Aligning Diffusion Behaviors with Q-functions for Efficient Continuous Control

Huayu Chen[1,2], **Kaiwen Zheng**[1,2], **Hang Su**[1,2,3], **Jun Zhu**[1,2,3*]
[1]Department of Computer Science and Technology, Tsinghua University
[2]Institute for AI, BNRist Center, Tsinghua-Bosch Joint ML Center, THBI Lab, Tsinghua University
[3]Pazhou Lab (Huangpu), Guangzhou, China

Abstract

Drawing upon recent advances in language model alignment, we formulate offline Reinforcement Learning as a two-stage optimization problem: First pretraining expressive generative policies on reward-free behavior datasets, then fine-tuning these policies to align with task-specific annotations like Q-values. This strategy allows us to leverage abundant and diverse behavior data to enhance generalization and enable rapid adaptation to downstream tasks using minimal annotations. In particular, we introduce Efficient Diffusion Alignment (EDA) for solving continuous control problems. EDA utilizes diffusion models for behavior modeling. However, unlike previous approaches, we represent diffusion policies as the derivative of a scalar neural network with respect to action inputs. This representation is critical because it enables direct density calculation for diffusion models, making them compatible with existing LLM alignment theories. During policy fine-tuning, we extend preference-based alignment methods like Direct Preference Optimization (DPO) to align diffusion behaviors with continuous Q-functions. Our evaluation on the D4RL benchmark shows that EDA exceeds all baseline methods in overall performance. Notably, EDA maintains about 95% of performance and still outperforms several baselines given only 1% of Q-labelled data during fine-tuning. Code: `https://github.com/thu-ml/Efficient-Diffusion-Alignment`

1 Introduction

Learning diverse behaviors is generative modeling; transforming them into optimized policies is reinforcement learning. Recent studies have identified diffusion policies as a powerful tool for representing heterogeneous behavior datasets [21, 38]. However, these behavior policies incorporate suboptimal decisions in datasets, making them unsuitable for direct deployment in downstream tasks. To get optimized policies, typical methods involve either augmenting the behavior policy with an additional guidance/evaluation network [21, 32, 15] or training a new evaluation policy supervised by the behavior policy [13, 4]. While functional, these methods fail to leverage the full potential of pretrained behaviors as they require constructing some new policy models from scratch. This raises the question: *Can we directly fine-tune pretrained diffusion behaviors into optimized policies?*

Recent advances in Large Language Model (LLM) alignment techniques [57, 37, 43] offer valuable insights for fine-tuning diffusion behavior policies due to the fundamental similarity of the issues they aim to address (Fig. 1). While pretrained LLMs accurately imitate language patterns from web-scale corpus, they also capture toxic or unwanted content within the dataset. Alignment algorithms, such as Direct Preference Optimization (DPO, [41]), are designed to remove harmful or useless content learned during pretraining. They enable quick adaptations of pretrained LLMs to human intentions

*The corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

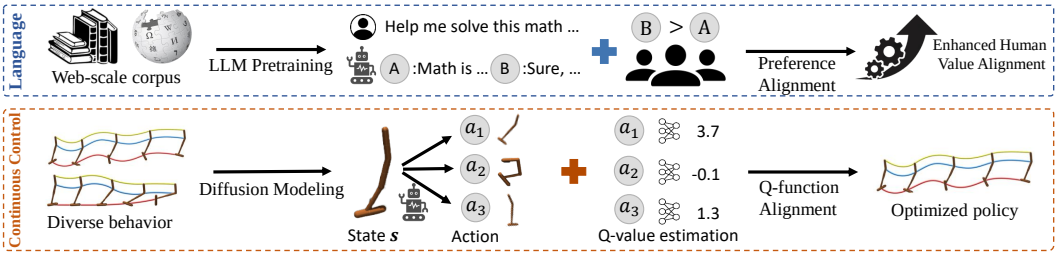


Figure 1: Comparison between alignment strategies for LLMs and diffusion policies (ours).

by fine-tuning them on a small dataset annotated with human preference labels. These strategies, due to their simplicity and effectiveness, have seen widespread applications in academia and industry.

Despite the high similarity in problem formulation and the immense potential of LLM alignment techniques, they cannot be directly applied to fine-tune diffusion policy in domains like continuous control. This is primarily because LLMs employ Categorical models to deal with discrete actions (tokens). Their alignment relies on computing data probabilities for maximum likelihood training (Sec. 2.2). However, diffusion models lack a tractable probability calculation method in continuous action space [5]. Additionally, the data annotation method differs significantly between two areas: LM alignment uses binary preference labels for comparing responses, while continuous control uses scalar Q-functions for evaluating actions (Fig. 1).

To allow aligning diffusion behavior models with Q-functions for policy optimization, we introduce Efficient Diffusion Alignment (EDA). EDA consists of two stages: behavior pretraining and policy fine-tuning. During the pretraining stage, we learn a conditional diffusion behavior model on reward-free datasets. Different from previous work which constructs diffusion models as an end-to-end network, we represent diffusion policies as the derivative of a scalar neural network with respect to action inputs. This representation is critical because it enables direct density calculation for diffusion policies. We demonstrate that the scalar network exactly outputs the unnormalized density of behavior distributions, making diffusion policies compatible with existing LLM alignment theories.

During the fine-tuning stage, we propose a novel algorithm that directly fine-tunes pretrained behavior models into optimized diffusion policies. The training objective is strictly derived by constructing a classification task to predict the optimal action using log-probability ratios between the policy and the behavior model. Our approach innovatively expands DPO by allowing fine-tuning on an arbitrary number of actions annotated with explicit Q-values, beyond just the typical binary preference data.

One main advantage of EDA is that it enables fast and data-efficient adaptations of behavior models in downstream tasks. Our experiments on the D4RL benchmark [10] show that EDA maintains 95 % of its performance and still surpasses baselines like IQL [25] with just 1% of Q-labelled data relative to the pretraining phase. Besides, EDA exhibits rapid convergence during fine-tuning, requiring only about 20K gradient steps (about 2% of the typical 1M policy training steps) to achieve convergence. Finally, EDA outperforms all reference baselines in overall performance with access to the full datasets. We attribute the high efficiency of EDA to its exploitation of the diffusion behavior models' generalization ability acquired during pretraining.

Our key contributions: 1. We represent diffusion policies as the derivative of a scalar value network to allow direct density estimation. This makes diffusion policies compatible with the existing alignment framework. 2. We extend preference-based alignment methods and propose EDA to align diffusion behaviors with scalar Q-functions, showcasing its vast potential in continuous control.

2 Background

2.1 Offline Reinforcement Learning

Offline RL aims to tackle decision-making problems by solely utilizing a pre-collected behavior dataset. Consider a typical Markov Decision Process (MDP) described by the tuple $\langle \mathcal{S}, \mathcal{A}, P, r, \gamma \rangle$. $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a})$ is the transition function, $r(\boldsymbol{s}, \boldsymbol{a})$ is the reward function and γ is the discount factor. Given a static dataset $\mathcal{D}^\mu := \{\boldsymbol{s}, \boldsymbol{a}, r, \boldsymbol{s}'\}$ representing

interaction history between an implicit policy μ and the MDP environment, our goal is to learn a new policy that maximizes cumulative rewards in this MDP while staying close to the behavior policy μ.

Offline RL can be formalized as a constrained policy optimization problem [26, 35, 53]:

$$\max_{\pi} \mathbb{E}_{\boldsymbol{s}\sim\mathcal{D}^{\mu},\boldsymbol{a}\sim\pi(\cdot|\boldsymbol{s})} Q(\boldsymbol{s},\boldsymbol{a}) - \beta D_{\mathrm{KL}}\left[\pi(\cdot|\boldsymbol{s})||\mu(\cdot|\boldsymbol{s})\right], \tag{1}$$

where $Q(\boldsymbol{s},\boldsymbol{a})$ is an action evaluation network that can be learned from $\mathcal{D}^{\mu}$. β is a temperature coefficient. Previous work [40, 39] proves that the optimal solution for solving Eq. 1 is:

$$\pi^*(\boldsymbol{a}|\boldsymbol{s}) = \frac{1}{Z(\boldsymbol{s})}\mu(\boldsymbol{a}|\boldsymbol{s})e^{Q(\boldsymbol{s},\boldsymbol{a})/\beta}. \tag{2}$$

In this paper, we focus on how to efficiently learn parameterized policies for modeling π^*.

2.2 Direct Preference Optimization for Language Model Alignment

Direct Preference Optimization (DPO, [41]) is a fine-tuning technique for aligning pretrained LLMs with human feedback. Suppose we already have a pretrained LLM model $\mu(\boldsymbol{a}|\boldsymbol{s})$, where $\boldsymbol{s}$ represents user instructions, and $\boldsymbol{a}$ represents generated responses. The goal is to align μ_ϕ with some implicit evaluation rewards $r^{\mathrm{LM}}(\boldsymbol{s},\boldsymbol{a})$ that reflect human preference. Our target model is $\pi^*(\boldsymbol{a}|\boldsymbol{s}) \propto \mu_\phi(\boldsymbol{a}|\boldsymbol{s})e^{r^{\mathrm{LM}}(\boldsymbol{s},\boldsymbol{a})/\beta}$.

DPO assumes we only have access to some pairwise preference data $\{\boldsymbol{s} \rightarrow (\boldsymbol{a}_w > \boldsymbol{a}_l)\}$ and the preference probability is influenced by $r^{\mathrm{LM}}(\boldsymbol{s},\boldsymbol{a})$. Formally,

$$p(\boldsymbol{a}_w \succ \boldsymbol{a}_l|\boldsymbol{s}) := \frac{e^{r^{\mathrm{LM}}(\boldsymbol{s},\boldsymbol{a}_w)}}{e^{r^{\mathrm{LM}}(\boldsymbol{s},\boldsymbol{a}_l)} + e^{r^{\mathrm{LM}}(\boldsymbol{s},\boldsymbol{a}_w)}} = \sigma(r^{\mathrm{LM}}(\boldsymbol{s},\boldsymbol{a}_w) - r^{\mathrm{LM}}(\boldsymbol{s},\boldsymbol{a}_l)), \tag{3}$$

where σ is the sigmoid function.

In order to learn $\pi_\theta \approx \pi^*(\boldsymbol{a}|\boldsymbol{s}) \propto \mu_\phi(\boldsymbol{a}|\boldsymbol{s})e^{r^{\mathrm{LM}}(\boldsymbol{s},\boldsymbol{a})/\beta}$, DPO first parameterizes a reward model using the log-probability ratio between π_θ and μ_ϕ, and then optimizes this reward model through maximum likelihood training:

$$\mathcal{L}_{\mathrm{DPO}} = -\mathbb{E}_{\{\boldsymbol{s},\boldsymbol{a}_w\succ\boldsymbol{a}_l\}} \log\sigma(r^{\mathrm{LM}}_\theta(\boldsymbol{s},\boldsymbol{a}_w) - r^{\mathrm{LM}}_\theta(\boldsymbol{s},\boldsymbol{a}_l)), \tag{4}$$

$$\text{where} \qquad r^{\mathrm{LM}}_\theta(\boldsymbol{s},\boldsymbol{a}) := \beta\log\frac{\pi_\theta(\boldsymbol{a}|\boldsymbol{s})}{\mu_\phi(\boldsymbol{a}|\boldsymbol{s})}$$

The key insight behind DPO's loss function is the equivalence and mutual convertibility between the policy model and the reward model. This offers a new perspective for solving generative policy optimization problems by applying discriminative classification loss.

2.3 Diffusion Modeling for Estimating Behavior Score Functions

Recent studies show that diffusion models [45, 20, 49] excel at representing heterogeneous behavior policies in continuous control [21, 5, 38]. To train a diffusion behavior model, we first gradually inject Gaussian noise into action points according to the forward diffusion process:

$$\boldsymbol{a}_t = \alpha_t\boldsymbol{a} + \sigma_t\boldsymbol{\epsilon}, \tag{5}$$

where $t \in [0,1]$, and $\boldsymbol{\epsilon}$ is standard Gaussian noise. $\alpha_t, \sigma_t \in [0,1]$ are manually defined so that at time $t=0$, we have $\boldsymbol{a}_t = \boldsymbol{a}$ and at time $t=1$, we have $\boldsymbol{a}_t \approx \boldsymbol{\epsilon}$. When $\boldsymbol{a}$ is sampled from the behavior policy $\mu(\boldsymbol{a}|\boldsymbol{s})$, the marginal distribution of $\boldsymbol{a}_t$ at various time t satisfies

$$\mu_t(\boldsymbol{a}_t|\boldsymbol{s},t) = \int \mathcal{N}(\boldsymbol{a}_t|\alpha_t\boldsymbol{a},\sigma_t^2\boldsymbol{I})\mu(\boldsymbol{a}|\boldsymbol{s},t)\mathrm{d}\boldsymbol{a}. \tag{6}$$

Intuitively, the diffusion training objective predicts the noise added to the original behavior actions:

$$\min_{\phi} \mathbb{E}_{t,\boldsymbol{\epsilon},\boldsymbol{s},\boldsymbol{a}\sim\mu(\cdot|\boldsymbol{s})}\left[\|\boldsymbol{\epsilon}_\phi(\boldsymbol{a}_t|\boldsymbol{s},t) - \boldsymbol{\epsilon}\|_2^2\right]_{\boldsymbol{a}_t=\alpha_t\boldsymbol{a}+\sigma_t\boldsymbol{\epsilon}}. \tag{7}$$

More formally, it can be proved that the learned "noise predictor" $\boldsymbol{\epsilon}_\phi$ actually represents the *score function* $\nabla_{\boldsymbol{a}_t}\log\mu_t(\boldsymbol{a}_t|\boldsymbol{s},t)$ of the diffused behavior distribution μ_t [49]:

$$\nabla_{\boldsymbol{a}_t}\log\mu_t(\boldsymbol{a}_t|\boldsymbol{s},t) = -\boldsymbol{\epsilon}^*(\boldsymbol{a}_t|\boldsymbol{s},t)/\sigma_t \approx -\boldsymbol{\epsilon}_\phi(\boldsymbol{a}_t|\boldsymbol{s},t)/\sigma_t. \tag{8}$$

With such a score-function estimator, we can employ existing numerical solvers [46, 33] to reverse the diffusion process, and sample actions from the learned behavior policy μ_ϕ.

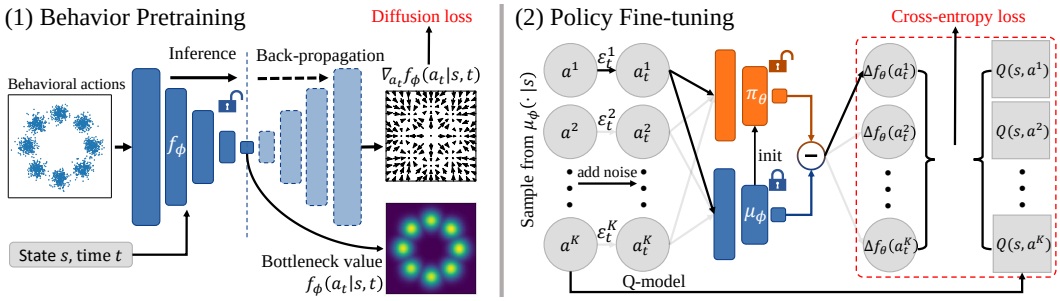


Figure 2: Algorithm overview. **Left:** In behavior pretraining, the diffusion behavior model is represented as the derivative of a scalar neural network with respect to action inputs. The scalar outputs of the network can later be utilized to estimate behavior density. **Right:** In policy fine-tuning, we predict the optimality of actions in a contrastive manner among K candidates. The prediction logit for each action is the density gap between the learned policy model and the frozen behavior model. We use cross-entropy loss to align prediction logits $\triangle f_\theta := f_\theta^\pi - f_\theta^\mu$ with dataset Q-labels.

3 Method

We decompose the policy optimization problem into two stages: behavior pretraining (Sec. 3.1) and policy alignment (Sec. 3.2).

3.1 Bottleneck Diffusion Models for Efficient Behavior Density Estimation

Recent advances in alignment techniques cannot be readily applied to continuous control tasks. Their successful applications in LLM fine-tuning require two essential prerequisites:

1. A powerful foundation model capable of capturing diverse behaviors within datasets.
2. A tractable probability calculation method that allows direct density estimation (Eq. 4).

Language models primarily deal with discrete actions (tokens) defined by a vocabulary set $\mathcal{V}$, and thus employ Categorical models. This modeling method enables easy calculation of data probability through a softmax operation and is capable of representing *any* discrete distribution. In contrast, for continuous action space, direct density estimation is not so feasible. Diffusion policies only estimate the gradient field (a.k.a., score) of data density instead of the density value itself [49], making it impossible to directly apply LLM alignment techniques [41, 9, 3]. Conventional Gaussian policies have a tractable probability formulation but lack enough expressivity and multimodality needed to accurately model behavior datasets [52, 13, 5], and catastrophically fail in our initial experiments.

To address the above limitation of diffusion models, we propose a new diffusion modeling technique to enable direct density estimation. Normally, a conditional diffusion policy $\epsilon_\phi(a_t|s,t) : \mathcal{A}\times\mathcal{S}\times\mathbb{R} \to \mathbb{R}^{|\mathcal{A}|}$ maps noisy actions a_t to predicted noises $\epsilon \in \mathbb{R}^{|\mathcal{A}|}$. In our approach, we redefine ϵ_ϕ as the derivative of a scalar network $f_\phi(a_t|s,t) : \mathcal{A}\times\mathcal{S}\times\mathbb{R} \to \mathbb{R}$ with respect to input a_t:

$$\epsilon_\phi(a_t|s,t) := -\sigma_t \nabla_{a_t} f_\phi(a_t|s,t). \tag{9}$$

Given that f_ϕ is a parameterized network, its gradient computation can be conveniently performed by auto-differential libraries. The new training objective for f_ϕ can then be reformulated from Eq. 7:

$$\min_\phi \mathcal{L}_\mu(\phi) = \mathbb{E}_{t,\boldsymbol{\epsilon},(\boldsymbol{s},\boldsymbol{a})\sim\mathcal{D}^\mu}\left[\|\sigma_t \nabla_{a_t} f_\phi(a_t|s,t) + \boldsymbol{\epsilon}\|_2^2\right]_{\boldsymbol{a}_t=\alpha_t\boldsymbol{a}+\sigma_t\boldsymbol{\epsilon}}. \tag{10}$$

As noted by [49], with unlimited model capacity, the optimal solution for solving Eq. 10 is:

$$\boldsymbol{\epsilon}^*(\boldsymbol{a}_t|\boldsymbol{s},t) = -\sigma_t \nabla_{\boldsymbol{a}_t} \log \mu_t(\boldsymbol{a}_t|\boldsymbol{s},t) \Longrightarrow f^*(a_t|s,t) = \log \mu_t(\boldsymbol{a}_t|\boldsymbol{s},t) + C(\boldsymbol{s},t). \tag{11}$$

An illustration is provided in Figure 2 (left). Intuitively, our proposed modeling method first compresses the input action into a scalar value with one single dimension. Then, this bottleneck value is expanded back to $\mathbb{R}^{|\mathcal{A}|}$ through back-propagation. We thus refer to f_ϕ as *Bottleneck* Diffusion Models (BDMs).

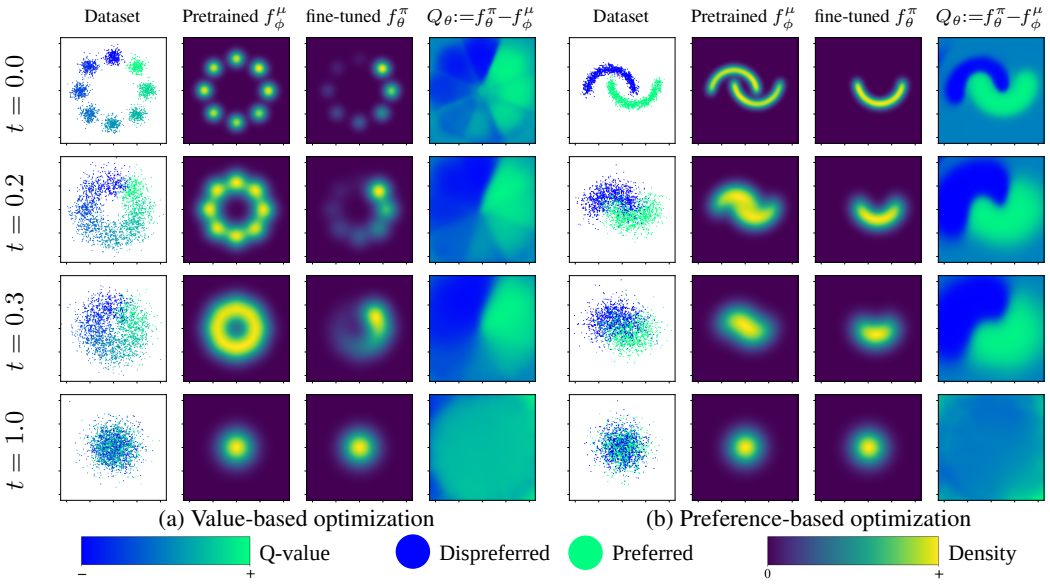


Figure 3: Experimental results of EDA in 2D bandit settings at different diffusion times. **Column 1:** Visualization of diversified behavior datasets. Each dot represents a two-dimensional behavioral action. Its color reflects the action's Q-value. **Column 2 & 3:** Density maps of the action distribution as estimated by the pretrained or fine-tuned BDM models. The density for low-Q-value actions has been effectively decreased after fine-tuning. **Column 4:** The predicted action Q-values, calculated by Eq. 13, align with dataset Q-values in Column 1. See appendix B for complete results.

The primary advantage of BDMs is their ability to efficiently estimate behavior densities in a single forward pass. Moreover, BDMs are fully compatible with existing diffusion-based codebases regarding training and sampling procedures, inheriting their key benefits such as training stability and model expressivity. BDMs can also be viewed as a diffused version of Energy-Based Models (EBMs, [8]). We refer interested readers to Appendix A for a detailed discussion.

3.2 Policy Optimization by Aligning Diffusion Behaviors with Q-functions

In this section, our goal is to learn a new policy $\pi_\theta \propto \mu_\phi e^Q$ by fine-tuning the previously pretrained behavior policy μ_ϕ on a new dataset annotated by an existing Q-function $Q(\boldsymbol{s}, \boldsymbol{a})$. We show that this policy optimization problem can actually be transformed into a simple classification task for predicting the optimal action among multiple candidates. We elaborate on our method below.

Dataset construction. For any state $\boldsymbol{s}$, we draw $K > 1$ independent action samples $\boldsymbol{a}^{1:K}$ from $\mu_\phi(\cdot|\boldsymbol{s})$. Assume we already have an Q-function $Q(\boldsymbol{s}, \boldsymbol{a})$ that evaluates input state-action pairs in scalar values, our dataset is formed as $\mathcal{D}^f := \{\boldsymbol{s}, \boldsymbol{a}^{1:K}, Q(\boldsymbol{s}, \boldsymbol{a}^k)|_{k\in 1:K}\}$.

Action optimality. We first introduce a formal notion of action optimality. We draw from the control-as-probabilistic-inference framework [29] and define a random optimality variable $\mathcal{O}_K$, which is a one-hot vector of length K. The k-th index of $\mathcal{O}_K$ being 1 indicates that $\boldsymbol{a}^k$ is the optimal action within K action candidates $\boldsymbol{a}^{1:K}$. We have

$$p(\mathcal{O}_K^k = 1|\boldsymbol{s}, \boldsymbol{a}^{1:K}) = \frac{e^{Q(\boldsymbol{s},\boldsymbol{a}^k)}}{\sum_{i=1}^K e^{Q(\boldsymbol{s},\boldsymbol{a}^i)}}.$$

The optimality probability of a behavioral action $\boldsymbol{a}$ is proportional to the exponential of its Q-value, aligning with the optimal policy definition $\pi^*(\boldsymbol{a}|\boldsymbol{s}) \propto \mu(\boldsymbol{a}|\boldsymbol{s})e^{Q(\boldsymbol{s},\boldsymbol{a})/\beta}$.

From policy optimization to action classification. We construct a classification task by predicting the optimal action among K candidates. This requires learning a Q-model termed Q_θ first. Drawing inspiration from DPO (Sec. 2.2), we parameterize Q_θ as the log probability ratio between π_θ and μ_ϕ:

$$Q_\theta(\boldsymbol{s}, \boldsymbol{a}) := \beta \log \frac{\pi_\theta(\boldsymbol{a}|\boldsymbol{s})}{\mu_\phi(\boldsymbol{a}|\boldsymbol{s})} + \beta \log Z(\boldsymbol{s}),$$

This parameterization allows us to directly optimize π_θ during training because $\pi_\theta(\boldsymbol{a}|\boldsymbol{s}) = \frac{1}{Z(\boldsymbol{s})}\mu_\phi(\boldsymbol{a}|\boldsymbol{s})e^{Q_\theta(\boldsymbol{s},\boldsymbol{a})/\beta}$ constantly holds. Since Q-values in datasets define the probability of being the optimal action, the training objective can be derived by applying cross-entropy loss:

$$\max_\theta \mathcal{L}_\pi(\theta) = \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a}^{1:K})\sim\mathcal{D}^f}\left[\sum_{k=1}^{K} \underbrace{\frac{e^{Q(\boldsymbol{s},\boldsymbol{a}^k)}}{\sum_{i=1}^{K} e^{Q(\boldsymbol{s},\boldsymbol{a}^i)}}}_{\text{optimality probability}} \log \underbrace{\frac{e^{\beta \log \frac{\pi_\theta(\boldsymbol{a}^k|\boldsymbol{s})}{\mu_\phi(\boldsymbol{a}^k|\boldsymbol{s})} \cancel{+\beta \log Z(\boldsymbol{s})}}}{\sum_{i=1}^{K} e^{\beta \log \frac{\pi_\theta(\boldsymbol{a}^i|\boldsymbol{s})}{\mu_\phi(\boldsymbol{a}^i|\boldsymbol{s})} \cancel{+\beta \log Z(\boldsymbol{s})}}}}_{\text{predicted probability}}\right]. \tag{12}$$

The unknown partition function $Z(\boldsymbol{s})$ automatically cancels out during division. β is a hyperparameter that can be tuned to control how far π_θ deviates from μ_ϕ.

Expanding to bottleneck diffusion behavior. A reliable behavior density estimation of $\mu_\phi(\boldsymbol{a}|\boldsymbol{s})$ is critical and is a main challenge for applying Eq. 12. Our initial experiments tested with Gaussian policies drastically failed and even underperformed vanilla behavior cloning. This highlights the necessity of adopting a much more powerful generative policy, such as the BDM model (Sec. 3.1).

Diffusion policies define a series of distributions $\pi_t(\boldsymbol{a}_t|\boldsymbol{s},t) = \int \mathcal{N}(\boldsymbol{a}_t|\alpha_t\boldsymbol{a}_0, \sigma_t^2\boldsymbol{I})\pi_0(\boldsymbol{a}_0|\boldsymbol{s},t)\mathrm{d}\boldsymbol{a}_0$ at different timesteps $t \in [0,1]$, rather than just a single distribution $\pi = \pi_0$. Consequently, instead of directly predicting the optimal action given raw actions $\boldsymbol{a}^{1:K}$, we perturb all actions with K independent Gaussian noises according to the diffusion forward process by applying $\boldsymbol{a}_t^k = \alpha_t\boldsymbol{a}^k + \sigma_t\boldsymbol{\epsilon}^k$. Then we predict action optimality given K noisy action $\boldsymbol{a}_t^{1:K}$:

$$\begin{aligned} Q_\theta(\boldsymbol{s},\boldsymbol{a}_t,t) :=& \beta \log \frac{\pi_{t,\theta}(\boldsymbol{a}_t|\boldsymbol{s},t)}{\mu_{t,\phi}(\boldsymbol{a}_t|\boldsymbol{s},t)} + \beta \log Z(\boldsymbol{s}) \\ =& \beta[f_\theta^\pi(\boldsymbol{a}_t|\boldsymbol{s},t) - f_\phi^\mu(\boldsymbol{a}_t|\boldsymbol{s},t)] + \beta[\log Z(\boldsymbol{s},t) - C^\pi(\boldsymbol{s},t) + C^\mu(\boldsymbol{s},t)], \end{aligned} \tag{13}$$

Similar to Eq. 12, all unknown terms above automatically cancel out in the training objective:

$$\max_\theta \mathcal{L}_f(\theta) = \mathbb{E}_{t,\boldsymbol{\epsilon}^{1:K},(\boldsymbol{s},\boldsymbol{a}^{1:K})\sim\mathcal{D}^f}\left[\sum_{k=1}^{K} \underbrace{\frac{e^{Q(\boldsymbol{s},\boldsymbol{a}^k)}}{\sum_{i=1}^{K} e^{Q(\boldsymbol{s},\boldsymbol{a}^i)}}}_{\text{optimality probability}} \log \underbrace{\frac{e^{\beta[f_\theta^\pi(\boldsymbol{a}_t^k|\boldsymbol{s},t) - f_\phi^\mu(\boldsymbol{a}_t^k|\boldsymbol{s},t)]}}{\sum_{i=1}^{K} e^{\beta[f_\theta^\pi(\boldsymbol{a}_t^i|\boldsymbol{s},t) - f_\phi^\mu(\boldsymbol{a}_t^i|\boldsymbol{s},t)]}}}_{\text{predicted probability on noisy actions}}\right]. \tag{14}$$

Proposition 3.1. *(Proof in Appendix C) Let f_θ^* be the optimal solution of Problem 14 and $\pi_{t,\theta}^* \propto e^{f_\theta^*}$ be the optimal diffusion policy. Assuming unlimited model capacity and data samples, we have the following results:*

(a) Optimality Guarantee. *At time $t = 0$, the learned policy π_θ^* converges to the optimal target policy.*

$$\pi_\theta^*(\boldsymbol{a}|\boldsymbol{s}) = \pi_{t=0,\theta}^*(\boldsymbol{a}|\boldsymbol{s}) \propto \mu_\phi(\boldsymbol{a}|\boldsymbol{s})e^{Q(\boldsymbol{s},\boldsymbol{a})/\beta}$$

(b) Diffusion Consistency. *At time $t > 0$, $\pi_{t>0,\theta}$ models the diffused distribution of π_θ^*:*

$$\pi_{t,\theta}^*(\boldsymbol{a}|\boldsymbol{s},t) = \int \mathcal{N}(\boldsymbol{a}_t|\alpha_t\boldsymbol{a}, \sigma_t^2\boldsymbol{I})\pi_\theta^*(\boldsymbol{a}_0|\boldsymbol{s})\mathrm{d}\boldsymbol{a}_0$$

asymptotically holds when $K \to \infty$ and $\beta = 1$, satisfying the definition of diffusion process (Eq. 6).

Fine-tuning Efficiency. In practice, the policy and the behavior model share the same architecture, so we can initialize $\theta = \phi$ to fully exploit the generalization capabilities acquired during the pretraining phase. This fine-tuning technique allows us to perform policy optimization with an incredibly small amount of Q-labelled data (e.g., 10k samples) and optimization steps (e.g., 20K gradient steps). These are less than 5% of previous approaches (Sec. 5.2).

4 Related Work

4.1 Diffusion Modeling for Offline Continuous Control

Recent advancements in offline RL have identified diffusion models as an effective approach for behavior modeling [38, 16], which excels at representing complex and multimodal distributions

Environment	Dataset	BCQ	CQL	IQL	DT	D-Diffuser	IDQL	Diffusion-QL	QGPO	BDM (Ours)
HalfCheetah	Medium-Expert	64.7	91.6	86.7	86.8	90.6	**95.9**	**96.8**	**93.5**	**93.2 ± 1.2**
HalfCheetah	Medium	40.7	44.0	47.4	42.6	49.1	51.0	51.1	54.1	**57.0 ± 0.5**
HalfCheetah	Medium-Replay	38.2	45.5	44.2	36.6	39.3	45.9	47.8	47.6	**51.6 ± 0.9**
Hopper	Medium-Expert	100.9	105.4	91.5	**107.6**	**111.8**	**108.6**	**111.1**	**108.0**	104.9 ± 7.4
Hopper	Medium	54.5	58.5	66.3	67.6	79.3	65.4	90.5	**98.0**	**98.4 ± 3.9**
Hopper	Medium-Replay	33.1	95.0	94.7	82.7	**100.0**	92.1	**100.7**	**96.9**	92.7 ± 10.0
Walker2d	Medium-Expert	57.5	**108.8**	**109.6**	**108.1**	**108.8**	**112.7**	**110.1**	**110.7**	**111.1 ± 0.7**
Walker2d	Medium	53.1	72.5	78.3	74.0	82.5	82.5	**87.0**	**86.0**	**87.4 ± 1.1**
Walker2d	Medium-Replay	15.0	77.2	73.9	66.6	75.0	85.1	**95.5**	84.4	89.2 ± 5.5
Average (D4RL Locomotion)		50.9	77.6	76.9	74.7	81.8	82.1	**88.0**	**86.6**	**87.3**
AntMaze	Umaze	73.0	74.0	87.5	59.2	-	**94.0**	**93.4**	**96.4**	**93.0 ± 4.5**
AntMaze	Umaze-Diverse	61.0	84.0	62.2	**53.0**	-	**80.2**	66.2	74.4	**81.0 ± 7.4**
AntMaze	Medium-Play	0.0	61.2	71.2	0.0	-	**84.5**	76.6	**83.6**	79.0 ± 4.2
AntMaze	Medium-Diverse	8.0	53.7	70.0	0.0	-	**84.8**	78.6	**83.8**	**84.0 ± 8.2**
Average (D4RL Locomotion)		35.5	68.2	72.7	28.1	-	**85.9**	78.7	**84.6**	**84.3**
Kitchen	Complete	8.1	43.8	62.5	-	-	-	**84.0**	62.8	**81.5 ± 7.3**
Kitchen	Partial	18.9	49.8	46.3	-	57.0	-	60.5	**66.0**	**69.3 ± 4.6**
Kitchen	Mixed	8.1	51.0	51.0	-	**65.0**	-	**62.6**	45.5	**65.3 ± 2.2**
Average (D4RL Locomotion)		11.7	48.2	53.3	-	-	-	**69.0**	58.1	**72.0**
Average (D4RL Overall)		39.7	69.7	71.4	-	-	-	**82.1**	**80.7**	**83.7**

Table 1: Evaluation results of D4RL benchmarks (normalized according to [10]). We report mean ± standard deviation of algorithm performance across 5 random seeds at the end of training. Numbers within 5 % of the maximum are highlighted.

compared with other modeling methods like Gaussians [54, 55, 24, 35, 53] or VAEs [11, 26, 12]. However, optimizing diffusion models can be a bit more tricky due to the unavailability of a tractable probability calculation method [5, 51]. Existing approaches include learning a separate guidance network to guide the sampling process during evaluation [21, 32], applying classifier-free guidance [1, 7], backpropagating sampled actions through the diffusion policy model to maximize Q-values [52, 22], distilling new policies from diffusion behaviors [13, 4], using rejection sampling to filter out behavioral actions with low Q-values [5, 15] and applying planning-based techniques [30, 36, 16]. Our proposed method differs from all previous work in that it directly fine-tunes the pretrained behavior to align with task-specific annotations, instead of learning a new downstream policy.

4.2 Preference-based Alignment in Offline Reinforcement Learning

Existing alignment algorithms are largely preference-based methods. Preference-based Reinforcement Learning (PbRL) assumes the reward function is unknown, and must be learned from data. Previous work usually applies inverse RL to learn a reward model first and then uses this reward model for standard RL training [19, 57, 28, 44, 34]. This separate learning of a reward model adds complexity to algorithm implementation and thus limits its application. To address this issue, recent work like OPPO [23], DPO [41], and CPL [17] respectively proposes methods to align Gaussian or Categorical policies directly with human preference. Despite their simplicity and effectiveness, these techniques require calculating model probability, and thus cannot be applied to diffusion policies. Existing diffusion alignment strategies [7, 56, 2, 51] are incompatible with mainstream PbRL methods. Our work effectively closes this gap by introducing bottleneck diffusion models. We also extend existing PbRL methods to align with continuous Q-functions instead of just binary preference data.

5 Experiments

We conduct experiments to answer the following questions:

- How does EDA perform compared with other baselines in standard benchmarks? (Sec. 5.1)
- Is the alignment stage data-efficient and training-efficient? How many training steps and annotated data does EDA require for aligning pretrained behavior models? (Sec. 5.2)
- Does value-based alignment outperform preference-based alignment? (Sec. 5.3)
- How do contrastive action number K and other choices affect the performance? (Sec. 5.4)

5.1 D4RL Evaluation

Benchmark. In Table 1, we evaluate the performance of EDA in D4RL benchmarks [10]. All evaluation tasks have a continuous action space and can be broadly categorized into three types: `MuJoCo locomotion` are tasks for controlling legged robots to move forward, where datasets are generated by a variety of policies, including expert, medium, and mixed levels. `Antmaze` is about an ant robot navigating itself in a maze and requires both low-level control and high-level navigation. `FrankaKitchen` are manipulation tasks containing real-world datasets. It is critical to faithfully imitate human behaviors in these tasks.

Experimental setup. Throughout our experiments, we set the contrastive action number $K = 16$. For each task, we train an action evaluation model $Q_\psi(\boldsymbol{s}, \boldsymbol{a})$ for annotating behavioral data during the fine-tuning stage. Implicit Q-learning [25] is employed for training Q_ψ due to its simplicity and orthogonality to policy training. We compare with other critic training methods in Figure 4. The rest of the implementation details are in Appendix D.

Baselines. We mainly consider diffusion-based RL methods with various optimization techniques. `Decision Diffuser` [1] employs classifier-free guidance for optimizing behavior models. `QGPO` employs energy guidance. `IDQL` [15] does not optimize the behavior policy and simply uses rejection sampling during evaluation. `Diffusion-QL` [52] has no explicit behavior model, but instead adopts a diffusion regularization loss. We also reference well-studied conventional algorithms for different classes of generative policies. `BCQ` [11] features a VAE-based policy. `DT` [6] has a transformer-based architecture. `CQL` [27] and `IQL` [25] targets Gaussian policies.

Result analysis. From table 1, we find that EDA surpasses all referenced baselines regarding overall performance and provides a competitive number in each D4RL task. To ascertain whether this improvement stems from the alignment algorithm rather than from a superior critic model or policy class, we conduct additional controlled experiments. As outlined in Figure 4, we evaluate three variants of EDA using different Q-learning approaches and compare these against both diffusion and Gaussian baselines. The experimental results highlight the superiority of diffusion policies and further validate the effectiveness of our proposed method.

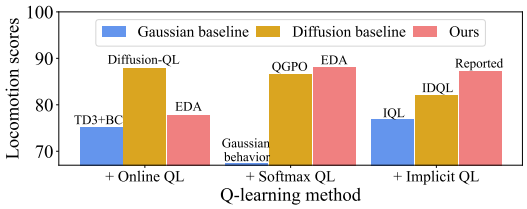
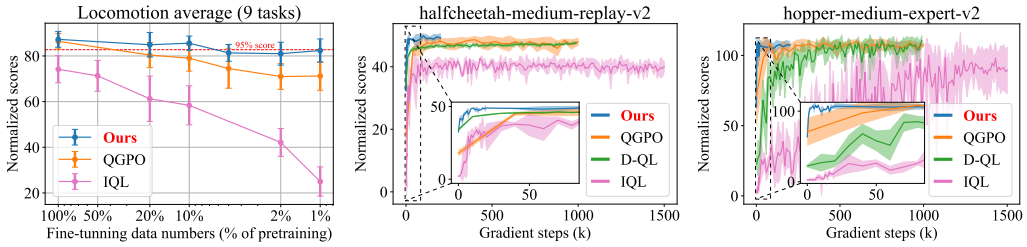


Figure 4: Average performance of EDA combined with different Q-learning methods in Locomotion tasks.



(a) Sample Efficiency (b) Training Efficiency.

Figure 5: Aligning pretrained diffusion behaviors with task Q-functions is fast and data-efficient.

5.2 Fine-tuning Efficiency

The success of the pretraining, fine-tuning paradigm in language models is largely due to its high fine-tuning efficiency, allowing pretrained models to adapt quickly to various downstream tasks. Similarly, we aim to explore data efficiency and training efficiency during EDA's fine-tuning phase.

To investigate EDA's data efficiency, we reduce the training data used for aligning with pretrained Q-functions by randomly excluding a portion of the available dataset (Figure 5 (a)). We compare this with IQL, which uses the same Q-model as EDA but extracts a Gaussian policy via weighted regression. We also compare with QGPO, which shares our pretrained diffusion behavior models

but employs a separate guidance network to augment the behavior model during evaluation, instead of directly fine-tuning the pretrained policy. Experimental results reveal that EDA is significantly more data-efficient than the QGPO and IQL baselines. Notably, EDA maintains about 95% of its performance in locomotion tasks when using only 1% of the Q-labeled data for policy fine-tuning. This even surpasses several baselines that use the full dataset for policy training.

In Figure 5 (b), we plot EDA's performance throughout the fine-tuning phase. EDA rapidly converges in roughly 20K gradient steps, a negligible count compared to the typical 1M steps used for behavior pretraining. Note that behavior modeling and task-oriented Q-definition are largely orthogonal in offline RL. The high fine-tuning efficiency of EDA demonstrates the vast potential of pretraining on large-scale diversified behavior data and quickly adapting to individual downstream tasks.

5.3 Value Optimization v.s. Preference Optimization

A significant difference between EDA and existing preference-based RL methods such as DPO is that EDA is tailored for alignment with scalar Q-values instead of just preference data.

For preference data without explicit Q-labels, we can similarly derive an alignment loss:

$$\max_{\theta} \mathcal{L}_f^{\text{pref}}(\theta) = \mathbb{E}_{t,\boldsymbol{\epsilon}^{1:K},(\boldsymbol{s},\boldsymbol{a}^{1:K})\sim\mathcal{D}^f}\left[\log \frac{e^{\beta[f_\theta^\pi(\boldsymbol{a}_t^w|\boldsymbol{s},t)-f_\phi^\mu(\boldsymbol{a}_t^w|\boldsymbol{s},t)]}}{\sum_{i=1}^K e^{\beta[f_\theta^\pi(\boldsymbol{a}_t^i|\boldsymbol{s},t)-f_\phi^\mu(\boldsymbol{a}_t^i|\boldsymbol{s},t)]}}\right]. \tag{15}$$

Here $\boldsymbol{a}^w$ represents the most preferred action among $\boldsymbol{a}^{1:K}$. In practice, we select $\boldsymbol{a}^w$ as the action with the highest Q-value but abandon the absolute number. We'd like to note that when $K = 2$, the above objective becomes exactly the DPO objective (Eq. 4) in preference learning:

$$\max_{\theta} \mathcal{L}_f^{\text{DPO}}(\theta) = \mathbb{E}_{t,\boldsymbol{\epsilon}^{\{w,l\}},\boldsymbol{s},\boldsymbol{a}^w\succ\boldsymbol{a}^l} \log\sigma\left[\beta[f_\theta^\pi(\boldsymbol{a}_t^w|\boldsymbol{s},t)-f_\phi^\mu(\boldsymbol{a}_t^w|\boldsymbol{s},t)]-\beta[f_\theta^\pi(\boldsymbol{a}_t^l|\boldsymbol{s},t)-f_\phi^\mu(\boldsymbol{a}_t^l|\boldsymbol{s},t)]\right] \tag{16}$$

We ablate different choices of K and compare our proposed value-based alignment with preference methods in 9 D4RL Locomotion tasks (Figure 6). Results show that EDA generally outperforms preference-based alignment approaches. We attribute this improvement to its ability to exploit the Q-value information provided by the pretrained Q-model. Besides, we notice the performance gap between the two methods becomes larger as K increases. This is expected because preference-based optimization greedily follows a single action with the highest Q-value. However, as more action candidates are sampled from the behavior model, the final selected action will have a higher probability of being out-of-behavior-distribution data. This further hurts performance. In contrast, EDA is a softer version of preference learning. This leads to greater tolerance for K.

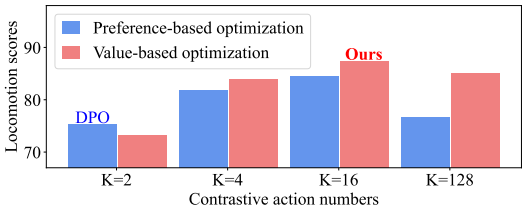


Figure 6: Ablation of action numbers K and optimization methods.

5.4 Ablation Studies

We study the impact of varying temperature β on algorithm performance in Appendix E. We also illustratively compare our proposed diffusion models with other generative models for behavior modeling in 2D settings in Appendix A.

6 Conclusion

We propose Efficient Diffusion Alignment (EDA) for solving offline continuous control tasks. EDA allows leveraging abundant and diverse behavior data to enhance generalization through behavior pretraining and enables rapid adaptation to downstream tasks using minimal annotations. Our experimental results show that EDA exceeds numerous baseline methods in D4RL tasks. It also demonstrates high sample efficiency during the fine-tuning stage. This indicates its vast potential in learning from large-scale behavior datasets and efficiently adapting to individual downstream tasks.

Acknowledgments and Disclosure of Funding

We would like to thank Chengyang Ying and Cheng Lu for discussion. This work was supported by NSFC Projects (Nos. 62350080, 92248303, 92370124, 62276149, 62061136001), BNRist (BNR2022RC01006), Tsinghua Institute for Guo Qiang, and the High Performance Computing Center, Tsinghua University. J. Zhu was also supported by the XPlorer Prize.

References

[1] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B. Tenenbaum, Tommi S. Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *The Eleventh International Conference on Learning Representations*, 2023.

[2] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023.

[3] Huayu Chen, Guande He, Hang Su, and Jun Zhu. Noise contrastive alignment of language models with explicit rewards. *arXiv preprint arXiv:2402.05369*, 2024.

[4] Huayu Chen, Cheng Lu, Zhengyi Wang, Hang Su, and Jun Zhu. Score regularized policy optimization through diffusion behavior. *arXiv preprint arXiv:2310.07297*, 2023.

[5] Huayu Chen, Cheng Lu, Chengyang Ying, Hang Su, and Jun Zhu. Offline reinforcement learning via high-fidelity generative behavior modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

[6] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *Advances in Neural Information Processing Systems*, 2021.

[7] Zibin Dong, Yifu Yuan, Jianye HAO, Fei Ni, Yao Mu, YAN ZHENG, Yujing Hu, Tangjie Lv, Changjie Fan, and Zhipeng Hu. Aligndiff: Aligning diverse human preferences via behavior-customisable diffusion model. In *The Twelfth International Conference on Learning Representations*, 2024.

[8] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. *Advances in Neural Information Processing Systems*, 32, 2019.

[9] Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

[10] 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.

[11] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[12] Seyed Kamyar Seyed Ghasemipour, Dale Schuurmans, and Shixiang Shane Gu. Emaq: Expected-max q-learning operator for simple yet effective offline and online rl. In *International Conference on Machine Learning*, pages 3682–3691. PMLR, 2021.

[13] Wonjoon Goo and Scott Niekum. Know your boundaries: The necessity of explicit behavioral cloning in offline rl. *arXiv preprint arXiv:2206.00695*, 2022.

[14] Ulf Grenander and Michael I Miller. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(4):549–581, 1994.

[15] Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

[16] Haoran He, Chenjia Bai, Kang Xu, Zhuoran Yang, Weinan Zhang, Dong Wang, Bin Zhao, and Xuelong Li. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. *Advances in neural information processing systems*, 36, 2024.

[17] Joey Hejna and Dorsa Sadigh. Inverse preference learning: Preference-based rl without a reward function. *Advances in Neural Information Processing Systems*, 36, 2024.

[18] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[19] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in neural information processing systems*, volume 29, pages 4565–4573, 2016.

[20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

[21] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022.

[22] Bingyi Kang, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan. Efficient diffusion policies for offline reinforcement learning. *arXiv preprint arXiv:2305.20081*, 2023.

[23] Yachen Kang, Diyuan Shi, Jinxin Liu, Li He, and Donglin Wang. Beyond reward: Offline preference-guided policy optimization. *arXiv preprint arXiv:2305.16217*, 2023.

[24] Ilya Kostrikov, Rob Fergus, Jonathan Tompson, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, pages 5774–5783. PMLR, 2021.

[25] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning. In *International Conference on Learning Representations*, 2022.

[26] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 2019.

[27] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2020.

[28] Kimin Lee, Laura Smith, Anca Dragan, and Pieter Abbeel. B-pref: Benchmarking preference-based reinforcement learning. In *Conference on Neural Information Processing Systems Datasets and Benchmarks Track (round 1)*, 2021.

[29] Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

[30] Wenhao Li, Xiangfeng Wang, Bo Jin, and Hongyuan Zha. Hierarchical diffusion for offline decision making. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*. PMLR, 23–29 Jul 2023.

[31] Zengyi Li, Yubei Chen, and Friedrich T Sommer. Learning energy-based models in high-dimensional spaces with multiscale denoising-score matching. *Entropy*, 25(10):1367, 2023.

[32] Cheng Lu, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[33] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *arXiv preprint arXiv:2206.00927*, 2022.

[34] Michaël Mathieu, Sherjil Ozair, Srivatsan Srinivasan, Caglar Gulcehre, Shangtong Zhang, Ray Jiang, Tom Le Paine, Richard Powell, Konrad Żołna, Julian Schrittwieser, et al. Alphastar unplugged: Large-scale offline reinforcement learning. *arXiv preprint arXiv:2308.03526*, 2023.

[35] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[36] Fei Ni, Jianye Hao, Yao Mu, Yifu Yuan, Yan Zheng, Bin Wang, and Zhixuan Liang. Metadiffuser: Diffusion model as conditional planner for offline meta-rl. In *International Conference on Machine Learning*, pages 26087–26105. PMLR, 2023.

[37] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[38] Tim Pearce, Tabish Rashid, Anssi Kanervisto, David Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, et al. Imitating human behaviour with diffusion models. In *Deep Reinforcement Learning Workshop NeurIPS*, 2022.

[39] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

[40] Jan Peters, Katharina Mulling, and Yasemin Altun. Relative entropy policy search. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[41] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[42] Saeed Saremi, Arash Mehrjou, Bernhard Schölkopf, and Aapo Hyvärinen. Deep energy estimator networks. *arXiv preprint arXiv:1805.08306*, 2018.

[43] John Schulman, Barret Zoph, Christina Kim, Jacob Hilton, Jacob Menick, Jiayi Weng, Juan Felipe Ceron Uribe, Liam Fedus, Luke Metz, Michael Pokorny, et al. Chatgpt: Optimizing language models for dialogue. *OpenAI blog*, 2022.

[44] Daniel Shin and Daniel S Brown. Offline preference-based apprenticeship learning. *arXiv preprint arXiv:2107.09251*, 2021.

[45] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265, 2015.

[46] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[47] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

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

[49] 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.

[50] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

[51] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. *arXiv preprint arXiv:2311.12908*, 2023.

[52] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023.

[53] Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, and Nando de Freitas. Critic regularized regression. In *Advances in Neural Information Processing Systems*, 2020.

[54] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[55] Haoran Xu, Xianyuan Zhan, Jianxiong Li, and Honglei Yin. Offline reinforcement learning with soft behavior regularization. *arXiv preprint arXiv:2110.07395*, 2021.

[56] Zhilong Zhang, Yihao Sun, Junyin Ye, Tian-Shuo Liu, Jiaji Zhang, and Yang Yu. Flow to better: Offline preference-based reinforcement learning via preferred trajectory generation. In *The Twelfth International Conference on Learning Representations*, 2024.

[57] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

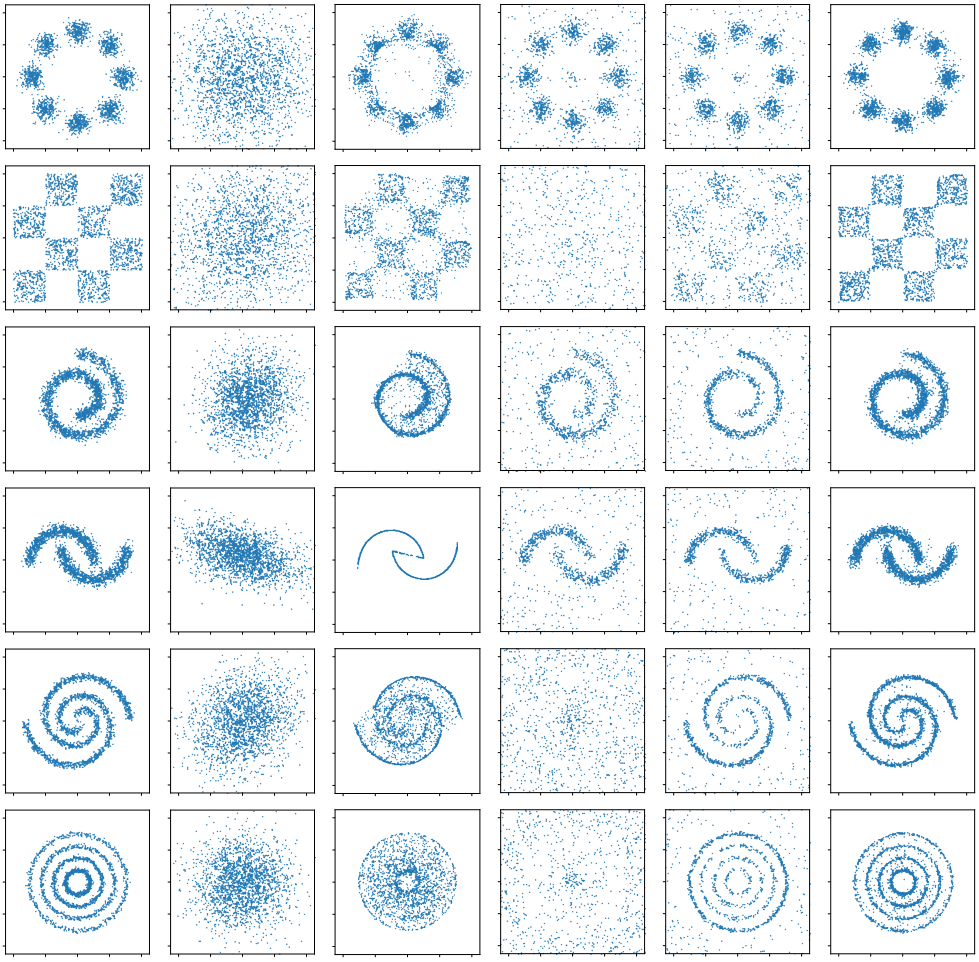
A Comparing Bottleneck Diffusion Models with Energy-Based Models

Our proposed Bottleneck Diffusion Models (BDMs) can be viewed as a diffused variant of Energy-Based Models (EBMs, [8]). Both methods aim to model data distribution's unnormalized log probability:

$$p_\theta(\mathbf{x}) \propto e^{-E_\theta(\mathbf{x})}.$$

Despite their conceptual similarity, the sampling and training approaches differ between diffusion models and EBMs. A prevalent sampling method for EBMs is Langevin MCMC [14], which employs the energy gradient $\nabla_{\mathbf{x}} E_\theta(\mathbf{x})$ to progressively transform Gaussian noise into data samples. Langevin MCMC generally necessitates hundreds or thousands of iterative steps to achieve convergence, a significantly higher number compared with 15-50 steps required by diffusion models (Figure A).

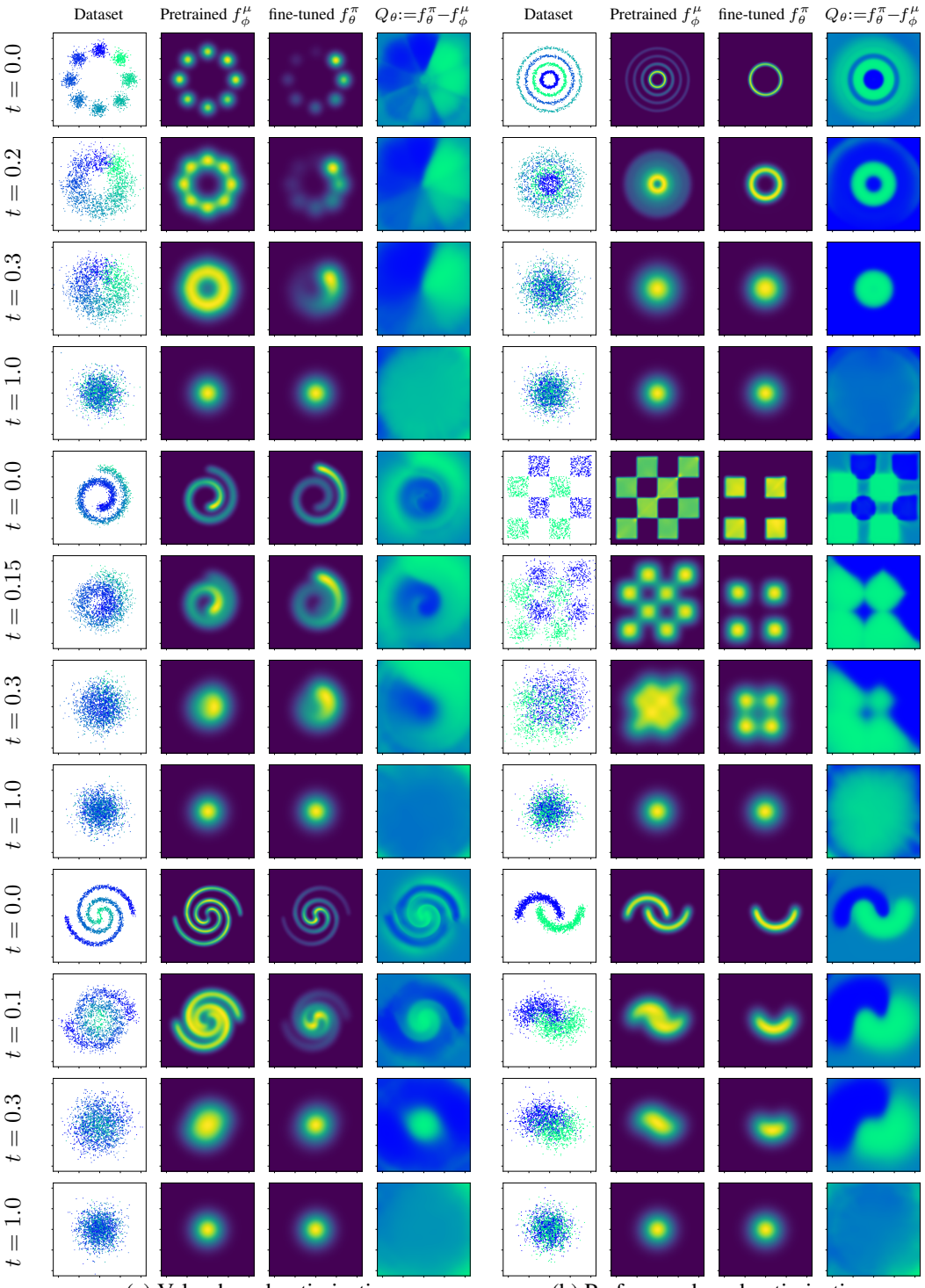
Furthermore, the maximum-likelihood training of EBMs (e.g., Contrastive Divergence [18]) is more computationally expensive, as it involves online data sampling from the model during the training process [48]. To avoid online data sampling, subsequent research [50, 47] has shifted away from directly modeling $E_\theta(\mathbf{x})$ towards developing score-based models defined as $s_\theta(\mathbf{x}) := -\nabla_{\mathbf{x}} E_\theta(\mathbf{x})$. The score-matching objectives utilized in training these score-based models have subsequently been adapted for training diffusion models [49]. Our work is inspired by some prior work [42, 47, 31] that employed such score-matching objectives for training EBMs.



(a) Ground truth (b) Gaussians (c) VAEs (d) EBMs (100 steps) (e) EBMs (1k steps) (f) **Ours (25 steps)**

Figure 7: Comparison of various generative modeling methods in 2D modeling and sampling.

B Complete 2D-bandit Experiment Results



(a) Value-based optimization (b) Preference-based optimization

Figure 8: Illustration of EDA's performance in 2D bandit settings at different diffusion times.

C Theoretical Analysis

In this section, we present the theoretical proof for Proposition 3.1.

Our proof is based on the theoretical framework of Contrastive Energy Prediction (CEP) for diffusion energy guidance [32]. For the ease of readers, we incorporate the relevant theories from their work as lemmas below.

Lemma C.1. *Let $\hat{Q}(\boldsymbol{s}, \boldsymbol{a}) : \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ be a scalar function approximator. Consider the optimization problem*

$$\max_{\hat{Q}} \mathbb{E}_{\mu(\boldsymbol{s}) \prod_{i=1}^{K} \mu(\boldsymbol{a}^i|\boldsymbol{s})} \left[\sum_{k=1}^{K} \frac{e^{Q(\boldsymbol{s},\boldsymbol{a}^k)}}{\sum_{i=1}^{K} e^{Q(\boldsymbol{s},\boldsymbol{a}^i)}} \log \frac{e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}^k)}}{\sum_{i=1}^{K} e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}^i)}} \right]. \tag{17}$$

The solution for Problem 17 satisfies

$$\hat{Q}^*(\boldsymbol{s}, \boldsymbol{a}) = Q(\boldsymbol{s}, \boldsymbol{a}) + C(\boldsymbol{s}),$$

where $C(\boldsymbol{s})$ can be arbitrary scalar functions conditioned on $\boldsymbol{s}$.

Proof. The proof is quite straightforward. Consider two discrete distributions

$$P := \left\{ \frac{e^{Q(\boldsymbol{s},\boldsymbol{a}^1)}}{\sum_{i=1}^{K} e^{Q(\boldsymbol{s},\boldsymbol{a}^i)}}, \frac{e^{Q(\boldsymbol{s},\boldsymbol{a}^2)}}{\sum_{i=1}^{K} e^{Q(\boldsymbol{s},\boldsymbol{a}^2)}} ..., \frac{e^{Q(\boldsymbol{s},\boldsymbol{a}^K)}}{\sum_{i=1}^{K} e^{Q(\boldsymbol{s},\boldsymbol{a}^i)}} \right\}$$

$$\hat{P} := \left\{ \frac{e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}^1)}}{\sum_{i=1}^{K} e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}^i)}}, \frac{e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}^2)}}{\sum_{i=1}^{K} e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}^i)}} ..., \frac{e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}^K)}}{\sum_{i=1}^{K} e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}^i)}} \right\}$$

For any $\boldsymbol{s}$ and any $\boldsymbol{a}^{1:K}$, we have

$$\begin{aligned}
&\mathbb{E}_{\mu(\boldsymbol{s}) \prod_{i=1}^{K} \mu(\boldsymbol{a}^i|\boldsymbol{s})} \left[\sum_{k=1}^{K} \frac{e^{Q(\boldsymbol{s},\boldsymbol{a}^k)}}{\sum_{i=1}^{K} e^{Q(\boldsymbol{s},\boldsymbol{a}^i)}} \log \frac{e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}^k)}}{\sum_{i=1}^{K} e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}^i)}} \right] \\
=&\mathbb{E}_{\mu(\boldsymbol{s}) \prod_{i=1}^{K} \mu(\boldsymbol{a}^i|\boldsymbol{s})} - D_{\mathrm{KL}}(P||\hat{P}) - H(P) \\
\leq&\mathbb{E}_{\mu(\boldsymbol{s}) \prod_{i=1}^{K} \mu(\boldsymbol{a}^i|\boldsymbol{s})} - H(P)
\end{aligned}$$

According to the properties of KL divergence, the equality holds if and only if $P = \hat{P}$ for any $\boldsymbol{s}$ and $\boldsymbol{a}^{1:K}$. This implies that

$$\hat{Q}^*(\boldsymbol{s}, \boldsymbol{a}) = Q(\boldsymbol{s}, \boldsymbol{a}) + C(\boldsymbol{s}),$$

constantly holds. □

Lemma C.2. *Let $\hat{Q}(\boldsymbol{s}, \boldsymbol{a}_t, t) : \mathcal{A} \times \mathcal{S} \times \mathbb{R} \to \mathbb{R}$ be a scalar function approximator. $p(\boldsymbol{a}_t|\boldsymbol{a}, t)$ is any conditional transition probability. Consider the optimization problem*

$$\max_{\hat{Q}} \mathbb{E}_{\mu(\boldsymbol{s}) \prod_{i=1}^{K} \mu(\boldsymbol{a}^i|\boldsymbol{s}) p(\boldsymbol{a}_t^i|\boldsymbol{a}^i,t)} \left[\sum_{k=1}^{K} e^{Q(\boldsymbol{s},\boldsymbol{a}^k)} \log \frac{e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}_t^k,t)}}{\sum_{i=1}^{K} e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}_t^i,t)}} \right]. \tag{18}$$

The solution for Problem 18 satisfies

$$\hat{Q}^*(\boldsymbol{s}, \boldsymbol{a}_t, t) = \log \mathbb{E}_{\mu_t(\boldsymbol{a}|\boldsymbol{a}_t,\boldsymbol{s},t)} e^{Q(\boldsymbol{s},\boldsymbol{a})} + C(\boldsymbol{s}),$$

where $\mu_t(\boldsymbol{a}|\boldsymbol{a}_t, \boldsymbol{s}, t) = \mu(\boldsymbol{a}|\boldsymbol{s}) p(\boldsymbol{a}_t|\boldsymbol{a}, t) / \mu_t(\boldsymbol{a}_t|\boldsymbol{s}, t)$ is the posterior action distribution, $C(\boldsymbol{s})$ can be arbitrary scalar functions conditioned on $\boldsymbol{s}$.

Proof.

$$
\begin{aligned}
&\mathbb{E}_{\mu(\boldsymbol{s})\prod_{i=1}^{K}\mu(\boldsymbol{a}^i|\boldsymbol{s})p(\boldsymbol{a}_t^i|\boldsymbol{a}^i,t)}\left[\sum_{k=1}^{K}e^{Q(\boldsymbol{s},\boldsymbol{a}^k)}\log\frac{e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}_t^k,t)}}{\sum_{i=1}^{K}e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}_t^i,t)}}\right]\\
=&\mathbb{E}_{\mu(\boldsymbol{s})\prod_{i=1}^{K}\mu_t(\boldsymbol{a}_t^i|\boldsymbol{s})\mu_t(\boldsymbol{a}^i|\boldsymbol{a}_t^i,\boldsymbol{s},t)}\left[\sum_{k=1}^{K}e^{Q(\boldsymbol{s},\boldsymbol{a}^k)}\log\frac{e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}_t^k,t)}}{\sum_{i=1}^{K}e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}_t^i,t)}}\right].\\
=&\mathbb{E}_{\mu(\boldsymbol{s})\prod_{i=1}^{K}\mu_t(\boldsymbol{a}_t^i|\boldsymbol{s})}\left[\sum_{k=1}^{K}\mathbb{E}_{\mu_t(\boldsymbol{a}^k|\boldsymbol{a}_t^k,\boldsymbol{s},t)}e^{Q(\boldsymbol{s},\boldsymbol{a}^k)}\log\frac{e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}_t^k,t)}}{\sum_{i=1}^{K}e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}_t^i,t)}}\right].\\
=&\mathbb{E}_{\mu(\boldsymbol{s})\prod_{i=1}^{K}\mu'_t(\boldsymbol{a}_t^i|\boldsymbol{s})}\left[\sum_{k=1}^{K}\frac{\mathbb{E}_{\mu_t(\boldsymbol{a}^k|\boldsymbol{a}_t^k,\boldsymbol{s},t)}e^{Q(\boldsymbol{s},\boldsymbol{a}^k)}}{\sum_{i=1}^{K}\mathbb{E}_{\mu_t(\boldsymbol{a}^i|\boldsymbol{a}_t^i,\boldsymbol{s},t)}e^{Q(\boldsymbol{s},\boldsymbol{a}^i)}}\log\frac{e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}_t^k,t)}}{\sum_{i=1}^{K}e^{\hat{Q}(\boldsymbol{s},\boldsymbol{a}_t^i,t)}}\right].
\end{aligned}
$$

According to Lemma C.1, for any state $\boldsymbol{a}$ and any diffused action $\boldsymbol{a}_t$ at time t, the optimal solution satisfies

$$\hat{Q}^*(\boldsymbol{s},\boldsymbol{a}_t,t)=\log\mathbb{E}_{\mu_t(\boldsymbol{a}|\boldsymbol{a}_t,\boldsymbol{s},t)}e^{Q(\boldsymbol{s},\boldsymbol{a})}+C(\boldsymbol{s}).$$

□

Lemma C.3. *Consider the behavior distribution $\mu(\boldsymbol{a}|\boldsymbol{s})$ and the policy distribution $\pi^*(\boldsymbol{a}|\boldsymbol{s})\propto\mu(\boldsymbol{a}|\boldsymbol{s})e^{Q(\boldsymbol{s},\boldsymbol{a})}$. Their diffused distribution at time t are both defined by the forward diffusion process (Eq. 5). Let $p(\boldsymbol{a}_t|\boldsymbol{a},t):=\mathcal{N}(\boldsymbol{a}_t|\alpha_t\boldsymbol{a},\sigma_t^2\boldsymbol{I})$, such that*

$$\mu_t(\boldsymbol{a}_t|\boldsymbol{s},t)=\int\mathcal{N}(\boldsymbol{a}_t|\alpha_t\boldsymbol{a},\sigma_t^2\boldsymbol{I})\mu(\boldsymbol{a}|\boldsymbol{s},t)\mathrm{d}\boldsymbol{a},$$

and

$$\pi_t^*(\boldsymbol{a}_t|\boldsymbol{s},t)=\int\mathcal{N}(\boldsymbol{a}_t|\alpha_t\boldsymbol{a},\sigma_t^2\boldsymbol{I})\pi^*(\boldsymbol{a}|\boldsymbol{s},t)\mathrm{d}\boldsymbol{a}.$$

Then the relationship between π_t^* and μ_t can be derived as

$$\pi_t^*(\boldsymbol{a}_t|\boldsymbol{s},t)\propto\mu_t(\boldsymbol{a}_t|\boldsymbol{s},t)e^{Q_t(\boldsymbol{s},\boldsymbol{a}_t,t)},$$

where $Q_t(\boldsymbol{s},\boldsymbol{a}_t,t):=\log\mathbb{E}_{\mu_t(\boldsymbol{a}|\boldsymbol{a}_t,\boldsymbol{s},t)}e^{Q(\boldsymbol{s},\boldsymbol{a})}$.

Proof. According to the definition, we have

$$
\begin{aligned}
\pi_t^*(\boldsymbol{a}|\boldsymbol{s})&=\int\mathcal{N}(\boldsymbol{a}_t|\alpha_t\boldsymbol{a},\sigma_t^2\boldsymbol{I})\pi^*(\boldsymbol{a}|\boldsymbol{s},t)\mathrm{d}\boldsymbol{a}\\
&=\int\mathcal{N}(\boldsymbol{a}_t|\alpha_t\boldsymbol{a},\sigma_t^2\boldsymbol{I})\mu(\boldsymbol{a}|\boldsymbol{s})\frac{e^{Q(\boldsymbol{s},\boldsymbol{a})}}{Z(\boldsymbol{s})}\mathrm{d}\boldsymbol{a}\\
&=\int p(\boldsymbol{a}_t|\boldsymbol{a},t)\mu(\boldsymbol{a}|\boldsymbol{s})\frac{e^{Q(\boldsymbol{s},\boldsymbol{a})}}{Z(\boldsymbol{s})}\mathrm{d}\boldsymbol{a}\\
&=\int\mu_t(\boldsymbol{a}|\boldsymbol{a}_t,\boldsymbol{s},t)\mu_t(\boldsymbol{a}_t|\boldsymbol{s},t)\frac{e^{Q(\boldsymbol{s},\boldsymbol{a})}}{Z(\boldsymbol{s})}\mathrm{d}\boldsymbol{a}\\
&=\frac{1}{Z(\boldsymbol{s})}\mu_t(\boldsymbol{a}_t|\boldsymbol{s},t)\int\mu_t(\boldsymbol{a}|\boldsymbol{a}_t,\boldsymbol{s},t)e^{Q(\boldsymbol{s},\boldsymbol{a})}\mathrm{d}\boldsymbol{a}\\
&\propto\mu_t(\boldsymbol{a}_t|\boldsymbol{s},t)e^{Q_t(\boldsymbol{s},\boldsymbol{a}_t,t)}
\end{aligned}
$$

□

Proposition C.4. *Let f_θ^* be the optimal solution of Problem 14 and $\pi_{t,\theta}^*\propto e^{f_\theta^*}$ be the optimal diffusion policy. Assuming unlimited model capacity and data samples, we have the following results:*

(a) Optimality Guarantee. *At time $t=0$, the learned policy π_θ^* converges to the optimal target policy.*

$$\pi_\theta^*(\boldsymbol{a}|\boldsymbol{s})=\pi_{t=0,\theta}^*(\boldsymbol{a}|\boldsymbol{s})\propto\mu_\phi(\boldsymbol{a}|\boldsymbol{s})e^{Q(\boldsymbol{s},\boldsymbol{a})/\beta}$$

(b) Diffusion Consistency. *At time $t > 0$, $\pi_{t>0,\theta}$ models the diffused distribution of π^*_θ:*

$$\pi^*_{t,\theta}(\boldsymbol{a}|\boldsymbol{s},t) = \int \mathcal{N}(\boldsymbol{a}_t|\alpha_t\boldsymbol{a}, \sigma_t^2\boldsymbol{I})\pi^*_\theta(\boldsymbol{a}_0|\boldsymbol{s})\mathrm{d}\boldsymbol{a}_0$$

asymptotically holds when $K \to \infty$ and $\beta = 1$, satisfying the definition of diffusion process (Eq. 6).

Proof. We first rewrite Problem 14 below:

$$\max_\theta \mathcal{L}_f(\theta) = \mathbb{E}_{\mu(\boldsymbol{s})\prod_{i=1}^K \mu(\boldsymbol{a}^i|\boldsymbol{s})p(\boldsymbol{a}_t^i|\boldsymbol{a}^i,t)}\left[\sum_{k=1}^K \frac{e^{Q(\boldsymbol{s},\boldsymbol{a}^k)}}{\sum_{i=1}^K e^{Q(\boldsymbol{s},\boldsymbol{a}^i)}} \log \frac{e^{\beta[f_\theta^\pi(\boldsymbol{a}_t^k|\boldsymbol{s},t) - f_\phi^\mu(\boldsymbol{a}_t^k|\boldsymbol{s},t)]}}{\sum_{i=1}^K e^{\beta[f_\theta^\pi(\boldsymbol{a}_t^i|\boldsymbol{s},t) - f_\phi^\mu(\boldsymbol{a}_t^i|\boldsymbol{s},t)]}}\right]. \tag{19}$$

(a) Optimality Guarantee. At time $t = 0$, we have $p(\boldsymbol{a}_t|\boldsymbol{a},t) = \mathcal{N}(\boldsymbol{a}_t|\alpha_t\boldsymbol{a}, \sigma_t^2\boldsymbol{I}) = \mathcal{N}(\boldsymbol{a}_t|\boldsymbol{a}, 0\boldsymbol{I})$ such that $\boldsymbol{a}_t = \boldsymbol{a}$.

Define $\hat{Q}^*(\boldsymbol{s},\boldsymbol{a}) := \beta[f_\theta^\pi(\boldsymbol{a}_t^k|\boldsymbol{s}, t=0) - f_\phi^\mu(\boldsymbol{a}_t^k|\boldsymbol{s}, t=0)]$. Since we assume unlimited model capacity for f_θ, $\hat{Q}^*$ can be arbitrary scalar functions. Lemma C.1 can then be applied:

$$\begin{aligned}
\pi^*_\theta(\boldsymbol{a}|\boldsymbol{s}) =& \pi^*_{t=0,\theta}(\boldsymbol{a}|\boldsymbol{s}) \\
\propto& e^{f_\theta^\pi(\boldsymbol{a}_t^i|\boldsymbol{s},t=0)} \\
=& e^{f_\phi^\mu(\boldsymbol{a}_t^i|\boldsymbol{s},t=0) + \hat{Q}^*(\boldsymbol{s},\boldsymbol{a})/\beta} \\
=& e^{f_\phi^\mu(\boldsymbol{a}_t^i|\boldsymbol{s},t=0)} e^{[Q(\boldsymbol{s},\boldsymbol{a}) + C(\boldsymbol{s})]/\beta} \\
\propto& \mu_\phi(\boldsymbol{a}|\boldsymbol{s}) e^{Q(\boldsymbol{s},\boldsymbol{a})/\beta}
\end{aligned}$$

(b) Diffusion Consistency. When $K \to \infty$, we $\sum_{i=1}^K e^{Q(\boldsymbol{s},\boldsymbol{a}^i)} = \mathbb{E}_\mu(\boldsymbol{a}|\boldsymbol{s})e^{Q(\boldsymbol{s},\boldsymbol{a})}$ becomes constant and can be removed. Set $\beta = 1$, the optimization problem equation 14 becomes

$$\max_\theta \mathcal{L}_f(\theta) = \mathbb{E}_{\mu(\boldsymbol{s})\prod_{i=1}^K \mu(\boldsymbol{a}^i|\boldsymbol{s})p(\boldsymbol{a}_t^i|\boldsymbol{a}^i,t)}\left[\sum_{k=1}^K e^{Q(\boldsymbol{s},\boldsymbol{a}^k)} \log \frac{e^{\,[f_\theta^\pi(\boldsymbol{a}_t^k|\boldsymbol{s},t) - f_\phi^\mu(\boldsymbol{a}_t^k|\boldsymbol{s},t)]}}{\sum_{i=1}^K e^{[f_\theta^\pi(\boldsymbol{a}_t^i|\boldsymbol{s},t) - f_\phi^\mu(\boldsymbol{a}_t^i|\boldsymbol{s},t)]}}\right]. \tag{20}$$

We can then similarly apply Lemma C.2 and get

$$\log \frac{\pi^*_{t,\theta}(\boldsymbol{a}_t|\boldsymbol{s},t)}{\mu_t(\boldsymbol{a}_t|\boldsymbol{s},t)} = f_\theta^\pi(\boldsymbol{a}_t|\boldsymbol{s},t) - f_\phi^\mu(\boldsymbol{a}_t|\boldsymbol{s},t) = \log \mathbb{E}_{\mu_t(\boldsymbol{a}|\boldsymbol{a}_t,\boldsymbol{s},t)} e^{Q(\boldsymbol{s},\boldsymbol{a})} + Z(\boldsymbol{s})$$

$$\pi^*_{t,\theta}(\boldsymbol{a}_t|\boldsymbol{s},t) \propto \mu_t(\boldsymbol{a}_t|\boldsymbol{s},t)\mathbb{E}_{\mu_t(\boldsymbol{a}|\boldsymbol{a}_t,\boldsymbol{s},t)} e^{Q(\boldsymbol{s},\boldsymbol{a})}$$

According to Lemma C.3, we have

$$\pi^*_{t,\theta}(\boldsymbol{a}|\boldsymbol{s},t) = \int \mathcal{N}(\boldsymbol{a}_t|\alpha_t\boldsymbol{a}, \sigma_t^2\boldsymbol{I})\pi^*_\theta(\boldsymbol{a}_0|\boldsymbol{s})\mathrm{d}\boldsymbol{a}_0$$

□

D Implementation Details for D4RL Tasks

We use NVIDIA A40 GPU cards to run all experiments.

Behavior pretraining. For pretraining the bottleneck diffusion model, we extract a reward-free behavior dataset $\{\boldsymbol{s}, \boldsymbol{a}\}$ from $\mathcal{D}^\mu := \{\boldsymbol{s}, \boldsymbol{a}, r, \boldsymbol{s}'\}$. We adopt the model architecture used by IDQL [15] and SRPO [4] but sum up the final $|\mathcal{A}|$-dimensional output to form a scalar network. The resulting model is basically a 6-layer MLP with residual connections, layer normalizations, and dropout regularization. We train the behavior network for 1M steps to ensure convergence. The batch size is 2048. The optimizer is Adam with a learning rate of 3e-4. We adopt default VPSDE [49] hyperparameters as the diffusion data perturbation method.

Constructing alignment dataset. First, we leverage $\mathcal{D}^\mu$ and use existing methods such as IQL [25] to learn a critic network that will later be utilized for data annotation. Then, for a random portion of state $\boldsymbol{s}$ in $\mathcal{D}^\mu$, we leverage the pretrained behavior model to generate $K = 16$ action samples. The original action in $\mathcal{D}^\mu$ is thrown away. We use the critic model to annotate each state-action pair and store them together as the alignment dataset $\mathcal{D}^f := \{\boldsymbol{s}, \boldsymbol{a}^{1:K}, Q(\boldsymbol{s}, \boldsymbol{a}^k)|_{k\in 1:K}\}$. We use 2-layer MLPs with 256 hidden states. Critic training details is exactly the same with previous work [25]. Other critic learning methods used in ablation studies are also consistent with respective prior work [15, 32].

Policy fine-tuning. The policy network is initialized to be the behavior network. Throughout the training, we fix the behavior model and only optimize the policy model. Behavior weights are frozen. The optimizer is Adam and the learning rate is 5e-5. All policy models are trained for 200k gradient steps though we observe convergence at 20K steps in most tasks. We do not employ dropout regularization during fine-tuning because we find it harms performance. For the temperature coefficient, we sweep over $\beta \in \{0.1, 0.2, 0.3, 0.5, 0.8, 1.0, 2.0\}$ (Figure 9&10). Similarly to [11, 15, 5, 15], we find the performance can be improved by adopting a rejection sampling technique. We select the action with the highest Q-value among 4 candidates during evaluation. We do not use such techniques in experimental plots (Figure 5&11) to better reflect policy improvement. We use 20 test seeds in all experiments and report numbers at the end of training. We train all experiments independently with 5 seeds in our main experiments and 3-5 seeds in ablation studies.

E Additional Experiment Results

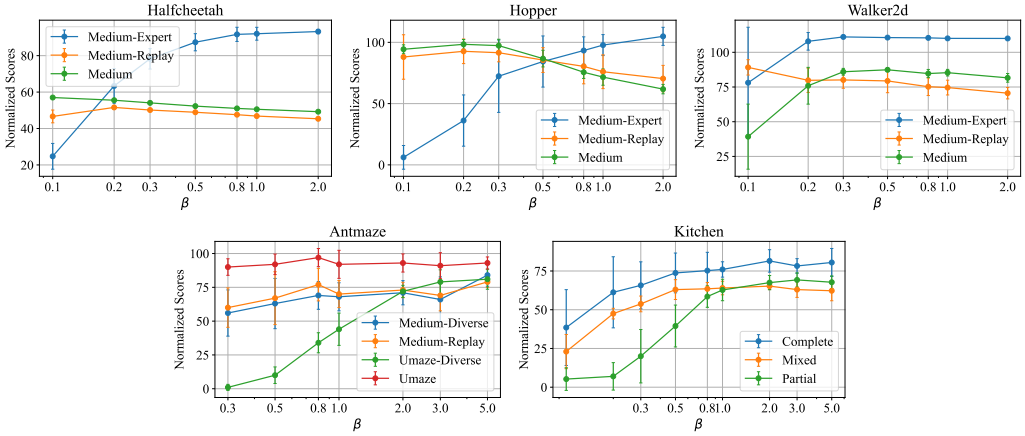


Figure 9: Ablation of the temperature coefficient β in D4RL benchmarks.

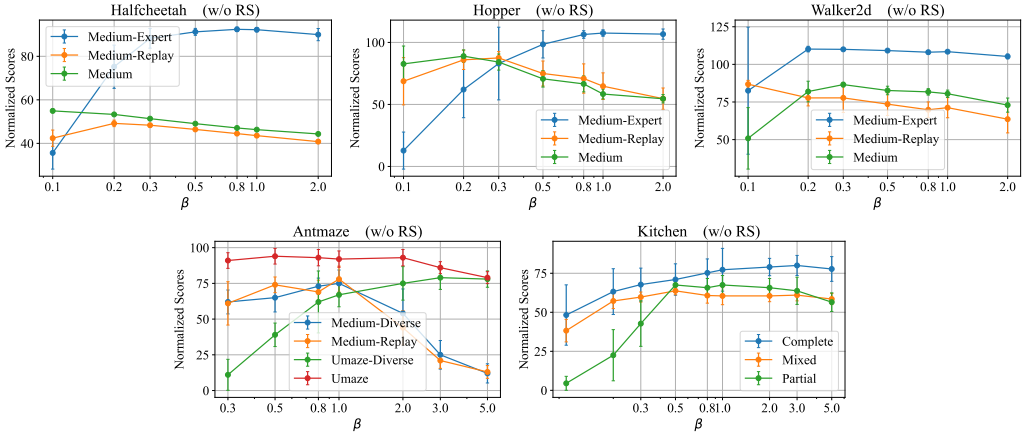


Figure 10: Ablation of the temperature coefficient β without rejection sampling.

Halfcheetah	Medium-expert 2.0	Medium 0.1	Medium-Replay 0.2
Hopper	Medium-expert 2.0	Medium 0.2	Medium-Replay 0.2
Walker2d	Medium-expert 0.3	Medium 0.5	Medium-Replay 0.1
AntMaze	Umaze 0.5	Umanze-diverse 5.0	Medium (both) 1.0
Kitchen	Complete 2.0	Mixed 3.0	Partial 3.0

Table 2: Temperature coefficient β for every individual task.

F Additional Training Curves

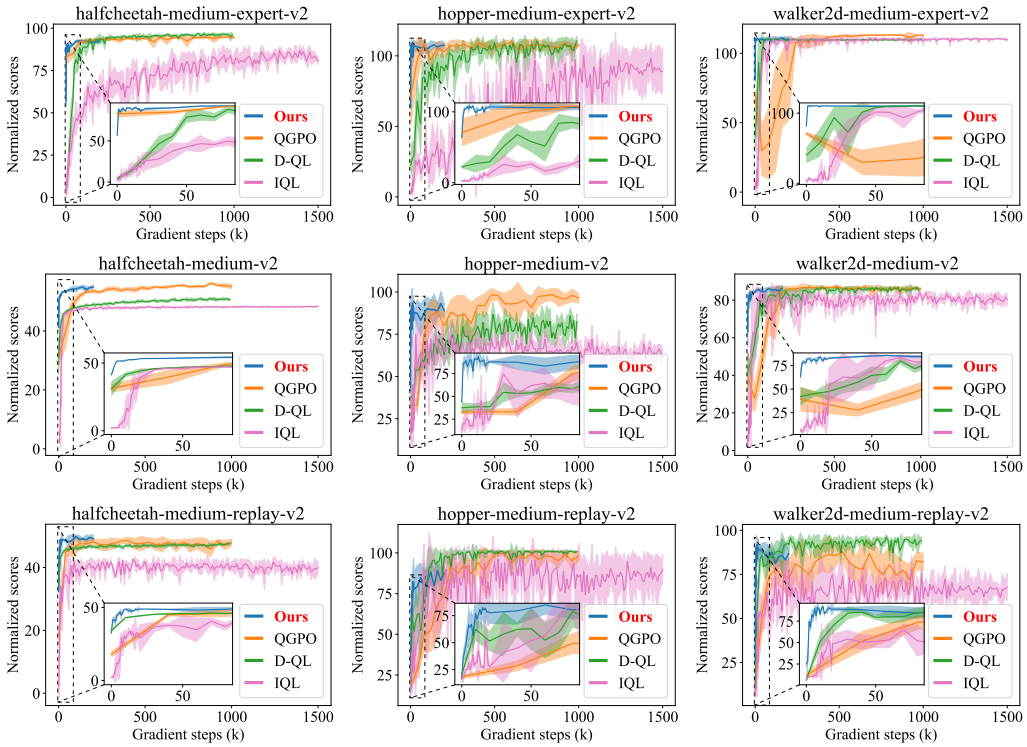


Figure 11: Training curves of EDA (ours) and several baselines.

G Reproducibility

To ensure that our work is reproducible, we submit the source code as supplementary material. The code will go open-source upon publication. Reported experimental numbers are averaged under multiple random seeds. We provide implementation details of our work in Appendix D.

H Limitations and Broader Impacts

Limitations. EDA employs a diffusion policy, which is a natural fit for continuous control problems. However, it cannot be directly applied in decision-making tasks with a discrete action space. Additionally, EDA places restrictions on the diffusion model design. The diffusion policy has to be the gradient of a scalar neural network with respect to action inputs to enable direct density estimation. Lastly,as task reward is human-defined and varies from task to task, achieving optimal performance with EDA in downstream tasks requires tailored network architecture design and hyperparameter tuning. How to ensure that the same set of algorithm parameters can be applicable across various tasks remains a topic worthy of further research.

Broader Impacts. EDA is a rather theoretical paper and all experiments are done in simulation environments. However, its further application could assist the deployment of highly optimized AI systems in critical real-world applications, such as autonomous driving or healthcare robots. This could have unintended consequences if the systems fail or produce incorrect results. Robust testing and validation are essential to mitigate these risks. There is also a risk that EDA could be misused for malicious purposes. For instance, optimized AI policies could be employed in autonomous systems for surveillance or other invasive activities without proper regulation and oversight.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the paper's contributions and scope

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Appendix H

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors

should reflect on how these assumptions might be violated in practice and what the implications would be.

- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Appendix C.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: yes we have.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same

dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Source code is in supplemental material. We will open-source it upon publication.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Appendix D

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: See Sec. 5.1 and Appendix E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix D and Sec. 5.2

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

Justification: Yes.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Appendix H.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This paper is highly theoretical, there's little risk the released training code is misused.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Yes, we have.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Yes, we have.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.