Imitation Learning from Pixel-Level Demonstrations by HashReward

Xin-Qiang Cai, Yao-Xiang Ding, Yuan Jiang, Zhi-Hua Zhou
National Key Laboratory for Novel Software Technology
Nanjing University
{caixq,dingyx,jiangy,zhouzh}@lamda.nju.edu.cn

ABSTRACT

One of the key issues for imitation learning lies in making policy learned from limited samples to generalize well in the whole state-action space. This problem is much more severe in high-dimensional state environments, such as game playing with raw pixel inputs. Under this situation, even state-of-the-art adversary-based imitation learning algorithms fail. Through empirical studies, we find that the main cause lies in the failure of training a powerful discriminator to generate meaningful rewards in high-dimensional environments. Although it seems that dimensionality reduction can help, a straightforward application of off-the-shelf methods cannot achieve good performance. In this work, we show in theory that the balance between dimensionality reduction and discriminative training is essential for effective learning. To achieve this target, we propose HashReward, which utilizes the idea of supervised hashing to realize such an ideal balance. Experimental results show that HashReward could outperform state-of-the-art methods for a large gap under the challenging high-dimensional environments.

KEYWORDS

Imitation Learning; High-Dimensional Environments; Hashing

ACM Reference Format:
Xin-Qiang Cai, Yao-Xiang Ding, Yuan Jiang, Zhi-Hua Zhou. 2021. Imitation Learning from Pixel-Level Demonstrations by HashReward. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 9 pages.

1 INTRODUCTION

In recent years, reinforcement learning (RL) has achieved a great breakthrough in many domains including robot controlling and game playing [18, 20]. In spite of remarkable success, there are two main issues unsolved. First, in common situations, RL algorithms rely so much on the well-specified reward functions and exploration strategies, which require delicate designs in many complex problems. Second, the sample and computational complexities for practical RL algorithms are usually large, making it an unacceptable choice in solving many practical problems. On the other hand, imitation learning (IL), aiming at learning a good policy from demonstrations, enables the possibility of sample efficient policy learning without the need of designing rewarding and exploration strategies by hand.

Nevertheless, since collecting expert demonstrations is costly, the number of expert trajectories for IL is usually limited in practice. This may significantly increase the risks of over-fitting. One of the key ideas to improve generalization performance is to learn a proper reward function from expert demonstrations. The learner can get high rewards only when it generates behaviors similar to the demonstrations. Guided by such rewards, the learner is encouraged to mine out the expert's policy by RL algorithms, instead of directly performing behavior cloning [1, 13]. This idea motivates the emergence of adversary-based IL. Under this family of approaches, the policy and the adversarial discriminator are jointly trained during learning, in order to force the learner to minimize the discrepancy between the generated distribution and the demonstrated data. One representative approaches among them, generative adversarial imitation learning (GAIL) [13], achieves state-of-the-art performance in many tasks with relatively low-dimensional state spaces. However, it has been verified that the performance of such approaches, even deep neural network-based GAIL, degenerates seriously in tasks with high-dimensional state spaces [26, 29]. On the other hand, it is well known that deep RL algorithms can achieve even superhuman performance under these domains [18].

In this work, we propose thorough studies towards understanding why adversary-based algorithms fail in high-dimensional IL environments. For obtaining a good policy, the discriminator plays an important role. The cause for performance degeneration in high-dimensional space can be interpreted by the hardness of properly dealing with the discrimination-rewarding trade-off in the discriminator learning process. Because the number of demonstration samples is limited, under high-dimensional state space, the learned discriminator may bias towards the discrimination side, providing meaningless rewarding signals for policy learning. Experimental observations and theoretical results are provided to support the above interpretations: They disclose that an ideal balance between performing dimensionality reduction (DR) and discriminative training is essential for learning.

Based on the above findings, we study a practical approach to enhance discriminator learning. It is non-trivial to guarantee that discriminative information in high-dimensional space can be preserved after performing DR. This suggests that directly utilizing off-the-shelf unsupervised DR algorithms, such as unsupervised autoencoder and hashing, is not the right choice because they make the processes of DR and discriminator learning totally isolated. As a result, essential information for discrimination can be heavily lost due to DR, making discriminator training biased towards discrimination or rewarding only. To address this issue, we propose a novel IL algorithm named HashReward, which utilizes a supervised hashing strategy to incorporate DR and discriminator training into

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

a unified procedure. Experiments show that HashReward achieves significant improvement comparing to other state-of-the-art IL approaches under high-dimensional environments, which are broadly recognized to be challenging for IL algorithms.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 provides preliminaries. Section 4 demonstrates how to solve the underlying discrimination-rewarding trade-off problem for adversary-based IL methods, and then introduces HashReward algorithm. Section 5 reports the experimental setup and results. Finally, Section 6 concludes the paper.

2 RELATED WORK

Recently, adversary-based methods have achieved great success in a wide range of IL scenarios, including robotic control [13], game playing [12], and simulating environments [22]. As the outputs from the discriminator are utilized as reward signals for the learner, they can be treated as a generalization of the family of IRL approaches. Among them, GAIL [13] achieves state-of-the-art performance in handling low-dimensional IL problems, while its performance degenerates significantly in high-dimensional environments as discussed in Section 1. Though effective high-dimensional IL is challenging, there are a few works related to this topic, which assume additional signals besides provided demonstrations. [12] and [3] augmented RL to learn from both environment rewards and expert demonstrations, meanwhile [9], [7] and [14] utilized human preference to enhance IL. In comparison, we consider pure IL without using additional environmental signals, which is more challenging. There are some related works proposed for this scenario, e.g., CNN-AIRL [26], which uses an adversarial IRL method to play Atari game *Enduro* with pixels inputs. But it utilizes autoencoder as the DR method, whose unsupervised information is not enough illustrated in the experiments. D-REX [8] improves [7] by learning a behavior cloning model to generate the ranked samples, and has achieved promising performance under the IL problems with suboptimal demonstrations. But in order to train the behavior cloning model efficiently, D-REX requires samples from quite different experts, which are not available in the settings of most IL tasks. GIRIL [29] is a non-GAIL method, and deals with the high-dimensional IL problem from a different angle than DR, which encodes action signals into VAE and utilizes the mechanism of curiosity to produce reward signals. But as will be discussed in subsequent sections, appropriate dimensionality reduction is inevitable for solving high-dimensional IL problems. VAIL [19] is the most closely related work to ours. It improves GAIL by employing information-theoretic regularization to learn a better feature representation as the input to the discriminator, and successfully used image feature in a continuous domain. Nevertheless, VAIL does not utilize explicit supervised loss in DR, which is crucial as shown in our experiments. In summary, the high-dimensional IL problem remains challenging for existing IL approaches.

Several recent works [5, 23, 24, 28] have shown that the latent hashing features with unsupervised information learned by autoencoder can help address the exploration issue in challenging RL problems. Through hashing, a high-dimensional state is effectively discretized to make similar states mapping into the same hashing code [15], leading to the convenience for utilizing counting-based exploration. Though unsupervised hashing seems to be promising for improving IL, we observe that the performance is not that satisfactory in the experiments. This phenomenon is reasonable since direct unsupervised hashing may lead to the risk of losing discriminative information in the original state space. This motivates us to propose HashReward to address this issue.

3 PRELIMINARIES

In policy learning problems, a Markov Decision Process (MDP) can be represented by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r, T \rangle$, in which $\mathcal{S}$ denotes the set of states, $\mathcal{A}$ denotes the set of actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ denotes the transition probability distributions of the state and action pairs, $\gamma \in (0, 1]$ denotes the discount factor, $r : \mathcal{S} \to \mathbb{R}$ denotes the reward function, $S_0 : \mathcal{S} \to \mathbb{R}$ denotes the initial state distribution and T denotes the horizon. The objective of RL is to learn a policy π to maximize the expected total rewards $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$ obtained by π.

Different from RL, in IL, the learner has no access to r. Instead, there are m expert demonstrations $\{\tau_{E,1}, \tau_{E,2}, \ldots, \tau_{E,m}\}$ available, where $\tau_{E,i}, i \in [m]$ is an expert trajectory (a series of state-action pairs) drawn independently from *expert's trajectory distribution* μ_{π_E}, induced by the expert's policy π_E, initial state distribution S_0 and the transition probability distribution $\mathcal{P}$. The goal of the learner is to generate π_G such that the induced *learner's trajectory distribution* μ_{π_G} matches μ_{π_E}.

Instead of directly minimizing the discrepancy between trajectory distributions to solve IL problem, existing adversary-based methods turn to the equivalent goal of minimizing the distance between the learner and expert occupancy measures $d(\rho_{\pi_\mathrm{G}}, \rho_{\pi_\mathrm{E}})$, where $\rho_\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is defined as $\rho_\pi(s, a) = \pi(a|s) \sum_{t=0}^{\infty} \gamma^t Pr(s_t = s|\pi)$. Based on the idea of generative adversarial training, they perform policy learning by solving a min-max optimization problem, i.e. $\min_{\pi_\mathrm{G}} \max_D D(\rho_{\pi_\mathrm{E}}, \rho_{\pi_\mathrm{G}})$, in which D is the discriminator. Among these approaches, GAIL [13] achieves state-of-the-art performance in many task environments. The objective of GAIL is

$$\min_{\pi_\mathrm{G}} \max_D \mathbb{E}_{\rho \sim \rho_{\pi_\mathrm{E}}}[\log D(\rho)] + \mathbb{E}_{\rho \sim \rho_{\pi_\mathrm{G}}}[\log(1 - D(\rho))], \quad (1)$$

where the discriminator $D : \mathcal{S} \times \mathcal{A} \to [0, 1]$ has the formulation of a classifier trying to discriminate state-action pairs generated by the learner and the expert. It is proved that by GAIL, the learned π_G can minimize the regularized version of Jensen-Shannon divergence, i.e.,

$$\pi_\mathrm{G} = \arg\min_{\pi \in \Pi} -\mathbb{H}(\pi) + d_{JS}(\rho_\pi, \rho_{\pi_\mathrm{E}}), \quad (2)$$

where Π denotes the policy set, $d_{JS}(\rho_\pi, \rho_{\pi_\mathrm{E}})$ is the Jensen-Shannon divergence between ρ_π and ρ_{π_E}, and $\mathbb{H}(\pi)$ is the causal entropy used as a policy regularizer. GAIL solves Equation (1) by alternatively taking a gradient ascent step to train the discriminator D and a minimization step to learn policy π_G based on off-the-shelf RL algorithm which utilizes $-\log D(s, a)$ as the pseudo reward function.

4 HIGH-DIMENSIONAL IMITATION LEARNING BY HASHREWARD

As learning an effective discriminator plays an essential rule in adversary-based methods, in this section, first we analyze the reason why existing state-of-the-art adversary-based methods fail by

proposing a generalization bound about learning a discriminator in IL scenarios. Inspired by the theoretical conclusion, then we propose HashReward to solve the high-dimensional IL problem.

4.1 Balancing the Discrimination-Rewarding Trade-Off

Intuitively, a well-trained discriminator should balance the following two capabilities. On the one hand, the discriminator should be powerful enough, in order to generate negative rewards on trajectories that are significantly different from the expert's demonstrations. On the other hand, the discriminator should not be too strong to discriminate trajectories that are sufficiently similar to the expert's demonstrations. This will ensure that positive rewards are generated on these good trajectories, and encourage the policy to improve in the right direction. We identify this problem as the *discrimination-rewarding trade-off*. The reward curves on Atari game *Qbert* in Figure 5 evince that the deficiency of state-of-the-art adversary-based methods (i.e., GAIL) in high-dimensional environments is due to the tendency of learning too powerful discriminators. Intuitively, this is due to the curse of dimensionality: We usually cannot collect sufficient expert's demonstration data to meet the demand of high-dimensional learning, leading to overfitting of discriminator training. In such a case, dimensionality reduction (DR) alone is not hard to think of, but the key to finding out a solution is how to do the DR. Thus we start from a theoretical analysis motivated by the generalization theory of GAN [30] over the trajectory learning. Let $\hat{\mu}_{\pi_{\mathrm{E},m}}$ be expert's empirical trajectory distribution obtained from m expert trajectories $\tau_{E,i}, i \in [m]$, over the trajectory space $\mathcal{T}$. Generally, we assume that a feature transformation $\phi(\tau_{\mathrm{E}})$, which is a bijective mapping from $\mathcal{T}$ to another trajectory space $\mathcal{T}'$ exists. We can see that ϕ plays a crucial role in the following discussions. For simplicity, we assume that the learner directly minimizes the neural distance [2] over trajectory distributions *under the mapped feature space*, i.e,

$$\min_{\pi_{\mathrm{G}} \in \mathcal{G}} [d_{\mathcal{D}'}(\hat{\mu}_{\pi_{\mathrm{E},m}}, \mu_{\pi_{\mathrm{G}}})], \tag{3}$$

where

$$\begin{aligned} d_{\mathcal{D}'}(\hat{\mu}_{\pi_{\mathrm{E},m}}, \mu_{\pi_{\mathrm{G}}}) = \sup_{D \in \mathcal{D}'} \{ & \mathbb{E}_{\tau_{\mathrm{E}} \sim \hat{\mu}_{\pi_{\mathrm{E},m}}} [D(\phi(\tau_{\mathrm{E}}))] \\ & - \mathbb{E}_{\tau_{\mathrm{G}} \sim \mu_{\pi_{\mathrm{G}}}} [D(\phi(\tau_{\mathrm{G}}))] \}, \end{aligned} \tag{4}$$

in which $\mathcal{G}$ is the policy hypothesis space, and $\mathcal{D}'$ is the neural network based discriminator hypothesis space under trajectory space $\mathcal{T}'$. To proceed the analysis, we utilize $\mathcal{D}$ to denote the hypothesis space under the original trajectory space $\mathcal{T}$, which is different from $\mathcal{D}'$ only in input dimension. By solving Equation (3), we expect to obtain a π_{G} that minimizes the expected neural distance in the original trajectory space, i.e., $d_{\mathcal{D}}(\mu_{\pi_{\mathrm{E}}}, \mu_{\pi_{\mathrm{G}}}) = \sup_{D \in \mathcal{D}} \{\mathbb{E}_{\tau_{\mathrm{E}} \sim \mu_{\pi_{\mathrm{E}}}} [D(\phi(\tau_{\mathrm{E}}))] - \mathbb{E}_{\tau_{\mathrm{G}} \sim \mu_{\pi_{\mathrm{G}}}} [D(\phi(\tau_{\mathrm{G}}))]\}$ by minimizing $d_{\mathcal{D}'}(\hat{\mu}_{\pi_{\mathrm{E},m}}, \mu_{\pi_{\mathrm{G}}})$, which would guarantee a good π_{G} when $\mathcal{D}$ is rich enough. In practice, the optimization in Equation (3) may not be exactly solved, thus we utilize $\hat{d}_{\mathcal{D}'}, \hat{\pi}_{\mathrm{G}}$ to denote the resulted neural distance and policy after training. We further introduce the following assumptions.

Assumption 1. *$\mathcal{D}'$ is a class of neural networks, whose definition could be found in the supplementary material. Furthermore, $\mathcal{D}'$ is even, i.e., $D \in \mathcal{D}'$ implies $-D \in \mathcal{D}'$. Meanwhile $\forall D \in \mathcal{D}'$, $\|D\|_\infty \leq \Delta$, in which $\|D\|_\infty = \sup_{\tau \in \mathcal{T}'} |D(\tau)|$.*

Assumption 2. *$\hat{d}_{\mathcal{D}'}(\hat{\mu}_{\pi_{\mathrm{E},m}}, \mu_{\hat{\pi}_{\mathrm{G}}}) \leq \eta$.*

Assumption 1 is easily satisfied by general neural network models. Furthermore, if $\hat{\pi}_{\mathrm{G}}$ is sufficiently trained w.r.t. $\hat{d}_{\mathcal{D}'}$, then η in Assumption 2 is also small. We then have the following sample complexity result, whose proof is included in the supplementary material.

Theorem 1. *Let $\Delta_1 = |d_{\mathcal{D}}(\mu_{\pi_{\mathrm{E}}}, \mu_{\hat{\pi}_{\mathrm{G}}}) - d_{\mathcal{D}'}(\mu_{\pi_{\mathrm{E}}}, \mu_{\hat{\pi}_{\mathrm{G}}})|$, $\Delta_2 = |\hat{d}_{\mathcal{D}'}(\hat{\mu}_{\pi_{\mathrm{E},m}}, \mu_{\hat{\pi}_{\mathrm{G}}}) - d_{\mathcal{D}'}(\hat{\mu}_{\pi_{\mathrm{E},m}}, \mu_{\hat{\pi}_{\mathrm{G}}})|$. Given expert trajectory data X which consists of m trajectories $\tau_{\pi_{\mathrm{E}}} \in \mathcal{T}$, if $m \geq 3\|\phi(\mathbf{X})\|_{\mathrm{F}} \mathcal{R}$, then with probability at least $1-\delta$, we have*

$$\begin{aligned} d_{\mathcal{D}}(\mu_{\pi_{\mathrm{E}}}, \mu_{\hat{\pi}_{\mathrm{G}}}) \leq \Delta_1 + \Delta_2 + 6\Delta\sqrt{\frac{\log(2/\delta)}{2m}} \\ + \frac{24\,\|\phi(\mathbf{X})\|_{\mathrm{F}}\,\mathcal{R}}{m}\left(1 + \log\frac{m}{3\,\|\phi(\mathbf{X})\|_{\mathrm{F}}\,\mathcal{R}}\right) + \eta, \end{aligned} \tag{5}$$

where $\mathcal{R}$ denotes the spectral normalized complexity of $\mathcal{D}'$ as defined in the supplementary material.

The above result reveals the key factors to learn a good policy. First, see the sample complexity terms involving m. Under a properly chosen $\mathcal{D}'$ which ensures small $\mathcal{R}$, the key for sharpening the bound is to control the Frobenius norm of $\phi(\mathbf{X})$, which can be achieved by learning a good DR version of ϕ. Furthermore, Δ_1 measures the gap of optimal discriminator between $\mathcal{D}$ and $\mathcal{D}'$. This shows that ϕ should also be distance-preserving. These two observations show the advantage of learning a hashing function as ϕ: It is well-known that hashing mappings usually perform DR with sparsity as well as distance-preserving property [11], thus fit for our needs desirably. We should also pay attention to Δ_2, which measures the quality of discriminator training in $\mathcal{D}'$. We find this should be stressed not only for learning the discriminator under $\mathcal{D}'$ mapped *after* ϕ, but also for *learning* ϕ *itself*. These observations from Theorem 1 motivate our HashReward approach.

4.2 HashReward

According to Theorem 1, learning ϕ in a proper way is essential to our task: A good ϕ should reduce the input norm ($\|\phi(\mathbf{X})\|_{\mathrm{F}}$) meanwhile preserving discrimination properties (Δ_1 and Δ_2). This particularly shows that directly applying off-the-shelf unsupervised DR, like autoencoder or unsupervised hashing, could lead to unsatisfying results for risks of losing discriminative information.

To achieve a good balance, we propose HashReward, which is a novel adversary-based IL approach utilizing supervised hashing for learning effective discriminators to achieve the balance between discrimination and rewarding. The key idea lies in using unsupervised reconstructive information to learn the hashing code (reducing $\|\phi(\mathbf{X})\|_{\mathrm{F}}$ and Δ_1) as well as leading supervised discriminative information into the whole DR part (reducing Δ_2), so that the hashing code could obtain the ability to represent the original high-dimensional states. To generate such effective representation, the network structure utilized for the autoencoder and discriminator training is illustrated in Figure 1. We utilize autoencoder to train the hashing code that maintains reconstructive information

Algorithm 1 HashReward

Input: Expert demonstrations $\tau_{\mathrm{E}} \sim \mu_{\pi_{\mathrm{E}}}$; Initialized learner's policy $\pi_{\mathrm{G},0}$.

1: Pretrain autoencoder with samples from expert demonstrations and the random policy.
2: **for** iteration $t = 1, 2, \cdots, T$ **do**
3: Utilize $\pi_{\mathrm{G},t-1}$ to generate learner's trajectories, i.e. $\tau_{\mathrm{G}} \sim \mu_{\pi_{\mathrm{G}}}$.
4: Sample a mini-batch of state-action pairs $\{(s,a)\}_t$ from both τ_{G} and τ_{E}.
5: Update HashReward network by Equation (6) using $\{(s,a)\}_t$, then generate rewards $\hat{r}$ for all state-action pairs in $\{(s,a)\}_t$.
6: $\pi_{\mathrm{G},t-1} \rightarrow \pi_{\mathrm{G},t}$ using $\hat{r}$ by RL update.
7: **end for**

of the original pixels. Meanwhile, the action signal is concatenated to the hashing code to formulate the input of the discriminator training. By this way, the supervised discriminative information is directly propagated back to learn hashing codes. The loss function for discriminator training is divided into two parts, i.e,

$$L = L_H + L_D, \tag{6}$$

where L_H denotes the hashing training loss, which propagates error for training the autoencoder and hashing code layer, and L_D denotes the discriminator training loss, which is utilized for training the discriminator layers as well as enhancing the supervision of DR part. L_D is similar to the inner maximization in Equation (1), except that the input state s is replaced by the binary hashing code $b(s)$. Inspired by Liu et al. [17], we define the hashing loss L_H as

$$\begin{aligned} L_H(\{s_i, y_i\}, \{s_j, y_j\}) = & \left\|s_i - s_i'\right\|_2^2 + \left\|s_j - s_j'\right\|_2^2 \\ & + \lambda\left(\|1 - |b(s_i)|\,\|_2^2 + \|1 - \left|b(s_j)\right|\,\|_2^2\right) \\ & + \frac{1}{2}\mathbb{I}(y_{ij})\left\|b(s_i) - b(s_j)\right\|_2^2 \\ & + \frac{1}{2}(1 - \mathbb{I}(y_{ij}))\max(2l - \left\|b(s_i) - b(s_j)\right\|_2^2, 0), \end{aligned} \tag{7}$$

in which l denotes the length of the hashing code and $\mathbb{I}(y_{ij})$ is the indicator function which takes 1 if y_i equals to y_j, and 0 otherwise. In Equation (7), (s_i, y_i), (s_j, y_j) denote a pair of state-label instance. For one state-label instance (s, y), we utilize s to denote a state sampled from the learner's policy or a state from the trajectory generated by the expert. Furthermore, we utilize y to indicate where s is sampled, such that $y = 1$ if s is sampled from the demonstration and $y = 0$ otherwise. The first two terms of L_H represent the reconstruction error, making the reconstructed states s' similar to the original states s. The next two terms (regularization terms weighted by λ) are used to enforce $b(s)$ to get close to binary values in $\{-1, 1\}$, where $b(s)$ is the logit output of the hashing layer. The last two terms in L_H are essential for introducing the supervision into hashing code training. From these terms, the unbinarized hashing codes $b(s_i)$ and $b(s_j)$ of two states s_i, s_j will get similar only when they have the same labels. By this way, the discriminative information is effectively propagated for the learning hashing representations.

Overall, the output of HashReward network is $D(s,a) \in (0,1)$, and we utilize $-logD(s,a)$ as the pseudo reward for the agent. This process in all experiments can be compared to the network architecture in the supplementary material.

It can be seen that by utilizing Equation (7), the separability of the hashing codes, which are the inputs to the discriminative layers, can be reliably preserved even when the dimension of hashing layer is much smaller than the original input dimension. Instead of fixing the parameters of the hashing code layer in the training process, the HashReward model and the policy are updated alternatively during learning. By this way, the training of the hashing code layer and the discriminator module are coupled together. In order to achieve faster convergence, a pretraining stage with samples from expert demonstrations and the random policy for the autoencoder module is included in learning. The learning procedure of HashReward is illustrated in Algorithm 1.

5 EXPERIMENT

5.1 Experimental Setup

Environment. We choose 15 games in Arcade Learning Environment [4] and 5 simulators in MuJoCo [25]. The experiment is implemented in OpenAI Gym platform [6], which contains Atari 2600 video games with high-dimensional observation space (raw pixels), and pixels of each state in MuJoCo are obtained by a camera. The input states for the learner are set as low-dim continuous control inputs in MuJoCo. We train converged DQN-based agents as experts in Atari, and DDPG-based [16] agents in MuJoCo. 20 expert trajectories are collected for each game, also 3 trials with different random seeds are conducted for each environment. All experiments are conducted on server clusters with NVIDIA Tesla K80 GPUs.

Contenders. There are five basic contenders in the experiment, i.e., GAIL [13], VAIL [19], GIRIL [29], GAIL with autoencoder (GAIL-AE) which utilizes only the first two autoencoder loss terms in Equation (7), and GAIL with unsupervised hashing (GAIL-UH) which utilizes only the first four unsupervised hashing loss terms in Equation (7). The codes of GAIL-AE and VAIL are real numbers, while those of GAIL-UH and HashReward belong to $\{-1, 1\}$. We initialize autoencoder pretraining for 40M frames of updates for GAIL-AE, GAIL-UH, GIRIL, and HashReward in Atari (1M in MuJoCo). To find out whether keeping autoencoder stable during training will increase the performance of GAIL-AE and GAIL-UH, we conduct experiments of GAIL-AE and GAIL-UH with updating autoencoder during training as GAIL-AE-Up and GAIL-UH-Up. Besides, to show the necessity of hashing for HashReward, we remove the third and fourth terms in Equation (7) as HashReward-AE. The basic RL algorithm is PPO [21], and the reward signals of all methods are scaled into $[0, 1]$ to enhance the performance of RL part. We set all hyper-parameters and network architectures of the policy part the same to [10]. Also, the hyper-parameters of DR and discriminator for all methods are the same: The DR and discriminator updates using Adam with a decayed learning rate of 3e-4; the batch size is 256; λ is 0.01. We implement VAIL and GIRIL with the recommended hyper-parameters in their paper. The ratio of update frequency between the learner and discriminator is 3: 1.

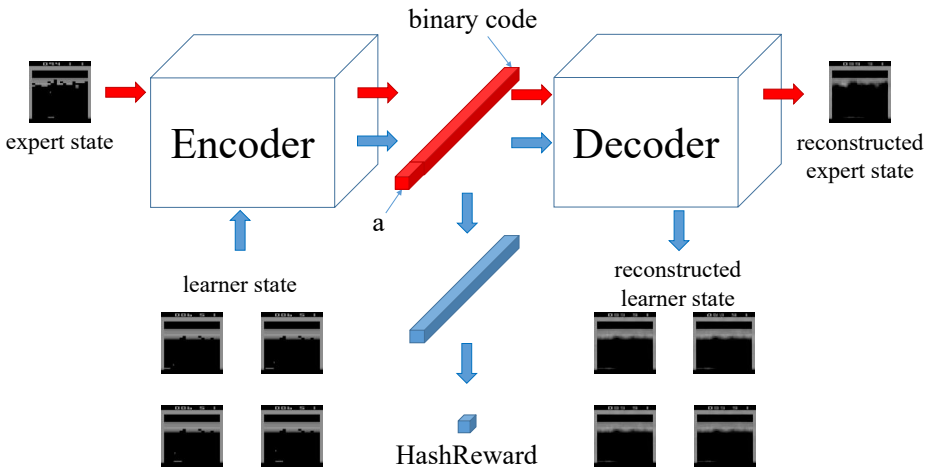


Figure 1: Illustration of the HashReward model architecture which contains two modules: the autoencoder module and the discriminator module. The red solid block represents the concatenation of the hashing code layer and the action signal a. The blue solid block represents hidden dense layers of the discriminator module.

Table 1: The performance of each method on Atari after 10M timesteps. Boldface numbers indicate the best results. The state space is $84 \times 84 \times 4$.

	Expert Reward	GAIL	VAIL	GIRIL	GAIL-AE	GAIL-AE-Up	GAIL-UH	GAIL-UH-Up	HashReward-AE	HashReward
BeamRider	2139.20 ± 41.60	854.47±220.63	615.63±258.67	**2973.27±224.00**	638.04±95.95	2182.59±1424.69	1412.91±230.91	2089.55±1232.42	596.00±8.14	1613.68±203.83
Breakout	144.35 ± 29.27	10.48±1.70	24.32±2.60	61.53±17.98	17.32±6.83	28.70±0.22	1.04±0.52	49.85±9.78	32.10±2.84	**67.73±13.77**
Boxing	95.70 ± 2.63	26.78±3.24	2.47±1.55	-3.64±1.57	26.05±19.83	-5.47±23.43	0.59±1.17	-15.36±15.18	-0.72±1.15	**84.71±2.13**
BattleZone	23000.00 ± 2549.51	11863.33±767.09	7566.67±1503.43	9070.00±2203.47	11030.00±4734.64	9133.33±5255.45	4670.00±1432.29	11400.00±3559.52	7043.33±1728.01	**15623.33±278.61**
ChopperCommand	3135.00 ± 145.86	1469.33±135.22	1190.00±37.50	604.67±52.93	1151.67±17.46	1148.00±74.69	1144.33±446.19	995.33±362.83	924.00±107.34	**1522.67±78.36**
CrazyClimber	95245.00 ± 2477.39	35451.33±1002.73	41170.67±5024.62	5020.00±599.03	10321.00±1539.70	9590.00±5689.36	4049.00±907.23	3159.33±2228.60	44766.67±21591.68	**63076.00±1841.86**
Enduro	469.85 ± 18.21	26.53±26.35	119.87±10.06	0.00±0.00	70.02±70.66	0.04±0.05	0.00±0.00	0.00±0.00	209.07±19.05	**219.88±72.06**
Kangaroo	4175.00 ± 94.21	1695.67±87.96	1482.00±427.65	32.00±2.83	935.00±63.90	542.00±44.50	80.00±94.36	26.33±5.31	1352.00±249.99	**1925.67±145.11**
MsPacman	3163.00 ± 160.88	298.00±22.56	1316.87±182.96	121.23±75.75	711.43±9.19	674.67±3.07	655.00±41.36	701.33±16.21	726.93±42.11	**1463.07±52.68**
Pong	21.00 ± 0.00	-18.40±0.29	-18.78±0.18	-20.14±0.07	-14.57±5.65	-20.90±0.15	-17.89±0.79	-17.59±0.26	-17.97±0.94	**-5.25±4.79**
Qbert	4750.00 ± 50.00	3634.42±388.09	3260.62±209.12	1050.33±197.75	1126.25±416.21	1303.62±33.38	360.33±111.00	287.17±67.09	4000.50±327.93	**4553.58±155.14**
Seaquest	1835.00 ± 23.56	761.13±38.89	826.93±40.99	628.93±52.75	1124.00±400.68	1151.33±628.84	689.93±5.04	664.80±10.47	1339.80±188.92	**1400.73±67.41**
SpaceInvaders	743.50 ± 26.03	341.67±23.38	346.70±27.42	**550.10±12.95**	302.23±38.18	318.75±36.39	309.83±74.85	512.32±49.89	496.33±67.44	546.48±42.09
UpNDown	34200.50 ± 2083.35	21086.17±848.61	31637.43±2301.67	24399.07±9663.36	20252.53±10582.64	43972.33±1105.54	4227.27±1264.39	8715.40±4807.56	**64648.43±17684.03**	36989.70±12739.30
Zaxxon	11190.00 ± 490.82	8130.33±626.00	8473.00±1507.70	2644.33±63.73	8577.67±852.58	9062.00±1336.34	1614.67±64.63	1546.67±28.77	9005.00±1404.70	**10068.33±410.69**

Table 2: The performance of each method on MuJoCo after 1M timesteps. The state space is $84 \times 84 \times 1$.

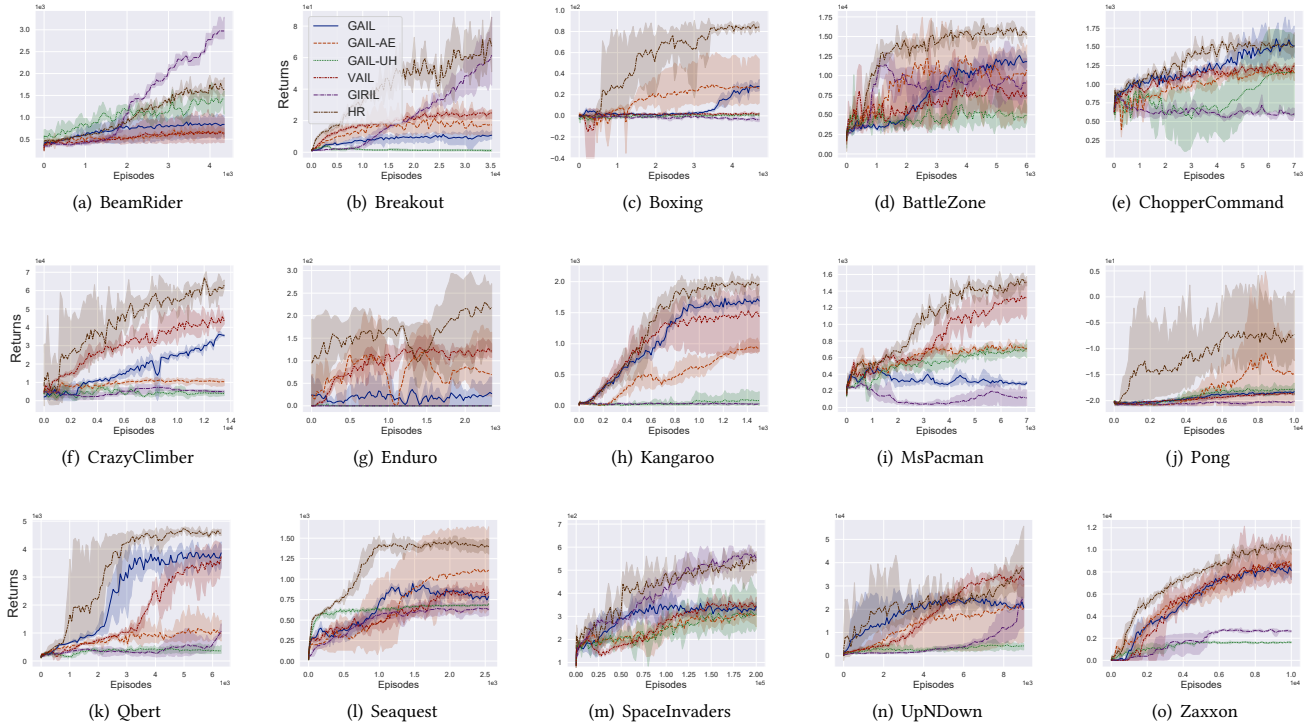
	Expert Reward	GAIL	VAIL	GIRIL	GAIL-AE	GAIL-AE-Up	GAIL-UH	GAIL-UH-Up	HashReward-AE	HashReward
Humanoid	1029.15 ± 53.95	360.13±9.53	392.69±9.87	207.27±20.07	**435.95±7.52**	422.90±28.37	424.11±22.36	433.74±9.76	364.61±1.98	370.92±3.65
HalfCheetah	1189.01 ± 140.28	-1198.28±344.95	-138.48±36.00	-280.27±159.74	-488.56±232.77	-706.58±339.33	-402.19±49.28	-420.65±54.38	-149.73±253.05	**-78.92±168.05**
Hopper	2304.87 ± 287.45	496.31±294.86	2210.22±5.88	975.13±17.02	753.02±60.52	1149.54±309.80	968.16±16.20	943.70±149.26	2116.94±35.24	**2216.83±76.57**
HumanoidStandup	110722.99 ± 6360.51	67946.69±3690.98	77412.49±2948.07	81217.79±1013.14	79006.35±2857.10	75210.63±2109.55	68248.25±3586.59	70330.63±2901.27	82718.25±2706.36	**83423.17±1878.71**
Reacher	-10.00 ± 2.83	-28.99±5.42	-74.98±13.82	**-13.53±1.31**	-18.79±1.01	-20.23±0.80	-15.54±0.36	-15.55±0.33	-100.10±25.82	-39.35±12.24

5.2 Results

Experimental results in Atari are reported in Table 1 and Figure 2, and that of the major methods in MuJoCo are shown in Table 1 and Figure 3. More results including the performances of other contenders are reported in the supplementary material.

We can observe that VAIL outperforms GAIL on most environments, and achieves near-expert performance on *UpNDown* and *Hopper*, but still fails compared with HashReward. GIRIL achieves the best performance in *BeamRider*, *SpaceInvaders* and *Reacher*, but its behavior is also unsatisfactory in some environments. For variants of GAIL, GAIL-AE outperform GAIL on several environments, but fail on most compared with VAIL and HashReward. Meanwhile, GAIL-UH only achieve satisfactory performance on *BeamRider*, *Humanoid* and *Reacher*. It also evinces that without supervision, the change of feature could confuse the discriminator training. HashReward-AE outperforms GAIL on most games, while remains a gap compared with HashReward. This demonstrates that hashing is necessary for HashReward even with supervision. As expected, HashReward outperform its contenders significantly and achieve the best performance on most of environments (12/15 in Atari, 3/5 in MuJoCo), meanwhile gains expert-level reward on *Boxing*, *Qbert*, *SpaceInvaders*, *UpNDown*, *Zaxxon* and *Hopper*.

It can be concluded that GAIL could not be improved with simple unsupervised DR; also the usage of supervision only from discriminator (regular term of loss in VAIL) to enhance DR part is not enough; besides, another potential reason for not that satisfactory results of VAIL on Atari is that its discriminator does not

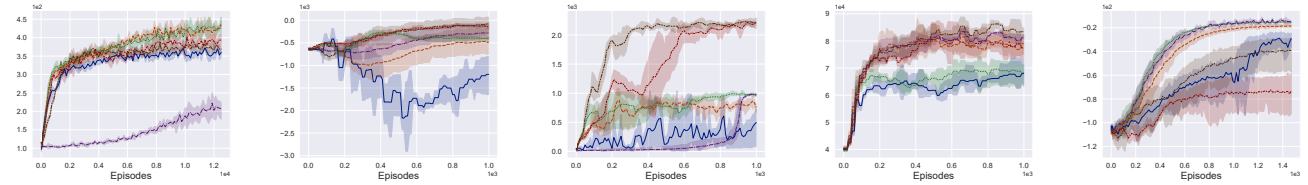


(a) BeamRider (b) Breakout (c) Boxing (d) BattleZone (e) ChopperCommand

(f) CrazyClimber (g) Enduro (h) Kangaroo (i) MsPacman (j) Pong

(k) Qbert (l) Seaquest (m) SpaceInvaders (n) UpNDown (o) Zaxxon

Figure 2: Learning curves of on Atari after 10M timesteps, where shaded regions indicate the standard deviation. The state space is $84 \times 84 \times 4$.



(a) Humanoid (b) HalfCheetah (c) Hopper (d) HumanoidStandup (e) Reacher

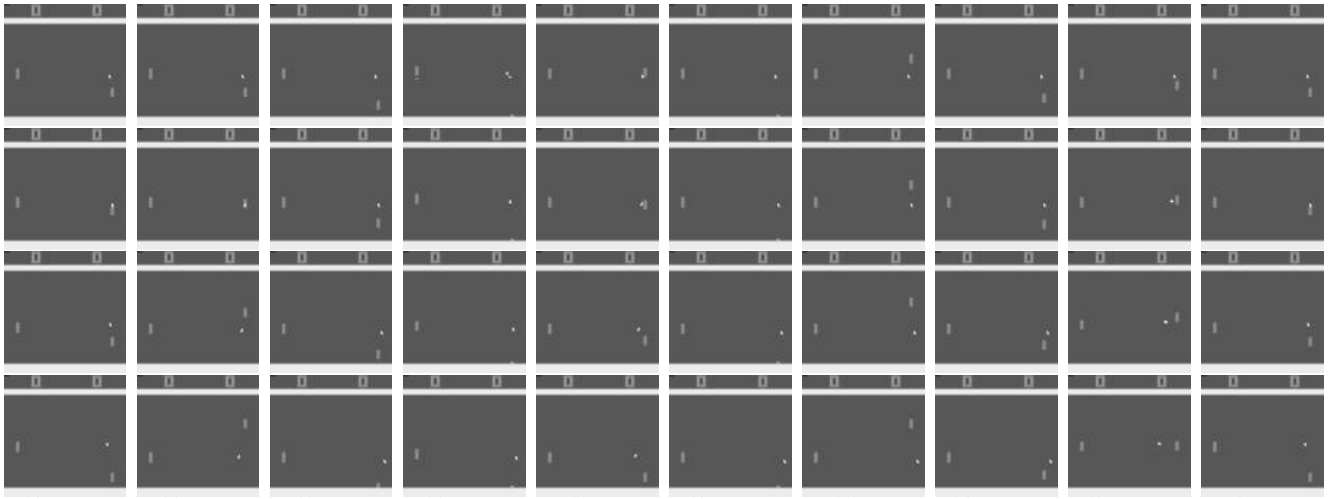
Figure 3: Learning curves of on MuJoCo after 1M timesteps, where shaded regions indicate the standard deviation. The state space is $84 \times 84 \times 1$.

utilize action signal, which is of a great importance signal for these games [26]; meanwhile, although GIRIL achieves decent performance on some environments, it does not solve DR problems directly, which may limit its performance; and HashReward provides a powerful approach for tackling high-dimensional IL problems, which can encourage the learner to generate expert comparable policies in challenging IL tasks.

5.3 Is HashReward Meaningful?

Comparisons on *Pong*. To find out whether HashReward has dug out the expert policy closely, we report the comparisons of a sequence of expert demonstrations and the sequence generated by each method during the same period of time on *Pong*, shown in Figure 4. We can observe that the expert hits the ball with the short side of the bar, which is the 'kill-shot' and hard to imitate. For contenders, they try to learn the 'kill-shot' but hit the ball by the long side of the bar instead of the short side, except for GAIL-AE-Up, GAIL-UH, GAIL-UH-Up, GIRIL, and VAIL which miss the ball. While the sequence of HashReward is the same as that of demonstrations, which shows that HashReward has successfully learned the 'kill-shot'.

Pseudo Reward Curves. In order to understand why HashReward outperforms its contenders and whether HashReward tackles the discrimination-rewarding trade-off, first we analyze the true reward and pseudo reward (generated by reward function) curves of five basic methods for a single training process on *Qbert*, illustrated in Figure 5. For GAIL and GAIL-AE, the discriminator seems to excessively focus on discriminating between the learner's and expert's samples, such that the pseudo reward does not rise even when the true reward has increased, which verifies our perspective



(a) Expert (b) GAIL (c) VAIL (d) GIRIL (e) AE (f) AE-Up (g) UH (h) UH-Up (i) HR-AE (j) HR

Figure 4: The sequence generated by each approach on *Pong*, with the comparison of expert demonstrations (the first column), where 'AE', 'AE-Up', 'UH', 'UH-Up', 'HR-AE' and 'HR' indicate GAIL-AE, GAIL-AE-Up, GAIL-UH, GAIL-UH-Up, HashReward-AE and HashReward respectively. The timestamp for each row of images and seed for each environment are the same.

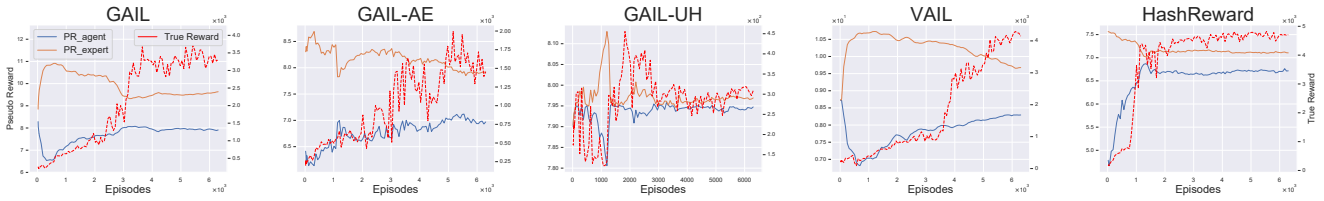
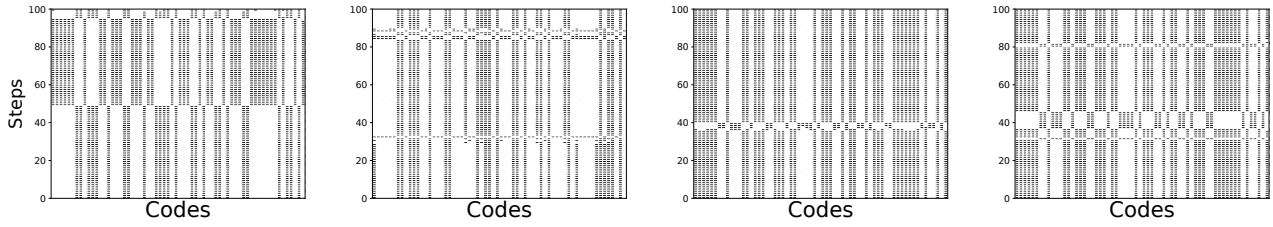


Figure 5: Pseudo reward generated by discriminator and true reward curves of five basic approaches on *Qbert*. 'PR_agent' and 'PR_expert' indicate pseudo reward for the agent and expert samples respectively. The blue curve denotes pseudo reward provided by the discriminator, and the red one denotes the true reward.



(a) 0 Step (b) 1e6 Steps (c) 1e7 Steps (d) Expert

Figure 6: The 64-bit HashReward codes for samples from different policy steps as well as expert on *Qbert* game. The samples are gathered from the same continual period within an episode with a fixed random seed.

on why GAIL (and with unsupervised DR) fails in such tasks. For GAIL-UH, the discriminator fails to discriminate between learner's and expert's samples, as the pseudo reward curve for learner almost overlaps with that for the expert in the whole training process. This indicates that including supervised information in hashing code training is indeed essential. For VAIL, the change of the pseudo reward for agent and expert is similar to that of GAIL, which reveals that the supervision of VAIL in DR is not enough. For HashReward, we can observe that the change of HashReward ideally reflects that of the true reward. Furthermore, the over-discrimination of

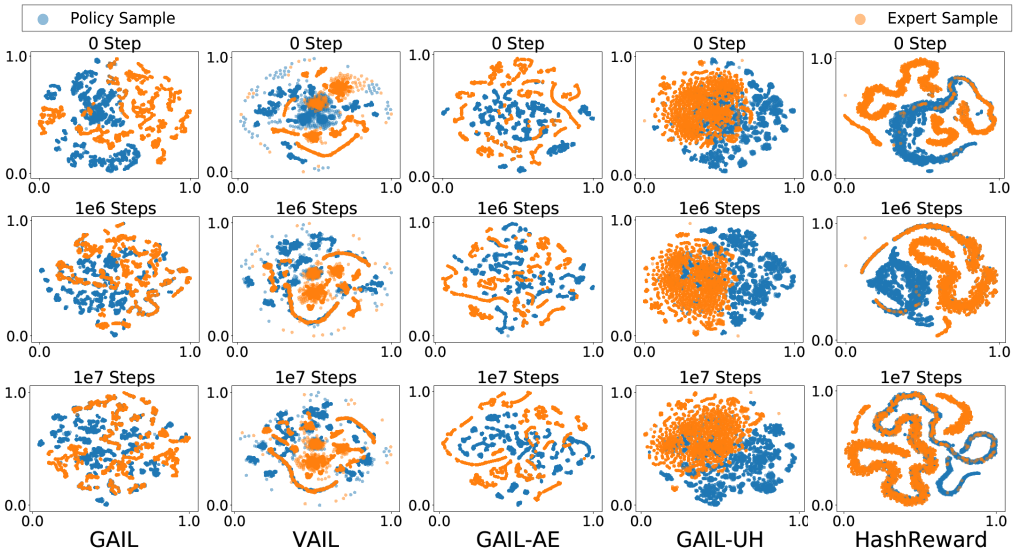


Figure 7: t-SNE Visualizations of policy samples (blue points) and expert samples (orange points) on *Qbert*.

pseudo reward is successfully avoided by HashReward. More results from the rest environments and other methods can be found in the supplementary material, which reflects similar phenomena.

64-bit Embeddings. Besides, to verify whether HashReward generates meaningful codes, we train a 64-bit embedding version of HashReward model on *Qbert*. Afterward we input samples generated from policies of different training stages (after 0 steps, 1e6 steps, and 1e7 steps) to the learned HashReward model to compare their codes. To collect samples, we run each policy for an episode with a fixed random seed and collect 100 samples within the same interval of time-steps. The codes are demonstrated in Figure 6. At 0 step and 1e6 step, the codes are clearly different from that of the expert, showing that the policies are not sufficiently trained. On the contrary, at 1e7 step, as the learner has dug out the latent expert policy, their codes are close to each other. This shows that HashReward learns a meaningful embedding space to reflect the quality of imitation.

t-SNE for Embeddings. Furthermore, Figure 7 illustrates t-SNE [27] visualization of the embedding layer outputs from five of the comparison methods under *Qbert*, and each figure contains 5000 policy and expert samples respectively. We show how embeddings of expert demonstrations, as well as policy samples, evolve for the same learning stages as above. We can observe that for GAIL and GAIL-AE, the embeddings for individual samples remain isolated throughout training, thus are easy to be fully discriminated. By using information bottleneck loss, VAIL partially solves this problem by generating some local clustered structures, which however still easy to be distinguished by the discriminator. For GAIL-UH, even though the embeddings seem to be globally clustered, we can observe that the policy and expert sample embeddings keep overlapping during the whole training process. This means that it fails to keep high-dimensional discriminative information when performing DR. For HashReward, we can observe that both expert and policy embeddings are globally clustered. Meanwhile, the overlaps between the two embedding sets increase along with the learning process, revealing the improvement of policy learning. This verifies HashReward could desirably dig out the hidden manifold structure of input space, which could lead to successful learning.

The above results reveal the close relationship between making the discriminator provide ground-truth consistent reward signals and properly dealing with the discrimination-rewarding trade-off. Meanwhile, they also indicate that effective supervision is essential for learning a proper discriminator, which is the key leading to the superior performance of HashReward in high-dim IL problems.

6 CONCLUSIONS

In this paper, we tackle the challenging problem of IL in high-dimensional environments, under which even state-of-the-art IL algorithms fail. Based on theoretical and empirical studies, we identify that such failure results from their improper treatment of tackling the discrimination-rewarding trade-off. Through this finding, we propose a novel high-dimensional IL method named HashReward, which utilizes supervised hashing to learn an effective discriminator, encouraging learners to dig out latent expert policies from demonstrations by providing efficient and stable reward signals. Experiments under both Atari and MuJoCo environments verify the effectiveness of HashReward, which outperforms state-of-the-art contenders with significant gaps. We expect HashReward can also provide inspirations in designing other hashing strategies containing effective semantic information, for solving other challenging IL problems, e.g., exploration-demanding games like *MontezumaRevenge*. We will explore more on these possibilities.

Acknowledgment: This research was supported by the NSFC (61673201, 61921006). The authors would like to thank Yang Yu, Jieping Ye, and the anonymous reviewers for their insightful comments and suggestions.

REFERENCES

[1] Pieter Abbeel and Andrew Y. Ng. 2010. Inverse Reinforcement Learning. In *Encyclopedia of Machine Learning*. 554–558.

[2] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. [n.d.]. Generalization and Equilibrium in Generative Adversarial Nets (GANs). In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017 (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). 224–232.

[3] Yusuf Aytar, Tobias Pfaff, David Budden, Thomas Paine, Ziyu Wang, and Nando de Freitas. 2018. Playing hard exploration games by watching YouTube. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*. 2935–2945.

[4] Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. 2013. The Arcade Learning Environment: An Evaluation Platform for General Agents. *J. Artif. Intell. Res.* 47 (2013), 253–279.

[5] Charles Blundell, Benigno Uria, Alexander Pritzel, Yazhe Li, Avraham Ruderman, Joel Z. Leibo, Jack W. Rae, Daan Wierstra, and Demis Hassabis. 2016. Model-Free Episodic Control. *CoRR* abs/1606.04460 (2016).

[6] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. *CoRR* abs/1606.01540 (2016).

[7] Daniel S. Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. 2019. Extrapolating Beyond Suboptimal Demonstrations via Inverse Reinforcement Learning from Observations. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. 783–792.

[8] Daniel S. Brown, Wonjoon Goo, and Scott Niekum. 2019. Better-than-Demonstrator Imitation Learning via Automatically-Ranked Demonstrations. In *Proceedings of the 3rd Conference on Robot Learning*.

[9] Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep Reinforcement Learning from Human Preferences. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. 4302–4310.

[10] Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, Yuhuai Wu, and Peter Zhokhov. 2017. OpenAI Baselines.

[11] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. 1999. Similarity Search in High Dimensions via Hashing. In *VLDB'99, Proceedings of 25th International Conference on Very Large Data Bases, September 7-10, 1999, Edinburgh, Scotland, UK*. 518–529.

[12] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, Gabriel Dulac-Arnold, John Agapiou, Joel Z. Leibo, and Audrunas Gruslys. 2018. Deep Q-learning From Demonstrations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. 3223–3230.

[13] Jonathan Ho and Stefano Ermon. 2016. Generative Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. 4565–4573.

[14] Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. 2018. Reward learning from human preferences and demonstrations in Atari. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*. 8022–8034.

[15] Qing-Yuan Jiang and Wu-Jun Li. 2018. Asymmetric Deep Supervised Hashing. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. 3342–3349.

[16] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).

[17] Haomiao Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. 2016. Deep Supervised Hashing for Fast Image Retrieval. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. 2064–2072.

[18] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[19] Xue Bin Peng, Angjoo Kanazawa, Sam Toyer, Pieter Abbeel, and Sergey Levine. 2019. Variational Discriminator Bottleneck: Improving Imitation Learning, Inverse RL, and GANs by Constraining Information Flow. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

[20] John Schulman, Sergey Levine, Pieter Abbeel, Michael I. Jordan, and Philipp Moritz. 2015. Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. 1889–1897.

[21] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347 (2017).

[22] Jing-Cheng Shi, Yang Yu, Qing Da, Shi-Yong Chen, and Anxiang Zeng. 2019. Virtual-Taobao: Virtualizing Real-World Online Retail Environment for Reinforcement Learning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. 4902–4909.

[23] Alexander L. Strehl and Michael L. Littman. 2008. An analysis of model-based Interval Estimation for Markov Decision Processes. *J. Comput. Syst. Sci.* 74, 8 (2008), 1309–1331.

[24] Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. 2017. #Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. 2750–2759.

[25] Emanuel Todorov, Tom Erez, and Yuval Tassa. [n.d.]. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*. 5026–5033.

[26] Aaron Tucker, Adam Gleave, and Stuart Russell. 2018. Inverse reinforcement learning for video games. *CoRR* abs/1810.10593 (2018).

[27] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9 (2008), 2579–2605.

[28] Haiyan Yin, Jianda Chen, and Sinno Jialin Pan. 2018. Hashing over Predicted Future Frames for Informed Exploration of Deep Reinforcement Learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. 3026–3032.

[29] Xingrui Yu, Yueming Lyu, and Ivor Tsang. 2020. Intrinsic Reward Driven Imitation Learning via Generative Model. In *International Conference on Machine Learning*. 6672–6682.

[30] Pengchuan Zhang, Qiang Liu, Dengyong Zhou, Tao Xu, and Xiaodong He. 2018. On the Discrimination-Generalization Tradeoff in GANs. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.