Neurocomputing 458 (2021) 327–335

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Variance aware reward smoothing for deep reinforcement learning

Yunlong Dong [a], Shengjun Zhang [b], Xing Liu [c], Yu Zhang [a], Tan Shen [a,*]

[a] School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, China
[b] Department of Electrical Engineering, University of North Texas, United States
[c] School of Information Science and Engineering, East China University of Science and Technology, China

ARTICLE INFO

Article history:
Received 20 April 2021
Revised 29 May 2021
Accepted 6 June 2021
Available online 8 June 2021
Communicated by Zidong Wang

Keywords:
Deep reinforcement learning
Rewards drop
Variance reduction
Rewards smoothing

ABSTRACT

A Reinforcement Learning (RL) agent interacts with the environment to learn a policy with high accumulated rewards through attempts and failures. However, RL suffers from its own trial-and-error learning nature, which results in an unstable learning process. In this paper, we investigate a common phenomenon called rewards drop at the late-stage RL training session, where the rewards trajectory oscillates dramatically. In order to solve such a problem, we propose a novel rewards shaping technique named Variance Aware Rewards Smoothing (VAR). We show that the proposed method reduces the variance of rewards and mitigates the rewards drop problem without changing the formulation of the value function. Furthermore, the theoretical analysis of convergence of VAR is provided, which is derived from the γ-contraction operator and the fixed point attribute of the value function. Finally, the theoretical results are illustrated by extensive results on various benchmarks and advanced algorithms across different random seeds to demonstrate the effectiveness and the compatibility of VAR.

1. Introduction

In Reinforcement Learning (RL), an agent learns from and interacts with the environment via a great amount of attempts and failures, which can be labeled as a trial-and-error learning process. Typically, an RL agent takes actions regarding to states, then gets rewards from the environment and the states will be updated corresponding to the agent's actions back and forth. The ultimate goal of an RL agent is to find an optimal policy that maximizes the accumulated rewards. Reinforcement learning is able to handle extremely complex and difficult scenarios by formulating them into a RL framework. In the recent decades, with the growth of computational ability, RL has been successfully applied to various tasks in different fields, for instance, game playing [23,19,15], robotics [1], continuous control [16,32], autonomous driving [10], and quantitative analysis [34,35,37,36], just to name a few.

In the literature, most common approaches to solve RL problems utilize function approximations to fit value functions V and policy π. There are two main approaches to solve RL problems: methods based on value functions, for instance Q-learning, and Deep Q-Networks (DQN) [11], and methods based on policy search and policy gradient, such as REINFORCE [30], PPO [22] and DDPG [24]. Even though a RL agent can achieve super-human performance in many tasks, RL training process can be unstable and sometimes even divergent.

1.1. Related works

The well-known "deadly triad" (combing the off-policy, bootstrapping and function approximations) [26] was blamed for the instabilities and divergence during RL training process. Off-policy learning methods may not update the value fucntions often enough to avoid divergence. Bootstrapping adn function approximations can lead to the divergence when estimating the Q-value using the action based on the next state. However, such a "deadly triad" could be mitigated via modulating the divergence with multi-step returns and different prioritizations in empirical settings [27]. Prior to the work of [27], the trust region method [20] and the proximal clipping approach [22] were proposed to relieve the divergence during the training. The idea behind both of the methods is to directly restrict the scenario where the new policy is dramatically different from the old policy, which ensures stabilities. Recently, several works have been proposed in the literature to address unstable RL training problems and make RL algorithms more sample-efficient. In order to achieve efficient exploration, max-entropy regularization [8] were proposed. Instead of directly constraining the update of policy, [18] attenuated rapid policy changes to rarely sampled actions.

* Corresponding author.
E-mail address: tans@hust.edu.cn (T. Shen).

https://doi.org/10.1016/j.neucom.2021.06.014
0925-2312/

One category of the approaches to solve unstable RL training process is to avoid overestimation of value function approximation [12]. Double Q-learning [28] was proposed to be generalized in large-scale function approximation and reduce the overestimations. The authors in [6] limited the overestimation of function approximation errors via choosing the minimum value between twin critic networks. However, the minimum over twin critics would introduce the underestimation of value function, so in [31], the triplet averaging over three critic networks was proposed to reduce the estimation bias. The authors in [8] introduced the maximum entropy RL framework to achieve stability. Moreover, the Boltzmann softmax operator for value function estimation was applied in continuous control to improve the overestimation and underestimation bias [14]. Another interesting work was to utilize representation learning to formally guarantee stability of value function learning on the combination of off-policy learning and Bellman updates [7].

Another category is to modify the rewards functions, which has great influence on RL [26]. There are several works in this category have been proposed to stablize the training process. Back to 1990, the reward shaping technique was proposed [13] to provide a scheme to modify the reward without changing the optimality but guarantee the convergence of the training process. Recently, based on the Lyapunov stability theory from control system community, the authors in [4] established a theoretical result on choosing a potential function to modify the reward function, which guaranteed the convergence of RL trainings. Such a method was tested and verified in robotics applications [38] on different state-of-the-art deep RL algorithms such as PPO and DDPG. Additionally, the averaging estimation method [17] was utilized to deal with noisy rewards and reduce the variance of the Bellman estimation [2] to ensure the learning process can converge. The automatic selections of auxiliary rewards technique was proposed and shown that can boost up the learning rate and effectiveness in different learning stages [5].

By the nature of reinforcement learning, RL algorithms are vulnerable since the real environment has lots of complex random processes that only can be learnt by trials but cannot be modeled explicitly. Therefore, it is difficult to accurately select the optimal training model. Even though the works aforementioned can assure the whole training process stable, they still suffer the rewards drop behavior at the late-stage training session, where the return rewards during the training would suddenly drops extremely and vibrates at a certain range even with well-tuned hyperparameters and previous given random seeds. In this paper, we focus on the statistical property of reward trajectory and aim to mitigate the rewards drop phenomenon during the late-stage deep reinforcement learning process.

1.2. Contributions

This paper proposes a novel simple universal variance reduced rewards function smoothing technique named VAR for deep reinforcement learning, which can promote more stable training process during RL. In real-word application, training a stable RL agent can a lot of time and cost. We highlight the main contributions of the paper as follows:

1. We investigate the variance of reward trajectory in the late-stage RL training behavior and propose the VAR method to deal with it.
2. We provide the theoretical proof of the proposed method on variance reduction of rewards. Unlike the state-of-the-art algorithms focusing on gradient variance reduction [9] and etc., the proposed method focuses on reward itself directly, so the VAR method is able to generalized to standard RL algorithms directly and easily.
3. We show that value function utilizing the proposed method converges via γ-contraction operator and its fixed-point attribute.
4. We conduct extensive experimental studies on multiple popular benchmarks and advanced algorithms. The results empirically demonstrate that our methods avoid rewards drop phenomenon and achieve better results compared with existing state-of-the-art algorithms.

The rest of this paper is organized as follows. In Section 2, we provide some preliminaries and notations that will be used in this paper. In Section 3, we describe rewards drop phenomenon with an intuitive experimental example on a simple environment `Pendumlum-v0` in OpenAI gym [3]. In Section 4, we propose the novel variance reduced rewards function smoothing technique and provide the theoretical variance analysis of the proposed method in Section 4.1. The convergence results of the value function utilizing our method is given in Section 4.2. Numerical results and experiments on various benchmarks are given in Section 5. Finally, Section 6 concludes and remarks this paper.

2. Preliminaries

In this section, we present some preliminaries and notations needed to present the proposed method in this paper. Typically, RL [26] can be modeled and treated as a Markov Decision Process (MDP), which is formulated by a tuple $M \triangleq (\mathcal{S}, \mathcal{A}, T, R, \gamma)$. $\mathcal{S}$ denotes the state set, $\mathcal{A}$ denotes the action set. $s \in \mathcal{S}$ presents a single state, and $a \in \mathcal{A}$ is a single action. $T(s'|s,a)$ represents the transition probability to get from state s to state s' with action a. $R(s,a,s')$ defines the reward function and γ indicates the discount rate for the onward rewards. $\pi(a|s)$ denotes a policy, a function mapping from $\mathcal{S}$ to $\mathcal{A}$. Additionally, suppose X is a random variable, then $\mathbb{E}[X]$ denotes the expectation of X, and $\mathrm{var}[X]$ represents the variance of X. The purpose of an RL agent is to find the optimal policy π^* which maximizes the accumulated rewards $\sum_{t=0}^{\infty}\gamma^t R_t(s_t, a_t, s_{t+1})$. The value function V of π is defined by

$$V_\pi(s) \triangleq \mathbb{E}_{a_t\sim\pi(a_t|s_t)}\left[\gamma^t R_t(s_t,a_t)|s_0 = s\right]. \tag{1}$$

The action value function Q of π is defined by

$$Q_\pi(s,a) \triangleq \mathbb{E}_{a_t\sim\pi(a_t|s_t)}\left[\gamma^t R_t(s_t,a_t)|s_0 = s, a_0 = a\right]. \tag{2}$$

The advantage function A of π is defined by

$$A_\pi(s,a) \triangleq Q_\pi(s,a) - V_\pi(s). \tag{3}$$

Recent RL algorithms [6,8,14] adopted Actor-Critic architecture [26] in their training. The "Critic" applied deep neural networks with parameters θ to approximate the value function by minimizing the value loss

$$\mathcal{L}_\theta = \mathbb{E}_{t\sim\mathcal{D}}\|G_t - V_\pi(s_t;\theta)\|_2^2, \tag{4}$$

where $t \sim \mathcal{D}$ denotes sampling over the experience $\mathcal{D}$ of interacting with the environments, $G_t = R_t + \gamma V_\pi(s_{t+1})$ by using the TD-target [25]. The "Actor" utilized another deep neural networks to approximate a policy $\pi(a|s;\phi)$ parameterized by ϕ, where the policy gradient is used to update ϕ by minimizing the policy loss:

$$\mathcal{L}_\phi = \mathbb{E}_{t\sim\mathcal{D}}\left[\log\pi(a_t|s_t;\phi)\widehat{A}_t\right], \tag{5}$$

where $\widehat{A}_t$ represents the estimation of the advantage function A_t [21]. During training session, the value loss and policy loss are updated alternately through policy evaluation and policy improvement to find the optimal policy π^*.

3. Rewards drop phenomenon

The rewards drop is a common phenomenon where the returned rewards oscillates dramatically from a higher value to a lower one during late-stage training. Most of the widely-used Deep RL algorithms suffer this problem during their trainings, for example TD3 [6], SD3 [14] and etc.

To visualize the rewards drop problem, we give a simple intuitive example on `Pendumlum-v0` from OpenAI gym [3] when applied TD3 algorithm. The goal of `Pendumlum-v0` is to swing up a pendumlum and holds it vertically as long as possible. In the `Pendumlum-v0` environment[1], suppose the angular displacement measured from the equilibrium position is α, then three states will be observed as $\cos(\alpha) \in [-1.0, 1.0], \sin(\alpha) \in [-1.0, 1.0]$, and $\ddot{\alpha} \in [-8.0, 8.0]$, the action is the joint effort $j \in [-2.0, 2.0]$. The precise equation for the reward in a single step is given as $-\left(\alpha^2 + 0.1\ddot{\alpha}^2 + 0.001j^2\right) \in [-16.2736044, 0]$. In this illustrative example, we set 200 steps for 50 episodes and plot the rewards trajectory. We applied the advanced state-of-the-art Deep RL algorithm TD3 to solve this task and recorded the rewards evolution for 50 episodes shown in Fig. 1. For demonstration purpose only, the training log was not smoothed. We can clearly see the rewards drop phenomenon happened at Episode 34, where the rewards value dropped from 0 (labeled as "good") to −500 (labeled as "bad"). The shaded area visualized the oscillation range in the late-stage training episodes.

To further investigate the undergoing details of rewards drop phenomenon, we record the "Good" and the "Bad" rewards trajectory from Episode 33 and 34 seperately, shown in Fig. 2. It is surprised to see that the "Bad" rewards trajectory is volatile compared to the "Good" one. This observation motivated us to investigate the rewards drop phenomenon deeply and find a method to avoid it in RL trainings.

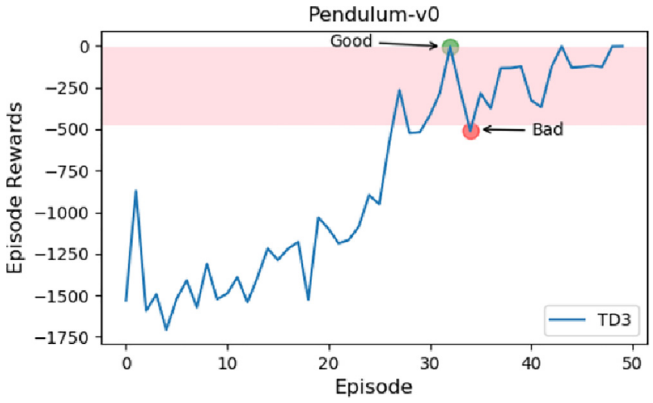

Fig. 1. The training procedure (not smoothed) of TD3 trained on Pendulum-v0 environments is shown. The "Good" and the "Bad" training episodes are annotated by arrows. The shaded area indicates the oscillation range.

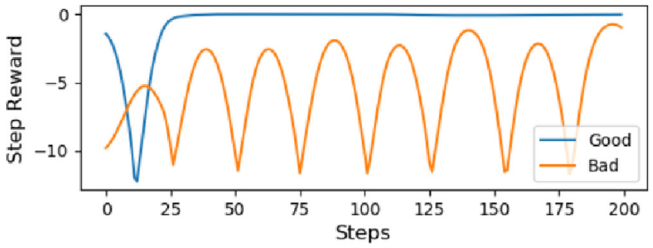

Fig. 2. The step reward trajectory of the "Good episode" 33 and the "Bad episode" 34" separately.

4. Variance aware reward smoothing

In this section, we introduce our method *Variance Aware Rewards Smoothing (VAR)* to mitigate the rewards drop phenomenon via variance reduction and give the theoretical analysis on convergence of the proposed approach. We first give the definition of VAR as follows.

Definition 1. For a reward trajectory $\{R_t\}$, the Variance Aware Rewards Smoothing (VAR) operation is defined by

$$\widehat{R}_t \triangleq \beta \widehat{R}_{t-1} + (1-\beta)R_t, \tag{6}$$

where $\left\{\widehat{R}_t\right\}$ is the VAR trajectory and $\beta \in [0, 1)$ controls the updating factor.

4.1. Variance reduction

Theorem 1. *Let $\left\{\widehat{R}_t\right\}$ be the VAR reward trajectory defined by Definition 1, the variance of $\left\{\widehat{R}_t\right\}$ is reduced, which gives*

$$\mathrm{var}\left[\widehat{R}_t\right] \leqslant \mathrm{var}[R_t]. \tag{7}$$

Proof. From Definition 1, we have

$$\begin{aligned}\mathrm{var}\left[\widehat{R}_t\right] &= \mathrm{var}\left[\beta\widehat{R}_{t-1} + (1-\beta)R_t\right] \\ &= \beta^2\mathrm{var}\left[\widehat{R}_{t-1}\right] + (1-\beta)^2\mathrm{var}[R_t] \\ &\quad + \mathrm{cov}\left[\beta\widehat{R}_{t-1}, (1-\beta)R_t\right].\end{aligned} \tag{8}$$

Since RL has the Markov property, where the conditional probability distribution of future states of the process depends only upon the present state, the covariance between $\widehat{R}_{t-1}$ and R_t is zero. Then we have

$$\mathrm{var}\left[\widehat{R}_t\right] = \beta^2\mathrm{var}\left[\widehat{R}_{t-1}\right] + (1-\beta)^2\mathrm{var}[R_t]$$

Utilizing $\beta \in [0, 1)$, we can derive that

$$\mathrm{var}\left[\widehat{R}_t\right] = \frac{(1-\beta)^2}{1-\beta^2}\mathrm{var}[R_t] \leqslant \mathrm{var}[R_t], \tag{9}$$

where the equality holds when $\beta = 0$ and VAR reward trajectory reduces to the original reward trajectory. And smaller β will cause more smooth reward trajectory. □

Then we define the value function $\widehat{V}_\pi(s)$ under VAR operation.

Definition 2. For a given VAR reward trajectory $\left\{\widehat{R}_t\right\}$, a discount factor γ and a policy π, the value function $\widehat{V}_\pi(s)$ is defined as

$$\widehat{V}_\pi(s) \triangleq \mathbb{E}_{a_t \sim \pi(a_t|s_t)}\left[\gamma^t \widehat{R}_t | s_0 = s\right]. \tag{10}$$

Besides the variance reduction property of VAR reward trajectory, we further investigate the TD-target term under VAR operation.

[1] https://github.com/openai/gym/wiki/Pendulum-v0.

Definition 3. For a given VAR reward trajectory $\left\{\widehat{R}_t\right\}$, a discount factor γ and a value function $\widehat{V}_\pi(s)$, the VAR TD-target is defined as

$$\widehat{G}_t = \widehat{R}_t + \gamma \widehat{V}_\pi(s_{t+1}). \tag{11}$$

Before the proof of variance reduction of $\widehat{G}_t$, we present necessary lemmas for the proof.

Lemma 1. *Let* $\left\{\widehat{R}_t\right\}$ *be the VAR rewards trajectory, we have* $\mathbb{E}\left[\widehat{R}_t\right] = \mathbb{E}[R_t]$.

Proof. Take the expectation on both sides of Eq. (6), we have

$$\mathbb{E}\left[\widehat{R}_t\right] = \beta \mathbb{E}\left[\widehat{R}_{t-1}\right] + (1-\beta)\mathbb{E}[R_t]. \tag{12}$$

Since $\mathbb{E}\left[\widehat{R}_t\right] = \mathbb{E}\left[\widehat{R}_{t-1}\right]$, we can easily get the results that $\mathbb{E}\left[\widehat{R}_t\right] = \mathbb{E}[R_t]$.

□

Lemma 2. $\widehat{V}_\pi(s)$ *is an identical transform of* $V_\pi(s)$.

Proof. Recall the definition of original value function $V_\pi(s)$ in Eq. (1) and the definition of $\widehat{V}_\pi(s)$. We have

$$\begin{aligned}\widehat{V}_\pi(s) - V_\pi(s) &= \mathbb{E}_{a_t\sim\pi(a_t|s_t)}\left[\gamma^t\left(\widehat{R}_t\right)|s_0 = s\right] \\ &\quad -\mathbb{E}_{a_t\sim\pi(a_t|s_t)}[\gamma^t(R_t)|s_0 = s] \\ &= \mathbb{E}_{a_t\sim\pi(a_t|s_t)}\left[\gamma^t\left(\widehat{R}_t - R_t\right)|s_0 = s\right].\end{aligned}$$

Applying Lemma 1, we know that $\widehat{V}_\pi(s) = V_\pi(s)$. □

Theorem 2. *Let* $\left\{\widehat{G}_t\right\}$ *be the VAR TD-target defined by Definition 3, the variance of* $\left\{\widehat{G}_t\right\}$ *is reduced, which gives*

$$\mathrm{var}\left[\hat{G}_t\right] \leqslant \mathrm{var}[G_t]. \tag{13}$$

Proof. We first derive the variance of the original TD-target G_t by

$$\begin{aligned}\mathrm{var}[G_t] &= \mathrm{var}[R_t + \gamma V_\pi(s_{t+1})] \\ &= \mathrm{var}[R_t] + \mathrm{var}[\gamma V_\pi(s_{t+1})] \\ &\quad +2\mathrm{cov}[R_t, \gamma V_\pi(s_{t+1})].\end{aligned} \tag{14}$$

According to Lemma 2, we can rewrite $\hat{G}_t = \widehat{R}_t + \gamma V_\pi(s_{t+1})$then, the variance of VAR TD-target $\hat{G}_t$ can be derived as

$$\begin{aligned}\mathrm{var}\left[\hat{G}_t\right] &= \mathrm{var}\left[\hat{R}_t\right] + \mathrm{var}[\gamma V_\pi(s_{t+1})] \\ &\quad +2\mathrm{cov}\left[\beta\widehat{R}_{t-1}, \gamma V_\pi(s_{t+1})\right] \\ &\quad +2\mathrm{cov}[(1-\beta)R_t, \gamma V_\pi(s_{t+1})].\end{aligned} \tag{15}$$

Applying Theorem 1, $\beta \in [0, 1)$ and the fact that $\widehat{R}_{t-1}$ is independent with $V_\pi(s_{t+1})$, it can be easily inferred that

$$\mathrm{var}\left[\hat{G}_t\right] \leqslant \mathrm{var}[G_t], \tag{16}$$

where the equality holds when $\beta = 0$. □

4.2. Convergence

In this section, we provide the convergence analysis of the proposed method to guarantee the performance theoretically. Before proof, we give some essential materials first. We use $\mathcal{T}$ to denote Bellman operator. The estimated VAR value function of $\widehat{V}_\pi(s)$ at n^{th} updating step is defined as

$$\widehat{V}_\pi^n(s_t) \leftarrow \widehat{R}_t + \gamma \widehat{V}_\pi^{n-1}(s_{t+1}). \tag{17}$$

The estimated value function of $V_\pi(s)$ at n^{th} updating step is defined as

$$V_\pi^n(s_t) \leftarrow R_t + \gamma V_\pi^{n-1}(s_{t+1}). \tag{18}$$

Lemma 3. *(Theorem 1 and 4 in [33]) If operator* $\mathcal{T}$ *is* γ*-contraction operator in* L^∞*-norm, that is,*

$$\|\mathcal{T}V_1 - \mathcal{T}V_2\|_\infty \leqslant \gamma\|V_1 - V_2\|_\infty \quad \forall V_1, V_2 \in \left\{V_\pi^n(s)\right\},$$

then the value iteration $\left\{V_\pi^n(s)\right\}$ *converges to* $V_\pi(s)$ *with probability 1.*

Now we can give the convergence proof of our proposed method based on the previous lemmas.

Theorem 3. *The value iterations of* $\left\{\widehat{V}_\pi^n(s)\right\}$ *converges to* $V_\pi(s)$ *with probability 1, which gives*

$$\lim_{n\to\infty} p\left(\widehat{V}_\pi^n(s) = V_\pi(s)\right) = 1. \tag{19}$$

Proof. Applying Bellman operator $\mathcal{T}$ on the estimated value function $\widehat{V}_\pi^n(s_t)$, we can get

$$\mathcal{T}\widehat{V}_\pi^n(s_t) \triangleq \mathbb{E}_{a_t\sim\pi(a_t|s_t)}\left[\widehat{R}_t + \gamma\widehat{V}_\pi^n(s_{t+1})\right]. \tag{20}$$

It is clear that $\widehat{V}_\pi(s)$ is the fixed-point of $\mathcal{T}$, which means

$$\mathcal{T}\widehat{V}_\pi^n(s) = \widehat{V}_\pi(s). \tag{21}$$

We derive that

$$\begin{aligned}\|\mathcal{T}\widehat{V}_1 - \mathcal{T}\widehat{V}_2\|_\infty &= \gamma\|\mathbb{E}_{a_t\sim\pi(a_t|s_t)}\left[\widehat{V}_1 - \widehat{V}_2\right]\|_\infty \\ &\leqslant \gamma\|\widehat{V}_1 - \widehat{V}_2\|_\infty.\end{aligned} \tag{22}$$

Now based on the γ-contraction property of $\mathcal{T}$ and Lemma 2 and 3, the value iteration $\left\{\widehat{V}_\pi^n(s)\right\}$ converges to $V_\pi(s)$ with probability 1. □

As described in Eq. (6), the VAR operation is a bootstrapping technique, which updates the current elements based on the previous ones iteratively. The bootstrapping property is consistent with the canonical bootstrapping RL algorithms such as Q-Learning [29] and is compatible with most of the recent Deep RL algorithms. When $\beta = 0$, the VAR reward trajectory degrades to the original one $\{R_t\}$. Moveover, thoretically, Theorem 3 shows that the proposed VAR method is able to be easily generalized into different value function based Deep RL algorithms. Practically, we provide the Q-Learning algorithm equipped with VAR shown in Algorithm 1 to demonstrate the simplicity of using it.

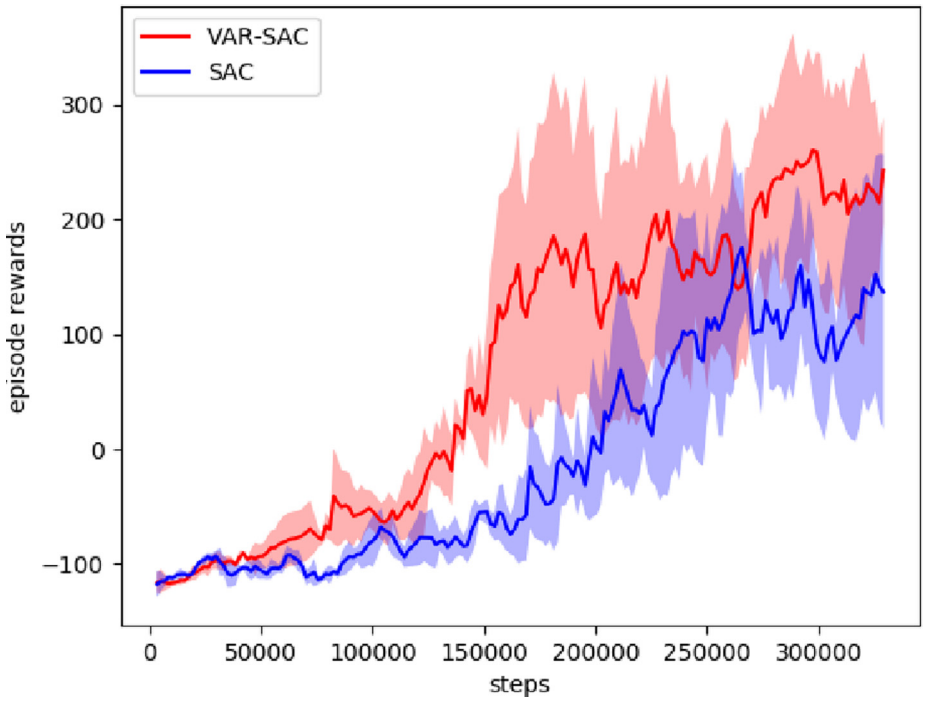

Fig. 3. The experiment on the `BipedalWalker` environment. Ten different random seeds are chosen while keeping other hyper-parameters the same. The solid line represents the mean value and shaded area indicates the standard variance.

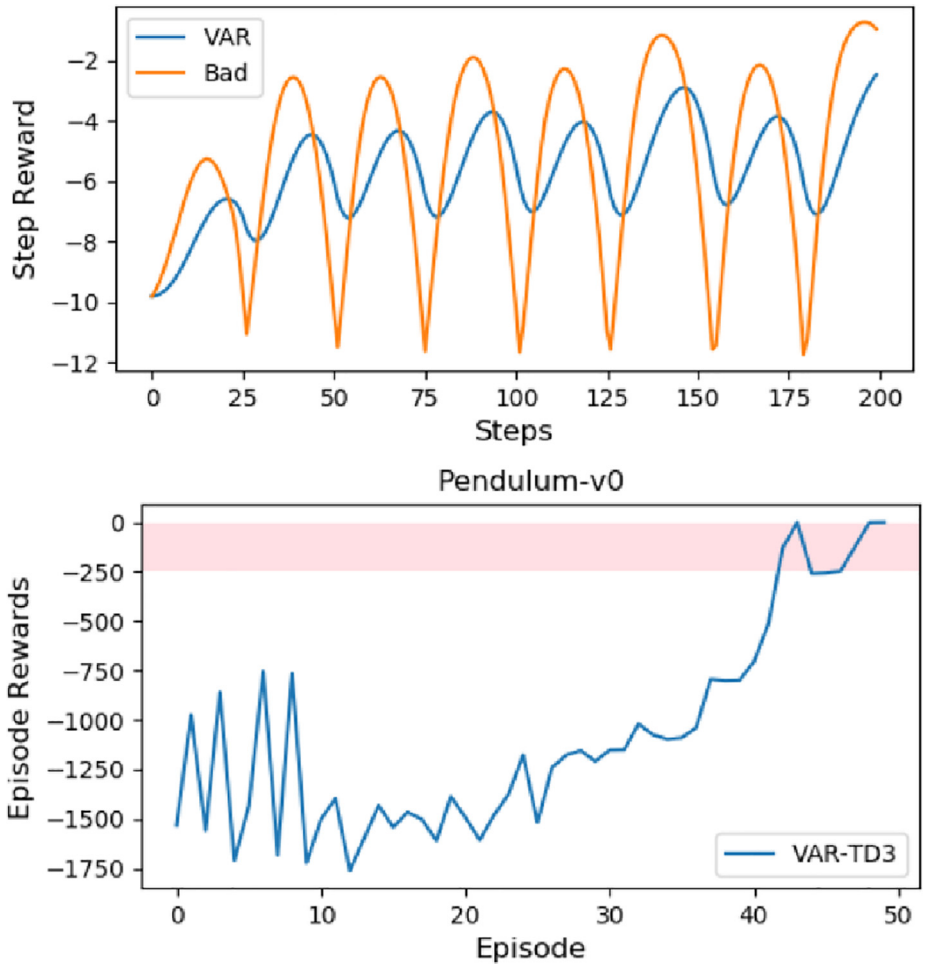

Fig. 4. The effectiveness of VAR operation. The first row shows the VAR operation on the "Bad" reward trajectory. The second row indicates the VAR operation on TD3 (VAR-TD3). The shaded area means the oscillation range.

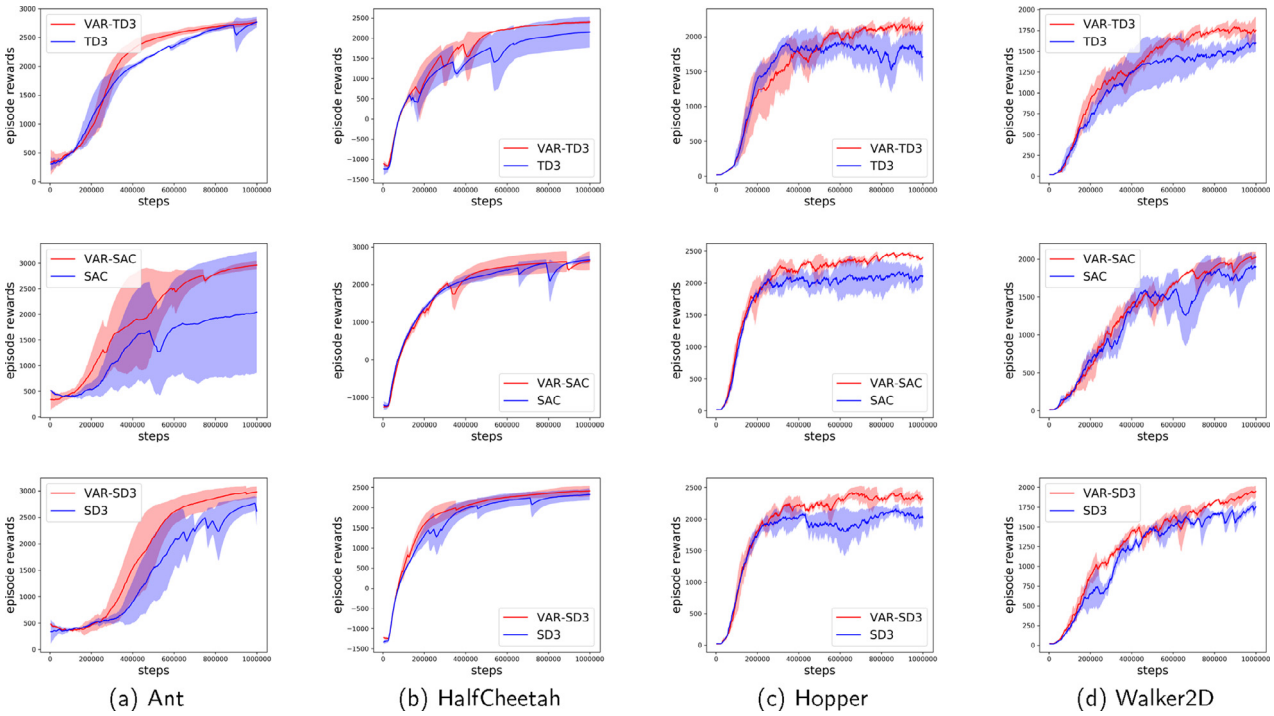

Fig. 5. The experiments on various environments (Ant, HalfCheetah, Hopper and Walker2D). VAR is applied on three well developed algorithms, i.e., TD3, SAC and SD3. In order to overcome the interference of random factors, all the experiments are conducted with ten different random seeds. The red lines indicates that VAR is activated. The solid lines represent the mean value and shaded area indicates the standard variance.

Algorithm 1: *Q*-Learning with VAR.

$\forall s \in \mathcal{S}, \forall a \in \mathcal{A}, \tilde{Q}^0(s,a) \leftarrow 0, \widehat{R} \leftarrow 0, t \leftarrow 0$
while not convergence **do**
  Sampling transition $(s_t, a_t, R_t, s_{t+1}, d_i)$ from the interacting with the environment, where d_i indicates if the training is finished or the trajectory is valid.
  $\widehat{R}_t \leftarrow \beta \widehat{R}_{t-1} + (1-\beta) R_t$
  $\tilde{Q}^{t+1}(s_t, a_t) \leftarrow (1-\alpha)\tilde{Q}^t(s_t, a_t)$
    $+\alpha\left\{\widehat{R}_t + \gamma \max_{a \in \mathcal{A}} \tilde{Q}^t(s_{t+1}, a)\right\}$, where α is the learning rate.
  $t \leftarrow t+1$
  if $d_i = 1$ **then**
    $\widehat{R} \leftarrow 0$
  end if
end while

5. Experiments

In this section, we first verify our proposed method on some motivating toy examples, which can effectively show the promoting ability of our proposed method. Then we give comprehensive numerical results on various environments over several advanced algorithms.

5.1. The pendulum example revisited

Consider the intuitive example described in Section 3 on the `Pendulum-v0` with TD3. We applied VAR operation directly on the orginial TD3 algorithm, and the results are shown in Fig. 4. Compared with Fig. 1, we can clearly see that VAR can improve TD3 performance and rewards drop phenomenon effectively.

Moreover, VAR can relieve the sudden crash reward problem, where a extremely large negative reward happened in training, described in `BipedalWalker` environment in gym. The dramatic change of rewards is pernicious for learning. However, our VAR method can reduce the magnitude of rewards drop because of the smaller variance. We tested VAR on `BipedalWalker` and compared with the existing algorithm Soft Actor Critic (SAC) with automatic temperature adjusting [8]. The experimental results are shown in Fig. 3. Obviously, VAR can promote the learning performance under the crash reward. Additionally, the higher rewards of VAR-SAC indicates that our proposed method not only helped the original algorithm avoid rewards drop problem, but it also improve the learning performance significantly.

5.2. Comprehensive numerical results

In this section, we provide more experiments to demonstrate the effectiveness of the proposed method. We applied VAR directly to existing advanced RL algorithms, namely TD3,[2] SAC[3] and SD3[4] [14] and compare the performance with the original algorithms on various environments, `Ant`, `HalfCheetah`, `Hopper` and `Walker2D` from PyBullet.[5] It is also worth noting that our proposed method is well compatible with the various mainstream on-policy or off-policy reinforcement learning training methods.

In order to eliminate the influence of random factors on the performance of the algorithms, each experiment was conducted across ten different random seeds. Moreover, for fairly benchmarking, we fixed total 1×10^6 steps in trainings. In all the experiments, we set β in VAR to be 0.2 and the all other hyper-parameters are aligned with their original papers.

[2] https://github.com/sfujim/TD3.
[3] https://github.com/quantumiracle/SOTA-RL-Algorithms.
[4] https://github.com/ling-pan/SD3.
[5] https://github.com/bulletphysics/bullet3.

Table 1
The statistical results of the conducted experiments on ten different random seeds. All the indicators are based on the last 2×10^5 steps during 1×10^6 steps training. And std denotes the standard deviation. The better results on the indicators are marked bolded. It can be seen that VAR obviously improves the performance during the later training stage especially in terms of mean and std.

		Ant			HalfCheetah			Hopper			Walker2D		
	use VAR?	mean	max	std	mean	max	std	mean	max	std	mean	max	std
TD3	✗	2331	**2952**	425	1772	**2739**	617	1795	**2432**	674	1370	1955	437
	✓	**2474**	2867	**382**	**2102**	2507	**522**	**1932**	2426	**606**	**1552**	**2069**	**413**
SAC	✗	1659	**3140**	1090	2325	2863	**460**	2064	2548	605	1508	**2277**	652
	✓	**2390**	3107	**773**	**2379**	**3092**	506	**2321**	**2705**	**504**	**1632**	2185	**522**
SD3	✗	1837	3000	937	2094	2628	458	2112	2564	558	1582	**2432**	606
	✓	**2282**	**3144**	**897**	**2227**	**2640**	**237**	**2304**	**2831**	**522**	**1742**	2427	**552**

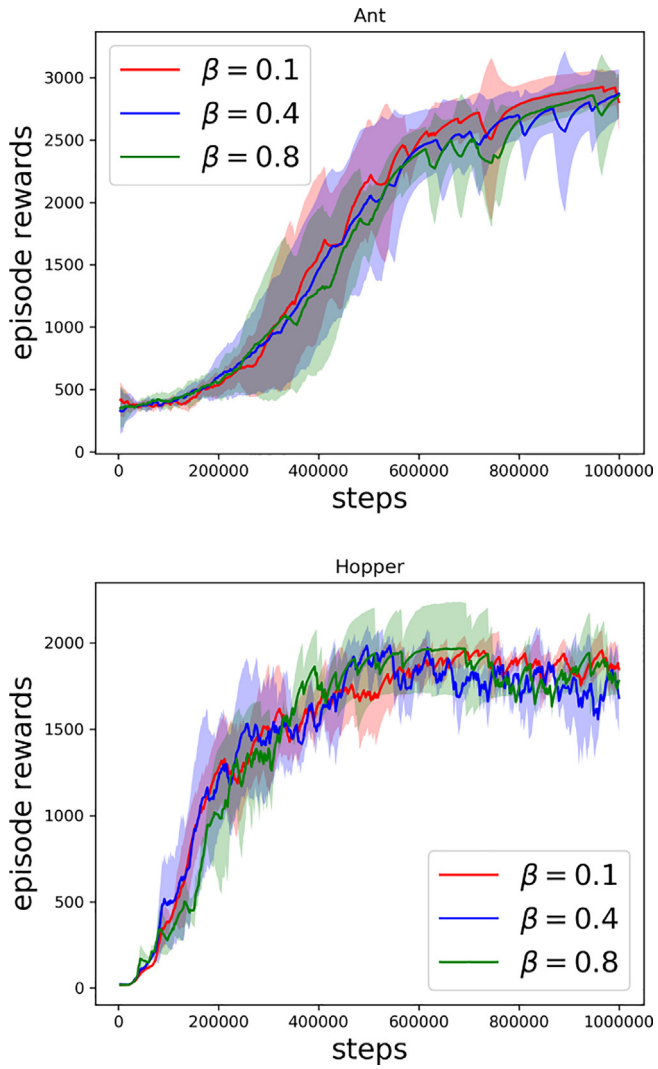

Fig. 6. The experiments are conducted on the `Ant` and `Hopper` environment with SD3. Ten different random seeds are chosen. The solid line represents the mean value and shaded area indicates the standard variance.

The experimental results are shown in Fig. 5. We can easily see that all episode rewards trajectories generated by the algorithms equipped with VAR have lower variance than the ones generated by the original algorithms, which matches Theorem 1. Additionally, in terms of the mean values of episode rewards regarding to the randomness in the experiments across different seeds, we can conclude that the algorithms equipped with VAR performed better than the original algorithms themselves. More importantly, the experiments results show that VAR can significantly mitigate the rewards drop phenomenon and produce more stable training performance during the late-stage training sessions.

In Table 1, we summarize the statistical results that contains the mean value, the maximum and the standard deviations of episode rewards associated with the experiments across different seeds.

The substantial advantages in the mean and standard deviation indicators obviously support that our proposed method improves the training performance and produces more stable learning tendency.

5.3. Sensitivity analysis

Our proposed method introduces one hyper-parameter $\beta \in [0, 1)$ to control the degree of smoothing on the reward trajectory. The combination of Definition 1 and Lemma 1 imply that the rewards trajectories generated by different valid β will not change dramatically in terms of expectation.

To investigate and demonstrate the sensitivity of the hyper-parameter β, we conduct experiments on `Ant` and `Hopper` environment over SD3. The experimental results are depicted in Fig. 6, which indicates that a reasonable range of β does not result in widely varying results.

6. Conclusions

In this paper, we proposed a variance aware rewards smoothing method, which aims to mitigate the rewards drop phenomenon through the variance reduction approach. We showed that our proposed method can reduce the variance of both the reward trajectory and TD-target theoretically. Then, we also proved that our method does not change the original value function. Furthermore, we gave the convergence proof of our method via γ-contraction property and the fixed-point attribute of the value function. We also indicated that the proposed method can be easily generalized to other value based Deep RL algorithms. Finally, extensive experimental results on various environments and advanced algorithms were conducted to demonstrate the effectiveness of our proposed method, which matches our theoretical results.

CRediT authorship contribution statement

Yunlong Dong: Conceptualization, Methodology, Writing - original draft. **Shengjun Zhang:** Reviewing & editing. **Xing Liu:** Experiments. **Yu Zhang:** Visualization. **Tan Shen:** Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] M. Andrychowicz, B. Baker, M. Chociej, et al., Learning dexterous in-hand manipulation, Int. J. Robot. Res. 39 (2020) 3–20.
[2] R. Bellman, Dynamic programming, Science 153 (1966) 34–37.
[3] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, W. Zaremba, Openai gym, 2016. arXiv preprint arXiv:1606.01540. .
[4] Y. Dong, X. Tang, Y. Yuan, Principled reward shaping for reinforcement learning via lyapunov stability theory, Neurocomputing (2020).
[5] Z.Y. Fu, D.C. Zhan, X.C. Li, Y.X. Lu, Automatic successive reinforcement learning with multiple auxiliary rewards, in, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence IJCAI-19, 2019, pp. 2336–2342.
[6] S. Fujimoto, H. Van Hoof, D. Meger, Addressing function approximation error in actor-critic methods, 2018. arXiv preprint arXiv:1802.09477. .
[7] D. Ghosh, M.G. Bellemare, Representations for stable off-policy reinforcement learning, in: International Conference on Machine Learning, PMLR, 2020, pp. 3556–3565.
[8] T. Haarnoja, A. Zhou, K. Hartikainen, et al., Soft actor-critic algorithms and applications, 2018. arXiv preprint arXiv:1812.05905. .
[9] K. Ji, Z. Wang, B. Weng, Y. Zhou, W. Zhang, Y. Liang, History-gradient aided batch size adaptation for variance reduced algorithms, International Conference on Machine Learning, PMLR (2020) 4762–4772.
[10] B.R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A.A.A. Sallab, S. Yogamani, P. Pérez, Deep reinforcement learning for autonomous driving: a survey, 2020. arXiv preprint arXiv:2002.00444. .
[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, 2013. arXiv preprint arXiv:1312.5602. .
[12] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (2015) 529–533.
[13] A.Y. Ng, D. Harada, S. Russell, Policy invariance under reward transformations: theory and application to reward shaping, ICML (1999) 278–287.
[14] L. Pan, Q. Cai, L. Huang, Softmax deep double deterministic policy gradients, Adv. Neural Inf. Process. Syst. 33 (2020).
[15] Z. Peng, J. Hu, Y. Zhao, B.K. Ghosh, Understanding the mechanism of human–computer game: a distributed reinforcement learning perspective, Int. J. Syst. Sci. 51 (2020) 2837–2848.
[16] A. Perrusquía, W. Yu, Continuous-time reinforcement learning for robust control under worst-case uncertainty, Int. J. Syst. Sci. (2020) 1–15.
[17] J. Romoff, P. Henderson, A. Piché, V. Francois-Lavet, J. Pineau, Reward estimation for variance reduction in deep reinforcement learning, 2018. arXiv preprint arXiv:1805.03359. .
[18] E. Sarafian, A. Tamar, S. Kraus, Constrained policy improvement for efficient reinforcement learning, in: C. Bessiere (Ed.), Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, 2020, pp. 2863–2871, Main track.
[19] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, et al., Mastering atari, go, chess and shogi by planning with a learned model, 2019. arXiv preprint arXiv:1911.08265. .
[20] J. Schulman, S. Levine, P. Abbeel, M. Jordan, P. Moritz, Trust region policy optimization, International Conference on Machine Learning (2015) 1889–1897.
[21] J. Schulman, P. Moritz, S. Levine, M. Jordan, P. Abbeel, High-dimensional continuous control using generalized advantage estimation. arXiv preprint arXiv:1506.02438. .
[22] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, 2017. arXiv preprint arXiv:1707.06347. .
[23] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of go with deep neural networks and tree search, Nature 529 (2016) 484–489.
[24] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, M. Riedmiller, Deterministic policy gradient algorithms, in: Proceedings of the 31st International Conference on Machine Learning, PMLR, 2014, pp. 387–395.
[25] R.S. Sutton, Learning to predict by the methods of temporal differences, Mach. Learn. 3 (1988) 9–44.
[26] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 2018.
[27] H. Van Hasselt, Y. Doron, F. Strub, M. Hessel, N. Sonnerat, J. Modayil, Deep reinforcement learning and the deadly triad, 2018. arXiv preprint arXiv:1812.02648. .
[28] H. Van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double q-learning, 2015. arXiv preprint arXiv:1509.06461. .
[29] C.J. Watkins, P. Dayan, Q-learning, Mach. Learn. 8 (1992) 279–292.
[30] R.J. Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning, Mach. Learn. 8 (1992) 229–256.
[31] D. Wu, X. Dong, J. Shen, S.C. Hoi, Reducing estimation bias via triplet-average deep deterministic policy gradient, IEEE Trans. Neural Netw. Learn. Syst. 8 (1992) 229–256.
[32] Q. Wu, B. Zhao, D. Liu, Adaptive dynamic programming-based decentralised control for large-scale nonlinear systems subject to mismatched interconnections with unknown time-delay, Int. J. Syst. Sci. 51 (2020) 2883–2898.
[33] L. Yang, M. Shi, Q. Zheng, W. Meng, G. Pan, A unified approach for multi-step temporal-difference learning with eligibility traces in reinforcement learning, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, 2018, pp. 2984–2990.
[34] N. Zeng, H. Li, Z. Wang, W. Liu, S. Liu, F.E. Alsaadi, X. Liu, Deep-reinforcement-learning-based images segmentation for quantitative analysis of gold immunochromatographic strip, Neurocomputing 425 (2021) 173–180.
[35] N. Zeng, Z. Wang, H. Zhang, K.E. Kim, Y. Li, X. Liu, An improved particle filter with a novel hybrid proposal distribution for quantitative analysis of gold immunochromatographic strips, IEEE Trans. Nanotechnol. 18 (2019) 819–829.
[36] N. Zeng, Z. Wang, H. Zhang, W. Liu, F.E. Alsaadi, Deep belief networks for quantitative analysis of a gold immunochromatographic strip, Cogn. Comput. 8 (2016) 684–692.
[37] N. Zeng, H. Zhang, B. Song, W. Liu, Y. Li, A.M. Dobaie, Facial expression recognition via learning deep sparse autoencoders, Neurocomputing 273 (2018) 643–649.
[38] D. Zhang, C.P. Bailey, Obstacle avoidance and navigation utilizing reinforcement learning with reward shaping, in: Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II, International Society for Optics and Photonics, 2020, p. 114131H.

Yunlong Dong received the B.E. degree from the School of Automation, Huazhong University of Science and Technology, Wuhan, China, in 2017. He is currently working towards the Ph.D. degree at School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan, China. His research interests include RL, control theory and robotics.

Shengjun Zhang received the B.E. degree in Automation of Honors Program from China Agricultural University, Beijing, China, in 2014, and the M.S. degree in Electrical Engineering from New York University, New York, New York, USA, in 2017. He is currently working toward a Ph. D. degree with the Department of Electrical Engineering, University of North Texas, Denton, Texas, USA. His current research interests include distributed optimization, statistical learning, and control theory.

Xing Liu received the B.E. degree from the School of Electrical and Automation Engineering, East China Jiaotong University in 2020. He is currently working towards the Master degree at School of Information Science and Engineering, East China University of Science and Technology, Shanghai, China. His research interests include intelligent control, deep learning and image processing.

Yu Zhang received the B.E. degree from the School of Information Science and Technology, Southwest Jiao-Tong University, ChengDu, China, in 2018. She is currently working towards the Master degree at School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan, China. Her research interests include RL and low level image process.

Tan Shen received the B.E. degree from the School of Automation, Huazhong University of Science and Technology, Wuhan, China, in 2018. She is currently working towards the Ph.D. degree at School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan, China. Her research interests include system identification, control theory and robotics.