Probabilistic Bellman Consistency in Reinforcement Learning

Luca Biggio, Carl Rasmussen
Engineering Department, University of Cambridge

UNIVERSITY OF CAMBRIDGE

Motivation

- Bellman's equations are the basis of most Reinforcement Learning algorithms:

$$Q^\pi(x,a) = \mathbb{E}R(x,a) + \gamma \mathop{\mathbb{E}}_{P,\pi} Q^\pi(x',a') \qquad (1)$$

- These equations are written in terms of the expectation of the discounted sum of rewards (*return*).

$$Q^\pi(x,a) := \mathbb{E}Z^\pi(x,a) = \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t R(x_t,a_t)\right]$$

- Could a richer representation of the return $Z^\pi(x,a)$ (not limited to its expected value) benefit the learning process and the exploration vs. exploitation trade-off?

A Distributional Approach to Reinforcement Learning

- The behaviours of popular RL algorithms such as SARSA or Q-Learning [1] can be understood by introducing the so-called Bellman operator $\mathcal{T}^\pi$ and optimality operator $\mathcal{T}$:

$$\mathcal{T}^\pi Q(x,a) := \mathbb{E}R(x,a) + \gamma \mathop{\mathbb{E}}_{P,\pi} Q(x',a')$$
$$\mathcal{T}Q(x,a) := \mathbb{E}R(x,a) + \gamma \mathbb{E}_P \max_{a'\in\mathcal{A}} Q(x',a')$$

- $\mathcal{T}^\pi$ and $\mathcal{T}$ are *contraction mappings*, i.e. their repeated application to some initial Q_0 converges exponentially to Q^π (see Eq. 1) or Q^* (the optimal value function), respectively.

- The basic idea of Distributional RL (DRL) [2] is to take into account the entire return distribution instead of just its expected value. To this purpose, the *distributional Bellman operator* $\mathcal{T}^\pi_D$ and the *distributional optimality operator* $\mathcal{T}_D$ are defined as follows:

$$\mathcal{T}^\pi Z(x,a) :\overset{D}{=} R(x,a) + \gamma Z(X',A')$$
$$\mathcal{T}Z(x,a) :\overset{D}{=} R(x,a) + \gamma Z\left(X', \arg\max_{a'\in\mathcal{A}} \mathbb{E}Z(X',a')\right)$$

- Adopting a distributional perspective on RL introduces a number of complications compared to the standard case. The following diagram lists three of them.

- **Convergence**: We are not in general guaranteed that by repeatedly applying the distributional Bellman operators a fixed point can be reached as in the standard case.
- **Control Setting**: The convergence properties of the distributional operators of the policy evaluation and control settings may be different.
- **Algorithms**: What is a proper way of describing a distribution over returns from an algorithmic point of view?

C51

- Distributions defined on a **fixed support** $\{z_1,\dots,z_N\}$ [2].
- Discrete distribution $\{p_i(x,a)\}_{1\le i\le N}$:

$$Z_\theta(x,a) = z_i \quad \text{w.p. } p_i(x,a) := \frac{e^{\theta_i(x,a)}}{\sum_j e^{\theta_j(x,a)}}$$

where $\theta : \mathcal{X}\times\mathcal{A}\to\mathbb{R}^N$ is a parametric model (neural network).
- $\mathcal{T}Z_\theta$ and our parametrisation Z_θ have disjoint supports. Therefore, the sample Bellman update $\hat{\mathcal{T}}Z_\theta$ is projected onto the support of Z_θ. The projected update is indicated by $\Phi\hat{\mathcal{T}}Z_\theta(x,a)$.
- The cross entropy term of the KL divergence

$$D_{\mathrm{KL}}\left(\Phi\hat{\mathcal{T}}Z_{\tilde\theta}(x,a) \| Z_\theta(x,a)\right)$$

can be minimized by stochastic gradient descent.

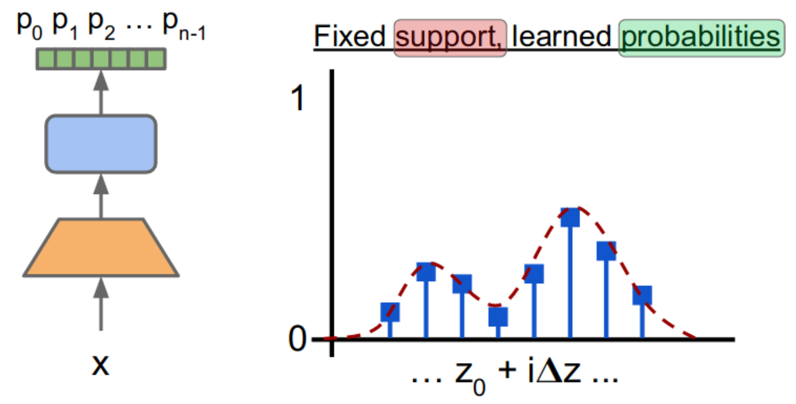

Figure 1: C51 algorithm: the network outputs the probability values associated with each (fixed) support point.

Quantile Regression DQN (QR-DQN)

- The goal is to estimate the *quantiles* of the target distribution (the *quantile distribution*) [3]. **Variable support** locations but fixed cumulative probabilities $\tau_1,\dots,\tau_N$, so that $\tau_i = \frac{i}{N}$ for $i = 1,\dots,N$.
- The quantile distribution associated with a state-action pair (x,a) is defined as:

$$Z_\theta(x,a) := \frac{1}{N}\sum_{i=1}^{N}\delta_{\theta_i(x,a)} \qquad (2)$$

where $\theta : \mathcal{X}\times\mathcal{A}\to\mathbb{R}^N$ is a parametric model (a neural network) whose outputs are the support points $\{\theta_i(x,a)\}$ of the distribution.
- The quantile regression loss computed at the *quantile midpoints* $\hat\tau_i = \frac{\tau_{i-1}+\tau_i}{2}$

$$\mathcal{L}^\tau_{\mathrm{QR}}(\theta) := \sum_{i=1}^{N}\mathbb{E}_j\left[\rho_{\hat\tau_i}(\mathcal{T}\theta_j - \theta_i(x,a))\right], \text{ where}$$
$$\rho_{\hat\tau}(u) = u\left(\hat\tau - \delta_{\{u<0\}}\right), \forall u\in\mathbb{R} \qquad (3)$$

provides unbiased sample gradients and its minimisation yields the set of support points $\{\theta_1,\dots,\theta_N\}$ minimizing the 1-Wasserstein distance $W_1(\mathcal{T}Z_{\tilde\theta}, Z_\theta)$.

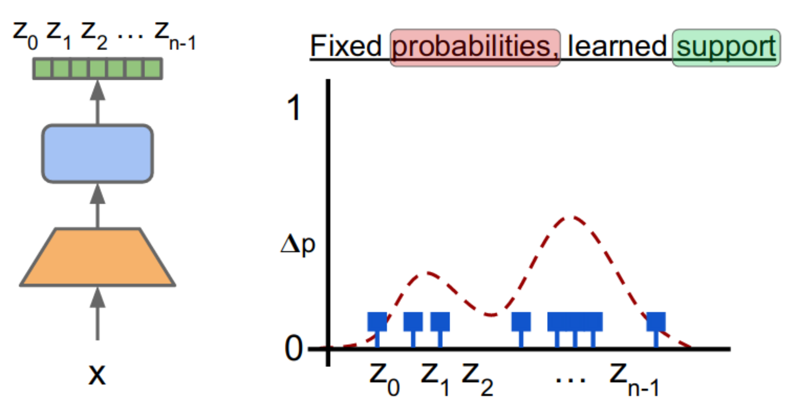

Figure 2: QR-DQN algorithm: the network outputs support points associated with (fixed) quantile values.

Preliminary Results

- OpenAI gym MountainCar-v0: reward is -1 for each time step, until the goal position (top of the hill) is reached. Three actions available: left, right, no action.

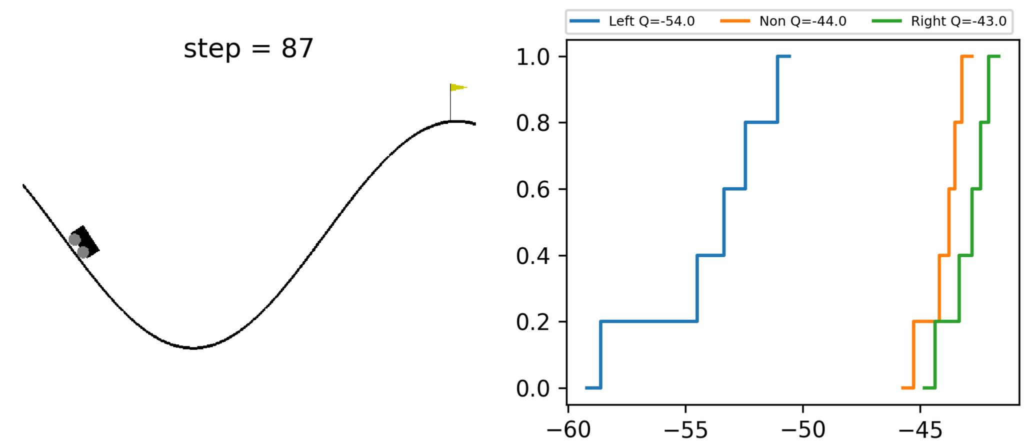

Figure 3: Support points distribution for the MountainCar-v0 environment (5 quantiles)

- OpenAI gym CartPole-v0: the pendulum starts upright, and the goal is to prevent it from falling over. A reward of +1 is provided for each time step. Two actions available: left, right.

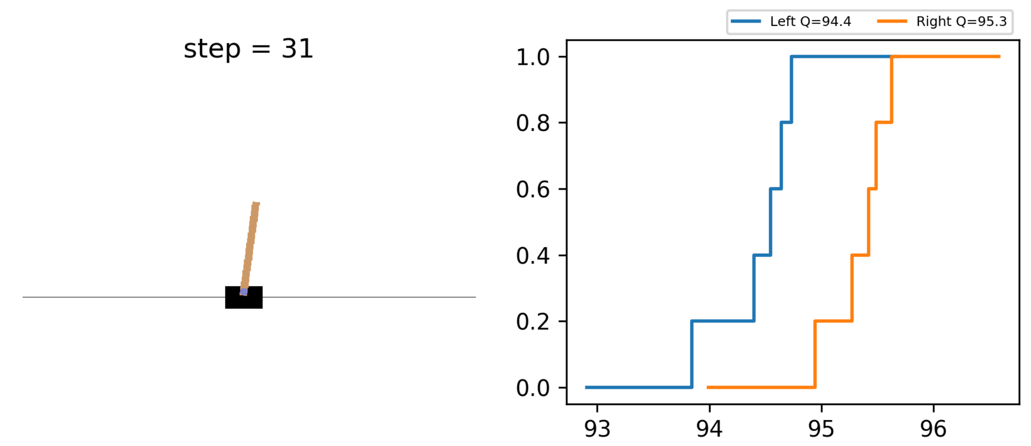

Figure 4: Support points distribution for the CartPole-v0 environment (5 quantiles)

Recent Trends

- Two types of uncertainty in RL algorithms: *aleatoric* and *epistemic* uncertainty.
- DRL has very recently [4], [5], [6] been used for exploiting these two types of uncertainty to design better exploration strategies.

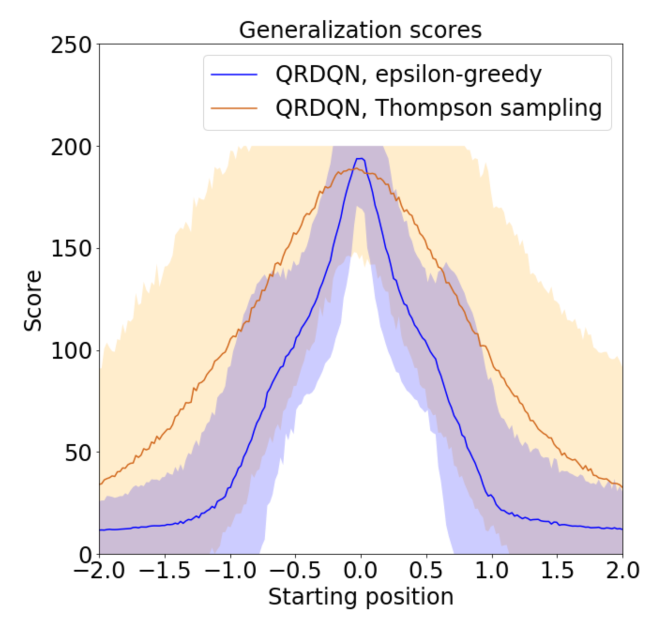

Figure 5: Score achieved as a function of the starting position in the CartPole-v0 environment with a DRL-based (orange) and ϵ-greedy (blue) exploration methods. Taken from [4].

Future Research Directions

- Investigate if DRL approaches can be successfully applied to a *model-based* RL framework. Ideas to explore are for instance:
  - Model based exploration: learning the model dynamics to guide the agent towards unvisited states.

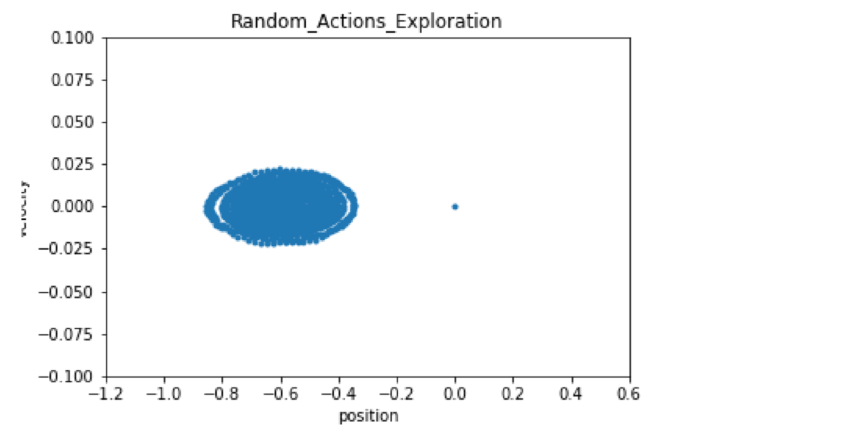

Figure 6: 50 episodes of ϵ-greedy exploration in the MountainCar-v0 environment.

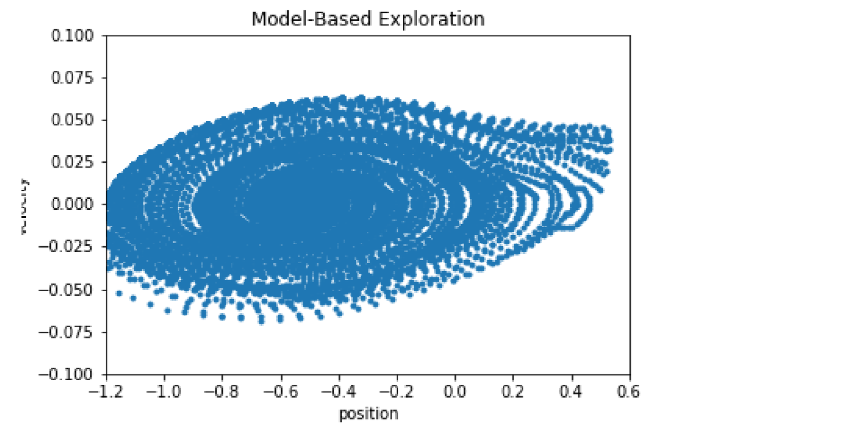

Figure 7: 50 episodes of model-based exploration in the MountainCar-v0 environment.

  - Learning the model's dynamics to generate simulated experience and allow planning (Dyna [7])
- Investigate more complex environments (e.g. Atari games [8]).

References

[1] Christopher J. C. H. Watkins and Peter Dayan.
Q-learning.
Machine Learning, 8(3):279–292, May 1992.

[2] Marc G. Bellemare, Will Dabney, and Rémi Munos.
A distributional perspective on reinforcement learning.
In *ICML*, 2017.

[3] Will Dabney, Mark Rowland, Marc G. Bellemare, and Rémi Munos.
Distributional reinforcement learning with quantile regression.
In *AAAI*, 2018.

[4] William R. Clements, BenoÃőt-Marie Robaglia, Bastien Van Delft, Reda Bahi Slaoui, and SÃľbastien Toth.
Estimating risk and uncertainty in deep reinforcement learning, 2019.

[5] Nikolay Nikolov, Johannes Kirschner, Felix Berkenkamp, and Andreas Krause.
Information-directed exploration for deep reinforcement learning.
CoRR, abs/1812.07544, 2018.

[6] Borislav Mavrin, Shangtong Zhang, Hengshuai Yao, Linglong Kong, Kaiwen Wu, and Yaoliang Yu.
Distributional reinforcement learning for efficient exploration, 2019.

[7] Richard S. Sutton.
Dyna, an integrated architecture for learning, planning, and reacting.
SIGART Bull., 2(4):160–163, July 1991.

[8] Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling.
The arcade learning environment: An evaluation platform for general agents.
J. Artif. Int. Res., 47(1):253–279, May 2013.