Diffusion Model-Augmented Behavioral Cloning

Anonymous authors
Paper under double-blind review

Abstract

Imitation learning addresses the challenge of learning by observing an expert's demonstrations without access to reward signals from environments. Most existing imitation learning methods that do not require interacting with environments either model the expert distribution as the conditional probability $p(a|s)$ (*e.g.*, behavioral cloning, BC) or the joint probability $p(s, a)$. Despite the simplicity of modeling the conditional probability with BC, it usually struggles with generalization. While modeling the joint probability can improve generalization performance, the inference procedure is often time-consuming, and the model can suffer from manifold overfitting. This work proposes an imitation learning framework that benefits from modeling both the conditional and joint probability of the expert distribution. Our proposed diffusion model-augmented behavioral cloning (DBC) employs a diffusion model trained to model expert behaviors and learns a policy to optimize both the BC loss (conditional) and our proposed diffusion model loss (joint). DBC outperforms baselines in various continuous control tasks in navigation, robot arm manipulation, dexterous manipulation, and locomotion. We design additional experiments to verify the limitations of modeling either the conditional probability or the joint probability of the expert distribution, as well as compare different generative models. Ablation studies justify the effectiveness of our design choices.

1 Introduction

Recently, the success of deep reinforcement learning (DRL) (Mnih et al., 2015; Lillicrap et al., 2016; Arulkumaran et al., 2017) has inspired the research community to develop DRL frameworks to control robots, aiming to automate the process of designing sensing, planning, and control algorithms by letting the robot learn in an end-to-end fashion. Yet, acquiring complex skills through trial and error can still lead to undesired behaviors even with sophisticated reward design (Christiano et al., 2017; Leike et al., 2018; Lee et al., 2019). Moreover, the exploring process could damage expensive robotic platforms or even be dangerous to humans (Garcıa & Fernández, 2015; Levine et al., 2020).

To overcome this issue, imitation learning (*i.e.*, learning from demonstration) (Schaal, 1997; Osa et al., 2018) has received growing attention, whose aim is to learn a policy from expert demonstrations, which are often more accessible than appropriate reward functions for reinforcement learning. Among various imitation learning directions, adversarial imitation learning (Ho & Ermon, 2016; Zolna et al., 2021; Kostrikov et al., 2019) and inverse reinforcement learning (Ng & Russell, 2000; Abbeel & Ng, 2004) have achieved encouraging results in a variety of domains. Yet, these methods require interacting with environments, which can still be expensive or even dangerous.

On the other hand, behavioral cloning (BC) (Pomerleau, 1989; Bain & Sammut, 1995) does not require interacting with environments. BC formulates imitation learning as a supervised learning problem — given an expert demonstration dataset, an agent policy takes states sampled from the dataset as input and learns to replicate the corresponding expert actions. One can view a BC policy as a discriminative model $p(a|s)$ that models the *conditional probability* of actions a given a state s. Due to its simplicity and training stability, BC has been widely adopted for various applications. However, BC struggles at generalizing to states unobserved during training (Nguyen et al., 2023).

To alleviate the generalization issue, we propose to augment BC by modeling the *joint probability* $p(s, a)$ of expert state-action pairs with a generative model (*e.g.*, diffusion models). This approach is motivated

by Bishop & Nasrabadi (2006) and Fisch et al. (2013), who illustrate that modeling joint probability allows for better generalizing to data points unobserved during training. However, with a learned joint probability model $p(s, a)$, retrieving a desired action a requires actions sampling and optimization (*i.e.*, $\arg\max_{a \in \mathcal{A}} p(s, a)$), which can be extremely inefficient with a large action space. Moreover, modeling joint probabilities can suffer from manifold overfitting (Wu et al., 2021; Loaiza-Ganem et al., 2022) when observed high-dimensional data lies on a low-dimensional manifold (*e.g.*, state-action pairs collected from a script expert policies).

This work proposes an imitation learning framework that combines both the efficiency and stability of modeling the *conditional probability* and the generalization ability of modeling the *joint probability*. Specifically, we propose to model the expert state-action pairs using a state-of-the-art generative model, a diffusion model, which learns to estimate how likely a state-action pair is sampled from the expert dataset. Then, we train a policy to optimize both the BC objective and the learning signals the trained diffusion model produces. Therefore, our proposed framework not only can efficiently predict actions given states via capturing the *conditional probability* $p(a|s)$ but also enjoys the generalization ability induced by modeling the *joint probability* $p(s, a)$ and utilizing it to guide policy learning.

We evaluate our proposed framework and baselines in various continuous control domains, including navigation, robot arm manipulation, and locomotion. The experimental results show that the proposed framework outperforms all the baselines or achieves competitive performance on all tasks. Extensive ablation studies compare our proposed method to its variants, justifying our design choices, such as different generative models, and investigating the effect of hyperparameters.

2 Related Work

Imitation learning addresses the challenge of learning by observing expert demonstrations without access to reward signals from environments. It has various applications such as robotics (Schaal, 1997; Zhao et al., 2023), autonomous driving (Ly & Akhloufi, 2020), and game AI (Harmer et al., 2018).

Behavioral Cloning (BC). BC (Pomerleau, 1989; Torabi et al., 2018) formulates imitating an expert as a supervised learning problem. Due to its simplicity and effectiveness, it has been widely adopted in various domains. Yet, it often struggles at generalizing to states unobserved from the expert demonstrations (Ross et al., 2011; Florence et al., 2022). Some approaches prevent the agent from navigating to states that deviate from the expert demonstrations by mitigating compounding error (Ross et al., 2011; Zhao et al., 2023). However, the problem of generalization still exists even when the compounding errors are alleviated. In this work, we improve the generalization ability of policies by augmenting BC with a diffusion model that learns to capture the joint probability of expert state-action pairs.

Adversarial Imitation Learning (AIL). AIL methods aim to match the state-action distributions of an agent and an expert via adversarial training. Generative adversarial imitation learning (GAIL) (Ho & Ermon, 2016) and its extensions (Torabi et al., 2019; Kostrikov et al., 2019; Zolna et al., 2021; Jena et al., 2021) resemble the idea of generative adversarial networks (Goodfellow et al., 2014), which trains a generator policy to imitate expert behaviors and a discriminator to distinguish between the expert and the learner's state-action pair distributions. While modeling state-action distributions often leads to satisfactory performance, adversarial learning can be unstable and inefficient (Chen et al., 2020). Moreover, even though scholars like Jena et al. (2021) propose to improve the efficiency of GAIL with the BC loss, they still require online interaction with environments, which can be costly or even dangerous. In contrast, our work does not require interacting with environments.

Inverse Reinforcement Learning (IRL). IRL methods (Ng & Russell, 2000; Abbeel & Ng, 2004; Fu et al., 2018; Lee et al., 2021) are designed to infer the reward function that underlies the expert demonstrations and then learn a policy using the inferred reward function. This allows for learning tasks whose reward functions are difficult to specify manually. However, due to its double-loop learning procedure, IRL methods are typically computationally expensive and time-consuming. Additionally, obtaining accurate estimates of the expert's reward function can be difficult, especially when the expert's behavior is non-deterministic or when the expert's demonstrations are sub-optimal.

Diffusion Policies. Recently, Pearce et al. (2023); Chi et al. (2023); Reuss et al. (2023) propose to represent and learn an imitation learning policy using a conditional diffusion model, which produces a predicted action conditioning on a state and a sampled noise vector. These methods achieve encouraging results in modeling stochastic and multimodal behaviors from human experts or play data. In contrast, instead of representing a policy using a diffusion model, our work employs a diffusion model trained on expert demonstrations to guide a policy as a learning objective.

3 Preliminaries

3.1 Imitation Learning

In contrast to reinforcement learning, whose goal is to learn a policy π based on rewards received while interacting with the environment, imitation learning methods aim to learn the policy from an expert demonstration dataset containing M trajectories, $D = \{\tau_1, ..., \tau_M\}$, where τ_i represents a sequence of n_i state-action pairs $\{s_1^i, a_1^i, ..., s_{n_i}^i, a_{n_i}^i\}$.

3.1.1 Modeling Conditional Probability $p(a|s)$

To learn a policy π, behavioral cloning (BC) directly estimates the expert policy π^E with maximum likelihood estimation (MLE). Given a state-action pair (s, a) sampled from the dataset D, BC optimizes $\max_\theta \sum_{(s,a)\in D} \log(\pi_\theta(a|s))$, where θ denotes the parameters of the policy π. One can view a BC policy as a discriminative model $p(a|s)$, capturing the *conditional probability* of an action a given a state s. On the other hand, Implicit BC (Florence et al., 2022; Ganapathi et al., 2022) propose to model the conditional probability with InfoNCE-style (Oord et al., 2018) optimization. Despite their success in various applications, BC-based methods tend to overfit and struggle at generalizing to states unseen during training (Ross et al., 2011; Codevilla et al., 2019; Wang et al., 2022).

3.1.2 Modeling Joint Probability $p(s, a)$

Explicit generative models, such as energy-based models (Du & Mordatch, 2019; Song & Kingma, 2021), variational autoencoder (Kingma & Welling, 2014), and flow-based models (Rezende & Mohamed, 2015; Dinh et al., 2017), aim to model the *joint probability* $p(s, a)$ of the expert dataset, and therefore yield improved generalization performance, as illustrated in Bishop & Nasrabadi (2006); Fisch et al. (2013).

However, these methods can be extremely inefficient in retrieving actions with a large action space during inference since sampling and optimizing actions (*i.e.*, $\arg\max_{a\in\mathcal{A}}\ p(s, a)$) are required. Moreover, they are known to struggle with modeling observed high-dimensional data that lies on a low-dimensional manifold (*i.e.*, manifold overfitting) (Wu et al., 2021; Loaiza-Ganem et al., 2022). As a result, these methods often perform poorly when learning from demonstrations produced by script policies or PID controllers, as discussed in Section 5.4.

We aim to develop an imitation learning framework that enjoys the advantages of modeling the *conditional probability* $p(a|s)$ and the *joint probability* $p(s, a)$. Specifically, we propose to model the *joint probability* of expert state-action pairs using an explicit generative model ϕ, which learns to produce an estimate indicating how likely a state-action pair is sampled from the expert dataset. Then, we train a policy to model the *conditional probability* $p(a|s)$ by optimizing the BC objective and the estimate produced by the learned generative model ϕ. Hence, our method can efficiently predict actions given states, generalize better to unseen states, and suffer less from manifold overfitting.

3.2 Diffusion Models

As described in the previous sections, this work aims to combine the advantages of modeling the *conditional probability* $p(a|s)$ and the *joint probability* $p(s, a)$. Hence, we leverage diffusion models to model the *joint probability* of expert state-action pairs. The diffusion model is a recently developed class of generative models

and has achieved state-of-the-art performance on various tasks (Sohl-Dickstein et al., 2015; Nichol & Dhariwal, 2021; Dhariwal & Nichol, 2021; Ko et al., 2023).

In this work, we utilize Denoising Diffusion Probabilistic Models (DDPMs) (J Ho, 2020) to model expert state-action pairs. Specifically, DDPM models gradually add noise to data samples (*i.e.*, concatenated state-action pairs) until they become isotropic Gaussian (*forward diffusion process*), and then learn to denoise each step and restore the original data samples (*reverse diffusion process*), as illustrated in Figure 1. In other words, DDPM learns to recognize a data distribution by learning to denoise noisy sampled data. More discussion on the relationship between diffusion models and the data distribution can be found in Section I.

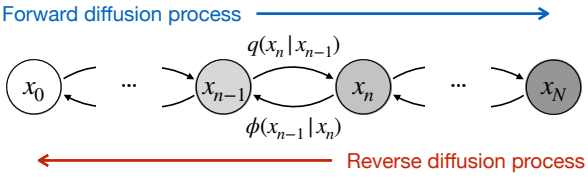


Figure 1: **Denoising Diffusion Probabilistic Model (DDPM).** Latent variables $x_1, ..., x_N$ are produced from the data point x_0 via the forward diffusion process, *i.e.*, gradually adding noises to the latent variables. The diffusion model ϕ learns to reverse the diffusion process by denoising the noisy data to reconstruct the original data point x_0.

4 Approach

Our goal is to design an imitation learning framework that enjoys both the advantages of modeling the *conditional probability* and the *joint probability* of expert behaviors. To this end, we first adopt behavioral cloning (BC) for modeling the *conditional probability* from expert state-action pairs, as described in Section 4.1. To capture the *joint probability* of expert state-action pairs, we employ a diffusion model that learns to produce an estimate indicating how likely a state-action pair is sampled from the expert state-action pair distribution, as presented in Section 4.2.1. Then, we propose to guide the policy learning by optimizing this estimate provided by a learned diffusion model, encouraging the policy to produce actions similar to expert actions, as discussed in Section 4.2.2. Finally, in Section 4.3, we introduce the framework that combines the BC loss and our proposed diffusion model loss, allowing for learning a policy that benefits from modeling both the *conditional probability* and the *joint probability* of expert behaviors. An overview of our proposed framework is illustrated in Figure 2, and the algorithm is detailed in Section 4.4.

4.1 Behavioral Cloning Loss

The behavioral cloning (BC) model aims to imitate expert behaviors with supervision learning. BC learns to capture the conditional probability $p(a|s)$ of expert state-action pairs. A BC policy $\pi(a|s)$ learns by optimizing

$$\mathcal{L}_{\text{BC}} = \mathbb{E}_{(s,a)\sim D, \hat{a}\sim\pi(s)}[d(a, \hat{a})], \tag{1}$$

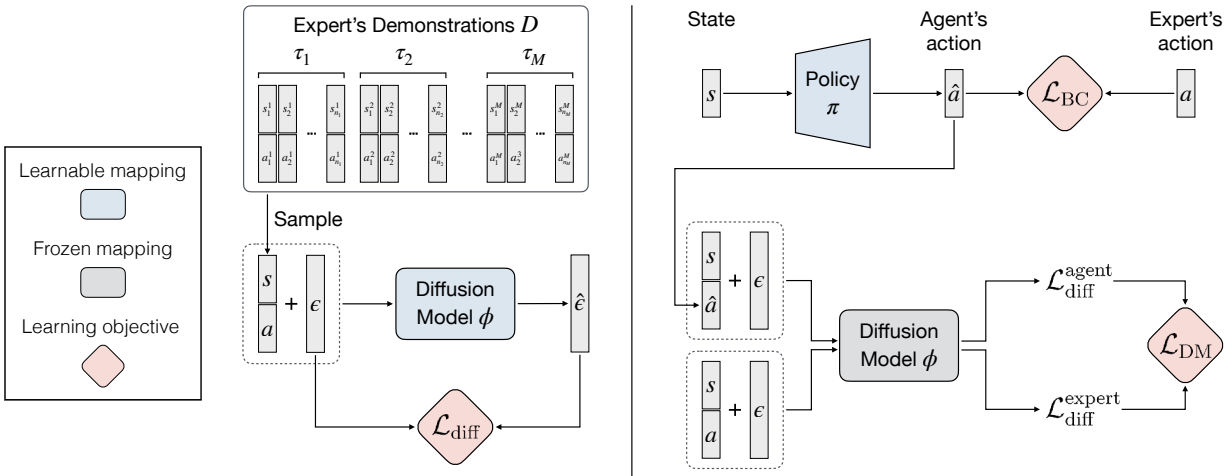
where $d(\cdot, \cdot)$ denotes a distance measure between a pair of actions. For example, we can adopt the mean-square error (MSE) loss $||a - \hat{a}||^2$ for most continuous control tasks.

4.2 Learning a Diffusion Model and Guiding Policy Learning

Instead of directly learning the conditional probability $p(a|s)$, this section discusses how to model the joint probability $p(s, a)$ of expert behaviors with a diffusion model in Section 4.2.1 and presents how to leverage the learned diffusion model to guide policy learning in Section 4.2.2.

4.2.1 Learning a Diffusion Model

We propose to model the joint probability of expert state-action pairs with a diffusion model ϕ. Specifically, we create a joint distribution by simply concatenating a state vector s and an action vector a from a state-action pair (s, a). To model such distribution by learning a denoising diffusion probabilistic model (DDPM) (J Ho, 2020), we inject noise $\epsilon(n)$ into sampled state-action pairs, where n indicates the number of steps of the Markov procedure, which can be viewed as a variable of the level of noise, and the total number of steps is



(a) Learning a Diffusion Model

(b) Learning a Policy with the Learned Diffusion Model

Figure 2: **Diffusion Model-Augmented Behavioral Cloning.** Our proposed method DBC augments behavioral cloning (BC) by employing a diffusion model. (a) **Learning a Diffusion Model**: the diffusion model ϕ learns to model the distribution of concatenated state-action pairs sampled from the demonstration dataset D. It learns to reverse the diffusion process (*i.e.*, denoise) by optimizing $\mathcal{L}_{\text{diff}}$ in Eq. 2. (b) **Learning a Policy with the Learned Diffusion Model**: we propose a diffusion model objective $\mathcal{L}_{\text{DM}}$ for policy learning and jointly optimize it with the BC objective $\mathcal{L}_{\text{BC}}$. Specifically, $\mathcal{L}_{\text{DM}}$ is computed based on processing a sampled state-action pair (s, a) and a state-action pair $(s, \hat{a})$ with the action $\hat{a}$ predicted by the policy π with $\mathcal{L}_{\text{diff}}$.

notated as N. Then, we train the diffusion model ϕ to predict the injected noises by optimizing

$$\mathcal{L}_{\text{diff}}(s, a, \phi) = \mathbb{E}_{n\sim N,(s,a)\sim D}\left[||\hat{\epsilon}(s, a, n) - \epsilon(n)||^2\right] = \mathbb{E}_{n\sim N,(s,a)\sim D}\left[||\phi(s, a, \epsilon(n)) - \epsilon(n)||^2\right], \tag{2}$$

where $\hat{\epsilon}$ is the noise predicted by the diffusion model ϕ. Once optimized, the diffusion model can *recognize* the expert distribution by perfectly predicting the noise injected into state-action pairs sampled from the expert distribution. On the other hand, predicting the noise injected into state-action pairs sampled from any other distribution should yield a higher loss value. Therefore, we propose to view $\mathcal{L}_{\text{diff}}(s, a, \phi)$ as an estimate of how well the state-action pair (s, a) fits the expert distribution that ϕ learns from and serve this estimate as a learning signal for the policy learning.

4.2.2 Learning a Policy with Diffusion Model Loss

A diffusion model ϕ trained on an expert dataset can produce an estimate $\mathcal{L}_{\text{diff}}(s, a, \phi)$ indicating how well a state-action pair (s, a) fits the expert distribution. We propose to leverage this signal to guide a policy π predicting actions $\hat{a}$ to imitate the expert. Specifically, the policy π learns by optimizing

$$\mathcal{L}_{\text{diff}}^{\text{agent}} = \mathcal{L}_{\text{diff}}(s, \hat{a}, \phi) = \mathbb{E}_{s\sim D,\hat{a}\sim\pi(s)}\left[||\hat{\epsilon}(s, \hat{a}, n) - \epsilon||^2\right]. \tag{3}$$

Intuitively, the policy π learns to predict actions $\hat{a}$ that are indistinguishable from the expert actions a for the diffusion model conditioning on the same set of states.

We hypothesize that learning a policy to optimize Eq. 3 can be unstable, especially for state-action pairs that are not well-modeled by the diffusion model, which yield a high value of $\mathcal{L}_{\text{diff}}$ even with expert state-action pairs. Therefore, we propose to normalize the agent diffusion loss $\mathcal{L}_{\text{diff}}^{\text{agent}}$ with an expert diffusion loss $\mathcal{L}_{\text{diff}}^{\text{expert}}$, which can be computed with expert state-action pairs (s, a) as follows:

$$\mathcal{L}_{\text{diff}}^{\text{expert}} = \mathcal{L}_{\text{diff}}(s, a, \phi) = \mathbb{E}_{(s,a)\sim D}\left[||\hat{\epsilon}(s, a, n) - \epsilon||^2\right]. \tag{4}$$

We propose to optimize the diffusion model loss $\mathcal{L}_{\text{DM}}$ for the policy based on calculating the difference between the above agent and expert diffusion losses:

$$\mathcal{L}_{\text{DM}} = \mathbb{E}_{(s,a)\sim D, \hat{a}\sim\pi(s)} \left[max \left(\mathcal{L}_{\text{diff}}^{\text{agent}} - \mathcal{L}_{\text{diff}}^{\text{expert}}, 0 \right) \right]. \tag{5}$$

4.3 Combining the Two Objectives

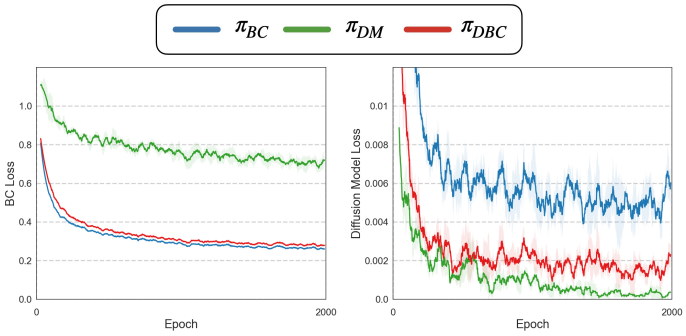
Our goal is to learn a policy that benefits from both modeling the conditional probability and the joint probability of expert behaviors. To this end, we propose to augment a BC policy, which optimizes the BC loss L_{BC} in Eq. 1, by combining L_{BC} with the proposed diffusion model loss L_{DM} in Eq. 5. By optimizing them together, we encourage the policy to predict actions that fit the expert joint probability captured by diffusion models. To learn from both the BC loss and the diffusion model loss, we train the policy to optimize

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{BC}} + \lambda \mathcal{L}_{\text{DM}}, \tag{6}$$

where λ is a coefficient that determines the importance of the diffusion model loss relative to the BC loss.

We notice that when the learned policy is optimal, *i.e.*, $\pi = \pi^E$, both objectives converge to 0 despite that $\mathcal{L}_{\text{BC}}$ models the conditional probability while $\mathcal{L}_{\text{DM}}$ models the joint probability. Therefore, we examine the roles of the above two objectives during the training procedure.

As Figure 3 shown, we train three policies on Maze environment with L_{BC}, L_{DM}, and both objectives. The derived policies are referred to π_{BC}, π_{DM}, and π_{DBC}, respectively. We observe that while optimizing one loss can also reduce the other loss to some extent (π_{BC} and π_{DM}), optimizing the combination leads to favorable convergence for both objectives (π_{DBC}). Therefore, considering both objectives, that is, considering both the conditional and the joint probability, is beneficial for policy learning and leads the learned policy π closer to the optimal one π^E. The above observation is also supported by the quantitative results presented in both Table 1 and Table 3 in the subsequent section. Further discussions on combining these two losses can be found in Section A.



(a) Training Progress of $\mathcal{L}_{\text{BC}}$ (b) Training Progress of $\mathcal{L}_{\text{DM}}$

Figure 3: **Compatibility of $\mathcal{L}_{\text{BC}}$ and $\mathcal{L}_{\text{DM}}$.** We report the training progress of three policies π_{BC}, π_{DM}, and π_{DBC} that are updated with $\mathcal{L}_{\text{BC}}$, $\mathcal{L}_{\text{DM}}$, and both objectives, respectively. Our proposed method can effectively optimize both $\mathcal{L}_{\text{BC}}$ and $\mathcal{L}_{\text{DM}}$, demonstrating the compatibility of the two losses.

4.4 Algorithm

Our proposed framework DBC is detailed in Algorithm 1. The algorithm consists of two parts. (1) **Learning a diffusion model**: The diffusion model ϕ learns to model the distribution of concatenated state-action pairs sampled from the demonstration dataset D. It learns to reverse the diffusion process (*i.e.*, denoise) by optimizing $\mathcal{L}_{\text{diff}}$. (2) **Learning a policy with the learned diffusion model**: We propose a diffusion model objective $\mathcal{L}_{\text{DM}}$ for policy learning and jointly optimize it with the BC objective $\mathcal{L}_{\text{BC}}$. Specifically, $\mathcal{L}_{\text{DM}}$ is computed based on processing a sampled state-action pair (s, a) and a state-action pair $(s, \hat{a})$ with the action $\hat{a}$ predicted by the policy π with $\mathcal{L}_{\text{diff}}$.

Algorithm 1 Diffusion Model-Augmented Behavioral Cloning (DBC)

Input: Expert's Demonstration Dataset D
Output: Policy π.

// Learning a diffusion model ϕ
Randomly initialize a diffusion model ϕ
for each diffusion model iteration **do**
  Sample (s, a) from D
  Sample noise level n from $\{0, ..., N\}$
  Update ϕ using L_{diff} from Eq. 2
end for
// Learning a policy π with the learned diffusion model ϕ
Randomly initialize a policy π
for each policy iteration **do**
  Sample (s, a) from D
  Predict an action $\hat{a}$ using π from s: $\hat{a} \sim \pi(s)$
  Compute the BC loss L_{BC} using Eq. 1
  Sample noise level n from $\{0, ..., N\}$
  Compute the agent diffusion loss $L_{\text{diff}}^{\text{agent}}$ with $(s, \hat{a})$ using Eq. 3
  Compute the expert diffusion loss $L_{\text{diff}}^{\text{expert}}$ with (s, a) using Eq. 4
  Compute the diffusion model loss L_{DM} using Eq. 5
  Update π using the total loss L_{total} from Eq. 6
end for
return π

5 Experiments

We design experiments in various continuous control domains, including navigation, robot arm manipulation, dexterous manipulation, and locomotion, to compare our proposed framework (DBC) to its variants and baselines.

5.1 Experimental Setup

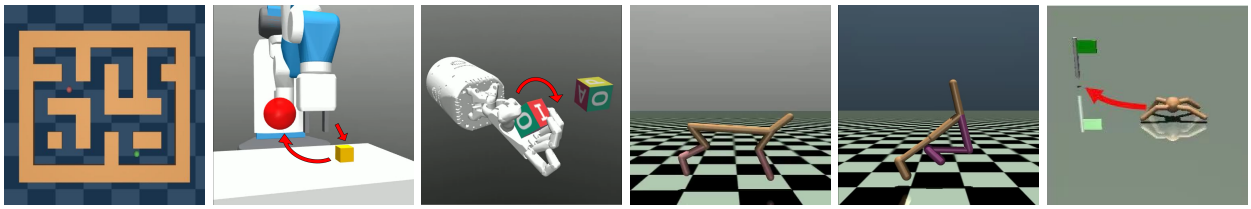
This section describes the environments, tasks, and expert demonstrations used for learning and evaluation. More details can be found in Section D.

Navigation. To evaluate our method on a navigation task, we choose Maze, a maze environment proposed in Fu et al. (2020) (maze2d-medium-v2), as illustrated in Figure 4a. This task features a point-mass agent in a 2D maze learning to navigate from its start location to a goal location by iteratively predicting its x and y acceleration. The agent's beginning and final locations are chosen randomly. We collect 100 demonstrations with 18,525 transitions using a controller.

Robot Arm Manipulation. We evaluate our method in FetchPick, a robot arm manipulation domain with a 7-DoF Fetch task, as illustrated in Figure 4b. FetchPick requires picking up an object from the table and lifting it to a target location. We use the demonstrations, consisting of 10k transitions (303 trajectories), provided by Lee et al. (2021) for these tasks.

Dexterous Manipulation. In HandRotate, we further evaluate our method on a challenging environment proposed in Plappert et al. (2018), where a 24-DoF Shadow Dexterous Hand learns to in-hand rotate a block to a target orientation, as illustrated in Figure 4c. This environment has a state space (68D) and action space (20D), which is high dimensional compared to the commonly-used environments in IL. We collected 10k transitions (515 trajectories) from a SAC (Haarnoja et al., 2018) expert policy trained for 10M environment steps.

Locomotion. For locomotion, we leverage the Cheetah and Walker (Brockman et al., 2016) environments. Both Cheetah and Walker require a bipedal agent (with different structures) to travel as fast as possible



(a) Maze (b) FetchPick (c) HandRotate (d) Cheetah (e) Walker (f) AntReach

Figure 4: **Environments & Tasks. (a) Maze**: A point-mass agent (green) in a 2D maze learns to navigate from its start location to a goal location (red). **(b) FetchPick**: The robot arm manipulation tasks employ a 7-DoF Fetch robotics arm to pick up an object (yellow cube) from the table and move it to a target location (red). **(c) HandRotate**: This dexterous manipulation task requires a Shadow Dexterous Hand to in-hand rotate a block to a target orientation. **(d)-(e) Cheetah and Walker**: These locomotion tasks require learning agents to walk as fast as possible while maintaining their balance. **(f) AntReach**: This task combines locomotion and navigation, instructing an ant robot with four legs to reach a goal location while maintaining balance.

while maintaining its balance, as illustrated in Figure 4d and Figure 4e, respectively. We use the demonstrations provided by Kostrikov (2018), which contains 5 trajectories with 5k state-action pairs for both the Cheetah and Walker environments.

Locomotion + Navigation. We further explore our method on the challenging AntReach environment. In the environment, the quadruped ant aims to reach a randomly generated target located along the boundary of a semicircle centered around the ant, as illustrated in Figure 4f. AntReach environment combines the properties of locomotion and goal-directed navigation tasks, which require robot controlling and path planning to reach the goal. We use the demonstrations provided by Lee et al. (2021), which contains 500 trajectories with 25k state-action pairs in AntReach.

5.2 Baselines

This work focuses on imitation learning problem *without* environment interactions. Therefore, approaches that require environmental interactions, such as GAIL-based methods, are not applicable. Instead, we extensively compared our proposed method to state-of-the-art imitation learning methods that do not require interaction with the environment, including Implicit BC (Florence et al., 2022) and Diffusion Policy (Chi et al., 2023; Reuss et al., 2023).

- **BC** learns to imitate an expert by modeling the conditional probability $p(a|s)$ of the expert behaviors via optimizing the BC loss $\mathcal{L}_{\text{BC}}$ in Eq. 1.
- **Implicit BC (IBC)** (Florence et al., 2022) models expert state-action pairs with an energy-based model. For inference, we implement the derivative-free optimization algorithm proposed in IBC, which samples actions iteratively and select the desired action according to the predicted energies.
- **Diffusion policy** refers to the methods that learn a conditional diffusion model as a policy (Chi et al., 2023; Reuss et al., 2023). Specifically, we implement this baseline based on Pearce et al. (2023). We include this baseline to analyze the effectiveness of using diffusion models as a policy or as a learning objective (ours).

5.3 Experimental Results

We report the experimental results in terms of success rate (Maze, FetchPick, HandRotate, and AntReach), and return (Cheetah and Walker) in Table 1. The details of model architecture can be found in Section E. Training and evaluation details can be found in Section F. Additional analysis and experimental results can be found in Section 5.5 and Section H.

Table 1: **Experimental Result.** We report the mean and the standard deviation of success rate (Maze, FetchPick, HandRotate, AntReach) and return (Cheetah, Walker), evaluated over three random seeds. Our proposed method (DBC) outperforms or performs competitively against the best baseline over all environments.

Method	Maze	FetchPick	HandRotate	Cheetah	Walker	AntReach
BC	92.1% ± 3.6%	91.6% ± 5.8%	57.5% ± 4.7%	**4873.3** ± 69.7	6954.4 ± 73.5	56.2% ± 4.9%
Implicit BC	78.3% ± 6.0%	69.4% ± 7.3%	13.8% ± 3.7%	1563.6 ± 486.8	839.8 ± 104.2	23.7% ± 4.9%
Diffusion Policy	**95.5**% ± 1.9%	83.9% ± 3.4%	**61.7**% ± 4.1%	4650.3 ± 59.9	6479.1 ± 238.6	61.8% ± 4.0%
DBC (Ours)	**95.4**% ± 1.7%	**97.5**% ± 1.9%	**60.1**% ± 4.4%	**4909.5** ± 73.0	**7034.6** ± 33.7	**70.1**% ± 4.9%

Overall Task Performance. In navigation (Maze) and manipulation (FetchPick and HandRotate) tasks, our DBC performs competitively, i.e., within a standard deviation, against the Diffusion Policy and outperforms the other baselines. We hypothesize that these tasks require the agent to learn from demonstrations with various behaviors. Diffusion policy has shown promising performance for capturing multi-modality distribution, while our DBC can also generalize well with the guidance of the diffusion models, so both methods achieve satisfactory results.

In locomotion tasks, i.e., Cheetah and Walker, our DBC outperforms Diffusion Policy and performs competitively against the simple BC baseline. We hypothesize that this is because locomotion tasks with sufficient expert demonstrations and little randomness do not require generalization during inference. The agent can simply follow the closed-loop progress of the expert demonstrations, resulting in both BC and DBC performing similarly to the expert demonstrations. On the other hand, the Diffusion Policy is designed for modeling multimodal behaviors and therefore, performs inferior results on single-mode locomotion tasks. For AntReach task, which combines locomotion and navigation, our method outperforms all the baselines.

In summary, our proposed DBC is able to perform superior results across all tasks, which verifies the effectiveness of combining conditional and joint distribution modeling.

Inference Efficiency. To evaluate the inference efficiency, we measure and report the number of evaluation episodes per second (↑) for Implicit BC (9.92), Diffusion Policy (1.38), and DBC (**30.79**) on an NVIDIA RTX 3080 Ti GPU in Maze. As a results of modeling the conditional probability $p(a|s)$, DBC and BC can directly map states to actions during inference. In contrast, Implicit BC samples and optimizes actions, while Diffusion Policy iteratively denoises sampled noises, which are both time-consuming. This verifies the efficiency of modeling the conditional probability.

Action Space Dimension. The Implicit BC baseline requires time-consuming action sampling and optimization during inference, and such a procedure may not scale well to high-dimensional action spaces. Our Implicit BC baseline with a derivative-free optimizer struggles in Cheetah, Walker, and HandRotate environments, whose action dimensions are 6, 6, and 20, respectively. This is consistent with Florence et al. (2022), which reports that the optimizer failed to solve tasks with an action dimension larger than 5. In contrast, our proposed DBC can handle high-dimensional action spaces.

5.4 Comparing Modeling Conditional Probability and Joint Probability

This section aims to empirically identify the limitations of modeling *either* the conditional *or* the joint probability in an open maze environment implemented with Fu et al. (2020).

Generalization. We aim to investigate if learning from the BC loss alone struggles at generalization (*conditional*) and examine if guiding the policy using the diffusion model loss yields improved generalization ability (*joint*). We collect trajectories of a PPO policy learning to navigate from $(5, 3)$ to goals sampled around $(1, 2)$ and $(1, 4)$ (green) on a 5×5 map, as shown in Figure 5a. Given these expert trajectories, we learn a policy π_{BC} to optimize Eq. 1 and another policy π_{DM} to optimize Eq. 5. Then, we evaluate the two policies by sampling goals around $(1, 1)$, $(1, 3)$, and $(1, 5)$ (red), which are unseen during training and require the ability to generalize. Visualized trajectories of the two policies in Figure 5a show that π_{BC} (orange) fails

(a) (b) (c)

Figure 5: **Comparing Modeling Conditional Probability and Joint Probability. (a) Generalization.** We collect expert trajectories from a PPO policy learning to navigate to goals sampled from the green regions. Then, we learn a policy π_{BC} to optimize $\mathcal{L}_{\text{BC}}$, and another policy π_{DM} to optimize $\mathcal{L}_{\text{DM}}$ with a diffusion model trained on the expert distribution. We evaluate the two policies by sampling goals from the red regions, which requires the ability to generalize. π_{BC} (orange) struggles at generalizing to unseen goals, whereas π_{DM} (blue) can generalize (*i.e.*, extrapolate) to some extent. **(b)-(c) Manifold overfitting.** We collect the green spiral trajectories from a script policy, whose actions are visualized as red crosses. We then train and evaluate π_{BC} and π_{DM}. The trajectories of π_{BC} (orange) can closely follow the expert trajectories (green), while the trajectories of π_{DM} (blue) deviates from expert's. This is because the diffusion model struggles at modeling such expert action distribution with a lower intrinsic dimension, which can be observed from incorrectly predicted actions (blue dots) produced by the diffusion model.

to generalize to unseen goals, whereas π_{DM} (blue) can generalize (*i.e.*, extrapolate) to some extent. This verifies our motivation to augment BC with the diffusion model loss.

Manifold overfitting. We aim to examine if modeling the joint probability is difficult when observed high-dimensional data lies on a low-dimensional manifold (*i.e.*, manifold overfitting). We collect trajectories from a script policy that executes actions $(0.5, 0)$, $(0, 0.5)$, $(-0.7, 0)$, and $(0, -0.7)$ (red crosses in Figure 5b), each for 40 consecutive time steps, resulting the green spiral trajectories visualized in Figure 5c.

Given these expert demonstrations, we learn a policy π_{BC} to optimize Eq. 1, and another policy π_{DM} to optimize Eq. 5 with a diffusion model trained on the expert distribution. Figure 5b shows that the diffusion model struggles at modeling such expert action distribution with a lower intrinsic dimension. As a result, Figure 5c show that the trajectories of π_{DM} (blue) deviates from the expert trajectories (green) as the diffusion model cannot provide effective loss. On the other hand, the trajectories of π_{BC} (orange) are able to closely follow the expert's and result in a superior success rate. This verifies our motivation to complement modeling the joint probability with modeling the conditional probability (*i.e.*, BC).

5.5 Generalization Experiments in FetchPick

This section further investigates the generalization capabilities of the policies learned by our proposed framework and the baselines. To this end, we evaluate the policies by injecting different noise levels to both the initial state and goal location in FetchPick. Specifically, we parameterize the noise by scaling the 2D sampling regions for the block and goal locations in both environments. We expect all the methods to perform worse with higher noise levels, while the performance drop of the methods with better generalization ability is less significant. In this experiment, we set the coefficient λ of DBC to 0.1 in FetchPick. The results are presented in Table 2 for FetchPick.

Overall Performance. Our proposed framework DBC consistently outperforms all the baselines with different noise levels, indicating the superiority of DBC when different levels of generalization are required.

Table 2: **Generalization Experiments in FetchPick**. We report the performance of our proposed framework DBC and the baselines regarding the mean and the standard deviation of the success rate with different levels of noise injected into the initial state and goal locations in FetchPick, evaluated over three random seeds.

Method	Noise Level				
	1	1.25	1.5	1.75	2
BC	92.4% ± 8.5%	91.6% ± 5.8%	85.5% ± 6.3%	77.6% ± 7.1%	**67.4**% ± 8.2%
Implicit BC	83.1% ± 3.1%	69.4% ± 7.3%	51.6% ± 4.2%	36.5% ± 4.7%	23.6% ± 3.0%
Diffusion Policy	90.0% ± 3.5%	83.9% ± 3.4%	72.3% ± 6.8%	64.1% ± 7.1%	58.2% ± 8.2%
DBC (Ours)	**99.5**% ± 0.5%	**97.5**% ± 1.9%	**91.5**% ± 3.3%	**83.3**% ± 4.8%	**73.5**% ± 6.8%

Performance Drop with Increased Noise Level. In FetchPick, DBC experiences a performance drop of 26.1% when the noise level increase from 1 to 2. However, BC and Implicit BC demonstrate a performance drop of 27.0% and 71.6%, respectively. Notably, Diffusion Policy initially performs poorly at a noise level of 1 but demonstrates its robustness with a performance drop of only 35.3% when the noise level increases to 2. This demonstrates that our proposed framework not only generalizes better but also exhibits greater robustness to noise compared to the baselines.

5.6 Comparing Different Generative Models

Our proposed framework employs a diffusion model (DM) to model the joint probability of expert state-action pairs and utilizes it to guide policy learning. To justify our choice, we explore using other popular generative models to replace the diffusion model in Maze. We consider energy-based models (EBMs) (Du & Mordatch, 2019; Song & Kingma, 2021), variational autoencoder (VAEs) (Kingma & Welling, 2014), and generative adversarial networks (GANs) (Goodfellow et al., 2014). Each generative model learns to model expert state-action pairs. To guide policy learning, given a predicted state-action pair $(s, \hat{a})$ we use the estimated energy of an EBM, the reconstruction error of a VAE, and the discriminator output of a GAN to optimize a policy with or without the BC loss.

Table 3 compares using different generative models to model the expert distribution and guide policy learning. All the generative model-guide policies can be improved by adding the BC loss, justifying our motivation to complement modeling the joint probability with modeling the conditional probability. With or without the BC loss, the diffusion model-guided policy achieves the best performance compared to other generative models. Specifically, DM outperforms the second-best baseline GAN by 24.8% improvement without BC and by 2.4% with BC, which verifies our choice of the generative model. Training details of learning the generative models and how to utilize them to guide policy learning can be found in Section F.4. We also report the results on FetchPick in Section G.1

5.7 Ablation Study

5.7.1 Effect of the Diffusion Model Loss Coefficient λ

We examine the impact of varying the coefficient of the diffusion model loss λ in Eq. 6 in FetchPick. The result presented in Figure 6 shows that $\lambda = 0.5$ yields the best performance of 97.5%. A higher or lower λ leads to worse performance. For instance, when λ is 0 (only BC), the success rate is 91.7%, and the performance drops to 51.46% when λ is 10. This result demonstrates that modeling the conditional probability ($\mathcal{L}_{\text{BC}}$) and the joint probability ($\mathcal{L}_{\text{DM}}$) can complement each other.

5.7.2 Effect of the Normalization Term

We aim to investigate whether normalizing the diffusion model loss $\mathcal{L}_{\text{DM}}$ with the expert diffusion model loss $\mathcal{L}_{\text{diff}}^{\text{expert}}$ yields improved performance. We train a variant of DBC where only $\mathcal{L}_{\text{diff}}^{\text{agent}}$ in Eq. 3 instead of $\mathcal{L}_{\text{DM}}$ in Eq. 5 is used to augment BC. For instance, the unnormalized variant performs worse than DBC in the Maze environment, where the average success rate is 94% and 95%, respectively. This justifies the

Table 3: **Comparing Generative Models in Maze**. We compare using different generative models to model the expert distribution and guide policy learning in MAZE. With or without the BC loss, the diffusion model-guided policy achieves the best performance compared to other generative models.

Method	without BC	with BC
BC	N/A	92.1% ± 3.6%
EBM	20.3% ± 11.8%	92.5% ± 3.0%
VAE	53.1% ± 8.7%	92.7% ± 2.7%
GAN	54.8% ± 4.4%	93.0% ± 3.5%
DM	**79.6**% ± 9.6%	**95.4**% ± 1.7%

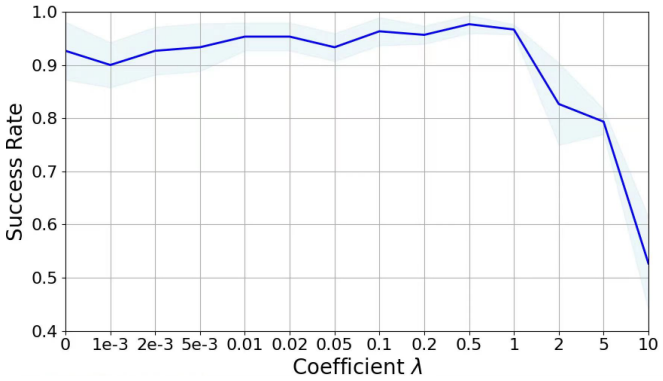


Figure 6: **Effect of Coefficient λ for L_{DM}.** We experiment with different values of λ in FETCHPICK, each evaluated over three random seeds.

effectiveness of the proposed normalization term $\mathcal{L}_{\text{diff}}^{\text{expert}}$ in $\mathcal{L}_{\text{DM}}$. We find consistent results in all of the environments except ANTREACH, and comprehensive results can be found in Table 9 of Section G.2

6 Discussion

We propose an imitation learning framework that benefits from modeling both the conditional probability $p(a|s)$ and the joint probability $p(s, a)$ of the expert distribution. Our proposed diffusion model-augmented behavioral cloning (DBC) employs a diffusion model trained to model expert behaviors and learns a policy to optimize both the BC loss and our proposed diffusion model loss. Specifically, the BC loss captures the conditional probability $p(a|s)$ from expert state-action pairs, which directly guides the policy to replicate the expert's action. On the other hand, the diffusion model loss models the joint distribution of expert state-action pairs $p(s, a)$, which provides an evaluation of how well the predicted action aligned with the expert distribution. DBC outperforms baselines or achieves competitive performance in various continuous control tasks in navigation, robot arm manipulation, dexterous manipulation, and locomotion. We design additional experiments to verify the limitations of modeling either the conditional probability or the joint probability of the expert distribution as well as compare different generative models. Ablation studies investigate the effect of hyperparameters and justify the effectiveness of our design choices. In terms of the limitations, our proposed framework, in its current form, is only designed to learn from expert trajectories without interacting with environments and cannot learn from trajectories produced by the learner policy. Extending our method to incorporate agent data can potentially allow for improvement when interacting environments are possible, which is left for future work.

7 Broader Impacts

This work proposes Diffusion Model-Augmented Behavioral Cloning, a novel imitation learning framework that aims to increase the ability of autonomous learning agents (*e.g.*, robots, game AI agents) to acquire skills by imitating demonstrations provided by experts (*e.g.*, humans). However, it is crucial to acknowledge that our proposed framework, by design, inherits any biases exhibited by the expert demonstrators. These biases can manifest as sub-optimal, unsafe, or even discriminatory behaviors. To address this concern, ongoing research endeavors to mitigate bias and promote fairness in machine learning hold promise in alleviating these issues. Moreover, research works that enhance learning agents' ability to imitate experts, such as this work, can pose a threat to job security. Nevertheless, in sum, we firmly believe that our proposed framework can offer tremendous advantages in terms of enhancing the quality of human life and automating laborious, arduous, or perilous tasks that pose risks to humans, which far outweigh the challenges and potential issues.

References

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *International Conference on Machine Learning*, 2004.

Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 2017.

Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, 1995.

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*. Springer, 2006.

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

Minshuo Chen, Yizhou Wang, Tianyi Liu, Zhuoran Yang, Xingguo Li, Zhaoran Wang, and Tuo Zhao. On computation and generalization of generative adversarial imitation learning. In *International Conference on Learning Representations*, 2020.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2023.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, 2017.

Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *International Conference on Computer Vision*, 2019.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Neural Information Processing Systems*, 2021.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *International Conference on Learning Representations*, 2017.

Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. In *Neural Information Processing Systems*, 2019.

Ruili Feng, Deli Zhao, and Zheng-Jun Zha. Understanding noise injection in gans. In *international conference on machine learning*, pp. 3284–3293. PMLR, 2021.

Dominik Fisch, Edgar Kalkowski, and Bernhard Sick. Knowledge fusion for probabilistic generative classifiers with data mining applications. *IEEE Transactions on Knowledge and Data Engineering*, 2013.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on Robot Learning*, 2022.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adverserial inverse reinforcement learning. In *International Conference on Learning Representations*, 2018.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Aditya Ganapathi, Pete Florence, Jake Varley, Kaylee Burns, Ken Goldberg, and Andy Zeng. Implicit kinematic policies: Unifying joint and cartesian action spaces in end-to-end robot learning. In *International Conference on Robotics and Automation*, 2022.

Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 2015.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of International Conference on Machine Learning*, 2018.

Jack Harmer, Linus Gisslén, Jorge del Val, Henrik Holst, Joakim Bergdahl, Tom Olsson, Kristoffer Sjöö, and Magnus Nordin. Imitation learning with concurrent actions in 3d games. In *IEEE Conference on Computational Intelligence and Games*, 2018.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, 2016.

A Jain J Ho. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

Rohit Jena, Changliu Liu, and Katia Sycara. Augmenting gail with bc for sample efficient imitation learning. In *Conference on Robot Learning*, pp. 80–90. PMLR, 2021.

Liyiming Ke, Sanjiban Choudhury, Matt Barnes, Wen Sun, Gilwoo Lee, and Siddhartha Srinivasa. Imitation learning as f-divergence minimization. In *International Workshop on the Algorithmic Foundations of Robotics*, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations*, 2015.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B. Tenenbaum. Learning to act from actionless videos through dense correspondences. *arXiv preprint arXiv:2310.08576*, 2023.

Ilya Kostrikov. Pytorch implementations of reinforcement learning algorithms. `https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail`, 2018.

Ilya Kostrikov, Kumar Krishna Agrawal, Debidatta Dwibedi, Sergey Levine, and Jonathan Tompson. Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. In *International Conference on Learning Representations*, 2019.

Youngwoon Lee, Shao-Hua Sun, Sriram Somasundaram, Edward S. Hu, and Joseph J. Lim. Composing complex skills by learning transition policies. In *Proceedings of International Conference on Learning Representations*, 2019.

Youngwoon Lee, Andrew Szot, Shao-Hua Sun, and Joseph J. Lim. Generalizable imitation learning from observation via inferring goal proximity. In *Neural Information Processing Systems*, 2021.

Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*, 2018.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

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. In *International Conference on Learning Representations*, 2016.

Gabriel Loaiza-Ganem, Brendan Leigh Ross, Jesse C Cresswell, and Anthony L Caterini. Diagnosing and fixing manifold overfitting in deep generative models. *Transactions on Machine Learning Research*, 2022.

Calvin Luo. Understanding diffusion models: A unified perspective. *arXiv preprint arXiv:2208.11970*, 2022.

Abdoulaye O Ly and Moulay Akhloufi. Learning to drive by imitation: An overview of deep behavior cloning methods. *IEEE Transactions on Intelligent Vehicles*, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 2015.

Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *International Conference on Machine Learning*, 2000.

Tung Nguyen, Qinqing Zheng, and Aditya Grover. Reliable conditioning of behavioral cloning for offline reinforcement learning. *arXiv preprint arXiv:2210.05158*, 2023.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, 2021.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, Jan Peters, et al. An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 2018.

Tim Pearce, Tabish Rashid, Anssi Kanervisto, David Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, and Sam Devlin. Imitating human behaviour with diffusion models. In *International Conference on Learning Representations*, 2023.

Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems*, 1989.

Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, Mikhail Kudinov, and Jiansheng Wei. Diffusion-based voice conversion with fast maximum likelihood sampling scheme. In *International Conference on Learning Representations*, 2022.

Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal conditioned imitation learning using score-based diffusion policies. In *Robotics: Science and Systems*, 2023.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, 2015.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, 2011.

Stefan Schaal. Learning from demonstration. In *Advances in Neural Information Processing Systems*, 1997.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, 2015.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Neural Information Processing Systems*, 2019.

Yang Song and Diederik P Kingma. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*, 2021.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *International Joint Conference on Artificial Intelligence*, 2018.

Faraz Torabi, Garrett Warnell, and Peter Stone. Generative adversarial imitation from observation. In *International Conference on Machine Learning*, 2019.

Lingguang Wang, Carlos Fernandez, and Christoph Stiller. High-level decision making for automated highway driving via behavior cloning. *IEEE Transactions on Intelligent Vehicles*, 2022.

Qitian Wu, Rui Gao, and Hongyuan Zha. Bridging explicit and implicit deep generative models via neural stein estimators. In *Neural Information Processing Systems*, 2021.

Jin Xu, Zishan Li, Bowen Du, Miaomiao Zhang, and Jing Liu. Reluplex made more practical: Leaky relu. In *IEEE Symposium on Computers and Communications*, 2020.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems*, 2023.

Konrad Zolna, Scott Reed, Alexander Novikov, Sergio Gomez Colmenarejo, David Budden, Serkan Cabi, Misha Denil, Nando de Freitas, and Ziyu Wang. Task-relevant adversarial imitation learning. In *Conference on Robot Learning*, 2021.