TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

Bachelorarbeit

Deep Distributional Reinforcement Learning: From DQN to C51 and QR-DQN

ausgeführt zum Zwecke der Elangung des akademischen Grades Bachelor of Science (BSc) eingereicht an der TU Wien, Fakultät für Mathematik und Geoinformation von

Patricia Daxbacher
Mat.Nr.:
01607403

unter der Leitung von
Assoc. Prof. Dipl.-Ing. Dr.techn.
Clemens Heitzinger

Institut für Analysis und Scientific Computing
Technische Universität Wien
Wiedner Hauptstraße 8-10
1040 Wien, Österreich

Eidesstaatliche Erklärung

Ich erkläre an Eides statt, dass die vorliegende Arbeit nach den anerkannten Grundsätzen für wissenschaftliche Abhandlungen von mir selbstständig erstellt wurde. Alle verwendeten Hilfsmittel, insbesondere die zugrunde gelegte Literatur, sind in dieser Arbeit genannt und aufgelistet. Die aus den Quellen wörtlich entnommenen Stellen, sind als solche kenntlich gemacht.

Das Thema dieser Arbeit wurde von mir bisher weder im In- noch Ausland einer Beurteilerin/einem Beurteiler zur Begutachtung in irgendeiner Form als Prüfungsarbeit vorgelegt. Diese Arbeit stimmt mit der von den Begutachterinnen/Begutachtern beurteilten Arbeit überein.

Wien, Juli 2020

Unterschrift

Contents

1 Introduction **1**

2 Reinforcement Learning Foundations **2**
2.1 Markov Decision Processes 2

3 Deep Q-Learning and the DQN algorithm **4**
3.1 Atari 2600 games 4
3.2 DQN algorithm 4
3.3 DQN within Deadly Triad 6

4 Distributional Reinforcement Learning **9**
4.1 Motivation and theoretical background 9
4.1.1 Policy Evaluation 10
4.1.2 Control 11
4.2 Categorical Distributional Reinforcement Learning 17
4.3 Distributional Reinforcement Learning with Quantile Regression 19

5 Results: Replicating DQN for Enduro **25**

1 Introduction

This bachelor thesis focuses on deep reinforcement learning (DRL) and its recent improvements using a distributional approach. In particular, it discusses DeepMind's DQN algorithm and its application to the Atari 2600 games. Furthermore, it presents DeepMind's approaches to distributional reinforcement learning by stating the mathematical motivation and main results supporting the distributional perspective to RL. In addition, two algorithms that implement deep distributional reinforcement learning, C51 and Quantile Regression-DQN, are discussed.

In the practical part of this thesis, a framework that implements the Deep Q-Network algorithm compatible to the Arcade Learning Environment is developed in Julia and the challenging Atari Enduro game is learned.

2 Reinforcement Learning Foundations

Reinforcement learning is a subcategory of machine learning and a form of learning which action to choose in a specific situation in order to maximize a numerical reward signal. In RL terminology, it is learning to map states to actions to maximize future reward. This framework is generally described by agent-environment interactions [8].

2.1 Markov Decision Processes

Finite Markov decision processes (MDPs) are a formalization of sequential decision making and can be used to mathematically formulate the reinforcement learning problem.

Definition 2.1.1. (Finite Markov decision process). A finite Markov decision process is a 5-tupel $(\mathcal{X}, \mathcal{A}, R, P, \gamma)$, where $\mathcal{X}$ is a finite state space and $\mathcal{A}$ a finite action space. The transition kernel $P : \mathcal{X} \times \mathcal{A} \to \mathscr{P}(\mathbb{R} \times \mathcal{X})$ defines a joint distribution over immediate reward and next state given the current state-action pair. $\gamma \in [0, 1)$ is the discount factor and the reward is denoted by $R : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$.

A state $x \in \mathcal{X}$ is said to fulfil the Markov property if

$$P(x_n | x_{n-1}, a_{n-1}, \ldots, x_0, a_0) = P(x_n | x_{n-1}, a_{n-1})$$

for any sequences $(x_i)_{i=0,\ldots,n} \in \mathcal{X}^{n+1}$ and $(a_i)_{i=0,\ldots,n-1} \in \mathcal{A}^n$.

The Markov property is assumed in the general reinforcement learning setting.

Definition 2.1.2. (Policy). A policy $\pi : \mathcal{X} \to \mathscr{P}(\mathcal{A})$ defines a probability distribution over the action space $\mathcal{A}$ given the current state.

The return of a policy π, starting in an initial state $x \in \mathcal{X}$ and initially taking action $a \in \mathcal{A}$, is defined as the random variable given the by the sum of discounted rewards following policy π after taking action a in x, i.e.,

$$Z^\pi(x, a) = \sum_{t=0}^{\infty} \gamma^t R(x_t, a_t), \text{ where}$$
$$x_0 = x, \quad a_0 = a, x_t \sim P(\cdot | x_{t-1}, a_{t-1}), \quad a_t \sim \pi(\cdot | x_t).$$

The random return $Z^\pi(x)$ starting in state x and thereafter following policy π is defined analogously.

The goal of almost all reinforcement learning algorithms is estimating value functions, i.e. functions that map states or state-action pairs to a numerical value that describes how favourable it is to be in a certain state given a policy π. In general reinforcement learning, also called value-based

RL, this is measured by the expectation of the return from starting in a state or state-action pair and following policy π thereafter:

$$Q^\pi(x,a) = \mathbb{E}\left[Z^\pi(x,a)\right],$$
$$V^\pi(x) = \mathbb{E}\left[Z^\pi(x)\right].$$

Formally, the RL problem is divided into two settings: the policy evaluation and the control problem.

Policy Evaluation and Control In the policy evaluation setting, the state-value function $V^\pi(x)$ or the action-value function $Q^\pi(x,a)$ for a given policy π is approximated.

The reason for computing these functions is to compare them to value functions of other policies and thereby to find better policies.

Definition 2.1.3. Given two policies π, π' on the same Markov decision process $(\mathcal{X}, \mathcal{A}, R, P, \gamma)$. Then,

$$\pi \geq \pi' \quad \Longleftrightarrow \quad V^\pi(x) \geq V^{\pi'}(x) \qquad \forall x \in \mathcal{X}$$

and we say π is a better policy than π'.

In the control setting, the goal is to approximate an optimal policy, i.e. a policy that maximizes reward. Such a policy is a fixed point of the Bellman optimality equation

$$Q(x,a) = \mathbb{E}[R(x,a)] + \gamma \mathbb{E}\left[\max_{A' \in \mathcal{A}} Q(X', A')\right], \tag{1}$$

which is the motivation for Q-learning, the reinforcement learning algorithm that is the fundamental algorithm of the extensions presented in this thesis.

Q-Learning It directly approximates the optimal action-value function Q^* by the update rule

$$Q(x,a) = Q(x,a) + \alpha[R(x,a) + \gamma \max_{a'} Q(x',a') - Q(x,a)].$$

Q-learning was first published in 1989 by Watkins [11] and is an off-policy temporal-difference learning algorithm. For $|\mathcal{X}| < \infty$ and $|\mathcal{A}| < \infty$, it can be shown that Q-learning converges to the optimal action values with probability one, if all actions are repeatedly sampled in all states and the step size parameters satisfy the usual stochastic approximation conditions. A detailed proof can be found in [10].

An algorithm is said to behave off-policy if it evaluates or improves a policy different from that used to make decisions. On-policy methods evaluate or improve the policy that is used to generate data.

3 Deep Q-Learning and the DQN algorithm

Approximating the value function with a deep neural network is generally referred to as deep reinforcement learning. A way to apply this function approximation to Q-learning is DeepMind's Deep Q-network algorithm first proposed in 2013 [5] and improved in 2015 [6]. This algorithm was designed to master 57 Atari 2600 games – a set of games that include a varied range of challenging tasks – with the same architecture and hyperparameters. The goal of the reinforcement learning agent is to maximize cumulative future reward in the form of the game score. The DQN algorithm is in the course of this thesis implemented and tested on the Enduro Atari game. Also, it is the fundament for the two distributional algorithms that are presented in Section 4. Thus, its architecture is discussed in detail in the following.

3.1 Atari 2600 games

Because the following algorithms are designed to master the Atari 2600 games, we give a brief summary of the environment: A state only consists of images of the screen of a game. To fulfil the Markov property every visited frame of the game should be contained in the state. Since this is computationally unfeasible, this is approximated by using the last k frames as an input to the approximator. In DQN $k = 4$ is used.

Also, the frames are preprocessed by rescaling them from 210×160 to 84×84 pixel frames. To reduce flickering, the maximum value for every pixel of two consecutive frames is taken. The preprocessing of an observed image will be denoted by a function ϕ. The rewards in every game are set to -1 for a negative signal and +1 for every positive signal. The number of available actions for the tested Atari 2600 games are game-specific and vary from 4 to 18.

3.2 DQN algorithm

Due to the large state space of the game environment, it is necessary to approximate the Q function. Since the agent is intended to learn from observing images of the screen of a game and convolutional neural networks have performed very well on tasks like image and video recognition, convolutional neural networks were used to approximate Q. We refer to the neural net approximator by its weights θ. To make learning and evaluation more efficient, the parametrization Q_θ of Q is a map from states to $\mathbb{R}^{|\mathcal{A}|}$, such that the neural net output for a given state $x \in \mathcal{X}$ is an array $((Q_\theta(x))_{a \in \mathcal{A}}$ with the corresponding approximated state-action values. Thus, it requires only a single forward pass through the network to compute Q values for all actions in a given state. The input layer of the neural net takes an $84 \times 84 \times 4$ array, i.e. one state representation. The first hidden layer convolves 32 filters

of 8×8 with stride 4 and applies a ReLU (rectified linear unit) activation function, followed by a layer that convolves 64 filters of 4×4 with stride 2 and a subsequent application of a ReLU. The third and final convolutional layer convolves 64 filters of 3×3 with stride 1 followed by a ReLU. Then, a fully connected layer of size with 512 rectifier units is applied and finally a fully connected layer of size $|\mathcal{A}|$ is used.

Additionally to the function approximation, there are three important modification to classical online Q-learning leading to DQN, which are necessary as RL combined with non-linear function approximation is known to be unstable (see also Section 3.3). The following problems arise:

1. Observations of the agent are highly correlated.
2. Small updates to Q_θ can significantly change the policy and thus the distribution of observations.
3. Action values and target values are correlated.

The DQN algorithm proposes the use of a target network $\hat{Q}$. Every $C := 10^4$ steps the weights of the active Q-network Q_θ are copied to $\hat{Q}_{\theta^-}$. The target network is used to compute the target but only the active network is updated. Thus, this modification reduces the effect of the latter point mentioned above.

To reduce correlations in observations for updates, the transitions used for an update are uniformly sampled from a replay buffer that saves the last $M = 10^6$ transitions of the agent.

The Q-networks weights are updated using the Huber loss

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

To minimize the loss, RMSProp with minibatches of size 32 is used. Combining the discussed elements of the Deep Q-learning algorithm, it can now be given in procedural form (see Algorithm 1).

Algorithm 1: Deep Q-learning [6]

Initialize replay memory D to capacity M
Initialize action-value function Q with random weights θ
Initialize target action-value function $\hat{Q}$ with weights $\theta^- = \theta$
foreach *episode* **do**
    Initialize $s_1 := \{x_1\}$ and $\phi(s_1)$
    repeat
        With probability ε choose a random action a_t
        otherwise select $a_t := \text{argmax}_a\, Q(\phi(s_t), a; \theta)$
        Execute action a_t, observe r_t and image x_{t+1}
        Set $s_{t+1} := (s_t, x_{t+1})$ and $\phi_{t+1} := \phi(s_{t+1})$
        Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D
        if $t \equiv 0 \mod K$ **then**
            Sample random minibatch of $(\phi_j, a_j, r_j, \phi_{j+1})$ from D
            $$y_j = \begin{cases} r_j & \text{if } s_{j+1} \text{ terminal,} \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^-) & \text{otherwise} \end{cases}$$
            Perform a RMSProp step on $\mathcal{L}_1(y_j - Q(\phi_j, a_j; \theta))$ w.r.t. θ
        Every C steps reset $\hat{Q} := Q$
    until s_{t+1} *is terminal*

Applying the deep Q-network algorithm, an agent was trained on a total of 200 Million frames of game experience, on 57 Atari 2600 games. DQN performed better than a professional human games tester in 24 games (see [3, Table 1]). A report on all game scores can be found in [6].

3.3 DQN within Deadly Triad

Q-learning is an off-policy temporal difference method. Thus, it combines bootstrapping and off-policy learning. Due to the high-dimensional state space of the Atari games environment, the Q-function must generalize across states and therefore can only be approximated, which leads to DQN combining all the three methods, which Sutton and Barto refer to as the deadly triad [8]. The combination of all three methods possibly lead to unstable and non-convergent algorithms in theory. In addition, there is little knowledge, how non-linear function approximation interacts within the deadly triad and if, in case of divergence, the deadly triad or the non-linear deep neural network is to blame. This fact arises the question why DQN performs so well on the Atari games. While there are in general no theoretical proofs for the convergence of algorithms that find themselves in the deadly triad, DQN is able to master many of the Atari 2600 games better than humans do. To examine the influence of the deadly triad in the DQN setting, empirical

examinations [9] attempted to show how the components of the deadly triad influence DQN's learning behaviour. The components of the deadly triad were examined using the following hypotheses:

Bootstrapping The influence of bootstrapping can be modulated using multi-step returns. If bootstrapping is applied after a single step, as in Q-learning, the contraction in the (linear or tabular) learning update is proportional to the discount factor $\gamma \in [0, 1)$. For n-step return updates the bootstrapped value is weighted by γ^n Increasing the number of steps before bootstrapping, intuitively, results in slower divergence. This hypothesis is supported with linear function approximation since the TD-operator is a contraction, but this does not hold true for non-linear parametrizations.

Function approximation The generalization of the function approximation can be modulated by the capacity of the neural net approximator. While the tabular TD-algorithms converge, it is hypothesized that a larger neural net generalizes less between states and thus behaves more like the tabular case. This hypothesis was not supported by the empirical experiments.

Off-policy learning To make updates "more off-policy" one can change the distribution by which sampling from the experience replay is conducted. If transitions are not uniformly sampled but prioritized [7], it can lead to more off-policy updates. The heavier the prioritization, the more off-policy updates are.

By varying these components, insights into the effects of them on the learning behaviour of the algorithm are made. Variants of DQN are tested and compared in the Atari 2600 environment and evaluated by multiple statistics. The most significant being the maximal absolute action-value estimate. Because rewards are clipped to $[-1, 1]$ and $\gamma = 0.99$, the estimates theoretically are bounded by $\frac{1}{1-\gamma} = 100$. If values $|q| > 100$ occur, this is referred to as soft divergence. A poor control performance was shown empirically to correlate with soft divergence. The following hypotheses were empirically shown in [9].

Hypotheses 1 *Unbounded divergence is uncommon when combining Q-learning and conventional deep reinforcement learning function spaces.*

Hypotheses 2 *There is less divergence when bootstrapping on separate networks (target network).*

Hypotheses 3 *Longer multi-step returns will diverge less easily.*

Hypotheses 4 *Stronger prioritization of updates will diverge more easily.*

The perspective that deep neural networks and reinforcement learning are usually unstable runs counter to experiments, where no setting resulted in unbounded value estimates, i.e. NaNs.

Assuming that multi-step returns lead to less bootstrapping, hypotheses 3 was empirically shown and thus, the influence of the bootstrapping component of the deadly triad on DQN was made visible. A similar conclusion can be drawn for hypotheses 4 and off-policy learning.

4 Distributional Reinforcement Learning

4.1 Motivation and theoretical background

In the general value-based reinforcement learning setting the agent is intended to learn action selection in order to maximize the *expected* return, i.e. the action-value function (3) or the state-value function (4)

$$Q(x,a) = \mathbb{E}[Z(x,a)] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(x_t, a_t)\right], \tag{3}$$

$$V(x) = \mathbb{E}[Z(x)] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(x_t)\right]. \tag{4}$$

Instead of restricting the algorithm to learn the expectation of the return, one can argue why not learn the whole distribution. Bellman's equation for value-based reinforcement learning can also be translated into the distributional setting as

$$Z(x,a) \overset{D}{=} R(x,a) + \gamma Z(X', A'). \tag{5}$$

The distributional Bellman equation states that the distribution of the return $Z(x,a)$ is characterized by the random reward $R(x,a)$, the transition $(x,a) \to (X', A')$ and the return of the next state-action pair $Z(X', A')$. For the theoretical results, these three quantities are assumed to be independent.

Let

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

denote the space of action-value distributions with finite moments. The set of probability distributions with finite p-th moment is denoted by $\mathscr{P}_p(\mathbb{R})$.

Definition 4.1.1. (Distributional Bellman operator). The distributional Bellman operator $\mathcal{T}^\pi : \mathcal{Z} \to \mathcal{Z}$ is defined using the distributional Bellman equation (5) by

$$\mathcal{T}^\pi(Z(x,a)) \overset{D}{:=} R(x,a) + \gamma Z(X', A'), \text{ where}$$
$$X' \sim P(\cdot|x,a), \quad A' \sim \pi(\cdot, X').$$

For further analysis of this operator the Wasserstein distance will be an important tool.

Definition 4.1.2. ((Supremum) p-Wasserstein-distance). The p-Wasserstein distance d_p for $1 \leq p < \infty$ and $F, G \in \mathscr{P}_p(\mathbb{R})$ is defined as

$$d_p(F,G) := \left(\int_0^1 |F^{-1}(u) - G^{-1}(u)|^p du\right)^{1/p}. \tag{6}$$

On $\mathcal{Z}$, the supremum p-Wasserstein distance $\bar{d}_p$ for $Z_1, Z_2 \in \mathcal{Z}$ is given by

$$\bar{d}_p(Z_1, Z_2) := \sup_{x,a} d_p(Z_1(x,a), Z_2(x,a)). \tag{7}$$

The p-Wasserstein distance is a metric on $\mathscr{P}_p(\mathbb{R})$. The supremum p-Wasserstein distance was introduced in [1] and is a metric on $\mathcal{Z}$. Given two random variables U, V with cumulative distribution functions F_U, F_V, we will write $d_p(U,V) := d_p(F_U, F_V)$ whenever convenient.

Lemma 4.1.3. *Let U, V, A be random variables with distributions in $\mathscr{P}_p(\mathbb{R})$ with A independent of U, V and $a \in \mathbb{R}$. The metric d_p has the following properties:*

$$d_p(aU, aV) \leq |a| d_p(U,V), \tag{8}$$

$$d_p(A+U, A+V) \leq d_p(U,V), \tag{9}$$

$$d_p(AU, AV) \leq \|A\|_p d_p(U,V). \tag{10}$$

The following analysis of the distributional Bellman operator is split into the policy evaluation and the control setting.

4.1.1 Policy Evaluation

Analogous to the policy evaluation setting of value-based reinforcement learning, the goal of policy evaluation is the approximation of the return distribution Z^π for a given policy π.

Lemma 4.1.4. *([1], Lemma 3). $\mathcal{T}^\pi : \mathcal{Z} \to \mathcal{Z}$ is a γ-contraction in $\bar{d}_p$.*

Proof. Let $Z_1, Z_2 \in \mathcal{Z}$. By definition,

$$\bar{d}_p(T^\pi Z_1, T^\pi Z_2) = \sup_{x,a} d_p(T^\pi Z_1(x,a), T^\pi(Z_2(x,a))).$$

Applying Lemma 4.1.3, we obtain

$$\begin{aligned} d_p(T^\pi Z_1(x,a), T^\pi(Z_2(x,a))) &= d_p(R(x,a) + \gamma Z_1(X', A'), R(x,a) + \gamma Z_2(X', A')) \\ &\overset{(9)}{\leq} d_p(\gamma Z_1(X', A'), \gamma Z_2(X', A')) \\ &\overset{(8)}{\leq} \gamma d_p(Z_1(X', A'), Z_2(X', A')) \\ &\leq \gamma \sup_{A', X'} d_p(Z_1(X', A'), Z_2(X', A')). \end{aligned}$$

Thus,

$$\begin{aligned} \bar{d}_p(T^\pi Z_1, T^\pi Z_2) &= \sup_{x,a} d_p(T^\pi Z_1(x,a), T^\pi(Z_2(x,a))) \\ &\leq \sup_{x,a} \gamma \sup_{A', X'} d_p(Z_1(X', A'), Z_2(X', A')) \\ &= \gamma \sup_{A', X'} d_p(Z_1(X', A'), Z_2(X', A')) \\ &= \gamma \bar{d}_p(Z_1, Z_2). \end{aligned}$$

□

By Banach's fixed point theorem, $\mathcal{T}^\pi$ has a unique fixed point Z and for $Z_{k+1} = \mathcal{T}^\pi Z_k$ it holds that $\lim_{k\to\infty} Z_k \to Z$ on the complete metric space $(\mathcal{Z}, \bar{d}_p)$. This fixed point is $Z = Z^\pi$.

Not all distributional metrics are equal. For example, $\mathcal{T}^\pi$ is not a contraction in total variance distance, Kullback-Leibler divergence and the Kolmogorov distance.

4.1.2 Control

In the control setting of distributional reinforcement learning, the goal of the agent is to learn an optimal return distribution. Following the analysis of [1] several definitions are needed first.

Definition 4.1.5. (Optimal return distribution). An optimal return distribution is the return distribution of an optimal policy. The set of optimal return distributions is $\mathcal{Z}^* := \{Z^{\pi^*} : \pi^* \in \Pi^*\}$.

Definition 4.1.6. (Greedy policies for Z). A greedy policy π for $Z \in \mathcal{Z}$ maximizes the expectation of Z. The set of greedy policies for Z is

$$\mathcal{G}_Z := \left\{ \pi : \quad \sum_a \pi(a|x)\mathbb{E}[Z(x,a)] = \max_{a' \in \mathcal{A}} \mathbb{E}[Z(x,a')] \quad \forall x \in \mathcal{X} \right\}.$$

Definition 4.1.7. (Distributional Bellman optimality operator). Any operator $\mathcal{T} : \mathcal{Z} \to \mathcal{Z}$ for which

$$\exists \pi \in \mathcal{G}_Z : \quad \mathcal{T}Z = \mathcal{T}^\pi Z$$

holds is called distributional Bellman optimality operator. Particularly, this is an operator $\mathcal{T}$ such that

$$\mathcal{T}(Z(x,a)) \overset{D}{:=} R(x,a) + \gamma Z(X', \pi_Z(X')), \text{ where}$$
$$X' \sim P(\cdot|x,a), \quad \pi_Z(X') = \operatorname*{argmax}_{a'} \mathbb{E}[Z(X',a')].$$

A distributional Bellman optimality operator $\mathcal{T}$ is not a contraction in $\bar{d}_p$ and does not necessarily have a fixed point $Z^* = \mathcal{T}Z^*$ as shown in [1] in Proposition 1 and 2.

Convergence in the control setting can only be shown pointwise and not to the set $\mathcal{Z}^*$ but to the larger set of nonstationary optimal return distributions $\mathcal{Z}^{**}$. A nonstationary optimal value distribution $Z^{**} \in \mathcal{Z}^{**}$ is the value distribution corresponding to a sequence of optimal policies.

Theorem 4.1.8. *([1], Theorem 1). Let $\mathcal{X}$ be measurable and $|\mathcal{A}| < \infty$. Then*

$$\lim_{k\to\infty} \inf_{Z^{**}\in\mathcal{Z}^{**}} d_p(Z_k(x,a), Z^{**}(x,a)) = 0 \quad \forall x \in \mathcal{X}, a \in \mathcal{A}.$$

If $|\mathcal{X}| < \infty$, then Z_k converges uniformly to $\mathcal{Z}^{**}$. If there is a total ordering $\prec$ on Π^* such that $\mathcal{T}Z^* = \mathcal{T}^\pi Z^*$ holds for any $Z^* \in \mathcal{Z}^*$ with $\pi \in \mathcal{G}_{Z^*}, \pi \prec \pi'$ for all $\pi' \in \mathcal{G}_{Z^*} \setminus \{\pi\}$, then $\mathcal{T}$ has a unique fixed point $Z^* \in \mathcal{Z}^*$.

Before we prove Theorem 4.1.8 we make several definitions and state additional lemmata needed for the proof.

Lemma 4.1.9. *([1], Lemma 4). Let $Z_1, Z_2 \in \mathcal{Z}$. Then*

$$\|\mathbb{E}\mathcal{T}Z_1 - \mathbb{E}\mathcal{T}Z_2\|_\infty \le \gamma \|Z_1 - Z_2\|_\infty.$$

Proof. The distributional Bellman optimality operator is denoted by $\mathcal{T}_D$ and for the usual Bellman optimality operator $\mathcal{T}_E$ is used. Therefore by linearity of the expectation we have

$$\begin{aligned}\|\mathbb{E}\mathcal{T}_D Z_1 - \mathbb{E}\mathcal{T}_D Z_2\| &= \|\mathcal{T}_E \mathbb{E}Z_1 - \mathcal{T}_E \mathbb{E}Z_2\| \\ &\le \gamma \|Z_1 - Z_2\|_\infty.\end{aligned}$$

which completes the proof. □

First, we proof the statement assuming a unique optimal policy π^* and later deduce the general case from this result. Note that from the uniqueness of π^* it follows that it is deterministic and we write $\pi^*(x) =: a^*$ for the optimal action in state x.

Let $Z_k := \mathcal{T}Z_{k-1}$ with $Z_0 \in \mathcal{Z}$, $B := 2\sup_{Z\in\mathcal{Z}} \|Z\|_\infty < \infty$, $\varepsilon_k := \gamma^k B$ and define $\mathcal{X}_k \subseteq \mathcal{X}$ at time step k by

$$\mathcal{X}_k := \left\{ x \in \mathcal{X} : \quad Q^*(x, \pi^*(x)) - \max_{a\neq\pi^*(x)} Q^*(x,a) > 2\varepsilon_k \right\}. \tag{11}$$

Using Lemma 4.1.9 and $Q_k := \mathbb{E}Z_k$ we obtain

$$\begin{aligned}|Q_k(x,a) - Q^*(x,a)| &\le \gamma^k |Q_0(x,a) - Q^*(x,a)| \\ &\le \gamma^k (|Q_0(x,a)| + |Q^*(x,a)|) \\ &\le \gamma^k 2 \sup_{Z\in\mathcal{Z}} \|Z\|_\infty = \gamma^k B.\end{aligned}$$

For $x \in \mathcal{X}, a \in \mathcal{A}$ it follows that $Q^*(x,a^*) - Q_k(x,a^*) \le \varepsilon_k$ and $Q_k(x,a) - Q^*(x,a) \le \varepsilon_k$. Adding the inequalities, it follows that

$$Q_k(x,a^*) - Q_k(x,a) \ge Q^*(x,a^*) - Q^*(x,a) - 2\varepsilon_k. \tag{12}$$

For $x \in \mathcal{X}_k$, we deduce from the definition (11) and equation (12) that $Q_k(x,a^*) > Q_k(x,a')$ holds for all $a' \neq a^*$. Thus, the greedy policy w.r.t. Q_k corresponds to the optimal policy π^* for the states in $\mathcal{X}_k$.

Lemma 4.1.10. *For $x \in \mathcal{X}$ it holds that*

$$\exists k \in \mathbb{N} \quad \forall k' \geq k : \quad x \in \mathcal{X}_{k'}.$$

In particular, $\operatorname{argmax}_{a \in \mathcal{A}} Q_k(x,a) = \pi^*(x)$.

Proof. Define

$$\Delta(x) := Q^*(x,a^*) - \max_{a \in \mathcal{A}, a \neq a^*} Q^*(x,a).$$

Since π^* is deterministic and $|\mathcal{A}| \leq \infty$, it follows that $\Delta(x) > 0$. By definition it follows that $\Delta(x) \leq 2 \sup_{Z \in \mathcal{Z}} \|Z\|_\infty = B$. Since $\gamma^k \to 0$, it holds that

$$\exists k' : \quad \forall k \geq k' : \quad \varepsilon_k = \gamma^k B < \gamma^{k'} B < \frac{\Delta(x)}{2}$$

and $x \in \mathcal{X}_{k'}$ for all $k' \geq k$. □

Therefore, it follows that $\mathcal{X}_k \nearrow \mathcal{X}$. We will refer to these states, where the greedy policy chooses the optimal policy, as "solved". But not only need these states to be solved, also most of their successors and most of the successors of those. This is formalized as follows: Let $\delta > 0$ be fixed and $\mathcal{X}_{k,0} := \mathcal{X}_k$. For $i > 0$ define

$$\mathcal{X}_{k,i} := \left\{ x \in \mathcal{X}_k : \quad P(\mathcal{X}_{k-1,i-1} | x, \pi^*(x)) \geq 1 - \delta \right\}. \tag{13}$$

Lemma 4.1.11. *For any $i \in \mathbb{N}$ and any $x \in \mathcal{X}$, there exists k such that*

$$\forall k' \geq k : \quad x \in \mathcal{X}_{k',i}.$$

Proof. We show that $\mathcal{X}_{k,i} \nearrow \mathcal{X}$ per induction on i. For $i = 0$ this follows from Lemma 4.1.10. Let P be any probability measure on $\mathcal{X}$. Assume $\mathcal{X}_{k,i} \nearrow \mathcal{X}$ and thus $P(\mathcal{X}_{k,i}) \to P(\mathcal{X}) = 1$. Let $x \in \mathcal{X}$, then there exists $k : x \in \mathcal{X}_k$. It follows that

$$P(\mathcal{X}_{k,i} | x, \pi^*(x)) \to P(\mathcal{X} | x, \pi^*(x)) = 1.$$

By equation (13), there exists $k' \geq k$ such that $x \in \mathcal{X}_{k',i+1}$ and x remains in $\mathcal{X}_{k,i+1}$ for all $k \geq k'$. Thus, $\mathcal{X}_{k,i+1} \nearrow \mathcal{X}$. □

For the proof of Theorem 4.1.8 one additional lemma is needed:

Lemma 4.1.12. *Let $(A_j)_{j \geq 1}$ be a set of random variables describing a partition of Ω, i.e. $A_i(\omega) \in \{0,1\}$ and for any $\omega \in \Omega$ there is exactly one A_i, such that $A_i(\omega) = 1$. Let U, V be two random variables. Then*

$$d_p(U,V) \leq \sum_i d_p(A_i U, A_i V).$$

Proof of Theorem 4.1.8. We write $W_k(x) := Z_k(x, \pi_k(x))$ and similarly $W^*(x) := Z^*(x, \pi^*(x))$ and use the notation $\mathcal{T}W_k(x) = W_{k+1}(x) = \mathcal{T}Z_k(x, \pi_k(x))$ for the application of the Bellman optimality operator. Also, we define the indicator functions $S_i^k(x) := \mathbb{I}[x \in \mathcal{X}_{k,i}]$ and $\bar{S}_i^k(x) := 1 - S_i^k(x)$. Fix $i > 0$ and $x \in \mathcal{X}_{k,i}$. Define the transition operator $P^\pi : \mathcal{Z} \to \mathcal{Z}$ as $P^\pi Z(x,a) \overset{D}{:=} Z(X', A')$ with $X' \sim P(\cdot|x,a)$ and $A' \sim \pi(\cdot|X')$. By using Lemma 4.1.12 we separate the transition from x into two terms, one for solved next states and one for unsolved next states X':

$$P^{\pi_k} W_k(x) = S_i^k(X') W_k(X') + \bar{S}_i^k(X') W_k(X'),$$
$$P^{\pi_k} W^*(x) = S_i^k(X') W^*(X') + \bar{S}_i^k(X') W^*(X').$$

Note that from Lemma 4.1.10 for $x \in \mathcal{X}_k$ it follows that $\pi_k(x) = \pi^*(x)$. Then,

$$\begin{aligned}
d_p(W_{k+1}(x), W^*(x)) &= d_p(\mathcal{T}W_k(x), \mathcal{T}W^*(x)) \\
&= d_p(R(x, \pi_k(x)) + \gamma P^{\pi_k} Z_k(x), R(x, \pi^*(x)) + \gamma P^{\pi^*} Z^*(x)) \\
&\overset{(8),(9)}{\leq} \gamma d_p(P^{\pi_k} Z_k(x), P^{\pi^*} Z^*(x))) \\
&= \gamma d_p(P^{\pi_k} W_k(x), P^{\pi^*} W^*(x)) \\
&\overset{4.1.12}{\leq} \gamma d_p(S_i^k(X') W_k(X'), S_i^k(X') W^*(X')) \\
&+ \gamma d_p(\bar{S}_i^k(X') W_k(X'), \bar{S}_i^k(X') W^*(X')) \qquad (14)
\end{aligned}$$

holds. Let

$$\delta_i := \mathbb{P}(\{X' \notin \mathcal{X}_{k,i}\}) = \mathbb{E}(\{\bar{S}_i^k(X')\}) = \|\bar{S}_i^k(X')\|_p.$$

Then,

$$\begin{aligned}
d_p(\bar{S}_i^k(X') W_k(X'), \bar{S}_i^k(X') W^*(X')) &\leq \sup_{x'} d_p(\bar{S}_i^k(X') W_k(x'), \bar{S}_i^k(X') W^*(x')) \\
&\overset{(10)}{\leq} \|\bar{S}_i^k(X')\|_p \sup_{x'} d_p(W_k(x'), W^*(x')) \\
&= \delta_i \sup_{x'} d_p(W_k(x'), W^*(x')) \\
&\leq \delta_i B.
\end{aligned}$$

For $x \in \mathcal{X}_{i+1,k+1}$ it holds that $1 - \delta_i = P(X' \in \mathcal{X}_{i,k} | x, \pi^*(x)) \geq 1 - \delta$ and it follows that $\delta_i \leq \delta$.

This yields

$$d_p(W_{k+1}(x), W^*(x)) \leq \gamma d_p(S_i^k(X') W_k(X'), S_i^k(X') W^*(X')) + \gamma \delta B. \quad (15)$$

Let $x \in \mathcal{X}_{k+i,i}$. By induction on $i > 0$, we show that

$$d_p(W_{k+i}(x), W^*(x)) \leq \gamma^i d_p(S_0^k(X'') W_k(X''), S_0^k(X'') W^*(X'')) + \delta B \sum_{j=1}^{i} \gamma^j$$

for a random state X'' i steps forward. For $i = 1$ it follows with (15) and the partition $\{S_0^k, \bar{S}_0^k\}$ that

$$d_p(W_{k+1}(x), W^*(x)) \leq \gamma d_p(S_0^k(X')W_k(X'), S_0^k(X')W^*(X')) + \gamma\delta B.$$

Using the induction hypotheses (IH) we conclude that

$$\begin{aligned} d_p(W_{k+i}(x), W^*(x)) &\overset{(14)}{\leq} \gamma d_p(S_0^{k+i-1}(X')W_{k+i-1}(X'), S_0^{k+i-1}(X')W^*(X')) \\ &+ \gamma d_p(\bar{S}_0^{k+i-1}(X')W_{k+i-1}(X'), \bar{S}_0^{k+i-1}(X')W^*(X')) \\ &\leq \gamma d_p(W_{k+i-1}(X'), W^*(X')) + \gamma\delta B \\ &\overset{(IH)}{\leq} \gamma^i d_p(W_k(X''), W^*(X'')) + \delta B \sum_{j=1}^{i} \gamma^j \\ &\leq \gamma^i d_p(W_k(X''), W^*(X'')) + \frac{\delta B}{1-\gamma}. \end{aligned}$$

Finally, for any $x \in \mathcal{X}$ and for any $\varepsilon > 0$ there are $\delta > 0$ small enough and i and k sufficiently large such that the inequality $d_p(W_k(x), W^*(x)) < \varepsilon$ holds. By one additional application of $\mathcal{T}$ the result extends to $Z_k(x, a)$. To expand the proof to the general case, where there are multiple optimal policies, we define the sets

$$\mathcal{X}_{k,i} := \{x \in \mathcal{X}_k : \quad \forall \pi^* \in \Pi^* : \quad \mathbb{E}P(\mathcal{X}_{k-1,i-1}|x, a^*) \geq 1 - \delta\}.$$

Since $|\mathcal{A}| < \infty$, Lemma 4.1.11 holds for the expanded definition of $\mathcal{X}_{k,i}$. Let $x \in \mathcal{X}_{k,i}$ and consider a sequence of greedy policies $\pi_k, \pi_{k-1}, \ldots$ selected by successive applications of $\mathcal{T}$ and define

$$\mathcal{T}^{\bar{\pi}_k} := \mathcal{T}^{\pi_k}\mathcal{T}^{\pi_{k-1}} \ldots \mathcal{T}^{\pi_{k-i+1}}$$

and write

$$Z_{k+1} = \mathcal{T}^{\bar{\pi}_k} Z_{k-i+1}.$$

Let $Z^* \in \mathcal{Z}^*$, then

$$\inf_{Z^{**} \in \mathcal{Z}^{**}} d_p(\mathcal{T}^{\bar{\pi}_k} Z^*(x, a), Z^{**}(x, a)) \leq \frac{\delta B}{1-\gamma}, \tag{16}$$

because Z^* corresponds to an optimal policy π^* and $\bar{\pi}_k$ is optimal along most of the trajectories from (x, a).

Finally, for this Z^* we obtain

$$\begin{aligned} \inf_{Z^* * \in \mathcal{Z}^{**}} d_p(Z_{k+1}(x, a), Z^{**}(x, a)) &\leq d_p(Z_{k+1}(x, a), \mathcal{T}^{\bar{\pi}_k} Z^*(x, a)) \\ &+ \inf_{Z^{**}} d_p(\mathcal{T}^{\bar{\pi}_k} Z^*(x, a), Z^{**}(x, a)) \\ &\leq d_p(\mathcal{T}^{\bar{\pi}_k} Z_{k-i+1}(x, a), \mathcal{T}^{\bar{\pi}_k} Z^*(x, a)) + \frac{\delta B}{1-\gamma} \\ &\leq \gamma^i B + \frac{2\delta B}{1-\gamma}, \end{aligned}$$

using the same argument as before for the newly defined $\mathcal{X}_{k,i}$. Thus we have

$$\inf_{Z^* * \in \mathcal{Z}^{**}} d_p(Z_k(x,a), Z^{**}(x,a)) \to 0. \tag{17}$$

We get uniform convergence for finite $\mathcal{X}$ since there is a fixed k after which $\mathcal{X} = \mathcal{X}_k$.

Now, consider the special case where there is a total ordering $\prec$ on Π^* such that for any $Z^* \in \mathcal{Z}^*$ the equation $\mathcal{T}Z^* = \mathcal{T}^\pi Z^*$ holds with $\pi \in \mathcal{G}_{Z^*}$ and $\pi \prec \pi'$ for all $\pi' \in \mathcal{G}_{Z^*} \setminus \{\pi\}$. The greedy policies for Z^*, i.e. the policies which maximize the expectation of Z^*, are exactly Π^*. Denote by π^* the minimal policy in Π^* w.r.t. $\prec$. Then $\mathcal{T} = \mathcal{T}^{\pi^*}$, which has a unique fixed point as seen in Section 4.1.1.

□

Theorem 4.1.8 shows differences for the distributional approach in the control setting. Lemma 4.1.9 shows that the expectation of Z_k converges to Q^* in $\|\cdot\|_\infty$, but its distribution only converges to the set of non-stationary optimal return distributions.

These theoretical results can now be used to design algorithms for the reinforcement learning framework. Particularly, two algorithms are presented, which were tested within the Atari 2600 environment. The first using a categorical approach for approximation, the other one applies quantile regression.

4.2 Categorical Distributional Reinforcement Learning

Along the theoretical results presented in Section 4.1, [6] presented an algorithm to approximate distributional learning called categorical algorithm.

The return distribution is modelled using a discrete distribution parametrized by the number of atoms $N \in \mathbb{N}$ and the boundaries $V_{\text{MIN}}, V_{\text{MAX}} \in \mathbb{R}$ with the support

$$\mathcal{S} := \{z_i = V_{\text{MIN}} + i\Delta z : 0 \leq i < N\}, \quad \Delta z := \frac{V_{\text{MAX}} - V_{\text{MIN}}}{N-1}.$$

Every atom z_i of the support stands for the value of a possibly observed return where V_{MIN} is the smallest observable return and V_{MAX} the largest. Given a state x and an action a, every atom z_i is assigned a probability, the probability of receiving reward z_i given the state-action-pair (x, a). These probabilities are given by a parametric model $\theta : \mathcal{X} \times \mathcal{A} \to \mathbb{R}^N$ that defines the distribution

$$Z_\theta(x, a) = z_i \text{ with probability } p_i(x, a) := \frac{e^{\theta_i(x,a)}}{\sum_j e^{\theta_j(x,a)}}$$

of the return. On the one hand, using a discrete distribution has the advantage of being highly expressive and computationally friendly. On the other hand, the fixed support of the return distribution poses a problem when applying the Bellman update $\mathcal{T}Z_\theta$. Often, the update exceeds the boundaries of the support of the discretized distribution.

Following the theoretical results of section 4.1, it would seem obvious to minimize the Wasserstein distance while training. This metric as defined in 4.1.2 would be robust to disjoint supports of distribution functions but toy problems from [1] showed that minimizing the Wasserstein loss with stochastic gradient descent performs poorly.

Instead, the Kullback–Leibler (KL) divergence, another distance between probability distributions, is used for the loss function. Note, that $\mathcal{T}^\pi$ is not a contraction in KL divergence. Thus, the use of this metric is not supported by convergence results.

Definition 4.2.1. (Kullback-Leibler divergence). For two discrete probability distributions P and Q defined on the same probability space Ω, the KL divergence from Q to P is defined as

$$\begin{aligned} D_{\text{KL}}(P\|Q) &:= \sum_{\omega \in \Omega} P(\omega) \log\left(\frac{P(\omega)}{Q(\omega)}\right) \\ &= -\sum_{\omega \in \Omega} P(\omega) \log\left(\frac{Q(\omega)}{P(\omega)}\right). \end{aligned} \tag{18}$$

As this distance relies on non-disjoint supports, there is a projection onto $\mathcal{S}$ applied to the Bellman update before computing the loss.

The Bellman update is approximated in the following way.

Definition 4.2.2. (Projected sample Bellman update $\Phi\hat{\mathcal{T}}Z_\theta(x,a)$). For a sample transition (x,a,r,x') the sample Bellman update is given by

$$\hat{\mathcal{T}}z_j = r + \gamma z_j \quad \forall z_j \in \mathcal{S}. \tag{19}$$

The i^{th} component of the projected update $\Phi\hat{\mathcal{T}}Z_\theta(x,a))$ is given by

$$(\Phi\hat{\mathcal{T}}Z_\theta(x,a)))_i = \sum_{j=0}^{N-1}\left[1 - \frac{|[\hat{\mathcal{T}}z_j]_{V_{MIN}}^{V_{MAX}} - z_i|}{\Delta z}\right]_0^1 p_j(x', \pi(x')), \tag{20}$$

where $[]_a^b$ bounds its argument in the range $[a,b]$.

The categorical algorithm can now be given as Algorithm 2.

Algorithm 2: Categorical Algorithm [1]

Input: A transition x_t, a_t, r_t, x_{t+1} and γ

$Q(x_{t+1}, a) \coloneqq \sum_i z_i p_i(x_{t+1}, a)$

$a^* \coloneqq \operatorname{argmax}_a Q(x_{t+1}, a)$

$m_i \coloneqq 0, \quad i \in 0, \dots, N-1$

for $j \in 0, \dots, N-1$ **do**

  $\hat{\mathcal{T}}z_j \coloneqq [r_t + \gamma z_j]_{V_{MIN}}^{V_{MAX}}$

  $b_j \coloneqq (\hat{\mathcal{T}}z_j - V_{MIN})/\Delta z$

  $l \coloneqq \lfloor b_j \rfloor, u \coloneqq \lceil b_j \rceil$

  $m_l \coloneqq m_l + p_j(x_{t+1}, a^*)(u - b_j)$

  $m_u \coloneqq m_u + p_j(x_{t+1}, a^*)(b_j - l)$

end

Output: $-\sum_i m_i \log p_i(x_t, a_t)$

C51 – Categorical DRL for Atari Combining the DQN Algorithm 1 with the Categorical Algorithm 2, C51 originates. For this additional parameters are needed: the number of atoms $N = 51$ and the boundaries of the support of the parametrized distribution $V_{\text{MAX}} = -V_{\text{MIN}} = 10$. Additionally, the neural net approximator needs to be adapted. It is now approximating return distributions and thus the output layer returns for a given state x the approximated return distributions for every action, i.e. an array of size $|\mathcal{A}| \times N$.

Approximating the whole distribution instead of just the expectation performed very well on the Atari games. C51 beat DQN in 50 out of 57 Atari games and performed better than a professional human games tester in 40 games (see [3, Table 1]) after being trained on 200 million frames.

4.3 Distributional Reinforcement Learning with Quantile Regression

Even though the categorical approach given in Algorithm 2 performed very well in numerical examples, it is not supported by the theoretical findings of Section 4.1. Thus, further designing of algorithms for approximated DRL was conducted by DeepMind and in [3] they proposed an algorithm which minimizes the Wasserstein loss applying quantile regression.

Their approach includes learning a distribution with fixed probabilities $q_1, \dots, q_n$ and variable locations $z_1, \dots, z_n$. Specifically, uniform probabilities $q_i := 1/n,\ i = 1, \dots, n$ were chosen and locations were learnt. This can be referred to as estimating quantiles of a distribution. A p-quantile of a probability distribution P on $(\mathbb{R}, \mathcal{B}(\mathbb{R}))$ is a real number x_p such that $P((-\infty, p]) \geq x_p$ and $P([p, \infty)) \geq 1 - x_p$. The values of the corresponding discrete cumulative distribution function are denoted by $\tau_1, \dots, \tau_n$ with $\tau_i = i/n$ and $\tau_0 = 0$.

Analogous to $\mathcal{Z}$, the space of action-value distributions, we define $\mathcal{Z}_Q$ as the set of quantile distributions for fixed n.

Definition 4.3.1. (Quantile Distribution). Let $\theta : \mathcal{X} \times \mathcal{A} \to \mathbb{R}^n$ be a parametric model. A quantile distribution $Z_\theta \in \mathcal{Z}_Q$,

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

is a mapping from $\mathcal{X} \times \mathcal{A}$ to a uniform probability distribution with support $\{\theta_i(x, a)\}$.

Comparing the approach of Section 4.2 to learning a quantile distribution three main benefits are stated in [3]:

1. There are no restrictions by predefined bounds on the support of the distribution.

2. Thus, there is no more need for a projection of the Bellman update onto the predefined support of the distribution. Together, these points mean that there is no more domain knowledge for the bounds of the support needed.

3. By applying quantile regression, this parametrization allows to minimize the Wasserstein loss, without suffering form biased gradients as discussed in the following.

Definition 4.3.2. (Quantile Projection). The projection of an arbitrary return distribution $Z \in \mathcal{Z}$ onto $\mathcal{Z}_Q$ w.r.t. the 1-Wasserstein distance is defined as

$$\Pi_{W_1} Z := \operatorname*{argmin}_{Z_\theta \in \mathcal{Z}_Q} W_1(Z, Z_\theta).$$

For a distribution with bounded first moment Y and a uniform distribution U with support $\{\theta_1, \dots, \theta_n\}$ the 1-Wasserstein metric, partitioning the preimage of U, is given by

$$W_1(Y,U) = \sum_{i=1}^{n} \int_{\tau_{i-1}}^{\tau_i} |F_Y^{-1}(\omega) - \theta_i| d\omega. \tag{21}$$

Quantile Regression To guarantee unbiased gradients for the approximation of the distribution, quantile regression [4] is applied. This method is widely known and is a method for unbiased stochastic approximation of the quantile function.

Definition 4.3.3. (Quantile regression loss $\mathcal{L}_{QR}^{\tau}$). For a given quantile $\tau \in [0,1]$ and a distribution Z the quantile regression loss $\mathcal{L}_{QR}^{\tau}$ is given by

$$\begin{aligned} &\mathcal{L}_{QR}^{\tau}(\theta) := \mathbb{E}_{\hat{Z}\sim Z}[\rho_\tau(\hat{Z} - \theta_i)], \text{ where} \\ &\rho_\tau(u) = u(\tau - \delta_{\{u<0\}}). \end{aligned}$$

The value of the quantile function $F_Z^{-1}(\tau)$ can be defined as the minimizer of $\mathcal{L}_{QR}^{\tau}$. This loss gives unbiased sample gradients and can therefore be minimized using stochastic gradient descent.

Lemma 4.3.4. *For $\tau, \tau' \in [0,1]$ with $\tau < \tau'$ and cumulative distribution function F, the set of $\theta \in \mathbb{R}$ minimizing*

$$\int_{\tau}^{\tau'} |F^{-1}(\omega) - \theta| d\omega$$

is given by

$$\left\{ \theta \in \mathbb{R} : \quad F(\theta) = \frac{\tau + \tau'}{2} \right\}.$$

In particular, if F^{-1} is the inverse CDF, $F^{-1}((\tau+\tau')/2)$ is always a valid minimizer and if F^{-1} is continuous at $(\tau+\tau')/2$, it is the unique minimizer.

Proof. Let $\omega \in [0,1]$. By the convexity of $|\cdot|$, the function $G : \theta \mapsto |F^{-1}(\omega) - \theta|$ is convex. Thus, the subgradient (in θ at θ_0)

$$[a,b] := \left[\lim_{\theta \to \theta_0^-} \frac{G(\theta) - G(\theta_0)}{\theta - \theta_0}, \lim_{\theta \to \theta_0^+} \frac{G(\theta) - G(\theta_0)}{\theta - \theta_0} \right],$$

is well-defined and given by

$$\theta \mapsto \begin{cases} -1 & \text{if } \theta < F^{-1}(\omega), \\ [-1,1] & \text{if } \theta = F^{-1}(\omega), \\ 1 & \text{if } \theta > F^{-1}(\omega). \end{cases}$$

From the triangle inequality, it follows that the function $\theta \mapsto \int_\tau^{\tau'} |F^{-1}(\omega) - \theta| d\omega$ is convex and its subgradient is given by

$$\theta \mapsto \int_\tau^{F(\theta)} 1 d\omega + \int_{F(\theta)}^{\tau'} -1 d\omega = -\tau' - \tau + 2F(\theta).$$

Setting it to zero yields $\tau' + \tau = 2F(\theta)$ and thus $\theta = F^{-1}((\tau' + \tau)/2)$ is a minimizer. For F continuous at $(\tau' + \tau)/2$ the minimizer is unique. □

These quantile midpoints will be denoted by

$$\hat{\tau}_i := \frac{\tau_{i-1} + \tau_i}{2} \quad \text{for } i = 1, \dots, n. \tag{22}$$

With Lemma 4.3.4 it follows that the set $\{\theta_1, \dots, \theta_n\}$ which minimizes $W_1(Y, U)$ is given by $\theta_i = F_Y^{-1}(\hat{\tau}_i)$.

Combining equation (21) and Lemma 4.3.4, it follows that if $\{\theta_1, \dots, \theta_n\}$ minimize $W_1(Z, Z_\theta)$, they also minimize

$$\sum_{i=1}^{n} \mathbb{E}_{\hat{Z} \sim Z}[\rho_{\hat{\tau}_i}(\hat{Z} - \theta_i)]. \tag{23}$$

The main result of [3], which supports combining DRL with quantile regression, is the following.

Proposition 4.3.5. *([3], Proposition 2). Let Π_{W_1} be the quantile projection. For return distributions $Z_1, Z_2 \in \mathcal{Z}$ and a Markov decision process $(\mathcal{X}, \mathcal{A}, R, P, \gamma)$ with countable action and state spaces $\mathcal{A}$ and $\mathcal{X}$, it holds that*

$$\bar{d}_\infty(\Pi_{W_1} \mathcal{T}^\pi Z_1, \Pi_{W_1} \mathcal{T}^\pi Z_2) \leq \gamma \bar{d}_\infty(Z_1, Z_2). \tag{24}$$

Before we proof Proposition 4.3.5, we need two supporting results:

Lemma 4.3.6. *Let $(\mathcal{X}, \mathcal{A}, R, P, \gamma)$ be a MDP with countable action and state spaces $\mathcal{A}$ and $\mathcal{X}$. Let Y, Z be return distributions such that each state-action distribution $Z(x, a), Y(x, a)$ is given by a discrete Dirac delta function. Consider the special case where rewards are equal to zero, $\gamma = 1$ and $\tau \in [0, 1]$. Denote by Π_τ the projection operator that maps a probability distribution onto a Dirac delta distribution located at its τ^{th}-quantile. Then, the inequality*

$$\bar{d}_\infty(\Pi_\tau \mathcal{T}^\pi Z, \Pi_\tau \mathcal{T}^\pi Y) \leq \bar{d}_\infty(Z, Y)$$

holds.

Proof. Let $Z(x, a) = \delta_\theta(x, a)$ and $Y(x, a) = \delta_\psi(x, a)$ for all $(x, a) \in \mathcal{X} \times \mathcal{A}$ and $\psi, \theta : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$. Let (x', a') be an arbitrary state-action pair and let $\{(x_i, a_i)_{i \in I}\}$ be the finite or countable set of state-action pairs that can be reached from (x', a') in a single transition with probability $p_i > 0$.

Then, we can write the application of the distributional Bellman operator on Z and Y as

$$\mathcal{T}^\pi Z(x', a') = \sum_{i \in I} p_i \delta_{\theta(x_i, a_i)}, \tag{25}$$

$$\mathcal{T}^\pi Y(x', a') = \sum_{i \in I} p_i \delta_{\psi(x_i, a_i)}. \tag{26}$$

Let $u \in I$ be such that θ_u is equal to the τ-th quantile of $\mathcal{T}^\pi Z(x', a')$ and $v \in I$ such that ψ_v is equal to the τ-th quantile of $\mathcal{T}^\pi Y(x', a')$.

By definition of Π_τ, it follows that

$$d_\infty(\Pi_\tau \mathcal{T}^\pi Z(x', a'), \Pi_\tau \mathcal{T}^\pi Y(x', a')) = |\theta_u - \psi_v|.$$

Assume that

$$\forall i \in I: \quad |\theta_u - \psi_v| > |\theta_i - \psi_i|. \tag{27}$$

We show that (27) leads to a contradiction:

W.l.o.g. let $\theta_u \leq \psi_v$. Define the partitions

$$\begin{aligned} I_{\leq \theta_u} &:= \{i \in I: \quad \theta_i \leq \theta_u\}, \\ I_{> \theta_u} &:= \{i \in I: \quad \theta_i > \theta_u\}, \\ I_{< \psi_v} &:= \{i \in I: \quad \psi_i \leq \psi_v\}, \\ I_{\geq \psi_v} &:= \{i \in I: \quad \psi_i > \psi_v\} \end{aligned}$$

of I that lead to the disjoint unions $I = I_{\leq \theta_u} \cup I_{> \theta_u} = I_{< \psi_v} \cup I_{\geq \psi_v}$. By the assumption in (27), it follows that $I_{\leq \theta_u} \cap I_{\geq \psi_v} = \varnothing$ and therefore $I_{\leq \theta_u} \subseteq I_{< \psi_v}$. Since θ_u is the τ-th quantile of $\mathcal{T}^\pi Z(x', a')$, we have that

$$\sum_{i \in I_{< \psi_v}} p_i \geq \sum_{i \in I_{\leq \theta_u}} p_i \geq \tau.$$

Thus, by definition of the τ-quantile it follows that the τ-th quantile of $\mathcal{T}^\pi Y(x', a')$ is less then ψ_v. Therefore the assumption is wrong and we conclude that

$$d_\infty(\Pi_\tau \mathcal{T}^\pi Z(x', a'), \Pi_\tau \mathcal{T}^\pi Y(x', a')) \leq \bar{d}_\infty(Z, Y).$$

Since, (x', a') was arbitrary, the result

$$\bar{d}_\infty(\Pi_\tau \mathcal{T}^\pi Z, \Pi_\tau \mathcal{T}^\pi Y) \leq \bar{d}_\infty(Z, Y)$$

follows for the metric $\bar{d}_\infty$. □

Lemma 4.3.7. *For two probability distributions ν_1, ν_2 over $\mathbb{R}$ and the Wasserstein projection operator Π_{W_1} that projects distributions onto support of size n, it holds that*

$$d_\infty(\Pi_{W_1} \nu_1, \Pi_{W_1} \nu_2) = \max_{i=1,\dots,n} \left| F_{\nu_1}^{-1}\left(\frac{2i-1}{2n}\right) - F_{\nu_2}^{-1}\left(\frac{2i-1}{2n}\right) \right|.$$

Proof. By Lemma 4.3.4 $\Pi_{W_1}\nu_k = \sum_{i=1}^n \frac{1}{n}\delta_{F_{\nu_k}^{-1}(\frac{2i-1}{2n})}$ for $k = 1, 2$ and it follows by definition that

$$\begin{aligned} d_\infty(\Pi_{W_1}\nu_1, \Pi_{W_1}\nu_2) &= d_\infty\left(\sum_{i=1}^n \frac{1}{n}\delta_{F_{\nu_1}^{-1}(\frac{2i-1}{2n})}, \sum_{i=1}^n \frac{1}{n}\delta_{F_{\nu_2}^{-1}(\frac{2i-1}{2n})}\right) \\ &= \max_{i=1,\dots,n}\left|F_{\nu_1}^{-1}\left(\frac{2i-1}{2n}\right) - F_{\nu_2}^{-1}\left(\frac{2i-1}{2n}\right)\right| \end{aligned}$$

holds. □

Proof of Proposition 4.3.5. First, several simplifications can be applied to the proposition: For a given state-action pair the instantaneous reward is assumed to be deterministic.

Furthermore, from the γ-contraction property of $\mathcal{T}^\pi$ in $\bar{d}_\infty$, it is sufficient to prove the claim for $\gamma = 1$.

Because the Wasserstein metrics are invariant under translation of the support of distributions, it is enough to consider the case $r(x, a) = 0$ for all $(x, a) \in \mathcal{X} \times \mathcal{A}$.

We write $Z(x, a) = \sum_{k=1}^N \frac{1}{N}\delta_{\theta_k(x,a)}$ and $Y(x, a) = \sum_{k=1}^N \frac{1}{N}\delta_{\psi_k(x,a)}$ for $\theta, \psi : \mathcal{X} \times \mathcal{A} \to \mathbb{R}^n$. Let $\{(x_i, a_i)_{i\in I}\}$ be the finite or countable set of state-action pairs that can be reached from an arbitrary state-action pair (x', a') in a single transition with probability $p_i > 0$.

Now, we construct a new MDP in which all distributions are given by single Dirac delta distributions to apply Lemma 4.3.6. Starting from (x', a') we define new states, actions, transitions and a policy $\tilde{\pi}$. The states that are reachable in one transition are now given by $((\tilde{x}_i^j, \tilde{a}_i^j)_{i\in I})_{j=1}^N$ and the probability of reaching $(\tilde{x}_i^j, \tilde{a}_i^j)$ is p_i/n.

For this new MDP, define the return distributions

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

Thus, we obtain

$$\begin{aligned} \bar{d}_\infty(\Pi_{W_1}\mathcal{T}^\pi\tilde{Z}, \Pi_{W_1}\mathcal{T}^\pi\tilde{Y}) &\overset{4.3.5}{\leq} \bar{d}_\infty(\tilde{Z}, \tilde{Y}) \\ &= \sup_{\substack{i\in I \\ j=1,\dots,N}} |\theta_j(x_i, a_i) - \psi_j(x_i, a_i)| \\ &\overset{4.3.7}{=} \sup_{i\in I} d_\infty(Z(x_i, a_i), Y(x_i, a_i)). \end{aligned} \tag{28}$$

By construction, $\mathcal{T}^{\tilde{\pi}}\tilde{Z}(x', a')$ has the same distribution as $\mathcal{T}^\pi Z(x', a')$ and the same holds for Y. By definition of the Wasserstein distance it follows that

$$d_\infty(\Pi_{W_1}\mathcal{T}^{\tilde{\pi}}\tilde{Z}(x', a'), \Pi_{W_1}\mathcal{T}^{\tilde{\pi}}\tilde{Y}(x', a')) = d_\infty(\Pi_{W_1}\mathcal{T}^\pi Z(x', a'), \Pi_{W_1}\mathcal{T}^\pi Y(x', a')).$$

By substituting this equality into inequality (28) and since (x', a') were arbitrary, it follows that

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

□

With Proposition 4.3.5 it follows that the combination of the quantile projection and the Bellman operator is a contraction. Thus, by Banach's fixed point theorem it follows that $\Pi_{W_1}\mathcal{T}^\pi$ has a unique fixed point $\hat{Z}^\pi$ and the sequence $Z_{k+1} := \Pi_{W_1}\mathcal{T}^\pi Z_k, Z_0 \in \mathcal{Z}$ converges to $\hat{Z}^\pi$. Since $\bar{d}_p \leq \bar{d}_\infty$, convergence results hold for $p \in [1, \infty]$.

QR-DQN The goal of the algorithm is to approximate the return distribution with a parametrized quantile distribution over the set of quantile midpoints defined in equation (22).

Combining Lemma 4.3.4 with the distributional Bellman equation yields the quantile regression temporal difference learning update

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

where Z_θ is a quantile distribution as defined in 4.3.1 and $\theta_i(x)$ is the estimated value of $F^{-1}_{Z^\pi(x)}(\hat{\tau}_i)$.

This update can now be used to adapt the DQN algorithm to learn quantile distributions of returns. The DQN architecture, as discussed in Section 3, is mostly maintained but three changes were made. Similar to C51, the output of the neural network approximator is of size $|\mathcal{A}| \times N$ where N is the number of quantile targets for the approximated distribution.

A so-called quantile Huber loss,

$$\rho^\kappa_\tau(u) = |\tau - \delta_{\{u<0\}}|\mathcal{L}_\kappa(u), \tag{29}$$

where $\mathcal{L}_\kappa(u)$ is the Huber Loss, replaces the Huber loss function, which was used by DQN. The optimizer is changed from RMSProp to ADAM.

The changes to the DQN algorithm can now be given in Algorithm 3.

Algorithm 3: Quantile Regression Q-Learning [3]

Input: A transition x, a, r, x' and γ

1 $Q(x', a') := \sum_i q_i \theta_i(x', a')$

2 $a^* := \text{argmax}_{a'} Q(x, a')$

3 $\mathcal{T}\theta_j := r + \gamma\theta_j(x', a^*)$

Output: $\sum_{i=1}^N \mathbb{E}[\rho^\kappa_{\hat{\tau}_i}(\mathcal{T}\theta_j - \theta_i(x,a))]$

5 Results: Replicating DQN for Enduro

Finally, the results of replicating DQN for one Atari game are reported despite the huge computational requirements. A framework that implements the Deep Q-Network algorithm compatible to the Arcade Learning Environment [2] has been developed in Julia and the challenging Atari Enduro game is learned.

Enduro The Atari 2600 game Enduro consists of racing the national Enduro, a long-distance endurance race. The object of the race is to pass other cars. In particular, the RL agent receives a reward of +1 for passing one car. To receive a negative reward when being overtaken by other cars, the agent needs to pass cars before. Then, the agent receives a reward of −1 per car. In general, rewards are given extremely sparse in Enduro. Also, Enduro involves different game modes and changes its visibility: There is a night mode, where only tail lights of the cars are visible. Also, the weather conditions change from sunny, to an icy patch on the road to a patch of fog. Thus, Enduro is an extremely challenging reinforcement learning environment.

Training Details For replicating the DQN algorithm, all the parameters and specifications reported in [6] were used. Unfortunately, since training is computationally very complex and memory-intense, the training duration was shortened and the replay buffer was reduced in size. In the first experiments a buffer with size $M = 10^5$ was used and later M was set to $4{\cdot}10^5$ while DQN originally uses a buffer with 1 Million transitions. Training was conducted on 10 million frames of game experience while DQN used a training time of 50 million frames. In addition, DeepMind's frame skipping technique was applied, such that the agent only sees every 4^{th} frame and the chosen action is repeatedly executed on the next three frames. This makes learning more efficient.

The learning behaviour of four different settings are compared: Three different optimizers, stochastic gradient descent, ADAM and RMSProp, are compared with a buffer of size $4 \cdot 10^5$ and the two different buffer sizes using RMSProp as an optimizer can be compared.

Evaluation Details DeepMind stated for the evaluation of DQN that the agents were frequently tested along the training period on 135 000 evaluation frames and the highest average episode score is reported. An agent is evaluated by reporting its game score acting greedily in one game (i.e., an episode). Using the Arcade Learning Environment for the game simulation, the Enduro game has around 3000 frames or $3000 \cdot 4$ frames, as it is not stated clearly if the skipped frames are counted or not by DeepMind. This leads to roughly 40 or 10 tested games per agent. Since, one game of Enduro

takes quite a long time and the training performance is tested every 50 000 frames, three games per agent are reported and all three performances are given in the scatter plots in Figure 5. Additionally, the average performance in the three games is stated in the line plot.

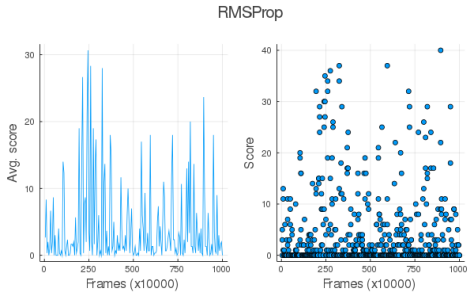

Figure 1: RMSProp with $M = 10^5$

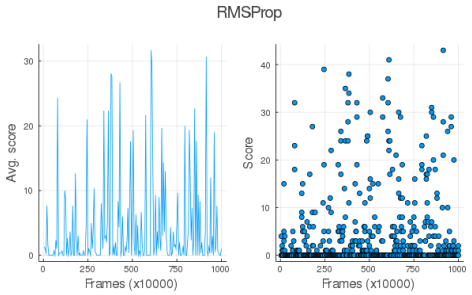

Figure 2: RMSProp with $M = 4 \cdot 10^5$

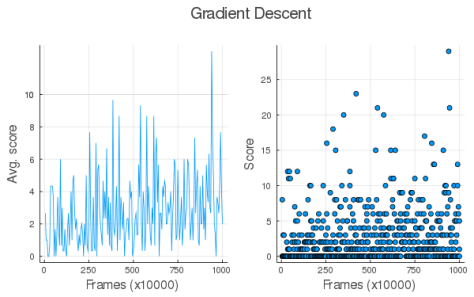

Figure 3: GD with $M = 4 \cdot 10^5$

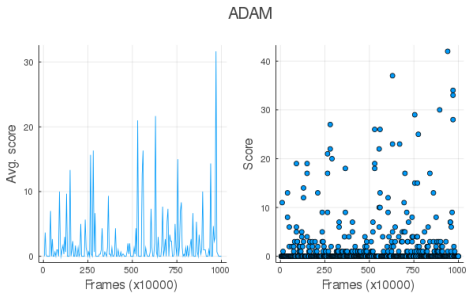

Figure 4: ADAM with $M = 4 \cdot 10^5$

Figure 5: Evaluation of four different DQN settings trained on Enduro for 10 million frames.

Results DeepMind reported as the highest score of DQN in Enduro 301.8 (±24.6) points. For comparison, a random agent receives on average 0 points in one game. The results achieved in the course of this thesis are visualized in Figure 5. The best agent achieved a score of 43 points. It uses RMSProp and the larger replay buffer of size $4 \cdot 10^5$.

By having a closer look at Figure 1, it is visible that the agent relatively early achieves a high score but then reducing its scored points again. We believe that this behaviour is connected to the smaller replay buffer. In general, all optimizers except gradient descent show large variances in their scores and behave like forgetting how to behave in already visited and learned states constantly. This could be connected to the more advanced optimizers that both implement an adaptive learning rate in combination to the small buffer of size $4 \cdot 10^5$.

Even though not directly visible through the reported score, gradient descent seems to implement the most reliable and consistent learning in combination with a smaller buffer as the averaged score given in Figure 3 shows a trend.

References

[1] M. G. Bellemare, W. Dabney, and R. Munos. A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning – Volume 70*, ICML'17, page 449–458. JMLR.org, 2017.

[2] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

[3] W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos. Distributional reinforcement learning with quantile regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, IAAA-18, page 1892–1901, 2017.

[4] R. Koenker, A. Chesher, and M. Jackson. *Quantile Regression*. Econometric Society Monographs. Cambridge University Press, 2005.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*, 2013.

[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. Bellemare, A. Graves, M. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–33, 2015.

[7] T. Schaul, J. Quan, I. Antonoglou, and D. Silver. Prioritized experience replay. In *Proceedings of the International Conference on Learning Representations*, 2016.

[8] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

[9] H. van Hasselt, Y. Doron, F. Strub, M. Hessel, N. Sonnerat, and J. Modayil. Deep reinforcement learning and the deadly triad. *ArXiv*, abs/1812.02648, 2018.

[10] C. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

[11] C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, May 1989.