REINFORCEMENT LEARNING: THEORY AND PRACTICE

Modern Landscape: Part I

Profs. Amy Zhang and Peter Stone

Logistics Questions?

- Midterm
- Project proposal feedback

Deep Q-Network (DQN)

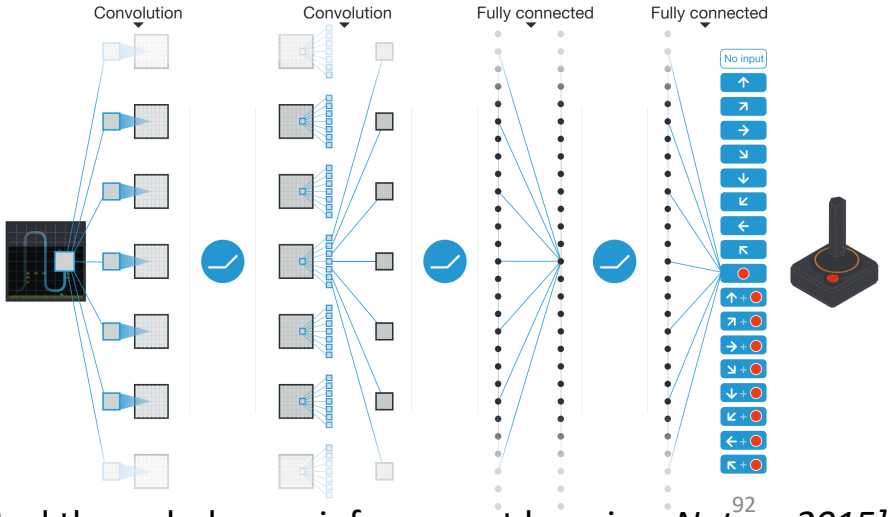
Parametric function

Q-learning

$$Q_\theta(s_t, a_t) \leftarrow r(s_t, a_t) + \gamma \max_{a'} Q_\theta(s_{t+1}, a')$$

$Q_\theta(s, a)$ now have generalization capability

How could you take the gradient w.r.t θ ?
Note that θ appears on both sides.



[*Mnih et al.* Human-level control through deep reinforcement learning, *Nature 2015*]

DQN

Q-learning

$$Q_\theta(s_t, a_t) \leftarrow r(s_t, a_t) + \gamma \max_{a'} Q_{\theta'}(s_{t+1}, a')$$

Old fixed parameters

Target network

Fixing RHS and learn θ from LHS.

[*Mnih et al.* Human-level control through deep reinforcement learning, *Nature 2015]*

Reading Questions: DQN

Karen Chen

I don't understand how using a target network in the MSBE (mean squared bellman error) loss works. If the target network is just copied from the main network(s) while the main network(s) are being optimized, how are they helpful in learning?

Emin Arslan

Can't using target networks have a negative effect on learning the Q-function since it uses an old estimation?

Reading Questions: DQN

Karen Chen

I don't understand how using a target network in the MSBE (mean squared bellman error) loss works. If the target network is just copied from the main network(s) while the main network(s) are being optimized, how are they helpful in learning?

In some sense they're not as helpful because it is a "stale" target, but it slows down value overestimation, which hurts more.

Emin Arslan

Can't using target networks have a negative effect on learning the Q-function since it uses an old estimation?

Yes, we would expect to see worse performance if the estimation is too old. Tradeoffs!

DQN

Q-learning (make the target even smoother)

Smoothing factor

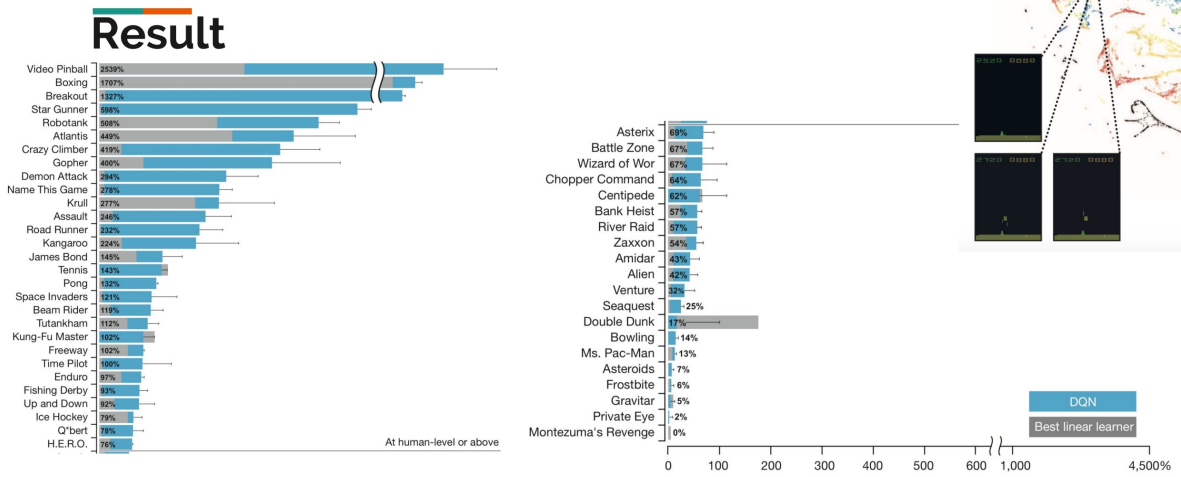
$$Q_\theta(s_t, a_t) \leftarrow (1-\alpha) Q_\theta(s_t, a_t) + \alpha \left[r(s_t, a_t) + \gamma \max_{a'} Q_{\theta'}(s_{t+1}, a') \right]$$

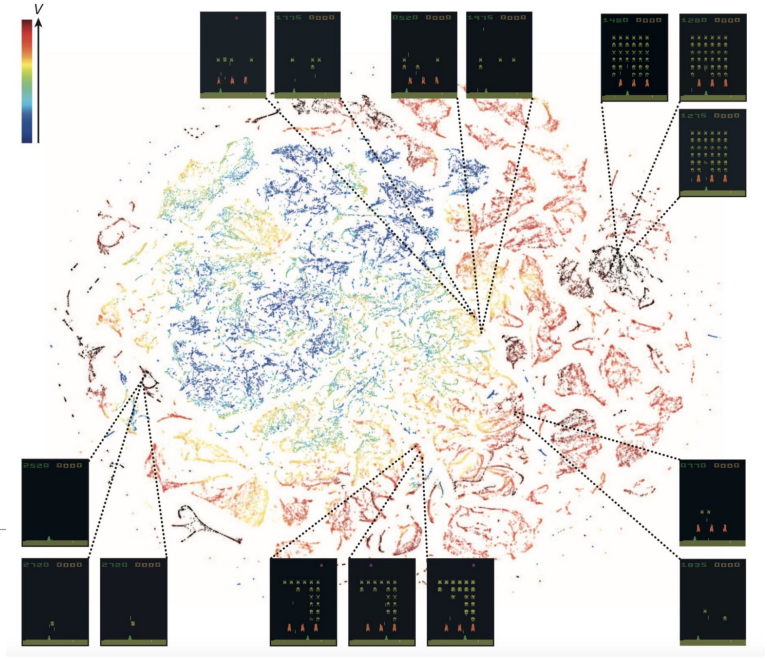
$$\Delta Q_\theta(s_t, a_t) \propto r(s_t, a_t) + \gamma \max_{a'} Q_{\theta'}(s_{t+1}, a') - Q_\theta(s_t, a_t)$$

Temporal Difference (TD) Error δ

[*Mnih et al.* Human-level control through deep reinforcement learning, *Nature 2015*]

DQN Results





t-SNE embedding

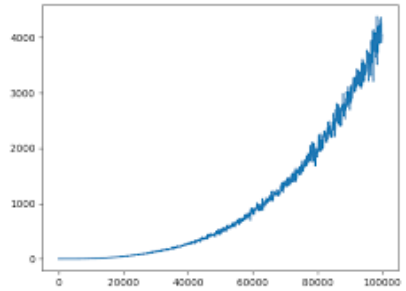
100 * (DQN score - random play score)/ (human score - random play score)

[*Mnih et al.* Human-level control through deep reinforcement learning, *Nature 2015]*

DQN Issue

- Exploding Q values

Q values



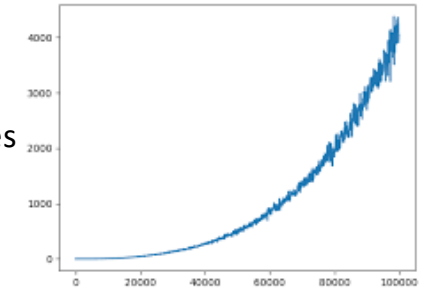
$$Q^*(s,a) = r(s,a) + \gamma \max_{a'} Q^*(s'(s,a),a')$$

[*Hado van Hasselt.* Double Q-learning. NeurIPS 2010]

DQN Issue

- Exploding Q values
- Fix: Double Q-Learning

Q values



Algorithm 1 Double Q-learning

Initialize Q^A,Q^B,s
repeat
  Choose a, based on $Q^A(s,\cdot)$ and $Q^B(s,\cdot)$, observe r, s'
  Choose (e.g. random) either UPDATE(A) or UPDATE(B)
  if UPDATE(A) **then**
    Define $a^* = \arg\max_a Q^A(s',a)$
    $Q^A(s,a) \leftarrow Q^A(s,a) + \alpha(s,a)\left(r + \gamma Q^B(s',a^*) - Q^A(s,a)\right)$
  else if UPDATE(B) **then**
    Define $b^* = \arg\max_a Q^B(s',a)$
    $Q^B(s,a) \leftarrow Q^B(s,a) + \alpha(s,a)(r + \gamma Q^A(s',b^*) - Q^B(s,a))$
  end if
  $s \leftarrow s'$
until end

- Use two Q networks instead of one to reduce bias.
  - One model get the optimal action
  - The other returns the Q value.

[*Hado van Hasselt.* Double Q-learning. NeurIPS 2010]

Distributional Reinforcement Learning




Marc Bellemare, Will Dabney, Georg Ostrovski, Mark Rowland, Rémi Munos

DeepMind

Distributional-RL

Shows interesting interactions between RL and deep-learning

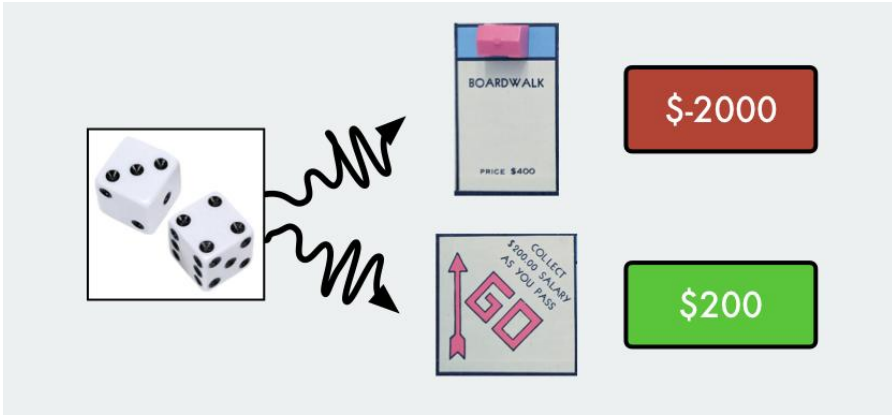
Outline:

- The idea of distributional-RL
- The theory
- How to represents distributions?
- Neural net implementation
- Results
- Why does this work?

Random immediate reward

Expected immediate reward

$$\mathbb{E}[R(x)] = \frac{1}{36} \times (-2000) + \frac{35}{36} \times (200) = 138.88$$



Random variable reward:

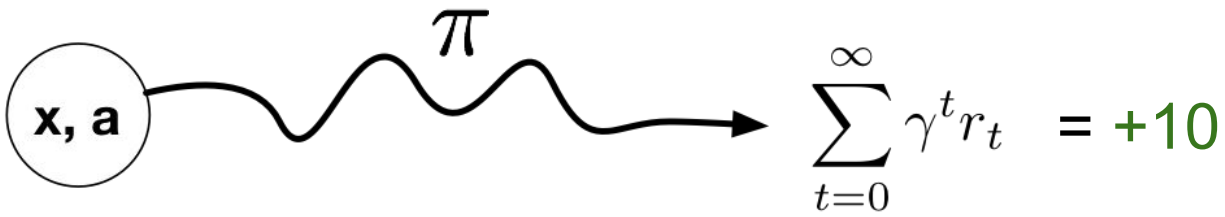
$$R(x) = \begin{cases} -2000 \text{ w.p. } 1/36 \\ 200 \text{ w.p. } 35/36 \end{cases}$$

The return = sum of future discounted rewards

$$R(x_0) + \gamma R(x_1) + \gamma^2 R(x_2) \dots$$

- Returns are often complex, multimodal
- Modelling the expected return hides this intrinsic randomness
- Model all possible returns!

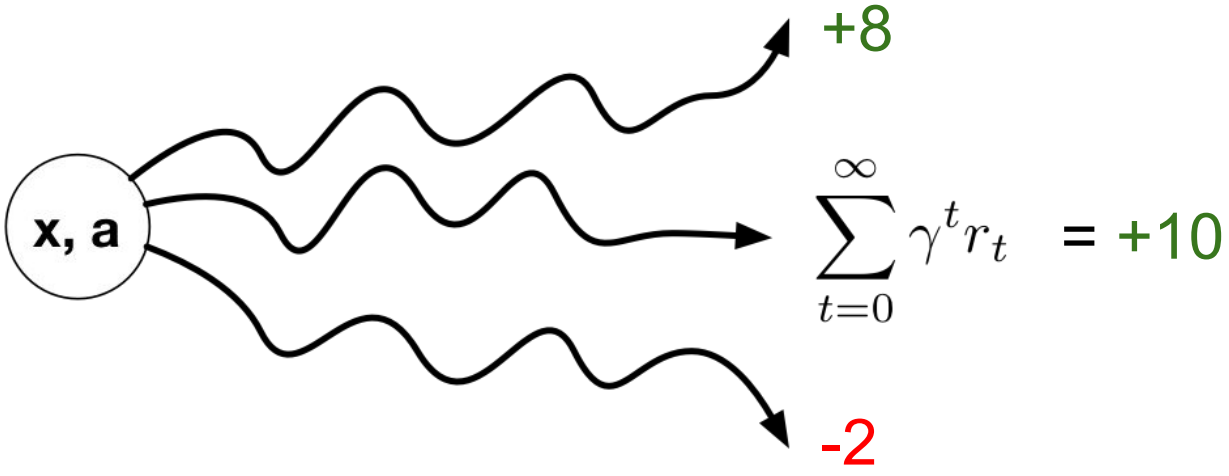
The r.v. Return $Z^{\pi}(x, a)$



Captures intrinsic randomness from:

- Immediate rewards
- Stochastic dynamics
- Possibly stochastic policy

The r.v. Return $Z^\pi(x,a)$



$$Z^\pi(x,a) = \sum_{t\geq 0} \gamma^t r(x_t,a_t)\Big|_{x_0=x,a_0=a,\pi}$$

The expected Return

The value function $Q^\pi(x,a) = \mathbb{E}[Z^\pi(x,a)]$

Satisfies the Bellman equation

$$Q^\pi(x,a) = \mathbb{E}[r(x,a) + \gamma Q^\pi(x',a')]$$

$$\text{where } x' \sim p(\cdot|x,a) \text{ and } a' \sim \pi(\cdot|x')$$

Distributional Bellman equation?

We would like to write a Bellman equation for the distributions:

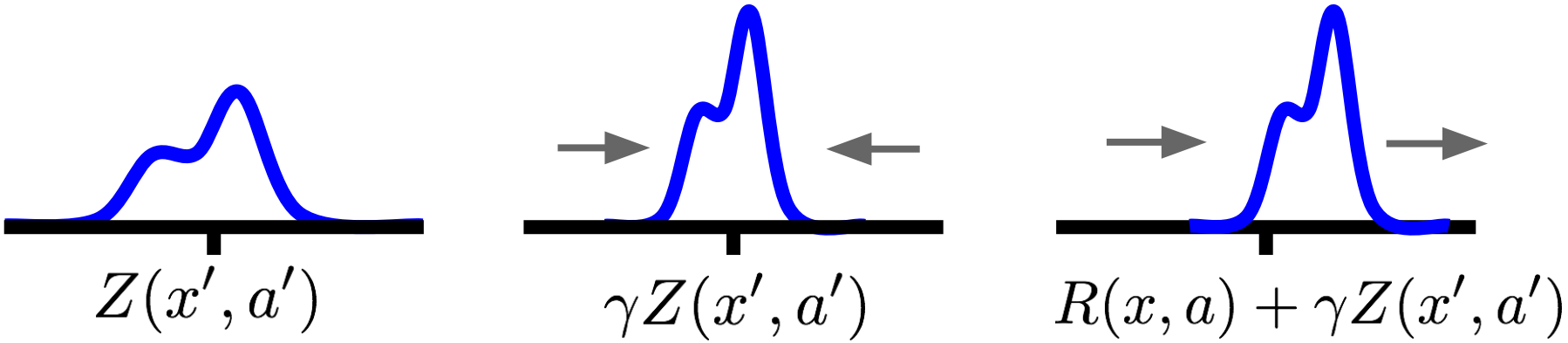
$$Z^\pi(x,a) \overset{D}{=} R(x,a) + \gamma Z^\pi(x',a')$$

$$\text{where } x' \sim p(\cdot|x,a) \text{ and } a' \sim \pi(\cdot|x')$$

Does this equation make sense?

Distributional Bellman operator

$$T^{\pi} Z(x, a) = R(x, a) + \gamma Z(x', a')$$



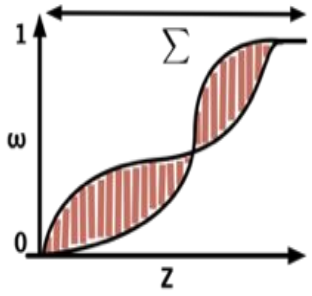
Does there exists a fixed point?

Properties

Theorem [Rowland et al., 2018]

T^π is a contraction in Cramer metric

$$\ell_2(X,Y) = \left(\int_{\mathbb{R}} \left(F_X(t) - F_Y(t)\right)^2 dt \right)^{1/2}$$

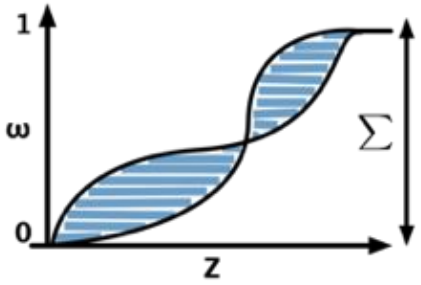


Cramer

Theorem [Bellemare et al., 2017]

T^π is a contraction in Wasserstein metric,

$$w_p(X,Y) = \left(\int_{\mathbb{R}} \left(F_X^{-1}(t) - F_Y^{-1}(t)\right)^p dt \right)^{1/p}$$



Wasserstein

(but not in KL neither in total variation)
Intuition: the size of the support shrinks.

Reading Questions: Distributional Q learning

Sam Ziegelbein

Why is the Wasserstein metric used in the distributional paper? Is it just because this metric is convenient for developing the theory in the paper, or is there a deeper reason why this metric might be preferable?

Arko Banerjee

It seems like the main aim of this paper is instead of learning a value function, it learns a value distribution function, where Z(s,a) returns a distribution over the reals. The bulk of the mathematics seems to be to show that regular iteration is guaranteed to converge (which they phrase by saying that T^pi is a gamma-contraction in d). What's special about the Wasserstein metric? It seems somewhat arbitrary.

Anthony Bao

Q1 [Bellemare et al]: What are the high-level implications of the Bellman operator T^pi not being a contraction in TV distance, KL divergence, or Kolmogorov distance? Does this mean that we don't have any provable guarantees for minimizing any other metric/divergence over distributions, and if we wanted to do that, how could we control the instability?

Reading Questions: Distributional Q learning

Wasserstein Metric

Theoretical results show that the distributional Bellman operator using Wasserstein metric will converge to a solution, unlike other metrics like KL divergence or total variation distance.

Distributional dynamic programming

Thus T^π has a unique fixed point, and it is Z^π

Policy evaluation:
For a given policy π, iterate $Z \leftarrow T^\pi Z$ converges to Z^π

Distributional dynamic programming

Thus T^π has a unique fixed point, and it is Z^π

Policy evaluation:
For a given policy π, iterate $Z \leftarrow T^\pi Z$ converges to Z^π

Policy iteration:

- For current policy π_k, compute Z^{π_k}
- Improve policy

$$\pi_{k+1}(x) = \arg\max_a \mathbb{E}[Z^{\pi_k}(x, a)]$$

Does Z^{π_k} converge to the return distribution for the optimal policy?

Distributional Bellman optimality operator

$$TZ(x,a) \overset{D}{=} r(x,a) + \gamma Z(x', \pi_Z(x'))$$

$$\text{where } x' \sim p(\cdot|x,a) \text{ and } \pi_Z(x') = \arg\max_{a'} \mathbb{E}[Z(x',a')]$$

Is this operator a contraction mapping?

Distributional Bellman optimality operator

$$TZ(x,a) \overset{D}{=} r(x,a) + \gamma Z(x', \pi_Z(x'))$$

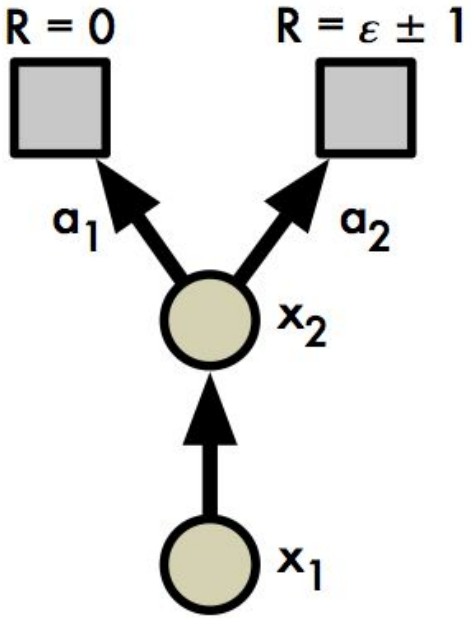
$$\text{where } x' \sim p(\cdot|x,a) \text{ and } \pi_Z(x') = \arg\max_{a'} \mathbb{E}[Z(x',a')]$$

Is this operator a contraction mapping?

No!

It's not even continuous

The dist. opt. Bellman operator is not smooth



Consider distributions Z_ϵ

If $\varepsilon > 0$ we back up a bimodal distribution

If $\varepsilon < 0$ we back up a Dirac in 0

Thus the map $Z_\epsilon \mapsto TZ_\epsilon$ is not continuous

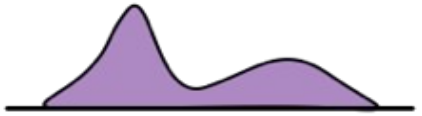
Distributional Bellman optimality operator

Theorem [Bellemare et al., 2017]

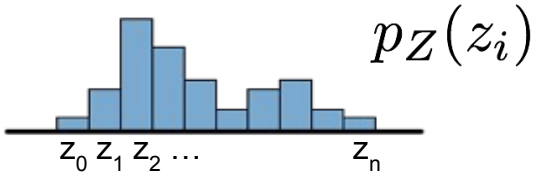
if the optimal policy is unique, then the iterates $Z_{k+1} \leftarrow TZ_k$ converge to Z^{π^*}

Intuition: The distributional Bellman operator preserves the mean, thus the mean will converge to the optimal policy π^* eventually. If the policy is unique, we revert to iterating T^{π^*}, which is a contraction.

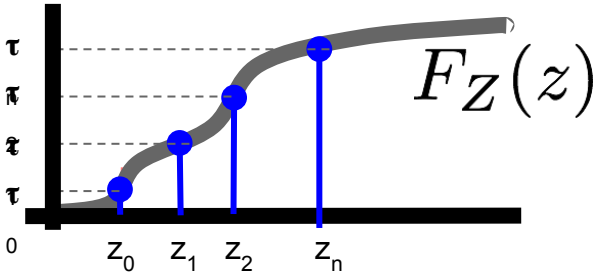
How to represent distributions?



- Categorical



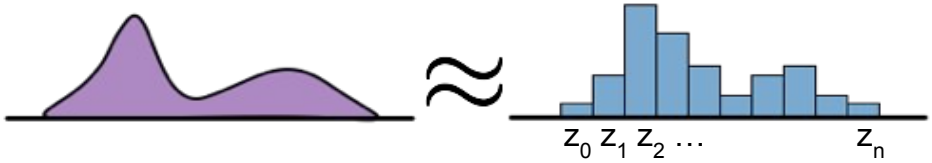
- Inverse CDF for specific quantile levels



- Parametric inverse CDF

$$\tau \mapsto F_Z^{-1}(\tau)$$

Categorical distributions

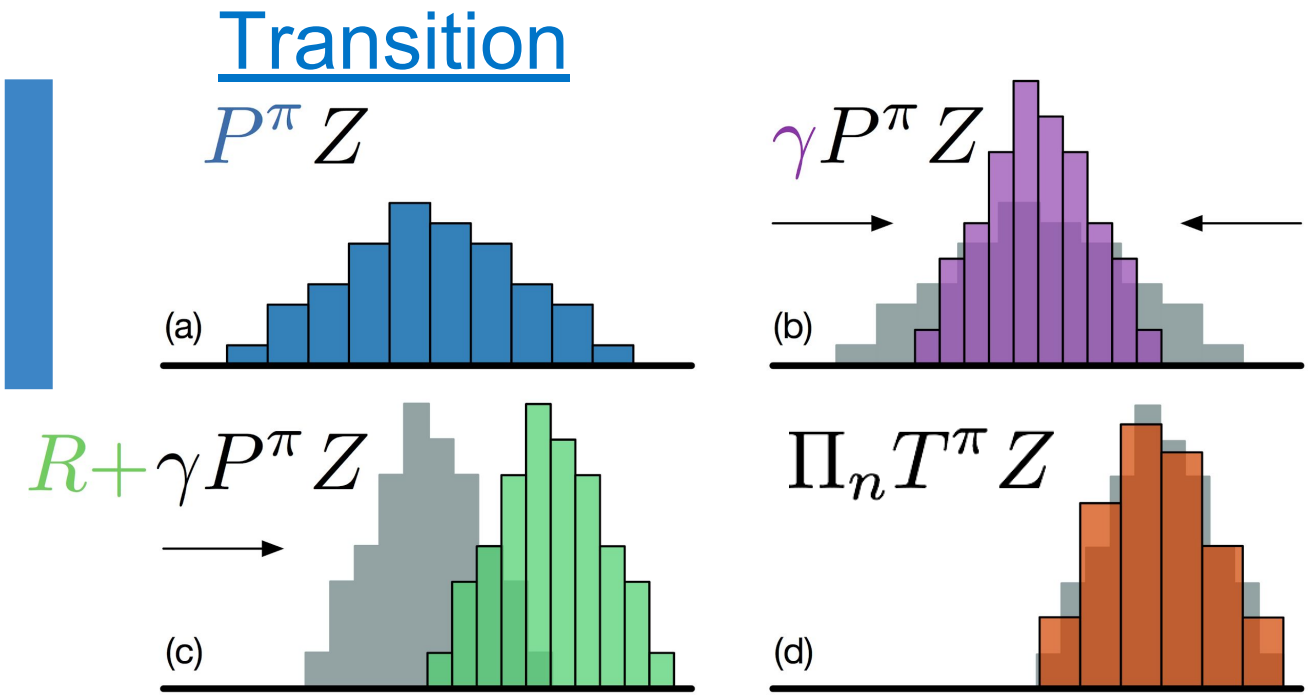


Distributions supported on a finite support $\{z_1, \ldots, z_n\}$

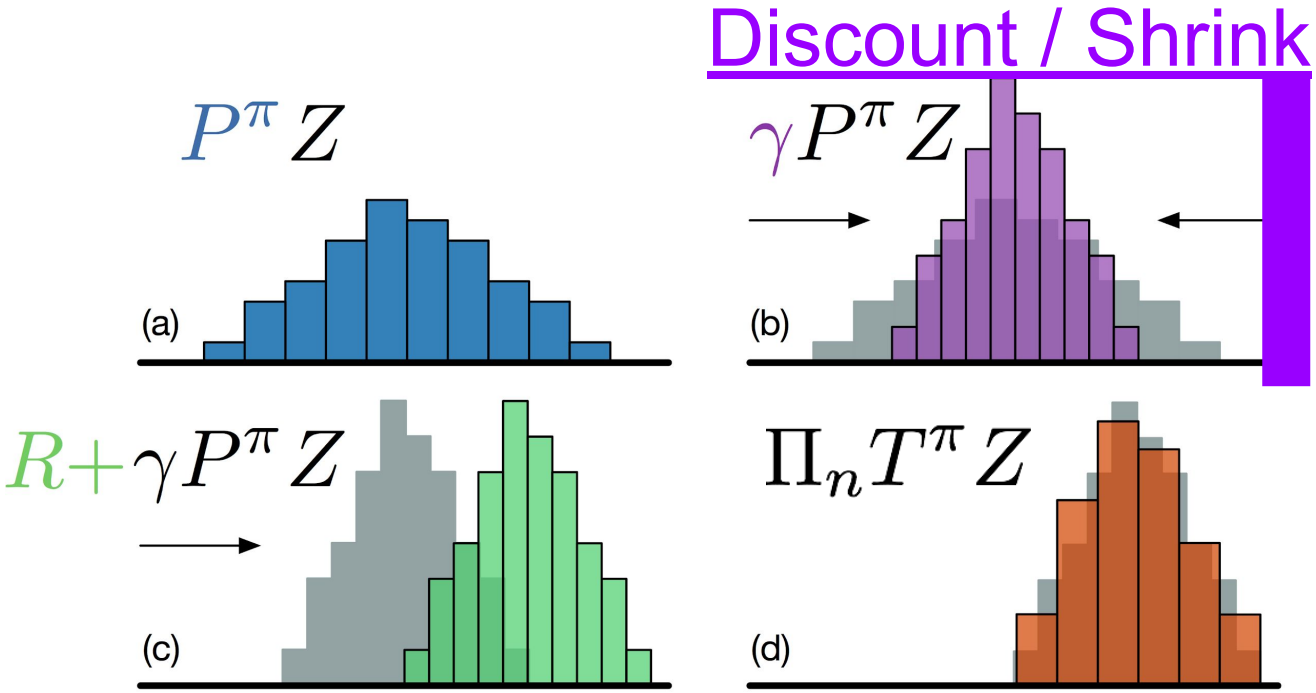
Discrete distribution $\{p_i(x, a)\}_{1 \leq i \leq n}$

$$Z(x, a) = \sum_i p_i(x, a)\delta_{z_i}$$

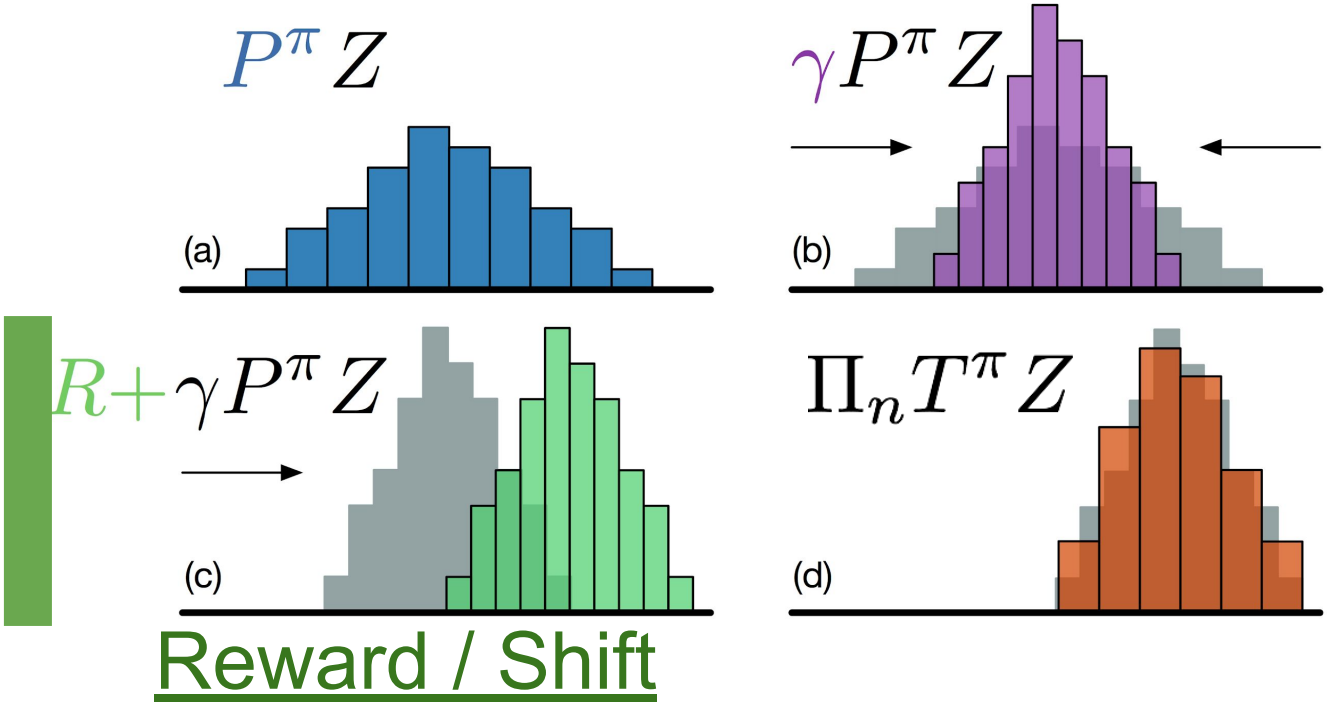
Projected Distributional Bellman Update



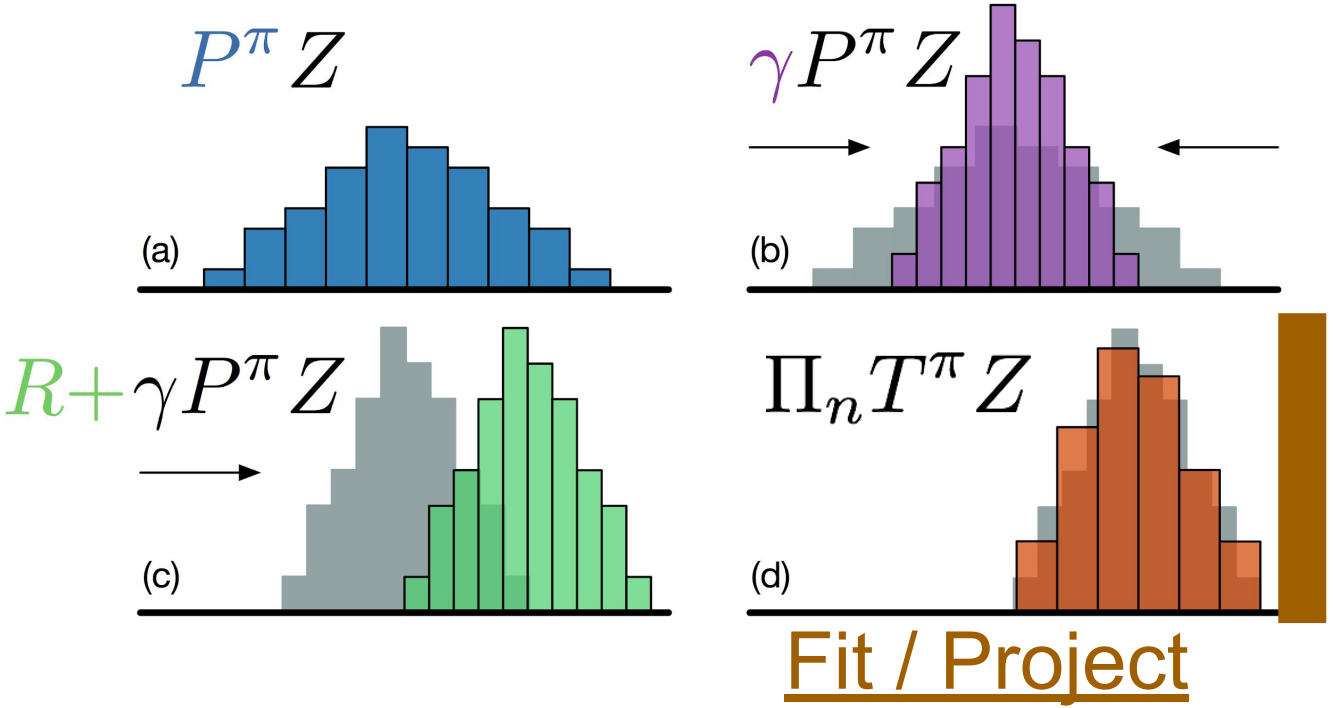
Projected Distributional Bellman Update



Projected Distributional Bellman Update



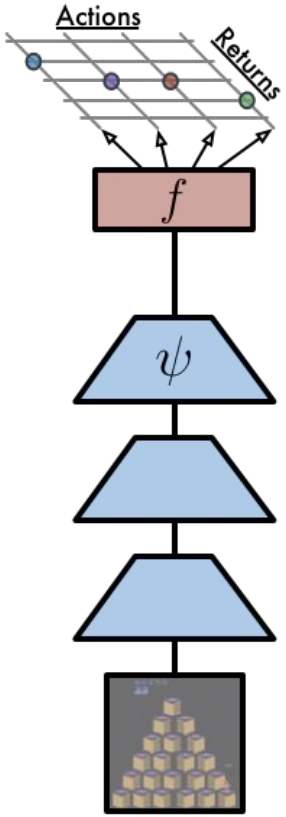
Projected Distributional Bellman Update



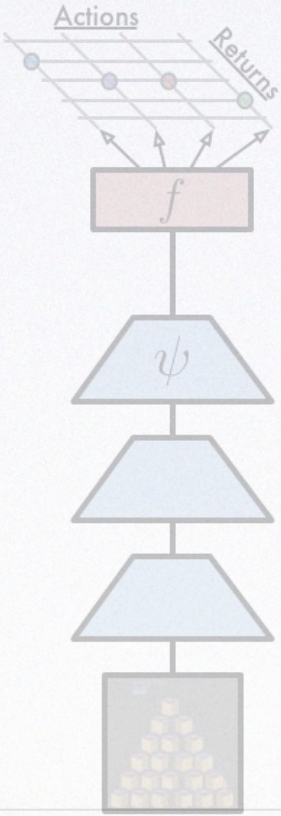
DeepRL implementation

DQN

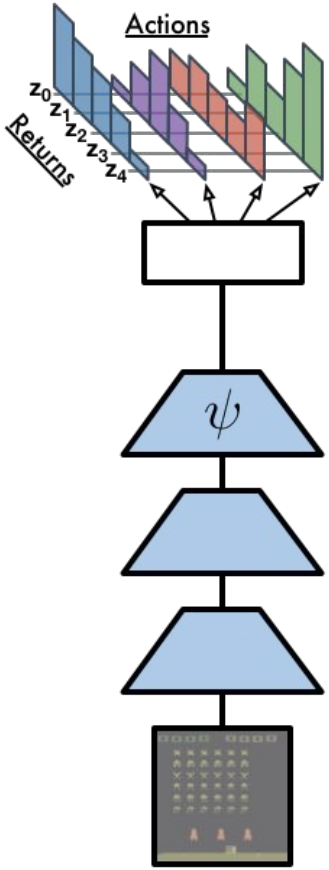
[Mnih et al., 2013]



DQN



C51

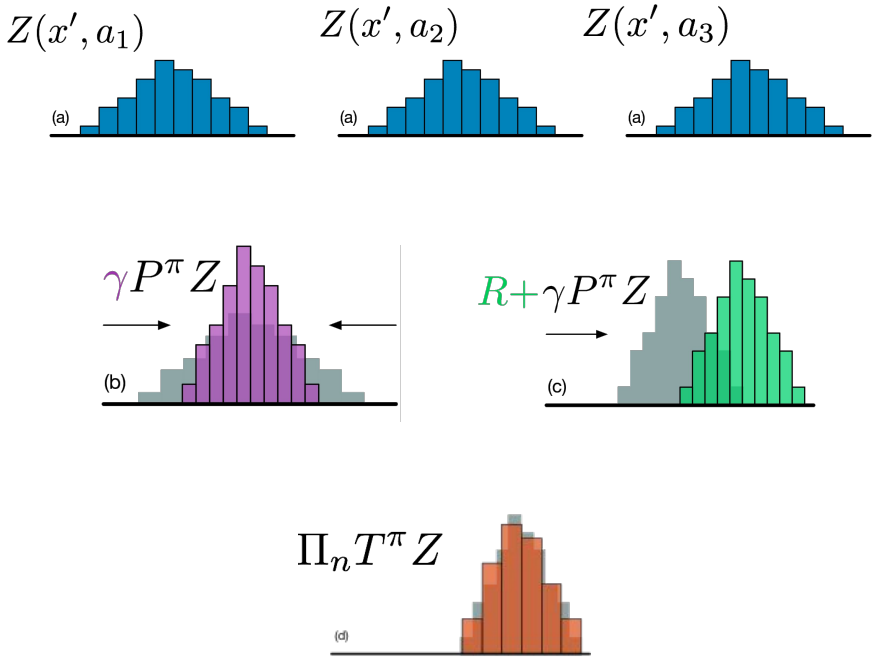


[Bellemare et al., 2017]

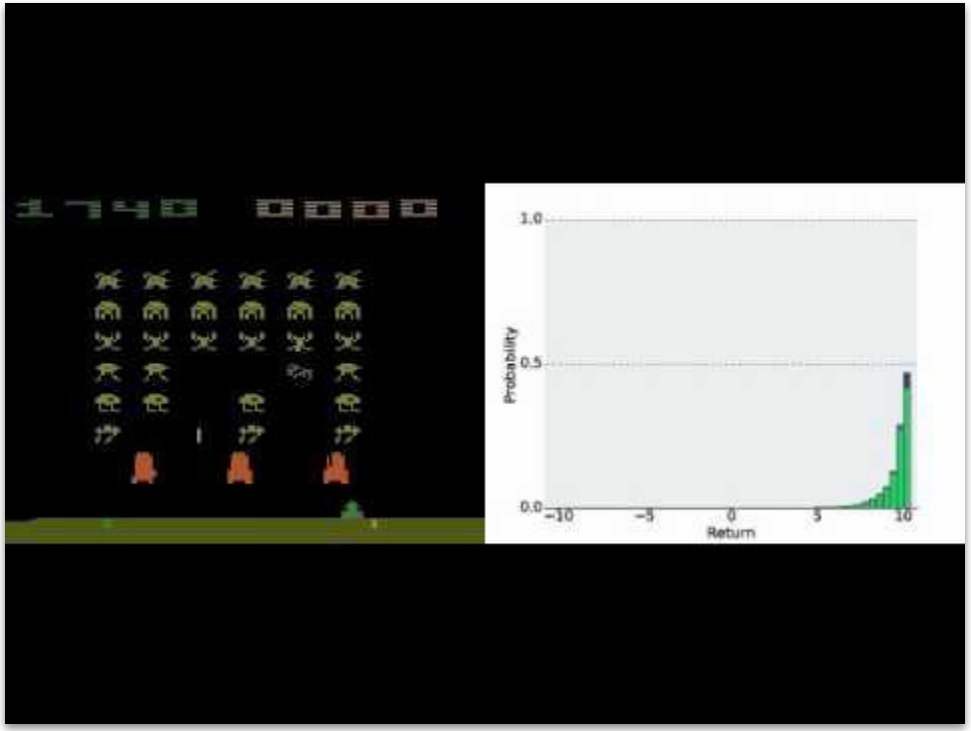
DeepMind

C51 (categorical distributional DQN)

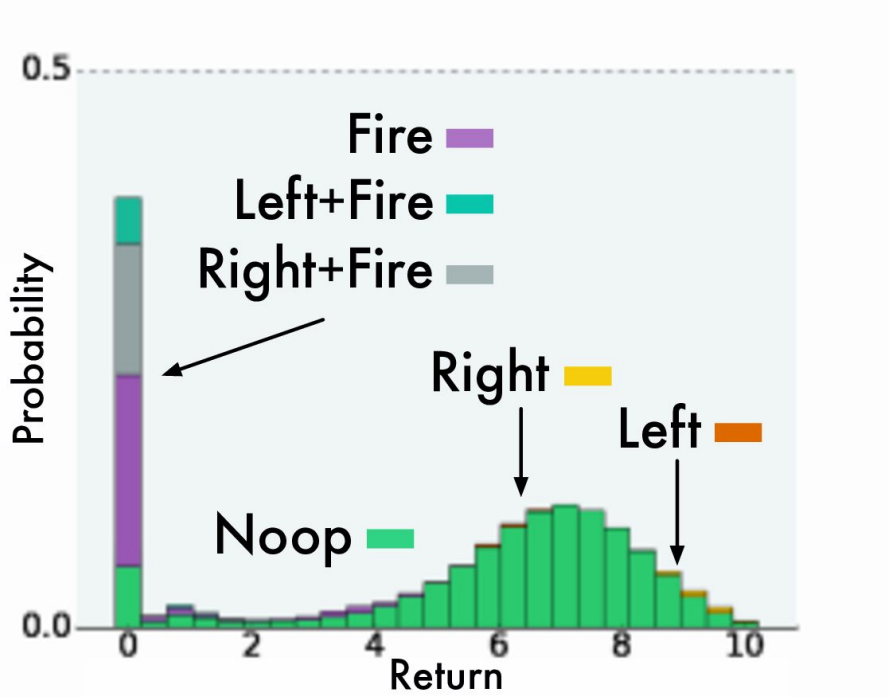
1. Transition x, a → x’
2. Select best action at x’
3. Compute Bellman backup
4. Project onto support
5. Update toward projection
   (e.g., by minimize a kl-loss)



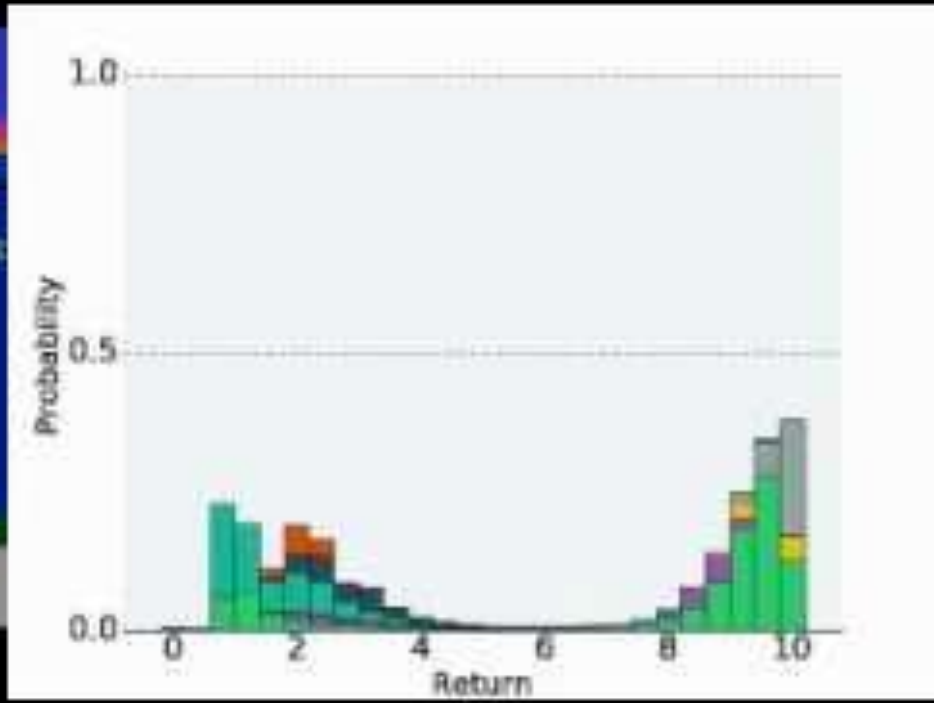
Categorical DQN

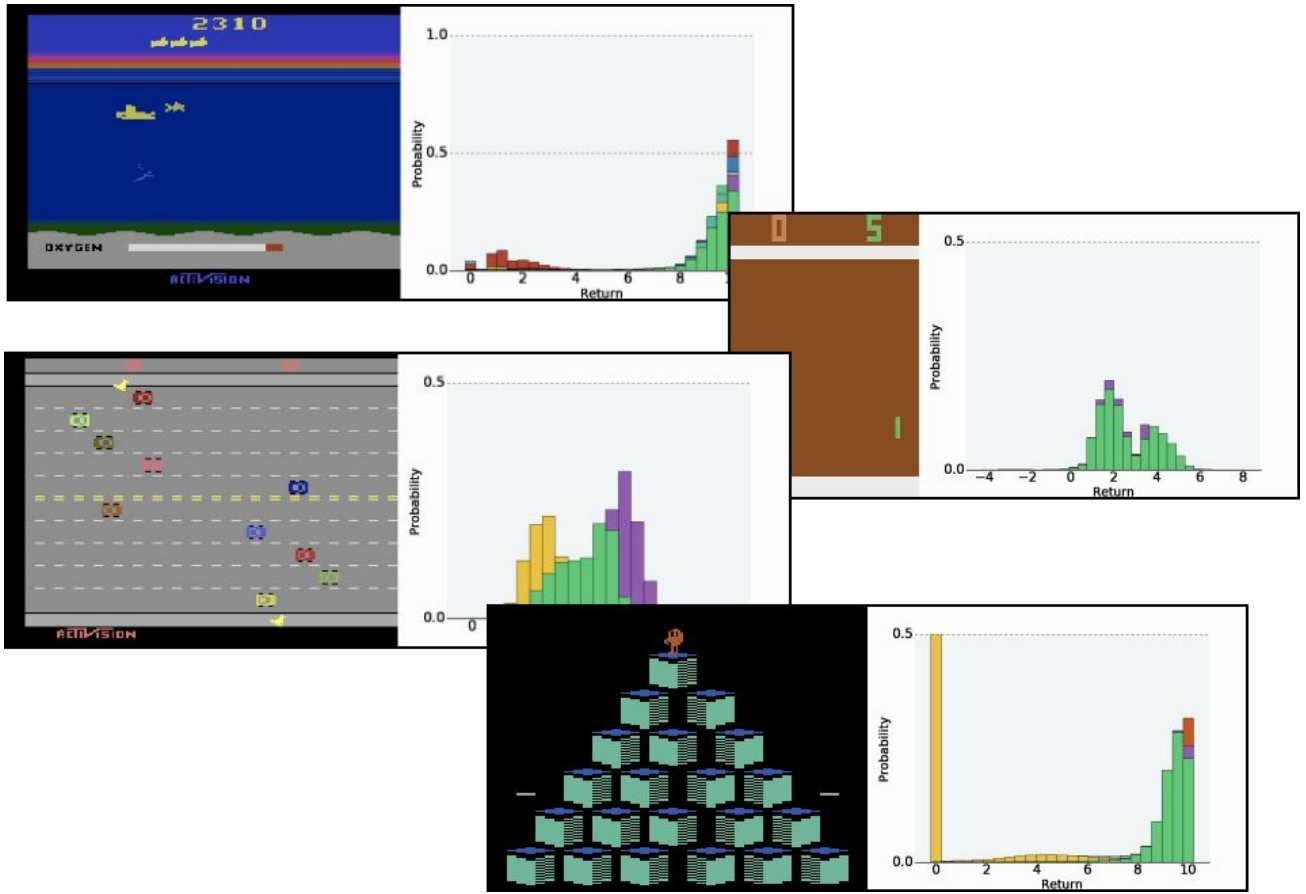


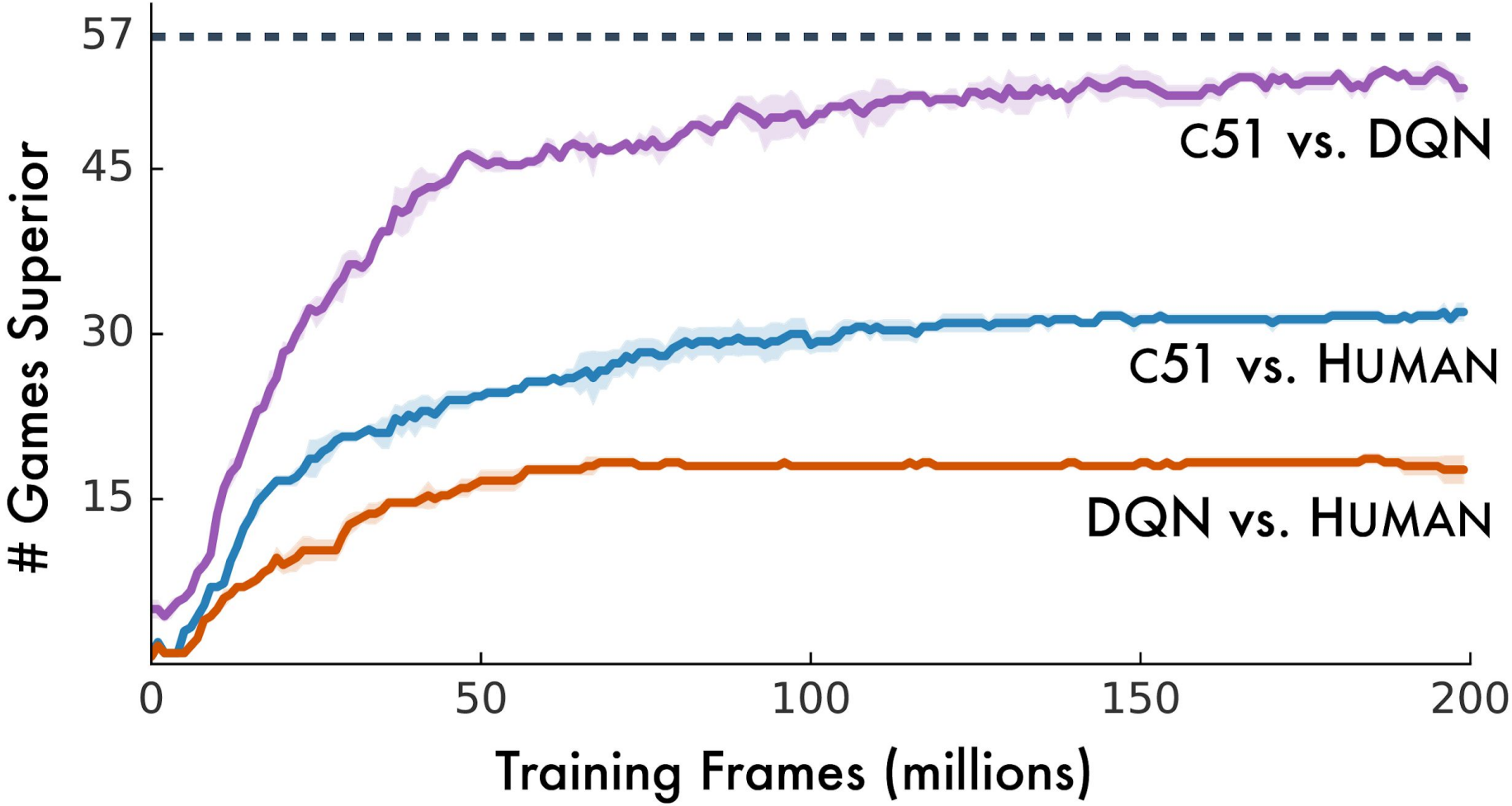
Randomness from future choices




27140
OXYGEN


1.0
0.5
0.0
Probability
0
2
4
6
8
10
Return


2310
OXYGEN
ACTIVISION
Probability
Return
1.0
0.5
0.0
0
2
4
6
8
10
−4
−2
ACTIVISION


57
45
30
15
Games Superior
C51 vs. DQN
C51 vs. HUMAN
DQN vs. HUMAN
0
50
100
150
200
Training Frames (millions)

Results on 57 games Atari 2600

	Mean	Median	>human
DQN	228%	79%	24
Double DQN	307%	118%	33
Dueling	373%	151%	37
Prio. Duel.	592%	172%	39
C51	701%	178%	40

Reading Questions: Distributional Q learning

Krystal An

What is the fundamental difference between the traditional expectation-based reinforcement learning (RL) approach and the distributional perspective introduced in this paper? Can the distributional approach to RL aid in better handling of uncertainty and risk in decision-making processes, compared to traditional methods?

William Avery

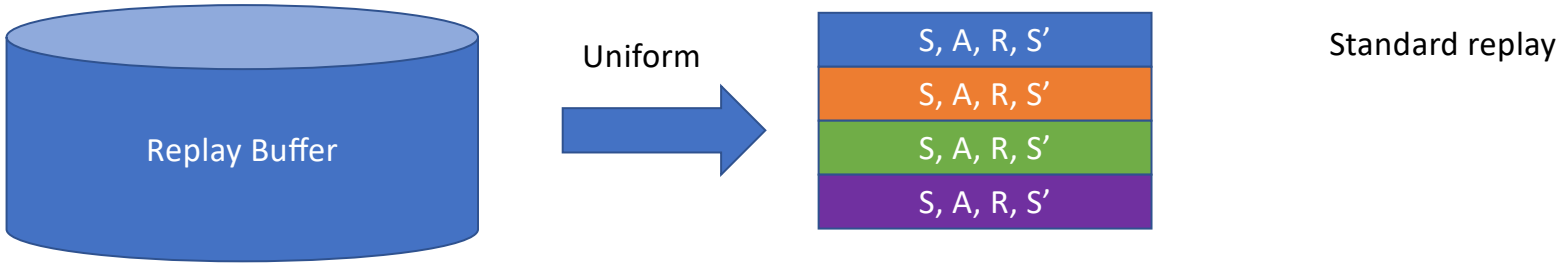
Is there a time and place for using value distributions, or is it generally better to use value distributions as opposed to maximizing expected value? It seems like with distributions, you have more information at your disposal, which might require more processing power and thus slower learning but ultimately better learning.

Alan Baade

Learning the distribution of the value function has the additional benefit of providing a stronger learning signal. How have modern generative methods, such as diffusion models, kept up with learning value function distributions, and have these had any application?

DQN Tricks

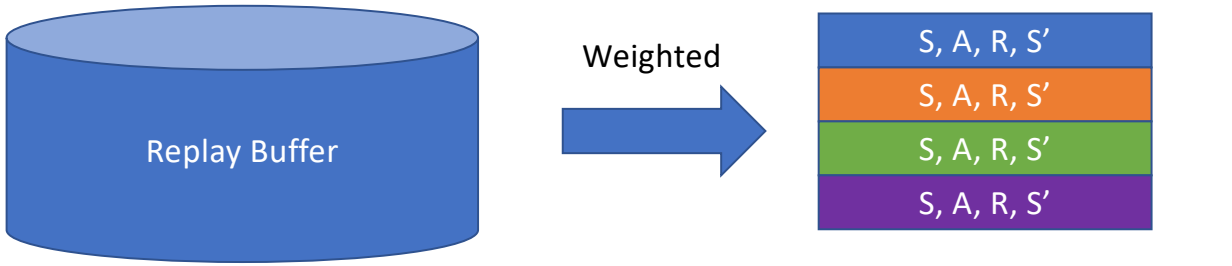
- Prioritized Experience Replay



[*Schaul et al.* Prioritized Experience Replay. ICLR 2016]

DQN Tricks

- Prioritized Experience Replay



Prioritized replay

$$P(i) = \frac{p_i^\alpha}{\sum_k p_k^\alpha}$$

Scoring mechanism: TD error

$$\delta_i = r_t + \gamma \max_{a \in \mathcal{A}} Q_{\theta^-}(s_{t+1}, a) - Q_\theta(s_t, a_t)$$

For vanilla DQN

$$\delta_i = r_t + \gamma Q_{\theta^-}(s_{t+1}, \text{argmax}_{a \in \mathcal{A}} Q_\theta(s_{t+1}, a)) - Q_\theta(s_t, a_t)$$

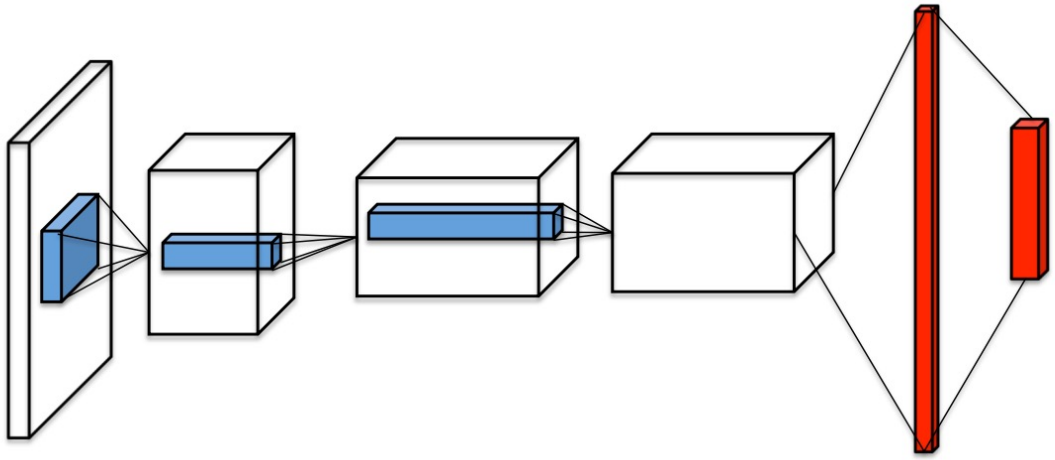
For Double DQN

[*Schaul et al.* Prioritized Experience Replay. ICLR 2016]

DQN Tricks

- Dueling networks

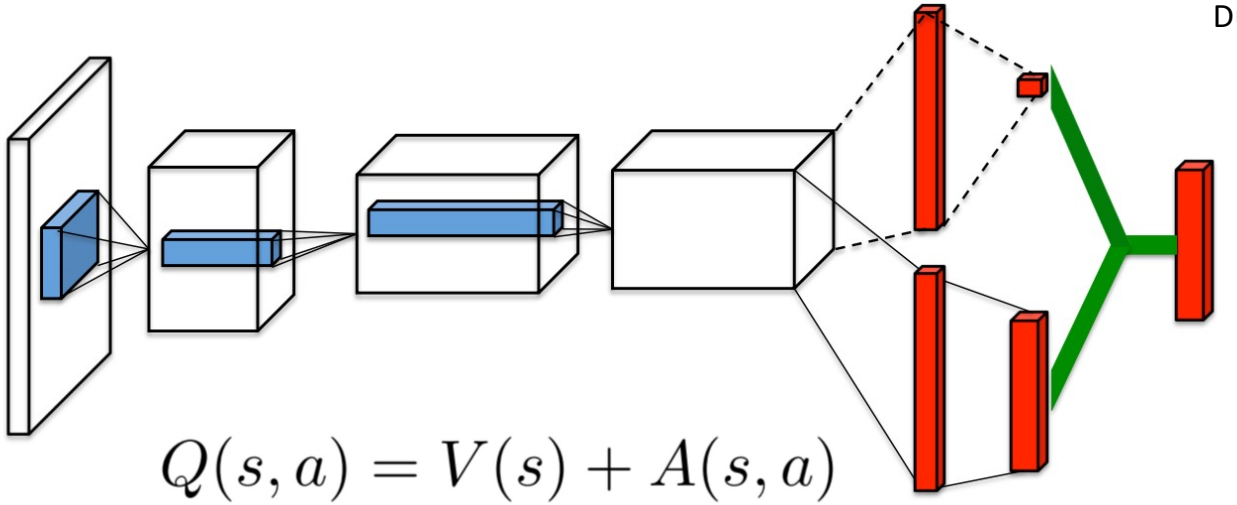
Standard Q-network



[*Wang et al.* Dueling Network Architectures for Deep Reinforcement Learning. ICML 2016]

DQN Tricks

- Dueling networks



Dueling Q-network

$$Q(s,a) = V(s) + A(s,a)$$

Where $A(s,a) := Q(s,a) - V(s)$ is the advantage function.

[*Wang et al.* Dueling Network Architectures for Deep Reinforcement Learning. ICML 2016]

DQN Tricks

- Multi-step learning (also known as n-step returns)

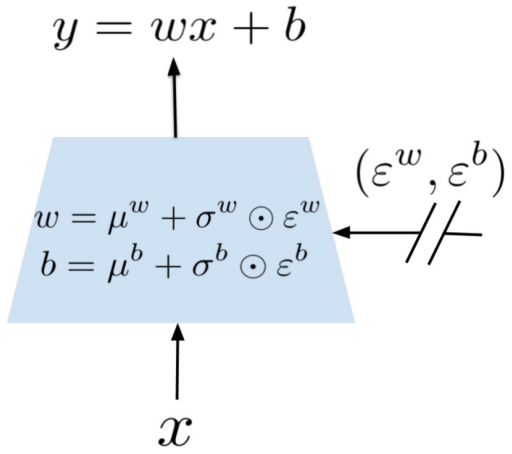
$$R_t^{(n)} \equiv \sum_{k=0}^{n-1} \gamma_t^{(k)} R_{t+k+1}$$

- Multi-step variant of DQN loss:

$$(R_t^{(n)} + \gamma_t^{(n)} \max_{a'} q_{\bar{\theta}}(S_{t+n}, a') - q_\theta(S_t, A_t))^2$$

DQN Tricks

- Noisy Nets: Noisy linear layers for exploration



[*Fortunato et al.* Noisy networks for exploration. ICLR 2018]

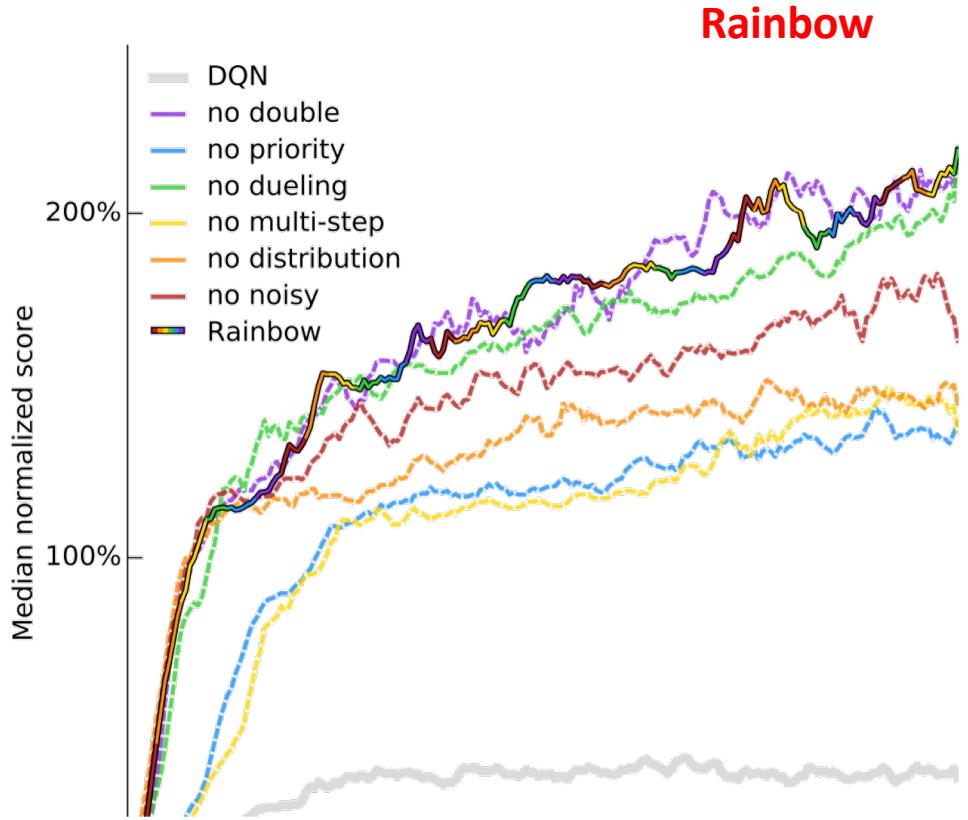
Rainbow DQN: Which tricks are most important?

Prioritized DQN

Distributional DQN

Multi-step learning

Noisy Nets



[Rainbow: Combining Improvements in Deep Reinforcement Learning, Hessel et al, 2017]

Final Logistics

Next lecture:
Modern Landscape II: TRPO, PPO, DDPG, Twin Delayed DDPG, SAC
Reading assignments due **2PM Monday**

Final project literature review due at **11:59pm on Thursday, 4/11**

Complete Programming Assignment for Chapters 12+13 on edx by Sunday 11:59 PM CST