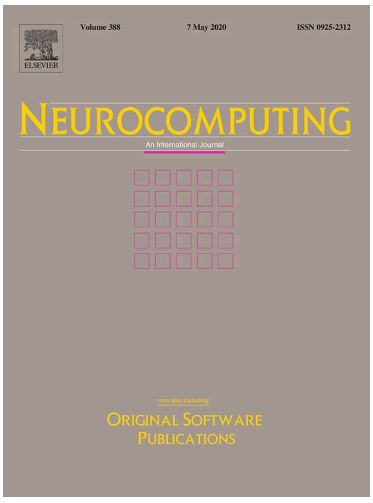
Journal Pre-proofs



Random Curiosity-driven Exploration in Deep Reinforcement Learning

Jing Li, Xinxin Shi, Jiehao Li, Xin Zhang, Junzheng Wang

PII: S0925-2312(20)31284-4
DOI: https://doi.org/10.1016/j.neucom.2020.08.024
Reference: NEUCOM 22683

To appear in: *Neurocomputing*

Received Date: 31 December 2019
Accepted Date: 11 August 2020

Please cite this article as: J. Li, X. Shi, J. Li, X. Zhang, J. Wang, Random Curiosity-driven Exploration in Deep Reinforcement Learning, *Neurocomputing* (2020), doi: https://doi.org/10.1016/j.neucom.2020.08.024

Random Curiosity-driven Exploration in Deep Reinforcement Learning

Jing Li[a,b], Xinxin Shi[a,b], Jiehao Li[a,b*], Xin Zhang[a,b] and Junzheng Wang[a,b]

[a] *State Key Laboratory of Intelligent Control and Decision of Complex Systems, Beijing Institute of Technology, Beijing, 100081, China;*

[b] *Key Laboratory of Servo Motion System Drive and Control, Beijing, 100081, China;*

**Corresponding: jiehao.li@bit.edu.cn*

Abstract

Reinforcement learning (RL) depends on carefully engineering environment rewards. However, rewards from environments are extremely sparse for many RL tasks, challenging for the agent to learn skills and interact with the environment. One solution to this problem is to create intrinsic rewards for agents and to make rewards dense and more suitable for learning. Recent algorithms, such as curiosity-driven exploration, usually estimate the novelty of the next state through the prediction error of dynamics models. However, these methods are typically limited by the capacity of their dynamics models. In this paper, a random curiosity-driven model using deep reinforcement learning is proposed, which uses a target network with fixed weights to maintain the stability of dynamics models and create more suitable intrinsic rewards. We integrate the parametric exploration method for further promoting sufficient exploration. Besides, a deeper and more closely connected network is utilized for encoding the pixel images for policy-gradient. By comparing our method against the previous approaches in several environments, the experiments show that our method achieves state-of-the-art performance on most but not all of the Atari games.

Keywords: Deep reinforcement learning, curiosity-driven exploration, intrinsic rewards

1. Introduction

Reinforcement learning has demonstrated remarkable performance by maximizing the sum of future rewards through learning policies in Atari games [1, 2, 3], 3D navigation tasks [4, 5], robotic arm grabbing objects [6, 7], and robot locomotion tasks [8, 9]. The dense and well-defined reward function can help the agent understand the task and learn skills that might be useful later in life. For example, Rajeswaran et al. [10] proposed a shaping reward function that was less sparse than an original reward function, so it leads to faster learning of a good policy. However, reward engineering or reward shaping can lead to unexpected or even bad results when the reward function is modified slightly. Designing a well-defined reward function becomes a notoriously challenging engineering problem. This well-designed reward function is contrary to the original intention of reinforcement learning to learn good behavior from a limited reward signal. In some real-world scenarios, the rewards coming from the environment and being extrinsic to the agent are extremely sparse or missing altogether, making it impossible to construct a shaped reward function. Standard reinforcement learning algorithms struggle with this sparse reward tasks because exploration behavior from entropy maximization or epsilon-greedy action selection actions is not enough. Encouraging exploration can help solve the problem of sparse reward.

Recently, multiple approaches are widely to achieve better exploration, and they can fall broadly into two categories of guided exploration. Imitation learning [11, 12] has been proposed to guide exploration by observing a human demonstrator generate state-action trajectories. Various imitation learning work focuses on pre-training a Q-function [11] or actor-critic architectures [13] with human demonstrations, which can achieve impressive performance. However, there are some limitations to these approaches. For instance, the "Domain gap" between the expert demonstration and the task environment could exist, especially the variations in color or texture, and the introduction of other visual artifacts [14]. The specific action and reward sequences that led to an expert demonstration trajectory were obtained out-and-out. These limitations have led to difficulties in generating high-quality expert demonstrations. The other category is intrinsic motivation methods [15, 16, 17], which provide an auxiliary reward to encourage the agent to explore environments and discover novel states. Motivation occurs when an agent acts without any obvious external rewards or regards it as an

opportunity to explore, learn, and actualize their potentials. Methods with intrinsic motivation include count-based [18, 19, 20, 21], where the agent uses state visitation counts as a measure of novelty and the "novelty" state that has not been seen will have higher rewards. However, the implementation and calculation of these methods are quite complicated, and these tend to become less effective as the number of possible states increases. Surprise [22, 23], which means the inability to predict the future and maximizes a notion of an agent's surprise about its experience. Empowerment [23], which means an information-theoretic formulation of the agent's influence shortly, and the agent enjoys the level of control it has about its future. Curiosity [15, 17, 24, 25, 26, 27, 28, 29], where the agent is more likely to see new states and usually uses predict error as a reward signal.

However, the intrinsic motivation methods such as count-based exploration [18, 19] and curiosity-driven exploration [15, 17] usually overlook the visited paths and repeatedly visit those new but actually old states, and this is called the catastrophic forgetting problem [30]. For example, the density model in count-based exploration [19] and the dynamic model in the curiosity-driven method are gradually changed as they are trained. The model parameters differed greatly between now and the beginning time, and the model would forget the state that was already experienced. In essence, a catastrophic forgetting problem occurs when the later model forgets the previous exploration, and the understanding of the "novel" creates an error. This catastrophic forgetting is usually related to the fact that the curiosity model needs enough plasticity to digest new information, but large weight changes will destroy the previous learning representation and lead to forgetting. In this paper, we propose a new curiosity-driven module called random curiosity module (RCM) to make an effort to the catastrophic forgetting challenge. A target network with fixed weights is proposed after random initialization, limiting the large change of the dynamic model weight and allowing the dynamic model to learn new information. Intrinsic reward from RCM is composed of state prediction error and feature embedding network bias. Nevertheless, another challenge of curiosity exploration is the "couch-potato" problem, which means that the agent trying to maximize the prediction error will tend to seek out stochastic transitions, such as switching the TV channels endlessly. We propose a solution to this undesirable stochasticity by fusing RCM and learning intrinsic rewards for policy gradient methods (LIRPG) [31] and predicting value function. Instead of simply seeking the unpredictable state or action and adding the intrinsic reward to extrinsic

reward, the proposed approach is learning parametric intrinsic rewards for a policy-gradient based learning agent, which updates the intrinsic rewards parameters to optimize the extrinsic rewards achieved by the policy. Another intrinsic learning reward achieved by RCM that, when added to the extrinsic rewards, can improve RL problems' performance. In addition, we use the predicted next state of RCM to predict value function and keep value prediction consistent with real value function from the policy model by minimizing prediction errors. We have adopted a deeper and more connected network instead of Nature-CNN [1], and can promote the agent's learning ability. The primary contributions of this paper are summarized as the following:

- An enhanced curiosity-driven version, called RCM, is presented to deal with the catastrophic forgetting problem of curiosity-driven exploration. RCM is based on a fixed randomly initialized network that limits the large change in feature spaces and ensures stable online training of dynamics.
- A RCM exploration and the learning parametric intrinsic rewards is utilized to improve the performance of policy-gradient based RL methods.
- For different feature spaces for policy networks, a deeper and more tightly connected network is proposed to promote the skills learning of agents. We evaluate our method on several Atari games with A2C (Advantage Actor-Critic) agents and show that our method can outperform several other exploration techniques.

2. Related Work

Policy gradient-based RL. The advances of RL in recent years are mostly based on policy gradients. The advantage of policy gradient-based approaches is that they directly optimize the quantity of interest while remaining stable under function. In order to improve sample efficiency, actor-critic methods have become popular, and they use value approximators to replace rollout estimates to reduce variance. A number of improved algorithms based on policy gradients have been proposed to improve the performance, such as A2C, A3C(Asynchronous advantage actor-critic) [4], PPO(Proximal Policy Optimization Algorithms) [32] and ACKTR(Actor Critic using Kronecker-Factored Trust Region) [33]. The RCM and parametric exploration methods

proposed in this paper are based on the policy gradient RL framework, and we choose A2C as the policy architecture. Actually, our method can be combined with multiple policy-based algorithms.

Reward shaping and Imitation learning. Several ways of using intrinsic rewards for exploration have been proposed in reinforcement learning papers. For example, Ng et al. [34] proposed reward shaping function to improve the optimal policy, and some auxiliary rewards designing methods have been proposed to derive policy with desired properties. Sorg et al. [35] introduced a policy gradient for reward design, which only operates with lookahead-search based planning agents. The rewards shaped or designed can assist agents to explore environments, but computing the optimal reward is still a big challenge. Imitation learning for exploration methods have used in sparse-reward tasks [11, 12, 36] and have good performance. Other attempts [37, 38, 39] have been made to introduce the expert demonstrations and trajectory preferences, and the agent is trained to imitate the demonstrator. Moreover, Schmidhuber et al. [40] studied a method that improved performance through the reward transformations. However, reward transformations are expert-designed and not learned. Getting high-quality expert information is difficult. Besides, these methods do not operate in sparse reward settings. The main contribution of this paper is to learn the parameters of an intrinsic reward function and integrate RCM that can provide dense rewards for agents.

Random features. Random features have made surprising effectiveness in RL problems [41], and the theoretical study of the random network has been widely used in recent years. Features of randomly initialized neural networks have been applied in the context of exploration [29, 42], and classification [43, 44, 45].

Intrinsic motivation. Intrinsic motivation has been developed to explain the need to explore the environment and discover novel states [15]. A CTS density model was presented in [20] to measure uncertainty and derive a pseudo-count from the density model as the intrinsic motivation. Ostrovski et al. [19] also explained the importance of the quality of the density model for exploration and combined PixelCNN pseudo-counts with different agent architectures to improve the performance of several hard Atari games. Many other visitations counts in [18, 21, 46] have also been investigated for exploration. However, visitation counts exploration usually need a high-quality density model obtained difficultly, and can not work well when the number of visit states is too large. Prior work on surprise in [22, 47, 48] focuses on

maximizing a notion of the agent surprise about its experiences via intrinsic motivation. For instance, Gregor et al. [23] proposed empowerment for exploration. Curiosity-driven exploration includes prediction error [15, 16, 26, 49], prediction uncertainty [24, 50], improvement [51, 52], and information gain [17].

This paper aims at the curiosity-driven exploration using deep reinforcement learning. Prior curiosity-based methods maximize the prediction error or prediction uncertainty. As a result, the agent would be driven to seek "novel" states, which might lead to the "couch-potato" problem and gives rise to complex or invalid behaviors. This is because of the limitation of stochastic dynamics. When the transitions in the environment are random, the predict error or uncertainty will increase and promote the continuous occurrence of random transition, resulting in the final fall into the "couch-potato" dilemma. Such curiosity methods have also suffered from unstable feature spaces, resulting in the agents forgetting the visited states and exploring those states repeatedly, which leads to a sudden drop in performance. The catastrophic forgetting is related to large weight changes that will destroy previous learning and lead to forgetting. In this paper, we mainly center on the curiosity framework, and a fixed network is utilized to limit excessive changes in the feature space. It can combine the learning parametric intrinsic rewards method to prevent the agent from falling into a catastrophic forgetting dilemma and "couch-potato" problem. Firstly, a target network with fixed weights after random initialization is applied to the RCM, limiting the substantial change of weights and reducing forgetting. Second, a value function prediction is used to delivers RCM consistent with the A2C module, avoiding meaningless random transitions. Finally, a solution to solve this undesirable stochasticity by combing RCM and LIRPG is discussed.

3. Material and Methods

Consider a standard reinforcement learning agent that interacts with an environment. At each timestep t, the agent sees an observation s_t, and takes an action a_t sampled from a parametric distribution π. The environment returns a reward r_t and switches to state s_{t+1} at time $t+1$. The external reward r_t obtained from the environment is sparse or missing in many real-world scenarios. Researchers in recent years came up with solutions to those environments with sparse rewards by providing agents with intrinsic rewards for exploration. However, curiosity-driven [15] or count-based [19]

have "couch-potato" and catastrophic forgetting problems. In order to solve these problems and provide more effective intrinsic reward signal, our new intrinsic reward has two components: (a) a curiosity reward r_t^c from RCM, which is added with external reward r_t^e from environment resulting in r_t to be maximized by the policy. (b) a learning parametric reward r_η^{in}, where the extrinsic rewards r^{ex} are the sum of r_t^e and r_t^c. The policy is updated to maximize both extrinsic rewards r^{ex} and intrinsic rewards r_η^{in}, while the intrinsic reward function is updated to maximize only the extrinsic reward. For better sample-efficient and more reliable performance, we replace the Nature-CNN policy network with a deeper and more tightly connected network. Our work can potentially be applied with a range of policy gradient methods such as A2C, A3C, PPO, and so on. A2C is an asynchronous, deterministic variant of A3C, which waits for each actor to finish its segment of experience before performing an update, averaging over all of the actors. Researchers from OpenAI have found that A2C performs better than A3C. A2C is more cost-effective than A3C when using single-GPU machines, and is faster than a CPU-only A3C implementation when using larger policies. In this paper, we choose the popular A2C as the benchmark algorithm. There are many advances in policy gradient algorithms, and our method is also compatible with such policy gradient architectures, such as A3C and PPO.

3.1. Random curiosity module (RCM)

The curiosity-driven method that the novelty of state is evaluated based on the prediction error of the forward dynamics model performs well on sparse rewards tasks. However, there is still a challenging problem called catastrophic forgetting. Catastrophic forgetting is often known and familiar in machine learning. The occurrence of catastrophic forgetting is related to the fact that the forward dynamics model's parameters are evolving with the progress of training, and the features are changing as they learn. The previous states have seen early become unrecognizable and novel after numerous timesteps of exploring and training. Then the understanding of "novel" has created errors, and sudden drop of performance caused by forgetting occurs. Due to insufficient information on the forward dynamics model, prediction tends to fail in large and high-dimensional state spaces. We propose a new curiosity-driven frame called a random curiosity module to address the catastrophic forgetting problem.

Random feature is inspired by an uncertainty quantification method introduced in [42], and the random network presents a practical and straight-

forward approach to encoding high-dimensional states to latent spaces with deep reinforcement learning.

RCM uses a fixed randomly initialized network to limit the unstable changes in the dynamic model, as shown in Figure 1. This involves two neural networks and three sub-modules: the first network encodes the raw pixel state s_t into a feature vector $\phi(s_t)$ and the second network is a fixed and randomly initialized target network; the first sub-module is forward dynamics model that takes $\phi(s_t)$ and a_t as inputs to predict the next state $\hat{\phi}(s_{t+1})$; the second sub-module is a inverse dynamics model that takes $\phi(s_t)$ and $\phi(s_{t+1})$ as inputs to predict $\hat{a}_t$, and the last sub-module is a constraint model which uses a fixed target network to limit the large changes in feature spaces. The curiosity intrinsic rewards r_t^c is defined as,

$$r_t^c = r_1 + r_2, \tag{1}$$

$$r_1 = \frac{\lambda}{2}\left\|\hat{\phi}(s_{t+1}) - \phi(s_{t+1})\right\|_2^2, \tag{2}$$

$$r_2 = \frac{1}{2}\left\|\varphi(s_{t+1}) - \phi(s_{t+1})\right\|_2^2, \tag{3}$$

where $\hat{\phi}(s_{t+1})$ is the predicted state in feature space from forward dynamics model. $\varphi(s_{t+1})$ is the output of the target network, and it is the state in feature space at timesteps $t+1$. The output of the target network is stable and not changed over time on account of the fixed parameters of the target network. r_1 is measured by the next state prediction error, and the larger the error, the better the novelty and return rewards are higher. r_2 is the uncertainty quantification of feature encoding network, and can also detect the novelty. The forward model parameters θ^F is optimized by minimizing

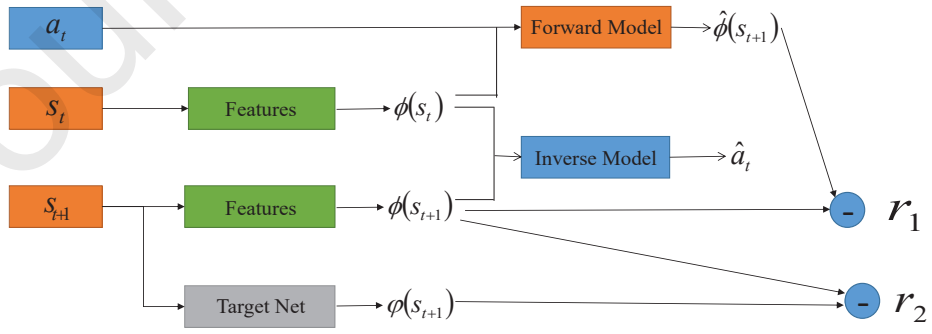


Figure 1: Random curiosity module.

the loss function L_F:

$$L_F = L_P + L_R, \tag{4}$$

$$L_P = \frac{1}{2}\left\|\hat{\phi}\left(s_{t+1}\right) - \phi\left(s_{t+1}\right)\right\|_2^2, \tag{5}$$

$$L_R = \frac{1}{2}\left\|\varphi\left(s_{t+1}\right) - \phi\left(s_{t+1}\right)\right\|_2^2. \tag{6}$$

We also use inverse dynamics model to predict action given the previous s_t and s_{t+1}, and it is optimized by minimizing the loss function L_I which is the inverse loss that measures the discrepancy between the predicted and actual actions. L_I denotes the standard soft-max cross entropy function,

$$L_I = H\left(a_t, \hat{a}_t\right). \tag{7}$$

The inverse dynamics model uses a common network ϕ to embed s_t and s_{t+1}. Therefore, feature learning is more sufficient and stable. The inverse dynamics model guarantees the correctness of feature learning through the prediction action, and the target network guarantees the stability of feature learning through the constraint feature embedding network ϕ. In order to further ensure the accuracy and stability of the next state prediction, value function prediction is considered in this paper. RCM predicts the next state $\hat{\phi}\left(s_{t+1}\right)$, and this includes a plan for future. The A2C algorithm has a value function $V^{\pi}\left(s_t\right)$, which sums up all the expected discounted future rewards. So, this value function should be consistent with the prediction of the next state. As Brunner et al. [53] said, value function includes a forecast of future rewards. Consider the value function:

$$V^{\pi}\left(s_t\right) = E_{\pi}\left[\sum_{k=0}^{\infty}\gamma^k r_{t+k}\right] = E_{\pi}\left[r_t\right] + \gamma V^{\pi}\left(s_{t+1}\right). \tag{8}$$

So, the value function at time $t+1$ is,

$$V^{\pi}\left(s_{t+1}\right) = \frac{V^{\pi}\left(s_t\right) - E_{\pi}\left[r_t\right]}{\gamma}, \tag{9}$$

where the $\gamma^k r_{t+k}$ means the sum of future reward and γ is the discount factor, usually setting as 0.99. At time t RCM and A2C model have experienced the same information. Thus, the actual value function $V^{\pi}\left(s_{t+1}\right)$ and the predicted value function $V^{\pi}\left(\hat{\phi}\left(s_{t+1}\right)\right)$ should be consistent with each other.

The actual value function with input features $\phi(s)$ is denoted by $V^{\pi}(\phi(s))$. We use the output of forward dynamics model $\hat{\phi}(s_{t+1})$ to predict value function at time $t+1$, and optimize the value prediction network parameters θ^{V} by minimizing the loss function L_V,

$$L_V = \frac{1}{2}\left\| V^{\pi}\left(\hat{\phi}(s_{t+1})\right) - V^{\pi}(\phi(s_{t+1}))\right\|_2^2, \tag{10}$$

$$= \frac{1}{2}\left\| V^{\pi}\left(\hat{\phi}(s_{t+1})\right) - \frac{V^{\pi}(\phi(s_t)) - E_{\pi}[r_t]}{\gamma}\right\|_2^2. \tag{11}$$

$E_{\pi}[r_t]$ is difficult to calculate, because at each state there is only one action that can be taken. When policy π is adopted, the reward $\bar{r}_t$ that is obtained in each step is the unbiased sample of random variable r_t. Thus, we use $\bar{r}_t$ as an approximation of $E_{\pi}[r_t]$. The representation of immediate reward in this paper is r_t, which is the sum of intrinsic and extrinsic rewards. In RCM, r_t is the sum of r_t^c and r_t^e. In next section, r_t is the sum of r_t^c, r_t^e and r_{η}^{in}. The value prediction loss can be calculated as,

$$L_V \approx \frac{1}{2}\left\| V^{\pi}\left(\hat{\phi}(s_{t+1})\right) - \frac{V^{\pi}(\phi(s_t)) - r_t}{\gamma}\right\|_2^2. \tag{12}$$

The overall optimization problem that is solved by RCM is a composition of L_F, L_I and L_V and can be written as,

$$min\left[-\lambda_P E_{\pi}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right] + \lambda_I L_I + \lambda_F L_F + \lambda_V L_V\right], \tag{13}$$

where λ_P, λ_I, λ_F and λ_V are the weights of each loss function.

3.2. Combining learning parametric rewards

From the perspective of implementation means, exploration can be divided into two types: one is state-action exploration that systematically explores state or action space such as picking different action a_t each time s_t is visited; the other is parameter exploration that parameterizes policy $\pi(a_t|s_t;\mu)$ such as picking different parameters and trying for a while. Parameter exploration method has the advantage of consistent exploration, but does not know the state-action space. One of the methods of state-action space exploration is curiosity exploration that is adding intrinsic rewards to

extrinsic rewards and maximize the sum of the intrinsic rewards and the extrinsic rewards, but there is no direct participation in policy-gradient updates. Learning parametric rewards building on the optimal rewards framework [31] belongs to parameter exploration method, which updates the policy parameters θ and the intrinsic reward parameters θ'. We will briefly describe how learning parametric rewards based policy-gradient RL works, and then we will present our method that incorporates it. Policy gradient based RL is directly calculating the direction in which the gradient may be updated to maximize the external rewards. The policy π_θ is a representation of states transforming to a probability distribution over actions. The value of a policy is denoted $J(\theta)$, is the expected discounted sum of rewards when agent executes actions according to policy π_θ:

$$J(\theta) = E_{s_t,\pi_\theta}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right]. \tag{14}$$

The gradient of the value J can be computed as,

$$\bigtriangledown_\theta J(\theta) = E_\theta\left[G(s_t,a_t)\bigtriangledown_\theta log\pi_\theta(a_t|s_t)\right], \tag{15}$$

where $G(s_t,a_t) = \sum_{i=t}^{\infty}\gamma^{i-t}r_i$ is the return until termination. This $G(s_t,a_t)$ is a simple policy gradient formulation in order to simplify the calculation. In recent advanced algorithms, $G(s_t,a_t)$ can use the advantage function $A^\pi(s_t,a_t)$ or TD-error $r_t + V^\pi(s_{t+1}) - V^\pi(s_t)$ instead of the return to reduce the variance. For example, A2C and A3C both learn an advantage critic,

$$G(s_t,a_t) = A(s_t,a_t) = \sum_{i=0}^{k-1}\gamma^i r_{t+i} + \gamma^k V(s_{t+k}) - V(s_t). \tag{16}$$

We use $r_\eta^{in}(s_t,a_t)$ to represent the intrinsic reward function from LIRPG, and η to represent the intrinsic reward parameters. The extrinsic reward in LIRPG is expressed as r^{ex}, which is the sum of r^e from the environment and r^c from RCM. The discounted returns from different rewards are defined as:

$$G^{ex}\left(s_t, a_t\right) = \sum_{i=0}^{k-1} \gamma^i r_{t+i}^{ex} + \gamma^k V\left(s_{t+k}\right) - V\left(s_t\right), \tag{17}$$

$$G^{in}\left(s_t, a_t\right) = \sum_{i=0}^{k-1} \gamma^i r_{t+i;\eta}^{in}\left(s_i, a_i\right) + \gamma^k V\left(s_{t+k}\right) - V\left(s_t\right), \tag{18}$$

$$G^{ex+in}\left(s_t, a_t\right) = \sum_{i=0}^{k-1} \gamma^i \left(r_{t+i}^{ex} + \lambda r_{t+i;\eta}^{in}\left(s_i, a_i\right)\right) + \gamma^k V\left(s_{t+k}\right) - V\left(s_t\right), \tag{19}$$

where λ is relative weight of intrinsic reward. The intrinsic rewards r_η^{in} serve only to influence the change in policy parameters. $J\left(\theta\right)$ can be rewritten as:

$$J^{ex} = E_\theta\left[\sum_{t=0}^{\infty} \gamma^t r_t^{ex}\right], \tag{20}$$

$$J^{in} = E_\theta\left[\sum_{t=0}^{\infty} \gamma^t r_\eta^{in}\left(s_t, a_t\right)\right], \tag{21}$$

$$J^{ex+in} = E_\theta\left[\sum_{t=0}^{\infty} \gamma^t \left(r_t^{ex} + \lambda r_\eta^{in}\left(s_t, a_t\right)\right)\right]. \tag{22}$$

We first use the sum of extrinsic and intrinsic rewards as the reward to update θ in the direction of J^{ex+in},

$$\theta' = \theta + \alpha \bigtriangledown_\theta J^{ex+in}\left(\theta\right) \tag{23}$$

$$\approx \theta + \alpha G^{ex+in}\left(s_t, a_t\right) \bigtriangledown_\theta log\pi_\theta\left(a_t|s_t\right). \tag{24}$$

Then we use the chain rule to update intrinsic reward parameters η in the direction of J^{ex},

$$\bigtriangledown_\eta J^{ex} = \bigtriangledown_{\theta'} J^{ex} \bigtriangledown_\eta \theta', \tag{25}$$

where the first term is an approximate stochastic gradient of the extrinsic value with respect with θ', and it can be sampled as:

$$\bigtriangledown_{\theta'} J^{ex} \approx G^{ex}\left(s_t, a_t\right) \bigtriangledown_{\theta'} log\pi_{\theta'}\left(a_t|s_t\right). \tag{26}$$

When updating η, we need two episode values such as θ'. To improve data efficiency we use off-policy [31] to compute $\bigtriangledown_{\theta'} J^{ex}$:

$$\bigtriangledown_{\theta'} J^{ex} = G^{ex}\left(s_t, a_t\right) \frac{\bigtriangledown_{\theta'} \pi_{\theta'}\left(a_t|s_t\right)}{\pi_{\theta}\left(a_t|s_t\right)}. \tag{27}$$

The second term can be computed as:

$$\begin{aligned}
\bigtriangledown_{\eta}\theta' &= \bigtriangledown_{\eta}\left(\theta + \alpha G^{ex+in}\left(s_t, a_t\right) \bigtriangledown_{\theta} log\pi_{\theta}\left(a_t|s_t\right)\right) && (28)\\
&= \bigtriangledown_{\eta}\left(\alpha G^{ex+in}\left(s_t, a_t\right) \bigtriangledown_{\theta} log\pi_{\theta}\left(a_t|s_t\right)\right) && (29)\\
&= \bigtriangledown_{\eta}\left(\alpha\lambda G^{in}\left(s_t, a_t\right) \bigtriangledown_{\theta} log\pi_{\theta}\left(a_t|s_t\right)\right) && (30)\\
&= \alpha\lambda \sum_{i=0}^{k-1} \gamma^i \bigtriangledown_{\eta} r^{in}_{t+i;\eta}\left(s_i, a_i\right) \bigtriangledown_{\theta} log\pi_{\theta}\left(a_t|s_t\right). && (31)
\end{aligned}$$

The parameters η are updated using the product of Equations 27 and 31 as follows:

$$\eta' = \eta + \beta \bigtriangledown_{\eta} J^{ex}, \tag{32}$$

where α and β are step-size parameters.

3.3. Deeper and more tightly connected network

Nature-CNN using the same three-layer convolutional architecture is proposed in [1] for mapping the high dimensional image inputs to low dimensional space. However, the data efficiency of Nature-CNN is not enough. As is common in supervised learning, using deeper and wider neural networks can make better. IMPALA-CNN or IMPALA-Large [54] use residual blocks add to convolution layers and are deeper and tightly connected, which have success in RL problems. These networks use LSTM as the last part of feature embedding to learn long-term dependencies and remember that long-term information. Although LSTM can help to remember historical information and be more sample efficient, such RNN-based networks generally have difficulty in parameter fine-tuning. Since the network structure is too large and the computing resources are too large, we use a simplified version of the IMPALA network called DMTCN(Deeper and More Tightly Connected Network). As shown in Figure 2, we only use some convolution layers and residual blocks for observation embedding in policy networks. This network incorporates residual blocks, which can solve gradient vanishing in backpropagation when using a deeper and broader network. This deeper network makes it more data-efficient and has a better computational performance.

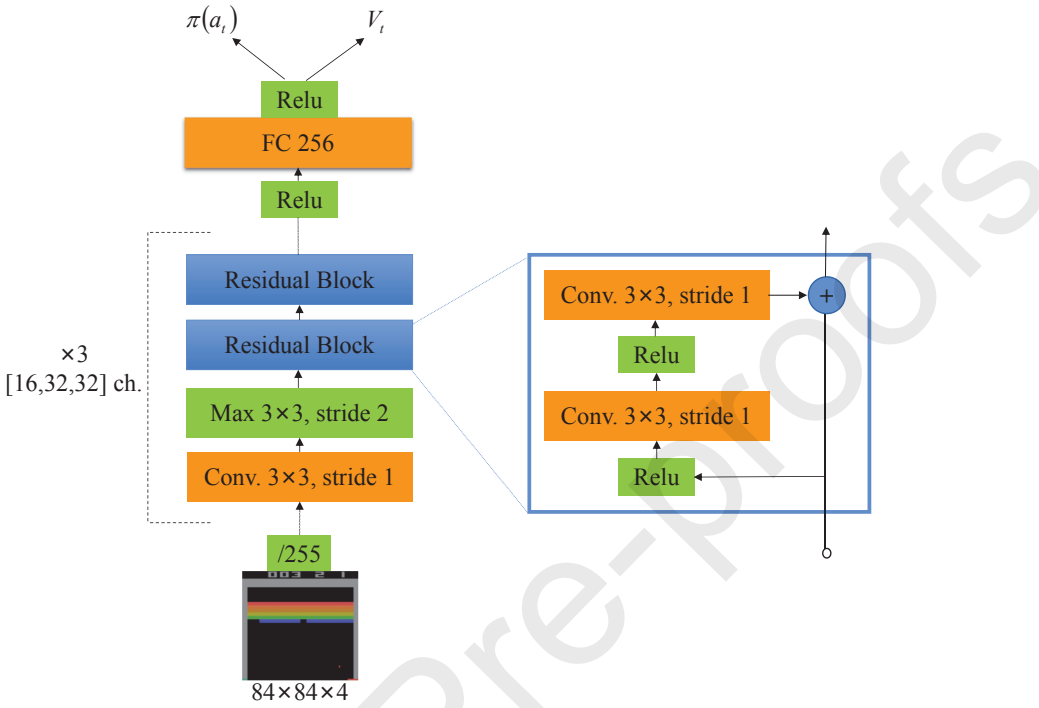


Figure 2: Model Architectures of our deeper and more tightly connected network.

4. Experimental Results

In this section, we evaluate our algorithm on the Arcade Learning Environment [55], including continuous control and discrete control tasks. Both the policy and the feature embedding network work directly from pixels. We take the maximum value at each pixel from 4 consecutive frames to compress them into one frame, which is rescaled to a 84×84 grayscale image. Our proposed method can improve the performance of policy-gradient based learning methods for solving RL problems. In this paper, we use the A2C algorithm [4] that is an open-source implementation and has succeeded in many RL tasks. We compare the proposed methodology against the previous approaches such as Baselines (A2C), ICM, RND, and LIRPG. The input state s_t is passed through a sequence of three convolution layers and one fully connected layer. The first convolution layer has 32 8×8 filters with stride 4. Besides, the second convolution layer has 64 4×4 filters with stride 2. Similarly, the third convolution layer has 64 3×3 filters with stride

1. The fully connected layer with 512 hidden units is the fourth layer. A rectified linear unit(ReLU) is used after each convolution layer or fully connected layer. The output of the fully connected layer is separated into two fully connected layers, which predict the value function and the action from Nature-CNN feature representation. We also keep the default values of all hyper-parameters in the original OpenAI implementation of the A2C-based policy module unchanged for both the augmented and baseline agents. The extrinsic reward used is the game score change, which is standard for the work on Atari games. The intrinsic reward from RCM and the parameter intrinsic reward from parameter exploration are both clipped into [-1,1]. We used RMSProp optimizer with all parameters. The decay factor used for RMSProp is 0.99, and the ϵ is 0.00001.

Random curiosity module results: Figure 3(a) exhibits the improvements of the augmented agents with random curiosity module over baseline agents. We evaluated our RCM on the Atari benchmark, i.e., Tennis. As noted above, to demonstrate the performance of random curiosity exploration, we trained the baseline, ICM, and augmented agents for 50 million steps on each environment. For the augment agents, we explored the environments with a random curiosity module, and the intrinsic reward provided by the RCM module is used and normalized to be superimposed on the extrinsic reward provided by the task. The random curiosity module consists of the forward module, inverse module, value predict module, and the random module. The input state s_t and s_{t+1} are encoded as feature vectors $\phi(s_t)$ and $\phi(s_{t+1})$ using a series of three convolution layers and a fully connected layer. The forward model is constructed by concatenating $\phi(s_t)$ with a_t and passing it into a series fully connected layers with 256 and 288 units. The inverse model is constructed by concatenating $\phi(s_t)$ and $\phi(s_{t+1})$ and passing it into a single fully connected layer and. The RCM and ICM are consistent in the design of the forward and inverse models. For the RCM method, the random model first maps the next time input state s_{t+1} into a fixed feature vector $\varphi(s_{t+1})$ using the same network frame with three convolution layers and a fully connected layer. For the random model, the $\varphi(s_{t+1})$ and $\phi(s_t)$ are concatenated into a single feature vector and made a mean square error. The output of the value predict model is the output of the forward model and passes as inputs into a fully connected layer with 1 unit. The value of loss weights are 2, 8, and 10. As shown in Figure 3(a), we compare the"A2C+RCM" approach, which combines a random curiosity model with A2C with two baselines. First is the "A2C" algorithm with e-greedy explo-

ration. Second is "ICM+A2C" which combines the intrinsic curiosity model with A2C. We see that learning RCM rewards significantly improves the performance of A2C. Due to the weights changes of the dynamics model of ICM and the occurrence of random state transitions, there is almost no performance improvement for the "A2C+ICM" agent. However, our "A2C+RCM" agent uses a target network to keep the dynamics model stable and uses value prediction to avoid random transitions. Thus, our "A2C+RCM" agent has better performance. In the early time steps of training, exploration means that the agent has to experience more unknown states, which are likely to cause bad actions. In the later timesteps of training, exploration behavior is more effective, so learning speed and performance are greatly improved.

Combining parameter exploration results: Our object of this section in the following set of experiments is to evaluate whether augmenting an agent combing parameter exploration and state-action exploration (RCM) improves performance. To this end, we compare our approach with three baselines on multiple Atari games. First is the "A2C" algorithm, and second is the "A2C+RCM". The third is the "A2C+LIRPG" with parameter exploration. Our approach is named the "A2C+LIRPG+RCM". For our approach, the external reward used to update the policy gradient is the sum of the extrinsic reward provided by the task and the intrinsic reward provided by the RCM model. The intrinsic reward from parameter exploration is only to influence the change in policy parameters. We keep the default values of

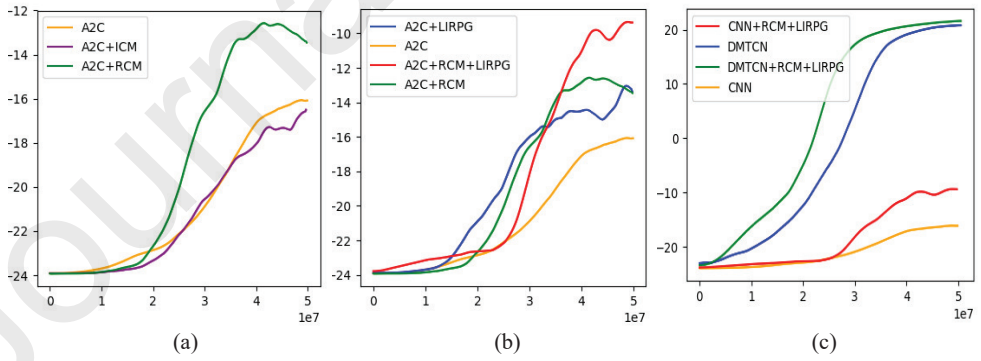


Figure 3: The x-axis is time steps during learning. The y-axis is the extrinsic rewards. (a) We compare our RCM module with baseline and ICM methods. (b) We compare our combing RCM and parameter exploration method with the others. (c) We compare our deeper and more tightly connected network with Nature-CNN.

all hyper-parameters in the parameter exploration module [31] unchanged for both the augmented and baseline agents. Figure 3(b) shows the performance of the four algorithms. RCM and LIRPG both improve the performance compared to "A2C". Our augmenting agent has the best performance compared to the single RCM exploration agent and the single parameter exploration agent.

Comparison of different policy network: Figure 3(c) shows the improvements of the augmented agents with DMTCN policy network over baseline agent with Nature-CNN policy network on Ataris games such as Tennis. For DMTCN, the input state s_t is passed through a sequence of three subnetworks. The filters of three subnets are 16, 32 and 32, respectively. The kernel size of all layers is 3×3. Convolution layers construct each subnetwork with the stride of 1, max-pooling layers with the stride of 2, and two residual blocks. A rectified linear unit(ReLU) is used after each convolution layer. The output of the last subnetwork is fed into a fully connected layer with 256 units. For Nature-CNN, the input state s_t is passed through a sequence of three convolution layers with 32, 64, and 64 filters, kernel size of 8×8, 2×2 and 1×1, and stride of 4, 2, and 1, respectively. As shown in Figure 3(c), we see that our augmenting agent with DMTCN performs better than the baseline agent with Nature-CNN.

Overall performance: Figure 4 shows the improvements of the augmented agents with random curiosity module over baseline agents on 18 Ataris games: Asterix, Alien, Altantis, Amidar, BeamRider, Breakout, DemonAttack, Gopher, Mspaman, Pong, Qbert, Riverraid, RoadRunner, Seaquest, SpaceInvaders, Tennis, UpNDown, and Zaxxon. Our augmenting agent uses the DMTCN network for the policy gradient-based RL algorithm and combines parameter exploration and RCM exploration. In order to verify the validity of our approach, we perform the four baselines including A2C, A2C+ICM [15], A2C+RND [41], A2C+LIRPG [31]. Note that our augmenting agent achieves better performance compared with the others. Among them, the performance of our method is better than all other methods in 17 experiments. In Pong, our method performs similarly to LIRPG and outperforms other methods.

5. Discussion

Our method is at the intersection of multiple topics: random curiosity exploration, parameter exploration and deeper network. Therefore, our paper

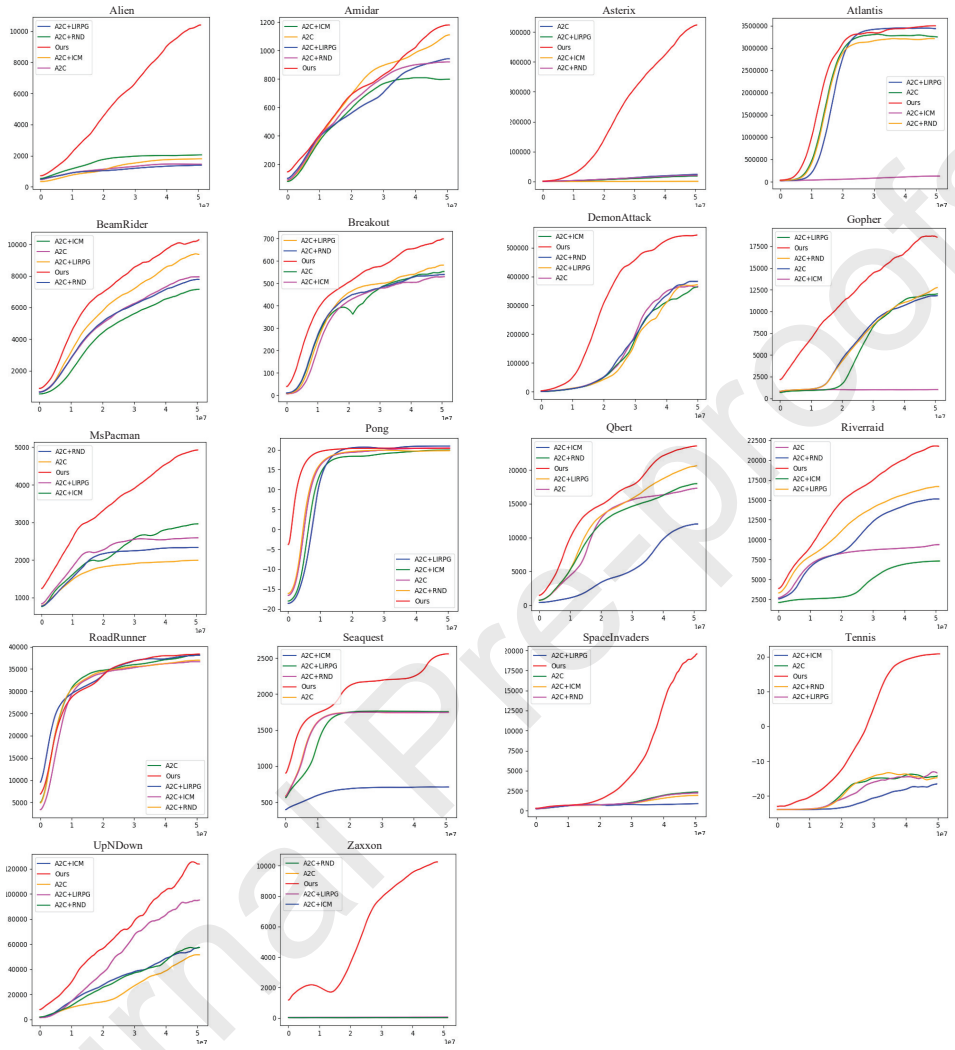


Figure 4: The x-axis is time steps during learning. The y-axis is the extrinsic rewards. We compare our method with other 4 algorithms on 18 Atari games. The red curves are for the proposed architecture in this paper.

has carried out sufficient experimental verification on these topics.

Recently, a few works demonstrated the success of curiosity exploration in reinforcement learning tasks. However, the previous methods have a catastrophic forgetting problem. With the training, the agent will make an error in the environmental novelty judgment, which will make the model perfor-

mance unstable. For example, Burda et al. [29] proposed that the collected trajectory samples can be used to pre-train the agent's curiosity dynamic model, but obtaining high-quality trajectory samples is a challenging problem. The prediction error based curiosity has some limitations, which may make the agent more inclined to transition to a meaningless random state, such as a "couch-potato" problem. Our random curiosity method is still a curiosity exploration based on prediction error. However, a target network is added to constrain the dynamic model, which can solve the catastrophic forgetting problem caused by dynamic model changes. The output of the forward dynamic model is used to predict value function during training. Compared with the single-action prediction in ICM, our value prediction performs better, which could prevent the couch-potato problem from occurring and reduce meaningless random exploration. It can be seen from the experimental results in Figure 3(a) that the agent using RCM has better performance than the ICM agent and A2C agent.

Both state-action based and parameter-based exploration are effective exploration methods in reinforcement learning. The exploration in the state or action space is to directly change the state or action of the agent to improve training. This makes the results of exploration easier to observe. The parameter exploration is to indirectly change agent behavior by perturbing parameters in the RL algorithm. Combining these two methods can more effectively promote agent exploration and improve performance. As shown in Figure 3(b), agents combing RCM and parameter exploration can learn skills faster and better. As shown in Figure 3(c), the agent using a deeper and more tightly connected network performs far better than the agent using Nature-CNN in RL tasks, which sufficiently proves the superior performance of our DMTCN. In the final comparative experiment, our method in this paper is a combination of RCM, parameter exploration, and DMTCN network. Compared with several other algorithms, our method has the best performance in multiple Atari games.

6. Conclusions

This paper proposes a novel exploration method that uses a random curiosity module to explore the environments in the state-action space and combines the parameter exploration method. A deeper and more tightly connected network is also used to extract features, further improving the agent's performance in multiple game tasks. Our experimental results show

that learning the reward function by combining the random curiosity method and the parametric exploration method can effectively improve the performance of policy gradient framework algorithms such as A2C. Using a deeper and more tightly connected network can more effectively represent the state of the agent. In summary, the proposed method in this paper achieves state-of-the-art performance in multiple Atari games. In the future, a lightweight network is more suitable for industrial applications. For tasks requiring only light exploration or even no exploration, it is a future research direction to adjust the exploration intensity automatically.

Funding

This work was supported by the National Key Research and Development Program of China under Grant 2019YFC1511401, and the National Natural Science Foundation of China under Grant 61103157.

References

[1] 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.

[2] J. Veness, K. S. Ng, M. Hutter, M. Bowling, Context tree switching, 2012 Data Compression Conference, IEEE, 2012, pp. 327–336.

[3] I. Adamski, R. Adamski, T. Grel, A. Jedrych, K. Kaczmarek, H. Michalewski, Distributed deep reinforcement learning: Learn how to play atari games in 21 minutes, International Conference on High Performance Computing, Springer, 2018, pp. 370–388.

[4] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning, International conference on machine learning, 2016, pp. 1928–1937.

[5] M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Silver, K. Kavukcuoglu, Reinforcement learning with unsupervised auxiliary tasks, arXiv preprint arXiv:1611.05397 (2016).

[6] A. R. Mahmood, D. Korenkevych, B. J. Komer, J. Bergstra, Setting up a reinforcement learning task with a real-world robot, 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2018, pp. 4635–4640.

[7] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, arXiv preprint arXiv:1509.02971 (2015).

[8] N. Heess, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. Eslami, M. Riedmiller, et al., Emergence of locomotion behaviours in rich environments, arXiv preprint arXiv:1707.02286 (2017).

[9] X. B. Peng, P. Abbeel, S. Levine, M. van de Panne, Deepmimic: Example-guided deep reinforcement learning of physics-based character skills, ACM Transactions on Graphics (TOG) 37 (2018) 143.

[10] A. Rajeswaran, K. Lowrey, E. V. Todorov, S. M. Kakade, Towards generalization and simplicity in continuous control, Advances in Neural Information Processing Systems, 2017, pp. 6550–6561.

[11] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, et al., Deep q-learning from demonstrations, Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[12] M. Večerík, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, M. Riedmiller, Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards, arXiv preprint arXiv:1707.08817 (2017).

[13] X. Zhang, H. Ma, Pretraining deep actor-critic reinforcement learning algorithms with expert demonstrations, arXiv preprint arXiv:1801.10459 (2018).

[14] Y. Aytar, T. Pfaff, D. Budden, T. Paine, Z. Wang, N. de Freitas, Playing hard exploration games by watching youtube, Advances in Neural Information Processing Systems, 2018, pp. 2930–2941.

[15] D. Pathak, P. Agrawal, A. A. Efros, T. Darrell, Curiosity-driven exploration by self-supervised prediction, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2017, pp. 16–17.

[16] N. Haber, D. Mrowca, S. Wang, L. F. Fei-Fei, D. L. Yamins, Learning to play with intrinsically-motivated, self-aware agents, Advances in Neural Information Processing Systems, 2018, pp. 8388–8399.

[17] I. M. de Abril, R. Kanai, Curiosity-driven reinforcement learning with homeostatic regulation, 2018 International Joint Conference on Neural Networks (IJCNN), IEEE, 2018, pp. 1–6.

[18] M. C. Machado, M. G. Bellemare, M. Bowling, Count-based exploration with the successor representation, arXiv preprint arXiv:1807.11622 (2018).

[19] G. Ostrovski, M. G. Bellemare, A. van den Oord, R. Munos, Count-based exploration with neural density models, Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 2721–2730.

[20] M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, R. Munos, Unifying count-based exploration and intrinsic motivation, Advances in Neural Information Processing Systems, 2016, pp. 1471–1479.

[21] H. Tang, R. Houthooft, D. Foote, A. Stooke, O. X. Chen, Y. Duan, J. Schulman, F. DeTurck, P. Abbeel, # exploration: A study of count-based exploration for deep reinforcement learning, Advances in neural information processing systems, 2017, pp. 2753–2762.

[22] J. Achiam, S. Sastry, Surprise-based intrinsic motivation for deep reinforcement learning, arXiv preprint arXiv:1703.01732 (2017).

[23] K. Gregor, D. J. Rezende, D. Wierstra, Variational intrinsic control, arXiv preprint arXiv:1611.07507 (2016).

[24] R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. De Turck, P. Abbeel, Vime: Variational information maximizing exploration, Advances in Neural Information Processing Systems, 2016, pp. 1109–1117.

[25] S. Mohamed, D. J. Rezende, Variational information maximisation for intrinsically motivated reinforcement learning, Advances in neural information processing systems, 2015, pp. 2125–2133.

[26] J. Schmidhuber, A possibility for implementing curiosity and boredom in model-building neural controllers, Proc. of the international conference on simulation of adaptive behavior: From animals to animats, 1991, pp. 222–227.

[27] N. Chentanez, A. G. Barto, S. P. Singh, Intrinsically motivated reinforcement learning, Advances in neural information processing systems, 2005, pp. 1281–1288.

[28] N. Haber, D. Mrowca, L. Fei-Fei, D. L. Yamins, Emergence of structured behaviors from curiosity-based intrinsic motivation, arXiv preprint arXiv:1802.07461 (2018).

[29] Y. Burda, H. Edwards, D. Pathak, A. Storkey, T. Darrell, A. A. Efros, Large-scale study of curiosity-driven learning, arXiv preprint arXiv:1808.04355 (2018).

[30] R. M. French, Catastrophic forgetting in connectionist networks, Trends in cognitive sciences 3 (1999) 128–135.

[31] Z. Zheng, J. Oh, S. Singh, On learning intrinsic rewards for policy gradient methods, Advances in Neural Information Processing Systems, 2018, pp. 4644–4654.

[32] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms (????).

[33] S. L. R. B. G. J. B. Yuhuai Wu, Elman Mansimov, Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation, CoRR (2017).

[34] A. Y. Ng, D. Harada, S. Russell, Policy invariance under reward transformations: Theory and application to reward shaping, ICML, volume 99, 1999, pp. 278–287.

[35] J. Sorg, R. L. Lewis, S. P. Singh, Reward design via online gradient ascent, Advances in Neural Information Processing Systems, 2010, pp. 2190–2198.

[36] T. Pohlen, B. Piot, T. Hester, M. G. Azar, D. Horgan, D. Budden, G. Barth-Maron, H. Van Hasselt, J. Quan, M. Večerík, et al., Observe and look further: Achieving consistent performance on atari, arXiv preprint arXiv:1805.11593 (2018).

[37] J. Ho, S. Ermon, Generative adversarial imitation learning, Advances in neural information processing systems, 2016, pp. 4565–4573.

[38] F. Torabi, G. Warnell, P. Stone, Behavioral cloning from observation, arXiv preprint arXiv:1805.01954 (2018).

[39] D. Pathak, P. Mahmoudieh, G. Luo, P. Agrawal, D. Chen, Y. Shentu, E. Shelhamer, J. Malik, A. A. Efros, T. Darrell, Zero-shot visual imitation, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2018, pp. 2050–2053.

[40] Schmidhuber, J., Formal theory of creativity, fun, and intrinsic motivation (1990–2010), IEEE Transactions on Autonomous Mental Development 2 (2010) 230–247.

[41] Y. Burda, H. Edwards, A. Storkey, O. Klimov, Exploration by random network distillation, arXiv preprint arXiv:1810.12894 (2018).

[42] I. Osband, J. Aslanides, A. Cassirer, Randomized prior functions for deep reinforcement learning, Advances in Neural Information Processing Systems, 2018, pp. 8617–8629.

[43] K. Jarrett, K. Kavukcuoglu, M. Ranzato, Y. LeCun, What is the best multi-stage architecture for object recognition?, 2009 IEEE 12th international conference on computer vision, IEEE, 2009, pp. 2146–2153.

[44] A. M. Saxe, P. W. Koh, Z. Chen, M. Bhand, B. Suresh, A. Y. Ng, On random weights and unsupervised feature learning., ICML, volume 2, 2011, p. 6.

[45] Z. Yang, M. Moczulski, M. Denil, N. de Freitas, A. Smola, L. Song, Z. Wang, Deep fried convnets, Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1476–1483.

[46] J. Oh, X. Guo, H. Lee, R. L. Lewis, S. Singh, Action-conditional video prediction using deep networks in atari games, Advances in neural information processing systems, 2015, pp. 2863–2871.

[47] L. Itti, P. Baldi, Bayesian surprise attracts human attention, Vision research 49 (2009) 1295–1306.

[48] Y. Sun, F. Gomez, J. Schmidhuber, Planning to be surprised: Optimal bayesian exploration in dynamic environments, International Conference on Artificial General Intelligence, Springer, 2011, pp. 41–51.

[49] B. C. Stadie, S. Levine, P. Abbeel, Incentivizing exploration in reinforcement learning with deep predictive models, arXiv preprint arXiv:1507.00814 (2015).

[50] S. Still, D. Precup, An information-theoretic approach to curiosity-driven reinforcement learning, Theory in Biosciences 131 (2012) 139–148.

[51] M. Lopes, T. Lang, M. Toussaint, P.-Y. Oudeyer, Exploration in model-based reinforcement learning by empirically estimating learning progress, Advances in neural information processing systems, 2012, pp. 206–214.

[52] J. Schmidhuber, Curious model-building control systems, Proc. international joint conference on neural networks, 1991, pp. 1458–1463.

[53] G. Brunner, M. Fritsche, O. Richter, R. Wattenhofer, Using state predictions for value regularization in curiosity driven deep reinforcement learning, 2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI), IEEE, 2018, pp. 25–29.

[54] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, et al., Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures, arXiv preprint arXiv:1802.01561 (2018).

[55] M. G. Bellemare, Y. Naddaf, J. Veness, M. Bowling, The arcade learning environment: An evaluation platform for general agents, Journal of Artificial Intelligence Research 47 (2013) 253–279.

Jing Li was born in 1982. She received M.S. degree of engineering from Shandong University of Technology in 2007, and Ph.D. degree of engineering from Beijing Institute of Technology in 2011. She is now an associate professor of School of Automation, Beijing Institute of Technology. Her research interests include image detection technology, object detection and tracking, etc.

Xinxin Shi was born in 1994. She received her B.S. degree from Beijing Institute of Technology, in 2017. She is pursuing a Master Degree as a member of Key Laboratory of Intelligent Control and Decision of Complex Systems, School of Automation, Beijing Institute of Technology, China. Her research interests include deep reinforcement learning and lane detection.

Jiehao Li received the M.Sc. degree in Control Engineering in South China University of Technology, Guangzhou, China, in 2017. He is pursuing a Ph.D. degree as a member of State Key Laboratory of Intelligent Control and Decision of Complex Systems, Beijing Institute of Technology, China, and as a Research Fellow of the Medical and Robotic Surgery Group (NEARLab) in Politecnico di Milano, Milano, Italy, working on the control theory and engineering application for mobile robots. He participated in the development of electric parallel wheel-legged robot. His interests mainly include robot motion control, autonomous tracking control, model predictive control, etc.

Xin Zhang was born in 1996. He received his B.S. degree from China University of Mining and Technology, Jiangsu, China, in 2018. He is pursuing a Master Degree as a member of Key Laboratory of Intelligent Control and Decision of Complex Systems, School of Automation, Beijing Institute of Technology, China. His research interests include semantic image segmentation, compressive sensing and model compressing.

Junzheng Wang received the Ph.D. degree in control science and engineering from the Beijing Institute of Technology, Beijing, China, in 1994. He is the Deputy Director with the State Key Laboratory of Intelligent Control and Decision of Complex Systems, the director of the Key Laboratory of Servo Motion System Drive and Control, and the Dean of the Graduate School of Beijing Institute of Technology, where he is a Professor and a Ph.D. Supervisor. His current research interests include motion control, static and dynamic performance testing of electric and electric hydraulic servo system, and dynamic target detection and tracking based on image technology. Prof. Wang is a senior member of the Chinese Mechanical Engineering Society and the Chinese Society for Measurement. He received the Second Award from the National Scientific and Technological Progress (No.1) in 2011.