Diffusion Model-Augmented Behavioral Cloning

Anonymous Author(s)
Affiliation
Address
email

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 $p(s, a)$ (*e.g.*, implicit behavioral cloning). Despite its simplicity, modeling the conditional probability with BC usually struggles with generalization. While modeling the joint probability can lead to improved generalization performance, the inference procedure can be time-consuming and it often suffers 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.

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 and 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 and Ermon, 2016, Zolna et al., 2021, Kostrikov et al., 2019] and inverse reinforcement learning [Ng and Russell, 2000, Abbeel and Ng, 2004] have achieved encouraging results in a variety of domains. Yet, these methods require interacting with environments, which can still be expensive or unsafe.

On the other hand, behavioral cloning (BC) [Pomerleau, 1989, Bain and 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

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

a discriminative model $p(a|s)$ that models the *conditional probability* of an action 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 address this issue, implicit behavioral cloning (IBC) [Florence et al., 2022] aims to model the *joint probability* of the expert state-action pairs $p(s, a)$ with energy-based models. IBC demonstrates superior performance when generalization is required. Yet, imitation learning methods in a similar vein [Ganapathi et al., 2022] that model the *joint probability* of state-action pairs $p(s, a)$ instead of directly predicting actions $p(a|s)$ require time-consuming actions sampling and optimization to retrieve a desired action $\arg\max_{a \in \mathcal{A}} p(s, a)$ during inference despite the choice of models.

This work proposes an imitation learning framework that combines both the efficiency 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 estimate produced by the learned diffusion model. 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], autonomous driving [Ly and Akhloufi, 2020], and game AI [Harmer et al., 2018].

Behavioral Cloning (BC). BC [Pomerleau, 1989, Torabi et al., 2018] formulate 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]. In this work, we augment BC by employing 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 and Ermon, 2016] and its extensions [Torabi et al., 2019, Kostrikov et al., 2019, Zolna 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, AIL methods 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 and Russell, 2000, Abbeel and 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

Without loss of generality, the reinforcement learning problem can be formulated as a Markov decision process (MDP), which can be represented by a tuple $M = (S, A, R, P, \rho, \gamma)$ with states S, actions A, reward function $R(S, A) \in (0, 1)$, transition distribution $P(s^{'}|s, a) : S \times A \times S \rightarrow [0, 1]$, initial state distribution ρ, and discounted factor γ. Based on the rewards received while interacting with the environment, the goal is to learn a policy $\pi(\cdot|s)$ to maximize the expectation of the cumulative discounted return (*i.e.*, value function): $V(\pi) = \mathbb{E}[\sum_{t=0}^{T} \gamma^t R(s_t, a_t)|s_0 \sim \rho(\cdot), a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(s_{t+1}|s_t, a_t)]$, where T denotes the episode length. Instead of interacting with the environment and receiving rewards, imitation learning aims to learn an agent 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 $\{s_1^i, a_1^i, ..., s_{n_i}^i, a_{n_i}^i\}$.

3.2 Behavioral Cloning: 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 $\underset{\theta}{max} \sum_{(s,a)\in D} \log(\pi_\theta(a|s))$, 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. Despite its success in various applications, BC tends to overfit and struggle at generalizing to states unseen during training [Ross et al., 2011, Codevilla et al., 2019, Wang et al., 2022].

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

Aiming for improved generalization ability, implicit behavioral cloning [Florence et al., 2022] and methods in a similar vein [Ganapathi et al., 2022] model the *joint probability* $p(s, a)$ of expert state-action pairs. These methods demonstrate superior generalization performance in diverse domains. Yet, without directly modeling the *conditional probability* $p(a|s)$, the action sampling and optimization procedure to retrieve a desired action $\arg\max_{a\in\mathcal{A}}\ p(s, a)$ during inference is often time-consuming.

Moreover, explicit generative models such as energy-based models [Du and Mordatch, 2019, Song and Kingma, 2021], variational autoencoder [Kingma and Welling, 2014], and flow-based models Rezende and Mohamed [2015], Dinh et al. [2017] 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.

3.4 Diffusion Models

As described in the previous sections, this work aims to combine the advantages of modeling both the *conditional probability* $p(a|s)$ and the *joint probability* $p(s, a)$. To this end, 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 and Dhariwal [2021], Dhariwal and Nichol [2021].

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 diffusion models can be found in the Section G.

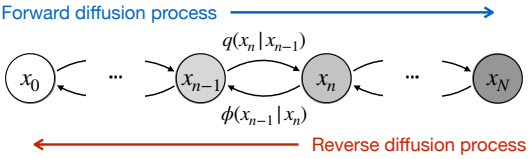


Figure 1: **Denoising Diffusion Probabilistic Model (DDPM).** Latent variables $x_1, ..., x_N$ are produced from the data point x_0 via the forward diffusion process, *i.e.*, gradually adding noises to the latent variables. The diffusion model ϕ learns to reverse the diffusion process by denoising the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

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. Given a sampled expert state-action pair (s, a), a policy π learns to predict an action $\hat{a} \sim \pi(s)$ by optimizing

$$\mathcal{L}_{\text{BC}} = d(a, \hat{a}), \tag{1}$$

where $d(\cdot, \cdot)$ denotes a distance measure between a pair of actions. For example, we can adapt 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

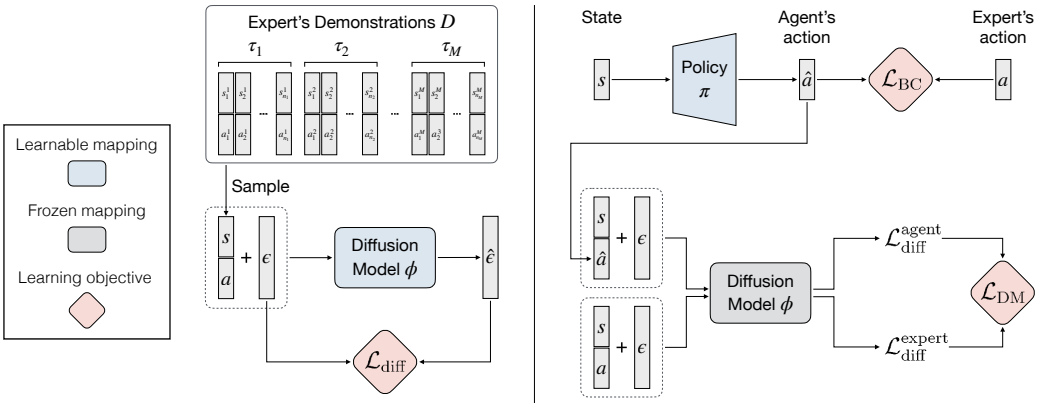
Instead of directly learning the conditional probability $p(a|s)$, this section discusses how to model the joint probability $p(s, a)$ of expert behaviors with a diffusion model in Section 4.2.1 and presents how to leverage the learned diffusion model to guide policy learning in Section 4.2.2.

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

$$\mathcal{L}_{\text{diff}}(s, a, \phi) = ||\hat{\epsilon}(s, a, n) - \epsilon(n)||^2 = ||\phi(s, a, \epsilon(n)) - \epsilon(n)||^2, \tag{2}$$

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



(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 $\mathcal{L}_{\text{diff}}$.

propose to view $\mathcal{L}_{\text{diff}}(s, a, \phi)$ as an estimate of how well the state-action pair (s, a) fits the state-action distribution that ϕ learns from.

4.2.2 Learning a Policy with Diffusion Model Loss

A diffusion model ϕ trained on the expert distribution 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 to imitate the expert. Specifically, given a state-action (s, a) sampled from D, the π predicts an action given the state $\hat{a} \sim \pi(s)$ by optimizing

$$\mathcal{L}_{\text{diff}}^{\text{agent}} = \mathcal{L}_{\text{diff}}(s, \hat{a}, \phi) = ||\hat{\epsilon}(s, \hat{a}, n) - \epsilon||^2. \tag{3}$$

Intuitively, the policy learns to predict actions that are indistinguishable from the expert actions 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:

$$\mathcal{L}_{\text{diff}}^{\text{expert}} = \mathcal{L}_{\text{diff}}(s, a, \phi) = ||\hat{\epsilon}(s, a, n) - \epsilon||^2. \tag{4}$$

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

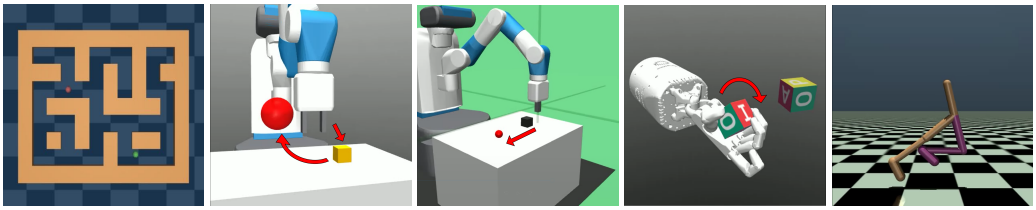
$$\mathcal{L}_{\text{DM}} = max(\mathcal{L}_{\text{diff}}^{\text{agent}} - \mathcal{L}_{\text{diff}}^{\text{expert}}, 0). \tag{5}$$

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 that optimizes the BC loss L_{BC} in Eq. 1 by jointing optimizing the proposed diffusion model loss L_{DM} in Eq. 5, which encourages the policy to predict actions that fit the expert joint probability captured by a diffusion model. 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 analyze the effect of the coefficient in Section 5.6.1.



(a) MAZE (b) FETCHPICK (c) FETCHPUSH (d) HANDROTATE (e) WALKER

Figure 3: **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)-(c) FETCHPICK and FETCHPUSH**: The robot arm manipulation tasks employ a 7-DoF Fetch robotics arm. FETCHPICK requires picking up an object (yellow cube) from the table and moving it to a target location (red); FETCHPUSH requires the arm to push an object (black cube) to a target location (red). **(d) HANDROTATE**: This dexterous manipulation task requires a Shadow Dexterous Hand to in-hand rotate a block to a target orientation. **(e) WALKER**: This locomotion task requires learning a bipedal walker policy to walk as fast as possible while maintaining its balance.

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 A.

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 3a. 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 a robot arm manipulation domain with two 7-DoF Fetch tasks: FETCHPICK and FETCHPUSH, as illustrated in Figure 3c and Figure 3b. FETCHPICK requires picking up an object from the table and lifting it to a target location; FETCHPUSH requires the arm to push an object to a target location. We use the demonstrations provided in Lee et al. [2021] for these tasks. Each dataset contains 10k transitions (303 trajectories for FETCHPICK and 185 trajectories for FETCHPUSH).

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 3d. This environment has a high-dimensional state space (68D) and action space (20D). 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 WALKER environment Brockman et al. [2016], which requires a bipedal agent to walk as fast as possible while maintaining its balance, as illustrated in Figure 3e. We use the demonstrations provided by Kostrikov [2018], which contains 5 trajectories with 5k state-action pairs.

5.2 Baselines

We compare our method DBC with the following baselines.

- **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.

Table 1: **Experimental Result.** We report the mean and the standard deviation of success rate (MAZE, FETCHPICK, FETCHPUSH, HANDROTATE) and return (WALKER), evaluated over three random seeds. Our proposed method (DBC) outperforms the baselines on MAZE, FETCHPICK, FETCHPUSH, HANDROTATE, and performs competitively against the best performing baseline on WALKER.

Method	MAZE	FETCHPICK	FETCHPUSH	HANDROTATE	WALKER
BC	79.35% ± 5.05%	69.15% ± 5.00%	66.02% ± 6.88%	55.48% ± 3.97%	**7066.61** ± 22.79
Implicit BC	81.43% ± 4.88%	72.27% ± 6.71%	77.70% ± 4.42%	14.52% ± 3.04%	685.92 ± 150.26
Diffusion Policy	73.34% ± 5.30%	74.37% ± 3.80%	86.93% ± 3.26%	58.59% ± 2.85%	6429.87 ± 356.70
DBC	**86.99%** ± 2.84%	**88.71%** ± 6.46%	**89.50%** ± 3.99%	**60.34%** ± 4.60%	7057.42 ± 36.19

- **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 to select the desired action with the minimum predicted energy. This baseline serves a representative of the methods that solely model the joint probability $p(s, a)$ of the expert behaviors.
- **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, FETCHPUSH, HANDROTATE), and return (WALKER) in Table 1. The details of model architecture can be found in Section C. Training and evaluation details can be found in Section D. Additional analysis and experimental results can be found in Section E and Section F.

Overall Task Performance. Our proposed method DBC achieves the highest success rates, outperforming our baselines in all the goal-directed tasks (MAZE, FETCHPICK, FETCHPUSH, and HANDROTATE) and perform competitively in WALKER compared to the best-performing baseline (BC). We hypothesize the improvement in the goal-directed tasks can be mostly attributed to the better generalization ability since starting positions and the goals are randomized during evaluation and therefore requires the policy to deal with unseen situation. To verify this hypothesis, we further evaluate the baselines and our method in FETCHPICK and FETCHPUSH with different levels of randomization in Section E.

Locomotion. Unlike the goal-directed tasks, we do not observe significant improvement but competitive results from DBC compared to the best-performing baseline (BC). We hypothesize that this is because locomotion tasks such as WALKER, 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 (7066.61) and DBC (7057.42) performing similarly to the expert with an average return of 7063.72. On the other hand, we hypothesize that Diffusion Policy performs slightly worse due to its design for modeling multimodal behaviors, which is contradictory to learning from this single-mode simulated locomotion task.

Action Space Dimension. While Implicit BC models the joint distribution and generalizes better, it requires time-consuming actions sampling and optimization during inference. Moreover, such procedure may not scale well to high-dimensional action spaces. Our Implicit BC baseline with a derivative-free optimizer struggles in HANDROTATE and WALKER environments, whose action dimensions are 20 and 6, 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.

Inference Efficiency. To evaluate the inference efficiency, we measure and report the number of evaluation episodes per second (↑) for IBC (9.92), Diffusion Policy (1.38), and DBC (**30.79**) on an NVIDIA RTX 3080 Ti GPU in MAZE. This can be attributed to the fact that DBC and BC model the conditional probability $p(a|s)$ and can directly map states to actions during inference. In contrast, Implicit BC requires action sampling and optimization, while Diffusion Policy is required to iteratively denoise sampled noises. This verifies the efficiency of modeling the conditional probability.

(a) (b) (c)

Figure 4: **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) drastically 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 poorly predicted actions (blue dots) produced by the diffusion model.

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), as shown in Figure 4a. 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 requires the ability to generalize. Visualized trajectories of the two policies in Figure 4a show that π_{BC} (orange) fails 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 4b), each for 40 consecutive time steps, resulting the green spiral trajectories visualized in Figure 4c.

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 4b shows that the diffusion model struggles at modeling such expert action distribution with a lower intrinsic dimension. As a result, Figure 4c show that the trajectories of π_{DM} (blue) drastically deviates from the expert trajectories (green) as the diffusion model cannot provide effective loss. On the other hand, the trajectories of π_{BC} (orange) is able to closely follow expert's. This verifies our motivation to complement modeling the joint probability with modeling the conditional probability (*i.e.*, BC).

5.5 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 and Mordatch, 2019, Song and Kingma, 2021], variational autoencoder (VAEs) [Kingma and Welling, 2014], and generative adversarial networks (GANs) Goodfellow et al. [2014]. Each

Table 2: **Generative Models.** We compare using different generative models to model the expert distribution and guide policy learning in MAZE.

Method	without BC	with BC
BC	N/A	79.35% $\pm$ 5.05%
EBM	49.09% $\pm$ 15.15%	80.00% $\pm$ 4.06%
VAE	48.47% $\pm$ 7.57%	82.31% $\pm$ 5.84%
GAN	50.29% $\pm$ 8.27%	71.64% $\pm$ 5.50%
DM	**53.51**% $\pm$ 4.20%	**86.99**% $\pm$ 2.84%

Table 3: **Effect of λ.** We experiment with different values of λ in MAZE, each evaluated over three random seeds.

λ	Success Rate
1	85.40% $\pm$ 4.37%
2	85.64% $\pm$ 3.69%
5	**86.99**% $\pm$ 2.84%
10	85.46% $\pm$ 4.47%
20	85.17% $\pm$ 2.61%

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. Training details can be found in Section D.3.

Table 2 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, verifying our choice of the generative model.

5.6 Ablation Study

In this section, we investigate the effect of the diffusion model loss coefficient λ (Section 5.6.1) and examine the effect of the normalization term $\mathcal{L}_{\text{diff}}^{\text{expert}}$ in the diffusion model loss $\mathcal{L}_{\text{DM}}$ (Section 5.6.2).

5.6.1 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 MAZE. The result presented in Table 3 shows that $\lambda = 5$ yields the best performance. A higher or lower λ leads to worse performance, demonstrating how modeling the conditional probability ($\mathcal{L}_{\text{BC}}$) and the joint probability ($\mathcal{L}_{\text{DM}}$) can complement each other.

5.6.2 Effect of the Normalization Term $\mathcal{L}_{\text{diff}}^{\text{expert}}$

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 in MAZE. 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. This variant learning from an unnormalized diffusion model loss achieves an average success rate of 80.20%, worse than the full DBC (86.99%). This justifies the effectiveness of the proposed normalization term $\mathcal{L}_{\text{diff}}^{\text{expert}}$ in $\mathcal{L}_{\text{DM}}$.

6 Conclusion

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's 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.

References

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.

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.

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

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.

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.

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.

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

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.

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

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.

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

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.

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.

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

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

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

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

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

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.

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.

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

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.

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

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.

Faraz Torabi, Garrett Warnell, and Peter Stone. Generative adversarial imitation from observation. *ICML*, 2019.

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.

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.

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

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.

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.

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.

Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023.

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.

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

Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. 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.

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

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

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

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

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.

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

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

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

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

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.

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.

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 (ICML)*, 2018.

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

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