
Neural Temporal-Difference Learning Converges to Global Optima

Qi Cai ^{*} Zhuoran Yang [†] Jason D. Lee [‡] Zhaoran Wang ^{*}

Abstract

Temporal-difference learning (TD), coupled with neural networks, is among the most fundamental building blocks of deep reinforcement learning. However, due to the nonlinearity in value function approximation, such a coupling leads to non-convexity and even divergence in optimization. As a result, the global convergence of neural TD remains unclear. In this paper, we prove for the first time that neural TD converges at a sublinear rate to the global optimum of the mean-squared projected Bellman error for policy evaluation. In particular, we show how such global convergence is enabled by the overparametrization of neural networks, which also plays a vital role in the empirical success of neural TD. Beyond policy evaluation, we establish the global convergence of neural (soft) Q-learning, which is further connected to that of policy gradient algorithms.

1 Introduction

Given a policy, temporal-different learning (TD) [49] aims to learn the corresponding (action-)value function by following the semigradients of the mean-squared Bellman error in an online manner. As the most-used policy evaluation algorithm, TD serves as the “critic” component of many reinforcement learning algorithms, such as the actor-critic algorithm [31] and trust-region policy optimization [47]. In particular, in deep reinforcement learning, TD is often applied to learn value functions parametrized by neural networks [36, 39, 24], which gives rise to neural TD. As policy improvement relies crucially on policy evaluation, the optimization efficiency and statistical accuracy of neural TD are critical to the performance of deep reinforcement learning. Towards theoretically understanding deep reinforcement learning, the goal of this paper is to characterize the convergence of neural TD.

Despite the broad applications of neural TD, its convergence remains rarely understood. Even with linear value function approximation, the nonasymptotic convergence of TD remains open until recently [6, 33, 14, 48, 45], although its asymptotic convergence is well understood [28, 55, 9, 32, 8]. Meanwhile, with nonlinear value function approximation, TD is known to diverge in general [4, 11, 55]. To remedy this issue, [7] propose nonlinear (gradient) TD, which uses the tangent vectors of nonlinear value functions in place of the feature vectors in linear TD. Unlike linear TD, which converges to the global optimum of the mean-squared projected Bellman error (MSPBE), nonlinear TD is only guaranteed to converge to a local optimum asymptotically. As a result, the statistical accuracy of the value function learned by nonlinear TD remains unclear. In contrast to such conservative theory, neural TD, which straightforwardly combines TD with neural networks without the explicit local linearization in nonlinear TD, often learns a desired value function that generalizes well to unseen states in practice [18, 2, 26]. Hence, a gap separates theory from practice.

^{*}Department of Industrial Engineering and Management Sciences, Northwestern University

[†]Department of Operations Research and Financial Engineering, Princeton University

[‡]Department of Electronic Engineering, Princeton University

There exist three obstacles towards closing such a theory-practice gap: (i) MSPBE has an expectation over the transition dynamics within the squared loss, which forbids the construction of unbiased stochastic gradients [50]. As a result, even with linear value function approximation, TD largely eludes the classical optimization framework, as it follows biased stochastic semigradients. (ii) When the value function is parametrized by a neural network, MSPBE is nonconvex in the weights of the neural network, which may introduce undesired stationary points such as local optima and saddle points [30]. As a result, even an ideal algorithm that follows the population gradients of MSPBE may get trapped. (iii) Due to the interplay between the bias in stochastic semigradients and the nonlinearity in value function approximation, neural TD may even diverge [4, 11, 55], instead of converging to an undesired stationary point, as it lacks the explicit local linearization in nonlinear TD [7]. Such divergence is also not captured by the classical optimization framework.

Contribution. Towards bridging theory and practice, we establish the first nonasymptotic global rate of convergence of neural TD. In detail, we prove that randomly initialized neural TD converges to the global optimum of MSPBE at the rate of $1/T$ with population semigradients and at the rate of $1/\sqrt{T}$ with stochastic semigradients. Here T is the number of iterations and the (action-)value function is parametrized by a sufficiently wide two-layer neural network. Moreover, we prove that the projection in MSPBE allows for a sufficiently rich class of functions, which has the same representation power of a reproducing kernel Hilbert space associated with the random initialization. As a result, for a broad class of reinforcement learning problems, neural TD attains zero MSPBE. Beyond policy evaluation, we further establish the global convergence of neural (soft) Q-learning, which allows for policy improvement. In particular, we prove that, under stronger regularity conditions, neural (soft) Q-learning converges at the same rate of neural TD to the global optimum of MSPBE for policy optimization. Also, by exploiting the connection between (soft) Q-learning and policy gradient algorithms [46, 24], we establish the global convergence of a variant of the policy gradient algorithm [58, 53, 50].

At the core of our analysis is the overparametrization of the two-layer neural network for value function approximation [59, 41, 1, 3], which enables us to circumvent the three obstacles above. In particular, overparametrization leads to an implicit local linearization that varies smoothly along the solution path, which mirrors the explicit one in nonlinear TD [7]. Such an implicit local linearization enables us to circumvent the third obstacle of possible divergence. Moreover, overparametrization allows us to establish a notion of one-point monotonicity [25, 19] for the semigradients followed by neural TD, which ensures its evolution towards the global optimum of MSPBE along the solution path. Such a notion of monotonicity enables us to circumvent the first and second obstacles of bias and nonconvexity. Broadly speaking, our theory backs the empirical success of overparametrized neural networks in deep reinforcement learning. In particular, we show that instead of being a curse, overparametrization is indeed a blessing for minimizing MSPBE in the presence of bias, nonconvexity, and even divergence.

More Related Work. There is a large body of literature on the convergence of linear TD under both asymptotic [28, 55, 9, 32, 8] and nonasymptotic [6, 33, 14, 48] regimes. See [16] for a detailed survey. In particular, our analysis is based on the recent breakthrough in the nonasymptotic analysis of linear TD [6] and its extension to linear Q-learning [60]. An essential step of our analysis is bridging the evolution of linear TD and neural TD through the implicit local linearization induced by overparametrization.

To incorporate nonlinear value function approximation into TD, [7] propose the first convergent nonlinear TD based on explicit local linearization, which however only converges to a local optimum of MSPBE. See [21, 5] for a detailed survey. In contrast, we prove that, with the implicit local linearization induced by overparametrization, neural TD, which is simpler to implement and more widely used in deep reinforcement learning than nonlinear TD, provably converges to the global optimum of MSPBE.

There exist various extensions of TD, including least-squares TD [12, 10, 34, 22, 56] and gradient TD [51, 52, 7, 37, 17, 57, 54]. In detail, least-squares TD is based on batch update, which loses the computational and statistical efficiency of the online update in TD. Meanwhile, gradient TD follows unbiased stochastic gradients, but at the cost of introducing another optimization variable. Such a reformulation leads to bilevel optimization, which is less stable in practice when combined with neural networks [42]. As a result, both extensions of TD are less widely used in deep reinforcement

learning [18, 2, 26]. Moreover, when using neural networks for value function approximation, the convergence to the global optimum of MSPBE remains unclear for both extensions of TD.

Our work is also related to the recent breakthrough in understanding overparametrized neural networks, especially their generalization error [59, 41, 1, 3]. See [20] for a detailed survey. In particular, [15, 1, 3, 13, 29, 35] characterize the implicit local linearization in the context of supervised learning, where we train an overparametrized neural network by following the stochastic gradients of the mean-squared error. In contrast, neural TD does not follow the stochastic gradients of any objective function, hence leading to possible divergence, which makes the convergence analysis more challenging.

2 Background

In Section 2.1, we briefly review policy evaluation in reinforcement learning. In Section 2.2, we introduce the corresponding optimization formulations.

2.1 Policy Evaluation

We consider a Markov decision process $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$, in which an agent interacts with the environment to learn the optimal policy that maximizes the expected total reward. At the t -th time step, the agent has a state $s_t \in \mathcal{S}$ and takes an action $a_t \in \mathcal{A}$. Upon taking the action, the agent enters the next state $s_{t+1} \in \mathcal{S}$ according to the transition probability $\mathcal{P}(\cdot | s_t, a_t)$ and receives a random reward $r_t = r(s_t, a_t)$ from the environment. The action that the agent takes at each state is decided by a policy $\pi : \mathcal{S} \rightarrow \Delta$, where Δ is the set of all probability distributions over $\mathcal{A}$. The performance of policy π is measured by the expected total reward, $J(\pi) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t | a_t \sim \pi(s_t)]$, where $\gamma < 1$ is the discount factor.

Given policy π , policy evaluation aims to learn the following two functions, the value function $V^\pi(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a_t \sim \pi(s_t)]$ and the action-value function (Q-function) $Q^\pi(s, a) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a_0 = a, a_t \sim \pi(s_t)]$. Both functions form the basis for policy improvement. Without loss of generality, we focus on learning the Q-function in this paper. We define the Bellman evaluation operator,

$$\mathcal{T}^\pi Q(s, a) = \mathbb{E}[r(s, a) + \gamma Q(s', a') | s' \sim \mathcal{P}(\cdot | s, a), a' \sim \pi(s')], \quad (2.1)$$

for which Q^π is the fixed point, that is, the solution to the Bellman equation $Q = \mathcal{T}^\pi Q$.

2.2 Optimization Formulation

Corresponding to (2.1), we aim to learn Q^π by minimizing the mean-squared Bellman error (MSBE),

$$\min_{\theta} \text{MSBE}(\theta) = \mathbb{E}_{(s,a) \sim \mu} [(\hat{Q}_\theta(s, a) - \mathcal{T}^\pi \hat{Q}_\theta(s, a))^2], \quad (2.2)$$

where the Q-function is parametrized as $\hat{Q}_\theta$ with parameter θ . Here μ is the stationary distribution of (s, a) corresponding to policy π . Due to Q-function approximation, we focus on minimizing the following surrogate of MSBE, namely the projected mean-squared Bellman error (MSPBE),

$$\min_{\theta} \text{MSPBE}(\theta) = \mathbb{E}_{(s,a) \sim \mu} [(\hat{Q}_\theta(s, a) - \Pi_{\mathcal{F}} \mathcal{T}^\pi \hat{Q}_\theta(s, a))^2]. \quad (2.3)$$

Here $\Pi_{\mathcal{F}}$ is the projection onto a function class $\mathcal{F}$. For example, for linear Q-function approximation [49], $\mathcal{F}$ takes the form $\{\hat{Q}_{\theta'} : \theta' \in \Theta\}$, where $\hat{Q}_{\theta'}$ is linear in θ' and Θ is the set of feasible parameters. As another example, for nonlinear Q-function approximation [7], $\mathcal{F}$ takes the form $\{\hat{Q}_\theta + \nabla_{\theta} \hat{Q}_\theta^\top (\theta' - \theta) : \theta' \in \Theta\}$, which consists of the local linearization of $\hat{Q}_{\theta'}$ at θ .

Throughout this paper, we assume that we are able to sample tuples in the form of (s, a, r, s', a') from the stationary distribution of policy π in an independent and identically distributed manner, although our analysis can be extended to handle temporal dependence using the proof techniques of [6]. With a slight abuse of notation, we use μ to denote the stationary distribution of (s, a, r, s', a') corresponding to policy π and any of its marginal distributions.

3 Neural Temporal-Difference Learning

TD updates the parameter θ of the Q-function by taking the stochastic semigradient descent step [49, 53, 50],

$$\theta' \leftarrow \theta - \eta \cdot (\hat{Q}_\theta(s, a) - r(s, a) - \gamma \hat{Q}_\theta(s', a')) \cdot \nabla_\theta \hat{Q}_\theta(s, a), \quad (3.1)$$

which corresponds to the MSBE in (2.2). Here $(s, a, r, s', a') \sim \mu$ and $\eta > 0$ is the stepsize. In a more general context, (3.1) is referred to as TD(0). In this paper, we focus on TD(0), which is abbreviated as TD, and leave the extension to TD(λ) to future work.

In the sequel, we denote the state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$ by a vector $x \in \mathcal{X} \subseteq \mathbb{R}^d$ with $d > 2$. We consider $\mathcal{S}$ to be continuous and $\mathcal{A}$ to be finite. Without loss of generality, we assume that $\|x\|_2 = 1$ and $|r(x)|$ is upper bounded by a constant $\bar{r}$ for any $x \in \mathcal{X}$. We use a two-layer neural network

$$\hat{Q}(x; W) = \frac{1}{\sqrt{m}} \sum_{r=1}^m b_r \sigma(W_r^\top x) \quad (3.2)$$

to parametrize the Q-function. Here σ is the rectified linear unit (ReLU) activation function $\sigma(y) = \max\{0, y\}$ and the parameter $\theta = (b_1, \dots, b_m, W_1, \dots, W_m)$ are initialized as $b_r \sim \text{Unif}(\{-1, 1\})$ and $W_r \sim N(0, I_d/d)$ for any $r \in [m]$ independently. During training, we only update $W = (W_1, \dots, W_m) \in \mathbb{R}^{md}$, while keeping $b = (b_1, \dots, b_m) \in \mathbb{R}^m$ fixed as the random initialization. To ensure global convergence, we incorporate an additional projection step with respect to W . See Algorithm 1 for a detailed description.

Algorithm 1 Neural TD

- 1: **Initialization:** $b_r \sim \text{Unif}(\{-1, 1\})$, $W_r(0) \sim N(0, I_d/d)$ ($r \in [m]$), $\bar{W} = W(0)$,
 $S_B = \{W \in \mathbb{R}^{md} : \|W - W(0)\|_2 \leq B\}$ ($B > 0$)
 - 2: **For** $t = 0$ to $T - 2$:
 - 3: Sample a tuple (s, a, r, s', a') from the stationary distribution μ of policy π
 - 4: Let $x = (s, a)$, $x' = (s', a')$
 - 5: Bellman residual calculation: $\delta \leftarrow \hat{Q}(x; W(t)) - r - \gamma \hat{Q}(x'; W(t))$
 - 6: TD update: $\bar{W}(t+1) \leftarrow W(t) - \eta \delta \cdot \nabla_W \hat{Q}(x; W(t))$
 - 7: Projection: $W(t+1) \leftarrow \text{argmin}_{W \in S_B} \|W - \bar{W}(t+1)\|_2$
 - 8: Averaging: $\bar{W} \leftarrow \frac{t+1}{t+2} \cdot \bar{W} + \frac{1}{t+2} \cdot W(t+1)$
 - 9: **End For**
 - 10: **Output:** $\hat{Q}_{\text{out}}(\cdot) \leftarrow \hat{Q}(\cdot; \bar{W})$
-

To understand the intuition behind the global convergence of neural TD, note that for the TD update in (3.1), we have from (2.1) that

$$\begin{aligned} & \mathbb{E}_{(s,a,r,s',a') \sim \mu} [(\hat{Q}_\theta(s, a) - r(s, a) - \gamma \hat{Q}_\theta(s', a')) \cdot \nabla_\theta \hat{Q}_\theta(s, a)] \\ &= \mathbb{E}_{(s,a) \sim \mu} [(\hat{Q}_\theta(s, a) - \mathbb{E}[r(s, a) + \gamma \hat{Q}_\theta(s', a') \mid s' \sim \mathcal{P}(\cdot \mid s, a), a' \sim \pi(s')]) \cdot \nabla_\theta \hat{Q}_\theta(s, a)] \\ &= \mathbb{E}_{(s,a) \sim \mu} [\underbrace{(\hat{Q}_\theta(s, a) - \mathcal{T}^\pi \hat{Q}_\theta(s, a))}_{(i)} \cdot \underbrace{\nabla_\theta \hat{Q}_\theta(s, a)}_{(ii)}]. \end{aligned} \quad (3.3)$$

Here (i) is the Bellman residual at (s, a) , while (ii) is the gradient of the first term in (i). Although the TD update in (3.1) resembles the stochastic gradient descent step for minimizing a mean-squared error, it is not an unbiased stochastic gradient of any objective function. However, we show that the TD update yields a descent direction towards the global optimum of the MSPBE in (2.3). Moreover, as the neural network becomes wider, the function class $\mathcal{F}$ that $\Pi_{\mathcal{F}}$ projects onto in (2.3) becomes richer. Correspondingly, the MSPBE reduces to the MSBE in (2.2) as the projection becomes closer to identity, which implies the recovery of the desired Q-function Q^π such that $Q^\pi = \mathcal{T}^\pi Q^\pi$. See Section 4 for a more rigorous characterization.

4 Main Results

In Section 4.1, we characterize the global optimality of the stationary point attained by Algorithm 1 in terms of minimizing the MSPBE in (2.3) and its other properties. In Section 4.2, we establish the

nonasymptotic global rates of convergence of neural TD to the global optimum of the MSPBE when following the population semigradients in (3.3) and the stochastic semigradients in (3.1), respectively.

We use the subscript $\mathbb{E}_\mu[\cdot]$ to denote the expectation over the randomness of the tuple (s, a, r, s, a') (or its concise form (x, r, x')) conditional on all other randomness, e.g., the random initialization and the random current iterate. Meanwhile, we use the subscript $\mathbb{E}_{\text{init}, \mu}[\cdot]$ when we are taking the expectation over all randomness, including the random initialization.

4.1 Properties of Stationary Point

We consider the population version of the TD update in Line 6 of Algorithm 1,

$$\widetilde{W}(t+1) \leftarrow W(t) - \eta \cdot \mathbb{E}_\mu[\delta(x, r, x'; W(t)) \cdot \nabla_W \widehat{Q}(x; W(t))], \quad (4.1)$$

where μ is the stationary distribution and $\delta(x, r, x'; W(t)) = \widehat{Q}(x; W(t)) - r - \gamma \widehat{Q}(x'; W(t))$ is the Bellman residual at (x, r, x') . The stationary point $W^\dagger$ of (4.1) satisfies the following stationarity condition,

$$\mathbb{E}_\mu[\delta(x, r, x'; W^\dagger) \cdot \nabla_W \widehat{Q}(x; W^\dagger)]^\top (W - W^\dagger) \geq 0, \quad \text{for any } W \in S_B. \quad (4.2)$$

Also, note that

$$\widehat{Q}(x; W) = \frac{1}{\sqrt{m}} \sum_{r=1}^m b_r \sigma(W_r^\top x) = \frac{1}{\sqrt{m}} \sum_{r=1}^m b_r \mathbb{1}\{W_r^\top x > 0\} W_r^\top x$$

and $\nabla_{W_r} \widehat{Q}(x; W) = b_r \mathbb{1}\{W_r^\top x > 0\} x$ almost everywhere in $\mathbb{R}^{md}$. Meanwhile, recall that $S_B = \{W \in \mathbb{R}^{md} : \|W - W(0)\|_2 \leq B\}$. We define the function class

$$\mathcal{F}_{B,m}^\dagger = \left\{ \frac{1}{\sqrt{m}} \sum_{r=1}^m b_r \mathbb{1}\{(W_r^\dagger)^\top x > 0\} W_r^\top x : W \in S_B \right\}, \quad (4.3)$$

which consists of the local linearization of $\widehat{Q}(x; W)$ at $W = W^\dagger$. Then (4.2) takes the following equivalent form

$$\langle \widehat{Q}(\cdot; W^\dagger) - \mathcal{T}^\pi \widehat{Q}(\cdot; W^\dagger), f(\cdot) - \widehat{Q}(\cdot; W^\dagger) \rangle_\mu \geq 0, \quad \text{for any } f \in \mathcal{F}_{B,m}^\dagger, \quad (4.4)$$

which implies $\widehat{Q}(\cdot; W^\dagger) = \Pi_{\mathcal{F}_{B,m}^\dagger} \mathcal{T}^\pi \widehat{Q}(\cdot; W^\dagger)$ by the definition of the projection induced by $\langle \cdot, \cdot \rangle_\mu$.

By (2.3), $\widehat{Q}(\cdot; W^\dagger)$ is the global optimum of the MSPBE that corresponds to the projection onto $\mathcal{F}_{B,m}^\dagger$.

Intuitively, when using an overparametrized neural network with width $m \rightarrow \infty$, the average variation in each W_r diminishes to zero. Hence, roughly speaking, we have $\mathbb{1}\{W_r(t)^\top x > 0\} = \mathbb{1}\{W_r(0)^\top x > 0\}$ with high probability for any $t \in [T]$. As a result, the function class $\mathcal{F}_{B,m}^\dagger$ defined in (4.3) approximates

$$\mathcal{F}_{B,m} = \left\{ \frac{1}{\sqrt{m}} \sum_{r=1}^m b_r \mathbb{1}\{W_r(0)^\top x > 0\} W_r^\top x : W \in S_B \right\}. \quad (4.5)$$

In the sequel, we show that, to characterize the global convergence of Algorithm 1 with a sufficiently large m , it suffices to consider $\mathcal{F}_{B,m}$ in place of $\mathcal{F}_{B,m}^\dagger$, which simplifies the analysis, since the distribution of $W(0)$ is given. To this end, we define the approximate stationary point W^* with respect to the function class $\mathcal{F}_{B,m}$ defined in (4.5).

Definition 4.1 (Approximate Stationary Point W^*). If $W^* = (W_1^*, \dots, W_m^*) \in \mathbb{R}^{md}$ satisfies

$$\mathbb{E}_\mu[\delta_0(x, r, x'; W^*) \cdot \nabla_W \widehat{Q}_0(x; W^*)]^\top (W - W^*) \geq 0, \quad \text{for any } W \in S_B, \quad (4.6)$$

where we define

$$\widehat{Q}_0(x; W) = \frac{1}{\sqrt{m}} \sum_{r=1}^m b_r \mathbb{1}\{W_r(0)^\top x > 0\} W_r^\top x, \quad (4.7)$$

$$\delta_0(x, r, x'; W) = \widehat{Q}_0(x; W) - r - \gamma \widehat{Q}_0(x'; W), \quad (4.8)$$

then we say that W^* is an approximate stationary point of the population update in (4.1). Here W^* depends on the random initialization $b = (b_1, \dots, b_m)$ and $W(0) = (W_1(0), \dots, W_m(0))$.

The next lemma proves that such an approximate stationary point uniquely exists, since it is the fixed point of the operator $\Pi_{\mathcal{F}_{B,m}} \mathcal{T}^\pi$, which is a contraction in the ℓ_2 -norm associated with the stationary distribution μ .

Lemma 4.2 (Existence, Uniqueness, and Optimality of W^*). There exists a unique approximate stationary point W^* for any $b \in \mathbb{R}^m$ and $W(0) \in \mathbb{R}^{md}$. Also, $\hat{Q}_0(\cdot; W^*)$ is the global optimum of the MSPBE that corresponds to the projection onto $\mathcal{F}_{B,m}$ in (4.5).

Proof. See Appendix C.1 for a detailed proof. $\square$

4.2 Global Convergence

In this section, we establish the main results on the global convergence of neural TD in Algorithm 1. We first lay out the following regularity condition on the stationary distribution μ .

Assumption 4.3 (Regularity of Stationary Distribution μ). There exists a constant $c_0 > 0$ such that for any $\tau \geq 0$ and $w \sim N(0, I_d/d)$, it holds almost surely that

$$\mathbb{E}_\mu[\mathbb{1}\{|w^\top x| \leq \tau\} | w] \leq c_0 \cdot \tau / \|w\|_2. \quad (4.9)$$

Assumption 4.3 regularizes the density of μ in terms of the marginal distribution of x . In particular, it is straightforwardly implied when the density of μ in terms of state s is upper bounded.

Population Update: The next theorem establishes the nonasymptotic global rate of convergence of neural TD when it follows population semigradients. Recall that the approximate stationary point W^* and $\hat{Q}_0(\cdot; W^*)$ are defined in Definition 4.1. Also, B is the radius of the set of feasible W , which is defined in Algorithm 1, T is the number of iterations, γ is the discount factor, and m is the width of the neural network in (3.2).

Theorem 4.4 (Convergence of Population Update). We set $\eta = (1 - \gamma)/8$ in Algorithm 1 and replace the TD update in Line 6 by the population update in (4.1). Under Assumption 4.3, the output $\hat{Q}_{\text{out}}$ of Algorithm 1 satisfies

$$\mathbb{E}_{\text{init}, \mu}[(\hat{Q}_{\text{out}}(x) - \hat{Q}_0(x; W^*))^2] \leq \frac{16B^2}{(1 - \gamma)^2 T} + O(B^3 m^{-1/2} + B^{5/2} m^{-1/4}),$$

where the expectation is taken with respect to all randomness, including the random initialization and the stationary distribution μ .

Proof. The key to the proof of Theorem 4.4 is the one-point monotonicity of the population semigradient $\bar{g}(t)$, which is established through the local linearization $\hat{Q}_0(x; W)$ of $\hat{Q}(x; W)$. See Appendix D.5 for a detailed proof. $\square$

Stochastic Update: To further prove the global convergence of neural TD when it follows stochastic semigradients, we first establish an upper bound of their variance, which affects the choice of the stepsize η . For notational simplicity, we define the stochastic and population semigradients as

$$g(t) = \delta(x, r, x'; W(t)) \cdot \nabla_W \hat{Q}(x; W(t)), \quad \bar{g}(t) = \mathbb{E}_\mu[g(t)]. \quad (4.10)$$

Lemma 4.5 (Variance Bound). There exists $\sigma_g^2 = O(B^2)$ such that the variance of the stochastic semigradient is upper bounded as $\mathbb{E}_{\text{init}, \mu}[\|g(t) - \bar{g}(t)\|_2^2] \leq \sigma_g^2$ for any $t \in [T]$.

Proof. See Appendix C.2 for a detailed proof. $\square$

Based on Theorem 4.4 and Lemma 4.5, we establish the global convergence of neural TD in Algorithm 1.

Theorem 4.6 (Convergence of Stochastic Update). We set $\eta = \min\{(1 - \gamma)/8, 1/\sqrt{T}\}$ in Algorithm 1. Under Assumption 4.3, the output $\hat{Q}_{\text{out}}$ of Algorithm 1 satisfies

$$\mathbb{E}_{\text{init}, \mu}[(\hat{Q}_{\text{out}}(x) - \hat{Q}_0(x; W^*))^2] \leq \frac{16(B^2 + \sigma_g^2)}{(1 - \gamma)^2 \sqrt{T}} + O(B^3 m^{-1/2} + B^{5/2} m^{-1/4}).$$

Proof. See Appendix D.6 for a detailed proof. $\square$

As the width of the neural network $m \rightarrow \infty$, Lemma 4.2 implies that $\hat{Q}_0(\cdot; W^*)$ is the global optimum of the MSPBE in (2.3) with a richer function class $\mathcal{F}_{B,\infty}$ to project onto. In fact, the function class $\mathcal{F}_{B,\infty} - \hat{Q}(\cdot; W(0))$ is a subset of an RKHS with $\mathcal{H}$ -norm upper bounded by B . Here $\hat{Q}(\cdot; W(0))$ is defined in (3.2). See Appendix B.2 for a more detailed discussion on the representation power of $\mathcal{F}_{B,\infty}$. Therefore, if the desired Q-function $Q^\pi(\cdot)$ falls into $\mathcal{F}_{B,\infty}$, it is the global optimum of the MSPBE. In such a case, by Lemma 4.2 and Theorem 4.6, we approximately obtain $Q^\pi(\cdot) = \hat{Q}_0(\cdot; W^*)$ through $\hat{Q}_{\text{out}}(\cdot)$.

More generally, the following proposition quantifies the distance between $\hat{Q}_0(\cdot; W^*)$ and $Q^\pi(\cdot)$ in the case that $Q^\pi(\cdot)$ does not fall into the function class $\mathcal{F}_{B,m}$. In particular, it states that the ℓ_2 -norm distance $\|\hat{Q}_0(\cdot; W^*) - Q^\pi(\cdot)\|_\mu$ is upper bounded by the distance between $Q^\pi(\cdot)$ and $\mathcal{F}_{B,m}$.

Proposition 4.7 (Convergence of Stochastic Update to Q^π). It holds that $\|\hat{Q}_0(\cdot; W^*) - Q^\pi(\cdot)\|_\mu \leq (1 - \gamma)^{-1} \cdot \|\Pi_{\mathcal{F}_{B,m}} Q^\pi(\cdot) - Q^\pi(\cdot)\|_\mu$, which by Theorem 4.6 implies

$$\begin{aligned} \mathbb{E}_{\text{init}, \mu} [(\hat{Q}_{\text{out}}(x) - Q^\pi(x))^2] &\leq \frac{32(B^2 + \sigma_g^2)}{(1 - \gamma)^2 \sqrt{T}} + \frac{2\mathbb{E}_{\text{init}, \mu} [(\Pi_{\mathcal{F}_{B,m}} Q^\pi(x) - Q^\pi(x))^2]}{(1 - \gamma)^2} \\ &\quad + O(B^3 m^{-1/2} + B^{5/2} m^{-1/4}). \end{aligned}$$

Proof. See Appendix C.3 for a detailed proof. $\square$

Proposition 4.7 implies that if $Q^\pi(\cdot) \in \mathcal{F}_{B,\infty}$, then $\hat{Q}_{\text{out}}(\cdot) \rightarrow Q^\pi(\cdot)$ as $T, m \rightarrow \infty$. In other words, neural TD converges to the global optimum of the MSPBE in (2.3), or equivalently, the MSBE in (2.2), both of which have objective value zero.

5 Extension to Policy Optimization

With the Q-function learned by TD, policy iteration may be applied to learn the optimal policy. Alternatively, Q-learning more directly learns the optimal policy and its Q-function using temporal-difference update. Compared with TD, Q-learning aims to solve the projected Bellman optimality equation

$$Q = \Pi_{\mathcal{F}} \mathcal{T} Q, \quad \text{with } \mathcal{T} Q(s, a) = \mathbb{E}[r(s, a) + \gamma \max_{a' \in \mathcal{A}} Q(s', a') \mid s' \sim \mathcal{P}(\cdot \mid s, a)], \quad (5.1)$$

which replaces the Bellman evaluation operator $\mathcal{T}^\pi$ in (2.3) with the Bellman optimality operator $\mathcal{T}$. When $\Pi_{\mathcal{F}}$ is identity, the fixed-point solution to (5.1) is the Q-function $Q^{\pi^*}(s, a)$ of the optimal policy π^* , which maximizes the expected total reward [53, 50]. Compared with TD, the max operator in $\mathcal{T}$ makes the analysis more challenging and hence requires stronger regularity conditions. In the following, we first introduce neural Q-learning and then establish its global convergence. Finally, we discuss the corresponding implication for policy gradient algorithms.

5.1 Neural Q-Learning

In parallel with (3.1), we update the parameter θ of the optimal Q-function by

$$\theta' \leftarrow \theta - \eta \cdot (\hat{Q}_\theta(s, a) - r(s, a) - \gamma \max_{a' \in \mathcal{A}} \hat{Q}_\theta(s', a')) \cdot \nabla_\theta \hat{Q}_\theta(s, a), \quad (5.2)$$

where the tuple (s, a, r, s') is sampled from the stationary distribution μ_{exp} of an exploration policy π_{exp} in an independent and identically distributed manner. We present the detailed neural Q-learning algorithm in Algorithm 2 and defer it to Appendix E due to space limitations. Similar to Definition 4.1, we define the approximate stationary point W^* of Algorithm 2 by

$$\mathbb{E}_{\mu_{\text{exp}}} [\delta_0(x, r, x'; W^*) \cdot \nabla_W \hat{Q}_0(x; W^*)]^\top (W - W^*) \geq 0, \quad \text{for any } W \in S_B, \quad (5.3)$$

where the Bellman residual is now $\delta_0(x, r, x'; W) = \hat{Q}_0(x; W) - r - \gamma \max_{a' \in \mathcal{A}} \hat{Q}_0(s', a'; W)$. Following the same analysis of neural TD in Lemma 4.2, we have that $\hat{Q}_0(\cdot; W^*)$ is the unique fixed-point solution to the projected Bellman optimality equation $Q = \Pi_{\mathcal{F}_{B,m}} \mathcal{T} Q$, where the function class $\mathcal{F}_{B,m}$ is defined in (4.5).

5.2 Global Convergence

To establish the global convergence of neural Q-learning, we lay out an extra regularity condition on the exploration policy π_{exp} , which is not required by neural TD. Such a regularity condition ensures that $x' = (s', a')$ with the greedy action a' in Line 4 of Algorithm 2 follows a similar distribution to that of $x = (s, a)$, which is the stationary distribution μ_{exp} of the exploration policy π_{exp} . Recall that $\widehat{Q}_0(x; W)$ is defined in (4.7) and γ is the discount factor.

Assumption 5.1 (Regularity of Exploration Policy π_{exp}). There exists a constant $\nu > 0$ such that for any $W_1, W_2 \in S_B$, it holds that

$$\mathbb{E}_{x \sim \mu_{\text{exp}}} [(\widehat{Q}_0(x; W_1) - \widehat{Q}_0(x; W_2))^2] \geq (\gamma + \nu)^2 \cdot \mathbb{E}_{s \sim \mu_{\text{exp}}} [(\widehat{Q}_0^\#(s; W_1) - \widehat{Q}_0^\#(s; W_2))^2], \quad (5.4)$$

where $\widehat{Q}_0^\#(s; W) = \max_{a \in \mathcal{A}} \widehat{Q}_0(s, a; W)$.

We remark that [38, 60] establish the global convergence of linear Q-learning based on an assumption that implies (5.4). Although Assumption 5.1 is strong, we are not aware of any weaker regularity condition in the literature, even for linear Q-learning. As our focus is to go beyond linear Q-learning to analyze neural Q-learning, we do not attempt to weaken such a regularity condition in this paper.

The following regularity condition on μ_{exp} mirrors Assumption 4.3, but additionally accounts for the max operator in the Bellman optimality operator.

Assumption 5.2 (Regularity of Stationary Distribution μ_{exp}). There exists a constant $c_3 > 0$ such that for any $\tau \geq 0$ and $w \sim N(0, I_d/d)$, it holds almost surely that

$$\mathbb{E}_{s \sim \mu_{\text{exp}}} \left[\max_{a \in \mathcal{A}} \mathbb{1}\{|w^\top(s, a)| \leq \tau\} \mid w \right] \leq c_3 \cdot \tau / \|w\|_2. \quad (5.5)$$

In parallel with Theorem 4.6, the following theorem establishes the global convergence of neural Q-learning in Algorithm 2.

Theorem 5.3 (Convergence of Stochastic Update). We set η to be of order $T^{-1/2}$ in Algorithm 2. Under Assumptions 5.1 and 5.2, the output $\widehat{Q}_{\text{out}}$ of Algorithm 2 satisfies

$$\mathbb{E}_{\text{init}, \mu_{\text{exp}}} [(\widehat{Q}_{\text{out}}(x) - \widehat{Q}_0(x; W^*))^2] = O(B^2 T^{-1/2} + B^3 m^{-1/2} + B^{5/2} m^{-1/4}).$$

Proof. See Appendix E.1 for a detailed proof. $\square$

Corresponding to Proposition 4.7, Theorem 5.3 also implies the convergence to $Q^{\pi^*}(s, a)$, which is omitted due to space limitations.

5.3 Implication for Policy Gradient

Theorem 5.3 can be further extended to handle neural soft Q-learning, where the max operator in the Bellman optimality operator is replaced by a more general softmax operator [23, 40]. By exploiting the equivalence between soft Q-learning and policy gradient algorithms [46, 24], we establish the global convergence of a variant of the policy gradient algorithm. Due to space limitations, we defer the discussion to Appendix F.

6 Conclusions

In this paper we prove that neural TD converges at a sublinear rate to the global optimum of the MSPBE for policy evaluation. In particular, we show how such global convergence is enabled by the overparametrization of neural networks. Moreover, we extend the convergence result to policy optimization, including (soft) Q-learning and policy gradient. Our results shed new light on the theoretical understanding of RL with neural networks, which is widely employed in practice.

References

- [1] Allen-Zhu, Z., Li, Y. and Liang, Y. (2018). Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*.
- [2] Amiranashvili, A., Dosovitskiy, A., Koltun, V. and Brox, T. (2018). TD or not TD: Analyzing the role of temporal differencing in deep reinforcement learning. *arXiv preprint arXiv:1806.01175*.
- [3] Arora, S., Du, S. S., Hu, W., Li, Z. and Wang, R. (2019). Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584*.
- [4] Baird, L. (1995). Residual algorithms: Reinforcement learning with function approximation. In *International Conference on Machine Learning*.
- [5] Bertsekas, D. P. (2019). Feature-based aggregation and deep reinforcement learning: A survey and some new implementations. *IEEE/CAA Journal of Automatica Sinica*, **6** 1–31.
- [6] Bhandari, J., Russo, D. and Singal, R. (2018). A finite time analysis of temporal difference learning with linear function approximation. *arXiv preprint arXiv:1806.02450*.
- [7] Bhatnagar, S., Precup, D., Silver, D., Sutton, R. S., Maei, H. R. and Szepesvári, C. (2009). Convergent temporal-difference learning with arbitrary smooth function approximation. In *Advances in Neural Information Processing Systems*.
- [8] Borkar, V. S. (2009). *Stochastic Approximation: A Dynamical Systems Viewpoint*, vol. 48. Springer.
- [9] Borkar, V. S. and Meyn, S. P. (2000). The ODE method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal on Control and Optimization*, **38** 447–469.
- [10] Boyan, J. A. (1999). Least-squares temporal difference learning. In *International Conference on Machine Learning*.
- [11] Boyan, J. A. and Moore, A. W. (1995). Generalization in reinforcement learning: Safely approximating the value function. In *Advances in Neural Information Processing Systems*.
- [12] Bradtke, S. J. and Barto, A. G. (1996). Linear least-squares algorithms for temporal difference learning. *Machine Learning*, **22** 33–57.
- [13] Chizat, L. and Bach, F. (2018). A note on lazy training in supervised differentiable programming. *arXiv preprint arXiv:1812.07956*.
- [14] Dalal, G., Szörényi, B., Thoppe, G. and Mannor, S. (2018). Finite sample analyses for TD(0) with function approximation. In *AAAI Conference on Artificial Intelligence*.
- [15] Daniely, A. (2017). SGD learns the conjugate kernel class of the network. In *Advances in Neural Information Processing Systems*.
- [16] Dann, C., Neumann, G. and Peters, J. (2014). Policy evaluation with temporal differences: A survey and comparison. *Journal of Machine Learning Research*, **15** 809–883.
- [17] Du, S. S., Chen, J., Li, L., Xiao, L. and Zhou, D. (2017). Stochastic variance reduction methods for policy evaluation. In *International Conference on Machine Learning*.
- [18] Duan, Y., Chen, X., Houthoofd, R., Schulman, J. and Abbeel, P. (2016). Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*.
- [19] Facchinei, F. and Pang, J.-S. (2007). *Finite-Dimensional Variational Inequalities and Complementarity Problems*. Springer Science & Business Media.

- [20] Fan, J., Ma, C. and Zhong, Y. (2019). A selective overview of deep learning. *arXiv preprint arXiv:1904.05526*.
- [21] Geist, M. and Pietquin, O. (2013). Algorithmic survey of parametric value function approximation. *IEEE Transactions on Neural Networks and Learning Systems*, **24** 845–867.
- [22] Ghavamzadeh, M., Lazaric, A., Maillard, O. and Munos, R. (2010). LSTD with random projections. In *Advances in Neural Information Processing Systems*.
- [23] Haarnoja, T., Tang, H., Abbeel, P. and Levine, S. (2017). Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning*.
- [24] Haarnoja, T., Zhou, A., Abbeel, P. and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*.
- [25] Harker, P. T. and Pang, J.-S. (1990). Finite-dimensional variational inequality and nonlinear complementarity problems: a survey of theory, algorithms and applications. *Mathematical Programming*, **48** 161–220.
- [26] Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D. and Meger, D. (2018). Deep reinforcement learning that matters. In *AAAI Conference on Artificial Intelligence*.
- [27] Hofmann, T., Schölkopf, B. and Smola, A. J. (2008). Kernel methods in machine learning. *Annals of Statistics* 1171–1220.
- [28] Jaakkola, T., Jordan, M. I. and Singh, S. P. (1994). Convergence of stochastic iterative dynamic programming algorithms. In *Advances in Neural Information Processing Systems*.
- [29] Jacot, A., Gabriel, F. and Hongler, C. (2018). Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*.
- [30] Jain, P. and Kar, P. (2017). Non-convex optimization for machine learning. *Foundations and Trends® in Machine Learning*, **10** 142–336.
- [31] Konda, V. R. and Tsitsiklis, J. N. (2000). Actor-critic algorithms. In *Advances in Neural Information Processing Systems*.
- [32] Kushner, H. and Yin, G. G. (2003). *Stochastic Approximation and Recursive Algorithms and Applications*. Springer Science & Business Media.
- [33] Lakshminarayanan, C. and Szepesvari, C. (2018). Linear stochastic approximation: How far does constant step-size and iterate averaging go? In *International Conference on Artificial Intelligence and Statistics*.
- [34] Lazaric, A., Ghavamzadeh, M. and Munos, R. (2010). Finite-sample analysis of LSTD. In *International Conference on Machine Learning*.
- [35] Lee, J., Xiao, L., Schoenholz, S. S., Bahri, Y., Sohl-Dickstein, J. and Pennington, J. (2019). Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720*.
- [36] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D. and Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.
- [37] Liu, B., Liu, J., Ghavamzadeh, M., Mahadevan, S. and Petrik, M. (2015). Finite-sample analysis of proximal gradient TD algorithms. In *Conference on Uncertainty in Artificial Intelligence*.
- [38] Melo, F. S., Meyn, S. P. and Ribeiro, M. I. (2008). An analysis of reinforcement learning with function approximation. In *International Conference on Machine Learning*.
- [39] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D. and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*.

- [40] Neu, G., Jonsson, A. and Gómez, V. (2017). A unified view of entropy-regularized markov decision processes. *arXiv preprint arXiv:1705.07798*.
- [41] Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y. and Srebro, N. (2018). Towards understanding the role of over-parametrization in generalization of neural networks. *arXiv preprint arXiv:1805.12076*.
- [42] Pfau, D. and Vinyals, O. (2016). Connecting generative adversarial networks and actor-critic methods. *arXiv preprint arXiv:1610.01945*.
- [43] Rahimi, A. and Recht, B. (2008). Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*.
- [44] Rahimi, A. and Recht, B. (2008). Uniform approximation of functions with random bases. In *Annual Allerton Conference on Communication, Control, and Computing*.
- [45] Scherrer, B. (2010). Should one compute the temporal difference fix point or minimize the bellman residual? the unified oblique projection view. In *International Conference on Machine Learning*.
- [46] Schulman, J., Chen, X. and Abbeel, P. (2017). Equivalence between policy gradients and soft Q-learning. *arXiv preprint arXiv:1704.06440*.
- [47] Schulman, J., Levine, S., Abbeel, P., Jordan, M. and Moritz, P. (2015). Trust region policy optimization. In *International Conference on Machine Learning*.
- [48] Srikant, R. and Ying, L. (2019). Finite-time error bounds for linear stochastic approximation and TD learning. *arXiv preprint arXiv:1902.00923*.
- [49] Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, 3 9–44.
- [50] Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT press.
- [51] Sutton, R. S., Maei, H. R., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, C. and Wiewiora, E. (2009). Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *International Conference on Machine Learning*.
- [52] Sutton, R. S., Maei, H. R. and Szepesvári, C. (2009). A convergent $o(n)$ temporal-difference algorithm for off-policy learning with linear function approximation. In *Advances in Neural Information Processing Systems*.
- [53] Szepesvári, C. (2010). Algorithms for reinforcement learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 4 1–103.
- [54] Touati, A., Bacon, P.-L., Precup, D. and Vincent, P. (2017). Convergent tree-backup and retrace with function approximation. *arXiv preprint arXiv:1705.09322*.
- [55] Tsitsiklis, J. N. and Van Roy, B. (1997). Analysis of temporal-difference learning with function approximation. In *Advances in Neural Information Processing Systems*.
- [56] Tu, S. and Recht, B. (2017). Least-squares temporal difference learning for the linear quadratic regulator. *arXiv preprint arXiv:1712.08642*.
- [57] Wang, Y., Chen, W., Liu, Y., Ma, Z.-M. and Liu, T.-Y. (2017). Finite sample analysis of the GTD policy evaluation algorithms in Markov setting. In *Advances in Neural Information Processing Systems*.
- [58] Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8 229–256.
- [59] Zhang, C., Bengio, S., Hardt, M., Recht, B. and Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*.
- [60] Zou, S., Xu, T. and Liang, Y. (2019). Finite-sample analysis for SARSA and Q-learning with linear function approximation. *arXiv preprint arXiv:1902.02234*.