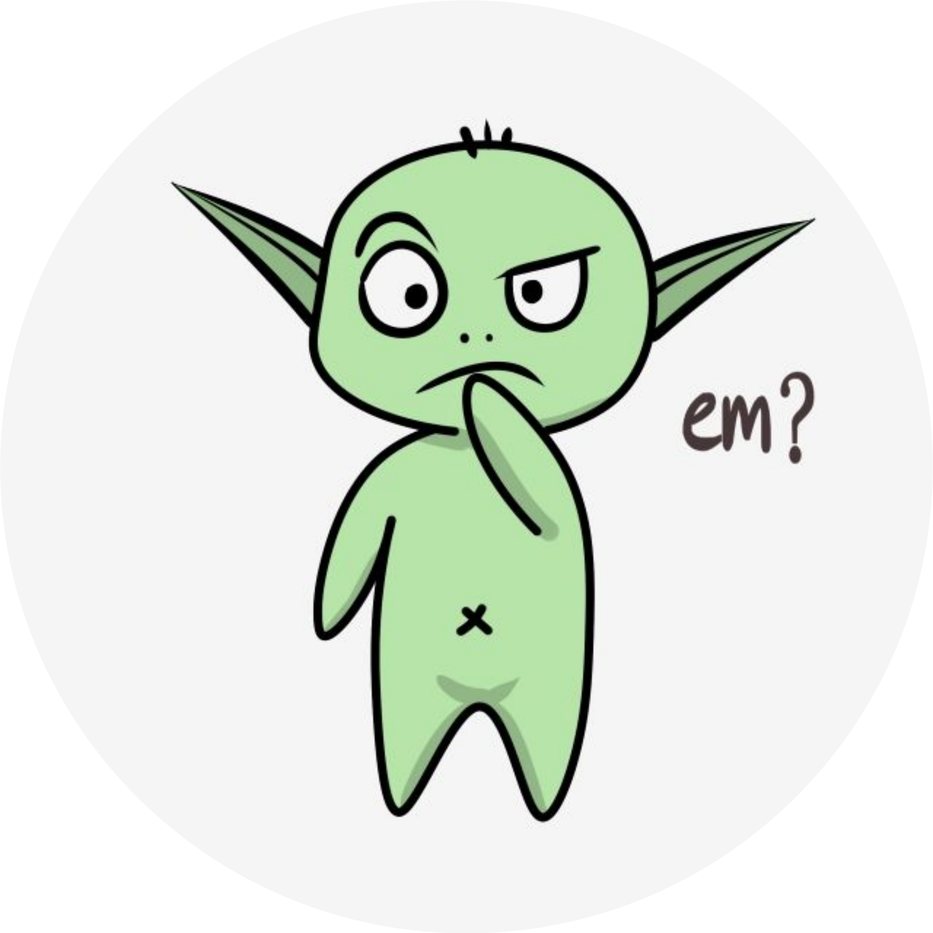


```
Algorithm 1 Quantile Regression Q-Learning
Require: $N, \kappa$
input $x, a, r, x', \gamma \in [0, 1)$
    # Compute distributional Bellman target
    $Q(x', a') := \sum_j q_j \theta_j(x', a')$
    $a^* \leftarrow \arg\max_{a'} Q(x, a')$
    $\mathcal{T}\theta_j \leftarrow r + \gamma\theta_j(x', a^*), \quad \forall j$
    # Compute quantile regression loss (Equation 10)
output $\sum_{i=1}^{N} \mathbb{E}_j \left[\rho^{\kappa}_{\hat{\tau}_i}(\mathcal{T}\theta_j - \theta_i(x, a))\right]$
```

Guess what? It works [Dabney et al., 2018]



3 seeds

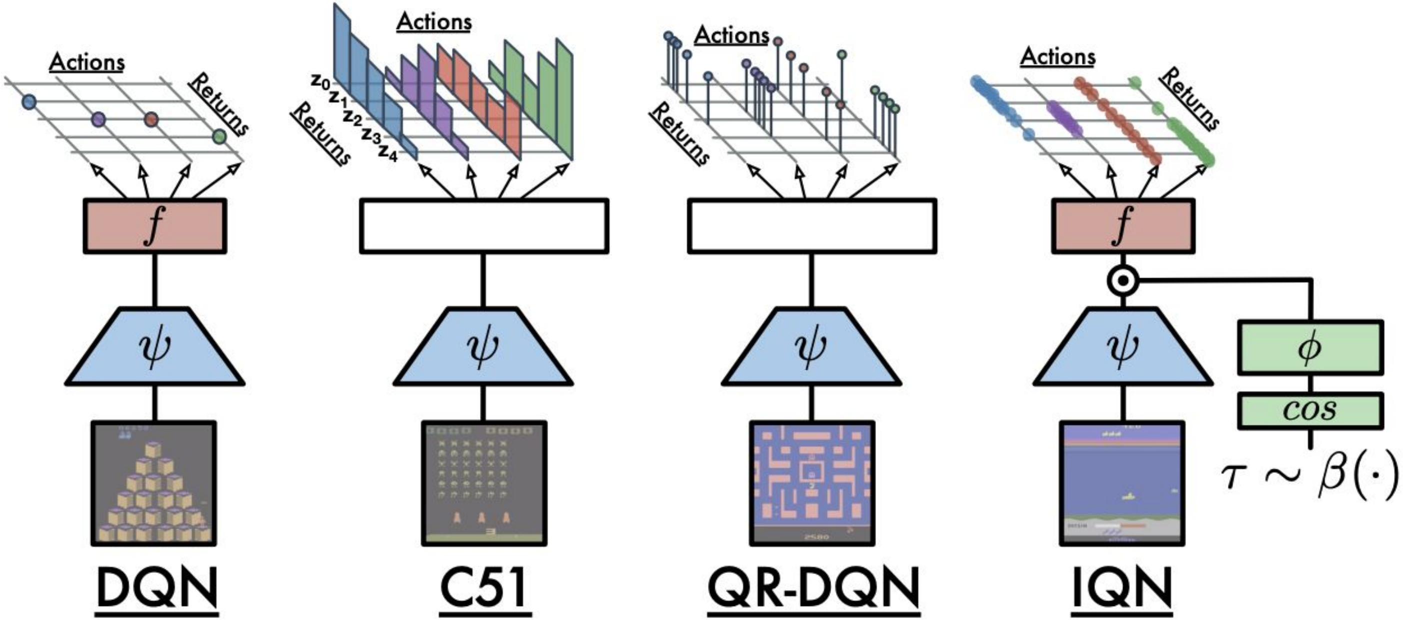


https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

IQN: Implicit Quantile Networks [Dabney et al., 2018]

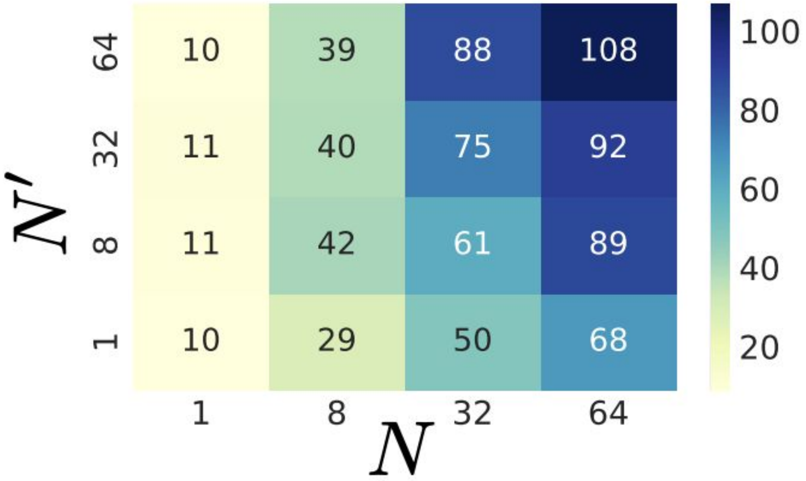
- “A simple distributional generalization of the DQN algorithm”
- Instead of “learning a discrete set of quantiles, one can learn the full quantile function, a continuous map from probabilities to returns”
- IQN is different from QR-DQN in two ways:
  - “Instead of approximating the quantile function at n fixed values of ρ we approximate it with $Z_\rho(o, a) \approx f(\psi(o), \phi(\rho))_a$ for some differentiable functions f, ψ, and ϕ.”
    - Let $f(\psi(o))_a$ be the output of DQN. All we are adding is ϕ(ρ), which is an embedding for the sample point ρ.
  - ρ, ρ’, $\tilde{\rho}$ are sampled from continuous, independent, distributions.

IQN: Implicit Quantile Networks [Dabney et al., 2018]



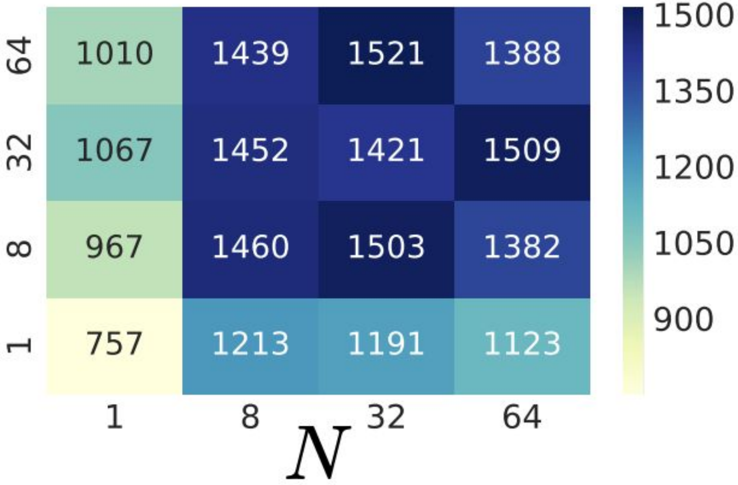
IQN: Implicit Quantile Networks [Dabney et al., 2018]

- What value of N should we consider?
- Should N = 1 resemble DQN?
  That would mean the performance benefits are due to the auxiliary task effect



	1	8	32	64
64	10	39	88	108
32	11	40	75	92
8	11	42	61	89
1	10	29	50	68

DQN (32, 253)



	1	8	32	64
64	1010	1439	1521	1388
32	1067	1452	1421	1509
8	967	1460	1503	1382
1	757	1213	1191	1123

QR-DQN (144, 1243).

Some More Atari 2600 Results [Dabney et al., 2018]

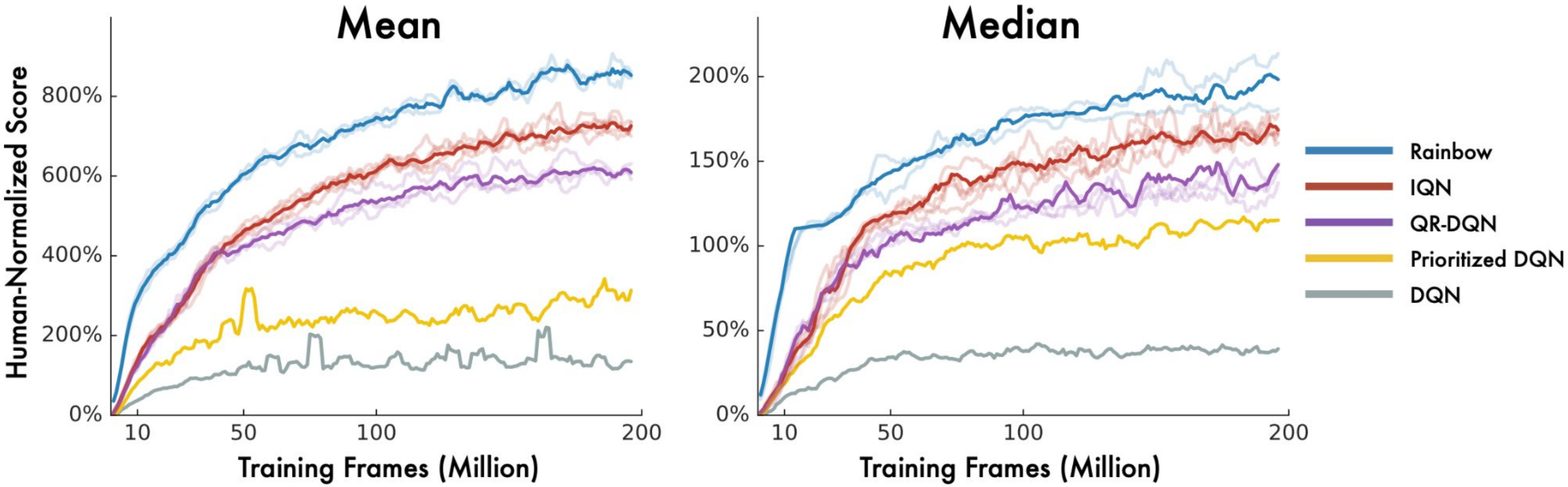
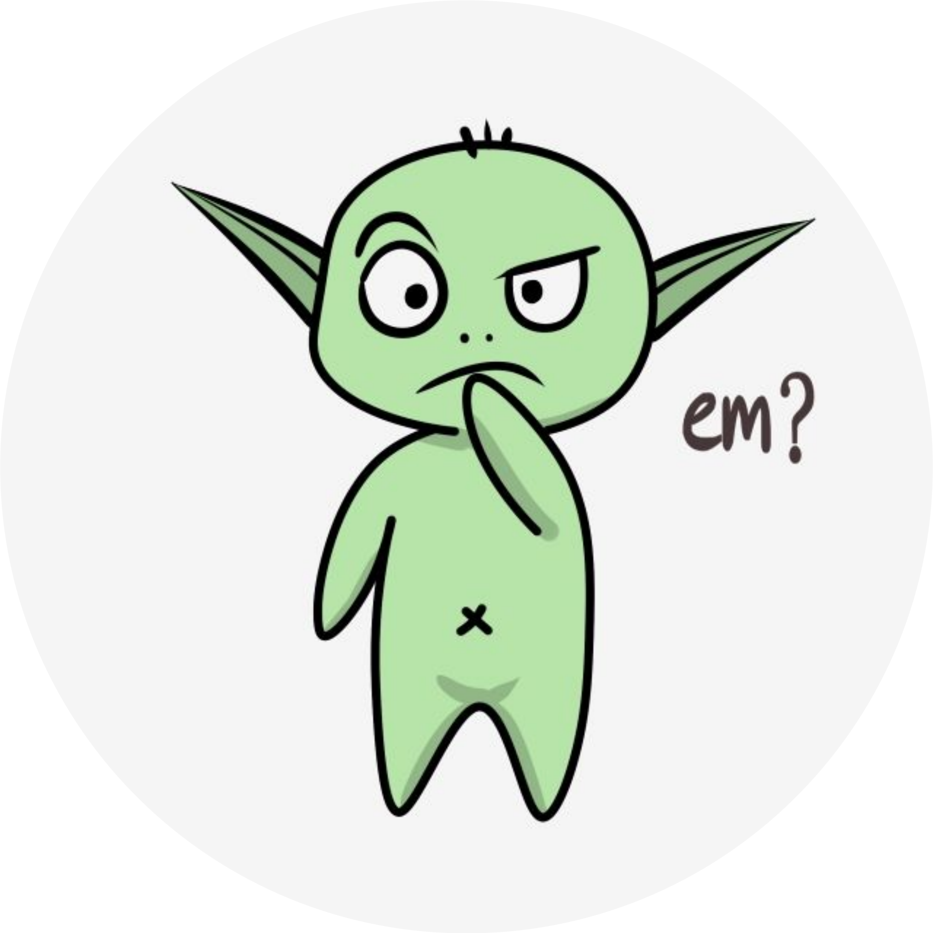


Figure 4. Human-normalized mean (left) and median (right) scores on Atari-57 for IQN and various other algorithms. Random seeds shown as traces, with IQN averaged over 5, QR-DQN over 3, and Rainbow over 2 random seeds.



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

But why does this work?

At the end of the day we are only using the expected value anyway.

A Theoretical and an Empirical Analysis of the Why [Lyle et al., 2019]

- Theoretically, “distributional RL often performs identically to expected RL”
- Empirically (in toy tasks such as Cartpole and Acrobot), “distributional reinforcement learning does not improve performance when combined with tabular representations or linear function approximation”
- “we find evidence of improved performance when combined with deep networks, suggesting that the answer lies (…) in distributional reinforcement learning’s interaction with nonlinear function approximation.”
- “the benefits of distributional reinforcement learning are primarily gained from improved learning in the earlier layers of deep neural networks, as well as in the nonlinear softmax used in C51” ← **Regularization? Auxiliary Task Effect?**

Stop Regressing! [Farebrother et al., 2024]

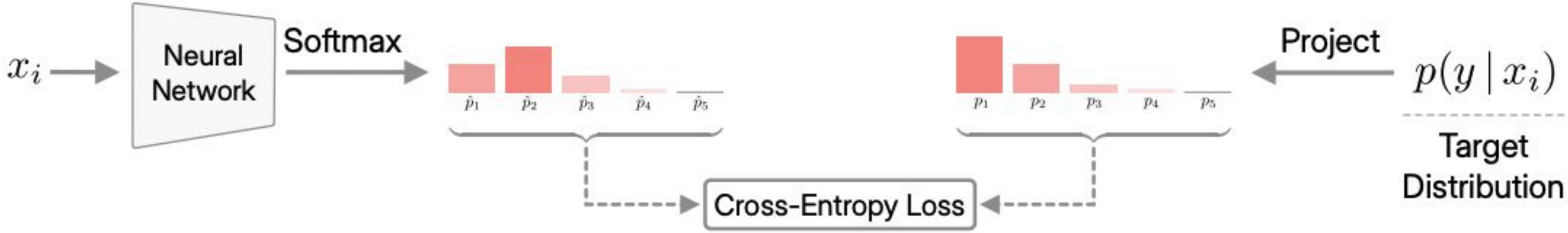


Figure 2 | Regression as Classification. Data points $\boldsymbol{x}_i$ are transformed by a neural network to produce a categorical distribution via a softmax. The prediction $\hat{y}$ is taken to be the expectation of this categorical distribution. The logits of the network are reinforced by gradient descent on the cross-entropy loss with respect to a target distribution whose mean is the regression target y_i. Figure 3 depicts three methods for constructing and projecting the target distribution in RL.

Stop Regressing! [Farebrother et al., 2024]

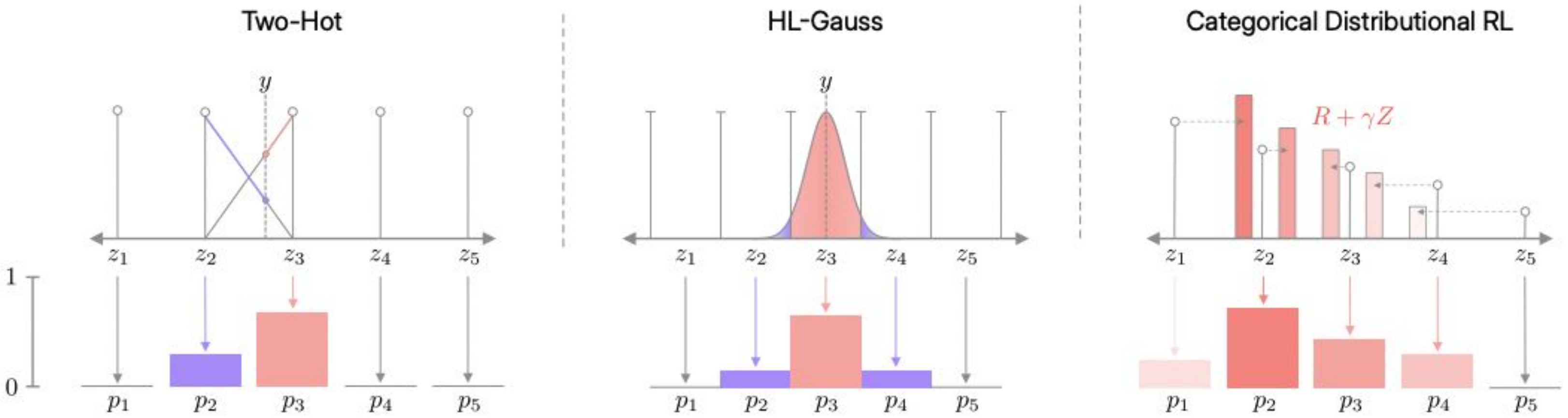
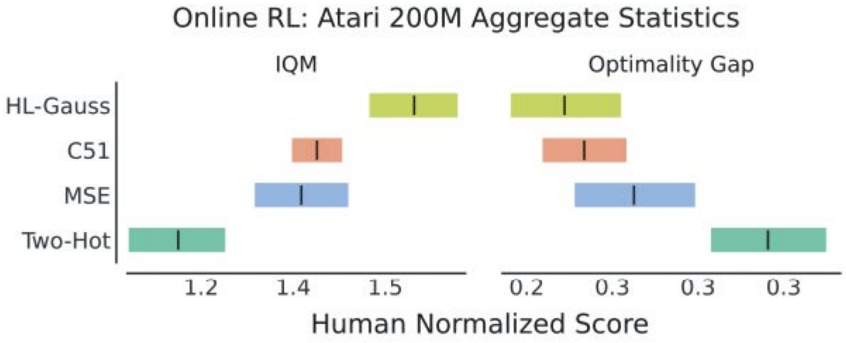
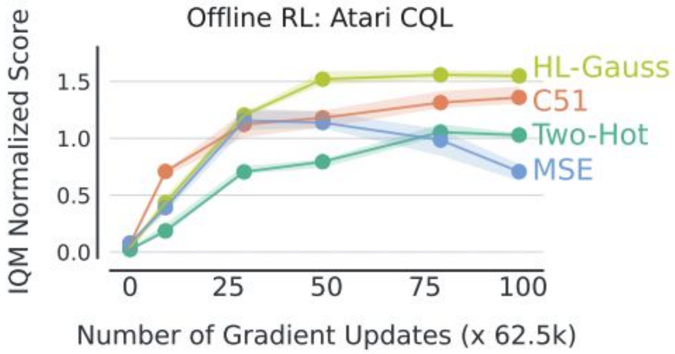
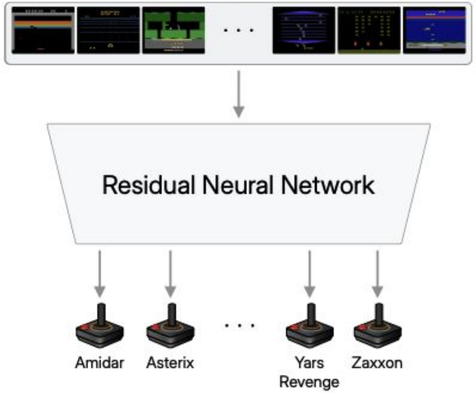

Two-Hot
HL-Gauss
Categorical Distributional RL
y
y
$R + \gamma Z$
z_1 z_2 z_3 z_4 z_5
1
0
p_1 p_2 p_3 p_4 p_5

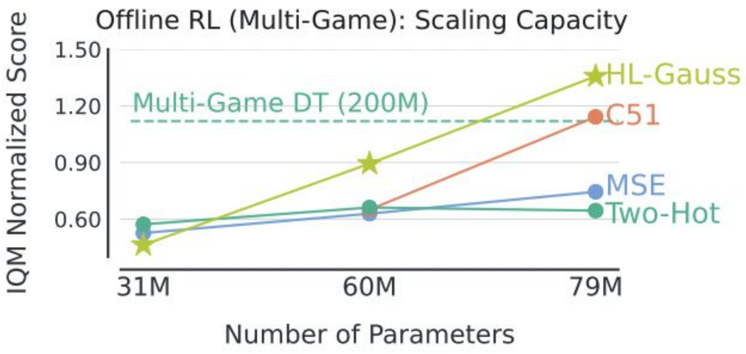
Figure 3 | Visualizing target-value categorical distribution in cross-entropy based TD learning. While Two-Hot (left, §3.1) puts probability mass on exactly two locations, HL-Gauss (middle, §3.1) distributes the probability mass to neighbouring locations (which is akin to smoothing the target value). CDRL (right, §3.2) models the categorical return distribution, distributing probability mass proportionally to neighboring locations.

Stop Regressing! [Farebrother et al., 2024]

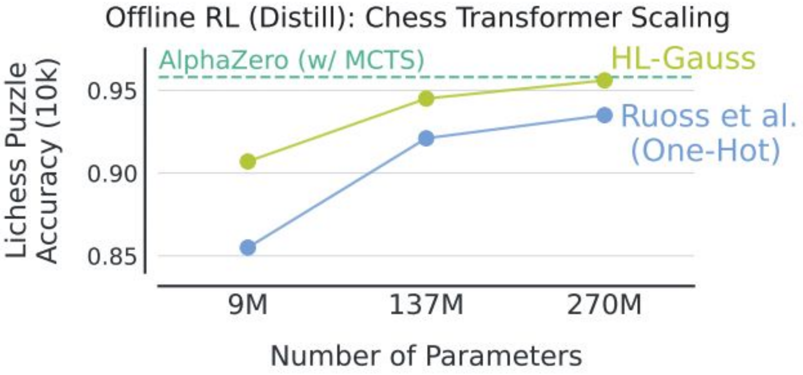


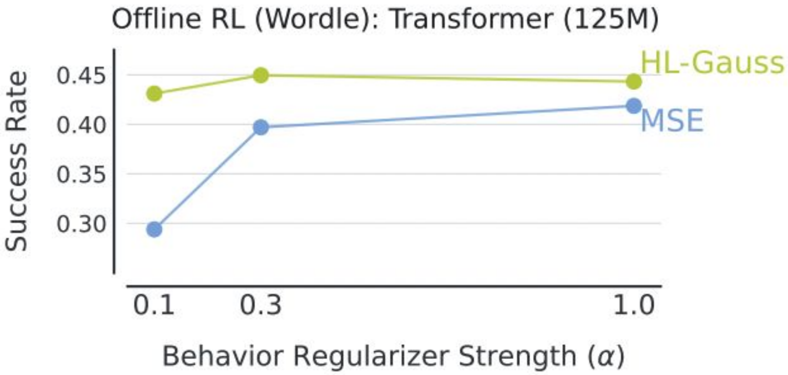


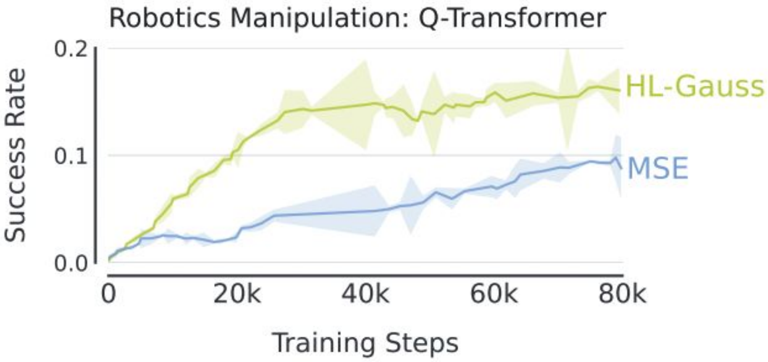


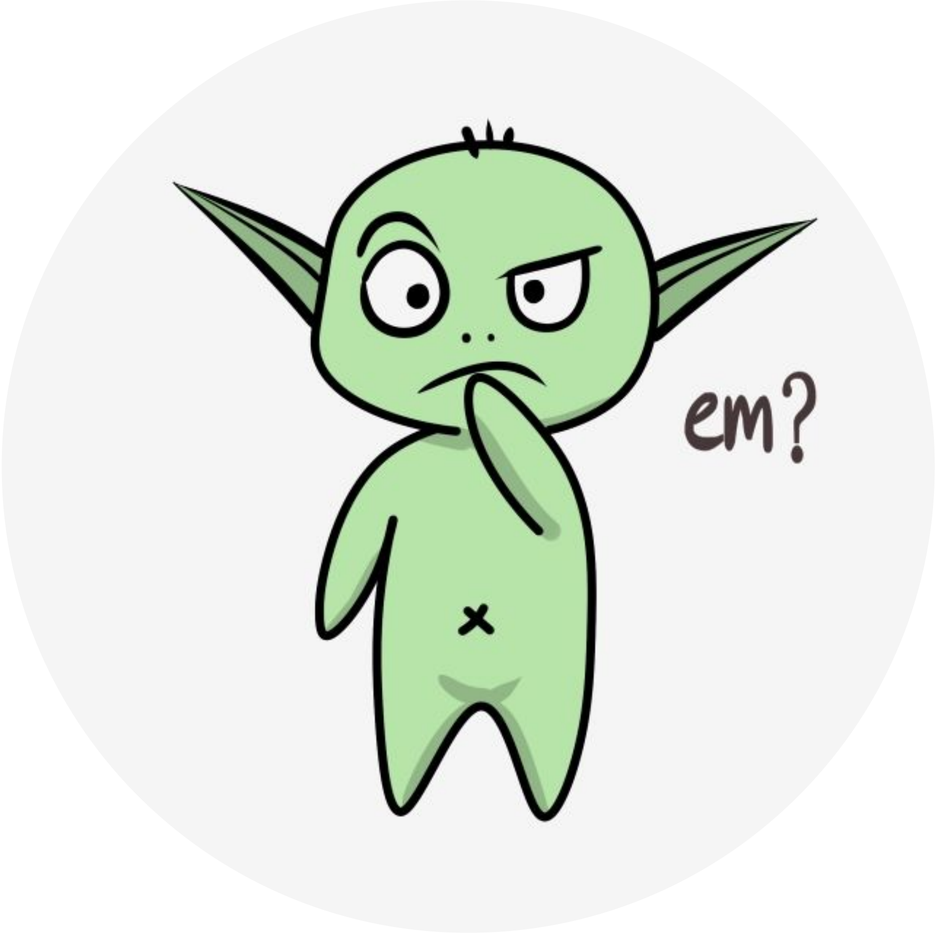


Stop Regressing! [Farebrother et al., 2024]









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

Next class

- What I plan to do:
  - Start talking about *auxiliary* objectives (instead of the *main* objective), that is: auxiliary tasks
- What I recommend YOU to do for next class:
  - Read

    - Jaderberg, M., et al.. (2017). Reinforcement Learning with Unsupervised Auxiliary Tasks. Preprint made available on November 16, 2016.

    - [Optional] Wang, H., et al. (2024). Investigating the Properties of Neural Network Representations in Reinforcement Learning. Artificial Intelligence (AIJ), 330:104100. Preprint made available on March 30, 2022.
  - Please start sending me the groups for the presentation / report, please do