BEHAVIORAL CLONING FROM NOISY DEMONSTRATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider the problem of learning an optimal expert behavior policy given noisy demonstrations that contain observations from both optimal and non-optimal expert behaviors. Popular imitation learning algorithms, such as generative adversarial imitation learning, assume that (clear) demonstrations are given from optimal expert policies but the non-optimal ones, and thus often fail to imitate the optimal expert behaviors given the noisy demonstrations. Prior works that address the problem require (1) learning policies through environment interactions in the same fashion as reinforcement learning, and (2) annotating each demonstration with confidence scores or rankings. However, such environment interactions and annotations in real-world settings take impractically long training time and a significant human effort. In this paper, we propose an imitation learning algorithm to address the problem without any environment interactions and annotations associated with the non-optimal demonstrations. The proposed algorithm learns ensemble policies with a generalized behavioral cloning (BC) objective function where we exploit another policy already learned by BC. Experimental results show that the proposed algorithm can learn behavior policies that are much closer to the optimal policies than ones learned by BC.

1 INTRODUCTION

Imitation learning (IL) has become a widely used approach to obtain autonomous robotics control systems. IL is often more applicable in real-world problems than reinforcement learning (RL) since expert demonstrations are often easier than designing appropriate rewards that RL requires. There have been several IL methods that involve RL (Ziebart et al., 2008; Ng et al., 2000; Abbeel & Ng, 2004; Ho & Ermon, 2016). Those IL methods inherit sample complexity from RL in terms of environment interactions during training. The complexity restricts applicabilities in real-world problems since a number of environment interactions in real-world settings often take a long time and cause damage to the robot or the environment. Therefore, we are interested in IL methods that do not require the environment interactions, such as behavioral cloning (BC) (Pomerleau, 1991) which learns an expert policy in a supervised fashion.

BC as well as popular IL methods, such as generative adversarial imitation learning (GAIL) (Ho & Ermon, 2016), assume the expert demonstration is optimal. Unfortunately, it is often difficult to obtain optimal demonstrations for many tasks in real-world problems because the expert who tries to operate the robot so that it can achieve tasks often makes mistakes due to various reasons, such as the difficulty of the task, difficulty to handle the controller, limited observability of the environment, or the presence of distraction. The mistakes include unnecessary and/or incorrect operations to achieve the tasks. Given such *noisy* expert demonstrations, which contain records of both optimal and non-optimal behavior, BC as well as the popular IL methods fail to imitate the optimal policy due to the optimal assumption on the demonstrations as shown in (Wu et al., 2019).

A naive solution to cope with the noisy demonstrations is discarding the non-optimal demonstrations among the ones that were already collected. This screening process is often impractical because it involves a significant human effort. Most of recent IL works suppose settings where a very limited number of *clean* expert demonstrations, which are composed of only the optimal behavior records, are available. Those methods are also vulnerable to the noisy demonstrations due to the optimal assumption on the demonstrations. Thus they implicitly suppose such impractical screening process

if they were applied in real-world problems, where a number of the noisy demonstrations other than the clear ones can be easily obtained. There have been IL methods addressing the noisy demonstrations. Instead of the screening process, they require to annotate each demonstration with confidence scores (Wu et al., 2019) or rankings (Brown et al., 2019). Even though they cope well with the noisy demonstrations to obtain the optimal behavior policies, such annotation costs a significant human effort as it is for the screening. Hence, we desire IL methods that can cope well with the noisy demonstrations, which can be easily obtained in real-world settings, without any screening and annotation process associated with the non-optimal behaviors.

In this paper, we propose a novel imitation learning algorithm to address the noisy demonstrations. The proposed algorithm does not require (1) any environment interactions during training, and (2) any screening and annotation process associated with the non-optimality of the expert behaviors. The proposed algorithm learns ensemble policies with a generalized behavioral cloning (BC) objective function where we exploit another policy already learned by BC. Experimental results in simulation environments show that the proposed algorithm can learn policies that are much closer to the optimal than ones learned by BC.

2 Related Works

A wide variety of IL methods have been proposed in these last few decades. BC (Pomerleau, 1991) is the simplest IL method among those and thus BC could be the first IL option when enough clear demonstrations are available. Ross and Bagnell Ross & Bagnell (2010) have theoretically pointed out a downside of the BC which is referred to as *compounding error* – the small errors of the learners trained by BC could compound over time and bring about the deterioration of their performance. On the other hand, experimental results in (Sasaki et al., 2018) show that BC given the clear demonstrations of sufficient amounts can easily obtain the optimal behavior even for complex continuous control tasks. Hence, the effect of the compounding error can be negligible in practice if the amount of clear demonstrations is sufficient. However, even if the amount of the demonstrations is large, BC can not obtain the optimal policy given the noisy demonstrations due to the optimal assumption on the demonstrations. Another widely used IL approaches are inverse reinforcement learning (IRL) (Ziebart et al., 2008; Ng et al., 2000; Abbeel & Ng, 2004) and adversarial imitation learning (AIL) (Ho & Ermon, 2016). Since those approaches also assume the optimality of the demonstrations, they are also not able to obtain the optimal policy given the noisy demonstrations, as shown in (Wu et al., 2019). As we will show in Section 6, our algorithm successfully can learn near-optimal policies if noisy demonstrations of sufficient amounts are given.

There have been several works that address the noisy demonstrations (Wu et al., 2019; Brown et al., 2019; Tangkaratt et al., 2019; Kaiser et al., 1995; Grollman & Billard, 2012; Kim et al., 2013). Those works address the noisy demonstrations by either screening the non-optimal demonstrations with heuristic non-optimal assessments (Kaiser et al., 1995), or annotations associated with the non-optimality (Wu et al., 2019; Brown et al., 2019; Grollman & Billard, 2012), or training through the environment interactions (Kim et al., 2013; Wu et al., 2019; Brown et al., 2019; Tangkaratt et al., 2019). Our algorithm does not require any screening process, annotations associated with the non-optimality, and the environment interactions during training.

3 Preliminaries and Problem Setup

In this work, we consider an episodic fixed-horizon Markov Decision Process (MDP) which is formalized as a tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{P}, R, d_0, T\}$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of possible actions agents can take, $\mathcal{P} : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to [0, 1]$ is a transition probability, $R : \mathcal{S}\times\mathcal{A} \to [0, 1]$ is a reward function, $d_0 : \mathcal{S} \to [0, 1]$ is a distribution over initial states, and T is the episode horizon. The agent's behavior is defined by a stochastic policy $\pi : \mathcal{S}\times\mathcal{A} \to [0, 1]$ and Π denotes a set of the stochastic policies. The expected one-step immediate reward for a policy π given a state s is defined as $R^\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\big[R(s, a)\big]$.

Let d_t^π and $d^\pi = \frac{1}{T}\sum_{t=1}^{T} d_t^\pi$ denote the distribution over states at time step t and the average distribution over T time steps induced by π, respectively. The distributions d_1^π at the first step correspond to d_0 for any π. When following a policy π throughout an episode, the expected one-step immediate reward at time step t and the expected T-step reward are defined as

$R_t^\pi = \mathbb{E}_{s\sim d_t^\pi, a\sim\pi(\cdot|s)}\big[R(s,a)\big] = \mathbb{E}_{s\sim d_t^\pi}\big[R^\pi(s)\big]$ and $\mathcal{J}(\pi, R) = \sum_{t=1}^T R_t^\pi = T\mathbb{E}_{s\sim d^\pi}\big[R^\pi(s)\big]$, respectively. We refer to $\mathcal{J}(\pi, R)$ as *on-policy* expected T-step reward since the policy the expected reward is evaluated for and the one inducing the state distribution d^π are the same. We also consider another T-step reward defined by $\mathcal{J}_\beta(\pi, R) = T\mathbb{E}_{s\sim d^\beta}\big[R^\pi(s)\big]$, which we call *off-policy* expected T-step reward, where $\beta \in \Pi$ is a policy that can differ from π being evaluated, while the distribution d^β is induced by β.

In our problem setting, the functions R is not given. Instead, we observe noisy demonstrations. We refer to the agent that generates the noisy demonstrations as the *noisy expert*. The decision process turns to be MDP$\backslash\{R\}$ as the common imitation learning settings, and our problem can be formalized as to find an optimal policy in MDP$\backslash\{R\}$. Here we refer to the *true expert* policy as ones being able to take the optimal (thus not noisy) behavior in episode tasks. We make the following four assumptions to further formalize our problem setting:

Assumption 1. *The T-step expected reward of π_e^* satisfies $\mathcal{J}(\pi, R) \leq \mathcal{J}(\pi_e^*, R)$, $\mathcal{J}_\beta(\pi, R) \leq \mathcal{J}_\beta(\pi_e^*, R)$, and $\mathcal{J}_\beta(\pi_e^*, R) \leq \mathcal{J}(\pi_e^*, R)$ for any non-optimal policies $\pi, \beta \in \Pi \setminus \{\pi_e^*\}$.*

Assumption 2. *With small probability ϵ, which we call non-optimal probability, the policies π_e the noisy experts follow during demonstrations are sampled at each time step as $\pi_e = \pi \sim p_\Pi$ if $\epsilon \geq z \sim \mathcal{U}(0,1)$, otherwise $\pi_e = \pi_e^*$, where p_Π is an unknown distribution over the set of policies, z is a random variable, and $\mathcal{U}(0,1)$ is a uniform distribution with range $[0,1]$.*

Assumption 3. *The reward $R_t^{\pi_e}$ is at least zero if the noisy expert has followed a policy $\pi \in \Pi\backslash\{\pi_e^*\}$ once or more so far, otherwise $R_t^{\pi_e} = \mathbb{E}_{s\sim d_t^{\pi_e}}\big[\epsilon\mathbb{E}_{\pi\sim p_\Pi}[R^\pi(s)] + (1-\epsilon)R^{\pi_e^*}(s)\big]$.*

Assumption 4. *The sequence $\{R_1^{\pi_e}, ..., R_T^{\pi_e}\}$ has monotonically decreasing property $R_t^{\pi_e} \geq R_{t+1}^{\pi_e}$.*

Assumption 1 indicates that both on-policy and off-policy expected T-step reward following π_e^* are always greater than or equal to ones following any other policies. In other words, we assume the true expert policy is an optimal one in the MDP, and the agent following the policy able to behave so that the expected immediate rewards at any states are maximized. Assumption 2 indicates that the noisy expert occasionally adopts non-optimal policies, which results in the noisy demonstrations, due to random events, such as the presence of distractions, associated with the random variable z. Assumption 3 indicates that states unseen from optimal behaviors could be observed with a high probability if the noisy expert followed non-optimal policies even once so far, and the rewards at those unseen states are lower than $R^{\pi_e^*}(s)$. Assumption 4 indicates that, since the probability the noisy expert consecutively follows the imaginary optimal policy decreases as time step increases according to Assumption 2, the immediate reward decrease as the number of time step increases according to Assumption 3.

4 Analysis Of Performance Deterioration

In this section, we firstly describe BC objective in 4.1. Then, we analyze why the learner trained by BC deteriorates its performance when using the noisy demonstrations from *the expected T-step reward maximization* and *KL-divergence minimization* perspectives in 4.2 and 4.3, respectively.

4.1 Behavioral Cloning Objective

Let $\pi_\theta \in \Pi$ is a learner policy parameterized by θ to be optimized by IL algorithms. The objective of BC in common is as follows:

$$\arg\max_\theta \mathbb{E}_{s\sim d^{\pi_e}, a\sim\pi_e(\cdot|s)}[\log \pi_\theta(a|s)] \tag{1}$$

4.2 The Expected T-step Reward Maximization

Assume that the policy π_θ trained by BC given the noisy demonstration has a probability of non-optimal behavior $\hat{\epsilon} = \epsilon + \zeta$ at most, where ζ is due to the remained loss in (1). Then we obtain the following lower bound of the expected T-step reward for π_θ in almost the same way to derive Theorem 2.1. in (Ross & Bagnell, 2010) where they showed the lower bound for the learner policies given the "clear" expert demonstrations.

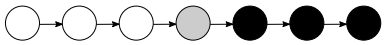

Figure 1: An example of the state trajectory observed in the noisy expert demonstrations. The white circles depict the states in $S_*^{\pi_e}$ where the noisy expert adopts π_e^* from the beginning. The gray circle depicts the states in $S_{e+*}^{\pi_e}$ where the noisy expert adopts $\pi \in \Pi \setminus \{\pi_e^*\}$. The black circles depict the states in $S_e^{\pi_e}$ after adopting $\pi \in \Pi \setminus \{\pi_e^*\}$.

Corollary 1. *If the Assumptions 1 - 4 hold and the policy π_θ has a probability of non-optimal behavior $\hat{\epsilon} = \epsilon + \zeta$, the following bound holds:*

$$\mathcal{J}(\pi_\theta, R) \geq \left\{ \sum_{t=0}^{T-1} (1-\hat{\epsilon})^t \right\} \cdot \mathcal{J}_{\pi_e}(\pi_\theta, R) \tag{2}$$

The detailed derivation can be found in Appendix A.1. This corollary shows that the probability of non-optimal behavior ϵ of the noisy expert significantly affects the learner performance $\mathcal{J}(\pi_\theta, R)$ even if ζ of the learner can be sufficiently minimized. It also shows that BC can not boost $\mathcal{J}(\pi_\theta, R)$ up to $\mathcal{J}(\pi_e^*, R)$ as long as $\zeta > 0$ since $\mathcal{J}_{\pi_e}(\pi_\theta, R) \leq \mathcal{J}(\pi_e^*, R)$ according to Assumption 1.

4.3 KL Divergence Minimization

Let S^{π_e} is a set of states that are observed in a finite number of the noisy expert demonstrations. S_{π_e} can be thought of as the domain of (empirical) state distributions d^{π_e}. We can decompose the set into three disjoint sets as $S^{\pi_e} = S_e^{\pi_e} \cup S_*^{\pi_e} \cup S_{e+*}^{\pi_e}$, where $S_e^{\pi_e}$ is one observed after the noisy expert adopts $\pi \in \Pi \setminus \{\pi_e^*\}$ in the episode, $S_*^{\pi_e}$ is one observed when the noisy expert adopts π_e^* sequentially from the beginning of the episode while not adopting any other policies over the episodes, and $S_{e+*}^{\pi_e}$ is one observed when the noisy expert adopts both π_e^* and $\pi \in \Pi \setminus \{\pi_e^*\}$ after sequentially adopting π_e^* from the beginning. Figure 1 graphically describes each set. Let $d_e^{\pi_e}$, , $d_*^{\pi_e}$ and $d_{e+*}^{\pi_e}$ be the state distributions the noisy expert policy induces in $S_e^{\pi_e}$, $S_*^{\pi_e}$, and $S_{e+*}^{\pi_e}$, respectively. Then we can define d^{π_e} as a mixture of those distributions as follows:

$$d^{\pi_e}(s) = \alpha d_e^{\pi_e}(s) + \beta d_*^{\pi_e}(s) + \gamma d_{e+*}^{\pi_e}(s) \tag{3}$$

where α, β, and γ are ratios the noisy expert enters states that belong to $S_e^{\pi_e}$, $S_*^{\pi_e}$, and $S_{e+*}^{\pi_e}$ during demonstrations respectively, and satisfy $\alpha + \beta + \gamma = 1$. Using Equation (3), the upper bound of loss function in Equation (1) is derived as follows:

$$\begin{aligned}
\mathbb{E}_{s\sim d^{\pi_e}, a\sim\pi_e(\cdot|s)}[\log \pi_\theta(a|s)] &\leq -\alpha\Omega_e(\theta) - \beta\Omega_*(\theta) - \gamma\Omega_{e+*}(\theta) && (4)\\
\Omega_e(\theta) &= \mathbb{E}_{s\sim d_e^{\pi_e}}[D_{KL}[\pi_e(\cdot|s)||\pi_\theta(\cdot|s)]] && (5)\\
\Omega_*(\theta) &= \mathbb{E}_{s\sim d_*^{\pi_e}}[D_{KL}[\pi_e^*(\cdot|s)||\pi_\theta(\cdot|s)]] && (6)\\
\Omega_{e+*}(\theta) &= \epsilon\mathbb{E}_{s\sim d_*^{\pi_{e+*}}, \pi\sim p_\Pi}[D_{KL}[\pi(\cdot|s)||\pi_\theta(\cdot|s)]] \\
&+ (1-\epsilon)\mathbb{E}_{s\sim d_*^{\pi_{e+*}}}[D_{KL}[\pi_e^*(\cdot|s)||\pi_\theta(\cdot|s)]] && (7)
\end{aligned}$$

where D_{KL} is forward Kullback-Leibler (KL) divergence. The full derivation can be found in Appendix A.2. Regarding the inequality (4), the BC objective with noisy demonstrations (1) is to minimizes the sum of KL-divergences (5)-(7). The inequality (4) indicates that which term mainly makes BC difficult to learn π_e^* from noisy demonstrations. From a conclusion, minimizing KL-divergence (7) makes it difficult.The set $S_e^{\pi_e}$ has states that the true expert rarely visits. Since the objective of IL is to obtain policies that can generate or keep being on optimal state trajectories as the true expert does, minimizing the divergence (5) over the states that are out of the optimal trajectories have a little effect to learn the π_e^*. The divergence (6) is not associated with the non-optimal policy $\pi \in \Pi \setminus \pi_e^*$. Thus, minimizing the divergence (6) does not cause the difficulty to learn π_e^*. Unlike the divergences (5) and (6), there are two terms of the KL-divergence in Equation (7). The first term leads the learner to imitate some non-optimal behaviors whereas the second term is to learn π_e^* on the same states. The first term can be interpreted as a *"noisy" regularizer* with a coefficient ϵ that makes the learner confused to learn π_e^*. Thus, Equation (7) clearly shows that the learner trained by BC more often tends to behave the non-optimal behavior as ϵ increases on the state set $S_{e+*}^{\pi_e}$. It suggests that minimizing the KL divergences in (7) mainly makes BC difficult to learn π_e^* from noisy demonstrations.

Algorithm 1 Overview of Our Algorithm

Given the expert demonstrations $\mathcal{D}$.
Set $\hat{R}(s,a) = 1$ for $\forall (s,a) \in \mathcal{D}$.
for iteration = $1, M$ **do**
  for $k = 1, K$ **do**
    Initialize parameters θ^k.
    Optimize θ^k with the objective (1).
  end for
  Set $\hat{R}(s,a) = \eta \bar{\pi}_\theta(a|s)$ for $\forall (s,a) \in \mathcal{D}$.
end for

5 ALGORITHM

In this section, we first propose our algorithm that learns π_e^* while reducing the influence of the noisy regularizer in Equation (7). Then we describe how our algorithm works to reduce the influence. We lastly provide limitations of our algorithm.

5.1 PROPOSED ALGORITHM

We consider a generalization of the BC objective as follows:

$$\arg\max_\theta \mathbb{E}_{s\sim d^{\pi_e}, a\sim\pi_e(\cdot|s)}[\log \pi_\theta(a|s)\hat{R}(s,a)] \tag{8}$$

where $\hat{R}$ denotes an arbitrary function which can differ from R. If $\hat{R}(s,a) = 1$ for $\forall (s,a) \in \mathcal{S} \times \mathcal{A}$, the objective (8) corresponds to the BC objective (1). If $\int_{\mathcal{A}} \hat{R}(s,a)da = 1$ for $\forall s \in \mathcal{S}$ is satisfied, $\hat{R}(s,a)$ can be interpreted as weights for action samples obtained by the demonstrations so that the actions are sampled according to their relative weights. The objective (8) can also be deemed as that of the off-policy actor-critic (Off-PAC) algorithm [1] (Degris et al., 2012) with reward functions $\hat{R}(s,a)$ and zero discount factors.

Let $\pi_{\theta^1}, \pi_{\theta^2}, ..., \pi_{\theta^K}$ be K parameterized policies with different initial parameters $\theta^1, \theta^2, ..., \theta^K$, and $\pi_\theta(a|s) = \sum_{k=1}^K \pi_{\theta^k}(a|s)/K$ denotes an ensemble of the parameterized policies with parameters $\theta = \{\theta^1, \theta^2, ..., \theta^K\}$. Let $\pi_{\theta_{old}}$ be a parameterized policy with θ_{old} which was already optimized with the noisy demonstrations. The main idea of our algorithm is to reuse the old policy $\pi_{\theta_{old}}$ as $\hat{R}(s,a)$ in the generalized BC objective (8).

$$\arg\max_\theta \mathbb{E}_{s\sim d^{\pi_e}, a\sim\pi_e(\cdot|s)}[\log \pi_\theta(a|s)\{\eta \pi_{\theta_{old}}(a|s)\}] \tag{9}$$

where $\eta > 0$ is a scaling coefficient. The overview of our algorithm is described in Algorithm 1.

5.2 WEIGHTED ACTION SAMPLING FOR π_e^* MODE SEEKING

Since $\pi_{\theta_{old}}$ satisfies $\int_{\mathcal{A}} \pi_{\theta_{old}} da = 1$ for $\forall s \in \mathcal{S}$, $\pi_{\theta_{old}}$ can be interpreted as the weights for the weighted action sampling. We below explain the weighted action sampling procedure in our algorithm on $S_*^{\pi_e}$[2]. Figure 2 depicts a toy example of the sampling procedure. The distribution of the noisy expert actions on $S_*^{\pi_e}$ is a mixture of two distributions as shown in Equation (7). If ϵ is sufficiently small, π_θ is optimized so that its mode is closer to that of π_e^* than $\pi \in \Pi \setminus \{\pi_e^*\}$ according to mode seeking properties of the forward KL divergence (Ghasemipour et al., 2020). Given the sampling weights $\pi_{\theta_{old}}(a|s) = \pi_\theta(a|s)$ for the empirical action samples, the weighted action distribution distorts so that its mode also gets closer to π_e^*. By iteratively distorting the weighted action distribution with the same procedure, its mode fits to near the mode of $S_*^{\pi_e}$. Whereas, the weights for actions sampled from $\pi \in \Pi \setminus \{\pi_e^*\}$ eventually become much smaller, and thus the influence

[1] Although Off-PAC multiplies $\log \pi_\theta(a|s)$ by a density ratio $\pi_e(s|a)/\pi_\theta(s|a)$, $\pi_\theta(s|a)$ is empirically approximated to be one in popular off-policy RL algorithms such as DDPG (Lillicrap et al., 2015).

[2] Since the set of state $S_e^{\pi_e} \setminus S_*^{\pi_e}$ have just a little effect to reduce the influence of noisy regularizer as mentioned in 4.3, we omit the explanation on those sets due to the space limit.

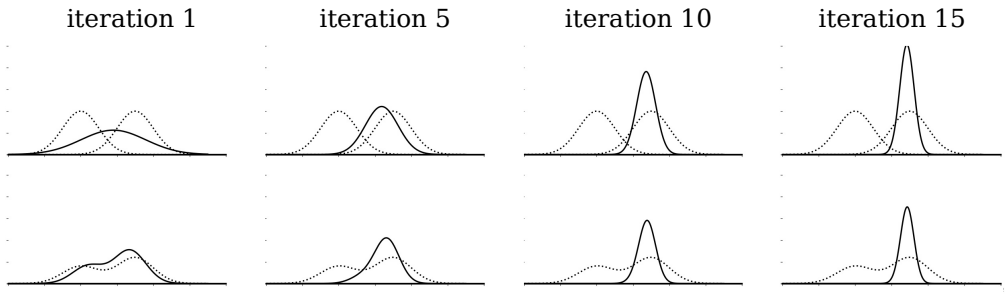

Figure 2: A toy example of the weighted action sampling procedure given a state $s \in S_*^{\pi_e}$ in our algorithm. On both rows, the horizontal line is the action domain. On the top row, the left dotted lines, the right dotted lines, and the solid line describe $\pi \in \Pi \setminus \{\pi_e^*\}$, $\pi_e^*(a|s)$, and $\pi_\theta(a|s)$, respectively. On the bottom row, the dotted lines and the solid line describe the mixture distribution $\epsilon\pi(a|s)+(1-\epsilon)\pi_e^*(a|s)$ with $\epsilon = 0.4$ and the mixture distribution weighted by $\pi_\theta(a|s)$, respectively.

of the noisy regularizer in Equation (7) can be reduced. The mode seeking procedure of our algorithm is analogous to the mean shift algorithm (Fukunaga & Hostetler, 1975) so that the mode of π_θ shifts towards that of π_e^* by minimizing the KL divergence between π_θ and the weighted action distribution.

5.3 Reward Maximization

As the Off-PAC objective, the objective (9) maximizes the expected (one-step) reward $\hat{R}(s,a) = \eta\pi_{\theta_{old}}(a|s)$. Recall that the learner policy $\pi_\theta(a|s) = \sum_{k=1}^{K} \pi_{\theta^k}(a|s)/K$ is an ensemble of the parameterized policies in our algorithm. Following the work in (Perrone, 1993), we obtain:

$$\frac{1}{K}\sum_{k=1}^{K} \mathbb{E}_{s\sim d^{\pi_e}, a\sim\pi_e(\cdot|s)}[\log \pi_{\theta^k}(a|s)\{\eta\pi_{\theta_{old}}(a|s)\}] \leq \mathbb{E}_{s\sim d^{\pi_e}, a\sim\pi_e(\cdot|s)}\left[\log \pi_\theta(a|s)\{\eta\pi_{\theta_{old}}(a|s)\}\right] \tag{10}$$

where we use Jensen's inequality with the concave property of logarithm : $\frac{1}{K}\sum_{k=1}^{K} \log \pi_{\theta^k}(a|s) \leq \log \pi_\theta(a|s)$. The inequality (10) indicates that the ensemble of policies $\pi_{\theta^1}, \pi_{\theta^2}, ..., \pi_{\theta^K}$, each of which was learned with (9), has greater or equal values of the objective function in (9) than the averaged value over the policies in the ensemble. As mentioned in 5.2, $\pi_{\theta_{old}}(a|s)$ becomes higher near the mode of π_e^*. Thus, making π_θ as the ensemble further encourages to shift its mode to that of π_e^* and reduce the influence of the noisy regularizer in Equation (7).

5.4 Limitations

Our algorithm has three limitations. First, $K \times M$ times computational cost is required in comparison with BC, where M is the number of iterations in Algorithm 1. Second, the compounding error due to the probability of non-optimal behavior ζ still remains unless sufficient amounts of the demonstrations are given. Lastly, π_θ is fitting to $\pi \in \Pi \setminus \{\pi_e^*\}$ rather than π_e^* if the major mode of $\epsilon\pi(a|s) + (1-\epsilon)\pi_e^*(a|s)$ due to the higher kurtosis of $\pi(a|s)$ or ϵ of large values.

6 Experiments

In our experiments, we aim to answer the following three questions:

- Q1. Does our algorithm improve the learner performance more than BC given the noisy demonstrations?
- Q2. Can the compounding error due to ζ be reduced as the number of noisy expert demonstrations increase?
- Q3. Is our algorithm competitive to the existing IL methods if both annotations associated with the non-optimality and environment interactions are allowed?

6.1 SETUP

To answer Q1 and Q2, we evaluated our algorithm against BC on four continuous control tasks (Ant-v2, HalfCheetah-v2, Walker2d-v2, and Hopper-v2) that are simulated with MuJoCo physics simulator (Todorov et al., 2012). We train an agent on each task by proximal policy optimization (PPO) algorithm (Schulman et al., 2017) using the rewards defined in the OpenAI Gym (Brockman et al., 2016). We use the resulting stochastic policy as the true expert policy π_e^*. We generate the noisy expert demonstrations using π_e^* while randomly adopting non-optimal policies π with probabilities of the non-optimal behavior ϵ. The non-optimal policies $\pi \in \Pi \setminus \pi_e^*$ are selected from uniform distributions $a \sim \mathcal{U}(0, I)$, Gaussian distributions $a \sim \mathcal{N}(a^*, I)$ with $a \sim \pi_e^*(\cdot|s)$, or a deterministic policy $a = 0$. ϵ are selected from $\{0.0, 0.1, 0.2, 0.3, 0.4, 0.5\}$. The noisy expert takes actions following π_e^* if $z \geq \epsilon$ otherwise π which is fixed to a selected one through an episode, where $z \sim \mathcal{U}(0, 1)$. Each noisy demonstration with the selected ϵ consists of N state-action pairs, where N is selected from $\{5000, 10000, 50000, 100000\}$. Then we perform our algorithm as well as BC to train the learners using each noisy demonstration. We also conducted the same experiments on four low-dimensional discrete control tasks (CartPole-v1, Acrobot-v1, MountainCar-v0, and LunarLander-v2). The results of the discrete control tasks can be found in Appendix A.4

To answer Q3, we evaluated our algorithm against IC-GAIL (Wu et al., 2019), 2IWIL (Wu et al., 2019), and T-REX (Brown et al., 2019) on three continuous control tasks (Ant-v2, HalfCheetah-v2, and Hopper-v2). The true expert policy π_e^* are obtained in the same way as mentioned above. The non-optimal policies $\pi \in \Pi \backslash \pi_e^*$ is fixed to $a \sim \mathcal{U}(0, I)$. We generate the noisy expert demonstrations which consists of 10000 state-action pairs for each $\epsilon \in \{0.05, 0.1, 0.15, ..., 1.0\}$. Then we perform our algorithm and the baselines using all noisy demonstrations. The detailed description of this experimental setup can be found in Appendix A.3.

In both experiments, the performance of the learners is measured by cumulative rewards they earned in a trajectory. The cumulative reward is normalized with ones earned by π_e^* and a random policy $a \sim \mathcal{U}(0, I)$ so that 1.0 and 0.0 indicate the performance of π_e^* and the random policy, respectively. We run five experiments on each task and setup, and measure the mean and standard deviation of the normalized cumulative rewards for each learner over the five experiments. In all experiments, we set the number of policies $K = 5$ in the ensemble learner policy π_θ. The implementation details of our algorithm on both experiments can be found in Appendix A.5.

6.2 RESULTS

Figure.3 depicts the experimental results against BC. Over all tasks, our algorithm obtains much better learner performance than BC-Single, which is a single (thus not an ensemble) policy learned by BC. It suggests that the policies learned by our algorithm are closer to π_e^* than ones learned by BC. The compounding error due to ζ is expected to be reduced as the number of demonstrations increase (Sasaki et al., 2018). Whereas BC-Ensemble which denotes the ensemble of policies learned by BC yields significant performance gains against BC-Single, increasing the number of noisy demonstrations has a little effect to boost the learner performance trained by BC-Ensemble as shown in Figure.3-(D). It indicates that BC-Ensemble can not reduce the compounding error due to ϵ. On the other hand, our algorithm can boost the learner performance up to that of π_e^* as increasing the number of demonstrations. It suggests that our algorithm can reduce the compounding error due to both ϵ and ζ if sufficient amounts of the noisy expert demonstrations are given, as is the case for BC with the clean expert demonstrations. The results with the deterministic non-optimal policy $\pi \in \Pi \setminus \{\pi_e^*\}$ which always takes an action $a = 0$ are worse than those with other non-optimal policies. It corresponds to the limitation of our algorithm as mentioned in 5.4, since the major mode of $\epsilon\pi(a|s) + (1 - \epsilon)\pi_e^*(a|s)$ might be around $a = 0$. We also conducted ablation experiments where the number of policies K are selected from $\{1, 5\}$ in our algorithm. See Appendix A.6 for details. The ablation experimental results show that the learner obtains better performance if K increases. In addition, the performance of the learner trained by our algorithm is significantly better than that of BC-Single even though $K = 1$. It suggests that our algorithm improves the learner performance by not only the ensemble approach but also using the old policies $\pi_{\theta_{old}}$.

Table.1 depicts the experimental results against IC-GAIL, 2IWIL and T-REX. In this experiment, these methods except for our algorithm use annotations associated with the non-optimality of the demonstration and interact with the environment during training. Over all tasks, 2IWIL and our

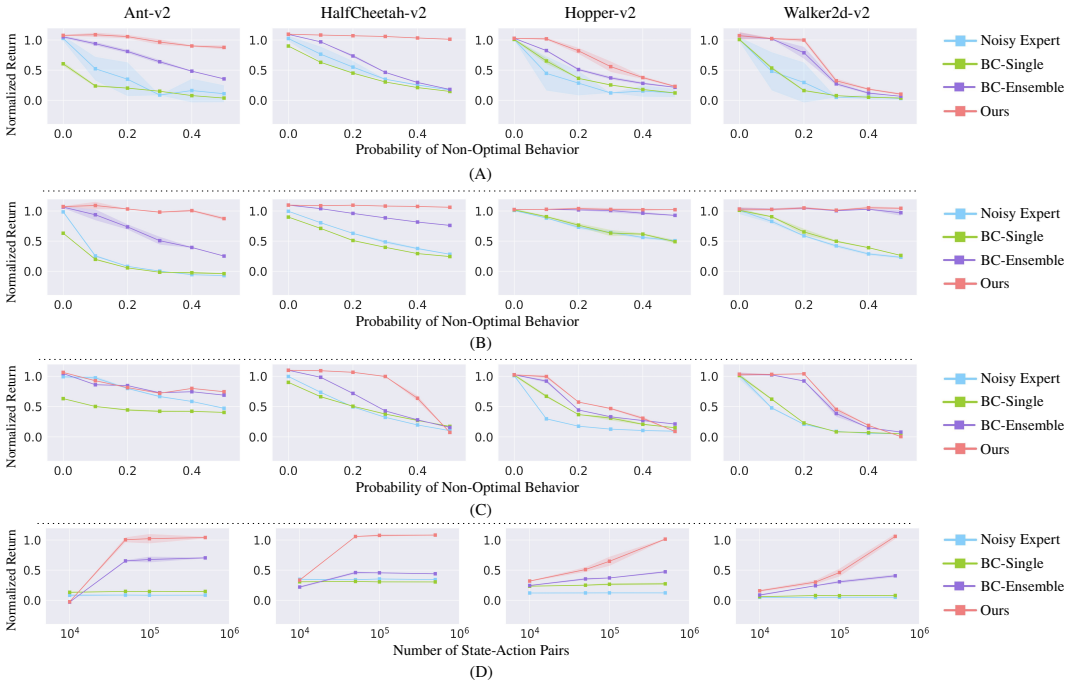

Figure 3: (A)-(C) The performance of policies vs. ϵ given 50000 state-action pairs of the noisy expert demonstrations where the non-optimal policies $\pi \in \Pi \setminus \{\pi_e^*\}$ are (A) $\mathcal{U}(0, I)$, (B) $\mathcal{N}(a^*, I)$ with $a \sim \pi_e^*(\cdot|s)$, and (C) the deterministic one $a = 0$, respectively. (D) The performance of policies vs. the number of state-action pairs N of the noisy demonstrations with $\epsilon = 0.3$ where $\pi(a|s) = \mathcal{U}(0, I)$. BC-Single is a policy learned by BC. BC-Ensemble is an ensemble of policies, each of which was learned by BC. Shaded regions indicate the standard deviation over five experiments.

algorithm can successfully obtain the true expert performance while others can not. It suggests that, our algorithm can obtain competitive results with that of existing IL methods even though the annotation and the environment interactions are not used.

Table 1: The experimental results against IC-GAIL, 2IWIL and T-REX.

	Ant-v2	HalfCheetah-v2	Hopper-v2
IC-GAIL	0.6313 ± 0.162	0.9416 ± 0.103	1.2334 ± 0.152
2IWIL	1.0420 ± 0.021	1.0248 ± 0.059	1.2238 ± 0.135
T-REX	0.5862 ± 0.124	0.0010 ± 0.113	0.4414 ± 0.219
Ours	1.0559 ± 0.053	1.0932 ± 0.092	1.0035 ± 0.045

7 Conclusion

In this paper, we proposed an imitation learning algorithm to copes with the noisy expert demonstrations. Experimental results showed that our algorithm can learn behavior policies that are much closer to the true expert policies than ones learned by the BC. Since our algorithm cope well with the noisy expert demonstrations while not requiring any environment interactions and annotations associated with the non-optimal demonstrations, our algorithm is more applicable to real-world problems than the prior works such as (Wu et al., 2019; Brown et al., 2019). Although our algorithm has a few limitations as mentioned in 5.4, we believe that the analysis of performance deterioration detailed in Section.4 contributes to step forward for solving the noisy demonstration problems. In future work, we will consider the setting where the probability of non-optimal behavior is state-dependent, which often occurs in the real world more than the state-independent case that we consider in this paper.

REFERENCES

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 1, 2004.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Daniel S Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. *arXiv preprint arXiv:1904.06387*, 2019.

Thomas Degris, Martha White, and Richard S Sutton. Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*, 2012.

Keinosuke Fukunaga and Larry Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on information theory*, 21(1):32–40, 1975.

Seyed Kamyar Seyed Ghasemipour, Richard Zemel, and Shixiang Gu. A divergence minimization perspective on imitation learning methods. In *Conference on Robot Learning*, pp. 1259–1277. PMLR, 2020.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

Daniel H Grollman and Aude G Billard. Robot learning from failed demonstrations. *International Journal of Social Robotics*, 4(4):331–342, 2012.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in neural information processing systems*, pp. 4565–4573, 2016.

Michael Kaiser, Holger Friedrich, and Rudiger Dillmann. Obtaining good performance from a bad teacher. In *Programming by Demonstration vs. Learning from Examples Workshop at ML*, volume 95, 1995.

Beomjoon Kim, Amir-massoud Farahmand, Joelle Pineau, and Doina Precup. Learning from limited demonstrations. In *Advances in Neural Information Processing Systems*, pp. 2859–2867, 2013.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, pp. 2, 2000.

Michael Peter Perrone. *Improving Regression Estimation: Averaging Methods for Variance Reduction with Extensions to General Convex Measure Optimization*. PhD thesis, Brown University, 1993.

Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 661–668, 2010.

Fumihiro Sasaki, Tetsuya Yohira, and Atsuo Kawaguchi. Sample efficient imitation learning for continuous control. In *International Conference on Learning Representations*, 2018.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Voot Tangkaratt, Bo Han, Mohammad Emtiyaz Khan, and Masashi Sugiyama. Vild: Variational imitation learning with diverse-quality demonstrations. *arXiv preprint arXiv:1909.06769*, 2019.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Yueh-Hua Wu, Nontawat Charoenphakdee, Han Bao, Voot Tangkaratt, and Masashi Sugiyama. Imitation learning from imperfect demonstration. *arXiv preprint arXiv:1901.09387*, 2019.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pp. 1433–1438. Chicago, IL, USA, 2008.

A APPENDIX

A.1 DETAILED DEVIATION OF THE COROLLARY 1

We obtain the lower bound of the expected T-step reward for the noisy expert policy in almost the same way to derive Theorem 2.1. in (Ross & Bagnell, 2010) where they showed the lower bound for the learner policies rather than that for the noisy expert.

Theorem 1. *If the Assumptions 1 - 4 hold, the following bound holds:*

$$\mathcal{J}(\pi_e, R) \geq \left\{ \sum_{t=0}^{T-1} (1-\epsilon)^t \right\} \cdot \mathbb{E}_{\pi \sim p_\Pi}[\mathcal{J}_{\pi_e}(\pi, R)] \tag{11}$$

Proof. Let $q_t = (1-\epsilon)^t$ denotes the probability the noisy expert consecutively follows π_e^* in the first t step, and $\chi = \sum_{t=1}^{T} q_{t-1}$ denotes sum of q_{t-1} over time steps. Then we obtain:

$$\begin{aligned}
\mathcal{J}(\pi_e, C) &\geq \sum_{t=1}^{T} q_{t-1} R_t^{\pi_e} + (1 - q_{t-1}) \cdot 0 && (12)\\
&\geq T \left\{ \frac{1}{T} \sum_{t=1}^{T} q_{t-1} \right\} \left\{ \frac{1}{T} \sum_{t=1}^{T} R_t^{\pi_e} \right\} && (13)\\
&= \chi \left\{ \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}_{s \sim d_t^{\pi_e}} \left[\epsilon \mathbb{E}_{\pi \sim p_\Pi}[R^\pi(s)] + (1-\epsilon) R^{\pi_e^*}(s) \right] \right\} \\
&= \chi \left\{ \epsilon \mathbb{E}_{\pi \sim p_\Pi}[\mathcal{J}_{\pi_e}(\pi, R)] + (1-\epsilon) \mathcal{J}_{\pi_e}(\pi_e^*, R) \right\} \\
&\geq \chi \left\{ \epsilon \mathbb{E}_{\pi \sim p_\Pi}[\mathcal{J}_{\pi_e}(\pi, R)] + (1-\epsilon) \mathbb{E}_{\pi \sim p_\Pi}[\mathcal{J}_{\pi_e}(\pi, R)] \right\} && (14)\\
&= \left\{ \sum_{t=0}^{T-1} (1-\epsilon)^t \right\} \mathbb{E}_{\pi \sim p_\Pi}[\mathcal{J}_{\pi_e}(\pi, R)]
\end{aligned}$$

The first inequality (12) is from Assumption 2 and 3. The second inequality (13) is from Chebyshev's sum inequality with the monotonically decreasing properties according to Assumption 4. The third inequality (14) is from Assumption 1 : $\mathcal{J}_\beta(\pi, R) \leq \mathcal{J}_\beta(\pi_e^*, R)$ for any $\pi, \beta \in \Pi \setminus \{\pi_e^*\}$. □

Assume that the policy π_θ has a probability of non-optimal behavior $\hat{\epsilon} = \epsilon + \zeta$ as the result of BC, where ζ is the additional probability of non-optimal behavior which causes further compounding error. The agent following a parameterized policy π_θ with $\hat{\epsilon}$ can be deemed as another noisy expert who samples a policy at each time step $\pi_\theta = \pi \sim p_{\pi\theta}$ if $\hat{\epsilon} \geq z \sim \mathcal{U}(0, 1)$, otherwise $\pi_\theta(a|s) = \pi_e^*(a|s)$, where $p_{\pi\theta}$ is a (special) distribution from which the same policy is always sampled. By replacing $\hat{\epsilon}$ and p_{π_θ} from ϵ and p_π in Theorem 1 respectively, we immediately obtain Corollary 1.

A.2 DETAILED DERIVATION OF THE DECOMPOSED KL DIVERGENCE

From the definition of (3), we obtain:

$$\begin{aligned}
\mathbb{E}_{s \sim d^{\pi_e}, a \sim \pi_e(\cdot|s)}[\log \pi_\theta(a|s)] &= \alpha \mathbb{E}_{s \sim d_e^{\pi_e}, a \sim \pi_e(\cdot|s)}[\log \pi_\theta(a|s)] && (15)\\
&+ \beta \mathbb{E}_{s \sim d_*^{\pi_e}, a \sim \pi_e^*(\cdot|s)}[\log \pi_\theta(a|s)] && (16)\\
&+ \gamma \mathbb{E}_{s \sim d_{e+*}^{\pi_e}, a \sim \pi_e(\cdot|s)}[\log \pi_\theta(a|s)] && (17)
\end{aligned}$$

The forward Kullback-Leibler (KL) divergence D_{KL} between π_e and π_θ over a state distribution d_{π_e} is defined as $\mathbb{E}_{s \sim d^{\pi_e}}[D_{KL}(\pi_e(\cdot|s) || \pi_\theta(\cdot|s))] = -\mathbb{E}_{s \sim d^{\pi_e}}[\mathbb{E}_{a \sim \pi_e(\cdot|s)}[\log \pi_\theta(a|s)] + H[\pi_e(\cdot|s)]]$, where H denotes the entropy. Note that the expectation is taken over π_e^* rather than π_e in (16) since the noisy expert follows π_e^* in $\mathcal{S}_*^{\pi_e}$ from the definition in 4.3. Since $H[\pi_e(\cdot|s)]$ always takes

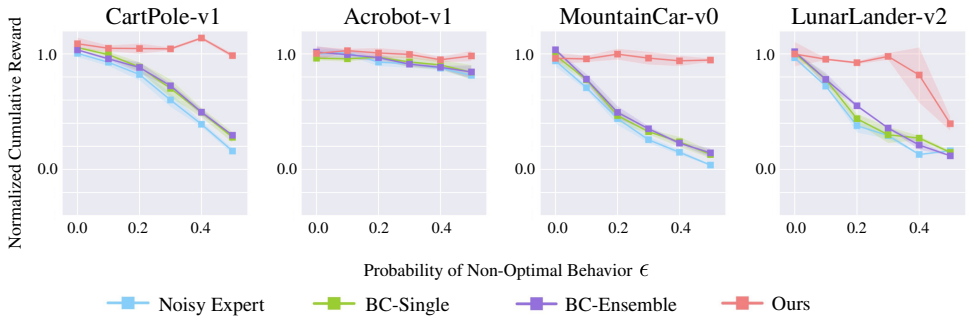

Figure 4: The performance of policies vs. ϵ given 50000 state-action pairs of the noisy expert demonstrations where the non-optimal policies $\pi \in \Pi \setminus \{\pi_e^*\}$ select actions uniformly at random. BC-Single is a policy learned by BC. BC-Ensemble is an ensemble of policies, each of which was learned by BC. Shaded regions indicate the standard deviation over five experiments.

positive value and is not associated with θ, we obtain an inequality : $\mathbb{E}_{s\sim d^{\pi_e}, a\sim\pi_e(\cdot|s)}[\log \pi_\theta(a|s)] \leq -\mathbb{E}_{\sim d^{\pi_e}}[D_{KL}(\pi_e(\cdot|s)||\pi_\theta(\cdot|s))]$. The same goes with (15) and (16) as:

$$\alpha\mathbb{E}_{s\sim d_e^{\pi_e}, a\sim\pi_e(\cdot|s)}[\log \pi_\theta(a|s)] \leq -\alpha\mathbb{E}_{\sim d^{\pi_e}}[D_{KL}(\pi_e(\cdot|s)||\pi_\theta(\cdot|s))] \tag{18}$$

$$\beta\mathbb{E}_{s\sim d_*^{\pi_e}, a\sim\pi_e^*(\cdot|s)}[\log \pi_\theta(a|s)] \leq -\beta\mathbb{E}_{\sim d_*^{\pi_e}}[D_{KL}(\pi_e^*(\cdot|s)||\pi_\theta(\cdot|s))] \tag{19}$$

Since π_e adopts both π_e^* and $\pi \in \Pi \setminus \{\pi_e^*\}$ following the probability ϵ, the third term (17) can be expanded as:

$$\begin{aligned}\gamma\mathbb{E}_{s\sim d_{e+*}^{\pi_e}, a\sim\pi_e(\cdot|s)}[\log \pi_\theta(a|s)] &= \gamma\mathbb{E}_{s\sim d_{e+*}^{\pi_e}}\big\{\epsilon\mathbb{E}_{\pi\sim p_\Pi, a\sim\pi(\cdot|s)}[\log \pi_\theta(a|s)] \\ &\quad +(1-\epsilon)\mathbb{E}_{a\sim\pi_e^*(\cdot|s)}[\log \pi_\theta(a|s)]\big\} \\ &\leq -\gamma\big\{\epsilon\mathbb{E}_{s\sim d_{e+*}^{\pi_e}, \pi\sim p_\Pi}[D_{KL}(\pi(\cdot|s)||\pi_\theta(\cdot|s))] \\ &\quad +(1-\epsilon)\mathbb{E}_{s\sim d_{e+*}^{\pi_e}}[D_{KL}(\pi_e^*(\cdot|s)||\pi_\theta(\cdot|s))]\big\}\end{aligned} \tag{20}$$

A.3 Detailed Description of the Experimental Setup

We annotate confidence scores for the noisy demonstrations so that the confidence is one if the demonstrations are obtained with $\epsilon = 0$ otherwise zero. The confidence scores are used IC-GAIL (Wu et al., 2019) as well as 2IWIL (Wu et al., 2019). We use publicly available code (`https://github.com/kristery/Imitation-Learning-from-Imperfect-Demonstration`) for the implementation of both IC-GAIL and 2IWIL. We follow the training procedure of both methods as described in Section 5 in (Wu et al., 2019). We annotate rankings for the noisy demonstrations so that the smaller ϵ. Then, we train the learner by T-REX (Wu et al., 2019) given the ranked demonstration data. We use publicly available code (`https://github.com/hiwonjoon/ICML2019-TREX`) for the implementation of T-REX.

A.4 Experimental Results on Discrete Control Tasks

Figure 4 shows the experimental results on four discrete control tasks. Over all tasks, our algorithm obtain much better results than BC.

A.5 Implementation Details of Our Algorithm

We implement our algorithm using K neural networks with two hidden layers to represent policies $\pi_{\theta^1}, \pi_{\theta^2}, ..., \pi_{\theta^K}$ in the ensemble. The input of the networks is vector representations of the state. Each neural network has 100 hidden units in each hidden layer followed by hyperbolic tangent nonlinearity, and the dimensionality of its final output corresponds to that of action space. The final output is followed by softmax function in the discrete control tasks. As for the continuous control

tasks, the final output represents the mean of a Gaussian policy as $\pi_{\theta^k} = \mathcal{N}(\mu_{\theta^k}(s), \sigma^2_{\theta^k})$, where $\sigma^2_{\theta^k}$ is implemented as a trainable independent vector from the networks. The neural network architecture for the policy trained by BC is the same as the ones for a single policy in our algorithm. We set the scaling parameter $\eta = K/\sum_{k=1}^{K} \pi_{\theta^k_{old}}(\mu_{\theta^k_{old}}(s)|s)$. We employ Adam (Kingma & Ba, 2014) for learning parameters with a learning rate of 10^{-4}. The parameters in all layers are initialized by Xavier initialization (Glorot & Bengio, 2010). The mini-batch size and the number of training epochs are 128 and 500, respectively.

A.6 Ablation Experiments

We conducted ablation experiments where we evaluate how the number of policies K in the ensemble policy π_θ as well as the number of the policies K_{old} used in the *old* ensemble policies $\pi_{\theta_{old}}$ affect the performance. Figure 2 summarizes the ablation experimental results. Even if our algorithm uses $K = 1$ as BC-Single does, the results fo our algorithm are better than BC. It indicates that the weighted action sampling described in 5.2 works to reduce the influence of noisy regularizer without relying on the ensemble approach. The same goes with $K = 5$ – our algorithm with $K = 5$ and $K_{old} = 1$ obtain much better performance than BC-Ensemble with $K = 5$. This result also supports the weighted action sampling works. The learner performance with fixed K increases as K_{old} increases. Similarly, the learner performance with fixed K_{old} increases as K increases. It suggests that both K and K_{old} affect the performance in our algorithm.

Table 2: The performance of policies on the ablation experiment. The number of state-action pairs of the noisy expert demonstrations is $N = 50000$. The non-optimal policies $\pi \in \Pi \backslash \{\pi^*_e\}$ is $\mathcal{U}(0, I)$. BC-Single is a policy learned by BC. BC-Ensemble is an ensemble of five policies, each of which was learned by BC. K denotes the number of policies in the ensemble policy π_θ. K_{old} denotes the number of policies used in the old ensemble policy $\pi_{\theta_{old}}$. The mean and standard deviation of the normalized cumulative rewards over three experiments are described.

	Ant-v2	HalfCheetah-v2	Hopper-v2	Walker2d-v2	Average
BC-Single	0.149 ± 0.001	0.305 ± 0.006	0.258 ± 0.017	0.071 ± 0.004	0.196 ± 0.105
BC-Ensemble($K = 5$)	0.664 ± 0.043	0.459 ± 0.014	0.352 ± 0.028	0.279 ± 0.039	0.438 ± 0.167
Ours($K = 1, K_{old} = 1$)	0.272 ± 0.184	0.505 ± 0.279	0.405 ± 0.206	0.281 ± 0.306	0.366 ± 0.111
Ours($K = 5, K_{old} = 1$)	0.903 ± 0.048	1.057 ± 0.008	0.602 ± 0.130	0.345 ± 0.049	0.731 ± 0.320
Ours($K = 1, K_{old} = 5$)	0.517 ± 0.015	0.907 ± 0.007	0.778 ± 0.093	0.414 ± 0.085	0.654 ± 0.227
Ours($K = 5, K_{old} = 5$)	0.995 ± 0.053	1.058 ± 0.053	0.573 ± 0.079	0.364 ± 0.044	0.747 ± 0.334