Ensemble Bootstrapping for Q-Learning

Oren Peer [1] **Chen Tessler** [1] **Nadav Merlis** [1] **Ron Meir** [1]

Abstract

Q-learning (QL), a common reinforcement learning algorithm, suffers from over-estimation bias due to the maximization term in the optimal Bellman operator. This bias may lead to sub-optimal behavior. Double-Q-learning tackles this issue by utilizing two estimators, yet results in an under-estimation bias. Similar to over-estimation in Q-learning, in certain scenarios, the under-estimation bias may degrade performance. In this work, we introduce a new bias-reduced algorithm called Ensemble Bootstrapped Q-Learning (EBQL), a natural extension of Double-Q-learning to ensembles. We analyze our method both theoretically and empirically. Theoretically, we prove that EBQL-like updates yield lower MSE when estimating the maximal mean of a set of independent random variables. Empirically, we show that there exist domains where both over and under-estimation result in sub-optimal performance. Finally, We demonstrate the superior performance of a deep RL variant of EBQL over other deep QL algorithms for a suite of ATARI games.

1. Introduction

In recent years, reinforcement learning (RL) algorithms have impacted a vast range of real-world tasks, such as robotics (Andrychowicz et al., 2020; Kober et al., 2013), power management (Xiong et al., 2018), autonomous control (Bellemare et al., 2020), traffic control (Abdulhai et al., 2003; Wiering, 2000), and more (Mahmud et al., 2018; Luong et al., 2019). These achievements are possible by the ability of RL agents to learn a behavior policy via interaction with the environment while simultaneously maximizing the obtained reward.

A common family of algorithms in RL is Q-learning (Watkins & Dayan, 1992, QL) based algorithms, which focuses on learning the value-function. The value represents the expected, discounted, reward-to-go that the agent will obtain. In particular, such methods learn the optimal policy via an iterative maximizing bootstrapping procedure. A well-known property in QL is that this procedure results in a positive estimation bias. Van Hasselt (2010) and Van Hasselt et al. (2016) argue that over-estimating the real value may negatively impact the performance of the obtained policy. They propose Double Q-learning (DQL) and show that, as opposed to QL, it has a negative estimation bias.

The harmful effects of *any* bias on value-learning are widely known. Notably, the deadly triad (Sutton & Barto, 2018; Van Hasselt et al., 2018) attributes the procedure of bootstrapping with biased values as one of the major causes of divergence when learning value functions. This has been empirically shown in multiple works (Van Hasselt et al., 2016; 2018). Although this phenomenon is well known, most works focus on avoiding positive-estimation instead of minimizing the bias itself (Van Hasselt et al., 2016; Wang et al., 2016; Hessel et al., 2018).

In this work, we tackle the estimation bias in an underestimation setting. We begin, similar to Van Hasselt (2010), by analyzing the estimation bias when observing a set of i.i.d. random variables (RVs). We consider three schemes for estimating the maximal mean of these RVs. *(i)* The *single-estimator*, corresponding to a QL-like estimation scheme. *(ii)* The *double-estimator*, corresponding to DQL, an estimation scheme that splits the samples into two equal-sized sets: one for determining the index of the RV with the maximal mean and the other for estimating the mean of the selected RV. *(iii)* The *ensemble-estimator*, our proposed method, splits the data into K sets. Then, a single set is used for estimating the RV with the maximal-mean whereas all other sets are used for estimating the mean of the selected RV.

We start by showing that the ensemble estimator can be seen as a double estimator that unequally splits the samples (Proposition 1). This enables, for instance, less focus on maximal-index identification and more on mean estimation. We prove that the ensemble estimator under-estimates the maximal mean (Lemma 1). We also show, both theoretically (Proposition 2) and empirically (Fig. 1), that in order to reduce the magnitude of the estimation error, a double estimator with equally-distributed samples is sub-optimal.

[1]Viterbi Faculty of Electrical Engineering, Technion Institute of Technology, Haifa, Israel. Correspondence to: Oren Peer <orenpeer@campus.technion.ac.il>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

Following this theoretical analysis, we extend the ensemble estimator to the multi-step reinforcement learning case. We call this extension Ensemble Bootstrapped Q-Learning (EBQL). We analyze EBQL in a tabular meta chain MDP and on a set of ATARI environments using deep neural networks. In the tabular case, we show that as the ensemble size grows, EBQL minimizes the magnitude of the estimation bias. Moreover, when coupled with deep neural networks, we observe that EBQL obtains superior performance on a set of ATARI domains when compared to Deep Q-Networks (DQN) and Double-DQN (DDQN).

Our contributions are as follows:

- We Analyze the problem of estimating the maximum expectation over independent random variables. We prove that the estimation mean-squared-error (MSE) can be reduced using the Ensemble Estimator. In addition, we show that obtaining the minimal MSE requires utilizing more than two ensemble members.
- Drawing inspiration from the above, we introduce Ensemble Bootstrapped Q-Learning (EBQL) and show that it reduces the bootstrapping estimation bias.
- We show that EBQL is superior to both Q-learning and double Q-learning in both a tabular setting and when coupled with deep neural networks (ATARI).

2. Preliminaries

2.1. Model Free Reinforcement Learning

A reinforcement learning (Sutton & Barto, 2018) agent faces a sequential decision-making problem in some unknown or partially known environment. We focus on environments in which the action space is discrete. The agent interacts with the environment as follows. At each time t, the agent observes the environment state $s_t \in \mathcal{S}$ and selects an action $a_t \in \mathcal{A}$. Then, it receives a scalar reward $r_t \in \mathbb{R}$ and transitions to the next state $s_{t+1} \in S$ according to some (unknown) transition kernel $P(s_{t+1}|s_t, a_t)$. A policy $\pi : \mathcal{S} \to \mathcal{A}$, is a function that determines which action the agent should take at each state and the sampled performance of a policy, a random variable starting from state s, is denoted by

$$R^\pi(s) = \sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a \sim \pi(s_t)\,. \tag{1}$$

where $\gamma \in [0, 1)$ is the discount factor, which determines how myopic the agent should behave. As R^π is a random variable, in RL we are often interested in the expected performance, also known as the policy value,

$$v^\pi(s) = \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t) | s_0 = s \right].$$

The goal of the agent is to learn an optimal policy $\pi^* \in \operatorname{argmax}_\pi v^\pi(s)$. Importantly, there exists a policy π^* such that $v^{\pi^*}(s) = v^*(s)$ for all $s \in \mathcal{S}$ (Sutton & Barto, 2018).

2.2. Q-Learning

One of the most prominent algorithms for learning π^* is Q-learning (Watkins & Dayan, 1992, QL). Q-learning learns the Q-function, which is defined as follows

$$Q^\pi(s, a) = \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t(s_t, a_t) | s_0 = s, a_0 = a \right].$$

Similar to the optimal value function, the optimal Q-function is denoted by $Q^*(s, a) = \max_\pi Q^\pi(s, a),\ \forall (s, a) \in \mathcal{S} \times \mathcal{A}$. Given Q^*, the optimal policy π^* can obtained by using the greedy operator $\pi^*(s) = \operatorname{argmax}_a Q^*(s, a)\ \forall s \in \mathcal{S}$.

QL is an off-policy algorithm that learns Q^* using transition tuples (s, a, r, s'). Here, s' is the state that is observed after performing action a in state s and receiving reward r. Specifically, to do so, QL iteratively applies the optimal Bellman equation (Bellman, 1957):

$$Q(s, a) \leftarrow (1-\alpha)Q(s, a) + \alpha(r_t + \gamma \max_{a'} Q(s', a'))\,. \tag{2}$$

Under appropriate conditions, QL asymptotically converges almost surely to an optimal fixed-point solution (Tsitsiklis, 1994), i.e. $Q_t(s, a) \xrightarrow{t\to\infty} Q^*(s, a)\ \forall s \in \mathcal{S}, a \in \mathcal{A}$.

A known phenomenon in QL is the over-estimation of the Q-function. The Q-function is a random variable, and the optimal Bellman operator, Eq. (2), selects the maximal value over all actions at each state. Such estimation schemes are known to be positively biased (Smith & Winkler, 2006).

2.3. Double Q-Learning

A major issue with QL is that it overestimates the true Q values. Van Hasselt (2010); Van Hasselt et al. (2016) show that, in some tasks, this overestimation can lead to slow convergence and poor performance. To avoid overestimation, Double Q-learning (Van Hasselt, 2010, DQL) was introduced. DQL combines two Q-estimators: Q^A and Q^B, each updated over a unique subset of gathered experience. For any two estimators A and B, the update step of estimator A is defined as (full algorithm in the appendix Algorithm 3):

$$\begin{aligned} \hat{a}^*_A &= \operatorname*{argmax}_{a'} Q^A(s_{t+1}, a') \\ Q^A(s_t, a_t) &\leftarrow (1 - \alpha_t) Q^A(s_t, a_t) \\ &\quad + \alpha_t \left(r_t + \gamma Q^B \left(s_{t+1}, \hat{a}^*_A \right) \right). \end{aligned} \tag{3}$$

Importantly, Van Hasselt (2010) proved that DQL, as opposed to QL, leads to underestimation of the Q-values.

3. Related Work

Q-learning stability. Over the last three decades, QL has inspired a large variety of algorithms and improvements, both in the tabular setting (Strehl et al., 2006; Kearns & Singh, 1999; Ernst et al., 2005; Azar et al., 2011) and in the function approximation-based settings (Schaul et al., 2015; Bellemare et al., 2017; Hessel et al., 2018; Jin et al., 2018; Badia et al., 2020). Many improvements focus on the estimation bias of the learned Q-values, first identified by Thrun & Schwartz (1993), and on minimizing the variance of the target approximation-error (Anschel et al., 2017). Addressing this bias is an ongoing effort in other fields as well, such as economics and statistics (Smith & Winkler, 2006; Thaler, 2012). Some algorithms (Van Hasselt, 2010; Zhang et al., 2017) tackle the QL estimator inherent bias by using double estimators, resulting in a *negative* bias. Recently, Lan et al. (2020) introduces an bias-controlling algorithm called MaxMin Q-Learning, that uses ensemble to control the bias direction and magnitude. For large ensembles, it results with large underestimation, whereas for small ensembles, it results with overestimation.

RL algorithms that use neural-network-based function approximators (deep RL) are known to be susceptible to overestimation and oftentimes suffer from poor performance in challenging environments. Specifically, estimation bias, a member of the *deadly triad* (Van Hasselt et al., 2018), is considered as one of the main reasons for the divergence of QL-based deep-RL algorithms.

In our work, we tackle the estimation bias by reducing the MSE of the next-state Q-values. While EBQL is negatively biased, we show that it better balances the bias-variance of the estimator. We also show that the bias magnitude of EBQL is governed by the ensemble size and reduces as the ensemble size grows.

Ensembles. One of the most exciting applications of ensembles in RL (Osband et al., 2018; Lee et al., 2020) is to improve exploration and data collection. These methods can be seen as a natural extension of Thompson sampling-like methods to deep RL. The focus of this work complements their achievements. While they consider how an ensemble of estimators can improve the learning process, we focus on how to better train the estimators themselves.

4. Estimating the Maximum Expected Value

As mentioned in Section 2, QL and DQL display opposing behaviors of over- and under-estimation of the Q-values. To better understand these issues, and our contribution, we take a step back from RL and statistically analyze the different estimators on i.i.d. samples.

Consider the problem of estimating the *maximal expected value* of m independent random variables $(X_1, \ldots, X_m) \triangleq X$, with means $\mu = (\mu_1, \ldots, \mu_m)$ and standard deviations $\sigma = (\sigma_1, \ldots, \sigma_m)$. Namely, we are interested in estimating

$$\max_a \mathbb{E}[X_a] = \max_a \mu_a \triangleq \mu^* \ . \tag{4}$$

The estimation is based on i.i.d. samples from the same distribution as X. Formally, let $S = \{S_a\}_{a=1}^m$ be a set of samples, where $S_a = \{S_a(n)\}_{n=1}^N$ is a subset of N i.i.d. samples from the same distribution as X_a. Specifically, this implies that $\mathbb{E}[S_a(n)] = \mu_a$ for all $n \in [N_a]$. We further assume all sets are mutually independent.

Denote the empirical means of a set S_a by $\hat{\mu}_a(S_a) = \frac{1}{N_a}\sum_{n\in[N_a]} S_a(n)$. This is an unbiased estimator of μ_a, and given sufficiently many samples, it is reasonable to approximate $\hat{\mu}_a(S_a) \approx \mu_a$. Then, a straightforward method to estimate (4) is to use the *Single-Estimator* (SE):

$$\hat{\mu}^*_{\text{SE}} \triangleq \max_a \hat{\mu}_a(S_a) \approx \max_a \mathbb{E}[\hat{\mu}_a(S_a)] = \mu^* \ .$$

As shown in (Smith & Winkler, 2006) (and rederived in Appendix D for completeness), this estimator is positively biased. This overestimation is believed to negatively impact the performance of SE-based algorithms, such as QL.

To mitigate this effect, (Van Hasselt, 2010) introduced the *Double Estimator* (DE). In DE, the samples of each random variable $a \in [m]$ are split into two disjoint, equal-sized subsets $S_a^{(1)}$ and $S_a^{(2)}$, such that $S_a^{(1)} \cup S_a^{(2)} = S_a$ and $S_a^{(1)} \cap S_a^{(2)} = \emptyset$ for all $a \in [m]$. For brevity, we denote the empirical mean of the samples in $S_a^{(j)}$ by $\hat{\mu}_a^{(j)} \triangleq \hat{\mu}_a(S_a^{(j)})$. Then, DE uses a two-phase estimation process: In the first phase, the **index** of the variable with the maximal expectation is estimated using the empirical means of $S^{(1)}$, $\hat{a}^* = \text{argmax}_a \hat{\mu}_a^{(1)}$. In the second phase, the **mean** of $X_{\hat{a}^*}$ is estimated using $S_{\hat{a}^*}^{(2)}$, $\hat{\mu}^*_{\text{DE}} = \hat{\mu}_{\hat{a}^*}^{(2)}$. The resulting estimator is negatively biased, i.e., $\mathbb{E}[\hat{\mu}^*_{\text{DE}}] \leq \mu^*$ (Van Hasselt, 2010).

4.1. The Ensemble Estimator

In this section, we introduce the *Ensemble Estimator* (EE), a natural generalization of DE to K estimators. We take another step forward and ask:

How can we benefit by using K estimators rather than 2?

In the DE, two sources are affecting the approximation error. One type of error rises when the wrong index $\hat{a}^*$ is selected (inability to identify the maximal index). The second, when the mean is incorrectly estimated. While increasing the number of samples used to identify the index reduces the chance of misidentification, it results with fewer samples for mean estimation. Notably, the DE naïvely allocates the same number of samples to both stages. As we show, the optimal "split" is not necessarily equal, hence, in an attempt to minimize the total MSE, a good estimator should more carefully tune the number of samples allocated to each stage.

Consider, for example, the case of two independent random variables $(X_1, X_2) \sim \big(\mathcal{N}(\mu_1, \sigma^2), \mathcal{N}(\mu_2, \sigma^2)\big)$. When the means μ_1 and μ_2 are dissimilar, i.e., $|\mu_1 - \mu_2|/\sigma \gg 1$, the chance of index misidentification is low, and thus, it is preferable to allocate more samples to the task of mean estimation. On the other extreme, as $|\mu_1 - \mu_2|/\sigma \to 0$, the two distributions are effectively identical, and the task of index identification becomes irrelevant. Thus, most of the samples should be utilized for mean estimation. Interestingly, in both these extremities, it is beneficial to allocate more samples to mean estimation over index identification.

As argued above, minimizing the MSE of the estimator can be achieved by controlling the relative number of samples in each estimation phase. We will now formally define the ensemble estimator and prove that it is equivalent to changing the relative allocation of samples between index-estimation and mean-estimation (Proposition 1). Finally, we demonstrate both theoretically and numerically that EE achieves lower MSE than DE, and by doing so, better balances the two sources of errors.

Ensemble estimation is defined as follows: While the DE divides the samples into two sets, the EE divides the samples of each random variable $a \in [m]$ into K equal-sized disjoint subsets, such that $S_a = \bigcup_{k=1}^{K} S_a^{(k)}$ and $S_a^{(k)} \cap S_a^{(l)} = \emptyset$, $\forall k \neq l \in [K]$. We further denote the empirical mean of the a^{th} component of X, based on the k^{th} set, by $\hat{\mu}_a^{(k)}$. As in double-estimation, EE uses a two-phase procedure to estimate the maximal mean. First, a single arbitrary set $\tilde{k} \in [K]$ is used to estimate the **index** of the maximal expectation, $\hat{a}^* = \operatorname{argmax}_a \hat{\mu}_a^{(\tilde{k})}$. Then, EE utilizes the remaining ensemble members to jointly estimate $\mu_{\hat{a}^*}$, namely,

$$\hat{\mu}^*_{\text{EE}} \triangleq \frac{1}{K-1} \sum_{j \in [K] \setminus \tilde{k}} \hat{\mu}^{(j)}_{\hat{a}^*} \approx \max_a \mathbb{E}\left[\frac{1}{K-1} \sum_{j \in [K] \setminus \tilde{k}} \hat{\mu}_a^{(j)}\right] = \mu^* \,.$$

Lemma 1 shows that by using EE, we still maintain the underestimation nature of DE.

Lemma 1. *Let $\mathcal{M} = \operatorname{argmax}_a \mathbb{E}[X_a]$ be the set of indices where $\mathbb{E}[X_a]$ is maximized. Let $\hat{a}^* \in \operatorname{argmax}_a \hat{\mu}_a^{(\tilde{k})}$ be the estimated maximal-expectation index according to the $\tilde{k}^{th}$ subset. Then $\mathbb{E}[\hat{\mu}^*_{EE}] = \mathbb{E}[\mu_{\hat{a}^*}] \leq \max_a \mathbb{E}[X_a]$. Moreover, the inequality is strict if and only if $P(\hat{a}^* \notin \mathcal{M}) > 0$.*

The proof can be found in Appendix A. In addition, below, we show that changing the size of the ensemble is equivalent to controlling the partition of data between the two estimation phases. As previously mentioned, the split-ratio greatly affects the MSE, and therefore, this property enables the analysis of the effect of the ensemble size K on the MSE.

Proposition 1 (The Proxy Identity)**.** *Let $S^{(1)}, \ldots, S^{(K)}$ be some subsets of the samples of equal sizes N/K and let $\tilde{k} \in [K]$ be an arbitrary index. Also, denote by $\hat{\mu}^*_{EE}$, the EE that uses the $\tilde{k}^{th}$ subset for its index-estimation phase. Finally, let $\hat{\mu}^*_{W\text{-}DE}$ be a DE that uses $S^{(\tilde{k})}$ for its index-selection and all other samples for mean-estimation; i.e., $\hat{a}^* = \operatorname{argmax}_a \hat{\mu}_a^{(\tilde{k})}$ and $\hat{\mu}^*_{W\text{-}DE} = \hat{\mu}_{\hat{a}^*}\left(\bigcup_{j \in [K] \setminus \tilde{k}} S^{(j)}\right)$. Then, $\hat{\mu}^*_{EE} = \hat{\mu}^*_{W\text{-}DE}$.*

We call the weighted version of the DE that uses $1/K$ of its samples to the index estimation W-DE. The proxy identity establishes that W-DE is equivalent to the ensemble estimator. Specifically, let N_1 be the number of samples used in the first phase of W-DE (index estimation) and say that the MSE of the W-DE is minimized at $N_1 = N_1^*$. Then, by the proxy-identity, the optimal ensemble size of the EE is $K \approx N/N_1^*$. Thus, the optimal split-ratio of W-DE serves as a *proxy* to the number of estimators to be used in EE.

To better understand the optimal split-ratio, we analyze the MSE of the EE. To this end, we utilize the proxy identity and instead calculate the MSE of the W-DE with $N_1 \approx N/K$ samples for index-estimation. Let X be a vector with independent component of means $(\mu_1, \ldots, \mu_m)$ and variances $\left(\sigma_1^2, \ldots, \sigma_m^2\right)$. Assume w.l.o.g. that the means are sorted such that $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_m$. Then, the following holds (see derivations in Appendix C.1):

$$\begin{aligned}
\text{bias}(\hat{\mu}^*_{\text{W-DE}}) &= \sum_{a=1}^{m} (\mu_a - \mu_1)\, P(\hat{a}^* = a), \\
\text{var}(\hat{\mu}^*_{\text{W-DE}}) &= \sum_{a=1}^{m} \left(\frac{\sigma_a^2}{N - N_1} + \mu_a^2\right) P(\hat{a}^* = a) \\
&\quad - \left(\sum_{a=1}^{m} \mu_a P(\hat{a}^* = a)\right)^2 .
\end{aligned}$$

As μ_1 is assumed to be largest, the bias is always negative; hence, EE underestimates the maximal mean. Furthermore, we derive a closed form for the MSE:

$$\text{MSE}\left(\hat{\mu}^*_{\text{W-DE}}\right) = \sum_{a=1}^{m} \left(\frac{\sigma_a^2}{N - N_1} + (\mu_1 - \mu_a)^2\right) P\left(\hat{a}^* = a\right). \quad (5)$$

We hypothesize that in most cases, the MSE is minimized for $K > 2$. As (5) is rather cumbersome to analyze, we prove this hypothesis in a simpler case and demonstrate it numerically for more challenging situations.

Proposition 2. *Let $X = (X_1, X_2) \sim \mathcal{N}\left((\mu_1, \mu_2)^T, \sigma^2 I_2\right)$ be a Gaussian random vector such that $\mu_1 \geq \mu_2$ and let $\Delta = \mu_1 - \mu_2$. Also, define the signal to noise ratio as $\text{SNR} = \frac{\Delta}{\sigma/\sqrt{N}}$ and let $\hat{\mu}^*_{W\text{-}DE}$ be a W-DE that uses N_1 samples for index estimation. Then, for any fixed even sample-size $N > 10$ and any N_1^* that minimizes $\text{MSE}(\hat{\mu}^*_{W\text{-}DE})$, it*

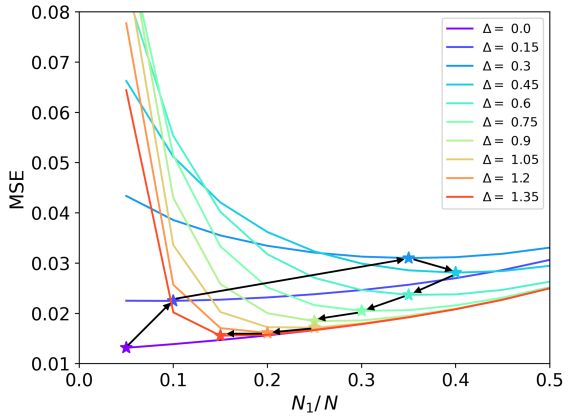


Figure 1: The MSE values as function of the split-ratio for two Gaussians with different mean-gaps $\Delta = \mu_1 - \mu_2$ and $\sigma^2 = 0.25$. Stars mark values in which the minimum MSE is achieved. As expected from Proposition 2, the optimal split-ratio is always smaller than $N/2$. The black arrows show the trend of the optimum as Δ increases. Notably, the optimal split-ratio increases from 0 to 0.4 and then decreases back to 0, with alignment to the SNR claims.

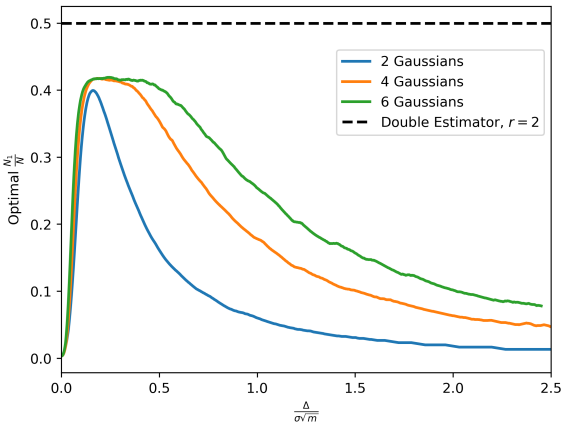


Figure 2: Numerical calculation of the optimal sample split-ratio as function of normalized gap for 2, 4 and 6 Gaussians with means uniformly spread over the interval $\Delta = \mu_{max} - \mu_{min}$.

holds that (1) As SNR $\to \infty$, $N_1^* \to 1$ *(2) As* SNR $\to 0$, $N_1^* \to 1$ *(3) For any* σ *and* Δ*, it holds that* $N_1^* < N/2$.

The proof can be found in Appendix C.2. Note that a similar analysis can be done for sub-Gaussian variables, using standard concentration bounds instead of using Φ. However, in this case, we have to bound $P(\hat{a}^* = a)$ and can only analyze an upper bound of the MSE.

Proposition 2 yields a rather intuitive, yet surprising result. From the first claim, we learn that when X_1 and X_2 are easily distinguishable (high SNR), samples should not be invested in index-estimation, but rather allocated to the mean-estimation. Moreover, the second claim establishes the same conclusion for the case where X_1 and X_2 are indiscernible (low SNR). Then, X_1 and X_2 have near-identical means, and samples are better used for reducing the variance of any of them. The most remarkable claim is the last one; it implies that that the optimal split ratio is *always* smaller than half. In turn, when N is large enough, this implies that there exists $K > 2$ such that the MSE of EE is strictly lower than the one of DE. To further demonstrate our claim for intermediate values of the SNR, Fig. 1 shows the MSE as a function of N_1 for different values of $\Delta = \mu_1 - \mu_2$ (and fixed variance $\sigma^2 = 0.25$).

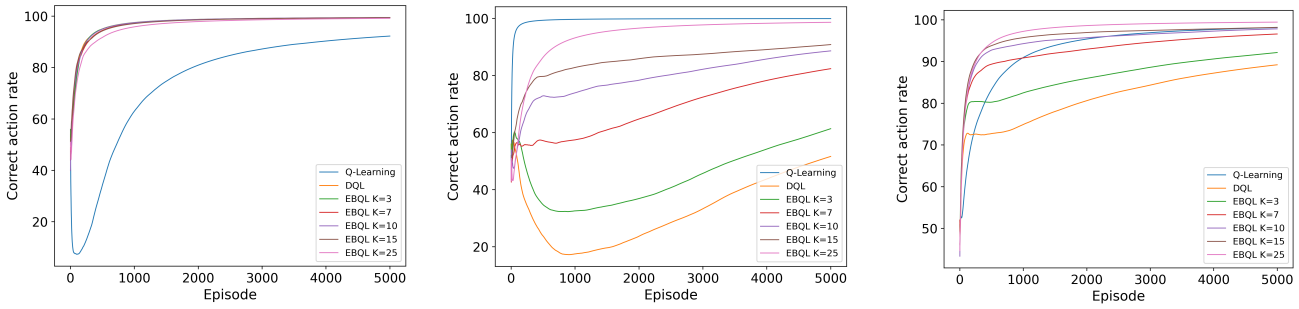
In Fig. 2 we plot the value of N_1 that minimizes the MSE as a function of the normalized-distance of the means, $\frac{\Delta}{\sigma\sqrt{m}}$, for a different number of random variables $m \in \{2, 4, 6\}$. The means are evenly spread on the interval Δ, all with $\sigma^2 = 0.25$. Fig. 2 further demonstrates that using $N_1 < \frac{N}{2}$ can reduce the MSE in a large range of scenarios. By the proxy-identity, this is equivalent to using ensemble sizes of $K > 2$. Therefore, we expect that in practice, EE will outperform DE in terms of the minimal MSE (MMSE).

5. Ensemble Bootstrapped Q-Learning

In Section 4.1, we presented the ensemble estimator and its benefits. Namely, ensembles allow unevenly splitting samples between those used to select the index of the maximal component and those used to approximate its mean. We also showed that allocating more samples to estimate the mean, rather than the index, reduces the MSE in various scenarios.

Although the analysis in Section 4.1 focuses on a statistical framework, a similar operation is performed by the QL agent during training when it bootstraps on next-state values. Specifically, recall that $R^\pi(s, a)$ is the reward-to-go when starting from state s and action a (see Eq. (1)). Then, denoting $(X_1, \ldots, X_m) = (R^\pi(s_{t+1}, a_1), \ldots, R^\pi(s_{t+1}, a_m))$ and $(\mu_1, \ldots, \mu_m) = (Q^\pi(s_{t+1}, a_1), \ldots, Q^\pi(s_{t+1}, a_m))$, the next-state value used by QL is determined by $\max_a \mu_a$. As the real Q-values are unknown, DQL uses two Q estimators $\left(\mu_1^{(1)}, \ldots, \mu_m^{(1)}\right) = Q^A$ and $\left(\mu_1^{(2)}, \ldots, \mu_m^{(2)}\right) = Q^B$ in a way that resembles the DE – the index is selected as $a^* = \text{argmax}_a \mu_a^{(1)}$ and the value as $\mu_{a^*}^{(2)}$.

Our method, Ensemble Bootstrapped Q-Learning (EBQL) presented in Algorithm 1, can be seen as applying the concepts of EE to the QL bootstrapping phase by using an ensemble of K Q-function estimators. Similar to DQL, when updating the k^{th} ensemble member in EBQL, we define the next-state action as $\hat{a}^* = \text{argmax}_a Q^k(s_{t+1}, a)$. However, while DQL utilizes two estimators, in EBQL, the value is



(a) (b) (c)

Figure 3: Correct-action rate from state A^i as function of episode in the Meta Chain MDP. Fig. 3a shows the results of a specific chain-MDP where $\mu_i = -0.2 < 0$, where in Fig. 3b, $\mu_i = 0.2 > 0$. In Fig. 3c, the average of 6 symmetric μ-values is presented. All results are averaged over 50 random seeds.

Algorithm 1 Ensemble Bootstrapped Q-Learning (EBQL)

Parameters: learning-rates: $\{\alpha_t\}_{t\geq 1}$
Initialize: Q-ensemble of size K : $\left\{Q^i\right\}_{i=1}^K$, s_0
for $t = 0, \ldots, T$ **do**
  Choose action $a_t = \text{argmax}_a \left[\sum_{i=1}^K Q^i(s_t, a)\right]$
  $a_t = \text{explore}(a_t)$ //e.g. ϵ-greedy
  $s_{t+1}, r_t \leftarrow \text{env.step}(s_t, a_t)$
  Sample an ensemble member to update: $k_t \sim \mathcal{U}([K])$
  $\hat{a}^* = \text{argmax}_a\, Q^{k_t}(s_{t+1}, a)$
  $Q^{k_t}(s_t, a_t) \leftarrow (1-\alpha_t) Q^{k_t}(s_t, a_t)$
    $+\alpha_t \left(r_t + \gamma Q^{EN\setminus k_t}(s_{t+1}, \hat{a}^*)\right)$
end for
Return $\left\{Q^i\right\}_{i=1}^K$

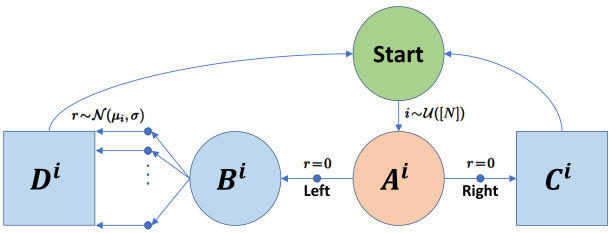


Figure 4: The Meta-Chain-MDP, a set of chain-MDPs differing by the reward at state D. Each episode the agent is randomly initialized in one of the MDPs. The variance $\sigma = 1$ is identical across the chain MDPs.

obtained by averaging over the remaining ensemble members $Q^{EN\setminus k} = \frac{1}{K-1}\sum_{j\in[K]\setminus k} Q^j(s_{t+1}, \hat{a}^*)$. Notice how when $K = 2$ (two ensemble members) we recover the same update scheme as DQL, showing how EBQL is a natural extension of DQL to ensembles. In practice, the update is done using a learning rate α_t, as in DQL (Eq. (3)).

6. Experiments

In this section, we present two main experimental results of EBQL compared to QL and DQL in both a tabular setting and on the ATARI ALE (Bellemare et al., 2013) using the deep RL variants. We provide additional details on the methodology in Appendix E.

6.1. Tabular Experiment - Meta Chain MDP

We begin by analyzing the empirical behavior of the various algorithms on a *Meta Chain MDP* problem (Fig. 4). In this setting, an agent repeatedly interacts with a predetermined set of N chain-MDPs. All chain-MDPs share the same structure, and by the end of the interaction with one MDP, the next MDP is sampled uniformly at random.

For each chain-MDP $i \in [N]$, the agent starts at state A^i and can move either left or right. While both immediate rewards are 0, transitioning to state C^i terminates the episode. However, from state B^i, the agent can perform one of m actions, all transitioning to state D^i and providing the same reward $r(D^i) \sim \mathcal{N}(\mu_i, \sigma^2)$, and the episode is terminated. The set of chain MDPs is a symmetrical mixture of chains with positive and negative means. While a negative μ_i amplifies the sub-optimality of QL due to the over-estimation bootstrapping bias, we observe that positive μ_i presents the sub-optimality of DQL under-estimation bias.

This simple, yet challenging MDP serves as a playground to examine whether an algorithm balances optimism and pessimism. It also provides motivation for the general case, where we do not know if optimism or pessimism is better.

Analysis: We present the empirical results in Fig. 3. In addition to the results on the meta chain MDP, we also present results on the single-chain setting, for both positive and negative reward means μ_i.

As expected, due to its optimistic nature, QL excels in sub-

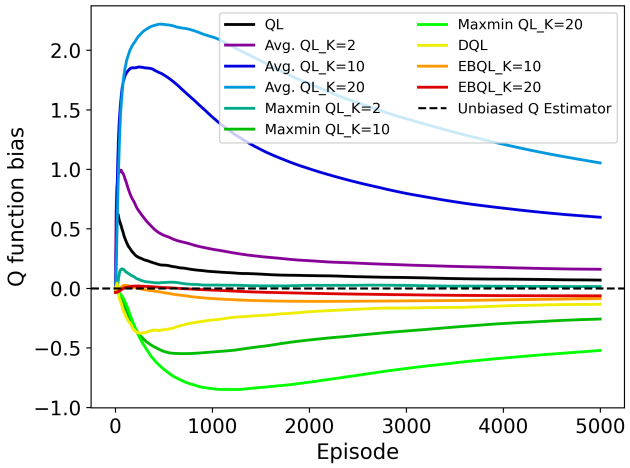


Figure 5: Estimation bias of the optimal action in state A^i as a function of time in the meta-chain MDP.

MDPs where $\mu_i > 0$ (Fig. 3b). The optimistic nature of QL drives the agent towards regions with higher uncertainty (variance). As argued above, we observe that in this case, DQL is sub-optimal due to the pessimistic nature of the estimator. On the other hand, due to its pessimistic nature, and as shown in Van Hasselt (2010), DQL excels when $\mu_i < 0$. We replicate their results in Fig. 3a.

Although QL and DQL are capable of exploiting the nature of certain MDPs, this may result in catastrophic failures. On the other hand, and as shown in Fig. 3c, EBQL excels in the meta chain MDP, a scenario that averages over the sub-MDPs, showing the robustness of the method to a more general MDP. Moreover, we observe that as the size of the ensemble grows, the performance of EBQL improves.

Estimation Bias: In addition to reporting the performance, we empirically compare the bias of the various learned Q-functions on the meta chain MDP. The results are presented in Fig. 5. As expected QL and DQL converge to 0^+ and 0^- respectively. In addition to QL and DQL, we compare EBQL to Average QL Anschel et al. (2017) and MaxMin QL (Lan et al., 2020). Notably, Average QL converges slowly while overestimating the Q function, as expected from a variance-reduced extension of QL. In addition, observe that the bias of MaxMin–QL strongly depends on the ensemble size K and varies from overestimation ($K = 2$) to underestimation ($K > 2$). Hence, for MaxMin–QL to be unbiased, we must tune its ensemble size using some prior knowledge on the environment. In particular, when the environment 'encourages' optimism, small ensembles are advantageous, and vice-versa for pessimism encouraging environments.

Interestingly, the absolute bias of EBQL decreases and the algorithm becomes less pessimistic as the ensemble size K grows. This explains the results of Fig. 3, where increasing the ensemble size greatly improves the performance in chain MDPs with $\mu_i > 0$. To maintain fairness, all Q-tables were set to zero, which can explain the positive bias of EBQL in the initial episodes.

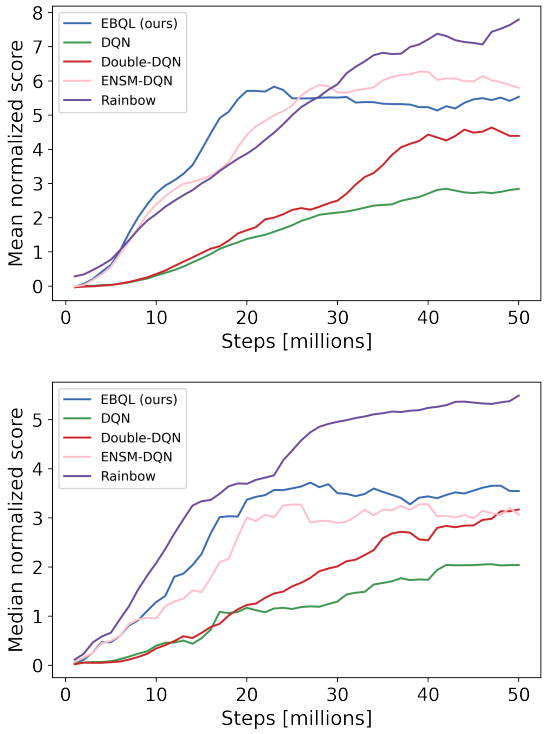


Figure 6: **Normalized scores:** We present both the mean and median normalized scores over the 11 environments (Table 1) with 5 seeds per environment. Curves are smoothed with a moving average over 5 points. In addition to DQN, DDQN and Ensemble DQN (Anschel et al., 2017), we provide a comparison against Rainbow (Hessel et al., 2018). Rainbow represents the state-of-the-art non-distributed Q-learning-based agent.

6.2. Atari

Here, we evaluate EBQL in a high dimensional task – ATARI ALE (Bellemare et al., 2013). We present EBQL's performance when $K = 5$ (ensemble size). Here, we compare to the DQN (Mnih et al., 2015) and DDQN (Van Hasselt et al., 2016) algorithms, the deep RL variants of Q-learning and Double-Q-Learning respectively.

While there exists a multitude of extensions and practical improvements to DQN, our work focuses on learning the Q-function at a fundamental level. Similarly to what is shown in Rainbow (Hessel et al., 2018), most of these methods are orthogonal and will thus only complement our approach.

Table 1: **ATARI:** Comparison of the DQN, DDQN and EBQL agents on 11 random ATARI environments. Each algorithm is evaluated for 50m steps over 5 random seeds. We present the average score of the final policies.

Environment \ Algorithm	Human	Random	DQN	DDQN	EBQL
Asterix	8503.3	210.0	4603.2	5718.1	**22152.5**
Breakout	31.8	1.7	283.5	333.4	**406.3**
CrazyClimber	35410.5	10780.5	93677.1	111765.4	**127967.5**
DoubleDunk	-15.5	-18.6	-15.8	-18.6	**-10.1**
Gopher	2321.0	257.6	3439.4	6940.5	**21940.0**
Pong	9.3	-20.7	17.4	19.9	**21.0**
PrivateEye	**69571.3**	24.9	78.1	100.0	100.0
Qbert	13455.0	163.9	5280.2	6277.1	**14384.4**
RoadRunner	7845.0	11.5	27671.4	40264.9	**55927.5**
Tennis	-8.9	-23.8	-1.2	-14.5	**-1.1**
VideoPinball	17297.6	16256.9	104146.4	230534.3	**361205.1**

In order to obtain a fair comparison, we *do not* use the ensemble for improved exploration (as was done in Osband et al. 2016; 2018; Lee et al. 2020). Rather, the action is selected using a standard ϵ-greedy exploration scheme, where the greedy estimator is taken on the average Q-values of the ensemble. All hyper-parameters are identical to the baselines, as reported in (Mnih et al., 2015), including the use of target networks (see Appendix E for pseudo-codes).

Analysis: We test all methods on 11 randomly-chosen environments. Each algorithm is trained over 50m steps and 5 random seeds. To ensure fairness, for DQN and DDQN we report the results as provided by Quan & Ostrovski (2020). The complete training curves over all environments are provided in Appendix F. The numerical results are shown in Table 1. In addition to the numerical results, we present the mean and median normalized scores in Fig. 6.

When comparing our method to DQN and DDQN, we observe that, in the environments we tested, EBQL consistently obtains the *highest* performance. In addition, PrivateEye is the only evaluated environment where EBQL did not outperform the human baseline. Remarkably, despite EBQL not containing many improvements (e.g., prioritized experience replay (Schaul et al., 2015), distributional estimators (Bellemare et al., 2017), and more (Hessel et al., 2018)), it outperforms the more complex Rainbow in the small-sample region ($<$ 20 million steps, Fig. 6). Particularly in the Breakout and RoadRunner domains (Appendix F), EBQL also outperforms Rainbow over all 50m training steps.

An interesting result is observed when comparing EBQL with ENSM-DQN (Anschel et al., 2017), see Fig. 6. Both methods are competitive with one another. This suggests that, as practical methods don't directly follow the theoretical guidelines (e.g., separated data-set per ensemble member), in practice the variance reduction obtained via the ensemble training has the biggest impact on performance. This in contrary to our theoretical results (Fig. 3) where we saw thatm as EBQL directly minimized the MSE and not only the variance, EBQL obtains superior performance.

7. Summary and Future Work

In this work, we presented Ensemble Bootstrapped Q-Learning, an under-estimating bias-reduction method. We started by analyzing the simple statistical task of estimating the maximal mean of a set of independent random variables. Here, we proved that the ensemble estimator is equivalent to a weighted double-estimator that unevenly splits samples between its two phases (index-selection and mean-estimation). Moreover, we showed both theoretically and empirically that in some scenarios, the optimal partition is not symmetrical. Then, using the ensemble estimator is beneficial.

Based on this analysis, we propose EBQL, a way of utilizing the ensemble estimator in RL. Our empirical results in the meta chain MDP show that, compared to QL (single estimator) and DQL (double estimator), EBQL dramatically reduces the estimation bias while obtaining superior performance. Finally, we evaluated EBQL on 11 randomly selected ATARI environments. To ensure fairness, we compared DQN, DDQN and EBQL without any additional improvements (Hessel et al., 2018). In all the evaluated environments, EBQL exhibits the best performance, in addition to outperforming the human baseline in all but a single environment. Finally, EBQL performs competitively with Rainbow in the small-samples regime, even though it does not utilize many of its well-known improvements.

Ensembles are becoming prevalent in RL, especially due to their ability to provide uncertainty estimates and improved exploration techniques (Osband et al., 2018; Lee et al., 2020). Although ensembles are integral to our method, to ensure a fair evaluation, we did not utilize their properties

for improved exploration. These results are exciting as they suggest that by utilizing these properties, the performance of EBQL can be dramatically improved.

We believe that our work leaves room to many interesting extensions. First, recall that EBQL fixes the ensemble size at the beginning of the interaction and then utilizes a single estimator to estimate the action index. However, in Section 4.1, we showed that the optimal split ratio depends on the distribution of the random variables in question. Therefore, one possible extension is to dynamically change the number of ensemble members used for the index-estimation, according to observed properties of the MDP.

Also, when considering the deep RL implementation, we provided a clean comparison of the basic algorithms (QL, DQL and EBQL). Notably, and in contrast to previous work on ensembles in RL, we did not use ensembles for improved exploration. However, doing so is non-trivial; for exploration needs, previous work decouples the ensemble members during training, whereas our update does the opposite.

Acknowledgments

This work is partially supported by the Ollendorff Center of the Viterbi Faculty of Electrical Engineering at the Technion, and by the Skillman chair in biomedical sciences. This work was also partially funded by the Israel Science Foundation under ISF grant number 2199/20. NM is partially supported by the Gutwirth scholarship.

References

Abdulhai, B., Pringle, R., and Karakoulas, G. J. Reinforcement learning for true adaptive traffic signal control. *Journal of Transportation Engineering*, 129(3):278–285, 2003.

Andrychowicz, O. M., Baker, B., Chociej, M., Jozefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

Anschel, O., Baram, N., and Shimkin, N. Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 176–185. PMLR, 2017.

Arulkumaran, K. Rainbow dqn. https://https://github.com/Kaixhin/Rainbow, 2019.

Azar, M. G., Munos, R., Ghavamzadeh, M., and Kappen, H. Speedy q-learning. In *Advances in Neural Information Processing Systems*, 2011.

Badia, A. P., Piot, B., Kapturowski, S., Sprechmann, P., Vitvitskyi, A., Guo, Z. D., and Blundell, C. Agent57: Outperforming the atari human benchmark. In *International Conference on Machine Learning*, pp. 507–517. PMLR, 2020.

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

Bellemare, M. G., Dabney, W., and Munos, R. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, pp. 449–458. PMLR, 2017.

Bellemare, M. G., Candido, S., Castro, P. S., Gong, J., Machado, M. C., Moitra, S., Ponda, S. S., and Wang, Z. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836):77–82, 2020.

Bellman, R. A markovian decision process. *Journal of mathematics and mechanics*, pp. 679–684, 1957.

Ernst, D., Geurts, P., and Wehenkel, L. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Jin, C., Allen-Zhu, Z., Bubeck, S., and Jordan, M. I. Is q-learning provably efficient? *arXiv preprint arXiv:1807.03765*, 2018.

Kearns, M. and Singh, S. Finite-sample convergence rates for q-learning and indirect algorithms. *Advances in neural information processing systems*, pp. 996–1002, 1999.

Kober, J., Bagnell, J. A., and Peters, J. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Lan, Q., Pan, Y., Fyshe, A., and White, M. Maxmin q-learning: Controlling the estimation bias of q-learning. *arXiv preprint arXiv:2002.06487*, 2020.

Lee, K., Laskin, M., Srinivas, A., and Abbeel, P. Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning. *arXiv preprint arXiv:2007.04938*, 2020.

Luong, N. C., Hoang, D. T., Gong, S., Niyato, D., Wang, P., Liang, Y.-C., and Kim, D. I. Applications of deep reinforcement learning in communications and networking: A survey. *IEEE Communications Surveys & Tutorials*, 21 (4):3133–3174, 2019.

Mahmud, M., Kaiser, M. S., Hussain, A., and Vassanelli, S. Applications of deep learning and reinforcement learning to biological data. *IEEE transactions on neural networks and learning systems*, 29(6):2063–2079, 2018.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Osband, I., Blundell, C., Pritzel, A., and Van Roy, B. Deep exploration via bootstrapped dqn. In *Advances in neural information processing systems*, pp. 4026–4034, 2016.

Osband, I., Aslanides, J., and Cassirer, A. Randomized prior functions for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 8617–8629, 2018.

Quan, J. and Ostrovski, G. DQN Zoo: Reference implementations of DQN-based agents, 2020. URL http://github.com/deepmind/dqn_zoo.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

Smith, J. E. and Winkler, R. L. The optimizer's curse: Skepticism and postdecision surprise in decision analysis. *Management Science*, 52(3):311–322, 2006.

Strehl, A. L., Li, L., Wiewiora, E., Langford, J., and Littman, M. L. Pac model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pp. 881–888, 2006.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Thaler, R. H. *The winner's curse: Paradoxes and anomalies of economic life*. Simon and Schuster, 2012.

Thrun, S. and Schwartz, A. Issues in using function approximation for reinforcement learning. In *Proceedings of the 1993 Connectionist Models Summer School Hillsdale, NJ. Lawrence Erlbaum*, 1993.

Tsitsiklis, J. N. Asynchronous stochastic approximation and q-learning. *Machine learning*, 16(3):185–202, 1994.

Van Hasselt, H. Double q-learning. *Advances in neural information processing systems*, 23:2613–2621, 2010.

Van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Van Hasselt, H., Doron, Y., Strub, F., Hessel, M., Sonnerat, N., and Modayil, J. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., and Freitas, N. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pp. 1995–2003. PMLR, 2016.

Watkins, C. J. and Dayan, P. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Wiering, M. A. Multi-agent reinforcement learning for traffic light control. In *Machine Learning: Proceedings of the Seventeenth International Conference (ICML'2000)*, pp. 1151–1158, 2000.

Xiong, R., Cao, J., and Yu, Q. Reinforcement learning-based real-time power management for hybrid energy storage system in the plug-in hybrid electric vehicle. *Applied energy*, 211:538–548, 2018.

Zhang, Z., Pan, Z., and Kochenderfer, M. J. Weighted double q-learning. In *IJCAI*, pp. 3455–3461, 2017.