535514: Reinforcement Learning
Lecture 22 — DQN, DDQN, and Distributional RL

Ping-Chun Hsieh

May 9, 2024

Announcements

- Team Project Milestones:
  - 1st Team-Mentor Meetup: 5/7-5/10 (Week 12)
  - 2nd Team-Mentor Meetup: 5/27-5/29 (Week 15)
  - Poster/Oral presentations: 6/11-6/13 (2.5-hour sessions, TBD)
  - Submission of technical report: by 6/17

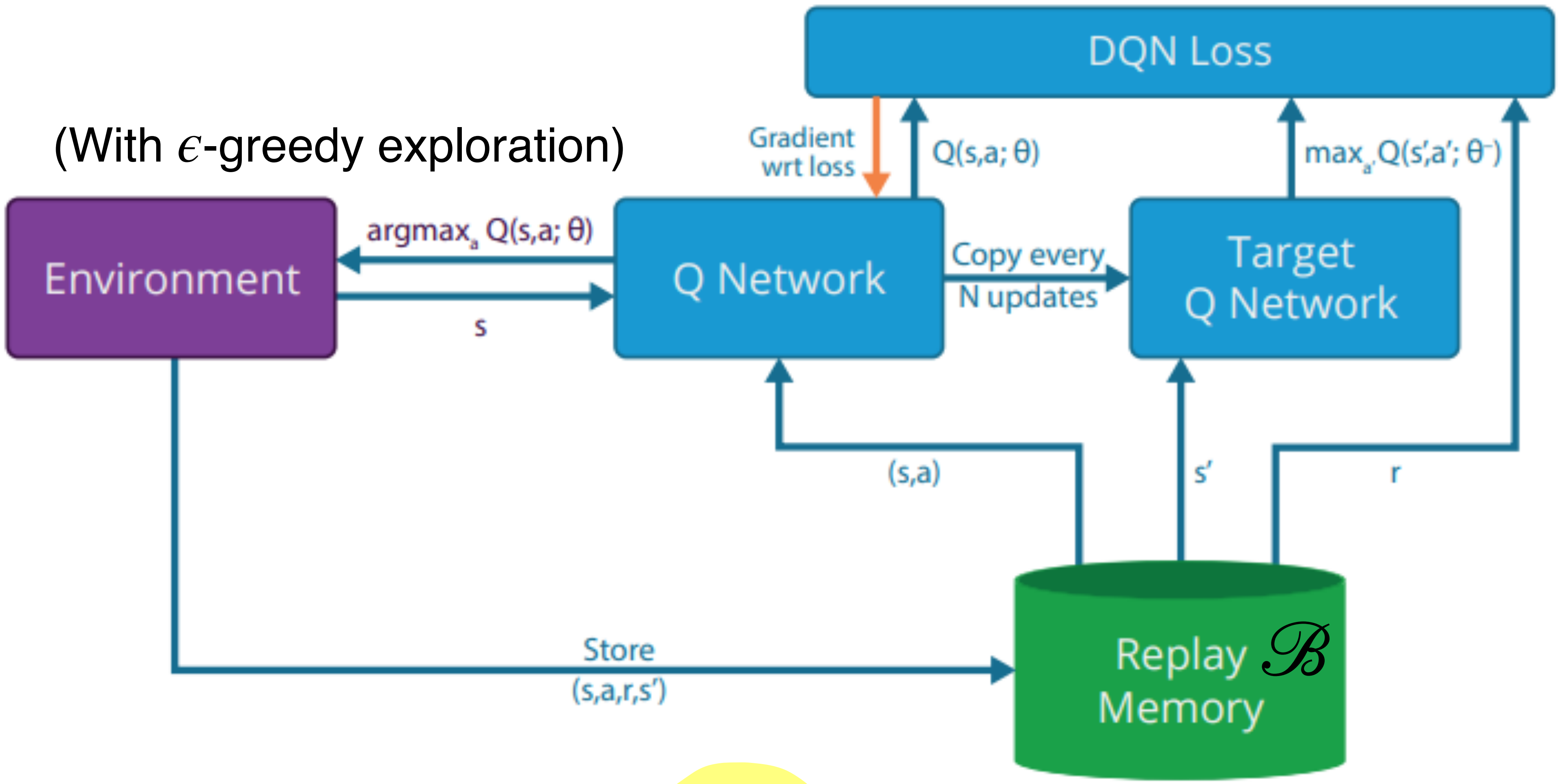
- Theory project:
  - Submit your Hackmd note: by 5/21 (Tuesday), 9pm
  - Hackmd template: https://hackmd.io/@pinghsieh/r1biYBHz0/edit
  - Peer reviews: 5/22-5/28

On-Policy vs Off-Policy Methods

<table>
<tr><th></th><th>Policy Optimization</th><th>Value-Based</th><th>Model-Based</th><th>Imitation-Based</th></tr>
<tr><td>On-Policy</td><td>Exact PG
REINFORCE (w/i baseline)
A2C
On-policy DAC
TRPO
Natural PG (NPG)
PPO-KL & PPO-Clip
RLHF by PPO-KL</td><td>Epsilon-Greedy MC
Sarsa
Expected Sarsa</td><td rowspan="2">Model-Predictive Control (MPC)
PETS</td><td rowspan="2">IRL
GAIL
IQ-Learn</td></tr>
<tr><td>Off-Policy</td><td>Off-policy DPG & DDPG
Twin Delayed DDPG (TD3)</td><td>Q-learning
Double Q-learning
DQN & DDQN
C51 / QR-DQN / IQN
Rainbow
Soft Actor-Critic (SAC)</td></tr>
</table>

Review: Deep Q-Network

(With ϵ-greedy exploration)



$$F(\mathbf{w}) := \frac{1}{2}\mathbb{E}_{(s,a,r,s')\sim\rho}\left[\left(r + \gamma \max_{a'\in A} Q(s',a';\bar{\mathbf{w}}) - Q(s,a;\mathbf{w})\right)^2\right]$$

target network

$$\approx \frac{1}{2}\sum_{(s,a,r,s')\in D \sim \mathscr{B}}\left[\left(r + \gamma \max_{a'\in A} Q(s',a';\bar{\mathbf{w}}) - Q(s,a;\mathbf{w})\right)^2\right]$$

(experience) relay buffer

3. Learn a generator of actions

Draw $a^{(1)}, \cdots, a^{(N)}$ actions

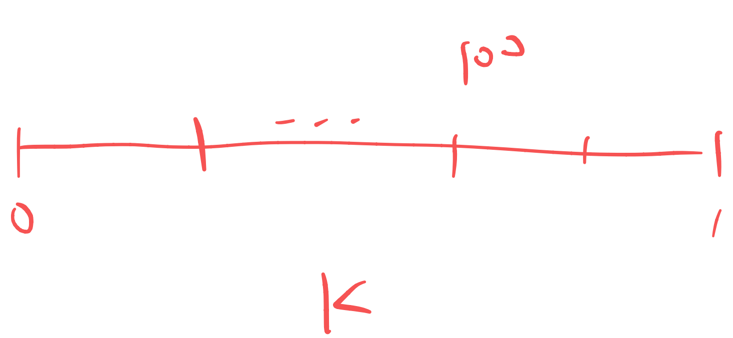
Take $\arg\max_i Q(s, a^{(i)}; w)$

Can DQN be Applied Under Continuous Actions?

The difficulty lies in the "$\arg\max_{a \in \mathcal{A}} Q(s, \underline{a}; \mathbf{w})$" operation

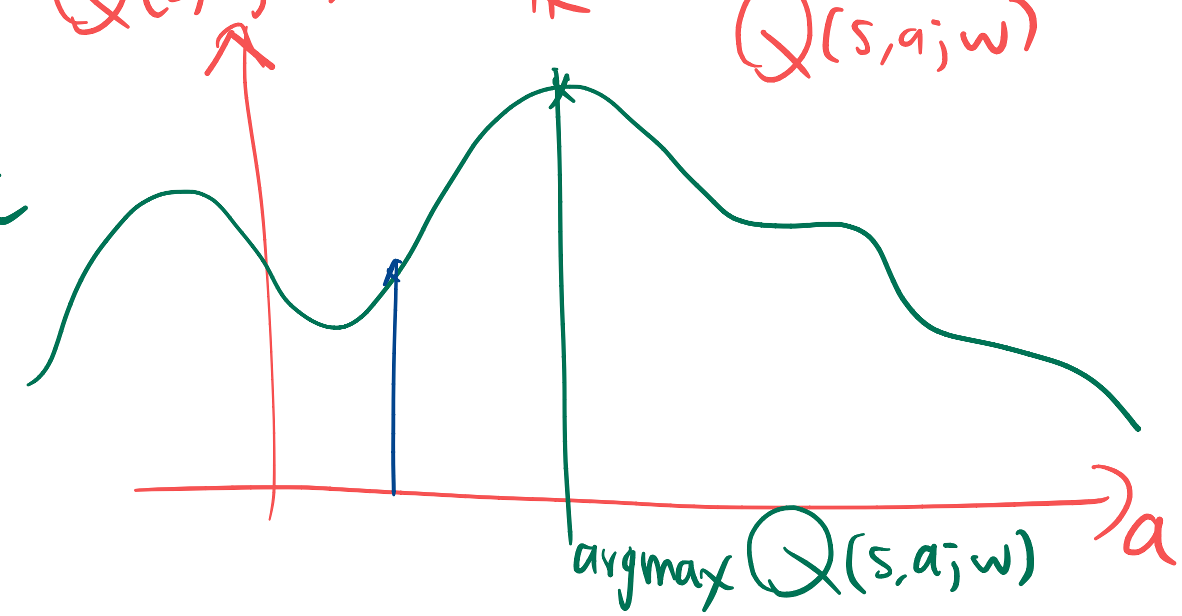
1. Discretization

$A = [0,1]^{100}$



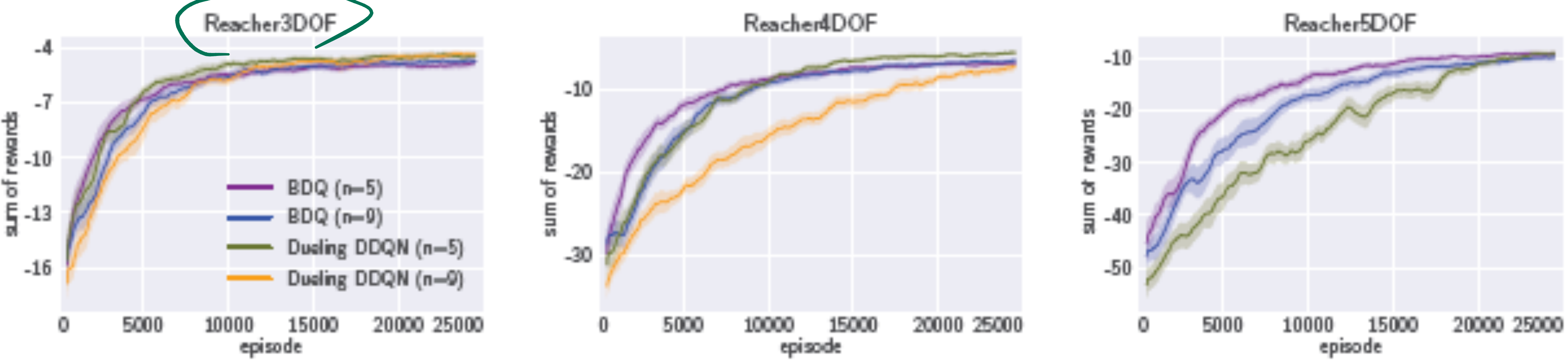
2. Gradient ascent

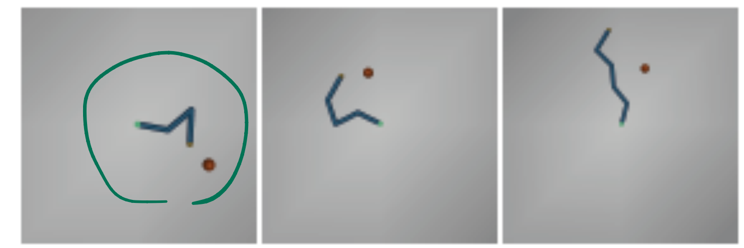
$Q(s, a; w)$ $\quad a \in \mathbb{R}^1$ $\quad Q(s, a; w)$



Existing methods that adapts DQN to continuous actions

1. Naive Action Discretization [Tavakoli et al., AAAI 20~~20~~18]



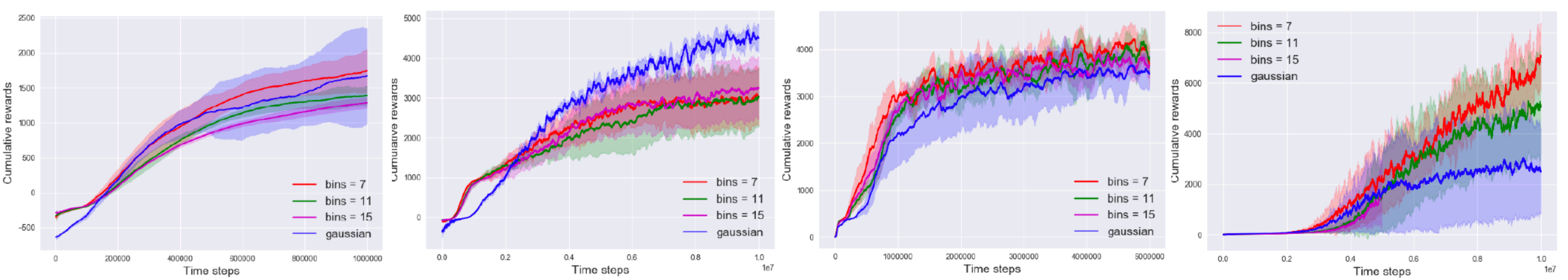


Reacher (3-5 DOF)

Issue: Naive discretization suffers from exponential growth of cardinality

2. Discretization + Factorization [Tang and Agrawal, AAAI 2020]

$$\pi(a|s) := \Pi_{i=1}^{d} \pi_{\theta_i}(a_i|s) \quad \text{where } a = [a_0, a_1, \cdots, a_{d-1}]^\top$$

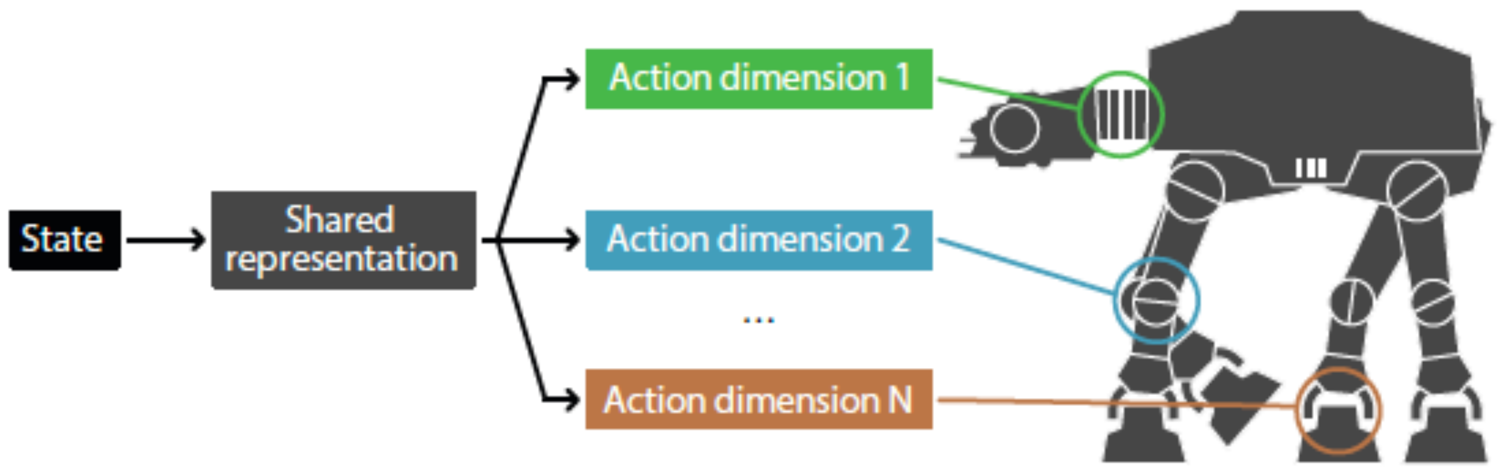


(a) **HalfCheetah + PPO** (b) **Ant + PPO** (c) **Walker + PPO** (d) **Humanoid (R) + PPO**

Tavakoli et al., Action Branching Architectures for Deep Reinforcement Learning, AAAI 2018

Tang and Agrawal et al., Discretizing Continuous Action Space for On-Policy Optimization, AAAI 2020

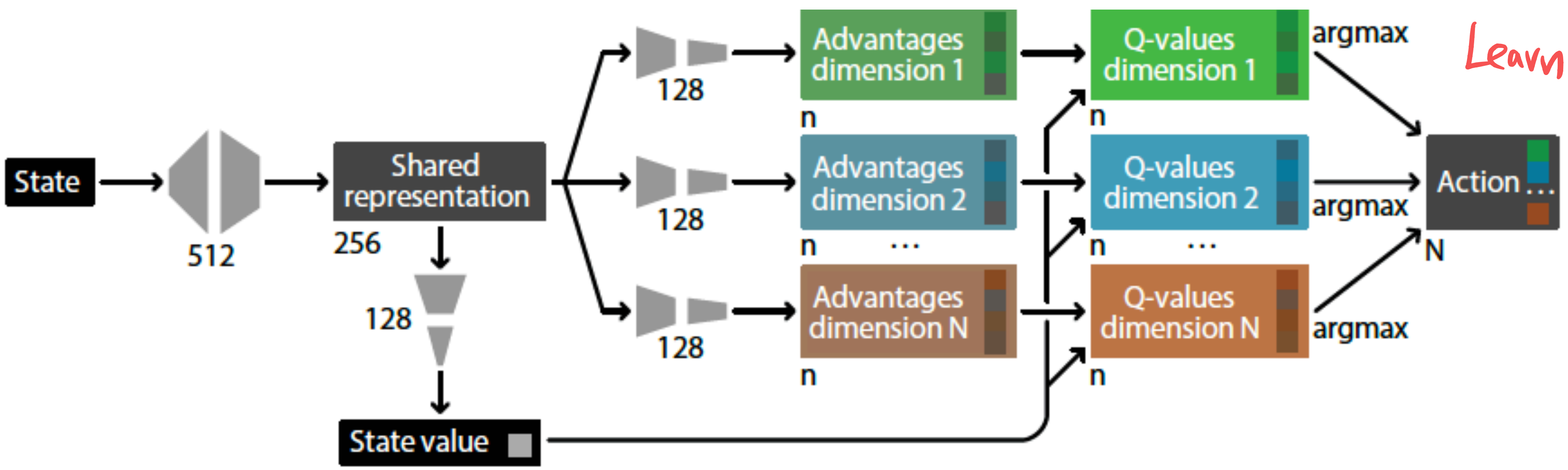
3. Discretization + Branching [Tavakoli et al., AAAI 2018]



Original:

$$\arg\max_{a \in A} Q(s,a;w)$$

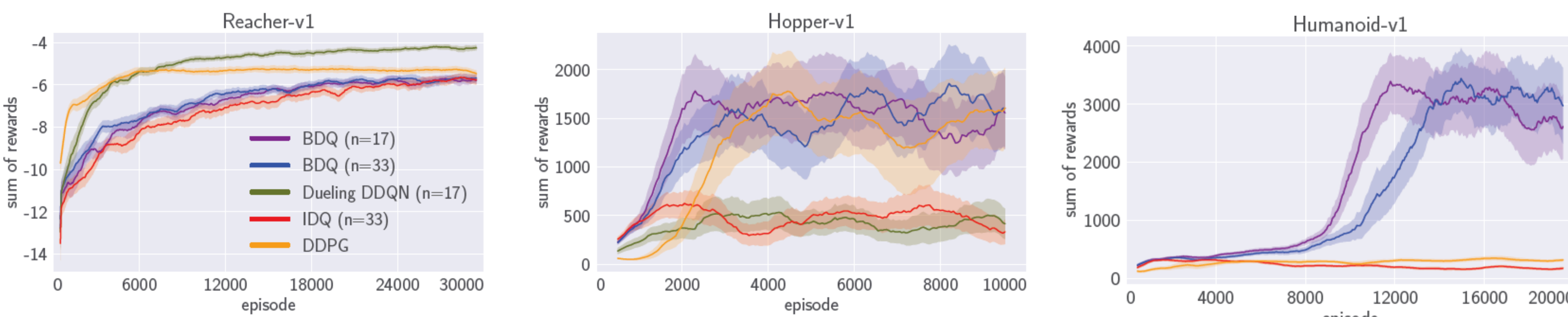
Now: Suppose $a \in \mathbb{R}^3$



Learn $Q^{(1)}(s,a^{(1)};w^{(1)})$

$Q^{(2)}(s,a^{(2)};w^{(2)})$

$Q^{(3)}(s,a^{(3)};w^{(3)})$



Tavakoli et al., Action Branching Architectures for Deep Reinforcement Learning, AAAI 2018

4. Normalized Advantage Functions (NAF): Quadratic Approximation!

Continuous Deep Q-Learning with Model-based Acceleration [Gu et al., ICML 2016]

Shixiang Gu[1 2 3] SG717@CAM.AC.UK
Timothy Lillicrap[4] COUNTZERO@GOOGLE.COM
Ilya Sutskever[3] ILYASU@GOOGLE.COM
Sergey Levine[3] SLEVINE@GOOGLE.COM
[1]University of Cambridge [2]Max Planck Institute for Intelligent Systems [3]Google Brain [4]Google DeepMind

$$Q(s,a;\phi_A,\phi_V) = A(s,a;\phi_A) + V(s;\phi_V)$$

(P is state-dependent, positive definite)

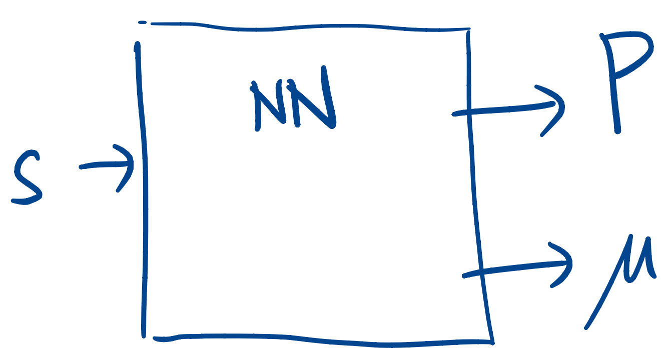
$$A(s,a;\phi_A) := -\frac{1}{2}\overset{1\times d}{(a-\mu(s;\phi_\mu))^\top}\overset{d\times d}{P(s;\phi_P)}\overset{d\times 1}{(a-\mu(s;\phi_\mu))}$$

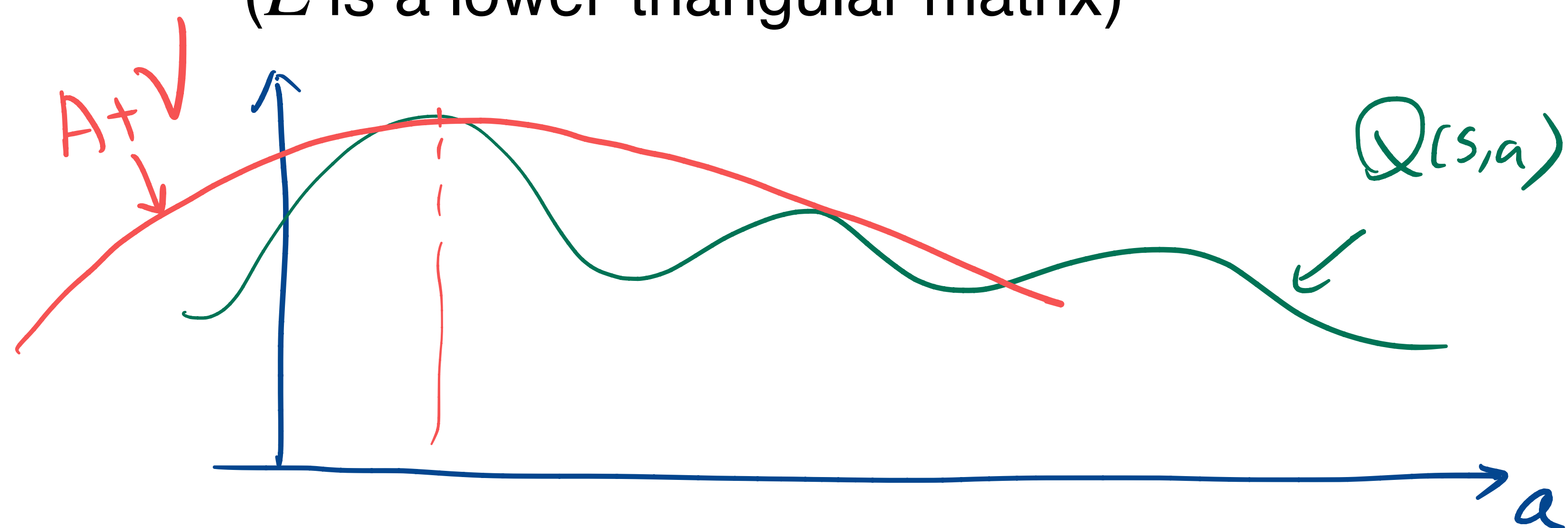
$\Rightarrow$ The maximizer of $A(s,a;\phi_A)$ is simply $\mu(s;\phi_\mu)$

$$P(s;\phi_P) := L(s|;\phi_P)L(s|;\phi_P)^\top$$

(This is known as the "Cholesky decomposition")

(L is a lower-triangular matrix)





Gu et al., Continuous Deep Q-Learning with Model-based Acceleration, ICML 2016

5. Amortized Q-Learning (AQL): Sampling!

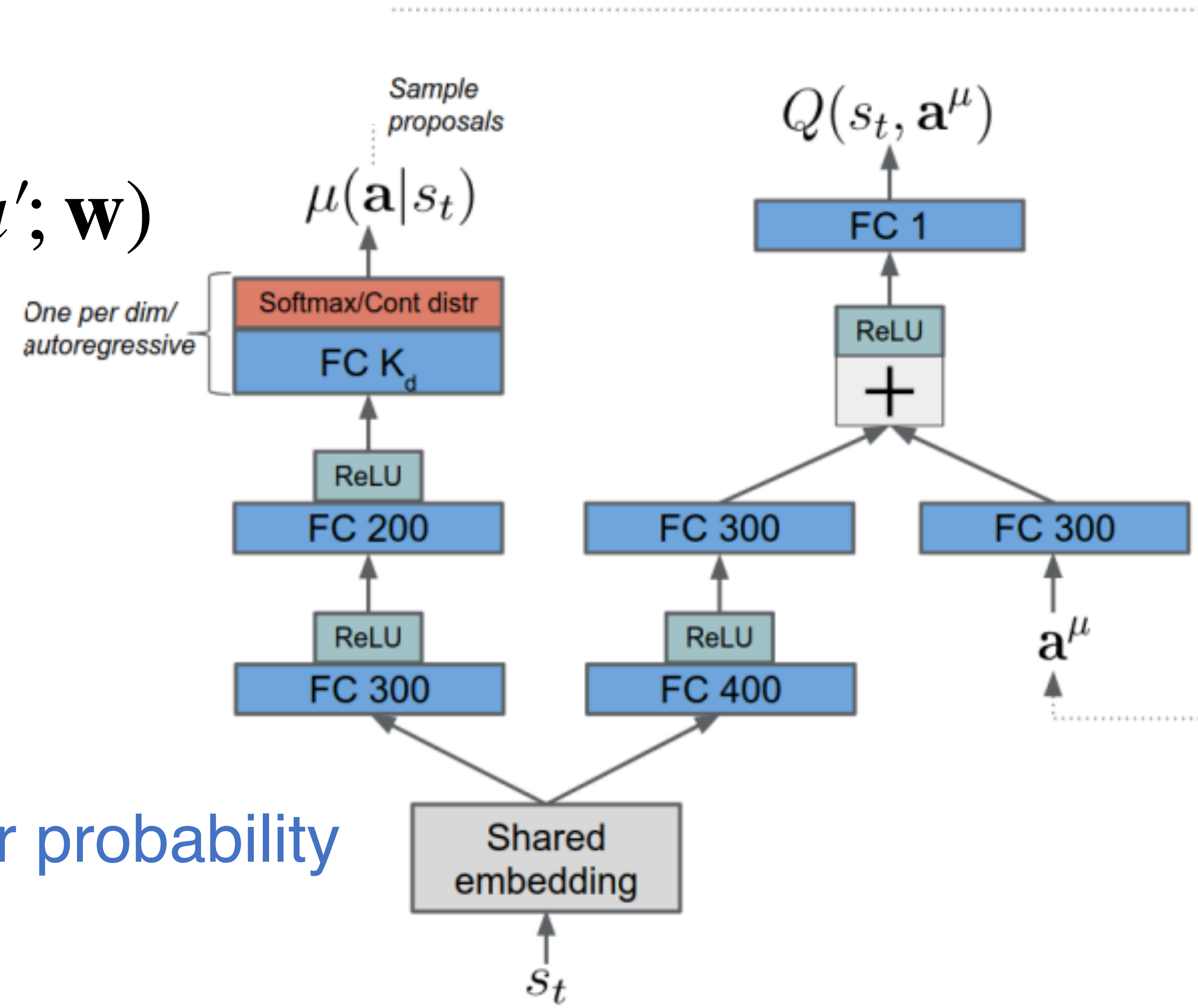
Q-LEARNING IN ENORMOUS ACTION SPACES VIA AMORTIZED APPROXIMATE MAXIMIZATION

[NeurIPS 2018 Workshop]

Tom Van de Wiele,* David Warde-Farley, Andriy Mnih & Volodymyr Mnih
DeepMind
London, United Kingdom
tvdwiele@gmail.com, {dwf,amnih,vmnih}@google.com

$$\max_{a \in \mathscr{A}} Q(s, a; \mathbf{w}) \approx \max_{a' \in D \sim \mu(a)} Q(s, a'; \mathbf{w})$$

"proposal distribution"
(Learned by maximum likelihood)

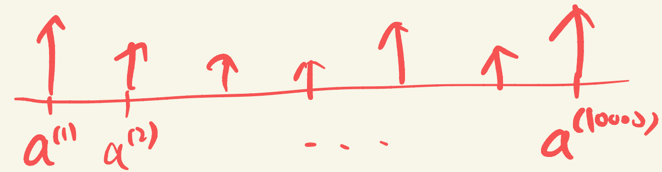


Intuition: Larger $|D|$ induces a higher probability of seeing max-Q actions

Van de Wiele et al., Q-Learning in Enormous Action Spaces via Amortized Approximate Maximization, NeurIPS Workshop 2018

$$A := \{a^{(1)}, a^{(2)}, \cdots, a^{(10000)}\}$$

Suppose $a^{(1)} = \underset{a \in A}{\arg\max}\, Q(s,a)$



Sampling distribution μ

Suppose $\mu(a^{(1)}) = 0.01$ and we draw K actions independently from μ.

$$P(\text{sample } a^{(1)} \text{ for at least once}) = 1 - (0.99)^{K = 100}$$

6. DDPG: Reinterpret DDPG as an Adaptation of DQN for Continuous Actions!

(Quick Review)

Off-Policy Deterministic PG: $\nabla_\theta J_\beta^{\pi_\theta} \approx \mathbb{E}_{s\sim d_\mu^\beta}\left[\nabla_\theta \pi_\theta(s) \nabla_a Q^{\pi_\theta}(s,a)|_{a=\pi_\theta(s)} \right]$

- Critic: estimate $Q_w \approx Q^{\pi_\theta}$ by bootstrapping
- Actor: updates policy parameters θ by deterministic policy gradient

Step 1: Initialize θ_0, w_0 and step sizes α_θ, α_w

Step 2: Sample a trajectory $\tau = (s_0, a_0, r_1, \cdots) \sim P_\mu^\beta$

For each step of the current trajectory $t = 0,1,2,\cdots$

$$\Delta w_k \leftarrow \Delta w_k + \alpha_w \left(r_t + \gamma Q_{w_k}(s_{t+1}, \pi_\theta(s_{t+1})) - Q_{w_k}(s_t, a_t) \right) \nabla_w Q_w(s_t, a_t)|_{w=w_k}$$

$$\Delta \theta_k \leftarrow \Delta \theta_k + \alpha_\theta \gamma^t \underline{\left(\nabla_\theta \pi_\theta(s_t) \nabla_a Q_{w_k}(s_t, a)|_{a=\pi_\theta(s_t)} \right)}$$

$$= \nabla_\theta Q_{w_k}(s_t, \pi_\theta(s_t))|_{\theta=\theta_k}$$

Alternative Interpretation of DDPG: An Adaptation of DQN for Continuous Actions (Cont.)

- DDPG can be reinterpreted as DQN for continuous actions

1. Deterministic policy: $\pi_\theta(s) \approx \arg\max_a Q_w(s, a)$

2. How to find θ: solve $\theta \leftarrow \arg\max_\theta Q_w(s, \pi_\theta(s))$ by SGD

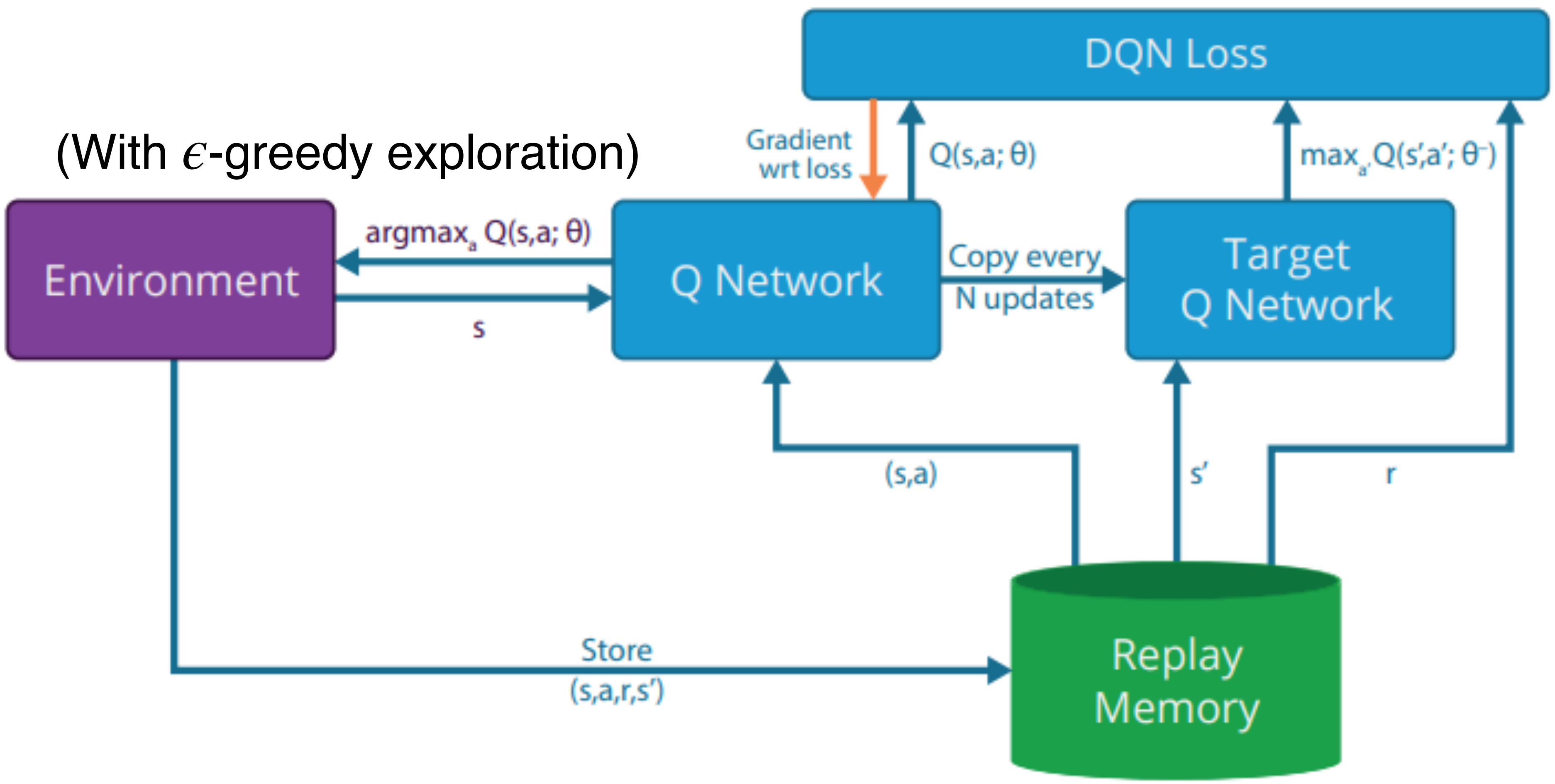
(Actor)

$$\Delta\theta_k \leftarrow \Delta\theta_k + \alpha_\theta \gamma^t \underbrace{\left(\nabla_\theta \pi_\theta(s_t) \nabla_a Q_{w_k}(s_t, a)|_{a=\pi_\theta(s_t)} \right)}_{= \nabla_\theta Q_{w_k}(s_t, \pi_\theta(s_t))|_{\theta=\theta_k}}$$

3. DQN and DDPG have a similar TD update scheme

$$\Delta w_k \leftarrow \Delta w_k + \alpha_w \left(r_t + \gamma Q_{w_k}(s_{t+1}, \pi_\theta(s_{t+1})) - Q_{w_k}(s_t, a_t) \right) \nabla_w Q_w(s_t, a_t)|_{w=w_k}$$

Next Topic: What to Improve in Vanilla DQN?



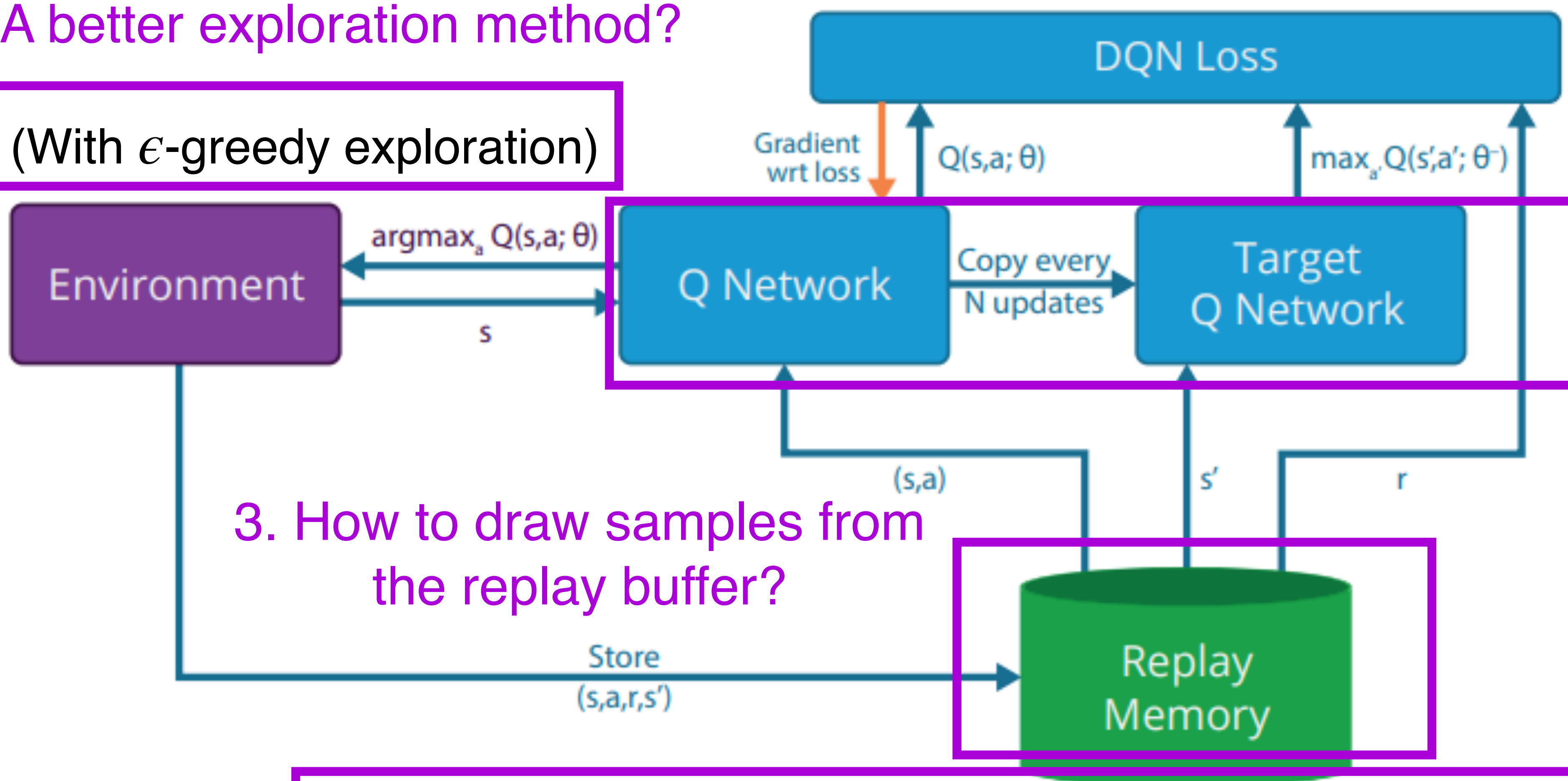
$$L(\mathbf{w}) := \sum_{(s,a,r,s') \in D} \frac{1}{2}\left[\left(r + \gamma \max_{a'} Q(s', a'; \bar{\mathbf{w}}) - Q(s, a; \mathbf{w})\right)^2\right]$$

Next Topic: What to Improve in Vanilla DQN?

5. A better way to represent Q function?

4. A better exploration method?

(With ϵ-greedy exploration)



3. How to draw samples from the replay buffer?

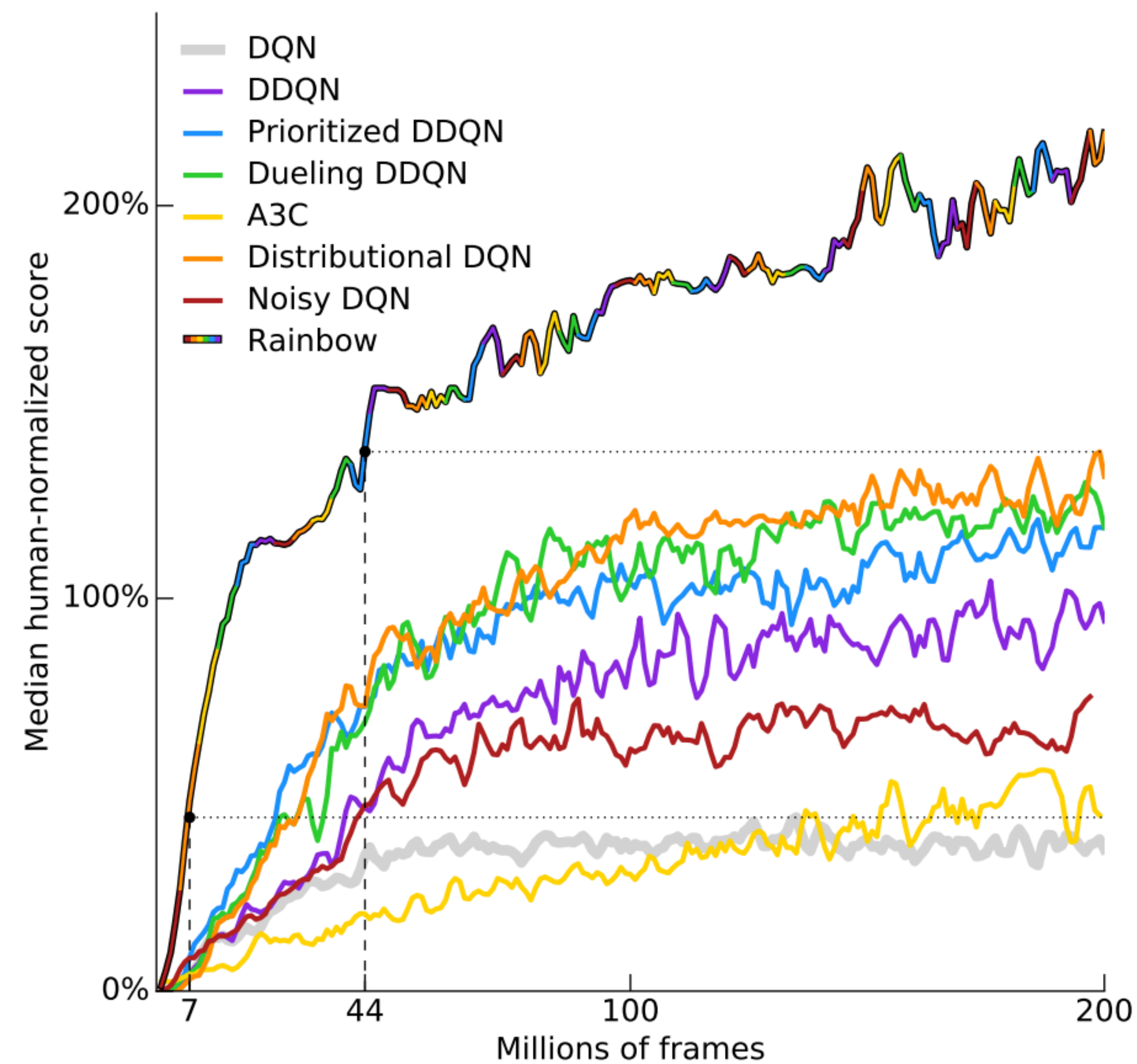
$$L(\mathbf{w}) := \sum_{(s,a,r,s')\in D} \frac{1}{2}\left[\left(r + \gamma \boxed{\max_{a'} Q(s', a'; \bar{\mathbf{w}})} - Q(s, a; \mathbf{w})\right)^2\right]$$

1. Overestimation Bias?

2. A better loss function?

Next Topic: Rainbow (= DQN with 6 Useful Tricks)

1. Double DQN (DDQN)
2. Distributional Q-learning
3. Prioritized experience replay (PER)
4. Dueling networks
5. Multi-step return in TD target
6. Noisy networks for exploration



Hessel et al., Rainbow: Combining Improvements in Deep Reinforcement Learning, AAAI 2018

Double DQN (DDQN)

Hado van Hasselt, Arthur Guez, and David Silver,
“Deep Reinforcement Learning with Double Q-learning,” AAAI 2016

Recall: Double Q-Learning

Step 1: Initialize $Q^A(s,a), Q^B(s,a)$ for all (s,a), and initial state s_0

Step 2: For each step $t = 0,1,2,\cdots$

Select a_t using ε-greedy w.r.t $Q^A(s_t, a) + Q^B(s_t, a)$

Observe (r_{t+1}, s_{t+1})

Choose one of the following updates uniformly at random

$$Q^A(s_t, a_t) \leftarrow Q^A(s_t, a_t) + \alpha\left(r_{t+1} + \gamma Q^B(s_{t+1}, \arg\max_a Q^A(s_{t+1}, a)) - Q^A(s_t, a_t)\right)$$

$$Q^B(s_t, a_t) \leftarrow Q^B(s_t, a_t) + \alpha\left(r_{t+1} + \gamma Q^A(s_{t+1}, \arg\max_a Q^B(s_{t+1}, a)) - Q^B(s_t, a_t)\right)$$

- Key Idea: Decouple "Q value" and "greedy action selection"
- Question: How to apply this to DQN?

Double DQN

- Loss function of DQN:

$$F(\mathbf{w}) := \frac{1}{2}\mathbb{E}_{(s,a,r,s')\sim\rho}\left[\left(r + \gamma \max_{a'\in A} Q(s', a'; \bar{\mathbf{w}}) - Q(s, a; \mathbf{w})\right)^2\right]$$

$$\approx \frac{1}{2}\sum_{(s,a,r,s')\in D}\left[\left(r + \gamma \max_{a'\in A} Q(s', a'; \mathbf{w}) - Q(s, a; \mathbf{w})\right)^2\right]$$

- Loss function of Double DQN:

$$F(\mathbf{w}) := \frac{1}{2}\mathbb{E}_{(s,a,r,s')\sim\rho}\left[\left(r + \gamma Q\left(s', \arg\max_{a'\in A} Q(s, a; \mathbf{w}); \bar{\mathbf{w}}\right) - Q(s, a; \mathbf{w})\right)^2\right]$$

$$\approx \frac{1}{2}\sum_{(s,a,r,s')\sim D}\left[\left(r + \gamma Q\left(s', \arg\max_{a'\in A} Q(s, a; \mathbf{w}); \bar{\mathbf{w}}\right) - Q(s, a; \mathbf{w})\right)^2\right]$$

"*We therefore propose to evaluate the greedy policy according to the online network, but using the target network to estimate its value.*" — [van Hasselt et al., AAAI 2016]

Distributional Q-Learning

(Learn value distribution $Z(s, a)$ & use $E[Z(s, a)]$ as $Q(s, a)$ in Q-Learning)

Why Shall We Consider “Value Distributions”?

- Risky vs safe choices
  - E.g., Same expected return but different variance
- Good empirical performance (despite that the underlying root cause is not fully known)
  - C51 [Belleware et al., ICML 2017]
  - QR-DQN [Dabney et al., AAAI 2018]
  - IQN [Dabney et al., ICML 2018]
- New approaches for exploration
  - Information-directed exploration [Nikolov et al., ICLR 2019]
  - Distributional RL for efficient exploration [Mavrin et al., ICML 2019]
- Learn better critics
  - Truncated Quantile Critics (TQC) [Kuznetsov et al., ICML 2020]

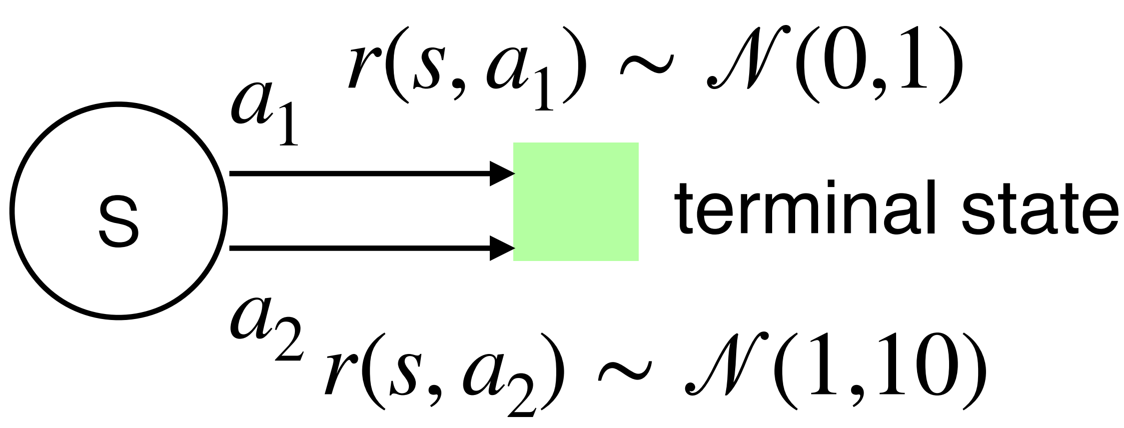
Question: How to learn the complete value distribution (instead of merely the expectation)?

Sample Action-Value $Z^\pi(s, a)$

- Sample action-value $Z^\pi(s, a)$: sample return if we start from state s and take action a, and then follow policy π

$$Q^\pi(s, a) = \mathbb{E}[Z^\pi(s, a)] = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\Big]$$

- $Z^\pi(s, a)$ is essentially a random variable (and hence follows some distribution)
- Example: 1-state MDP with 2 actions and $\pi(s) = a_1$



- $Q^\pi(s, a_1)$ = ? $Z^\pi(s, a_1)$?
- $Q^\pi(s, a_2)$ = ? $Z^\pi(s, a_2)$?

Finding Z^π via Distributional Bellman Equation

- Mild assumption: $Z^\pi(s,a)$ has bounded moments
- Distributional Bellman equation for $Z^\pi(s,a)$: Given s,a, we have

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

($\overset{D}{=}$: equal in distribution)

- Question: How to interpret this equation?
- Question: Are $r(s,a)$ and $Z^\pi(s',a')$ independent?
- Question: Is this consistent with Bellman expectation equation?

Distributional Bellman Operator B^{π}

- $\mathcal{Z}$: the space of all value distributions with bounded moments
- Transition operator $P^{\pi} : \mathcal{Z} \to \mathcal{Z}$

$$P^{\pi} Z(s,a) :\overset{D}{=} Z(s',a')$$

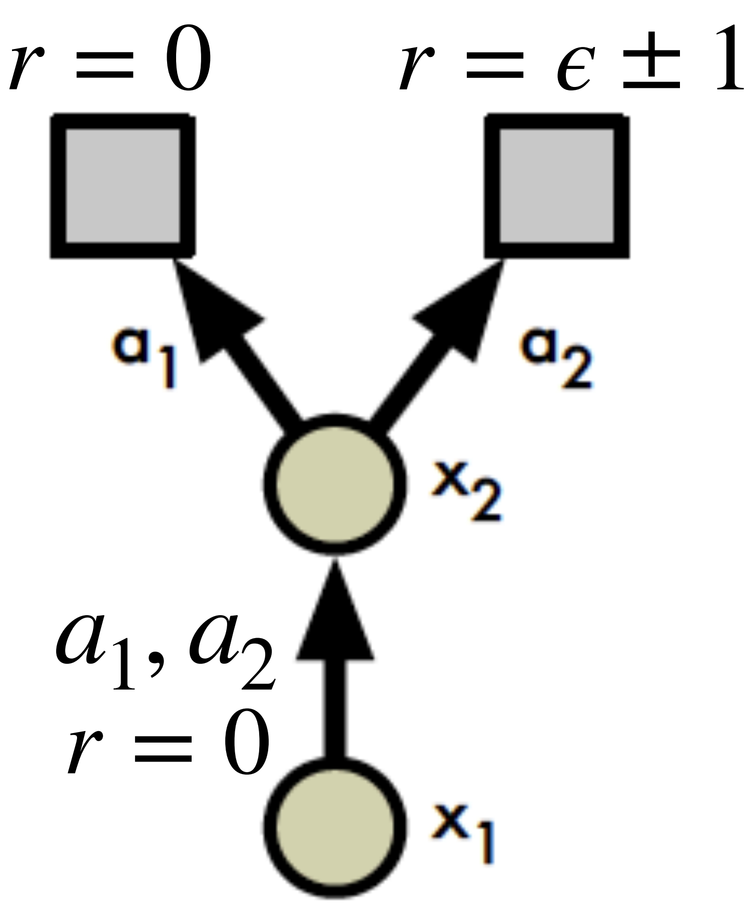
$$s' \sim P(\cdot \mid s,a), \;\; a' \sim \pi(\cdot \mid s')$$

- Distributional Bellman operator $B^{\pi} : \mathcal{Z} \to \mathcal{Z}$

$$B^{\pi} Z(s,a) :\overset{D}{=} r(s,a) + \gamma P^{\pi} Z(s,a)$$

An Example of Applying B^π

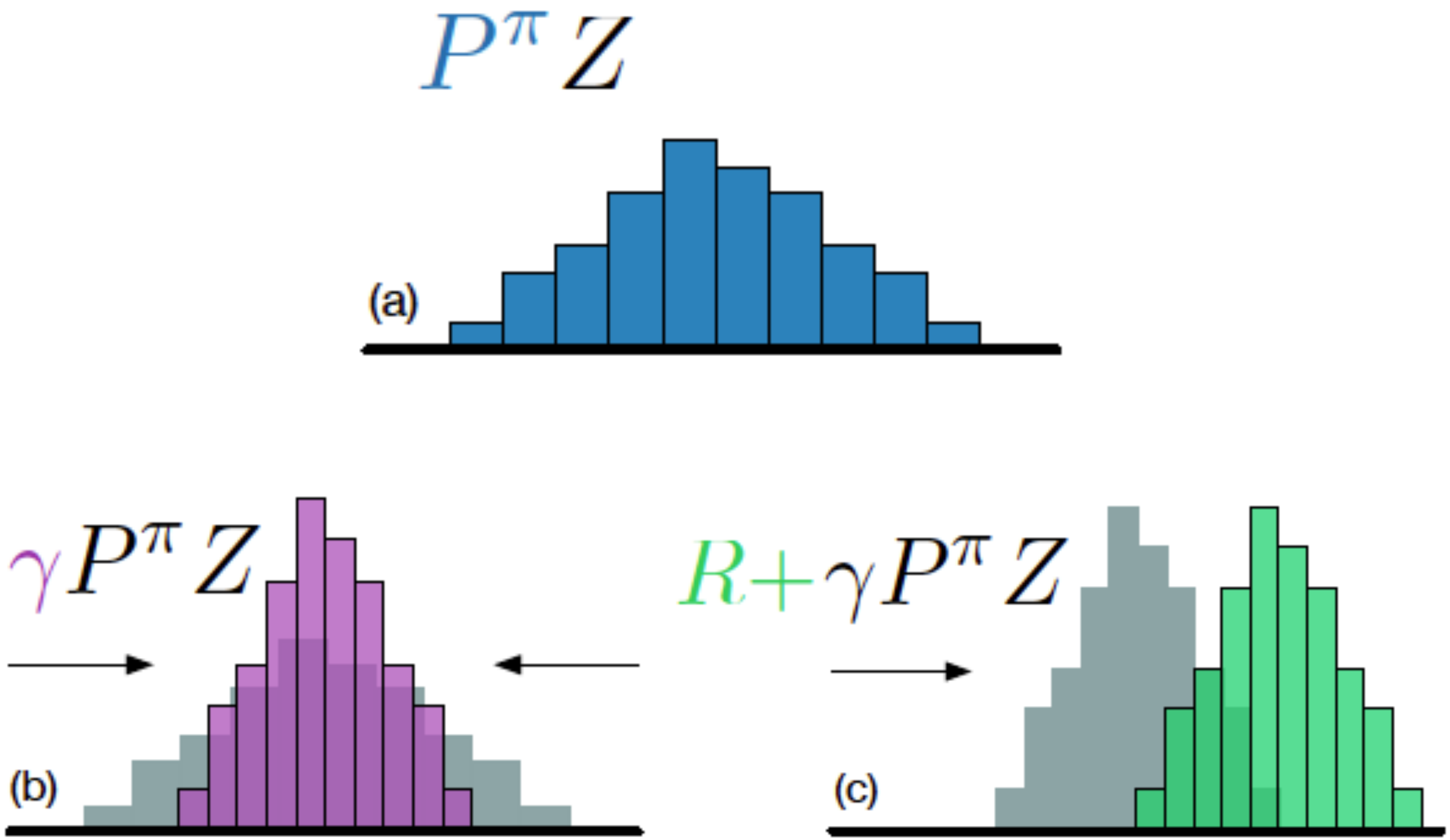
- Example: 2 states x_1, x_2 and 2 actions a_1, a_2
- $\pi(a_1 \mid x_2) = 0.3$, $\pi(a_2 \mid x_2) = 0.7$, and $\gamma = 0.9$



$$B^\pi Z(s,a) :\overset{D}{=} r(s,a) + \gamma P^\pi Z(s,a)$$

- Suppose $Z(x_1, a_1) = 0$, $Z(x_2, a_1) = 0$ with probability 1 and $Z(x_2, a_2) \sim \mathcal{N}(0,1)$
- Question: $B^\pi Z(x_2, a_2)$ =? $B^\pi Z(x_1, a_1)$ =?

Visualization of Distributional Bellman Operator



Distributional "Optimality" Operator

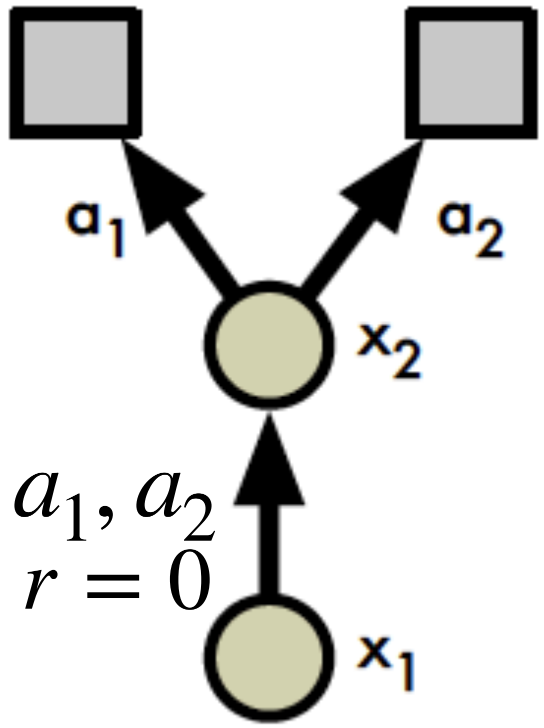
Recall— Distributional Bellman operator $B^\pi : \mathscr{Z} \to \mathscr{Z}$

$$B^\pi Z(s,a) :\overset{D}{=} r(s,a) + \gamma P^\pi Z(s,a)$$

- Distributional optimality operator B^*: The B^π resulting from a greedy policy π (what does "greedy" mean here?)

An Example of B^*

$r = 0$ $\quad$ $r = \epsilon \pm 1 \; (\epsilon > 0)$



a_1, a_2
$r = 0$

$B^*Z(s, a) :\overset{D}{=} r(s, a) + \gamma P^{\pi_{greedy}} Z(s, a)$

Suppose we have the following:

- $\pi(a_1 \mid x_2) = 0.3$, $\pi(a_2 \mid x_2) = 0.7$, and $\gamma = 1$
- $Z(x_1, a_1) = 0$, $Z(x_2, a_1) = 0$ with probability 1
- $Z(x_2, a_2) \sim \mathcal{N}(0,1)$

Question: What's the PDF of $B^*Z(x_1, a_1)$ =?