Experimental Verification of the Convergence of Temporal Difference Learning

Lun, Alan

Supervisor: Prof. WAI, Hoi To

1 Introduction

Temporal difference (TD) learning refers to a class of model-free reinforcement learning methods that aim to learn a (action-)value function of a given policy. TD learning has broad applications and plays an important role in deep reinforcement learning, and it has drawn attention of many researchers.

Due to the success of deep reinforcement learning, many works have been done to provide some theoretical guarantees for neural TD learning. For example, [1] proves that neural TD converges to the global optimum of MSPBE. In this project we have experimentally verified the theory and claims in [1] and also experimented the gradient TD methods which converge robustly.

2 Background

In section 2.1, we will discuss the environment for testing TD learning methods. In section 2.2, we formulate the objective function to optimize. In section 2.3, we briefly review different types of TD learning.

2.1 Garnet problem

A garnet problem is a randomly constructed finite Markov decision process. It is characterized by four parameters and is denoted by $garnet(N_s, N_a, B, \sigma)$. The parameter N_s is the number of states, N_a is the number of actions, B is the branching factor, and σ is the variance of each transition reward. For simplicity, we generate the rewards of each state from a uniform distribution $Unif(0, 1)$, and the transition matrix P for each action is composed of B

non-zero terms.

In each iteration, the agent will take an action $a \in A$(action space) according to the given policy. Upon taking the action, the agent enters the next state $s' \in S$(state space) according to the transition matrix $P(s'|s,a)$ and receives a random reward $r(s,a)$ from the environment. Our goal is to train an agent to learn the value function $V_\pi(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a \sim \pi(s)]$, where the given policy $\pi : S \to A$.

2.2 Optimization formulation

First, we define the Bellman error at state s,

$$\delta_w(s) = \mathbb{E}_\pi[r(s,a) + \gamma V_w(s') - V_w(s) \mid s' \sim P(.|s,a), a \sim \pi(s)] \tag{1}$$

where value function is parameterized as V_w with parameter w.

The TD learning methods aim to minimize the mean-squared Bellman error (MSBE),

$$MSBE(w) = ||\delta_w||^2 \tag{2}$$

However, it is not possible in general to reduce the MSBE to zero, since the function approximator cannot learn the exact $V_\pi(s)$. Instead, we focus on minimizing the projected mean-squared Bellman error (MSPBE),

$$MSPBE(w) = ||\Pi\delta_w||^2 \tag{3}$$

Here Π is a projection onto a class of functions which can be learned by the function approximator. Therefore, there always exists an approximate value function with zero MSPBE [2].

2.3 Temporal difference learning

For simplicity, we will only focus on TD(0) and use stochastic semigradient descent in following TD methods.

2.3.1 Linear TD

In order to optimize the MSBE, linear TD updates the parameter by,

$$w' \leftarrow w - \alpha(V_w(s) - r - \gamma V_w(s'))\nabla_w V_w(s) \tag{4}$$

which equals to,

$$w' \leftarrow w - \alpha(w^T\phi(s) - r - \gamma w^T\phi(s'))\phi(s) \tag{5}$$

Here $V_w(s)$ is a linear approximation function of the parameter w and ϕ (s) is a real-valued feature vector corresponding to every state s.

2.3.2 Neural TD

We refer to neural TD to the algorithm that is mentioned in [1]. Similar to linear TD, neural TD updates the parameter in the same way as in (4), except that $V_w(s)$ is replaced by a two-layer neural network,

$$V_w(s) := \frac{1}{\sqrt{m}} \sum_{r=1}^{m} b_r ReLU(w^T s) \tag{6}$$

where $w \in \Re^{md}$ and $b_r \in \Re^m$. The parameters are initialized as $b_r \sim Unif(-1, 1)$ and $w \sim N(0, I_d/d)$.

After the TD update, neural TD projects the w onto the set S_B by,

$$w' \leftarrow argmin_{W \in S_B} ||W - w'||_2 \tag{7}$$

$$S_B = \{W \in \Re^{md} : ||W - W(0)||_2 \leq B\}$$

where d is the number of states, m is the dimension of weights, $W(0)$ is the initial weight and $B > 0$.

2.3.3 Off-policy TD

Both the linear TD and neural TD we have mentioned before are on-policy methods. In most cases, however, the target policy and behavior policy are not the same. In order to learn the value function under such condition, most off-policy methods utilize a technique called *importance sampling*. We define the importance sampling ratio as

$$p_t := \frac{\pi(a_t|s_t)}{b(a_t|s_t)} \tag{8}$$

where π is the target policy and b is the behavior policy.

The off-policy version for liner TD is

$$w' \leftarrow w - \alpha \cdot p \cdot (V_w(s) - r - \gamma V_w(s')) \nabla_w V_w(s) \tag{9}$$

The off-policy version for neural TD follows the exact same way by replacing (4) with (9).

2.3.4 Gradient TD

One prominent consequence of off-policy methods are that they may not necessarily converge due to the difference between target policy and behavior policy [2]. To resolve the non-convergent issue, gradient TD methods

are introduced. There are many variation of GTD methods, such as GTD, GTD2 and TDC. Here we will only briefly talk about the neural version of GTD

We call it as neural GTD. It aims to minimize the approximated MSBE,

$$\min_{w \in S_B} \max_{g \in S_B} J(w, g) := \mathbb{E}[V_g(s)\bar{\delta}_w(s) - \frac{1}{2}V_g^2 + \frac{v}{2}V_w(s)^2] \tag{10}$$

The update steps are

$$w' \leftarrow P_{S_B}\{w - \beta V_g(s)\nabla\bar{\delta}_w - \beta v V_w(s)\nabla V_w(s)\} \tag{11}$$

$$g' \leftarrow P_{S_B}\{g + \beta\bar{\delta}_w\nabla V_g(s) - \beta V_g(s)\nabla V_g(s)\} \tag{12}$$

$$\bar{\delta}_w := p[V_w(s) - \gamma V_w(s') - r] \tag{13}$$

Here P_{S_B} is a projection onto the set S_B in (7) and β is the learning rate.

3 Main Results

3.1 Verification of the theory in [1]

[1] proved 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. They also stated that overparametrization is actually beneficial for minimizing MSPBE in the presence of bias, nonconvexity, and even divergence.

In order to verify the theory experimentally, we have implemented the neural TD and tested its convergence under the garnet environment. Although in our experiment the neural TD learns the value function instead of action-value function stated in [1], the theory should also applies and such change of setting will not lead to loss of generality. As we have all the information about the garnet environment, we can directly calculate the value function $V_B(s)$ by using the Bellman equation. Therefore, we choose the Mean-Square Bellman Error between $V_B(s)$ and $V_w(s)$ as the metric to measure the error of neural TD. In the experiment, we use a Garnet of $|S| = 500$ and $|A| = 5$ as the testing environment.

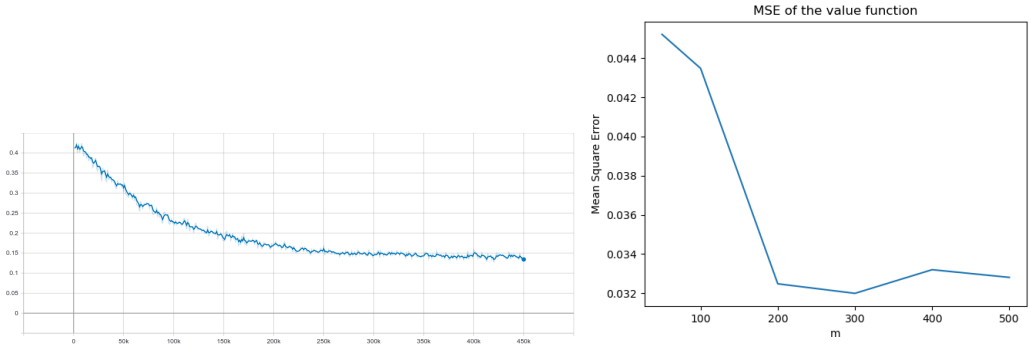

Figure 1: Left figure is training loss per 1000 iterations(m=350). Right is the MSBE between $V_B(s)$ and $V_w(s)$

In figure(1)(left), we can see that the training loss is decreasing in each iteration and it is clearly converging. Besides, as the dimension m increases, the MSE between $V_B(s)$ and $V_w(s)$ drops, which suggests that overparametrization may be indeed beneficial for minimizing the error.

3.2 Addresssing the divergence of off-policy methods

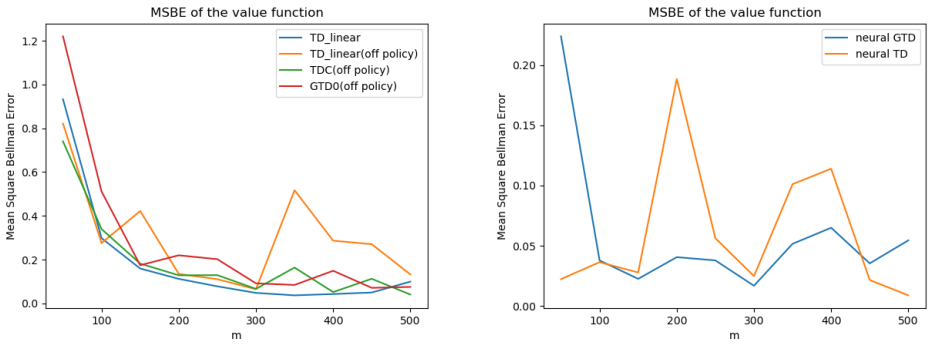

Figure 2: Left figure is the comparison between different types of non-neural methods. Right figure is the MSBE between neural TD and neural GTD

From figure(2), we can see that both off-policy linear TD and off-policy neural TD diverge, while GTD, TDC and neural GTD still converge. By changing the objective function, GTD methods converge robustly.

4 Conclusion

In this project we have experimentally verified that neural TD converges to the global optimum of MSPBE and shown overparametrization is may

indeed beneficial for minimizing MSPBE. Moreover, we have also experimented the gradient TD methods which have robust convergence properties even under off-policy training and nonlinear function approximation.

References

[1] 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 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 11315–11326.

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