Imitation from Observations with Trajectory-Level Generative Embeddings

Yongtao Qu
UNC Chapel Hill
Chapel Hill, NC 27599
yongtao@unc.edu

Shangzhe Li
UNC Chapel Hill
Chapel Hill, NC 27599
shangzhe@unc.edu

Weitong Zhang
UNC Chapel Hill
Chapel Hill, NC 27599
weitongz@unc.edu

Abstract

We consider the offline imitation learning from observations (LfO), where expert demonstrations are scarce and contain only state observations, and the suboptimal policy is far from expert behavior. In this regime, many existing imitation learning approaches struggle to extract useful information from imperfect data since they impose strict support constraints and rely on brittle one-step models. To tackle this challenge, we propose **T**rajectory-level **G**enerative **E**mbedding (TGE) for offline LfO. TGE constructs a dense, smooth surrogate reward by using particle based entropy estimation to maximize the log-likelihood of expert trajectories in the latent space of a temporal diffusion model trained on offline suboptimal data. By leveraging the structured geometry of the learned diffusion embedding, TGE captures long-horizon temporal dynamics and effectively bridges the gap under severe support mismatch, ensuring a robust learning signal even when offline data is distributionally distinct from the expert. Empirically, the proposed approach consistently matches or outperforms prior offline LfO methods across a range of D4RL locomotion and manipulation benchmarks.

1 Introduction

Imitation learning (IL) has demonstrated remarkable success in enabling agents to acquire complex behaviors across various domains, from robotic manipulation [Florence et al., 2021, Chi et al., 2023, Brohan et al., 2023] to autonomous navigation [Codevilla et al., 2018, Bansal et al., 2019]. While standard IL methods can effectively recover high-performance policies by supervising explicit action labels, applying these techniques to real-world scenarios poses substantial challenges. A primary bottleneck is the absence of explicit action information in many natural data sources, such as video demonstrations or third-person recordings, which precludes direct supervision of state-action mappings [Torabi et al., 2018, Stadie et al., 2017, Liu et al., 2018]. This has led to the emergence of Learning from Observations (LfO), where the agent must infer desirable behaviors from state-only expert trajectories.

To recover the underlying actions, many of the existing LfO methods attempt to estimate the inverse dynamics or auxiliary inference mechanisms [Hanna and Stone, 2017, Kidambi et al., 2020]. However, an equally critical challenge lies in the quality of the offline data itself. As large-scale offline datasets inevitably contain suboptimal or inconsistent behaviors arising from human error or task ambiguity, directly applying behavioral cloning to such datasets often leads to poor performance. Effectively leveraging these imperfect demonstrations therefore requires algorithms that can robustly filter noise and distill informative behavioral priors, without being misled by non-expert trajectories.

Existing LfO approaches face two structural limitations: surrogate-reward methods based on adversarial discriminators are unstable over long horizons [Zolna et al., 2020, Ross et al., 2011] and distribution-matching methods rely on density ratios [Ho and Ermon, 2016, Ma et al., 2022] that

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

degenerate when offline data has limited support over the expert distribution (see Sec. 2.1). These limitations raise a central question for learning from observations:

How can we extract a reliable imitation signal from imperfect offline data under support mismatch?

To address these limitations, we propose **Trajectory-level Generative Embeddings (TGE)**, a novel framework for offline LfO that bypasses the structural reliance on support coverage inherent in density ratio estimation. Instead of relying on unstable density ratios, TGE leverages the smooth representation learned by a trajectory-based diffusion model such as diffuser [Janner et al., 2022]. By encoding trajectories into this generative latent space, we construct a dense surrogate reward signal that bridges regions of sparse support. Specifically, we employ a logarithmic distance kernel for reward estimation inspired by the particle-based entropy estimation [Liu and Abbeel, 2021, Mutti et al., 2020, Singh et al., 2003], which ensures that maximizing the surrogate reward using standard offline RL algorithms, such as IQL [Kostrikov et al., 2021] or ReBRAC [Tarasov et al., 2023], is essentially maximizing the log-likelihood of the expert state distribution. Notably, this design of the surrogate reward also ensures an informative signal throughout the suboptimal data, preventing the reward collapse issues common in prior distance-based methods.

Our main contributions are summarized as follows:

- We introduce a trajectory-level representation learning approach based on a generative planner, demonstrating that the resulting latent embeddings are naturally separable and can distinguish between suboptimal and expert-like data in offline datasets without explicit discriminative training. To our knowledge, this is the first use of representations from generative planner in imitation learning.
- We derive a reward formulation defined over the trajectory-level latent embedding, which approximately minimizes the particle-based cross-entropy between the policy and expert occupancies, enabling the use of any offline RL methods for imitation learning with observation-only demonstration.
- Empirically, we identify and analyze a structural failure mode shared by existing commonly used DICE-based methods under support mismatch, and show empirically that TGE remains robust precisely in the regimes where density-ratio methods collapse, matching or outperforming state-of-the-art offline LfO baselines across diverse benchmarks.

2 Related work

2.1 Offline imitation learning

Offline imitation learning (IL) aims to recover an expert policy from a fixed dataset of demonstrations without access to online environment interactions. Behavioral Cloning (BC) [Pomerleau, 1989], treats this as a supervised regression problem but suffers from distribution shift and compounding errors on out-of-distribution states [Ross et al., 2011]. Distribution-matching approaches mitigate by minimizing the divergence between the learner and expert occupancy measures [Kostrikov et al., 2020, Garg et al., 2021].

Learning from Observations (LfO) arises as a more challenging subset where the expert demonstrations lack action labels. Offline LfO approaches generally fall into two distinct paradigms:

Occupancy Matching. This class of methods formulates imitation as an optimization problem, solving for the state-occupancy ratio $w(s) \approx d^{\pi_E}(s)/d^{\mathcal{D}_\mu}(s)$ between the expert and offline suboptimal data distributions. Methods like SMODICE [Ma et al., 2022] and LobsDICE [Kim et al., 2022], which minimize the KL-divergence, are often sensitive to the distribution of the offline buffer. Recent variants like PW-DICE [Yan et al., 2024] introduce Wasserstein geometry to relax the strict support constraints, while DILO [Sikchi et al., 2025] utilizes a dual formulation of the χ^2-divergence to improve stability. IOSTOM [Pham et al., 2025] reformulates the problem as state-transition occupancy matching, avoiding the discriminator pretraining required by earlier DICE variants. Despite these advances, these methods fundamentally rely on estimating density ratios, which can become ill-conditioned when the offline behavioral data has limited support to the expert policy.

Surrogate Reward Learning. Another line of work is to learn a surrogate reward from data, and then optimizes a policy using offline RL algorithms. Broadly speaking, this paradigm extends beyond imitation learning and is central to unsupervised reinforcement learning [Laskin et al., 2021], where methods typically derive intrinsic pseudo-rewards to guide policy acquisition or environment exploration in the absence of external supervision. In the context of offline LfO,

ORIL [Zolna et al., 2020] apply adversarial learning to train a discriminator-based reward, which is notoriously unstable and often produce signals that are difficult to optimize. A more recent non-adversarial alternative, SEABO [Lyu et al., 2024], constructs rewards from nearest-neighbor distances to expert demonstrations via a KD-tree, sidestepping discriminator instability but relying on a fixed observation-space metric that lacks temporal context. TGE falls into this surrogate reward paradigm but circumvents both limitations by deriving a dense, theoretically grounded reward from the latent space of a trajectory-level generative planner trained on the offline dataset.

2.2 Diffusion models in reinforcement learning

Diffusion models have been widely adopted in RL as policies, trajectory planners, value regularizers, and world models [Janner et al., 2022, Ajay et al., 2022, Chi et al., 2025, Wang et al., Ding et al., 2024, Ho et al., 2020, Alonso et al., 2024]. In the context of diffusion models in RL, our work focuses on a distinct application in using diffusion models for metric learning. While methods like Stable Rep [Tian et al., 2023] or Diffusion Reward [Huang et al., 2024] similarly derive learning signals from generative models, they typically rely on static visual semantics from video or image models. In contrast, TGE exploits the trajectory-level latent embedding of a temporal diffusion encoder to capture system dynamics, augmenting the reward-free offline dataset that enables robust imitation without the computational cost of generative sampling during downstream offline RL training.

2.3 Representation learning in reinforcement learning

Representation learning is central to scaling RL to high-dimensional observation spaces by extracting compact embeddings that filter task-irrelevant noise. Early approaches largely relied on reconstruction-based auxiliary tasks with world models or autoencoders to compress sensory inputs into latent states that capture the factors of the environment [Ha and Schmidhuber, 2018]. Subsequent research adopted contrastive learning and bi-simulation metrics to group states with similar transition dynamics [Laskin et al., 2020, Zhang et al., 2021] to improve the model robustness to task-irrelevant details. However, these approaches typically learn the representation for single-stage observation. To tackle this, we instead leverage the representation learned from a trajectory diffusion model (i.e., Diffuser [Janner et al., 2022]). These trajectory-level representations provide the temporal context necessary to distinguish behaviors that are visually similar but possess divergent underlying dynamics. Recent works have shown that diffusion models can be viewed as denoising autoencoders where intermediate features serve as robust representations for downstream tasks [Fuest et al., 2024], suggesting that diffusion training induces semantically meaningful features beyond image generation.

3 Preliminaries

Markov Decision Process. We consider the discounted Markov Decision Process (MDP) setting, defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, respectively. The transition kernel $P(s'|s, a)$ specifies the probability of transitioning to next state s' given the current state s and action a with the reward function defined by $r(s, a)$ and $\gamma \in [0, 1)$ is a discount factor. A policy $\pi(a|s)$ defines a distribution over actions conditioned on the state, and the objective of reinforcement learning is to identify a policy that maximizes the expected discounted return $J(\pi) = \mathbb{E}_{\pi,P}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$.

Offline Imitation Learning from Observations. In imitation learning from observation (LfO), the agent is given a small expert dataset containing only state observations, without any expert actions or reward annotations. In addition, our work focuses on the offline learning, where the agent cannot directly interact with the environment. Instead, we assume the agent can access an offline dataset composed of suboptimal state–action trajectories, which provides the necessary coverage for learning.

Let the expert dataset containing one complete state-only expert episode be denoted by $\mathcal{D}_E$, and let $\mathcal{D}_\mu$ denote the suboptimal offline dataset consisting of state–action trajectories generated by a mixed behavioral policy μ, where $\mathcal{D}_\mu$ contains a combination of expert and suboptimal trajectories. The agent does not have access to the trajectory labels; that is, it cannot directly distinguish which trajectories are expert and which are suboptimal. This setting is standard practice in offline imitation learning from observations [Sikchi et al., 2025, Yan et al., 2024]. We define the normalized discounted state visitation distribution of a policy π as $\rho_\pi(s) = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi)$. Accordingly, let

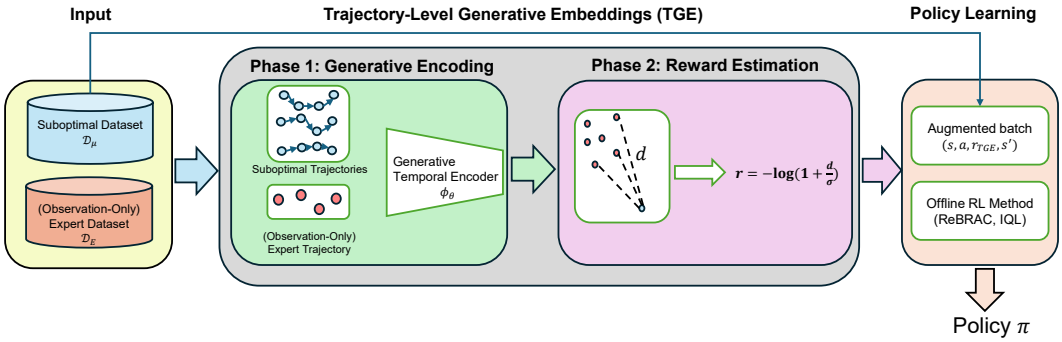


Figure 1: **Overview of Trajectory-level Generative Embeddings (TGE).** The framework employs a trajectory-level diffusion encoder to map trajectory segments into a latent embedding space. A surrogate reward is then computed using a logarithmic kernel over these embeddings to augment the reward-free suboptimal dataset, enabling offline RL training for policy learning.

ρ_E, ρ_π, and ρ_μ represent the distributions induced by the expert policy π_E, the learned policy π, and the data-collecting policy μ, respectively. The goal of the learning algorithm is to recover a policy π whose behavior closely matches that of the expert policy π_E.

Denoising Diffusion Probabilistic Models. Denoising Diffusion Probabilistic Models (DDPMs) [Ho et al., 2020] approximate complex data distributions via iterative denoising. The forward process gradually corrupts a clean sample $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ by adding Gaussian noise over K steps according to $q(\mathbf{x}_k \mid \mathbf{x}_{k-1}) = \mathcal{N}\big(\sqrt{1-\beta_k}\,\mathbf{x}_{k-1}, \beta_k I\big)$, where $\{\beta_k\}_{k=1}^K$ is a variance schedule, yielding $q(\mathbf{x}_K) \approx \mathcal{N}(0, I)$. The reverse process $p_\theta(\mathbf{x}_{k-1}|\mathbf{x}_k)$ is parameterized by a network $\boldsymbol{\epsilon}_\theta(\mathbf{x}_k, k)$ that predicts the injected noise. Training is performed by minimizing a mean-squared error objective:

$$\mathcal{L} = \mathbb{E}_{k,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_k}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_k}\,\boldsymbol{\epsilon}, k)\|_2^2\right],$$

where $\bar{\alpha}_k = \prod_{i=1}^k (1-\beta_i)$. Recent methods extend DDPMs to the trajectory level, treating $\mathbf{x}_0$ as a full state–action sequence to enable long-horizon planning [Janner et al., 2022, Ajay et al., 2022].

4 Methodology

We aim to develop a novel offline imitation learning from observations framework that leverages trajectory-level generative planners as encoders, which enable the algorithm to reason over extended horizons and incorporate rich temporal structures. To achieve this, we propose **T**rajectory-level **G**enerative **E**mbeddings (TGE), a method that constructs reward signals from the latent embeddings of a generative planner trained to perform density estimation over the suboptimal dataset $\mathcal{D}_\mu$. These trajectory-level latent embeddings provide a discriminative representation capable of distinguishing expert-like and suboptimal trajectories, thereby serving as an informative reward proxy for offline policy optimization on $\mathcal{D}_\mu$. Finally, we train an offline RL algorithm on the entire offline suboptimal dataset $\mathcal{D}_\mu$ using the TGE-derived reward function. The overall procedure is summarized in Algorithm 1, and an overview of the pipeline is illustrated in Figure 1.

4.1 An entropy perspective on distribution matching

The fundamental goal of imitation learning is to recover a policy π such that its induced latent state distribution ρ_π matches the expert distribution ρ_E. We formalize this by seeking to minimize the particle-based cross-entropy $H(\rho_\pi, \rho_E)$. Given n data points $\{z_i\}_{i=1}^n$ sampled from a latent state distribution induced by a policy π, we leverage non-parametric entropy estimation techniques Singh et al. [2003], Liu and Abbeel [2021] to express the particle-based cross-entropy approximation with respect to the expert distribution as:

$$H(\rho_\pi, \rho_E) = -\tfrac{1}{n}\textstyle\sum_{i=1}^n \log\left(\frac{m}{N_E v_i^m}\right) + b(m) \propto \sum_{i=1}^n \frac{\log v_i^m}{n},$$

where $b(m)$ is a bias correction term depending solely on the nearest neighbor parameter m. Here, v_i^m denotes the volume of a hypersphere with radius $\|z_i - z_E^m\|$, representing the Euclidean distance

between a policy datapoint z_i and its m-th nearest neighbor in the expert distribution:

$$v_i^m = \frac{\pi^{d_z/2}}{\Gamma(d_z/2+1)} \|z_i - z_E^m\|^{d_z},$$

where $d_z = |\mathcal{Z}|$ is the dimension of the latent space and Γ is the gamma function. To provide a robust learning signal, we further approximate the m-th nearest-neighbor log-volume by averaging the per-neighbor log-volumes over the m nearest expert neighbors $\mathcal{N}_m(\mathcal{Z}_E)$, yielding the approximation:

$$H(\rho_\pi, \rho_E) \approx \tfrac{1}{n} \textstyle\sum_{i=1}^n \tfrac{1}{m} \sum_{z_E^k \in \mathcal{N}_m(\mathcal{Z}_E)} \log \|z_i - z_E^k\|^{d_z}. \tag{1}$$

This suggests that minimizing the average log-distance between behavioral samples and their expert neighbors can serve as a surrogate for minimizing cross-entropy.

Since $\log \|z_i - z_E^k\|^{d_z} = d_z \log \|z_i - z_E^k\|$, the fixed positive factor d_z does not affect the minimizer of the unsmoothed objective and can be omitted. Following Liu and Abbeel [2021], we introduce an additive constant for numerical stability and define the surrogate reward by

$$r(z_i) = -\tfrac{1}{m} \textstyle\sum_{z_E^k \in \mathcal{N}_m(\mathcal{Z}_E)} \log\left(1 + \|z_i - z_E^k\|\right), \tag{2}$$

Using this reward, the cross-entropy objective in (1) can be written as:

$$\begin{aligned}
\widetilde{H}(\rho_\pi, \rho_E) &= \frac{1}{n} \sum_{i=1}^n \frac{1}{m} \sum_{z_E^k \in \mathcal{N}_m(\mathcal{Z}_E)} \log\left(1 + \|z_i - z_E^k\|\right) \\
&= -\frac{1}{n} \sum_{i=1}^n r(z_i) \\
&\approx -\mathbb{E}_{z_i \sim \rho_\pi}[r(z_i)] \\
&= -(1-\gamma)\mathbb{E}_\pi \left[\sum_{h=0}^\infty \gamma^h r(z_h)\right].
\end{aligned} \tag{3}$$

The expectation can be approximated by the sample average $-\frac{1}{n}\sum_{i=1}^n r(z_i)$. The equality in the second line follows from (2), and the last line uses the definition $\rho_\pi(\cdot) = (1-\gamma)\sum_{h=0}^\infty \gamma^h P_h(\cdot \mid \pi)$. Since $(1-\gamma) > 0$, minimizing the expected counterpart of this smoothed surrogate is equivalent to maximizing the cumulative surrogate reward. Having established this theoretical framework, we describe our method in the following sections.

4.2 Generative planner learns separable embeddings

We treat the encoder of the trajectory diffusion planner Diffuser Janner et al. [2022] as an unsupervised representation learner. As discussed in Section 2.3, the denoising objective encourages the model to learn temporally coherent features that are robust to noise, making them ideal for imitation learning. Specifically, as outlined in Line 1 of Algorithm 1, we train the diffusion noise model on the suboptimal dataset $\mathcal{D}_\mu$ using the objective

$$\mathcal{L}(\theta) = \mathbb{E}_{\tau_0 \sim \mathcal{D}_\mu,\, k,\, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\tau_k, k)\|_2^2\right], \tag{4}$$

where $k \sim \mathrm{Unif}[K]$ denotes the diffusion timestep, $\epsilon \sim \mathcal{N}(0, I)$ is the Gaussian noise target, and τ_k is the noisy trajectory obtained by adding noise to the clean trajectory τ_0 at diffusion timestep k. Let $\tau \in \mathcal{T}$ denote a trajectory segment of length H, where the trajectory space for training is defined as $\mathcal{T} = \mathbb{R}^{H \times (|\mathcal{S}|+|\mathcal{A}|)}$. To construct these segments from $\mathcal{D}_\mu$, we extract overlapping sequences of length H using a sliding window. While the diffusion model is trained on these state–action trajectories, the encoder ϕ_θ is designed to produce a state-only latent embedding. Specifically, during the embedding process, we mask the action dimensions of τ, ensuring the representation space $\mathcal{Z}$ captures temporal state dynamics consistent with the observation-only expert data in $\mathcal{D}_E$. We achieve this by conceptually decomposing the U-Net model ϵ_θ [Ronneberger et al., 2015] into a temporal encoder $\phi : \mathcal{T} \times \mathbb{R} \to \mathcal{Z}$ and a decoder $\psi : \mathcal{Z} \times \mathbb{R} \to \mathcal{T}$, such that $\epsilon_\theta(\tau_k, k) = (\psi_\theta \circ \phi_\theta)(\tau_k, k)$. Minimizing the denoising score-matching objective in Equation (4) encourages the latent representation $\mathcal{Z}$ to preserve information about the underlying clean trajectory distribution. Consequently, $\mathcal{Z}$ captures

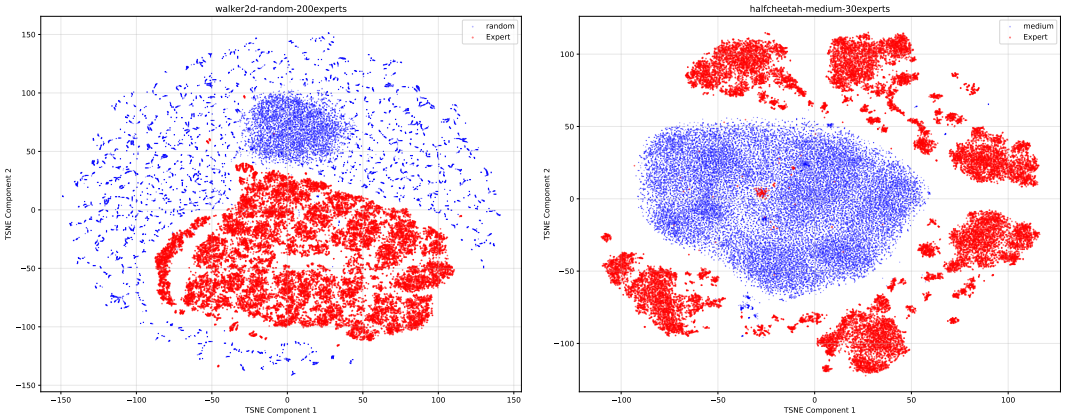


Figure 2: **T-SNE Embedding Visualization.** The latent embeddings produced by the diffuser naturally separate expert transitions from suboptimal ones. Expert (red) and suboptimal (blue) samples form distinct clusters in the embedding space, despite no discriminative supervision.

temporally coherent features that are stable under perturbations, yielding an embedding space that is more semantically discriminative for downstream similarity-based rewards.

Given a clean input trajectory $\tau_0 \sim \mathcal{D}_\mu$, we extract a latent embedding using the encoder. Figure 2 illustrates that, even without any explicit discriminative supervision between expert-like and suboptimal trajectories in $\mathcal{D}_\mu$, the encoder-induced embedding $z = \phi_\theta(\tau_0, 0)$ naturally separates expert-like samples from suboptimal ones in the mixed dataset. Given this desirable separation property of the embeddings, it is natural to leverage them for reward construction by directly measuring an appropriate distance between embeddings as we discussed in Sec. 4.1.

4.3 Reward estimation on trajectory-level embeddings

We now describe the procedure for generating synthetic rewards based on the trajectory-level embeddings induced by the encoder of the generative planner. Given the expert dataset $\mathcal{D}_E$, we extract expert embeddings $\mathcal{Z}_E$ by applying a sliding window with a stride of 1 to the state-only expert episode. Specifically, $\mathcal{Z}_E = \{\phi_\theta(\tau_{i:i+H}) \mid \tau \in \mathcal{D}_E, 0 \le i \le |\tau| - H\}$, where the action dimensions of the input to ϕ_θ are treated as null for these observation-only segments (Line 2 in Algorithm 1). For each state s_t in the suboptimal dataset $\mathcal{D}_\mu$, we consider the trajectory segment starting at time t, $\tau_\mu = s_{t:t+H}$, and encode it into a latent vector $z_t = \phi_\theta(\tau_\mu)$. In the expert embedding space $\mathcal{Z}_E$, we compute the ℓ_2-norms between z_t and all expert embeddings $z_E \in \mathcal{Z}_E$, and then identify the top-m nearest neighbors $\mathcal{N}_m(\mathcal{Z}_E)$.

To construct an entropy-based surrogate reward from the suboptimal trajectory segment τ_μ, we average the kernel-transformed distances between its latent embedding z_t and its m nearest expert neighbors:

$$r_{\mathrm{TGE}}(s_t) = \frac{1}{m} \sum_{z_E \in \mathcal{N}_m(\mathcal{Z}_E)} f\left(\frac{\|z_t - z_E\|_2}{\sigma}\right),$$

where σ is a kernel temperature. Note that we normalize all latent embeddings by projecting them onto the unit sphere (dividing each vector by its L_2-norm), ensuring that all embeddings have equal length and allowing for consistent distance computations under the kernel. We adopt a logarithmic kernel $f(d) = -\log(1 + d)$. This serves as a proxy for density estimation, consistent with the theoretical derivation linking this metric to particle-based entropy maximization Liu and Abbeel [2021] provided in Section 4.1. To justify our theoretical insights, we provide the ablation study in Appendix C which shows that our theoretically grounded selection provides the best performance.

4.4 Downstream policy learning

Since the original suboptimal dataset $\mathcal{D}_\mu$ consists of state–action transitions without rewards, the reward estimation procedure described in Section 4.3 enables us to annotate each transition with a synthetic reward $r_{\mathrm{TGE}}(s)$, which measures the similarity of the current state to the expert state

Algorithm 1 Trajectory-Level Generative Embeddings

Require: Offline suboptimal dataset $\mathcal{D}_\mu$, expert dataset $\mathcal{D}_E$, horizon H, number of candidates m, temperature σ, and kernel function $f(\cdot)$.

Train generative planner on $\mathcal{D}_\mu$ to obtain encoder ϕ_θ.
Compute embeddings $\mathcal{Z}_\mu = \phi_\theta(\mathcal{D}_\mu)$ and $\mathcal{Z}_E = \phi_\theta(\mathcal{D}_E)$ on trajectory segments of length H.
Project all latent vectors to a unit sphere: $z \leftarrow z/\|z\|_2$.
for each $z_t \in \mathcal{Z}_\mu$ **do**
    Let $d(z_t, z_E) = \|z_t - z_E\|_2$ for all $z_E \in \mathcal{Z}_E$.
    Identify the top-m nearest neighbors $\mathcal{N}_m(\mathcal{Z}_E)$.
    $r_{\text{TGE}}(s_t) \leftarrow \frac{1}{m} \sum_{z_{nb} \in \mathcal{N}_m(\mathcal{Z}_E)} f\left(d(z_t, z_{nb})/\sigma\right)$.
end for
Run any offline RL algorithm on the suboptimal dataset augmented with synthetic rewards: $\mathcal{D}'_\mu = \{(s_t, a_t, r_{\text{TGE}}(s_t), s_{t+1})\}$.

distribution. Higher values indicate stronger expert-likeness, while lower values correspond to more suboptimal behavior.

With these synthetic rewards, the augmented dataset $\mathcal{D}'_\mu = \{(s_t, a_t, r_{\text{TGE}}(s_t), s_{t+1})\}$ can be used to train any offline RL method for policy learning. In our experiments, we adopt two state-of-the-art offline RL algorithms, IQL [Kostrikov et al., 2021] and ReBRAC [Tarasov et al., 2023], and demonstrate that both methods effectively learn high-quality policies when equipped with our synthetic reward signal. Notably, for ReBRAC, we introduce an adaptive behavior cloning mechanism that weights the imitation penalty by the TGE reward to prioritize expert-like transitions during training. Full implementation details are provided in Appendix A.2.

4.5 Learning under support mismatch

We highlight a structural difference between TGE and occupancy-matching methods. DICE-based approaches rely on density ratios $w(s) \approx d^{\pi_E}(s)/d^{\mathcal{D}_\mu}(s)$, which typically assume a nontrivial degree of support overlap [Ma et al., 2022]. When the offline data poorly covers the expert support, these ratios become ill-conditioned. As $d^{\pi_E}(s) \to 0$, importance weights vanish and supervision collapses. In contrast, the entropy-derived reward $r(s) = -\log(1 + \|z_s - z_E\|)$ provides a smoothly decaying signal that persists even for states far from the expert support.

The effectiveness of this extrapolation depends on the semantic quality of the underlying metric. Standard distance metrics, such as L2 distance on raw observations, often fail due to state aliasing, where states that are close may be functionally distinct. By leveraging the encoder of a trajectory-level generative planner, the induced latent space is organized according to long-horizon dynamics rather than superficial single-step features. To examine the contribution of the learned latent geometry, we compare TGE with alternative reward constructions based on raw observations, VAE embeddings, and SEABO [Lyu et al., 2024], using ReBRAC as the shared downstream learner (Section 5.2.3). Temporal context also contributes to reward quality, with single-step embeddings substantially degrading downstream performance (Appendix C.1).

5 Experiments

5.1 Experimental setup

Environments. We evaluate our method based on D4RL [Fu et al., 2020] datasets, covering locomotion (MuJoCo: `hopper`, `halfcheetah`, `walker2d`, `ant`), dexterous manipulation (Adroit: `pen`, `door`, `hammer`), and long-horizon multi-task manipulation (Franka Kitchen: `partial`, `mixed`).

Expert data. All methods are additionally provided with an **observation-only** demonstration $\mathcal{D}_E$ consisting of one single expert episode, which supplies the expert state sequences required.

Suboptimal offline data. The composition of $\mathcal{D}_\mu$ varies by environment. For Adroit and Franka Kitchen, $\mathcal{D}_\mu$ follows the standard benchmark protocol: `cloned`/`human` trajectories for Adroit and `partial`/`mixed` datasets for Kitchen, augmented with 30 and 18 unlabeled expert trajectories respectively. For MuJoCo locomotion, we consider four mixture settings that vary the level of

Table 1: **Main Results.** TGE combined with ReBRAC(highlighted in cyan) is compared against state-of-the-art offline LfO baselines. The results demonstrate that TGE achieves superior performance. The top-performing results and those with overlapping standard deviations are highlighted in **bold**.

Dataset	Env	Diffuser (BC)	SMODICE	PW-DICE	DILO	IOSTOM	TGE+ReBRAC	Expert
random+expert	hopper	1.45±0.17	103.79±4.76	**106.22±8.21**	**104.96±6.52**	**109.51±0.29**	**109.72±0.68**	111.34
	halfcheetah	0.15±0.49	77.10±6.72	90.63±1.08	89.40±2.65	90.06±2.12	**93.20±0.70**	88.83
	walker2d	-0.01±0.04	**108.74±0.72**	**107.97±0.94**	**108.88±0.39**	**107.59±0.04**	**108.68±0.20**	106.93
	ant	9.58±4.77	124.10±2.19	123.15±8.48	122.87±4.18	**128.80±1.97**	122.72±4.19	130.75
random+few-expert	hopper	1.38±0.06	62.04±17.02	89.32±17.29	92.66±7.42	**103.97±3.48**	88.86±12.28	111.34
	halfcheetah	0.03±0.54	2.60±0.64	31.16±32.38	48.26±14.18	64.93±19.04	**89.70±1.63**	88.83
	walker2d	0.01±0.03	16.97±35.72	97.68±10.98	**107.32±1.96**	**107.58±0.40**	**105.48±2.52**	106.93
	ant	7.25±0.07	32.11±8.38	108.08±15.99	112.50±7.77	**123.01±0.82**	111.46±4.14	130.75
medium+expert	hopper	45.52±4.81	57.77±8.32	69.38±28.15	102.52±5.73	**110.54±0.69**	**109.64±0.25**	111.34
	halfcheetah	41.33±0.49	57.27±2.20	60.76±3.15	89.98±0.61	**92.17±0.78**	**93.10±0.28**	88.93
	walker2d	59.74±5.69	70.03±18.38	85.17±6.85	**108.52±0.71**	**108.14±0.19**	**108.95±0.61**	106.93
	ant	87.42±6.38	104.68±5.21	117.51±6.84	92.41±2.62	124.24±4.80	**137.89±0.45**	130.75
medium+few-expert	hopper	50.74±2.92	52.02±2.64	56.79±11.55	38.07±11.56	**104.89±5.22**	97.24±2.22	111.34
	halfcheetah	41.49±1.28	41.60±2.99	46.33±7.44	66.07±8.52	53.44±26.85	**91.83±0.80**	88.83
	walker2d	61.57±6.92	73.72±4.57	80.77±3.14	70.19±2.51	**108.10±0.25**	**108.02±0.76**	106.93
	ant	86.13±7.24	89.11±1.53	101.26±8.73	92.55±5.60	119.69±2.22	**130.23±1.36**	130.75
cloned+expert	pen	30.98±46.56	22.63±6.35	1.49±3.61	61.05±10.86	**72.53±12.67**	63.15±5.11	167.18
	door	0.27±1.84	-0.08±0.08	1.17±2.16	**100.90±2.48**	89.05±3.98	11.20±7.82	103.95
	hammer	0.59±0.19	0.48±0.46	6.00±5.44	54.29±23.25	**68.46±17.79**	37.84±8.05	125.72
human+expert	pen	73.84±18.47	47.64±14.83	27.71±6.32	**99.33±10.13**	91.88±7.01	90.30±5.36	167.18
	door	47.21±12.72	1.50±1.06	0.06±0.03	96.45±4.55	**100.16±6.84**	**101.43±3.73**	104.73
	hammer	42.33±20.70	0.34±0.23	29.26±34.09	91.24±13.50	88.83±28.02	**109.63±9.16**	125.72
kitchen	partial	0.0±0.0	35.0±4.08	12.33±5.38	25.33±25.03	**58.33±7.64**	**57.25±6.81**	75.00
	mixed	0.0±0.0	**48.33±4.24**	7.50±4.16	21.00±6.06	**46.67±1.44**	**45.92±2.93**	75.00

expert coverage in $\mathcal{D}_\mu$: *expert* (1M suboptimal transitions plus 200 unlabeled expert trajectories), *few-expert* (1M suboptimal transitions plus 30 unlabeled expert trajectories), *medium* (1M medium-quality transitions with no expert data), and *medium+random* (1M transitions split equally between medium and random data, no expert). The latter two settings are first introduced in this paper to test performance under severe support mismatch between $\mathcal{D}_\mu$ and the expert. Full mixture details are in Appendix B.

Baselines and Evaluation. We compare against recent offline learning-from-observations (LfO) methods, including SMODICE Ma et al. [2022], PW-DICE Yan et al. [2024], DILO Sikchi et al. [2025] and IOSTOM [Pham et al., 2025], which have been shown to outperform earlier approaches such as LobsDICE Kim et al. [2022] and DemoDICE Kim et al. [2021]. We additionally include trajectory-level behavior cloning (BC) as a baseline, trained on the suboptimal data using diffuser [Janner et al., 2022] without reward guidance.

We adopt ReBRAC as our default offline RL backbone, paired with adaptive BC weighting (Appendix C.5), which yields slightly stronger results than IQL. Training dynamics for both TGE+ReBRAC and TGE+IQL are shown in Appendix D, demonstrating that the TGE reward signal transfers across offline RL algorithms. We report the mean and standard deviation of normalized D4RL scores across five seeds. The pseudocode and hyperparameters are in Appendix A.

5.2 Experimental results

5.2.1 Learning from observations

Performance on MuJoCo Locomotion Benchmarks. As shown in Table 1, TGE consistently matches or outperforms all baselines across environments. In `random+expert` and `medium+expert`, where the offline buffer contains relatively sufficient expert-like coverage, TGE remains highly competitive and achieves strong performance. The performance gains are most pronounced in `random+few-expert` and `medium+few-expert`, where expert trajectories are scarce. The DICE-based methods (SMODICE, DILO, PW-DICE) degrade substantially, while IOSTOM remains competitive on most tasks but underperforms on `halfcheetah` and `ant`. Notably, TGE occasionally surpasses the provided expert demonstrations, as the single observation-only expert trajectory may not represent the best behavior in the dataset. By measuring trajectory-level proximity in the diffusion embedding space, TGE can recover stronger expert-like behaviors present in the offline buffer.

Performance on Manipulation Benchmarks. On the high-dimensional Adroit, TGE performs strongly under `human+expert`, achieving the highest mean scores on `door` and `hammer`, while

remaining competitive on `pen`. However, it underperforms leading occupancy-matching baselines under `cloned+expert`, particularly on `door`. One possible explanation is that latent similarity may reflect task phase more strongly than task success. In stage-dependent manipulation tasks, a segment may be closer to a failed segment from the same phase than to a successful expert segment from a different phase. Consequently, proximity in the embedding space may not reliably indicate task-relevant progress. On Franka Kitchen, TGE remains competitive with strong baselines on both `partial` and `mixed`, demonstrating its applicability beyond single-task manipulation.

5.2.2 LfO from severe support mismatch

We isolate the setting where the suboptimal dataset $\mathcal{D}_\mu$ contains *no* expert trajectories at all, providing the cleanest empirical test of the support-mismatch analysis in Section 4.5. Two settings are considered: **medium**, where $\mathcal{D}_\mu$ consists of 1M transitions from a medium-quality policy, and **medium+random**, where $\mathcal{D}_\mu$ contains 1M equally mixed medium and random data. These settings are particularly challenging because they push the data distribution further from the expert support.

TGE+ReBRAC maintains strong performance across all tasks, while DICE-based baselines degrade sharply or collapse entirely as the data distribution shifts further from the expert. The results confirm the prediction of Section 4.5, that DICE-based baselines lose graded supervision as $\mathcal{D}_\mu$ drifts from the expert support, while TGE's distance-shaped reward in the diffusion latent space remains informative. Training-dynamics curves for the medium+random setting are provided in Figure 8 (Appendix D).

Table 2: **Results under severe support mismatch.** TGE+ReBRAC (highlighted in cyan) maintains strong performance across all tasks, while several DICE-based baselines degrade substantially.

Dataset	Env	Diffuser (BC)	SMODICE	PW-DICE	DILO	IOSTOM	TGE+ReBRAC	Expert
medium	hopper	46.33±1.86	56.18±1.16	**65.82±11.39**	48.68±17.85	59.50±7.45	**67.90±1.59**	111.34
	halfcheetah	41.64±0.30	42.44±0.50	43.27±0.58	42.07±0.47	37.50±6.11	**48.00±0.30**	88.83
	walker2d	65.62±4.89	71.28±7.21	75.64±3.58	69.74±1.13	72.35±5.65	**84.56±0.68**	106.93
	ant	85.67±11.66	88.94±2.89	94.90±4.55	90.13±4.68	100.09±0.76	**124.12±1.79**	130.75
medium+random	hopper	6.62±7.10	54.86±2.49	51.77±16.16	58.71±3.86	48.20±11.74	**65.30±1.87**	111.34
	halfcheetah	12.81±17.20	16.49±2.32	3.17±1.69	2.25±0.01	2.24±0.00	**44.22±0.14**	88.83
	walker2d	0.96±2.06	**78.93±6.07**	41.46±38.23	46.52±14.22	45.09±40.05	**79.12±1.21**	106.93
	ant	16.77±5.72	89.88±1.64	87.02±1.72	98.70±1.61	92.86±8.14	**106.28±0.55**	130.75

5.2.3 Comparison of reward constructions

We compare five reward constructions using ReBRAC as the downstream policy learner: a uniform reward, a distance-based reward on raw observations, a reward based on VAE trajectory embeddings, SEABO [Lyu et al., 2024], and TGE. Both the VAE and TGE encoders use trajectory windows of length $H = 32$. This comparison evaluates alternative reward constructions under a shared policy-learning backbone.

Table 3: Comparison of reward constructions with ReBRAC. We report mean and standard deviation of normalized scores. The highest mean in each setting is highlighted in bold.

Environment	Uniform	Raw state	VAE	SEABO	TGE
Medium, without expert trajectories					
hopper	59.97 ± 1.51	61.28 ± 1.21	60.37 ± 2.53	65.18 ± 3.49	$\mathbf{67.90 \pm 1.59}$
halfcheetah	42.80 ± 0.25	44.24 ± 0.19	46.39 ± 0.12	$\mathbf{48.21 \pm 0.94}$	$\mathbf{48.00 \pm 0.30}$
walker2d	76.40 ± 1.11	80.20 ± 0.45	81.23 ± 1.93	75.61 ± 1.76	$\mathbf{84.56 \pm 0.68}$
ant	92.70 ± 3.23	114.47 ± 0.44	115.44 ± 0.04	108.17 ± 1.54	$\mathbf{124.12 \pm 1.79}$
Random + 30 unlabeled expert trajectories					
hopper	35.37 ± 26.52	80.77 ± 11.74	$\mathbf{98.63 \pm 20.13}$	82.22 ± 21.46	88.86 ± 12.28
halfcheetah	2.25 ± 0.01	82.84 ± 2.29	$\mathbf{87.65 \pm 1.91}$	67.12 ± 4.91	$\mathbf{89.70 \pm 1.63}$
walker2d	68.23 ± 57.74	101.24 ± 11.36	$\mathbf{103.93 \pm 3.66}$	104.43 ± 4.26	$\mathbf{105.48 \pm 2.52}$
ant	37.75 ± 7.05	108.34 ± 5.50	$\mathbf{111.15 \pm 7.85}$	96.65 ± 12.66	$\mathbf{111.46 \pm 4.14}$
Medium + 30 unlabeled expert trajectories					
hopper	59.32 ± 2.57	81.64 ± 8.24	$\mathbf{98.63 \pm 7.84}$	86.84 ± 8.68	$\mathbf{97.24 \pm 2.22}$
halfcheetah	82.84 ± 2.29	44.41 ± 0.17	89.91 ± 1.76	87.42 ± 3.63	$\mathbf{91.83 \pm 0.80}$
walker2d	77.84 ± 1.26	80.40 ± 0.91	$\mathbf{107.29 \pm 1.86}$	77.81 ± 1.86	$\mathbf{108.02 \pm 0.76}$
ant	93.05 ± 1.19	119.11 ± 1.31	117.48 ± 5.44	115.30 ± 1.91	$\mathbf{130.23 \pm 1.36}$

As shown in Table 3, TGE achieves best or competitive performance in 11 of 12 settings and the highest average score across the evaluated settings. Its gains over the uniform reward indicate the value of expert-informed reward construction. Compared with raw-state distances and SEABO, TGE generally provides stronger performance, while the VAE baseline also performs well and achieves higher mean scores on two Hopper settings. These results support the utility of learned trajectory representations, with diffusion embeddings providing the strongest overall performance in this comparison.

5.2.4 LfO from mismatched dynamics

Beyond support mismatch in the data distribution, we evaluate TGE under dynamics mismatch, where the expert and the offline data are generated by agents with different physical embodiments. Following SMODICE [Ma et al., 2022], we construct two settings: **HalfCheetah-Short**, where the expert operates with a halved torso, and **Ant-Disabled**, where the expert operates with two disabled legs. In both cases, $\mathcal{D}_E$ contains expert state trajectories from the modified embodiment, while $\mathcal{D}_\mu$ consists of suboptimal trajectories from the standard embodiment. The agent must imitate expert-like state visitation despite being unable to reproduce the expert's exact dynamics.

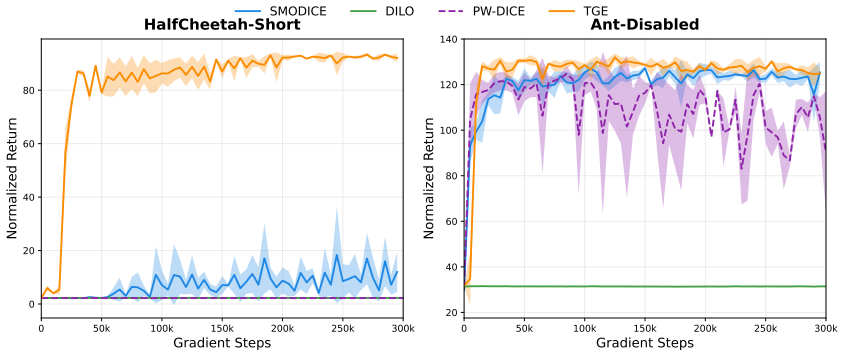


Figure 3: **Training dynamics under mismatched dynamics.**[1] TGE achieves stable and rapid convergence to expert-level performance on both HalfCheetah-Short and Ant-Disabled, while DICE-based baselines either fail to converge or remain substantially below expert performance.

As shown in Figure 3, TGE rapidly converges to near-expert performance on both tasks, while SMODICE plateaus well below expert performance on HalfCheetah-Short and DILO fails to learn a functional policy on either task. We attribute this robustness to the trajectory-level latent geometry. Even when the offline data is generated by a different embodiment, the diffusion encoder captures temporal dynamics that remain semantically meaningful relative to the expert state distribution, allowing a graded reward signal where occupancy-matching objectives become ill-conditioned.

6 Conclusion

In this work, we introduced Trajectory-level Generative Embeddings (TGE), a novel framework for offline imitation learning from observations that addresses the structural limitations of distribution-matching approaches. By shifting the paradigm from density ratio estimation to generative representation learning, TGE reduces reliance on expert-behavior support overlap that hampers prior methods. Motivated by the particle-based cross-entropy estimation, we demonstrated that encoding trajectories into the latent space of a temporal diffusion model allows for the construction of a dense, informative reward signal even when the support mismatch between offline data and the expert is severe.

These findings suggest that generative models can serve as powerful tools for structuring the state space in RL, going beyond their traditional roles in planning and policies. Future work could explore structured trajectory alignment to better distinguish task progress from phase similarity in stage-dependent manipulation, as well as extending the framework to online fine-tuning and imitation from cross-embodiment observations with video diffusion models.

[1]We omit IOSTOM from this experiment because its released implementation does not support the mismatched-dynamics setting, and our adaptation did not yield stable training despite hyperparameter tuning.

Acknowledgments and Disclosure of Funding

We thank the anonymous reviewers for their helpful comments. This research was supported by WZ's startup funding provided by the School of Data and Information Science at UNC Chapel Hill. This research was also supported by the NVIDIA Academic Grant Program with the cloud GPU credits and cloud infrastructure.

References

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos J Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *Advances in Neural Information Processing Systems*, 37:58757–58791, 2024.

Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. In *Robotics: Science and Systems (RSS)*, 2019.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. In *Robotics: Science and Systems (RSS)*, 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 (RSS)*, 2023.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 4693–4700. IEEE, 2018.

Shutong Ding, Ke Hu, Zhenhao Zhang, Kan Ren, Weinan Zhang, Jingyi Yu, Jingya Wang, and Ye Shi. Diffusion-based reinforcement learning via q-weighted variational policy optimization. *Advances in Neural Information Processing Systems*, 37:53945–53968, 2024.

Pete Florence, Corey Lynch, Andy Zeng, Oscar Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on Robot Learning (CoRL)*, 2021.

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.

Michael Fuest, Pingchuan Ma, Ming Gui, Johannes Schusterbauer, Vincent Tao Hu, and Bjorn Ommer. Diffusion models and representation learning: A survey. *arXiv preprint arXiv:2407.00783*, 2024.

Divyansh Garg, Shuvam Chakraborty, Chris Cundy, Jiaming Song, and Stefano Ermon. Iq-learn: Inverse soft-q learning for imitation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3), 2018.

Josiah Hanna and Peter Stone. Grounded action transformation for robot learning in simulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Tao Huang, Guangqi Jiang, Yanjie Ze, and Huazhe Xu. Diffusion reward: Learning rewards via conditional video diffusion. In *European Conference on Computer Vision*, pages 478–495. Springer, 2024.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pages 9902–9915. PMLR, 2022.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33: 21810–21823, 2020.

Geon-Hyeong Kim, Seokin Seo, Jongmin Lee, Wonseok Jeon, HyeongJoo Hwang, Hongseok Yang, and Kee-Eung Kim. Demodice: Offline imitation learning with supplementary imperfect demonstrations. In *International Conference on Learning Representations*, 2021.

Geon-Hyeong Kim, Jongmin Lee, Youngsoo Jang, Hongseok Yang, and Kee-Eung Kim. Lobsdice: Offline learning from observation via stationary distribution correction estimation. *Advances in Neural Information Processing Systems*, 35:8252–8264, 2022.

Ilya Kostrikov, Ofir Nachum, and Jonathan Tompson. Imitation learning via off-policy distribution matching. In *International Conference on Learning Representations*, 2020.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International conference on machine learning*, pages 5639–5650. PMLR, 2020.

Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. Urlb: Unsupervised reinforcement learning benchmark. *arXiv preprint arXiv:2110.15191*, 2021.

Hao Liu and Pieter Abbeel. Behavior from the void: Unsupervised active pre-training. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

YuXuan Liu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1118–1125. IEEE, 2018.

Jiafei Lyu, Xiaoteng Ma, Le Wan, Runze Liu, Xiu Li, and Zongqing Lu. Seabo: A simple search-based method for offline imitation learning. *arXiv preprint arXiv:2402.03807*, 2024.

Yecheng Ma, Andrew Shen, Dinesh Jayaraman, and Osbert Bastani. Versatile offline imitation from observations and examples via regularized state-occupancy matching. In *International Conference on Machine Learning*, pages 14639–14663. PMLR, 2022.

Mirco Mutti, Lorenzo Pratissoli, and Marcello Restelli. A policy gradient method for task-agnostic exploration. In *4th Lifelong Machine Learning Workshop at ICML 2020*, 2020.

Quang Anh Pham, Janaka Chathuranga Brahmanage, Tien Anh Mai, and Akshat Kumar. Iostom: Offline imitation learning from observations via state transition occupancy matching. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

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

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

Harshit Sikchi, Caleb Chuck, Amy Zhang, and Scott Niekum. A dual approach to imitation learning from observations with offline datasets. In *Conference on Robot Learning*, pages 1125–1147. PMLR, 2025.

Harshinder Singh, Neeraj Misra, Vladimir Hnizdo, Adam Fedorowicz, and Eugene Demchuk. Nearest-neighbor estimates of entropy. *American Journal of Mathematical and Management Sciences*, 23 (3-4):301–321, 2003.

Bradly C Stadie, Pieter Abbeel, and Ilya Sutskever. Third person imitation learning. In *International Conference on Learning Representations*, 2017.

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36:11592–11620, 2023.

Yonglong Tian, Lijie Fan, Phillip Isola, Huiwen Chang, and Dilip Krishnan. Stablerep: Synthetic images from text-to-image models make strong visual representation learners. *Advances in Neural Information Processing Systems*, 36:48382–48402, 2023.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 4950–4957. International Joint Conferences on Artificial Intelligence Organization, 2018.

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*.

Kai Yan, Alex Schwing, and Yu-Xiong Wang. Offline imitation from observation via primal wasserstein state occupancy matching. In *International Conference on Machine Learning*, pages 55814–55838. PMLR, 2024.

Amy Zhang, Rowan Thomas McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=-2FCwDKRREu`.

Konrad Zolna, Alexander Novikov, Ksenia Konyushkova, Caglar Gulcehre, Ziyu Wang, Yusuf Aytar, Misha Denil, Nando De Freitas, and Scott Reed. Offline learning from demonstrations and unlabeled experience. *arXiv preprint arXiv:2011.13885*, 2020.

A Hyperparameters and implementation details

A.1 Hyperparameters

We list the hyperparameters used for the diffuser, including details of the U-Net architecture, as well as those for TGE reward estimation in Table 4. We also report the hyperparameters for the downstream offline RL backbones, IQL and ReBRAC, in Tables 5 and 6, respectively. For all baseline methods, we strictly follow the original implementations and hyperparameter settings reported in their respective papers.

Table 4: **Hyperparameter settings for TGE.** The hyperparameters below are used to train the trajectory-level generative planner and estimate the rewards.

Hyperparameter	**Value**
Reward Estimation	
Kernel type	Logarithmic
Kernel temperature (σ)	1.0
Number of nearest neighbors (m)	10
Diffuser Architecture	
Diffuser horizon (H)	32
Channel multipliers	(1, 2, 4, 8)
Diffusion steps	20
Embedding dimension	256
Learning rate	2e-4
Batch size	32
EMA decay	0.995
Scheduler	DDPMScheduler

Table 5: **Hyperparameter Settings for IQL.**

Hyperparameter	**Value**
Optimizer	
Optimizer	AdamW
Actor Learning rate	$3e-4$
Critic Learning rate	$3e-4$
Value Function Learning rate	$3e-4$
Batch size	1024
Algorithm Specifics	
Discount factor (γ)	0.99
Target smoothing (τ_{target})	0.005
Expectile	0.8
Inverse Temperature (β)	0.5
Dropout rate	0 (Mujoco) / 0.1(Adroit)

A.2 Implementation details

We implement the TGE reward as the logarithmic distance to the nearest expert embeddings, consistent with the entropy minimization objective derived in Section 4.1. Since $\mathcal{D}_E$ contains only state observations, we mask the action dimensions of all trajectory segments by setting them to zero before encoding, applying the same masking to both expert and suboptimal segments for consistency. All embeddings are normalized to the unit hypersphere ($z \leftarrow z/\|z\|_2$). Distances are computed using Euclidean distance on the sphere, which is monotonic with cosine distance ($\|z_i - z_j\|_2 = \sqrt{2 - 2\cos\theta}$).

Table 6: **Hyperparameter Settings for ReBRAC.** For the actor BC coefficient, values $\{0.1, 0.4, 1.0\}$ are used for {Adroit-Cloned, Mujoco, Adroit-Human}, respectively.

Hyperparameter	**Value**
Optimizer & Architecture	
Optimizer	AdamW
Actor Learning rate	$5e-4$
Critic Learning rate	$5e-4$
Weight decay	$1e-4$
Batch size	4096
Hidden layers	3
Hidden dimension	256
Layer normalization	True
Actor Dropout	0 (Mujoco) / 0.1(Adroit)
Algorithm Specifics	
Discount factor (γ)	0.99
Target smoothing (τ_{target})	0.005
Policy update frequency	2
Policy noise (σ)	0.2
Noise clip	0.5
Actor BC coefficient (β_{actor})	0.1, 0.4, 1.0
Critic BC coefficient (β_{critic})	1.0

For each offline transition, we retrieve its m nearest expert neighbors in the embedding space (with m=10 by default) and compute the reward using a logarithmic kernel: $r = -\frac{1}{m}\sum_{k=1}^{m}\log(1+d_k/\sigma)$, where d_k is the distance to the k-th neighbor and σ is a temperature scaling parameter. This reward signal is then normalized and used to label transitions in $\mathcal{D}_\mu$.

Finally, we train offline policies using IQL and ReBRAC as drop-in backbones, relying solely on $\mathcal{D}_\mu$ and the learned TGE reward, without any online interaction. Additionally, to ensure stable learning and balanced sampling across the diverse trajectories in the suboptimal dataset, we utilize the normalized TGE reward as a weighting factor for the behavioral cloning term in our ReBRAC backbone. We treat rewards as fixed confidence weights that scale the MSE penalty between the actor's actions and the dataset actions. This ensures that the policy regularizes more strongly toward expert-labeled regions (high reward) while maintaining flexibility in suboptimal regions, effectively preventing the policy from collapsing toward poor-quality data points.

A.3 Computational overhead

As noted in the experimental evaluation, TGE involves a multi-stage pre-processing pipeline before policy learning begins. We report wall-clock times on Hopper-Random+Expert using a single NVIDIA L40S GPU in Table 7.

Table 7: Wall-clock time in seconds. Total runtime includes 300K policy updates for TGE and SEABO, and 500K updates for IOSTOM and DILO, following their respective convergence settings.

Method	Pretraining	Annotation	RL 300K	RL 500K	Total
TGE	≈3,600	42.9	448.7	–	≈4,092
SEABO	–	30.5	444.9	–	475.4
IOSTOM	–	–	413.2	680.7	680.7
DILO	–	–	2,287.2	3,841.5	3,841.5

TGE's main computational overhead is diffusion encoder pretraining. Including this stage, its total runtime exceeds that of SEABO and IOSTOM and is comparable to DILO. However, for a fixed offline dataset, the pretrained encoder can be reused across expert demonstrations, reward relabeling runs, and downstream policy training. With a pretrained encoder, annotation and 300K policy updates

take only 491.6 seconds in this setting, which is lower than the runtimes of IOSTOM and DILO at their respective convergence budgets.

B Details of the mixture strategy for the suboptimal dataset

We provide a detailed breakdown of the data mixture strategy used to construct the suboptimal dataset $\mathcal{D}_\mu$. Following established protocols in offline imitation learning from observations introduced in [Ma et al., 2022, Sikchi et al., 2025], we construct $\mathcal{D}_\mu$ by combining a large corpus of suboptimal transitions with a limited number of expert trajectories. Table 8 summarizes the detailed composition for all suboptimal datasets.

Table 8: **Dataset Mixture Details.** We summarize the data mixture strategies for the suboptimal datasets $\mathcal{D}_\mu$ used across different experimental settings.

Dataset Name	Suboptimal Data	Expert Trajectories	Episode Length
Locomotion Tasks			
Random + Expert	1e6 transitions	200	1000
Medium + Expert	1e6 transitions	200	1000
Random + Few-Expert	1e6 transitions	30	1000
Medium + Few-Expert	1e6 transitions	30	1000
Medium	1e6 transitions	0	1000
Medium + Random	1e6 transitions	0	1000
Adroit Tasks			
Pen (Cloned + Expert)	5e5 transitions	30	100
Pen (Human + Expert)	5e3 transitions	30	100
Door (Cloned + Expert)	1e6 transitions	30	200
Door (Human + Expert)	6.7e3 transitions	30	200
Hammer (Cloned + Expert)	1e6 transitions	30	200
Hammer (Human + Expert)	1.1e4 transitions	30	200
Kitchen-Partial	1.37e5 transitions	18	280
Kitchen-Mixed	1.37e5 transitions	18	280

C Additional ablation studies

C.1 Impact of horizons in the generative planner

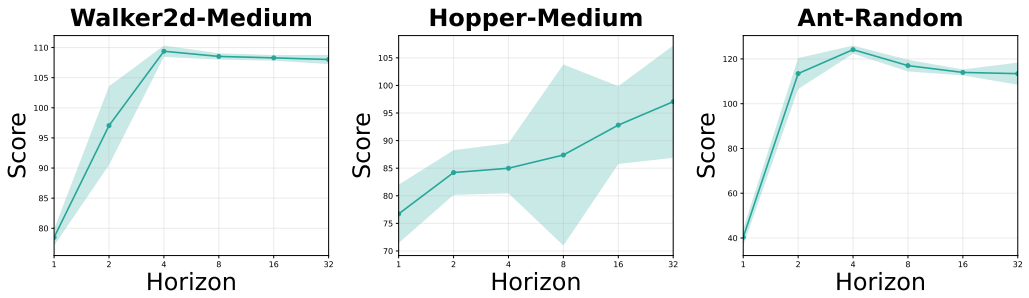


Figure 4: **Ablation of the Temporal Horizon.** The results demonstrate how performance varies with the context horizon H. We observe that larger H generally leads to improved performance, highlighting the importance of temporal context.

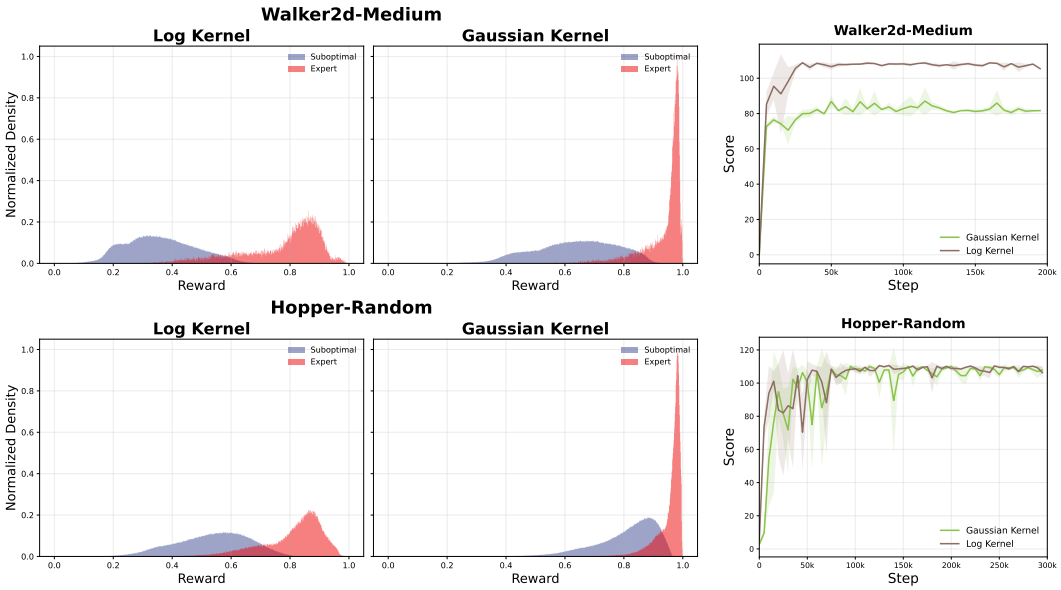
We examine how the length of temporal context used by the generative planner influences reward estimation quality and the performance of the downstream offline RL policy. Specifically, we vary the horizon length H employed during planner training and embedding computation. As shown in Figure 4, using a single-step encoding (H=1) leads to a substantial degradation in the final policy

performance, highlighting the importance of incorporating temporal context along the trajectory. Increasing H yields nearly consistent performance improvements across tasks when H is small, and saturates after certain horizon lengths. This trend suggests that short-to-mid temporal windows are sufficient to capture essential dynamic information. While longer horizons maintain consistent performance without significant degradation, they may introduce unnecessary computation or over-smoothing effects. Based on these observations, we adopt a conservative default of H=32 for a unified and stable hyperparameter choice across all environments and configurations. Additional ablation studies on other aspects of our proposed method is deferred to Appendix C.

C.2 Choice of kernels for reward estimation

We investigate the choice of kernel function $f(d)$ by analyzing the resulting reward distributions and their associated training dynamics. Figure 5(a) visualizes the density of synthetic rewards assigned to expert and suboptimal trajectories in the offline dataset under Logarithmic and Gaussian kernels, respectively.

As shown in Figure 5(a), the Gaussian kernel produces a reward distribution that is highly concentrated in narrow regions: expert rewards collapse near 1.0, forming a sharp peak, while suboptimal trajectories cluster nearby. As a result, meaningful behavioral differences correspond to only minor reward variations, which limits the clarity of the learning signal. In contrast, the Logarithmic kernel (Figure 5(a, left)) yields a broader, heavy-tailed reward distribution. The slow decay better preserves the relative ordering of trajectories in the embedding space, improving reward separability. This enhanced distinguishability facilitates more accurate value estimation by the critic and leads to more stable and optimal learning curves, as shown in Figure 5(b).



(a) Reward Distribution Comparison (b) Performance

Figure 5: **Ablation of the Reward Signal Density.** (Top) Walker2d-Medium. (Bottom) Hopper-Random. Distributions of synthetic rewards assigned to expert (red) and suboptimal (blue) data. Compared to Gaussian-kernel-based reward estimation, the Logarithmic kernel preserves a heavier-tailed reward signal and is associated with improved performance and training stability.

C.3 Learning from fragmented expert demonstrations

We evaluate whether TGE requires complete expert trajectories by constructing fragmented expert demonstrations. We randomly extract non-overlapping contiguous clips of length $H = 32$ from multiple expert episodes, discard their original episode identities and global temporal ordering, and treat each clip as an independent fragment. Embeddings are computed within individual fragments, without crossing fragment boundaries.

We compare three settings: **Complete**, which uses one complete expert episode; **Fragment-1**$\times$, which uses fragments containing the same total number of expert observations; and **Fragment-0.25**$\times$, which retains only 25% of that observation budget.

Table 9: Performance with complete and fragmented expert demonstrations. Fragmented settings retain local continuity while discarding global episode structure.

Environment	Complete	Fragment-1$\times$	Fragment-0.25$\times$
hopper-medium	67.90 ± 1.59	67.80 ± 3.75	67.08 ± 3.70
halfcheetah-medium	48.00 ± 0.30	45.11 ± 0.18	45.22 ± 0.06
walker2d-medium	84.56 ± 0.68	83.67 ± 0.10	82.80 ± 0.56
ant-medium	124.12 ± 1.79	118.45 ± 0.14	121.34 ± 0.76

As shown in Table 9, TGE maintains broadly comparable performance with fragmented demonstrations, although some degradation occurs on individual tasks. Performance also remains close to the complete-episode baseline when the expert observation budget is reduced to 25%. These results suggest that complete expert episodes and global trajectory ordering are not necessary for TGE.

C.4 Sensitivity to expert demonstrations

We examine sensitivity to the number of expert demonstrations and the choice of the expert episode.

Number of expert episodes. We vary the number of expert episodes from 1 to 3, 5, and 10 while keeping the ReBRAC configuration fixed. As shown in Table 10, performance changes little on the two evaluated tasks, suggesting that additional expert episodes provide limited benefit in these settings.

Table 10: Sensitivity to the number of expert episodes.

Environment	1 episode	3 episodes	5 episodes	10 episodes
hopper-medium	67.90 ± 1.59	67.78 ± 2.36	67.92 ± 0.32	68.17 ± 3.35
walker2d-medium	84.56 ± 0.68	83.94 ± 1.07	84.10 ± 0.66	83.74 ± 0.43

Choice of expert episode. We evaluate TGE using five independently selected single expert episodes. Table 11 reports the score for each selection and the sample standard deviation across the five scores. Performance varies modestly across expert selections in the evaluated locomotion tasks, indicating that the results are not specific to a single expert episode.

Table 11: Sensitivity to the choice of a single expert episode. The final column reports the sample standard deviation across the five episode selections, using $\mathrm{ddof} = 1$.

Environment	Episode 0	Episode 1	Episode 2	Episode 3	Episode 4	Std.
hopper-medium	67.90	69.92	64.20	70.60	64.53	2.97
halfcheetah-medium	48.00	47.26	47.25	47.27	47.86	0.37
walker2d-medium	84.56	84.01	84.48	84.37	83.31	0.51
ant-medium	124.12	123.19	121.85	123.82	123.14	0.87

C.5 Adaptive BC weighting in ReBRAC

In our ReBRAC backbone, we weight the behavioral cloning penalty by the normalized TGE reward, scaling the MSE between the actor's actions and the dataset actions according to the expert-likeness of each transition. This serves as a stabilization mechanism analogous to advantage-weighted behavioral cloning. As the surrogate rewards are inferred rather than ground truth, the weighted BC term prevents the policy from deviating too aggressively based on potentially noisy reward estimates in

low-quality regions of $\mathcal{D}_\mu$. Table 12 shows that the weighting improves both training stability and final performance.

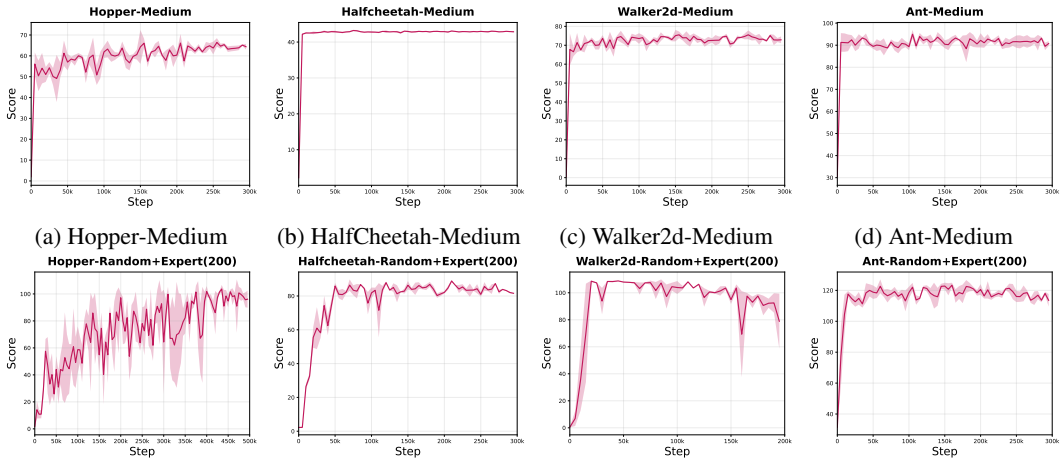
Table 12: Representative results with and without adaptive BC weighting.

Environment	w/o weighting	w/ weighting (default)
walker2d-medium	80.95 ± 1.36	84.56 ± 0.68
hopper-medium	63.14 ± 2.46	67.90 ± 1.59
halfcheetah-medium	44.78 ± 0.08	**48.00 ± 0.30**
halfcheetah-random+few-expert	67.00 ± 8.78	**89.70 ± 1.63**
walker2d-medium+few-expert	109.08 ± 0.59	108.02 ± 0.76
hopper-random+few-expert	91.09 ± 9.24	88.86 ± 12.28
door-human+expert	57.89 ± 5.55	**101.43 ± 3.73**
hammer-human+expert	93.33 ± 7.65	**109.63 ± 9.16**

D Training dynamics visualizations

D.1 TGE with IQL and ReBRAC backbones

We provide the training curves of TGE combined with IQL and ReBRAC as downstream offline RL methods across different environments to demonstrate the training stability of our proposed method. Figure 6 and Figure 7 visualize the normalized D4RL score (y-axis) over training steps (x-axis) for the IQL and ReBRAC backbones, respectively.



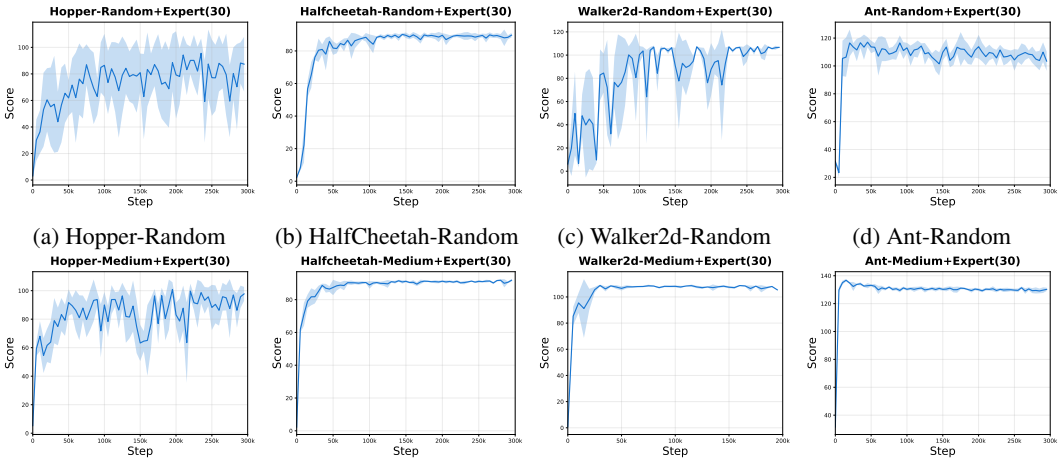
(a) Hopper-Medium (b) HalfCheetah-Medium (c) Walker2d-Medium (d) Ant-Medium

(e) Hopper-Random (f) HalfCheetah-Random (g) Walker2d-Random (h) Ant-Random

Figure 6: **Training Dynamics of TGE + IQL.** The curves display the mean normalized score and standard deviation (shaded region). TGE combined with IQL shows stable policy improvement, confirming that our geometric reward signal enables robust learning across different offline RL methods.

D.2 Comparison with baselines under support mismatch

To complement the comparison in Table 2, we visualize the training dynamics of TGE against offline LfO baselines on the medium+random setting, where the suboptimal dataset contains no expert trajectories. Figure 8 shows that TGE achieves better return and more stable learning compared to other methods across all four locomotion tasks.



(a) Hopper-Random (b) HalfCheetah-Random (c) Walker2d-Random (d) Ant-Random

(e) Hopper-Medium (f) HalfCheetah-Medium (g) Walker2d-Medium (h) Ant-Medium

Figure 7: **Training Dynamics of TGE + ReBRAC.** The curves display the mean normalized score and standard deviation (shaded region). Similarly, the results demonstrate that the agent achieves rapid convergence to expert-level performance and maintains asymptotic stability, effectively robust to the noise in suboptimal datasets.

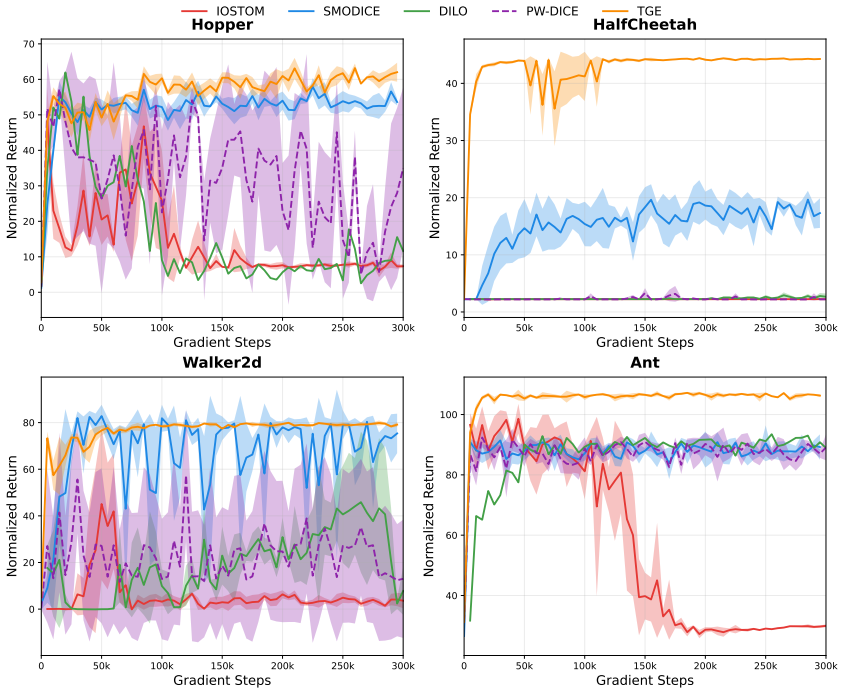


Figure 8: **Training dynamics under support mismatch.** Comparison of TGE with offline LfO baselines on the medium+random setting. TGE maintains strong performance across tasks.

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 abstract and introduction (Section 1) clearly state the three main contributions of TGE. These claims are supported by the theoretical derivation in Section 4.1 and the empirical results in Section 5 across benchmarks.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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:The paper discusses several limitations. Section 5.2.1 acknowledges that TGE's advantage over density-ratio methods diminishes when the offline data already overlaps strongly with the expert distribution, where occupancy-matching baselines can directly exploit the overlap. Appendix A.4 reports the additional pre-processing cost of training the trajectory diffusion model is higher than other baselines.

   Guidelines:

   - The answer [N/A] 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: Section 4.1 presents the entropy-based derivation linking the logarithmic-distance reward to particle-based cross-entropy minimization between the policy and expert occupancies. The derivation builds on the non-parametric entropy estimator and the numerical-stability approximation. All formulas are numbered and referenced where used in Sections 4.3.

   Guidelines:

   - The answer [N/A] 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: Algorithm 1 provides the full pseudocode for TGE. Appendix A.2 lists all hyperparameters for the trajectory diffusion model (Table 3) and the downstream offline RL backbones IQL and ReBRAC (Tables 4 and 5). Appendix A.3 provides implementation details. Appendix B (Table 6) details the data mixture composition for all suboptimal datasets. All experiments use publicly available D4RL benchmarks.

   Guidelines:

   - The answer [N/A] 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: We provide an code in the supplementary material, including scripts to reproduce the main experimental results. All datasets used are publicly available through the D4RL benchmark.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 5.1 describes the environments, expert and suboptimal data composition, baselines, and evaluation protocol. Appendix A.2 reports all hyperparameters for the trajectory diffusion model (Table 3), IQL (Table 4), and ReBRAC (Table 5). Appendix B (Table 6) details the data mixture strategy for all suboptimal datasets. For baseline methods, we follow the original implementations and hyperparameter settings reported in their respective papers.

   Guidelines:

   - The answer [N/A] 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: All reported results in tables are mean $\pm$ standard deviation across five random seeds. Training curves in Figures 3, 6, 7, and 8 show the mean and standard deviation across seeds.

   Guidelines:

   - The answer [N/A] 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 A.4 reports the compute requirements for the main stages of TGE.

   Guidelines:

   - The answer [N/A] 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: The research conforms with the NeurIPS Code of Ethics. The work uses standard publicly available simulation benchmarks. Anonymity has been preserved in the submission.

   Guidelines:

- The answer [N/A] 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: This paper presents foundational research on offline imitation learning, evaluated on standard simulation benchmarks. On the positive side, improvements in sample-efficient imitation from observation-only data could benefit downstream applications such as robot learning from human demonstrations. The work is not tied to a specific deployed system, and we do not foresee direct negative societal impacts beyond those generally associated with advances in reinforcement learning and robotics.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release pre-trained language models, image generators, scraped datasets, or any other assets with a high risk of misuse. The released code implements an offline imitation learning algorithm for standard simulation benchmarks and poses no such risks.

    Guidelines:

    - The answer [N/A] 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: All existing assets used in the paper are properly cited with their original publications.

    Guidelines:

    - The answer [N/A] 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: We release the code implementing TGE as an anonymized zip file in the supplementary material. The repository includes a README with installation instructions, scripts to reproduce the main experimental results, and the hyperparameter configurations reported.

    Guidelines:

    - The answer [N/A] 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: [N/A]

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

Guidelines:

- The answer [N/A] 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: [N/A]

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

    Guidelines:

    - The answer [N/A] 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.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not used as a component of the core methodology.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.