J. Nonlinear Var. Anal. 6 (2022), No. 3, pp. 227-238
Available online at http://jnva.biemdas.com
https://doi.org/10.23952/jnva.6.2022.3.05

VISUAL TRANSFER FOR REINFORCEMENT LEARNING VIA GRADIENT PENALTY BASED WASSERSTEIN DOMAIN CONFUSION

XIANCHAO ZHU, RUIYUAN ZHANG, TIANYI HUANG, XIAOTING WANG*

Institute of Fundamental and Frontier Sciences,
University of Electronic Science and Technology of China, Chengdu, China

Abstract. It is pretty challenging to transfer learned policies among different visual environments. The recently proposed Wasserstein Adversarial Proximal Policy Optimization (WAPPO) attempts to overcome this difficulty by definitely learning a representation, which is sufficient to express the originating and the target domains simultaneously. Specifically, WAPPO uses the Wasserstein Confusion target function to force reinforcement learning (RL) agents to learn the mapping from visually different environments to domain-independent expressions, thereby achieving better domain adaptation performance in RL. However, WAPPO uses weight clipping to strengthen the Lipschitz continuity of the Wasserstein Confusion target function, which results in poor manifestation. In this paper, we present Gradient Penalty based Wasserstein Adversarial Proximal Policy Optimization (GPWAPPO), a new approach for the visual transfer in RL that learns to match the distributions of distilled characteristics between an originating domain and the objective domain. Specifically, we propose a new target function, Gradient Penalty-based Wasserstein Confusion (GPWC), which uses selective clipping weights to catch up with the gradient norm of the target function relative to its input. GPWAPPO is superior to the previous methods in visual transfer and triumphantly transfers strategies across Visual Cartpole and 16 OpenAI Procgen domains.

Keywords. Gradient penalty; Reinforcement learning; Visual transfer; Wasserstein distance.

1. INTRODUCTION

Deep reinforcement learning (RL) enables agents to address complex voluntarily, standing issues based on pixel input, such as Atari games [1, 2]. However, reinforcement learning agents cannot learn from errors without undermining equipment in high-stake areas, such as autonomous cars. The better way to deal with this issue is to train the agent in the analog domain and migrate the learned policy to the actual world. However, deep RL agents trained in an identical environment cannot solve the same potential problems in environments with different visual inputs [3, 4]. The main reason is a real gap when transferring from the originating domain to the objective environment [5, 6]. In deep supervised learning (SL), this problem can be solved by smoothly learning functions using data enrichment [7], dropout [8], and normalization [9]. However, these methods still cannot be generalized to deep RL [3]. The main key point is that

*Corresponding author.
E-mail address: xiaoting@uestc.edu.cn (X. Wang).
Received November 1, 2021; Accepted February 2, 2022.

the transfer between different RL tasks is similar to the adaptation of the supervisory domain in which the data points used in the source task and the target task come from different distributions [10, 11]. These methods often require a good deal of source domains to migrate or adapt to the target environment [12, 13]. Some recent research works aligned the feature distributions of different environments more clearly by assuming that agents can access the pairwise correlation of states in various domains [14, 15]. Other research work used the style transfer network to "translate" the objective environment state before processing the source environment state [16]. Further research efforts try to learn a kind of causal structure behind the visual Markov decision process (MDP) family to obtain a causal expression isolated of the environment and successful transfer [17]. Recently, Roy and Konidaris proposed a Wasserstein Adversarial Proximal Policy Optimization (WAPPO) method for visual transfer in RL [10]. This algorithm learns to match the distribution of distilled characteristics between the originating and objective tasks. There is no need to add additional calculations or hypothetical visits to the coupled correlations of the observed states during the inference process. Specifically, WAPPO trains an RL agent to approximate the Wasserstein-1 distance between the feature distributions from the originating task and the objective task by minimizing the distance of the Wasserstein Confusion target function [18, 19]. However, WAPPO uses weight clipping to strengthen the Lipschitz constraint of the Wassersestein Confusion target function, which often leads to poor performance.

In this paper, we present Gradient Penalty based Wasserstein Adversarial Proximal Policy Optimization (GPWAPPO), a new approach that learns to match distributions of distilled characteristics between an originating and objective domain without accessional extra calculation during deduction or supposing access to coupled relations of states. By contemporaneously training an RL agent to address an originating task and minimizing the distance between distributions of distilled characteristics from the originating and objective environments, our method can seamlessly migrate to the objective environment. To enforce the Lipschitz constraint, we introduce a new objective function, Gradient Penalty-based Wasserstein Confusion (GPWC). When using GPWAPPO, the similar densities in the target and source features are clustered together to align their distributions, eventually leading to higher rewards. In this way, our proposed GPWAPPO algorithm surpasses the previous visual transfer approaches and triumphantly transfer strategies across Visual Cartpole and 16 OpenAI Procgen environments.

This paper is structured as follows. In Section 2, we begin with a brief exposition of the RL notions. Section 3 provides a brief review of transfer RL approaches, an introduction to visual transfer in RL and a concise review of adversarial distribution alignment. Section 4 provides our GPWAPPO algorithm. In Section 5, we present the experiments on several experimental environments to verify the effectiveness of our proposed method. Section 6, which is also the last section, concludes this paper.

2. BACKGROUND

Proximal Policy Optimization (PPO) is the most advanced strategy gradient method for deep RL[1]. It parameterizes the strategy $\pi_\vartheta(a|s)$ and the value function $\mathscr{V}_\vartheta(s)$ that shares most of its weight and splits it after the "feature extraction" part of the strategy net. The value net is trained to reduce the error of mean square $\mathscr{L}_{value} = \frac{1}{n}\sum_{i=1}^{N}(\mathscr{V}_\vartheta(s) - \mathscr{R})^2$, where $\mathscr{R}$ stands for the reward. The strategy net is trained to reduce

$$\mathscr{L}_{policy} = -\hat{\mathbb{E}}_t[min(r_t(\vartheta)\hat{\mathscr{A}_t}, clip(r_t(\vartheta), 1-\upsilon, 1+\upsilon)\hat{\mathscr{A}_t})],$$

where $\hat{\mathbb{E}}$ stands for the empirical expectation average at time step t, $\hat{\mathscr{A}}_t$ stands for the benefit of experience at timestep t, $r(\vartheta) = \frac{\pi_\vartheta(a_t|s_t)}{\pi_{\vartheta old}(a_t|s_t)}$ stands for the rate of adopting action a_t given state s_t between the present and anterior strategies, and υ stands for a hyper-parameter.

3. Related Work

3.1. **Transfer in reinforcement learning.** Transfer in RL can be divided into dynamic transfer, knowledge representation-based transfer, and goal-based transfer, where the state, action, transfer function, or reward and function of the environment are different [20, 21]. Transfer in RL aims to learn a strategy that can be perfectly trained on the originating MDP while obtaining an equivalent target reward on the target MDP.

Domain randomization is the most fashionable zero-sample transfer method in reinforcement learning. In domain randomization, the agent is trained on a set of n originating tasks to clandestinely learn knowledge representations and strategies wealthy for zero-sample transfer to the $n+1$ mission [22, 23, 24, 25]. Some research works directly solved the problem of dynamic transfer when the transfer function changes under the assumption that the state, action, and reward are the same [22, 23]. Some research works parameterized the transfer function in MDP and learned the conditional strategies that can be transferred between such tasks [24, 25]. Other research works through generating curriculums to help agents learn general strategies that adapt to different dynamic MDPs [26].

3.2. **Visual transfer in reinforcement learning.** Visual transfer occurs in the family $\mathscr{M}$ related to $M \in \mathscr{M}$. Block MDP is a subset of POMDP, which has a transmission function but no observation function [27, 28]. Both of these functions map from the hidden state to the observation value. Still, the observation value generated by the emission function is defined as Markovian and can uniquely identify its corresponding underlying state. In addition, Block MDP can be extended to real-world applications, such as transferring robot reinforcement learning simulation to reality. To transfer from the world of emulation to the actual world, the agent must transfer between tasks with different observations and the same basic state, action, transfer function, and reward [29].

Other research works suppose that the image pairs from the originating and target environments have markers and align the cross-domain feature distribution by minimizing the pairing distance [30, 31]. However, the speed of collecting sample pairs of originating and target environments with the same potential state will decrease rapidly as the complexity of the state space boosts. Therefore, such approaches are often impractical or impractical in practice.

By training the Generative Adversarial Network (GAN) to project input pictures to output pictures with the same semantics but different styles, the supervised domain adaptation, and style transfer methods can transform the pictures into diverse "styles," such as the style of celebrated painters [32, 33]. On the contrary, in complex visual reinforcement learning tasks, the number of pixels occupied by essential elements such as player characters is small, and its priority is low [34]. Although style transfer increases computation complexity and relies on image reconstruction, GAN and supervised domain transfer methods can align distributions without these shortcomings.

3.3. **Adversarial distribution alignment.** The domain adaptation method based on supervised learning adopts a GAN-like way to match the internal representation of the cross-domain classifier to solve the supervised classification similar to the visual transfer in reinforcement learning [22, 23]. This type of method minimizes different distribution alignment targets by reducing the classification accuracy of the adversarial network [22, 23].

However, the alignment distribution of adversarial algorithms like GAN is volatile and tends to problems such as mode collapse [35]. In addition, it is often impossible to minimize the JS divergence between two distributions in some cases [35]. Wasserstein GAN solves these two issues by replacing the oppositional classifier with an oppositional evaluator, avoiding model collapse by reducing the Wasserstein-1 distance between the actual and false distributions [35]. In reality, it is often impossible to directly calculate the value of Wasserstein-1 distance. Generally, Kantorovich-Rubinstein duality is used to reparameterize it for calculation.

3.4. **Wasserstein adversarial proximal policy optimization.** The goal of the Wasserstein Adversarial Proximal Policy Optimization (WAPPO) method is to branch out into a representation function $e_{\vartheta,(s,t)}$ to approximate the first-rank representing function $e^*_{s,t}$ [10]. WAPPO defines the representing function e_ϑ as the first few layers of an RL network with a parameter ϑ. To learn domain-independent knowledge representations, WAPPO jointly learns to solve the originating task M_s, and in the meantime, uses adversarial methods to match the representation distributions from the originating task M_s and the objective task M_t. Its mathematical expression is as follows: $\mathscr{L}_{WAPPO} = \mathscr{L}_{PPO} + \lambda \mathscr{L}_{WC}$, where $\mathscr{L}_{PPO}$ is the loss of the policy network [10], λ is a constant weight, and $\mathscr{L}_{WC}$ is the Wasserstein Confusion loss:

$$\mathscr{L}_{WC} = \mathscr{W}(\mathscr{P}_s, \mathscr{P}_t) = \mathrm{E}_{x\sim\mathscr{P}_s}[f(e_\vartheta(x))] - \mathrm{E}_{\tilde{x}\sim\mathscr{P}_t}[f(e_\vartheta(\tilde{x}))],$$

where f represents the confrontational critic.

To make WAPPO stable training, WAPPO uses the same method as WGAN to tailor the weights of $\mathscr{L}_{WC}$ to a compact space $[-c, c]$ to implement Lipschitz constraints on $\mathscr{L}_{WC}$. However, the interaction between the weight constraints and the cost function will cause the gradient to disappear or explode without the need to carefully adjust the clipping threshold c, which ultimately leads to poor performance.

4. Gradient Penalty based Wasserstein Adversarial Proximal Policy Optimization

In this section, we propose Gradient Penalty based Wasserstein Adversarial Proximal Policy Optimization (GPWAPPO), a new approach for visual transfer in RL that learns to match the distributions of distilled characteristics between an originating and objective task. Specifically, we introduce a new objective function, Gradient Penalty-based Wasserstein Confusion (GPWC), to enforce the Lipschitz constraint. Its mathematical expression is as follows:

$$\mathscr{L}_{GPWAPPO} = \mathscr{L}_{PPO} + \lambda \mathscr{L}_{GPWC},$$

where

$$\mathscr{L}_{GPWC} = \mathrm{E}_{x\sim\mathscr{P}_s}[f(e_\vartheta(x))] - \mathrm{E}_{\tilde{x}\sim\mathscr{P}_t}[f(e_\vartheta(\tilde{x}))] + \gamma \mathrm{E}_{x_\varepsilon\sim\mathscr{P}_{x_\varepsilon}}[(\|\nabla_{x_\varepsilon} f(e_\vartheta(x_\varepsilon))\|_2 - 1)^2],$$

where $x_\varepsilon = \varepsilon x + (1-\varepsilon)\tilde{x}$ and $0 \le \varepsilon \le 1$.

We adopt P_{x_ε} sampling evenly along the line between pairs of data points sampled from the data distribution P_s and the generator distribution P_t. This is because the optimal critic contains a

straight line with gradient norm 1 connecting the coupling data points from P_s and P_t (as shown in Theorem 4.1). In this way, our method can encourage the gradient of the standard toward 1 rather than stay below it. Empirically, this does not seem to constrain the cirtic network too much probably because the optimal GAWAPPO critic network anyway has gradients with norm 1 almost everywhere under P_s and P_t in most regions in between. In addition, we use layer regularization instead of batch regularization to penalize the norm of each input critic gradient independently [35]. The main reason is that batch regularization converts the mode of the discriminator issue from a simplex input to a single output mapping to a batch input to a batch output mapping, making our penalty training target no longer valid in this case.

Theorem 4.1. *Let P_s and P_t be two distributions in $\mathscr{U}$, an impact metric space. There is a 1-lipschitz function h^*, which is the optimum solution of* $\max_{\|h\|_L\leq 1} \mathrm{E}_{z\sim\mathscr{P}_s}[(h(z))]-\mathrm{E}_{x\sim\mathscr{P}_t}[(h(x))]$. *Let ξ be the optimal coupling between P_s and P_t, formalized as $W(P_s,P_t)=\inf_{\xi\in\prod(P_s,P_t)}\mathrm{E}_{(x,z)\sim\xi}[\| x-z \|]$, where $\prod(P_s,P_t)$ is the set of simultaneous distributions $\xi(x,z)$ whose marginal distributions are P_s and P_t, separately. Subsequently, if h^* is derivable, $\xi(x=z)=0$, and $x_\varepsilon=\varepsilon x+(1-\varepsilon)z$ with $0\leq\varepsilon\leq 1$, then $P_{(x,z)\sim\xi}[\nabla h^*(x_\varepsilon)=\frac{z-x_\varepsilon}{\|z-x_\varepsilon\|}]=1$.*

Proof. Since $\mathscr{U}$ is an impact metric space, we see from [35] that there is an optimal h^*. Moreover, according to [35], we also know that if ξ is an optimal coupling, then $P_{(x,z)\sim\xi}[h^*(z)-h^*(x)=\| z-x \|]=1$. Let (x,z) be such that $h^*(z)-h^*(x)=\| z-x \|$. We can suppose that $x\neq z$ as well since this occurs under ξ with probability 1. Let $\varphi(\varepsilon)=h^*(x_\varepsilon)-h^*(x)$. We define that $\varphi(\varepsilon)=\| x_\varepsilon-x \|=\varepsilon\| z-x \|$. Let $\varepsilon,\varepsilon'\in[0,1]$. Then, $|\varphi(\varepsilon)-\varphi(\varepsilon')|=\| h^*(x_\varepsilon)-h^*(x_{\varepsilon'}) \|]\leq\| x_\varepsilon-x_{\varepsilon'} \|\leq|\varepsilon-\varepsilon'|\| x-z \|$. Thus, φ is $\| z-x \|$-Lipschitz. This in turn implies that

$$|\varphi(1)-\varphi(0)|\leq(1-\varepsilon)\| x-z \|+\varphi(\varepsilon)-\varphi(0)\leq(1-\varepsilon)\| x-z \|+\varepsilon\| x-z \|=\| x-z \|.$$

However, $|\varphi(1)-\varphi(0)|=h^*(z)-h^*(x)=\| z-x \|$. Hence, the inequality above is virtually an equality. In particular, $|\varphi(1)-\varphi(0)|=\varepsilon\| x-z \|$ and $\varphi(0)=h^*(0)-h^*(0)=0$. Therefore, $\varphi(\varepsilon)=\varepsilon\| x-z \|$. Let $\nu=\frac{z-x_\varepsilon}{\|z-x_\varepsilon\|}=\frac{z-((1-\varepsilon)x-\varepsilon z)}{\|z-((1-\varepsilon)x-\varepsilon z)\|}=\frac{z-x}{\|z-x\|}$. Observe that $h^*(x_\varepsilon)-h^*(x)=\varphi(\varepsilon)=\varepsilon\| x-z \|$. One has $h^*(x_\varepsilon)=h^*(x)+\varepsilon\| x-z \|$. It follows that

$$\begin{aligned}
\frac{\partial}{\partial\nu}h^*(x_\varepsilon)&=\lim_{\delta\to 0}\frac{h^*(x_\varepsilon+\delta\nu)-h^*(x_\varepsilon)}{\delta}\\
&=\lim_{\delta\to 0}\frac{h^*(x+\varepsilon(z-x)+\frac{\delta}{\|z-x\|}(z-x))-h^*(x_\varepsilon)}{\delta}\\
&=\lim_{\delta\to 0}\frac{h^*(x_{\varepsilon+\frac{\delta}{\|z-x\|}})-h^*(x_\varepsilon)}{\delta}\\
&=\lim_{\delta\to 0}\frac{h^*(x)+(\varepsilon+\frac{\delta}{\|z-x\|})\| x-z \|-(h^*(x_\varepsilon)-\varepsilon\| x-z \|)}{\delta}=1.
\end{aligned}$$

If h^* is derivable at x_ε, we set that $\| \nabla h^*(x_\varepsilon) \|\leq 1$ because it is a 1-Lipschitz function. Thus, ν is as a normalized vector

$$\begin{aligned}
1\leq\| \nabla h^*(x_\varepsilon) \|^2&=\langle\nu,\nabla h^*(x_\varepsilon)\rangle^2+\| \nabla h^*(x_\varepsilon)-\langle\nu,\nabla h^*(x_\varepsilon)\rangle\nu \|^2\\
&=|\frac{\partial}{\partial\nu}h^*(x_\varepsilon)|+\| \nabla h^*(x_\varepsilon)-\nu\frac{\partial}{\partial\nu}h^*(x_\varepsilon) \|^2\\
&=1+\| \nabla h^*(x_\varepsilon)-\nu \|^2\leq 1.
\end{aligned}$$

The fact that the two extremes of the inequation coincide represents that it is both an equality and $1 = 1 + \| \nabla h^*(x_\varepsilon) - \nu \|^2$. So, $\| \nabla h^*(x_\varepsilon) - \nu \|^2 = 0$, and thus $\nabla h^*(x_\varepsilon) = \nu$. This demonstrates that $\nabla h^*(x_\varepsilon) = \frac{(z - x_\varepsilon)}{\|z - x_\varepsilon\|}$. Hence, if (x, z) have the nature that $h^*(z) - h^*(x) = \| z - x \|$, then $\nabla h^*(x_\varepsilon) = \frac{(z - x_\varepsilon)}{\|z - x_\varepsilon\|}$. Since this occurs with probability 1 under ξ, one has

$$P_{(x,z)\sim\xi}[\nabla h^*(x_\varepsilon) = \frac{(z - x_\varepsilon)}{\| z - x_\varepsilon \|}] = 1.$$

This completes the proof. □

The discrepancy in manifestation between GPWAPPO and other methods can be interpreted as follows. Consider the simplex $\Omega_n = \{q \in \mathbb{R}^n : q_i \geq 0, \sum_i q_i = 1\}$, and and the set of nodes on the simplex $Z_n = \{q \in \mathbb{R}^n : q_i \in (0,1), \sum_i q_i = 1\} \subset \Omega_n$. If we have a vocabulary of size n and a distribution P_s over a series of size K, then we have that P_t is a distribution on $Z_n^K = Z_n \times \cdots \times Z_n$. Since V_n^K is a subset of Ω_n^K, we can also consider P_s as a distribution over Ω_n^K. P_s is a discrete distribution on Ω_n^K, but P_s can easily be a continuous distribution on Ω_n^K. The way this behaves is that, in GPWAPPO, the Lipschitz restriction restricts the critic net to offer linear gradients from all Ω_n^K to the true points in Z_n^K. An advantage of our method is that its objective loss is related to sample quality and can converge to a minimum value. Furthermore, in GPWAPPO, the training loss increases gradually even as the validation loss decreases.

The Gradient Penalty based Wasserstein Adversarial Proximal Policy Optimization (GPWAPPO) approach is as follows:

Algorithm 1 Gradient Penalty based Wasserstein Adversarial Proximal Policy Optimization (GPWAPPO)

Require: The gradient regularization factor γ, the amount of critic repeats each generator iteration n_{critic}, the time steps $n_{timesteps}$, RMSProp hyperparameters α.

Ensure: Optimal ϑ^*

while ϑ has not converged **do**
  for $t = 0, ..., n_{timesteps}$ **do**
    for $j = 0, ..., n_{critic}$ **do**
      Sample $\{\mathfrak{s}_{s,i}\}_{i=1}^m \sim \mathscr{P}_s$ *a batch from the source domain*
      Sample $\{\mathfrak{s}_{t,i}\}_{i=1}^m \sim \mathscr{P}_t$ *a batch from the target domain buffer*
      $\mathfrak{s}_{\varepsilon,i} = \varepsilon \mathfrak{s}_{s,i} + (1 - \varepsilon)\mathfrak{s}_{t,i}$
      $\varsigma_\omega \leftarrow \nabla_\omega [\frac{1}{m}\sum_{i=1}^m f_\omega(h_\vartheta(\mathfrak{s}_{s,i})) - \frac{1}{m}\sum_{i=1}^m f_\omega(h_\vartheta(\mathfrak{s}_{t,i}))$
        $+ \frac{1}{m}\sum_{i=1}^m \gamma(\|\nabla_{\mathfrak{s}_{ep,i}}\{\omega(h_\vartheta(\mathfrak{s}_{ep,i}))\|_2 - 1)^2]$
      $\omega \leftarrow \omega + \alpha \times RMSProp(\omega, \varsigma_\omega)$
    end for
    Sample a batch of $\{\mathfrak{s}_{s,i}, a_{s,i}, r_{s,i}\}_{i=1}^m$ *and* $\{\mathfrak{s}_{t,i}\}_{i=1}^m$ *from the source domain and target domain buffer respectively*
    $\varsigma_\vartheta \leftarrow \nabla_\vartheta [-\frac{1}{m}\sum_{i=1}^m f_\omega(h_\vartheta(\mathfrak{s}_{s,i})) + \frac{1}{m}\sum_{i=1}^m f_\omega(h_\vartheta(\mathfrak{s}_{t,i}))$
      $+ \mathscr{L}(\mathfrak{s}_{s,1}, a_{s,1}, r_{s,1}, ..., \mathfrak{s}_{s,m}, a_{s,m}, r_{s,m})]$
    $\vartheta \leftarrow \vartheta + \alpha \times RMSProp(\omega, \varsigma_\vartheta)$
  end for
end while

5. EXPERIMENT

We show the effectiveness of our approach in transfer performance compared to PPO and WAPPO algorithms in Visual Cartpole and 16 OpenAI Procgen experimental environments [36, 37]. We adopt the neural network framework provided by [10, 38] as the baseline (as shown in Figure 1). This framework consists of the convolutional neural network element from the Impala net, which is shared between the strategy and value elements of Proximal Policy Optimization(PPO). The value and strategy network then branch each have one FC level which outputs the strategy or value, separately.

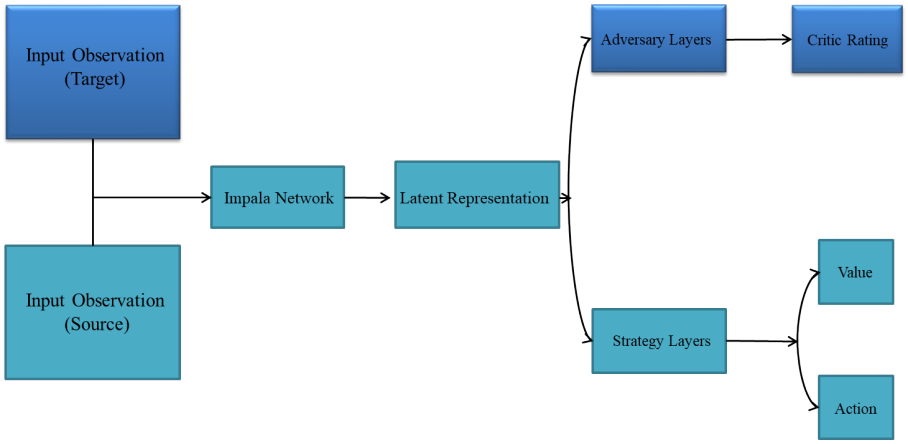


FIGURE 1. Network framework.

We adopt the reward of each mission and the standardization reward of all tasks as the performance evaluation criteria. The normalized return is obtained by averaging the standardization rewards of each mission, $R_{stand} = \frac{(R-R_{min})}{(R_{max}-R_{min})}$, where R represents the reward vector, R_{min} and R_{max} means each environment respectively minimum and maximum returns. This score evaluates how well each algorithm performs on the source and target tasks.

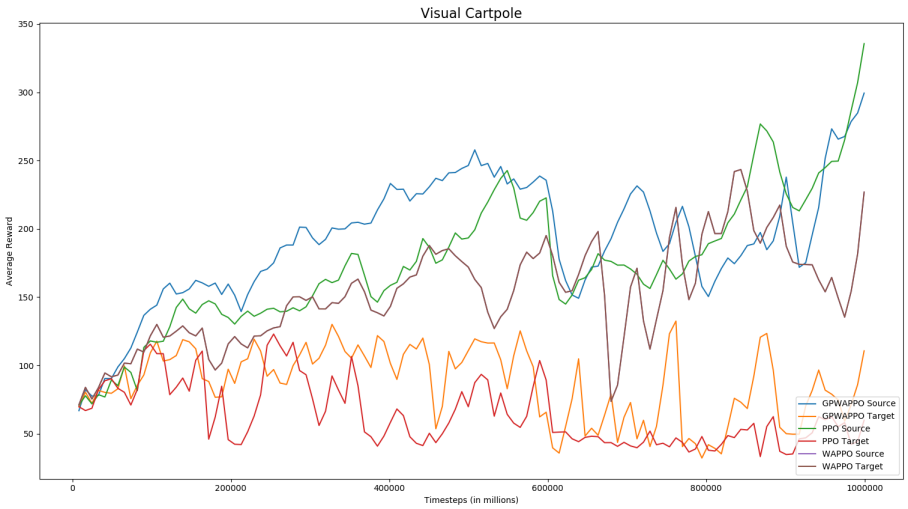
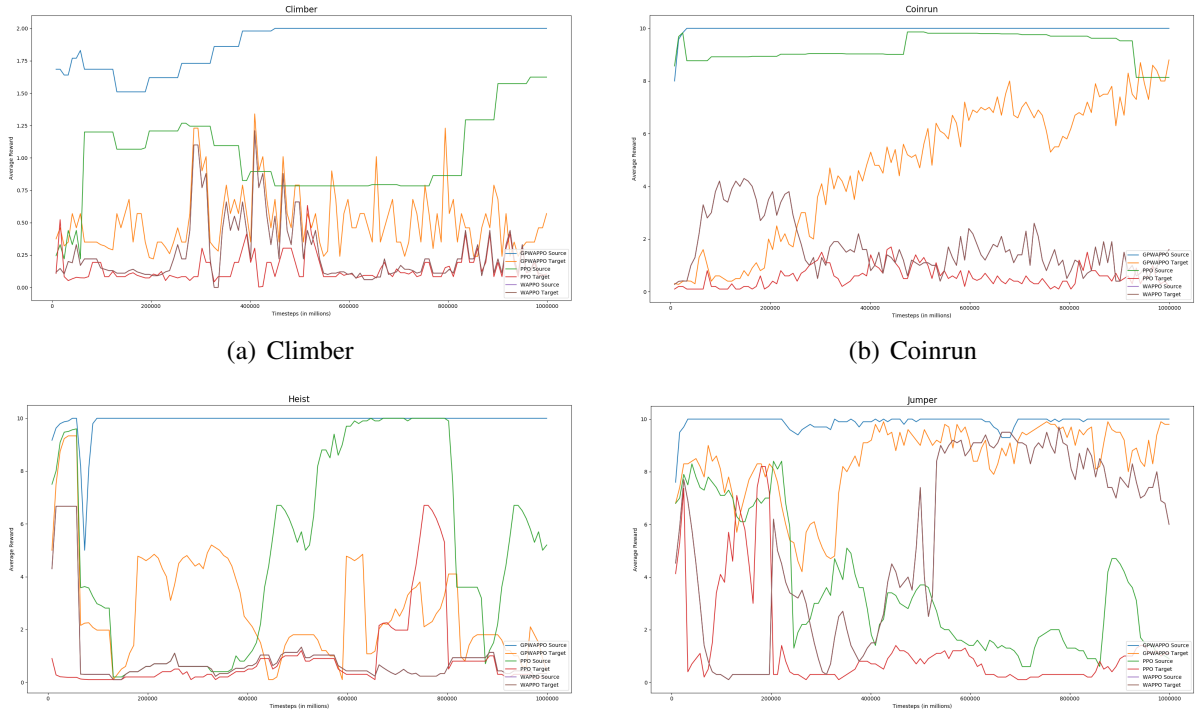


FIGURE 2. Experimental results on Visual Cartpole.

5.1. **Visual cartpole.** Firstly, we present the expected average score on the experimental environment Visual Cartpole, which is a variation of the normal Cartpole domain [36], where the observations are the Cartpole's RGB image instead of position and velocity. The colors of carts, poles, backgrounds, tracks, and axles are different in different domains. In this experimental environment, we adopt the learning rate of 5×10^{-4}. We also use Adam Optimizer [39] and RMSProp Optimizer [40] for training the RL agent in the A2C, PPO, WAPPO, and GPWAPPO. The adversarial network consists of 9 stacked FC layers of width 512 divided by Leaky ReLU activation units. Similar to WAPPO [10], when using GPWAPPO, the similar densities in the target and source features are clustered together to align their distributions, eventually leading to higher rewards. As shown in Figure 2, we observe that our algorithm performs markedly better than other approaches in terms of the average expected return.

5.2. **OpenAI Procgen.** Then, we present the performance on the 16 domains that come from OpenAI benchmark [37]. In these experimental environments, we adopt the learning rate of 2×10^{-4}. We also use Adam Optimizer [39] and RMSProp Optimizer [40] for training the RL agent in the A2C, PPO, WAPPO, and GPWAPPO. The adversarial network consists of 8 dense layers of width 512, divided by Leaky ReLU active function units. As shown in Figure 3-5, we observe that our approach achieves significantly better than other methods in terms of the average expected return in 16 environments.



(a) Climber (b) Coinrun

(c) Heist (d) Jumper

FIGURE 3. Experimental comparison results on Climber and Coinrun etc.

(a) Bossfight (b) Bigfish

(c) Leaper (d) Maze

(e) Miner (f) Ninja

(g) Plunder (h) Startpilot

FIGURE 4. Experimental comparison results in empirical domains Bossfight and Bigfish etc.

(a) Dodgeball (b) Fruitbot

(c) Caveflyer (d) Chaser

FIGURE 5. Experimental comparison results in empirical domains Dodgeball and Fruitbot etc.

6. CONCLUSION

In this paper, we presente Gradient Penalty based Wasserstein Adversarial Proximal Policy Optimization (GPWAPPO), a new approach that learns to match distributions of distilled characteristics between an originating and objective task without accessional extra calculation during deduction or supposing access to coupled relations of states. To reinforce the Lipschitz restriction, we introduce a new objective function, Gradient Penalty based Wasserstein Confusion (GPWC). When using GPWAPPO, the similar densities in the target and source features are clustered together to align their distributions, eventually leading to higher rewards. In this way, our method surpasses the previous visual transfer methods and triumphantly transfer strategies across Visual Cartpole and 16 OpenAI Procgen environments.

Acknowledgments

The authors are grateful to the referees for the useful suggestions and comments which improved this paper significantly. Xianchao Zhu and Xiaoting Wang were supported by the National Key R & D Program of China under Grant No. 2018YFA0306703. This work was also supported by the National Nature Science Foundation of China under Grant No. 61772120.

REFERENCES

[1] R.S. Sutton, A.G. Barto, Reinforcement Learning: An introduction, MIT Press, 2018.
[2] M. Mnih et al., Human-level control through deep reinforcement learning, Nature 518 (2015), 529-533.

[3] K. Cobbe, et al., Quantifying generalization in reinforcement learning, Proceedings of the 36th International Conference on Machine Learning, 97 (2019), 1282-1289.
[4] K. Cobbe, et al., Leveraging procedural generation to benchmark reinforcement learning, Proceedings of the 37th International Conference on Machine Learning, 119 (2020), 2048-2056.
[5] F. Sadeghi, et al., Sim2Real viewpoint invariant visual servoing by recurrent control, IEEE Conference on Computer Vision and Pattern Recognition, pp. 4691-4699, 2018.
[6] J. Tobin et al., Domain randomization for transferring deep neural networks from simulation to the real world, IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 23–30, 2017.
[7] Niels Justesen and Ruben Rodriguez Torrado and others, Illuminating generalization in deep reinforcement learning through procedural level generation, NeurIPS Workshop on Deep Reinforcement Learning, Montréal, 2018.
[8] N. Srivastava et al., Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014), 1929–1958.
[9] A.Y. Ng, Feature selection, L_1 vs. L_2 regularization, and rotational invariance, Proceedings of the twenty-first international conference on Machine learning, pp. 78, 2004.
[10] J. Roy, G. Konidaris, Visual transfer for reinforcement learning via Wasserstein domain confusion, Thirty-Fifth AAAI Conference on Artificial Intelligence, pp. 9454–9462, 2021.
[11] P.Y. Simard, Best practices for convolutional neural networks applied to visual document analysis, 7th International Conference on Document Analysis and Recognition, pp. 958–962, 2003.
[12] E. Tzeng et al., Adversarial discriminative domain adaptation, IEEE Conference on Computer Vision and Pattern Recognition, pp. 2962-2971, 2017.
[13] Y. Ganin et al., Domain-adversarial training of neural networks, J. Mach. Learn. Res. 17 (2016), 1-35.
[14] E. Tzeng, Simultaneous deep transfer across domains and tasks, IEEE International Conference on Computer Vision, pp. 4068-4076, 2015.
[15] M. Andrychowicz, et al., Learning dexterous in-hand manipulation, Int. J. Robot. Res. 39 (2020), 3-20.
[16] A. Gupta et al., Learning invariant feature spaces to transfer skills with reinforcement learning, 5th International Conference on Learning Representations, 2017.
[17] E. Tzeng et al., Adapting deep visuomotor representations with weak pairwise constraints, In: K. Goldberg et al. (ed.) Algorithmic Foundations of Robotics XII, Springer Proceedings in Advanced Robotics, vol 13. Springer, Cham, 2020.
[18] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein generative adversarial networks, Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 214-223, 2017.
[19] Y. Rubner, C. Tomasi, L.J. Guibas, The earth mover's distance as a metric for image retrieval, Int. J. Comput. Vision 40 (2020) 99-121.
[20] M.E. Taylor, P. Stone, Transfer learning for reinforcement learning domains: A survey, J. Mach. Learn. Res. 10 (2009), 1633-1685.
[21] A. Lazaric, Transfer in reinforcement learning: A framework and a survey, In: M. Wiering, M. van Otterlo (ed.) Reinforcement Learning. Adaptation, Learning, and Optimization, vol. 12., pp. 143-173, Springer, Berlin, Heidelberg, 2021.
[22] T. Carr, M. Chli, G. Vogiatzis, Domain adaptation for reinforcement learning on the atari, Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, pp. 1859-1861, Montréal, C2018.
[23] T. Killian et al., Robust and efficient transfer learning with hidden parameter markov decision processes, Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, pp. 4949-4950, Long Beach, CA, 2017.
[24] J. Yao et al., Direct policy transfer via hidden parameter markov decision processes, The 2nd Lifelong Learning: A Reinforcement Learning Approach (LLARLA) Workshop, Stockholm, Sweden, 2018.
[25] F. Doshi-Velez, G. Konidaris, Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations, Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, pp. 1432-1440, 2016.
[26] S. Mysore, R. Platt, K. Saenko, Reward-guided curriculum for robust reinforcement learning, Workshop on Multi-task and Lifelong Reinforcement Learning at ICML, 2019.

[27] S.S. Du, Provably efficient RL with rich observations via latent state decoding, Proceedings of the 36th International Conference on Machine Learning, pp. 1665-1674, 2019.
[28] Monahan, George E, State of the art—a survey of partially observable Markov decision processes: theory, models, and algorithms, Manage Sci. 28 (1982) 1-16.
[29] K. Cobbe, Leveraging procedural generation to benchmark reinforcement learning, Proceedings of the 37th International Conference on Machine Learning, vol. 119, pp. 2048-2056, 2020.
[30] J. Oh et al., Action-conditional video prediction using deep networks in Atari games, Annual Conference on Neural Information Processing Systems, pp. 2863-2871, 2015.
[31] Judy Hoffman and Eric Tzeng and others, CyCADA: Cycle-Consistent Adversarial Domain Adaptation, Proceedings of the 35th International Conference on Machine Learning, vol. 80, pp. 1994-2003, 2018.
[32] J.Y. Zhu et al., Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks, IEEE International Conference on Computer Vision, pp. 2242-2251, 2017.
[33] Y. Jing, Neural Style Transfer: A Review, IEEE Trans. Vis. Comput. Graph 26 (2020), 3365-3385.
[34] L.A. Gatys, A.S. Ecker, M. Bethge, A neural algorithm of artistic style, J. Vis. 16 (2016), 326-326.
[35] I. Gulrajani et al., Improved training of Wasserstein GANs, Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 5769-5779, 2017.
[36] P. Dhariwal et al., Openai baselines, 2017. https://github.com/openai/baselines.
[37] K. Cobbe et al., Leveraging procedural generation to benchmark reinforcement learning, Proceedings of the 37th International Conference on Machine Learning, vol. 119, pp. 2048-2056, 2020.
[38] L. Espeholt et al., Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures, Proceedings of the 35th International Conference on Machine Learning, pp. 1407-1416, 2018.
[39] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 3rd International Conference on Learning Representations, San Diego, 2015.
[40] T. Tieleman, G. Hinton, Geoffrey and others, Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude, COURSERA: Neural networks for machine learning, vol. 4, pp. 26-31, 2012.