A Finite-Time Analysis of Q-Learning with Neural Network Function Approximation

Pan Xu [1] **Quanquan Gu** [1]

Abstract

Q-learning with neural network function approximation (neural Q-learning for short) is among the most prevalent deep reinforcement learning algorithms. Despite its empirical success, the non-asymptotic convergence rate of neural Q-learning remains virtually unknown. In this paper, we present a finite-time analysis of a neural Q-learning algorithm, where the data are generated from a Markov decision process, and the action-value function is approximated by a deep ReLU neural network. We prove that neural Q-learning finds the optimal policy with $O(1/\sqrt{T})$ convergence rate if the neural function approximator is sufficiently overparameterized, where T is the number of iterations. To our best knowledge, our result is the first finite-time analysis of neural Q-learning under non-i.i.d. data assumption.

1. Introduction

Q-learning has been shown to be one of the most important and effective learning strategies in Reinforcement Learning (RL) over the past decades (Watkins & Dayan, 1992; Schmidhuber, 2015; Sutton & Barto, 2018), where the agent takes an action based on the action-value function (a.k.a., Q-value function) at the current state. Recent advance in deep learning has also enabled the application of Q-learning algorithms to large-scale decision problems such as mastering Go (Silver et al., 2016; 2017), robotic motion control (Levine et al., 2015; Kalashnikov et al., 2018) and autonomous driving (Shalev-Shwartz et al., 2016; Schwarting et al., 2018). In particular, the seminal work by Mnih et al. (2015) introduced the Deep Q-Network (DQN) to approximate the action-value function and achieved a superior performance versus a human expert in playing Atari games, which triggers a line of research on deep reinforcement learning such as Double Deep Q-Learning (Van Hasselt et al., 2016) and Dueling DQN (Wang et al., 2016).

Apart from its widespread empirical success in numerous applications, the convergence of Q-learning and temporal difference (TD) learning algorithms has also been extensively studied in the literature (Jaakkola et al., 1994; Baird, 1995; Tsitsiklis & Van Roy, 1997; Perkins & Pendrith, 2002; Melo et al., 2008; Mehta & Meyn, 2009; Liu et al., 2015; Bhandari et al., 2018; Lakshminarayanan & Szepesvari, 2018; Zou et al., 2019b). However, the convergence guarantee of deep Q-learning algorithms remains a largely open problem. The only exceptions are Yang et al. (2019) which studied the fitted Q-iteration (FQI) algorithm (Riedmiller, 2005; Munos & Szepesvári, 2008) with action-value function approximation based on a sparse ReLU network, and Cai et al. (2019a) which studied the global convergence of the Q-learning algorithm with an i.i.d. observation model and action-value function approximation based on a two-layer neural network. The main limitation of the aforementioned work is the unrealistic assumption that all the data used in the Q-learning algorithm are sampled i.i.d. from a fixed stationary distribution, which fails to capture the practical setting of neural Q-learning.

In this paper, in order to bridge the gap between the empirical success of neural Q-learning and the theory of conventional Q-learning (i.e., tabular Q-learning, and Q-learning with linear function approximation), we study the non-asymptotic convergence of a neural Q-learning algorithm under non-i.i.d. observations. In particular, we use a deep neural network with the ReLU activation function to approximate the action-value function. In each iteration of the neural Q-learning algorithm, it updates the network weight parameters using the temporal difference (TD) error and the gradient of the neural network function. Our work extends existing finite-time analyses for TD learning (Bhandari et al., 2018) and Q-learning (Zou et al., 2019b), from linear function approximation to deep neural network based function approximation. Compared with the very recent theoretical work for neural Q-learning (Yang et al., 2019; Cai et al., 2019a), our analysis relaxes the non-realistic i.i.d. data assumption and applies to neural network approximation with

[1]Department of Computer Science, University of California, Los Angeles. Correspondence to: Quanquan Gu <qgu@cs.ucla.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

an arbitrary number of layers. Our main contributions are summarized as follows

- We establish the first finite-time analysis of Q-learning with deep neural network function approximation when the data are generated from a Markov decision process (MDP). We show that, when the network is sufficiently wide, neural Q-learning converges to the optimal action-value function up to the approximation error of the neural network function class.
- We establish an $O(1/\sqrt{T})$ convergence rate of neural Q-learning to the optimal Q-value function up to the approximation error, where T is the number of iterations. This convergence rate matches the one for TD-learning with linear function approximation and constant step-size (Bhandari et al., 2018). Although we study a more challenging setting where the data are non-i.i.d. and the neural network approximator has multiple layers, our convergence rate also matches the $O(1/\sqrt{T})$ rate proved in Cai et al. (2019a) with i.i.d. data and a two-layer neural network approximator.

To sum up, we present a comprehensive comparison between our work and the most relevant work in terms of their respective settings and convergence rates in Table 1.

Notation We denote $[n] = \{1, \ldots, n\}$ for $n \in \mathbb{N}^+$. $\|\mathbf{x}\|_2$ is the Euclidean norm of a vector $\mathbf{x} \in \mathbb{R}^d$. For a matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$, we denote by $\|\mathbf{W}\|_2$ and $\|\mathbf{W}\|_F$ its operator norm and Frobenius norm respectively. We denote by $\text{vec}(\mathbf{W})$ the vectorization of $\mathbf{W}$, which converts $\mathbf{W}$ into a column vector. For a semi-definite matrix $\mathbf{\Sigma} \in \mathbb{R}^{d\times d}$ and a vector $\mathbf{x} \in \mathbb{R}^d$, $\|\mathbf{x}\|_{\mathbf{\Sigma}} = \sqrt{\mathbf{x}^\top \mathbf{\Sigma} \mathbf{x}}$ denotes the Mahalanobis norm. We reserve the notations $\{C_i\}_{i=0,1,\ldots}$ to represent universal positive constants that are independent of problem parameters. The specific value of $\{C_i\}_{i=1,2,\ldots}$ can be different line by line. We write $a_n = O(b_n)$ if $a_n \leq Cb_n$ for some constant $C > 0$ and $a_n = \widetilde{O}(b_n)$ if $a_n = O(b_n)$ up to some logarithmic terms of b_n.

2. Related Work

Due to the vast volume of work in the literature, we only review the most relevant work on value based reinforcement learning (e.g., TD learning and Q-learning algorithms). For policy based reinforcement learning algorithms (Sutton et al., 2000; Peters & Schaal, 2008; Silver et al., 2014; Papini et al., 2018; Xu et al., 2019a; 2020; Wang et al., 2020; Agarwal et al., 2020), we refer readers to the textbook by Sutton & Barto (2018) for details.

Asymptotic analysis The asymptotic convergence of TD learning and Q-learning algorithms has been well established in the literature (Jaakkola et al., 1994; Tsitsiklis & Van Roy, 1997; Konda & Tsitsiklis, 2000; Borkar & Meyn, 2000; Ormoneit & Sen, 2002; Melo et al., 2008; Devraj & Meyn, 2017). In particular, Tsitsiklis & Van Roy (1997) specified the precise conditions for TD learning with linear function approximation to converge and gave counterexamples that diverge. Melo et al. (2008) proved the asymptotic convergence of Q-learning with linear function approximation from standard ODE analysis and identified a critic condition on the relationship between the learning policy and the greedy policy that ensures the almost sure convergence.

Finite-time analysis The finite-time analysis of the convergence rate for Q-learning algorithms has been largely unexplored until recently. In specific, Dalal et al. (2018); Lakshminarayanan & Szepesvari (2018) studied the convergence of TD(0) algorithm with linear function approximation under i.i.d. data assumptions and constant step sizes. Concurrently, a seminal work by Bhandari et al. (2018) provided a unified framework of analysis for TD learning under both i.i.d. and Markovian noise assumptions with an extra projection step. The analysis has been extended by Zou et al. (2019b) to SARSA and Q-learning algorithms with linear function approximation. More recently, Srikant & Ying (2019) established the finite-time convergence for TD learning algorithms with linear function approximation and a constant step-size without the extra projection step under non-i.i.d. data assumptions through carefully choosing the Lyapunov function for the associated ordinary differential equation of TD update. A similar analysis was also extended to Q-learning with linear function approximation (Chen et al., 2019). Hu & Syed (2019) further provided a unified analysis for a class of TD learning algorithms using the Markov jump linear system. A multi-step Lyapunov function based approach (Wang et al., 2019) was recently proposed to remove the projection step used in Bhandari et al. (2018); Zou et al. (2019b).

Neural function approximation Despite the empirical success of DQN, the theoretical convergence of Q-learning with deep neural network approximation is still missing in the literature. Following the recent advances in the theory of deep learning for overparameterized networks (Jacot et al., 2018; Chizat & Bach, 2018; Du et al., 2019b;a; Allen-Zhu et al., 2019b;a; Zou et al., 2019a; Arora et al., 2019; Cao & Gu, 2019a; Zou & Gu, 2019; Cai et al., 2019b), two recent work by Yang et al. (2019) and Cai et al. (2019a) proved the convergence rates of fitted Q-iteration and Q-learning with a sparse multi-layer ReLU network and two-layer neural network approximation respectively, under i.i.d. observations.

3. Preliminaries

A discrete-time Markov Decision Process (MDP) is denoted by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$. $\mathcal{S}$ and $\mathcal{A}$ are the sets of all states and actions respectively. $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ is the

Table 1. Comparison with existing finite-time analyses of Q-learning.

	NON-I.I.D.	NEURAL APPROXIMATION	MULTIPLE LAYERS	RATE
BHANDARI ET AL. (2018)	✓	✗	✗	$O(1/T)$
ZOU ET AL. (2019B)	✓	✗	✗	$O(1/T)$
CHEN ET AL. (2019)	✓	✗	✗	$O(\log T/T)$
CAI ET AL. (2019A)	✗	✓	✗	$O(1/\sqrt{T})$
THIS PAPER	✓	✓	✓	$O(1/\sqrt{T})$

transition kernel such that $\mathcal{P}(s'|s,a)$ gives the probability of transiting to state s' after taking action a at state s. $r:\mathcal{S}\times\mathcal{A}\to[-1,1]$ is a deterministic reward function. $\gamma\in(0,1)$ is the discounted factor. A policy $\pi:\mathcal{S}\to\mathcal{P}(\mathcal{A})$ is a function mapping a state $s\in\mathcal{S}$ to a probability distribution $\pi(\cdot|s)$ over the action space. Let s_t and a_t denote the state and action at time step t. Then the transition kernel $\mathcal{P}$ and the policy π determine a Markov chain $\{s_t\}_{t=0,1,\ldots}$ For any fixed policy π, its associated value function $V^\pi:\mathcal{S}\to\mathbb{R}$ is defined as the expected total discounted reward:

$$V^\pi(s)=\mathbb{E}[\textstyle\sum_{t=0}^\infty\gamma^t r(s_t,a_t)|s_0=s],\quad\forall s\in\mathcal{S}.$$

The corresponding action-value function $Q^\pi:\mathcal{S}\times\mathcal{A}\to\mathbb{R}$ is defined as

$$\begin{aligned}Q^\pi(s,a)&=\mathbb{E}[\sum_{t=0}^\infty\gamma^t r(s_t,a_t)|s_0=s,a_0=a]\\&=r(s,a)+\gamma\int_{\mathcal{S}}V^\pi(s')\mathcal{P}(s'|s,a)\mathrm{d}s',\end{aligned}$$

for all $s\in\mathcal{S},a\in\mathcal{A}$. The optimal action-value function Q^* is defined as $Q^*(s,a)=\sup_\pi Q^\pi(s,a)$ for all $(s,a)\in\mathcal{S}\times\mathcal{A}$. Based on Q^*, the optimal policy π^* can be derived by following the greedy algorithm such that $\pi^*(a|s)=1$ if $Q(s,a)=\max_{b\in\mathcal{A}}Q^*(s,b)$ and $\pi^*(a|s)=0$ otherwise. We define the optimal Bellman operator $\mathcal{T}$ as follows

$$\mathcal{T}Q(s,a)=r(s,a)+\gamma\cdot\mathbb{E}\big[\max_{b\in\mathcal{A}}Q(s',b)|s'\sim\mathcal{P}(\cdot|s,a)\big].\tag{3.1}$$

It is worth noting that the optimal Bellman operator $\mathcal{T}$ is γ-contractive in the sup-norm and Q^* is the unique fixed point of $\mathcal{T}$ (Bertsekas, 1995).

4. The Neural Q-Learning Algorithm

In this section, we start with a brief review of Q-learning with linear function approximation. Then we will present the neural Q-learning algorithm.

4.1. Q-Learning with Linear Function Approximation

In many reinforcement learning algorithms, the goal is to estimate the action-value function $Q(\cdot,\cdot)$, which can be formulated as minimizing the mean-squared Bellman error (MSBE) (Sutton & Barto, 2018):

$$\min_{Q(\cdot,\cdot)}\mathbb{E}_{\mu,\pi,\mathcal{P}}\big[(\mathcal{T}Q(s,a)-Q(s,a))^2\big],\tag{4.1}$$

where state s is generated from the initial state distribution μ and action a is chosen based on a fixed learning policy π. To optimize (4.1), Q-learning iteratively updates the action-value function using the Bellman operator in (3.1), i.e., $Q_{t+1}(s,a)=\mathcal{T}Q_t(s,a)$ for all $(s,a)\in\mathcal{S}\times\mathcal{A}$. However, due to the large state and action spaces, whose cardinalities, i.e., $|\mathcal{S}|$ and $|\mathcal{A}|$, can be infinite for continuous problems in many applications, the aforementioned update is impractical. To address this issue, a linear function approximator is often used (Szepesvari, 2010; Sutton & Barto, 2018), where the action-value function is assumed to be parameterized by a linear function, i.e., $Q(s,a;\boldsymbol{\theta})=\phi(s,a)^\top\boldsymbol{\theta}$ for any $(s,a)\in\mathcal{S}\times\mathcal{A}$, where $\phi:\mathcal{S}\times\mathcal{A}\to\mathbb{R}^d$ maps the state-action pair to a d-dimensional vector, and $\boldsymbol{\theta}\in\boldsymbol{\Theta}\subseteq\mathbb{R}^d$ is an unknown weight vector. The minimization problem in (4.1) then turns to minimizing the MSBE over the parameter space $\boldsymbol{\Theta}$.

4.2. Neural Q-Learning

Analogous to Q-learning with linear function approximation, the action-value function can also be approximated by a deep neural network to increase the representation power of the approximator. Specifically, we define a L-hidden-layer neural network as follows

$$f(\boldsymbol{\theta};\mathbf{x})=\sqrt{m}\mathbf{W}_L\sigma_L(\mathbf{W}_{L-1}\cdots\sigma(\mathbf{W}_1\mathbf{x})\cdots),\tag{4.2}$$

where $\mathbf{x}\in\mathbb{R}^d$ is the input data, $\mathbf{W}_1\in\mathbb{R}^{m\times d}$, $\mathbf{W}_L\in\mathbb{R}^{1\times m}$ and $\mathbf{W}_l\in\mathbb{R}^{m\times m}$ for $l=2,\ldots,L-1$, $\boldsymbol{\theta}=(\mathrm{vec}(\mathbf{W}_1)^\top,\ldots,\mathrm{vec}(\mathbf{W}_L)^\top)^\top$ is the concatenation of the vectorization of all parameter matrices, and $\sigma(x)=\max\{0,x\}$ is the ReLU activation function. Then, we can parameterize $Q(s,a)$ using a deep neural network as $Q(s,a;\boldsymbol{\theta})=f(\boldsymbol{\theta};\phi(s,a))$, where $\boldsymbol{\theta}\in\boldsymbol{\Theta}$ and $\phi:\mathcal{S}\times\mathcal{A}\to\mathbb{R}^d$ is a feature mapping. Without loss of generality, we assume that $\|\phi(s,a)\|_2\le 1$ in this paper. Let π be an arbitrarily stationary policy. The MSBE minimization problem in (4.1) can be rewritten in the following form

$$\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}}\mathbb{E}_{\mu,\pi,\mathcal{P}}\big[(Q(s,a;\boldsymbol{\theta})-\mathcal{T}Q(s,a;\boldsymbol{\theta}))^2\big].\tag{4.3}$$

Algorithm 1 Neural Q-Learning with Gaussian Initialization

Input: learning policy π, learning rate $\{\eta_t\}_{t=0,1,\ldots}$, discount factor γ, randomly generate the entries of $\mathbf{W}_l^{(0)}$ from $N(0, 1/m)$, $l = 1, \ldots, m$
Initialization: $\boldsymbol{\theta}_0 = (\mathbf{W}_0^{(1)\top}, \ldots, \mathbf{W}_0^{(L)\top})^\top$
for $t = 0, \ldots, T-1$ **do**
  Sample data (s_t, a_t, r_t, s_{t+1}) from policy π
  $\Delta_t = f(\boldsymbol{\theta}_t; \phi(s_t, a_t)) - (r_t + \gamma \max_{b\in\mathcal{A}} f(\boldsymbol{\theta}_t; \phi(s_{t+1}, b)))$
  $\mathbf{g}_t(\boldsymbol{\theta}_t) = \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; \phi(s_t, a_t))\Delta_t$
  $\boldsymbol{\theta}_{t+1} = \Pi_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_t - \eta_t \mathbf{g}_t(\boldsymbol{\theta}_t))$
end for

Recall that the optimal action-value function Q^* is the fixed point of Bellman optimality operator $\mathcal{T}$ which is γ-contractive. Therefore Q^* is the unique global minimizer of (4.3).

The nonlinear parameterization of $Q(\cdot, \cdot)$ turns the MSBE in (4.3) to be highly nonconvex, which imposes difficulty in finding the global optimum $\boldsymbol{\theta}^*$. To mitigate this issue, we will approximate the solution of (4.3) by project the Q-value function into some function class parameterized by $\boldsymbol{\theta}$, which leads to minimizing the mean square projected Bellman error (MSPBE):

$$\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathbb{E}_{\mu,\pi,\mathcal{P}}\big[(Q(s, a; \boldsymbol{\theta}) - \Pi_{\mathcal{F}}\mathcal{T}Q(s, a; \boldsymbol{\theta}))^2\big], \quad (4.4)$$

where $\mathcal{F} = \{Q(\cdot, \cdot; \boldsymbol{\theta}) : \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$ is some function class parameterized by $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, and $\Pi_{\mathcal{F}}$ is a projection operator. Then the neural Q-learning algorithm updates the weight parameter $\boldsymbol{\theta}$ using the following descent step: $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \mathbf{g}_t(\boldsymbol{\theta}_t)$, where the gradient term $\mathbf{g}_t(\boldsymbol{\theta}_t)$ is defined as

$$\begin{aligned}\mathbf{g}_t(\boldsymbol{\theta}_t) &= \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; \phi(s_t, a_t))\big(f(\boldsymbol{\theta}_t; \phi(s_t, a_t)) \\ &\quad - r_t - \gamma \max_{b\in\mathcal{A}} f(\boldsymbol{\theta}_t; \phi(s_{t+1}, b))\big) \\ &\overset{\text{def}}{=} \Delta_t(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t)\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; \phi(s_t, a_t)), \quad (4.5)\end{aligned}$$

and Δ_t is the temporal difference (TD) error. It should be noted that $\mathbf{g}_t$ is not the gradient of the MSPBE nor an unbiased estimator for it. The details of the neural Q-learning algorithm are displayed in Algorithm 1, where $\boldsymbol{\theta}_0$ is randomly initialized, and the constraint set is chosen to be $\boldsymbol{\Theta} = \mathbb{B}(\boldsymbol{\theta}_0, \omega)$, which is defined as follows

$$\begin{aligned}\mathbb{B}(\boldsymbol{\theta}_0, \omega) \overset{\text{def}}{=} \big\{\boldsymbol{\theta} = (\mathrm{vec}(\mathbf{W}_1)^\top, \ldots, \mathrm{vec}(\mathbf{W}_L)^\top)^\top : \\ \|\mathbf{W}_l - \mathbf{W}_l^{(0)}\|_F \le \omega, l = 1, \ldots, L\big\} \quad (4.6)\end{aligned}$$

for some tunable parameter ω. It is easy to verify that $\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_2^2 = \sum_{l=1}^L \|\mathbf{W}_l - \mathbf{W}_l'\|_F^2$.

5. Convergence Analysis of Neural Q-Learning

In this section, we provide a finite-sample analysis of neural Q-learning. Note that the optimization problem in (4.4) is nonconvex. We focus on finding a surrogate action-value function in the neural network function class that well approximates Q^*.

5.1. Approximate Stationary Point in the Constrained Space

To ease the presentation, we abbreviate $f(\boldsymbol{\theta}; \phi(s, a))$ as $f(\boldsymbol{\theta})$ when no confusion arises. We define the function class $\mathcal{F}_{\boldsymbol{\Theta},m}$ as a collection of all local linearization of $f(\boldsymbol{\theta})$ at the initial point $\boldsymbol{\theta}_0$

$$\mathcal{F}_{\boldsymbol{\Theta},m} = \{f(\boldsymbol{\theta}_0) + \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}_0\rangle : \boldsymbol{\theta} \in \boldsymbol{\Theta}\}, \quad (5.1)$$

where $\boldsymbol{\Theta}$ is a constraint set. Following to the local linearization analysis in Cai et al. (2019a), we define the approximate stationary point of Algorithm 1 as follows.

Definition 5.1 (Cai et al. (2019a))**.** A point $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}$ is said to be the approximate stationary point of Algorithm 1 if for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ it holds that

$$\mathbb{E}_{\mu,\pi,\mathcal{P}}\big[\widehat{\Delta}(s, a, s'; \boldsymbol{\theta}^*)\langle \nabla_{\boldsymbol{\theta}} \widehat{f}(\boldsymbol{\theta}^*; \phi(s, a)), \boldsymbol{\theta} - \boldsymbol{\theta}^*\rangle\big] \ge 0, \quad (5.2)$$

where $\widehat{f}(\boldsymbol{\theta}; \phi(s, a)) := \widehat{f}(\boldsymbol{\theta}) \in \mathcal{F}_{\boldsymbol{\Theta},m}$ and the temporal difference error $\widehat{\Delta}$ is

$$\begin{aligned}&\widehat{\Delta}(s, a, s'; \boldsymbol{\theta}) \\ &= \widehat{f}(\boldsymbol{\theta}; \phi(s, a)) - \big(r(s, a) + \gamma \max_{b\in\mathcal{A}} \widehat{f}(\boldsymbol{\theta}; \phi(s', b))\big). \quad (5.3)\end{aligned}$$

For any $\widehat{f} \in \mathcal{F}_{\boldsymbol{\Theta},m}$, it holds that $\langle \nabla_{\boldsymbol{\theta}} \widehat{f}(\boldsymbol{\theta}^*), \boldsymbol{\theta} - \boldsymbol{\theta}^*\rangle = \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}^*\rangle = \widehat{f}(\boldsymbol{\theta}) - \widehat{f}(\boldsymbol{\theta}^*)$. Definition 5.1 immediately implies that for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ it holds that

$$\begin{aligned}&\mathbb{E}_{\mu,\pi}\big[\big(\widehat{f}(\boldsymbol{\theta}^*) - \mathcal{T}\widehat{f}(\boldsymbol{\theta}^*)\big)\big(\widehat{f}(\boldsymbol{\theta}) - \widehat{f}(\boldsymbol{\theta}^*)\big)\big] \\ &= \mathbb{E}_{\mu,\pi,\mathcal{P}}\big[\mathbb{E}_{\mathcal{P}}\big[\widehat{\Delta}(s, a, s'; \boldsymbol{\theta}^*)\big]\langle \nabla_{\boldsymbol{\theta}} \widehat{f}(\boldsymbol{\theta}^*; \phi(s, a)), \boldsymbol{\theta} - \boldsymbol{\theta}^*\rangle\big] \\ &\ge 0. \quad (5.4)\end{aligned}$$

According to Proposition 4.2 in Cai et al. (2019a), this further indicates $\widehat{f}(\boldsymbol{\theta}^*) = \Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}\mathcal{T}\widehat{f}(\boldsymbol{\theta}^*)$. In other words, $\widehat{f}(\boldsymbol{\theta}^*)$ is the unique fixed point of the MSPBE in (4.4). Therefore, we can show the convergence of neural Q-learning to the optimal action-value function Q^* by first connecting it to the minimizer $\widehat{f}(\boldsymbol{\theta}^*)$ and then adding the approximation error of $\mathcal{F}_{\boldsymbol{\Theta},m}$.

5.2. The Main Theory

Before we present the convergence of Algorithm 1, let us lay down the assumptions used throughout our paper. The first assumption controls the bias caused by the Markovian noise in the observations by assuming the uniform ergodicity of the Markov chain generated by the learning policy π.

Assumption 5.2. The learning policy π and the transition kernel $\mathcal{P}$ induce a Markov chain $\{s_t\}_{t=0,1,\ldots}$ such that there exist constants $\lambda > 0$ and $\rho \in (0,1)$ satisfying

$$\sup_{s\in\mathcal{S}} d_{TV}(\mathbb{P}(s_t \in \cdot | s_0 = s), \pi) \leq \lambda \rho^t,$$

for all $t = 0, 1, \ldots$.

Assumption 5.2 is a standard requirement in the literature (Bhandari et al., 2018; Zou et al., 2019b; Xu et al., 2019b), which can be easily satisfied as long as the Markov chain is irreducible (able to reach any state from another state with a nonzero probability) and aperiodic. The uniform ergodicity can also be established via the minorization condition for irreducible Markov chains (Meyn & Tweedie, 2012; Levin & Peres, 2017).

For the purpose of exploration, we also need to assume that the learning policy π satisfies some regularity condition. Denote $b_{\max}^{\boldsymbol{\theta}} = \operatorname{argmax}_{b\in\mathcal{A}} |\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, b), \boldsymbol{\theta}\rangle|$ for any $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. Similar to Melo et al. (2008); Zou et al. (2019b); Chen et al. (2019), we define

$$\boldsymbol{\Sigma}_\pi = \frac{1}{m}\mathbb{E}_{\mu,\pi}\big[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a)^\top\big], \quad (5.5)$$

$$\boldsymbol{\Sigma}_\pi^*(\boldsymbol{\theta}) = \frac{1}{m}\mathbb{E}_{\mu,\pi}\big[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, b_{\max}^{\boldsymbol{\theta}}) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, b_{\max}^{\boldsymbol{\theta}})^\top\big]. \quad (5.6)$$

Note that $\boldsymbol{\Sigma}_\pi$ is independent of $\boldsymbol{\theta}$ and only depends on the policy π and the initial point $\boldsymbol{\theta}_0$ in the definition of $\widehat{f}$. In contrast, $\boldsymbol{\Sigma}_\pi^*(\boldsymbol{\theta})$ is defined based on the greedy action under the policy associated with $\boldsymbol{\theta}$. The scaling parameter $1/m$ is used to ensure that the operator norm of $\boldsymbol{\Sigma}_\pi$ to be in the order of $O(1)$. It is worth noting that $\boldsymbol{\Sigma}_\pi$ is different from the neural tangent kernel (NTK) or the Gram matrix in Jacot et al. (2018); Du et al. (2019a); Arora et al. (2019), which are $n \times n$ matrices defined based on a finite set of data points $\{(s_i, a_i)\}_{i=1,\ldots,n}$. When f is linear, $\boldsymbol{\Sigma}_\pi$ reduces to the covariance matrix of the feature vector.

Assumption 5.3. There exists a constant $\alpha > 1$ such that $\boldsymbol{\Sigma}_\pi - \alpha\gamma^2\boldsymbol{\Sigma}_\pi^*(\boldsymbol{\theta}) \succ \mathbf{0}$ for all $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_0$.

In the above assumption, we essentially require that the learning policy π is not too bad compared with the greedy policy. It is worth noting that $\alpha > 0$ is of constant order even though the eigenvalues of $\boldsymbol{\Sigma}_\pi$ and $\boldsymbol{\Sigma}_\pi^*(\boldsymbol{\theta})$ could be rather small due to the scaling factor $1/m$. Assumption 5.3 is also made for Q-learning with linear function approximation in Melo et al. (2008); Zou et al. (2019b); Chen et al. (2019). Moreover, Chen et al. (2019) presented numerical simulations to verify the validity of Assumption 5.3. Cai et al. (2019a) imposed a slightly different assumption but with the same idea that the learning policy π should be not too far away from the greedy policy. The regularity assumption on the learning policy is directly imposed on the action value function in Cai et al. (2019a), which can be implied by Assumption 5.3 and thus is slightly weaker. We note that Assumption 5.3 can be relaxed to the one made in Cai et al. (2019a) without changing any of our analysis. Nevertheless, we choose to present the current version which is more consistent with existing work on Q-learning with linear function approximation (Melo et al., 2008; Chen et al., 2019).

Theorem 5.4. Suppose Assumptions 5.2 and 5.3 hold. The constraint set $\boldsymbol{\Theta}$ is defined as in (4.6). We set the radius as $\omega = C_0 m^{-1/2} L^{-9/4}$, the step size in Algorithm 1 as $\eta = 1/(2(1-\alpha^{-1/2})mT)$, and the width of the neural network as $m \geq C_1 \max\{dL^2 \log(m/\delta), \omega^{-4/3} L^{-8/3} \log(m/(\omega\delta))\}$, where $\delta \in (0,1)$. Then with probability at least $1 - 2\delta - L^2 \exp(-C_2 m^{2/3} L)$ over the randomness of the Gaussian initialization $\boldsymbol{\theta}_0$, it holds that

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\big[\big(\widehat{f}(\boldsymbol{\theta}_t) - \widehat{f}(\boldsymbol{\theta}^*)\big)^2 \big| \boldsymbol{\theta}_0\big]$$
$$\leq \frac{1}{\sqrt{T}} + \frac{C_2 \tau^* \log(T/\delta)\log T}{\beta^2\sqrt{T}} + \frac{C_3\sqrt{\log m \log(T/\delta)}}{\beta m^{1/6}},$$

where $\beta = 1 - \alpha^{-1/2} \in (0,1)$ is a constant, $\tau^* = \min\{t = 0, 1, 2, \ldots | \lambda\rho^t \leq \eta_T\}$ is the mixing time of the Markov chain $\{s_t, a_t\}_{t=0,1,\ldots}$, and $\{C_i\}_{i=0,\ldots,5}$ are universal constants independent of problem parameters.

Remark 5.5. Theorem 5.4 characterizes the distance between the output of Algorithm 1 to the approximate stationary point defined in function class $\mathcal{F}_{\boldsymbol{\Theta},m}$. From (5.4), we know that $\widehat{f}(\boldsymbol{\theta}^*)$ is the minimizer of the MSPBE (4.4). Note that τ^* is in the order of $O(\log(mT/\log T))$. Theorem 5.4 suggests that neural Q-learning converges to the minimizer of MSPBE with a rate in the order of $O((\log(mT))^3/\sqrt{T} + \log m \log T/m^{1/6})$, which reduces to $\widetilde{O}(1/\sqrt{T})$ when the width m of the neural network is sufficiently large.

In the following theorem, we show that neural Q-learning converges to the optimal action-value function within finite time if the neural network is overparameterized.

Theorem 5.6. Under the same conditions as in Theorem 5.4, with probability at least $1 - 3\delta - L^2\exp(-C_0 m^{2/3} L)$ over the randomness of $\boldsymbol{\theta}_0$, it holds that

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\big[(Q(s,a;\boldsymbol{\theta}_t) - Q^*(s,a))^2\big]$$

$$\leq \frac{3\mathbb{E}\big[\big(\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a)-Q^*(s,a)\big)^2\big]}{(1-\gamma)^2}+\frac{1}{\sqrt{T}} + \frac{C_1\tau^*\log(T/\delta)\log T}{\beta^2\sqrt{T}}+\frac{C_2\sqrt{\log(T/\delta)\log m}}{\beta m^{1/6}},$$

where all the expectations are taken conditional on $\boldsymbol{\theta}_0$, Q^* is the optimal action-value function, $\delta\in(0,1)$ and $\{C_i\}_{i=0,\ldots,2}$ are universal constants.

The optimal policy π^* can be obtained by the greedy algorithm derived based on Q^*.

Remark 5.7. The convergence rate in Theorem 5.6 can be simplifies as follows

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[(Q(s,a;\boldsymbol{\theta}_t)-Q^*(s,a))^2\big|\boldsymbol{\theta}_0] = \widetilde{O}\bigg(\mathbb{E}\big[\big(\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a)-Q^*(s,a)\big)^2\big]+\frac{1}{m^{1/6}}+\frac{1}{\sqrt{T}}\bigg).$$

The first term is the projection error of the optimal Q-value function on to the function class $\mathcal{F}_{\boldsymbol{\Theta},m}$, which decreases to zero as the representation power of $\mathcal{F}_{\boldsymbol{\Theta},m}$ increases. In fact, when the width m of the DNN is sufficiently large, recent studies (Cao & Gu, 2019a;b) show that $f(\boldsymbol{\theta})$ is almost linear around the initialization and the approximate stationary point $\widehat{f}(\boldsymbol{\theta}^*)$ becomes the fixed solution of the MSBE (Cai et al., 2019a). Moreover, this term diminishes when the Q function is approximated by linear functions when the underlying parameter has a bounded norm (Bhandari et al., 2018; Zou et al., 2019b). As m goes to infinity, we obtain the convergence of neural Q-learning to the optimal Q-value function with an $O(1/\sqrt{T})$ rate.

6. Proof of the Main Results

In this section, we provide the detailed proof of the convergence of Algorithm 1. To simplify the presentation, we write $f(\boldsymbol{\theta};\phi(s,a))$ as $f(\boldsymbol{\theta};s,a)$ throughout the proof when no confusion arises.

We first define some notations that will simplify the presentation of the proof. Recall the definition of $\mathbf{g}_t(\cdot)$ in (4.5). We define the following vector-value map $\overline{\mathbf{g}}$ that is independent of the data point.

$$\overline{\mathbf{g}}(\boldsymbol{\theta}) = \mathbb{E}_{\mu,\pi,\mathcal{P}}[\nabla_{\boldsymbol{\theta}}f(\boldsymbol{\theta};s,a)(f(\boldsymbol{\theta};s,a) - r(s,a)-\gamma\max_{b\in\mathcal{A}}f(\boldsymbol{\theta};s',b))], \tag{6.1}$$

where s follows the initial state distribution μ, a is chosen based on the policy $\pi(\cdot|s)$ and s' follows the transition probability $\mathcal{P}(\cdot|s,a)$. Similarly, we define the following gradient terms based on the linearized function $\widehat{f}\in\mathcal{F}_{\boldsymbol{\Theta},m}$

$$\begin{aligned}\mathbf{m}_t(\boldsymbol{\theta}) &= \widehat{\Delta}(s_t,a_t,s_{t+1};\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}\widehat{f}(\boldsymbol{\theta}),\\ \overline{\mathbf{m}}(\boldsymbol{\theta}) &= \mathbb{E}_{\mu,\pi,\mathcal{P}}\big[\widehat{\Delta}(s,a,s';\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}\widehat{f}(\boldsymbol{\theta})\big],\end{aligned} \tag{6.2}$$

where $\widehat{\Delta}$ is defined in (5.3), and a population version based on the linearized function.

Now we present the technical lemmas that are useful in our proof of Theorem 5.4. For the gradients $\mathbf{g}_t(\cdot)$ defined in (4.5) and $\mathbf{m}_t(\cdot)$ defined in (6.2), we have the following lemma that characterizes the difference between the gradient of the neural network function f and the gradient of the linearized function $\widehat{f}$.

Lemma 6.1. The gradient of neural network function is close to the linearized gradient. Specifically, if $\boldsymbol{\theta}_t\in\mathbb{B}(\boldsymbol{\Theta},\omega)$ and m and ω satisfy

$$m\geq C_0\max\{dL^2\log(m/\delta),\omega^{-4/3}L^{-8/3}\log(m/(\omega\delta))\},\quad\text{and}\quad C_1d^{3/2}L^{-1}m^{-3/4}\leq\omega\leq C_2L^{-6}(\log m)^{-3}, \tag{6.3}$$

then it holds that

$$\begin{aligned}&|\langle\mathbf{g}_t(\boldsymbol{\theta}_t)-\mathbf{m}_t(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle|\\ &\leq C_3(2+\gamma)\omega^{1/3}L^3\sqrt{m\log m\log(T/\delta)}\|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2\\ &\quad+\big(C_4\omega^{4/3}L^{11/3}m\sqrt{\log m}+C_5\omega^2L^4m\big)\|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2,\end{aligned}$$

with probability at least $1-2\delta-3L^2\exp(-C_6m\omega^{2/3}L)$ over the randomness of the initial point, and $\|\mathbf{g}_t(\boldsymbol{\theta}_t)\|_2\leq(2+\gamma)C_7\sqrt{m\log(T/\delta)}$ holds with probability at least $1-\delta-L^2\exp(-C_6m\omega^{2/3}L)$. where $\{C_i>0\}_{i=0,\ldots,7}$ are universal constants.

The next lemma upper bounds the bias of the non-i.i.d. data for the linearized gradient map.

Lemma 6.2. Suppose the step size sequence $\{\eta_0,\eta_1,\ldots,\eta_T\}$ is nonincreasing. Then it holds that

$$\begin{aligned}&\mathbb{E}[\langle\mathbf{m}_t(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle|\boldsymbol{\theta}_0]\\ &\leq C_0(m\log(T/\delta)+m^2\omega^2)\tau^*\eta_{\max\{0,t-\tau^*\}},\end{aligned}$$

for any fixed $t\leq T$, where $C_0>0$ is an universal constant and $\tau^*=\min\{t=0,1,2,\ldots|\lambda\rho^t\leq\eta_T\}$ is the mixing time of the Markov chain $\{s_t,a_t\}_{t=0,1,\ldots}$.

Since $\widehat{f}$ is a linear function approximator of the neural network function f, we can show that the gradient of $\widehat{f}$ satisfies the following nice property.

Lemma 6.3. Under Assumption 5.3, $\overline{\mathbf{m}}(\cdot)$ defined in (6.2) satisfies

$$\begin{aligned}&\langle\overline{\mathbf{m}}(\boldsymbol{\theta})-\overline{\mathbf{m}}(\boldsymbol{\theta}^*),\boldsymbol{\theta}-\boldsymbol{\theta}^*\rangle\\ &\geq(1-\alpha^{-1/2})\mathbb{E}\big[\big(\widehat{f}(\boldsymbol{\theta})-\widehat{f}(\boldsymbol{\theta}^*)\big)^2\big|\boldsymbol{\theta}_0\big].\end{aligned}$$

6.1. Proof of Theorem 5.4

Now we can integrate the above results and obtain proof of Theorem 5.4.

Proof of Theorem 5.4. By Algorithm 1 and the non-expansiveness of projection $\Pi_{\boldsymbol{\Theta}}$, we have

$$\begin{aligned}\|\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}^*\|_2^2 &= \|\Pi_{\boldsymbol{\Theta}}\big(\boldsymbol{\theta}_t-\eta_t\mathbf{g}_t\big)-\boldsymbol{\theta}^*\|_2^2\\ &\le \|\boldsymbol{\theta}_t-\eta_t\mathbf{g}_t-\boldsymbol{\theta}^*\|_2^2\\ &= \|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2^2+\eta_t^2\|\mathbf{g}_t\|_2^2-2\eta_t\langle\mathbf{g}_t,\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle.\end{aligned} \tag{6.4}$$

We need to find an upper bound for the gradient norm and a lower bound for the inner product. According to Definition 5.1, the approximate stationary point $\boldsymbol{\theta}^*$ of Algorithm 1 satisfies $\langle\overline{\mathbf{m}}(\boldsymbol{\theta}^*),\boldsymbol{\theta}-\boldsymbol{\theta}^*\rangle\ge 0$ for all $\boldsymbol{\theta}\in\boldsymbol{\Theta}$. The inner product in (6.4) can be decomposed into

$$\begin{aligned}&\langle\mathbf{g}_t,\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle\\ &=\langle\mathbf{g}_t-\mathbf{m}_t(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle+\langle\mathbf{m}_t(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle\\ &\quad+\langle\overline{\mathbf{m}}(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle\\ &\ge\langle\mathbf{g}_t-\mathbf{m}_t(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle+\langle\mathbf{m}_t(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle\\ &\quad+\langle\overline{\mathbf{m}}(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}^*),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle.\end{aligned} \tag{6.5}$$

Substituting (6.5) into (6.4), we have

$$\begin{aligned}\|\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}^*\|_2^2 \le{}& \|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2^2+\eta_t^2\|\mathbf{g}_t\|_2^2\\ &-2\eta_t\underbrace{\langle\mathbf{g}_t-\mathbf{m}_t(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle}_{I_1}\\ &-2\eta_t\underbrace{\langle\mathbf{m}_t(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle}_{I_2}\\ &-2\eta_t\underbrace{\langle\overline{\mathbf{m}}(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}^*),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle}_{I_3}.\end{aligned} \tag{6.6}$$

Note that the linearization error characterized in Lemma 6.1 only holds within a small neighborhood of the initial point $\boldsymbol{\theta}_0$. In the rest of this proof, we will assume that $\boldsymbol{\theta}_0,\boldsymbol{\theta}_1,\dots\boldsymbol{\theta}_T\in\mathbb{B}(\boldsymbol{\theta}_0,\omega)$ for some $\omega>0$. We will verify this condition at the end of this proof.

Recall constraint set defined in (4.6). We have $\boldsymbol{\Theta}=\mathbb{B}(\boldsymbol{\theta}_0,\omega)=\{\boldsymbol{\theta}:\|\mathbf{W}_l-\mathbf{W}_l^{(0)}\|_F\le\omega,\forall l=1,\dots,L\}$ and that m and ω satisfy the condition in (6.3).

Term I_1 is the error of the local linearization of $f(\boldsymbol{\theta})$ at $\boldsymbol{\theta}_0$. By Lemma 6.1, with probability at least $1-2\delta-3L^2\exp(-C_1m\omega^{2/3}L)$ over the randomness of the initial point $\boldsymbol{\theta}_0$, we have

$$\begin{aligned}&|\langle\mathbf{g}_t-\mathbf{m}_t(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle|\\ &\le C_2(2+\gamma)m^{-1/6}\sqrt{\log m\log(T/\delta)}\end{aligned} \tag{6.7}$$

holds uniformly for all $\boldsymbol{\theta}_t,\boldsymbol{\theta}^*\in\boldsymbol{\Theta}$, where we used the fact that $\omega=C_0m^{-1/2}L^{-9/4}$.

Term I_2 is the bias of caused by the non-i.i.d. data (s_t,a_t,s_{t+1}) used in the update of Algorithm 1. Conditional on the initialization, by Lemma 6.2, we have

$$\begin{aligned}&\mathbb{E}[\langle\mathbf{m}_t(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle|\boldsymbol{\theta}_0]\\ &\le C_3(m\log(T/\delta)+m^2\omega^2)\tau^*\eta_{\max\{0,t-\tau^*\}},\end{aligned} \tag{6.8}$$

where $\tau^*=\min\{t=0,1,2,\dots|\lambda\rho^t\le\eta_T\}$ is the mixing time of the Markov chain $\{s_t,a_t\}_{t=0,1,\dots}$.

Term I_3 is the estimation error for the linear function approximation. By Lemma 6.3, we have

$$\langle\overline{\mathbf{m}}(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}^*),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle\ge\beta\mathbb{E}\big[\big(\widehat{f}(\boldsymbol{\theta}_t)-\widehat{f}(\boldsymbol{\theta}^*)\big)^2\big|\boldsymbol{\theta}_0\big], \tag{6.9}$$

where $\beta=(1-\alpha^{-1/2})\in(0,1)$ is a constant. Substituting (6.7), (6.8) and (6.9) into (6.6), we have it holds that

$$\begin{aligned}&\|\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}^*\|_2^2\\ &\le\|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2^2+\eta_t^2C_4^2(2+\gamma)^2m\log(T/\delta)\\ &\quad+2\eta_tC_2(2+\gamma)m^{-1/6}\sqrt{\log m\log(T/\delta)}\\ &\quad+2\eta_tC_3(m\log(T/\delta)+m^2\omega^2)\tau^*\eta_{\max\{0,t-\tau^*\}}\\ &\quad-2\eta_t\beta\mathbb{E}\big[\big(\widehat{f}(\boldsymbol{\theta}_t)-\widehat{f}(\boldsymbol{\theta}^*)\big)^2\big|\boldsymbol{\theta}_0\big],\end{aligned} \tag{6.10}$$

with probability at least $1-2\delta-3L^2\exp(-C_1m\omega^{2/3}L)$ over the randomness of the initial point $\boldsymbol{\theta}_0$, where we used the fact that $\|\mathbf{g}_t\|_F\le C_4(2+\gamma)\sqrt{m\log(T/\delta)}$ from Lemma 6.1. Rearranging the above inequality yields

$$\begin{aligned}&\mathbb{E}\big[\big(\widehat{f}(\boldsymbol{\theta}_t)-\widehat{f}(\boldsymbol{\theta}^*)\big)^2\big|\boldsymbol{\theta}_0\big]\\ &\le\frac{\|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2^2-\|\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}^*\|_2^2}{2\beta\eta_t}\\ &\quad+\frac{C_2(2+\gamma)m^{-1/6}\sqrt{\log m\log(T/\delta)}}{\beta}\\ &\quad+\frac{C_4(2+\gamma)^2m\log(T/\delta)\eta_t}{\beta}\\ &\quad+\frac{C_3m(\log(T/\delta)+m\omega^2)\tau^*\eta_{\max\{0,t-\tau^*\}}}{\beta},\end{aligned}$$

with probability at least $1-2\delta-3L^2\exp(-C_1m\omega^{2/3}L)$ over the randomness of the initial point $\boldsymbol{\theta}_0$. Recall the choices of the step sizes $\eta_0=\dots=\eta_T=1/(2\beta m\sqrt{T})$ and the radius $\omega=C_0m^{-1/2}L^{-9/4}$. Dividing the above inequality by T and telescoping it from $t=0$ to T yields

$$\begin{aligned}&\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big[\big(\widehat{f}(\boldsymbol{\theta}_t)-\widehat{f}(\boldsymbol{\theta}^*)\big)^2\big|\boldsymbol{\theta}_0\big]\\ &\le\frac{m\|\boldsymbol{\theta}_0-\boldsymbol{\theta}^*\|_2^2}{\sqrt{T}}+\frac{C_2(2+\gamma)m^{-1/6}\sqrt{\log m\log(T/\delta)}}{\beta}\\ &\quad+\frac{C_4(2+\gamma)^2\log(T/\delta)\log T}{\beta^2\sqrt{T}}\\ &\quad+\frac{C_3(\log(T/\delta)+1)\tau^*\log T}{\beta\sqrt{T}}.\end{aligned}$$

For $\boldsymbol{\theta}_0,\boldsymbol{\theta}^*\in\boldsymbol{\Theta}$, again by $\omega=Cm^{-1/2}L^{-9/4}$, we have $\|\boldsymbol{\theta}_0-\boldsymbol{\theta}^*\|_2^2\le 1/m$. Since $\widehat{f}(\cdot)\in\mathcal{F}_{\boldsymbol{\Theta},m}$, by Lemma 6.1, it

holds with probability at least $1-2\delta-3L^2\exp(-C_0m^{2/3}L)$ over the randomness of the initial point $\boldsymbol{\theta}_0$ that

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big[\big(\widehat{f}(\boldsymbol{\theta}_t)-\widehat{f}(\boldsymbol{\theta}^*)\big)^2\big|\boldsymbol{\theta}_0\big] \le \frac{1}{\sqrt{T}}+\frac{C_1\tau^*\log(T/\delta)\log T}{\beta^2\sqrt{T}}+\frac{C_2\sqrt{\log m\log(T/\delta)}}{\beta m^{1/6}},$$

where we used the fact that $\gamma<1$. This completes the proof. □

6.2. Proof of Theorem 5.6

Before we prove the global convergence of Algorithm 1, we present the following lemma that shows that near the initialization point $\boldsymbol{\theta}_0$, the neural network function $f(\boldsymbol{\theta};\mathbf{x})$ is almost linear in $\boldsymbol{\theta}$ for all unit input vectors.

Lemma 6.4 (Theorems 5.3 and 5.4 in Cao & Gu (2019a))**.** Let $\boldsymbol{\theta}_0=(\mathbf{W}_0^{(1)\top},\ldots,\mathbf{W}_0^{(L)\top})^\top$ be the initial point and $\boldsymbol{\theta}=(\mathbf{W}^{(1)\top},\ldots,\mathbf{W}^{(L)\top})^\top\in\mathbb{B}(\boldsymbol{\theta}_0,\omega)$ be a point in the neighborhood of $\boldsymbol{\theta}_0$. If

$$m\ge C_1\max\{dL^2\log(m/\delta),\omega^{-4/3}L^{-8/3}\log(m/(\omega\delta))\},\quad \omega\le C_2L^{-5}(\log m)^{-3/2},$$

then for all $\mathbf{x}\in S^{d-1}$, with probability at least $1-\delta$ it holds that

$$|f(\boldsymbol{\theta};\mathbf{x})-\widehat{f}(\boldsymbol{\theta};\mathbf{x})| \le \omega^{1/3}L^{8/3}\sqrt{m\log m}\sum_{l=1}^{L}\big\|\mathbf{W}^{(l)}-\mathbf{W}_0^{(l)}\big\|_2 + C_3L^3\sqrt{m}\sum_{l=1}^{L}\big\|\mathbf{W}^{(l)}-\mathbf{W}_0^{(l)}\big\|_2^2.$$

Under the same conditions on m and ω, if $\boldsymbol{\theta}_t\in\mathbb{B}(\boldsymbol{\theta}_0,\omega)$ for all $t=1,\ldots,T$, then with probability at least $1-\delta$, we have $|f(\boldsymbol{\theta}_t;\phi(s_t,a_t))|\le C_4\sqrt{\log(T/\delta)}$ for all $t\in[T]$.

Proof of Theorem 5.6. To simplify the notation, we abbreviate $\mathbb{E}[\cdot|\boldsymbol{\theta}_0]$ as $\mathbb{E}[\cdot]$ in the rest of this proof. Therefore, we have

$$\begin{aligned}\mathbb{E}\big[(Q(s,a;\boldsymbol{\theta}_T)-Q^*(s,a))^2\big] &\le 3\mathbb{E}\big[\big(f(\boldsymbol{\theta}_T;s,a)-\widehat{f}(\boldsymbol{\theta}_T;s,a)\big)^2\big]\\ &\quad+3\mathbb{E}\big[\big(\widehat{f}(\boldsymbol{\theta}_T;s,a)-\widehat{f}(\boldsymbol{\theta}^*;s,a)\big)^2\big]\\ &\quad+3\mathbb{E}\big[\big(\widehat{f}(\boldsymbol{\theta}^*;s,a)-Q^*(s,a)\big)^2\big].\end{aligned}\tag{6.11}$$

By Lemma 6.4 and the parameter choice that $\omega=C_1/(\sqrt{m}L^{9/4})$, we have

$$\mathbb{E}[(f(\boldsymbol{\theta}_T;s,a)-\widehat{f}(\boldsymbol{\theta}_T;s,a))^2]\le C_2(\omega^{4/3}L^4\sqrt{m\log m})^2 \le C_1^{4/3}C_2m^{-1/3}\log m\tag{6.12}$$

with probability at least $1-\delta$. Recall that $\widehat{f}(\boldsymbol{\theta}^*;\cdot,\cdot)$ is the fixed point of $\Pi_{\mathcal{F}}\mathcal{T}$ and $Q^*(\cdot,\cdot)$ is the fixed point of $\mathcal{T}$. Then we have

$$\begin{aligned}&\big|\widehat{f}(\boldsymbol{\theta}^*;s,a)-Q^*(s,a)\big|\\ &=\big|\widehat{f}(\boldsymbol{\theta}^*;s,a)-\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a)+\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a)-Q^*(s,a)\big|\\ &=\big|\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}\mathcal{T}\widehat{f}(\boldsymbol{\theta}^*;s,a)-\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}\mathcal{T}Q^*(s,a)+\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a)-Q^*(s,a)\big|\\ &\le\big|\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}\mathcal{T}\widehat{f}(\boldsymbol{\theta}^*;s,a)-\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}\mathcal{T}Q^*(s,a)\big|+\big|\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a)-Q^*(s,a)\big|\\ &\le\gamma|\widehat{f}(\boldsymbol{\theta}^*;s,a)-Q^*(s,a)|+\big|\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a)-Q^*(s,a)\big|,\end{aligned}$$

where the first inequality follows the triangle inequality and in the second inequality we used the fact that $\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}\mathcal{T}$ is γ-contractive. This further leads to

$$(1-\gamma)|\widehat{f}(\boldsymbol{\theta}^*;s,a)-Q^*(s,a)|\le|\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a)-Q^*(s,a)|.\tag{6.13}$$

Combining (6.12), (6.13) and the result from Theorem 5.4 and substituting them back into (6.11), we have

$$\mathbb{E}\big[(Q(s,a;\boldsymbol{\theta}_T)-Q^*(s,a))^2\big] \le\frac{3\mathbb{E}\big[\big(\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a)-Q^*(s,a)\big)^2\big]}{(1-\gamma)^2}+\frac{1}{\sqrt{T}} +\frac{C_2\tau^*\log(T/\delta)\log T}{\beta^2\sqrt{T}}+\frac{C_3\sqrt{\log(T/\delta)\log m}}{\beta m^{1/6}},$$

with probability at least $1-3\delta-L^2\exp(-C_6m^{2/3}L)$, which completes the proof. □

7. Conclusions

In this paper, we provide the first finite-time analysis of Q-learning with neural network function approximation (i.e., neural Q-learning), where the data are generated from a Markov decision process and the action-value function is approximated by a deep ReLU neural network. We prove that neural Q-learning converge to the optimal action-value function up to the approximation error with $O(1/\sqrt{T})$ rate, where T is the number of iterations. Our proof technique is of independent interest and can be extended to analyze other deep reinforcement learning algorithms. One interesting future direction would be to remove the projection step in our algorithm by applying the ODE based analysis in Srikant & Ying (2019); Chen et al. (2019).

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. This research was sponsored in part by the National Science Foundation IIS-1904183 and Adobe Data Science Research Award. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

References

Agarwal, A., Kakade, S. M., Lee, J. D., and Mahajan, G. Optimality and approximation with policy gradient methods in markov decision processes. In *Conference On Learning Theory*, 2020.

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in neural information processing systems*, 2019a.

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pp. 242–252, 2019b.

Arora, S., Du, S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pp. 322–332, 2019.

Baird, L. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pp. 30–37. Elsevier, 1995.

Bertsekas, D. P. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995.

Bhandari, J., Russo, D., and Singal, R. A finite time analysis of temporal difference learning with linear function approximation. In *Conference On Learning Theory*, pp. 1691–1692, 2018.

Borkar, V. S. and Meyn, S. P. The ode method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal on Control and Optimization*, 38 (2):447–469, 2000.

Cai, Q., Yang, Z., Lee, J. D., and Wang, Z. Neural temporal-difference learning converges to global optima. In *Advances in Neural Information Processing Systems*, 2019a.

Cai, T., Gao, R., Hou, J., Chen, S., Wang, D., He, D., Zhang, Z., and Wang, L. A gram-gauss-newton method learning overparameterized deep neural networks for regression problems. *arXiv preprint arXiv:1905.11675*, 2019b.

Cao, Y. and Gu, Q. A generalization theory of gradient descent for learning over-parameterized deep relu networks. *arXiv preprint arXiv:1902.01384*, 2019a.

Cao, Y. and Gu, Q. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Advances in Neural Information Processing Systems*, 2019b.

Chen, Z., Zhang, S., Doan, T. T., Maguluri, S. T., and Clarke, J.-P. Performance of q-learning with linear function approximation: Stability and finite-time analysis. *arXiv preprint arXiv:1905.11425*, 2019.

Chizat, L. and Bach, F. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in neural information processing systems*, pp. 3036–3046, 2018.

Dalal, G., Szörényi, B., Thoppe, G., and Mannor, S. Finite sample analyses for td (0) with function approximation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Devraj, A. M. and Meyn, S. Zap q-learning. In *Advances in Neural Information Processing Systems*, pp. 2235–2244, 2017.

Du, S., Lee, J., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, pp. 1675–1685, 2019a.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019b. URL https://openreview.net/forum?id=S1eK3i09YQ.

Hu, B. and Syed, U. A. Characterizing the exact behaviors of temporal difference learning algorithms using markov jump linear system theory. In *Advances in Neural Information Processing Systems*, 2019.

Jaakkola, T., Jordan, M. I., and Singh, S. P. Convergence of stochastic iterative dynamic programming algorithms. In *Advances in Neural Information Processing Systems*, pp. 703–710, 1994.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pp. 8571–8580, 2018.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pp. 651–673, 2018.

Konda, V. R. and Tsitsiklis, J. N. Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, pp. 1008–1014, 2000.

Lakshminarayanan, C. and Szepesvari, C. Linear stochastic approximation: How far does constant step-size and iterate averaging go? In *International Conference on Artificial Intelligence and Statistics*, pp. 1347–1355, 2018.

Levin, D. A. and Peres, Y. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

Levine, S., Wagener, N., and Abbeel, P. Learning contact-rich manipulation skills with guided policy search. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 156–163. IEEE, 2015.

Liu, B., Liu, J., Ghavamzadeh, M., Mahadevan, S., and Petrik, M. Finite-sample analysis of proximal gradient td algorithms. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, pp. 504–513. AUAI Press, 2015.

Mehta, P. and Meyn, S. Q-learning and pontryagin's minimum principle. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pp. 3598–3605. IEEE, 2009.

Melo, F. S., Meyn, S. P., and Ribeiro, M. I. An analysis of reinforcement learning with function approximation. In *Proceedings of the 25th International Conference on Machine Learning*, pp. 664–671. ACM, 2008.

Meyn, S. P. and Tweedie, R. L. *Markov chains and stochastic stability*. Springer Science & Business Media, 2012.

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, 2015.

Munos, R. and Szepesvári, C. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9 (May):815–857, 2008.

Ormoneit, D. and Sen, Ś. Kernel-based reinforcement learning. *Machine learning*, 49(2-3):161–178, 2002.

Papini, M., Binaghi, D., Canonaco, G., Pirotta, M., and Restelli, M. Stochastic variance-reduced policy gradient. In *International Conference on Machine Learning*, pp. 4023–4032, 2018.

Perkins, T. J. and Pendrith, M. D. On the existence of fixed points for q-learning and sarsa in partially observable domains. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pp. 490–497. Morgan Kaufmann Publishers Inc., 2002.

Peters, J. and Schaal, S. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4): 682–697, 2008.

Riedmiller, M. Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *European Conference on Machine Learning*, pp. 317–328. Springer, 2005.

Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

Schwarting, W., Alonso-Mora, J., and Rus, D. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 2018.

Shalev-Shwartz, S., Shammah, S., and Shashua, A. Safe, multi-agent, reinforcement learning for autonomous driving. *CoRR*, abs/1610.03295, 2016. URL http://arxiv.org/abs/1610.03295.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, 2014.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T. P., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

Srikant, R. and Ying, L. Finite-time error bounds for linear stochastic approximation and td learning. *arXiv preprint arXiv:1902.00923*, 2019.

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

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, pp. 1057–1063, 2000.

Szepesvari, C. Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning*, 4(1):1–103, 2010.

Tsitsiklis, J. N. and Van Roy, B. Analysis of temporal-diffference learning with function approximation. In *Advances in Neural Information Processing Systems*, pp. 1075–1081, 1997.

Van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *Thirtieth AAAI conference on artificial intelligence*, 2016.

Wang, G., Li, B., and Giannakis, G. B. A multistep lyapunov approach for finite-time analysis of biased stochastic approximation. *arXiv preprint arXiv:1909.04299*, 2019.

Wang, L., Cai, Q., Yang, Z., and Wang, Z. Neural policy gradient methods: Global optimality and rates of convergence. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJgQfkSYDS.

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, 2016.

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

Xu, P., Gao, F., and Gu, Q. An improved convergence analysis of stochastic variance-reduced policy gradient. In *International Conference on Uncertainty in Artificial Intelligence*, 2019a.

Xu, P., Gao, F., and Gu, Q. Sample efficient policy gradient methods with recursive variance reduction. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJlxIJBFDr.

Xu, T., Zou, S., and Liang, Y. Two time-scale off-policy td learning: Non-asymptotic analysis over markovian samples. In *Advances in Neural Information Processing Systems*, pp. 10634–10644, 2019b.

Yang, Z., Xie, Y., and Wang, Z. A theoretical analysis of deep q-learning. *arXiv preprint arXiv:1901.00137*, 2019.

Zou, D. and Gu, Q. An improved analysis of training over-parameterized deep neural networks. In *Advances in Neural Information Processing Systems*, 2019.

Zou, D., Cao, Y., Zhou, D., and Gu, Q. Stochastic gradient descent optimizes over-parameterized deep relu networks. *Machine Learning*, 2019a.

Zou, S., Xu, T., and Liang, Y. Finite-sample analysis for sarsa and q-learning with linear function approximation. In *Advances in Neural Information Processing Systems*, 2019b.