2020 10 21 Distributional Reinforcement Learning with Quantile Regression

Chen Gong

21 October 2020

这篇文章，承接的是 Marc G. Bellemare 发表的工作 “A Distributional Perspective on Reinforcement Learning ”。这一系列工作非常的具有启发性。传统强化学习中将价值函数考虑为一个期望值，而当价值函数不仅仅是一个期望值而是一个分布会怎样? 相应的 Bellman Operator 是否还具有较好的性质? 是否能够形成一个有效的算法? 把分布考虑进来是否能够带来算法性能的提升? 这是分布式强化学习所要解决的问题。在 C51 算法中对分布的近似方式是将概率密度函数用格子来近似（categorical representation），而 “Distributional Reinforcement Learning with Quantile Regression” 中使用了更加有效的近似方式（quantile representation），同时理论上也更有保证。

1 Conference Name

Thirty-Second AAAI Conference on Artificial Intelligence (AAAI 2020)

2 Thesis statement

2.1 大致思想描述

熟悉这篇论文，有两个问题非常重要。1. C51 算法有怎样的缺陷？2. 为什么要提出 QR 算法，QR 算法和 C51 算法相比有怎样的好处？

C51 算法中证明了，分布式贝尔曼算子在无穷范数的 Wasserstein 度量下是 contraction 算子。并且不会出现 disjoint-support 问题（WGAN 的论文中提出的，KL 散度无法度量支集没有交叠的两个概率分布，而 Wasserstein 却可以很好地描述任意概率分布之间的距离）。

使用 Wasserstein 度量非常的好，可惜的是 C51 算法中，并没有将这个理论直接运用到实际算法中，因为作者在 **Proposition 5** 和 **Lemma 7** 中，给出了证明不能直接对 Wasserstein 度量直接使用随机梯度下降，因为随机梯度下降法采样得到的梯度和原梯度不一样（Bellemare 这帮作者非常喜欢举例子来帮助读者理解生涩的定理，这种写作方法非常棒，可以学习）：下面给出证明，

Lemma 7 (Sample Wasserstein distance)：令 $\{P_i\}$ 是随机变量的集合，其中 $I \in \mathbb{N}$ 是指示标量。那

么 $P = P_I$ 是一个混合变量，并且所有的随机变量都与 Q 独立。令 $A_i = \mathbb{I}[I = i]$，

$$\begin{aligned} d_p(P, Q) &= d_p\left(P_I, Q\right) \\ &= d_p\left(\sum_i A_i P_i, \sum_i A_i Q\right) \\ &\leq \sum_i d_p\left(A_i P_i, A_i Q\right) \\ &\leq \sum_i \Pr\{I = i\} d_p\left(P_i, Q\right) \\ &= \mathbb{E}_I d_P\left(P_i, Q\right) \end{aligned} \tag{1}$$

其中第二行到第三行是用到了距离空间的基本性质，两边之和大于第三边。根据公式 (1)，可以得到严格的不等式：

$$d_p(P_I, Q) \leq \mathbb{E}_I d_P\left(P_i, Q\right) \tag{2}$$

根据上述不等式可以得到：

$$\nabla_Q d_p(P_I, Q) \neq \nabla_Q \mathbb{E}_I d_P\left(P_i, Q\right) \tag{3}$$

所以，不能对 Wasserstein 测度直接使用随机梯度法进行优化。图 1 所示的为 Wesserstein 距离无法用于随机梯度计算举例。

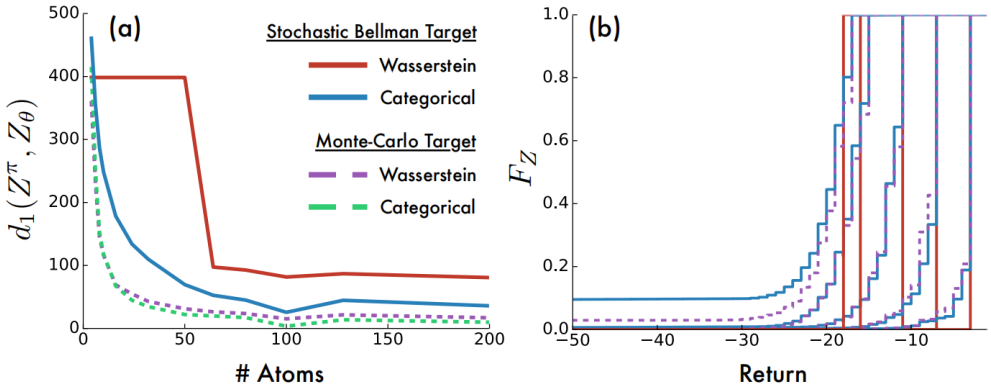

图 1: Wesserstein 距离无法用于随机梯度计算举例

由于上述种种问题，是否可以采用在线的分布式强化学习算法成为了一个问题。理论上，任何的实际表示方法都不能表示任意的随机概率分布。因此任何概率分布的表示方式，都可以看做真实分布向所能表示的空间上投影（基于从真实概率分布中的采样）。**使用 Wasserstein Metric 可以保证我们的 Bellman operator 是 $\gamma-$contraction 算子，可以收敛到唯一的不动点。但是对于 Q-learning 用到的 optimality operator $\mathcal{T}$，并不是 contraction 算子，而且也不能通过随机梯度下降来优化。**于是，作者们提出了一个启发式的方法，干脆用 KL 散度去衡量两个分布的距离。尽管 C51 算法可以取得较好的效果，但是实际上，这样的处理方法和作者提出的分布式强化学习的理论有较大的 GAP，并且不能用到 Wasserstein 距离良好的性质。

其中，理论上主要关心两件事情。第一件事情，在任意的 policy evaluation 下 Bellman 算子多轮迭代后能够收敛，它联合与表示有关的投影算子后（即 $\Pi\mathcal{T}^\pi$ ）是否还能在 Wasserstein 距离度量下收

敛呢；第二件事情，考虑采样之后，其实是在缩小所表示的概率分布和样本集所代表分布之间的距离，当所表示的概率分布和样本集上距离最小的时候，它是否也和真实分布距离最小。而 C51 算法，上述两个条件都没有达到。但是由于其做法，也能满足分布式强化学习的初衷，所以，实际取得了较好的效果。

2.2 C51 算法概况

在 DQN 算法中，假设动作空间 $\mathcal{A}$ 的大小为 m。网络的输入是状态 s，输出是一个关于动作的分布 $(Q(s,a_1),Q(s,a_2),\cdots,Q(s,a_N))$。

传统强化学习中的值函数 $Q(s,a)$ 是一维的数，而在 DRL 中，我们想要学习到值函数分布 $Z(s,a)$，其中 $Q(s,a)=\mathbb{E}[Z(s,a)]$。那么，在 Distributional DQN 中，输入依旧是一个状态 s，输出则应该变成了一个矩阵，这个矩阵的每一行代表一个动作 a，而且列数则和选择的 atoms 的个数相等，我们设这个数等于 N。在 C51 中，显然有 $N=51$。大家看到这里可能有些迷糊，下面来举个例子：

首先需要明确，atom 是固定的，假设我们估计值函数的区间大致为 $[-25,25]$。根据 $N=51$，将此区间划分成 51 等份：$z=\{z_0,z_1,z_2,\cdots,z_{N-1}\}=\{-25,-24,-23,\cdots,23,24,25\}$。那么 $C51$ 算法每行的输出为一个状态下，atom 中每个值出现的概率，比如在状态 s 下，选择状态 a_i，得到的价值 -25 为概率为 $p_0^a(s)$，得到的价值 -24 为概率为 $p_1^a(s)$，以此类推，即为 $\{p_0^a(s),p_1^a(s),\cdots,p_{N-1}^a(s)\}$。在 C51 算法中，输入一个状态 s，通过网络 $f(x,\theta)$，输出的为矩阵：

$$\begin{bmatrix} p_0^{a_0}(s) & p_1^{a_0}(s) & \cdots & p_{N-1}^{a_0}(s) \\ p_0^{a_1}(s) & p_1^{a_1}(s) & \cdots & p_{N-1}^{a_1}(s) \\ p_0^{a_2}(s) & p_1^{a_2}(s) & \cdots & p_{N-1}^{a_2}(s) \\ \vdots & \vdots & \ddots & \vdots \\ p_0^{a_m}(s) & p_1^{a_m}(s) & \cdots & p_{N-1}^{a_m}(s) \end{bmatrix}_{m\times n}$$

所以就会有 $Q(s,a_n)=\mathbb{E}[Z(s,a_n)]=\sum_{i=1}^{N} z_i\cdot p_i^{a_n}(s)$。这样 C51 算法的网络框架就定义完了。

前面 2.1 中讲到了，由于种种原因，作者采用了一个启发式的方法，干脆用 KL 散度去衡量两个分布的距离。但是使用 KL 散度会出现一个问题，atom 会移动不再对齐。把整个算法流程走一遍就清楚了。对于网络的输出可以计算出 $Q(s,a)$，

$$\begin{bmatrix} Q(s,a_1) \\ Q(s,a_2) \\ Q(s,a_3) \\ \vdots \\ Q(s,a_m) \end{bmatrix}_{m\times 1} = \begin{bmatrix} p_0^{a_0}(s) & p_1^{a_0}(s) & \cdots & p_{N-1}^{a_0}(s) \\ p_0^{a_1}(s) & p_1^{a_1}(s) & \cdots & p_{N-1}^{a_1}(s) \\ p_0^{a_2}(s) & p_1^{a_2}(s) & \cdots & p_{N-1}^{a_2}(s) \\ \vdots & \vdots & \ddots & \vdots \\ p_0^{a_m}(s) & p_1^{a_m}(s) & \cdots & p_{N-1}^{a_m}(s) \end{bmatrix}_{m\times n} \times \begin{bmatrix} z_0 \\ z_1 \\ z_1 \\ \vdots \\ z_{N-1} \end{bmatrix}_{n\times 1} \tag{4}$$

从 $Q(s,a)$ 中选择值最大的动作 a^*，并且 a^* 所在的那一列都取出来：$\{p_0^{a^*}(s),p_1^{a^*}(s),\cdots,p_{N-1}^{a^*}(s)\}$。计算下一个时刻 s' 所对应的向量为：$\{p_0^{a^*}(s'),p_1^{a^*}(s'),\cdots,p_{N-1}^{a^*}(s')\}$。这样我们将有：

- $\{p_0^{a^*}(s')$ 的概率取 $r+\gamma z_0$；
- $\{p_1^{a^*}(s')$ 的概率取 $r+\gamma z_1$；
- $\{p_1^{a^*}(s')$ 的概率取 $r+\gamma z_1$；

- ···

大家这时是不是发现原来的 atom:z_i 都变成了 $r+\gamma z_i$。而之前说到了 atom 是不能变的，所以作者采用了一个变换 Φ 来 atom 变得重新对齐。这就是 C51 算法的思路。整个过程如下图所示：

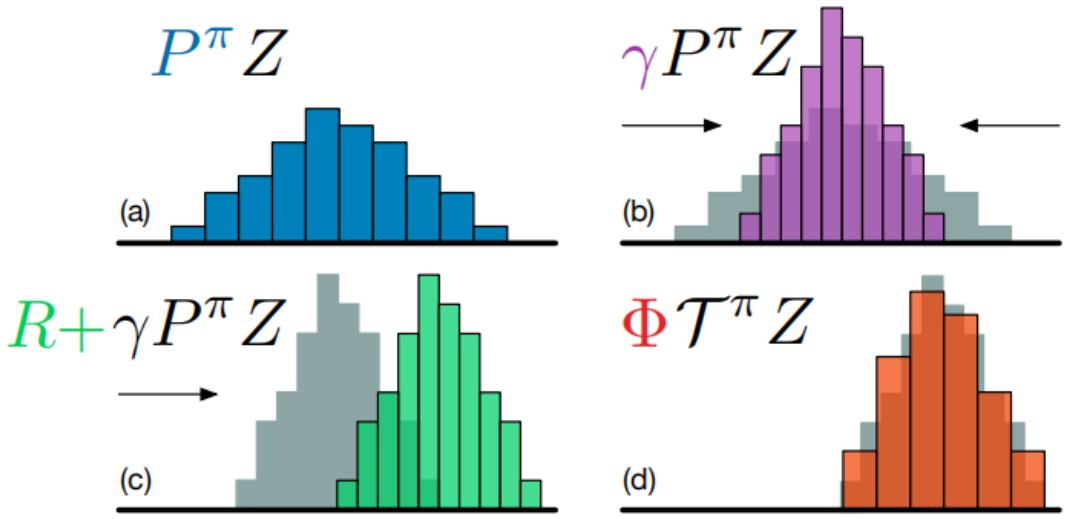

Figure 1. A distributional Bellman operator with a deterministic reward function: (a) Next state distribution under policy π, (b) Discounting shrinks the distribution towards 0, (c) The reward shifts it, and (d) Projection step (Section 4).

图 2: C51 算法训练过程

考虑价值函数分布的好处有以下三点：

1. 分布相比于均值能够对决策提供更多的信息，比如对于某些 risk sensitive 的情况，我们可能倾向于选择方差较小或者最坏情况较好的行动，而不是一味地选择均值较高的行动。

2. 对于某些表征上看起来一样的状态可能具有完全不一样的两个价值函数，如果仅仅考虑均值，信息就会被完全混淆。

3. 考虑价值函数的分布能够缓解奖励稀疏的问题，较为稀疏的奖励会在通常的迭代中慢慢“稀释”，如果像文中的做法，稀疏的奖励在传播过程中更容易被留存下来。大家通过这个算法流程，是不是可以清晰的感觉到即使是稀疏的奖励，也不会被扔掉，随着迭代的进行保留下来。

2.3 Wasserstein 距离

这里不做过多解释，网上资料很多，直观理解为：

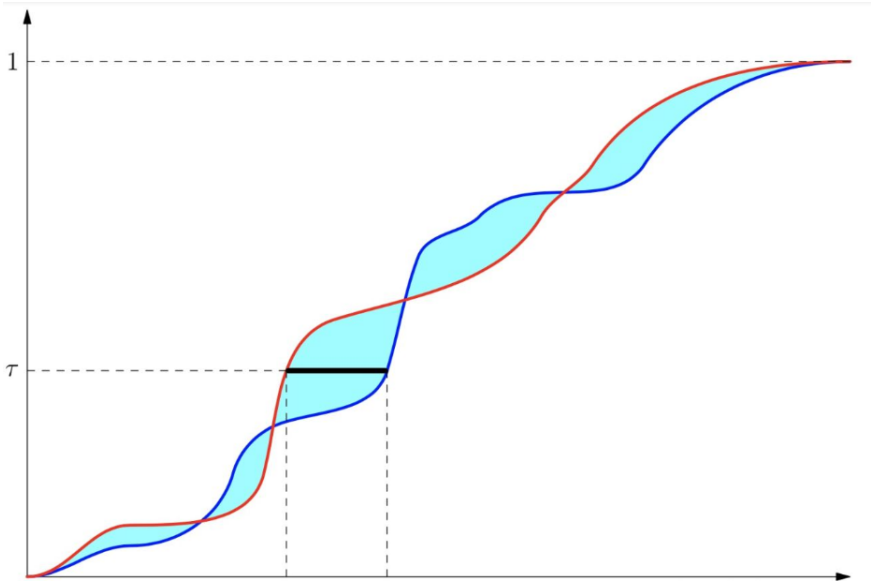

图 3: 两个分布间的 wasserstein 距离

2.4 本文的创新点

1. 改变了 C51 算法的参数形式，C51 算法中以值函数固定了一个 atom，实际上这个 atom 的变换求起来很麻烦。而本文使用的 atom 是分位对于的概率分布函数上的值，好处是不用再对齐了，因为我们的 atoms 的位置是可以改变的，而正是用这个变量来描述整个分布，自然没有对齐之说。(其实这里并不好理解，看到算法部分就知道了)。
2. QR 可以使用随机梯度下降法来求解，从而达到最小化两个分布间 Wasserstein 距离的效果。
3. 证明了本文的算法是 contraction 的，而且可以直接将对 Wasserstein 距离的优化用在 DRL 中。

(小编觉得这三句话放在前面，讲了跟没讲差不多，根本不能理解，需要看过 QR 算法的全貌之后，才能深刻的理解。)

直观的来讲，对比 C51 进步的地方就是，之前得到了很好的结论，但是没有找到对 Wasserstein 距离优化的方法，但是最终目的就是使两个分布靠近，所以有了用 KL 散度去近似的方法。但是实际应用和理论之间有很多的 GAP，而 QR 算法可以直接对 Wasserstein 距离进行优化，而且没那么复杂，所以和理论是一致的，当然效果会更好。而且本来 Wasserstein 距离对比 KL 散度就更优秀，KL 散度连距离都不算（KL 散度不具有对称性。problem：为什么 KL 散度不算距离不好 ?)。

3 Methods

下面我们将全面的分析，QR 算法的算法过程和理论原理。可以用 $Z(x,a)$ 表示值分布，准确的说，它可以表示成下面的形式:

$$Z : \mathcal{X} \times \mathcal{A} \to \mathscr{P}(\mathbb{R}) \tag{5}$$

表示为由一个状态，和动作，输出关于价值的分布。定义函数 Z 所在的空间:

$$\mathcal{Z} = \{Z : \mathcal{X} \times \mathcal{A} \to \mathscr{P}(\mathbb{R}) \mid \mathbb{E}\left[|Z(x,a)|^p\right] < \infty, \forall (x,a), p \geq 1\} \tag{6}$$

其中，对于 $\mathbb{E}\left[|Z(x,a)|^p\right] < \infty$，可以理解为 Q 的值，必然是小于 ∞。

3.1 分布式 Bellman 算子的收敛性

对于两个动作价值分布 $Z_1, Z_2 \in \mathcal{Z}$，他们之间的距离被定义为：

$$\bar{d}_p\left(Z_1, Z_2\right) := \sup_{x,a} W_p\left(Z_1(x,a), Z_2(x,a)\right) \tag{7}$$

在 C51 算法的文章中 (Lemma3)，已经证明了分布式贝尔曼算子 $\mathcal{T}^\pi$ 是 $\gamma-$contraction 算子。下面给出证明：

根据定义可以得到：

$$\bar{d}_p\left(\mathcal{T}^\pi Z_1, \mathcal{T}^\pi Z_2\right) = \sup_{x,a} d_p\left(\mathcal{T}^\pi Z_1(x,a), \mathcal{T}^\pi Z_2(x,a)\right) \tag{8}$$

$$\begin{aligned} d_p\left(\mathcal{T}^\pi Z_1(x,a), \mathcal{T}^\pi Z_2(x,a)\right) &= d_p\left(R(x,a) + \gamma P^\pi Z_1(x,a), R(x,a) + \gamma P^\pi Z_2(x,a)\right) \\ &\leq \gamma d_p\left(P^\pi Z_1(x,a), P^\pi Z_2(x,a)\right) \\ &\leq \gamma \sup_{x',a'} d_p\left(Z_1\left(x',a'\right), Z_2\left(x',a'\right)\right) \end{aligned} \tag{9}$$

结合上述两个不等式可以计算出：

$$\begin{aligned} \bar{d}_p\left(\mathcal{T}^\pi Z_1, \mathcal{T}^\pi Z_2\right) &= \sup_{x,a} d_p\left(\mathcal{T}^\pi Z_1(x,a), \mathcal{T}^\pi Z_2(x,a)\right) \\ &\leq \gamma \sup_{x',a'} d_p\left(Z_1\left(x',a'\right), Z_2\left(x',a'\right)\right) \\ &= \gamma \bar{d}_p\left(Z_1, Z_2\right) \end{aligned} \tag{10}$$

但是，在 2.1 中已经分析了，**1. 我们不能使用随机梯度下降法最小化 Wasserstein 度量，，导致价值分布只能近似**。然后，文章中回顾了 C51 算法的第二个缺陷，**2. Categorical representation 在样本上的最优不等于相对于真实分布的最优**。而且 C51 算法还不能保证最小化 p-Wasserstein 度量。但是 C51 算法中，使用 Categorical representation 仍然可以达到分布式强化学习原始的目的。所以，表现出了较好的效果。

3.2 近似最小化 Wasserstein

前面吐槽了 C51 算法的缺点，那么接下来当然就是说本文提出的 QR 算法有多么的好了，更加的符合 DRL 本身理论所提出的思想。由于不能对 Wasserstein 度量直接使用随机梯度下降进行优化，所以，只能近似最小化 Wasserstein 距离。

C51 的做法，我已经在 2.2 小节中做了详细的解释。QR 算法中，在参数的设置上有了一定的改动。因为 C51 算法中的设计机制，计算 KL 散度是极好的，但是不适合计算 Wasserstein 度量，QR 算法中采用的是分位数的描述方法。

QR 算法中，不再研究概率密度函数了，研究概率分布函数。将概率分布函数分成 N 等份，那么自然就会得到 10 个 $\{\tau_i\}_{i=1}^{10}$，这 10 个 τ 就定义了 10 个分位数，而在每个区间 $[\tau_i, \tau_{i+1}]$ 有那么多数，我们优化哪个数是最好的呢，这个数就是中点（之后 3.3 小节给出了证明）。

$$\hat{\tau}_i = \frac{2(i-1)+1}{2N}, \quad i = 1, \ldots, N \tag{11}$$

每一个 $\hat{\tau}_i$ 对应一个 z_i，表示的是 $P(z \leq z_i) = \hat{\tau}_i$。如下所示，分位点在概率密度函数上的对应：

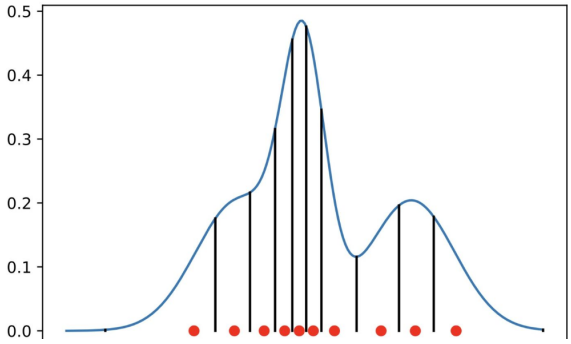

图 4: 分位点在概率密度函数上的体现

我们知道分位点将概率分布函数分成了 N 等份，那么在概率密度函数上反映，即为每两个分位点直接的面积为 $\frac{1}{N}$(反映在图中就是黑线)。可以近似等价于，z_i 出现的概率是 $\frac{1}{N}$，那么 $\sum_{i=1}^{N} \frac{1}{N} z_i$ 即为分布的均值。**这段话，对于理解分位点的本质非常重要**。

在深度学习中，令 $\theta : \mathcal{X} \times \mathcal{A} \to \mathbb{R}^N$ 为参数模型，分位点模型的输入为 (x, a) 输出是每个分位点对应的值，$P(z \leq z_i) = \hat{\tau}_i$，$\{z_1, z_2, \cdots, z_N\}$(前面 C51 算法是从 0 开始，QR 是从 1 开始，都是一样的，大家不用介意)。冲击函数 δ_t 代表的意思是：

$$\delta_t = \begin{cases} 1 & x = t \\ 0 & \text{else} \end{cases} \tag{12}$$

由于我们近似等价 z_i 出现的概率是 $\frac{1}{N}$，所以，**分位数分布**写为：

$$Z_\theta(x, a) := \frac{1}{N} \sum_{i=1}^{N} \delta_{\theta_i(x,a)} \tag{13}$$

分位数分布表示 $(\{P(z_1), P(z_2), \cdots, P(z_N)\})$ 对应的概率都是 $\frac{1}{N}$。(小编是这样理解的，如图 4 所示，概率密度约高的区域，得到 z_i 的可能性就更高，所以分位点出现的位置是由概率密度所决定的，有点类似于均匀的概率分布采样)

对比原来的参数，可以从三个方面来分析分位数分布的优点。

1. 最直观的感受即为，区间不需要固定了，因为 C51 算法中是固定价值函数值计算概率，而 QR 是固定概率计算函数值，正好反过来的。而在使用贝尔曼算子迭代的时候，价值函数值会改变，而概率值是固定的。

2. categorical representation 需要实现确定价值函数值所在的区间 $[V_{min}, V_{max}]$，这引入了超参数；同时，对于某些状态价值函数分布范围相对于 $[V_{min}, V_{max}]$ 很小的情况下，近似非常不准确。而 quantile representation 就不存在此问题。因为 KL 散度有 disjointed supports 的问题。

3. 我们可以对分位数分布进行优化，来最小化 Wasserstein 距离，不会遇到存在梯度偏差的情况。但是很可惜，当所表示的概率分布和样本集上距离最小的时候，它不是和真实分布距离最小。

3.3 分位点近似

既然已经知道了分位点回归的网络架构了，下一步则是如何训练的问题，以及背后的理论可行性。

本文中假设 Y 是一个一阶矩有界的分布，和我们公式 (13) 中所描述的 N 个冲击函数组成的分布 $\{z_1, \cdots, z_N\}$（论文中记为 $\{\theta_1, \cdots, \theta_N\}$），之间的 Wasserstein 距离被记为：

$$W_1(Y, U) = \sum_{i=1}^{N} \int_{\tau_{i-1}}^{\tau_i} \left| F_Y^{-1}(\omega) - \theta_i \right| d\omega \tag{14}$$

就如同论文中表示的一样，其中黑色的线是 Y 分布，蓝色的线是我们 N 个冲击函数组成的分布。

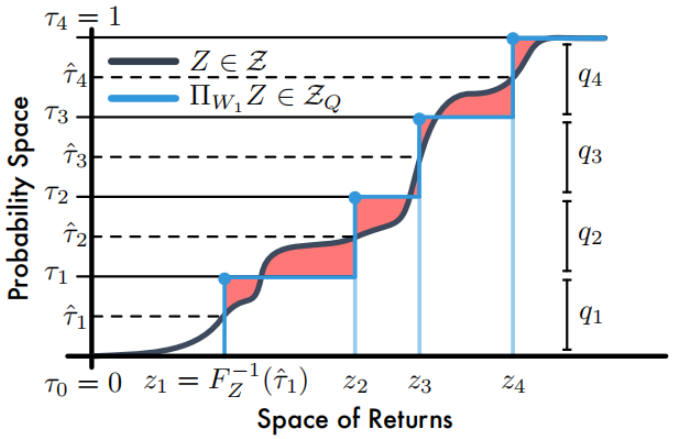

图 5: 两分布之间的 Wasserstein 距离

既然得到了近似表示方法，那么就要证明最优 quantile representation 存在并且合理。

Lemma 2 对于任意的 $\tau, \tau' \in [0, 1](\tau < \tau')$，概率分布函数为 F，其逆函数为 F^{-1}。最小化 θ，其中，

$$\int_{\tau}^{\tau'} \left| F^{-1}(\omega) - \theta \right| d\omega \qquad \left\{ \theta \in \mathbb{R} \mid F(\theta) = \left(\frac{\tau + \tau'}{2} \right) \right\} \tag{15}$$

其证明思路为观察到此函数是一个凸函数，然后直接求得令导数值等于零。解得，令 $\int_{\tau}^{\tau'} |F^{-1}(\omega) - \theta| \, d\omega$ 最小的点就是 $F(\theta) = \left(\frac{\tau + \tau'}{2} \right)$。**所以这就证明出了，我们为什么要用神经网络取拟合分位点的中点**（其实感觉这是显然的事情）。

任何实际的表示方式都不能完全准确表示一个任意概率分布，因此任何概率分布的表示方式都可以看做真实分布向所能表示的空间上投影（基于从真实概率分布中采集样本集）。理论上关心两件事情。第一件事情，在随意 policy evaluation 下 Bellman 算子多轮迭代后能够收敛，但是它联合与表示有关的投影算子后（即 $\Pi \mathcal{T}^{\pi}$ ）是否还能在 Wasserstein 距离度量下收敛呢；第二件事情，考虑采样之后，其实是在缩小所表示的概率分布和样本集所代表分布之间的距离，当所表示的概率分布和样本集上距离最小的时候，它是否也和真实分布距离最小。很可惜 C51 算法，上述两个条件都没有达到。下面我们将对这两个条件进行分析。

本文感兴趣的是量化任意值分布 $Z \in \mathcal{Z}$ 在 Z_Q 上的投影，就是，

$$\Pi_{W_1} Z := \underset{Z_\theta \in \mathcal{Z}_Q}{\arg\min} W_1 \left(Z, Z_\theta\right) \tag{16}$$

很可惜的是上述的第二条，QR 算法仍然不能成立。当所表示的概率分布和样本集上距离最小的时候，它不是和真实分布距离最小。

Proposition 1. 令 Z_θ 是分位数分布，$\hat{Z}_m$ 是从真实分布 Z 中采样出的 m 个样本。对于任意的 $p \leq 1$，

$$\arg\min \mathbb{E}\left[W_p\left(\hat{Z}_m, Z_\theta\right)\right] \neq \arg\min W_p\left(Z, Z_\theta\right) \tag{17}$$

证明：$Z_\theta(x,a) := \frac{1}{N}\sum_{i=1}^N \delta_{\theta_i(x,a)}$，其中 $\theta_1 \leq \theta_2 \leq \cdots \theta_N$。我们同样将任意的分布 Z 写成同样的形式，

$$Z := \frac{1}{N}\sum_{i=1}^N \delta_i$$

其证明思路就是举反例。假设真实的分位数分布为：$X = \{1, 2, \cdots, N\}$，其中每个数出现的概率都是 $\frac{1}{N}$。那么从真实分布采样得到的样本里面最小值或者是 1，或者大于 1。我们的目标是要找到一个 Z_θ，最小化 $W_p(Z, Z_\theta)$，使得 $Z = Z_\theta$。**证明思路就是当 $Z = Z_\theta$ 最小时，是否可以令 $\mathbb{E}[W_p(\hat{Z}_N, Z_\theta)]$ 也最小。**

假设估计分布的第一个参数已经最优了，即 $\theta_1 = 1$。那么，必然会有，

$$\nabla_{\theta_1} \mathbb{E}\left[W_p\left(\hat{Z}_N, Z_\theta\right)\right]\Big|_{\theta_1=1} = \mathbb{E}\left[\nabla_{\theta_1} W_p\left(\hat{Z}_N, Z_\theta\right)\Big|_{\theta_1=1}\right] = 0 \tag{18}$$

但实际情况是当样本的最小值等于 1 时，导数为零；采到的样本大于 1 的时候，导数小于零。因此其导数的期望一定为负，因此，最后必然会得到，

$$\nabla_{\theta_1} \mathbb{E}\left[W_p\left(\hat{Z}_N, Z_\theta\right)\right]\Big|_{\theta_1=1} < 0 \tag{19}$$

那么，θ_1 最后肯定会收敛到比 1 更大的某个位置。所以，$Z_\theta = Z$ 并不能使 $\mathbb{E}[W_p(\hat{Z}_N, Z_\theta)]$ 最小。

第二个大问题就是证明，上述的第一条，考虑 policy evaluation，quantile representation 的投影联合 Bellman 算子，仍然能形成 contraction。虽然第一个条件不能满足，但是第二个条件是可以满足的。所以，QR 算法比 C51 算法，理论上还是改进了很多。我们要证明的是：

$$\bar{d}_\infty\left(\Pi_{W_1}\mathcal{T}^\pi Z_1, \Pi_{W_1}\mathcal{T}^\pi Z_2\right) \leq \gamma \bar{d}_\infty\left(Z_1, Z_2\right) \tag{20}$$

在之前已经证明了在 $\bar{d}_\infty$ 中贝尔曼算子 $\mathcal{T}^\pi$ 是 contraction 算子。为了 quantile 表示算子和 Bellman 算子的联合投影算子是 contraction 算子，首先要对值分布 $Z(x,a)$ 使用 quantile 表示算子，将其转换为单一的冲击函数的和的形式。将 $Z(x,a)$ 和目标分布 $Y(x,a)$ 分别记为：

$$Z(x,a) = \sum_{k=1}^N \frac{1}{N}\delta_{\theta_k(x,a)}, \qquad Y(x,a) = \sum_{k=1}^N \frac{1}{N}\delta_{\psi_k(x,a)} \tag{21}$$

其中，$\theta, \psi : \mathcal{X} \times \mathcal{A} \to \mathbb{R}^n$。$(x,a)$ 是状态动作二元组。其中，$((x_i, a_i))_{i\in I}$ 表示从一个 (x', a') 可以转移到的 p_i 表示，从 $(x', a,)$ 转移到 (x_i, a_i) 的概率。下面将建立一个全部由单一的冲击函数组成的，MDP

和价值分布。新的 MDP 如下分析，从一个状态 (x', a') 在策略 $\widetilde{\pi}$ 下可能转移到 $\left(\left(\widetilde{x}_i^j, \widetilde{a}_i^j\right)_{i\in I}\right)_{j=1}^N$，并且到达每一个 $\left(\widetilde{x}_i^j, \widetilde{a}_i^j\right)$ 的概率都是 $\frac{\pi}{n}$。这样就可以定义新的价值分布了：

$$\begin{aligned}\widetilde{Z}\left(\widetilde{x}_i^j, \tilde{a}_i^j\right) &= \delta_{\theta_j(x_i,a_i)} \\ \widetilde{Y}\left(\widetilde{x}_i^j, \widetilde{a}_i^j\right) &= \delta_{\psi_j(x_i,a_i)}\end{aligned} \tag{22}$$

下图中，上面的图的表示之前的 MDP 和价值分布 Z，而下面的图表示的是转换后的 MDP 和价值分布 $\widetilde{Z}$。

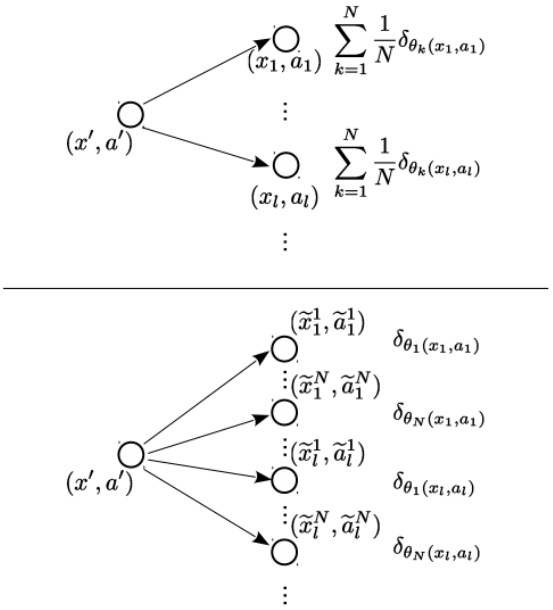

Figure 5: Initial MDP and value distribution Z (top), and transformed MDP and value distribution $\widetilde{Z}$ (bottom).

图 6: 传统的 MDP 和价值分布 Z 和转移后的 MDP 和价值分布 $\widetilde{Z}$ 对比

注意，Z 和 $\widetilde{Z}$ 是同一个价值分布的不同表现形式而已，其本质上都是一样的。作者提到，r 和 γ 对于研究 contraction 性质没什么关系，所以在文章证明中，就简单的认为 $r = 0$，$\gamma = 1$。那么，证明被简化成：

$$\bar{d}_\infty\left(\Pi_{W_1}\mathcal{T}^\pi Z_1, \Pi_{W_1}\mathcal{T}^\pi Z_2\right) \leq \bar{d}_\infty\left(Z_1, Z_2\right) \tag{23}$$

由于，不同分位数的投影算子互不影响，因此只需要分析一个分位数即可。将问题进一步简化，根据图 6 中看到的，我们把对于每一个 (x, a) 的冲击函数组成的分位数分布都展开了，所以可以将分布考虑为单冲击函数的情况。其他情况可以看做冲击函数的加和，本质上都是一样的。那么原始的证明被简化成了，证明单冲击函数组成的分位数分布中一个分位点的情况，

$$\bar{d}_\infty\left(\Pi_\tau\mathcal{T}^\pi Z_1, \Pi_\tau\mathcal{T}^\pi Z_2\right) \leq \bar{d}_\infty\left(Z_1, Z_2\right) \tag{24}$$

因为将分布考虑为单冲击函数，所以令 $Z(x, a) = \delta_{\theta(x,a)}, Y(x, a) = \psi_{\theta(x,a)}$。其中，$\theta(x_i, a_i)$ 和 $\psi(x_i, a_i)$

简写为 θ_i 和 ψ_i。另 (x', a') 转移到 (x_i, a_i) 的概率是 p_i，那么有

$$\begin{aligned}(\mathcal{T}^\pi Z)(x', a') &= \sum_{i \in I} p_i \delta_{\theta_i} \\ (\mathcal{T}^\pi Y)(x', a') &= \sum_{i \in I} p_i \delta_{\psi_i}\end{aligned} \tag{25}$$

令 θ_u 等于 $(\mathcal{T}^\pi Z)(x', a')$ 在分位点 τ 对应的值，ψ_u 也是同理。那么有，

$$\bar{d}_\infty(\Pi_\tau \mathcal{T}^\pi Z(x', a'), \Pi_\tau \mathcal{T}^\pi Z(x', a')) = |\theta_u - \psi_v| \tag{26}$$

我们要证的是 $|\theta_u - \psi_v| \leq |\theta_i - \psi_i|$。此处采用反证来进行证明，假设 $\theta_u < \theta_v$，有

$$|\theta_u - \psi_v| > |\theta_i - \psi_i|$$

我们将整体区域 I，进行如下所示的划分：

$$\begin{aligned}I_{\leq \theta_u} &= \{i \in I \mid \theta_i \leq \theta_u\} \\ I_{> \theta_u} &= \{i \in I \mid \theta_i > \theta_u\} \\ I_{< \psi_v} &= \{i \in I \mid \psi_i < \psi_v\} \\ I_{\geq \psi_v} &= \{i \in I \mid \psi_i \geq \psi_v\}\end{aligned} \tag{27}$$

且有：

$$\begin{aligned}I &= I_{\leq \theta_u} \cup I_{> \theta_u} \\ I &= I_{< \psi_v} \cup I_{\geq \psi_v}\end{aligned} \tag{28}$$

如何这个不等式成立的话，一定不会出现 $\theta_u < \theta_i$ 且 $\psi_v > \psi_i$ 的情况。所以，$I_c \leq \theta_u \cap I_{\geq \psi_v} = \emptyset$，可以推出，$I_{\leq \theta_u} \subset I_{< \psi_v}$。但是，$\theta_u$ 是 $(\mathcal{T}^\pi Z)(x', a')$ 的 τ 分位点对应的值，必然有：

$$\sum_{i \in I < \theta_u} p_i \geq \tau \rightarrow \sum_{i \in I < \psi_v} p_i \geq \tau \tag{29}$$

这显然是矛盾的，$\sum_{i \in I < \psi_v} p_i < \tau$。所以，我们可以证明对于单一冲击函数中的一个分位点有，

$$\bar{d}_\infty(\Pi_\tau \mathcal{T}^\pi Z_1, \Pi_\tau \mathcal{T}^\pi Z_2) \leq \bar{d}_\infty(Z_1, Z_2) \tag{30}$$

最后，将此结论推广到一般情况就很简单了。

3.4 QR 算法的目标函数

在传统的回归中，对于一个自变量 x，我们想要条件概率 $p(y|x)$ 的期望，因此我们的损失函数是，

$$\mathcal{L}_{\text{MSE}} = \min_\beta \sum_i^n (y_i - \mu(x_i, \beta))^2 \tag{31}$$

其中，$\mu(\cdot, \beta)$ 表示以 β 为参数的均值模型，输入 x_i 输出为对 label: y_i 的预测值。

如果我们想求 $p(y|x)$ 的中位值呢，损失函数被定义为：

$$\begin{aligned}\mathcal{L}_{\text{MAE}} &= \sum_i^n |y_i - \xi(x_i, \beta)| \\ &= \sum_{i: y_i \geq \xi(x_i, \beta)} (y_i - \xi(x_i, \beta_\tau)) + \sum_{i: y_i < \xi(x_i, \beta)} (\xi(x_i, \beta) - y_i)\end{aligned} \tag{32}$$

而且其梯度求解也比较简单，

$$\nabla_\beta \mathcal{L}_{MSE} = \sum_i^n \nabla_i, \quad \nabla_i = \begin{cases} 1 & i : y_i < \xi(x_i, \beta) \\ -1 & i : y_i \geq \xi(x_i, \beta) \end{cases} \tag{33}$$

实际上，很简单，可以理解为向中间推的力量。其实，中位数就是 0.5 分位数。其他分位数，可以看成推的力量不一样，根据权重，推向不同的位置。

$$\mathcal{L}_\tau = \sum_{i:y_i \geq \xi(x_i,\beta_\tau)} \tau \left(y_i - \xi(x_i, \beta_\tau)\right) + \sum_{i:y_i < \xi(x_i,\beta_\tau)} (1-\tau)\left(\xi(x_i, \beta_\tau) - y_i\right) \tag{34}$$

同样，梯度表示为：

$$\nabla_\beta \mathcal{L}_\tau = \sum_i^n \nabla_i, \quad \nabla_i = \begin{cases} 1-\tau & i : y_i < \xi(x_i, \beta) \\ -\tau & i : y_i \geq \xi(x_i, \beta) \end{cases} \tag{35}$$

同样可以用另一种方法，用冲击函数来表示：$\mathcal{L}_\tau$。

$$\left|\tau - \delta_{y_i < \xi(x_i,\beta)}\right| = \begin{cases} |\tau - 1| = 1 - \tau & \text{if } y_i < \xi(x_i, \beta) \\ \tau & \text{if } y_i \geq \xi(x_i, \beta) \geq \theta \end{cases} \tag{36}$$

所以，公式 (23) 可以被写做，

$$\begin{aligned} \mathcal{L}_\tau = & \sum_{i:y_i \geq \xi(x_i,\beta_\tau)} \left|\tau - \delta_{y_i < \xi(x_i,\beta)}\right| \left(y_i - \xi(x_i, \beta_\tau)\right) + \sum_{i:y_i < \xi(x_i,\beta_\tau)} \left|\tau - \delta_{y_i < \xi(x_i,\beta)}\right| \left(\xi(x_i, \beta_\tau) - y_i\right) \\ = & \mathbb{E}\left[|y_i - \xi(x_i, \beta_\tau)| \left|\tau - \delta_{y_i < \xi(x_i,\beta)}\right|\right] \end{aligned} \tag{37}$$

对比到本文中，目标函数可以写为：

$$\begin{aligned} \mathcal{L}_{\mathrm{QR}}^\tau(\theta) = & \mathbb{E}_{\hat{Z} \sim Z}\left[\left|\hat{Z} - \theta\right| \left|\tau - \delta_{\hat{Z} < \theta}\right|\right] \\ = & \mathbb{E}_{\hat{Z} \sim Z}\left[\left|\hat{Z} - \theta\right| \left|\tau - \delta_{\hat{Z} - \theta < 0}\right|\right] \end{aligned} \tag{38}$$

观察可得，$\left|\hat{Z} - \theta\right|$ 和 $\left|\tau - \delta_{\hat{Z}-\theta<0}\right|$ 一定是同号的，所以，损失函数被化简为：

$$\mathcal{L}_{\mathrm{QR}}^\tau(\theta) = \mathbb{E}_{\hat{Z} \sim Z}\left[\rho_\tau\left(\hat{Z} - \theta\right)\right], \qquad \rho_\tau(u) = u\left(\tau - \delta_{\{u<0\}}\right), \forall u \in \mathbb{R} \tag{39}$$

最后，对于需要优化的 N 个参数，$\{\theta_1, \theta_2, \cdots, \theta_N\}$ 使 $W_1(Z, Z_\theta)$ 最小的目标函数为：

$$\sum_{i=1}^N \mathbb{E}_{\hat{Z} \sim Z}\left[\rho_{\hat{\tau}_i}\left(\hat{Z} - \theta_i\right)\right] \tag{40}$$

但是，有个很严重的问题，u 在 0 处不可导，计算上不太稳定，这里又提出了一种平滑的方案，即 quantile Huber loss。所以，可以把它换成其他的等价函数，论文中使用的是 Huber loss，

$$\mathcal{L}_\kappa(u) = \begin{cases} \frac{1}{2}u^2, & \text{if } |u| \leq \kappa \\ \kappa\left(|u| - \frac{1}{2}\kappa\right), & \text{otherwise} \end{cases} \tag{41}$$

那么，quantile Huber loss 被定义为：

$$\rho_\tau^\kappa(u) = \left|\tau - \delta_{\{u<0\}}\right| \mathcal{L}_\kappa(u) \tag{42}$$

如下图所示，举例 $\kappa = 1$ 的情况，

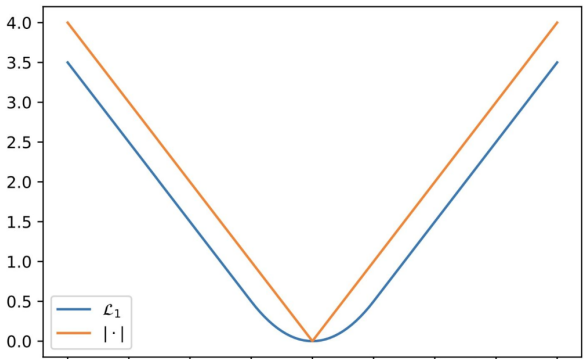

图 7: quantile Huber loss 与标准 quantile loss 之间的比较

结合公式 (40) 可以得到最终的损失函数。

4 QR 算法在强化学习中的使用

使用 QR 算法，可以在观测到目标分布 $Y(x)$ 中样本 $y \sim Y(x)$ 的情况下，优化上述的目标函数，最小化 $W_1(\hat{Z}, Z_\theta)$，达到近似分布的 QR 函数，从而达到逼近分布的效果。

根据前文给出的长篇大论，就给出了分位数回归时间差分学习 (QRTD) 算法，

$$\theta_i(x) \leftarrow \theta_i(x) + \alpha \left(\hat{\tau}_i - \delta_{\{r+\gamma z' < \theta_i(x))\}}\right) \quad a \sim \pi(\cdot \mid x), r \sim R(x,a), x' \sim P(\cdot \mid x,a), z' \sim Z_\theta(x') \tag{43}$$

其中，Z_θ 是分位数分布，$\theta_i(x)$ 是 $F^{-1}_{Z^\pi(x)}(\hat{\tau}_i)$。需要注意的是，对于每一个 $\hat{\tau}_i$ 和下一个状态价值分布的每一个样本之间进行更新，实际上，就是计算更新所有的 $(\theta_i(x), \theta_j(x'))$。这么说大家有点模糊，举个例子就可以了解了，下面开始介绍 QR-DQN。

首先从 Buffer 中采样出 (s, a, r, s')，接下来需要计算 a^*，那么首先需要计算 $Q(s', a')$。计算方法很简单：

$$Q(s', a') := \sum_j q_j \theta_j(x', a') \tag{44}$$

挑选其中最大的 a^*，$\arg\max_{a'} Q(x, a')$。根据 a^* 可以计算出 $Z(s', a^*) = \{\theta'_0, \theta'_1, \cdots, \theta'_{N-1}\}$。那么目标分布表示为：

$$\mathcal{T}\theta_j \leftarrow r + \gamma \theta_j(x', a^*), \quad \forall j \tag{45}$$

这里的好处是不用再对齐了，因为我们的 atoms 的位置是可以改变的，而正是用这个变量来描述整个分布，自然没有对齐之说。

那么，用 $\{\theta_0, \theta_1, \cdots, \theta_N\}$ 来描述 $Z(s, a)$，目标分布是

$$\mathcal{T}Z = r + \gamma Z(s', a^*) = \left\{r + \gamma\theta'_0, r + \gamma\theta'_1, \cdots, r + \gamma\theta'_{N-1}\right\}$$

其中，Z 和 $\mathcal{T}Z$ 中的元素分别可以用 θ_j 和 $\mathcal{T}\theta_j$ 来表示，都是一样的。此外，有 N 个 τ，需要计

算它们的损失函数的和:

$$\begin{aligned}\mathcal{L} &= \sum_{i=1}^{N} \mathbb{E}_j \left[\rho_{\hat{\tau}_i}^1 \left(\mathcal{T}\theta_j' - \theta_i\right)\right] \\ &= \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \left[\rho_{\hat{\tau}_i}^1 \left(\mathcal{T}\theta_j' - \theta_i\right)\right]\end{aligned} \tag{46}$$

其中,

$$\hat{\tau}_i = \frac{2i+1}{2N}, \quad i = 0, \ldots, N-1 \tag{47}$$

作者总结，DQN 和 QR-DQN 只有三点不一样:

1. 输出层数量从 $|\mathcal{A}|$ 变成了 $|\mathcal{A}| \times N$。
2. 使用 Quantile Huber loss 代替原来的 loss。
3. 将 RMSProp 优化器换成了 Adam 优化器。

依次看来，QR-DQN 算法的实现非常的简单。论文中给出的伪代码如下所示:

Algorithm 1 Quantile Regression Q-Learning

Require: N, κ

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 $\sum_{i=1}^{N} \mathbb{E}_j \left[\rho_{\hat{\tau}_i}^{\kappa}(\mathcal{T}\theta_j - \theta_i(x, a))\right]$

图 8: QR-DQN 伪代码

5 Conclusion and Discussion

这篇文章探究了 QR 算法在分布式强化学习中的应用，QR 算法对比 C51 算法，消除了分布式强化学习理论与实践上的 GAP，通过对分位数分布的优化，真正意义上实现了对 Wasserstein 度量的优化。

本文全文的思路，首先分析了分布式强化学习对比传统强化学习的各种优势。然后提出了之前提出的分布式强化学习算法 C51 算法的缺陷，1. 理论和实践之间有很大的 GAP；2. 无法使用随机梯度下降法对 Wasserstein 距离进行优化，所以文中提出了启发式格子表示方法，并采用 KL 散度进行优化；3. 而且 Bellman 算子和 Optimality 算子的联合投影算子并不是 contraction 的；4. 采样分布和模型分布之间的距离最小时，并不能保证模型分布和真实分布之间的距离也最小。

基于上述的种种问题，本文提出了 QR 算法，通过对分位数分布进行拟合，从而达到优化 Wasserstein 距离，而且对比 KL 散度，Wasserstein 距离的性质更好。提出了分位数分布的概念，使用一组冲击函数来表示分位数分布。然后分析了分位数表示比较与格子表示的优点：不要向格子表示那样考虑

区间对齐的问题。并且，作者对一个分位数区间 $[\tau, \tau']$，用哪一个数来表示可以使 Wasserstein 距离最小的问题进行了讨论，证明出中点表示最合适，所以在算法中，我们拟合的是分位数区间的中点分位数对应的值。这样，就自然而然的推出了网络的框架，确定输入和输出。网络框架确定之后，紧接着就是如何训练的问题。本文从传统的分位数回归算法出发，推出来本文适用的损失函数，由于原始的损失函数，在 0 点不可导的问题，文中使用了 Quantile Huber Loss，这是一种平滑的方案，可以使计算更加稳定。分析到这里，我们基本实现了用随机梯度下降法对 QUantile Huber Loss 进行优化，可以达到对 Wasserstein 距离进行优化的效果。但是，C51 问题中两个重要的理论问题，还没有解决。**本文中，证明 Bellman 算子和 Quantile 表示算子的联合投影算子是 contraction 的，但是采样分布和模型分布之间的距离最小时，还是不能保证模型分布和真实分布之间的距离也最小**。也算是解决了一个重要的问题。

最后，将 QR 算法结合 DQN 算法，将分布式强化学习的思路运用于实际中，详细的描述了 QR-DQN 算法的实际思路。实验的具体细节就不说了，有兴趣的同学自己看，反正就是实现了很好的效果。

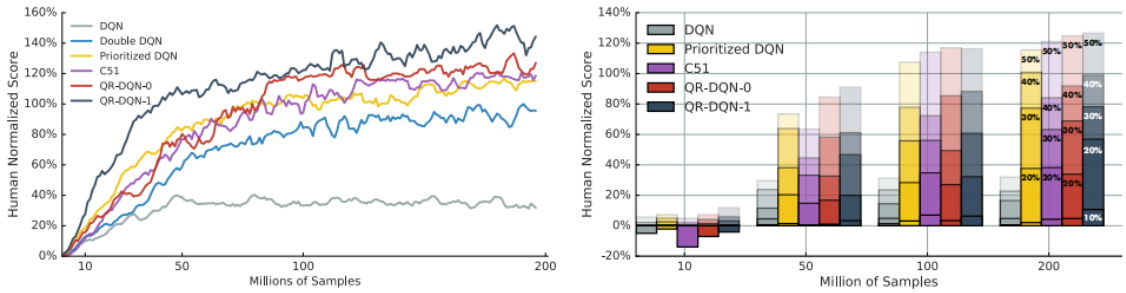

Figure 4: Online evaluation results, in human-normalized scores, over 57 Atari 2600 games for 200 million training samples. (Left) Testing performance for one seed, showing median over games. (Right) Training performance, averaged over three seeds, showing percentiles (10, 20, 30, 40, and 50) over games.

图 9: QR-DQN 实验效果对比

6 My Reflections

正如作者所说的那样，未来将考虑在控制算法和非线性函数近似上的应用。QR 算法的应用远远不止在分布式强化学习上使用那么简单，其可以解决，无法求解出其解析解，但是可以对其进行采样的复杂分布的近似问题。抛砖引玉!