UNSUPERVISED GENERATIVE REPLAY FOR EFFICIENT REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Leveraging the diffusion model to augment the replay buffer has become a useful paradigm to enhance the efficiency of reinforcement learning (RL). Can we pre-train a diffusion model from unlabeled exploratory data such that it can be adapted to new downstream tasks to enrich the RL buffer, even when the downstream reward function cannot be queried on these pre-collected data? In this work, we present an **U**nsupervised **G**enerative **R**eplay (UGR), a reusable generative replay framework for this setting. UGR pre-trains a diffusion model on exploratory transitions relabeled with diverse reward priors to serve as unsupervised generative priors. When adapting to a specific task, we only train a lightweight guidance model to steer data generation towards the target task. Experiments across complex control tasks on DeepMind Control Suite reveal that UGR effectively improves online data efficiency under a fixed interaction budget. These results suggest that reusing unsupervised generative priors is a promising way to improve online RL across downstream tasks.

1 INTRODUCTION

Reinforcement learning (RL) relies critically on the availability of diverse and informative experience. Unsupervised RL (URL) (Eysenbach et al., 2018; Laskin et al., 2021) provides an effective method to acquire diverse trajectories. Relying on informative-related intrinsic reward, the unsupervised policy is driven to explore heterogeneous regions that are rarely visited by a supervised policy Ying et al. (2026). Nevertheless, the effective application of knowledge derived from unsupervised exploration to specific tasks constitutes a pivotal issue. Previous works assume the reward function in the target task is known and focus on training the agent on the relabeled exploratory trajectories. However, in many real-world scenarios, the underlying mapping from the observation space to the goal signal in a specific task is not readily accessible. How to leverage the trajectories induced by the Unsupervised agent to facilitate online policy learning remains an open question (Yarats et al., 2022; Frans et al., 2024).

Recently, diffusion models (DM) have achieved remarkable success in modeling complex data distributions and generating various types of synthetic data, such as vision (Ho et al., 2020; Song et al., 2021; Karras et al., 2022; Ho et al., 2022), multimodal knowledge (Ramesh et al., 2022; Ruiz et al., 2023; Saharia et al., 2022), and time series (Tashiro et al., 2021; Gao et al., 2025). DMs are capable of fitting complex data priors, and under the guidance of auxiliary conditional signals or utility functions, they yield diversified data conforming to a given distribution (Ho & Salimans, 2022; Ouyang et al., 2024). In RL domains, their powerful modeling capacity assists the agent in improving the policy by fitting the replay buffer and synthesizing a specific data distribution (Lu et al., 2023b; Wang et al., 2025; 2026; Ying et al., 2026; Huang et al., 2025).

In this paper, we study how DMs can distill reusable knowledge from broad and diverse unsupervised exploration trajectories, and introduce **U**nsupervised **G**enerative **R**eplay (UGR) to accelerate online policy learning in specific tasks. Unlike existing DM-based generative buffers that primarily augment replay using limited observed target-task experience, UGR treats the DM as a plug-and-play unsupervised generative prior. Specifically, UGR is first pre-trained on exploratory datasets with functionally randomized rewards to capture the support of behaviorally diverse and dynamically feasible transitions. During policy learning, when the downstream reward function cannot be queried on the reward-free exploratory data, this prior is guided by a learned utility function

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

to synthesize task-relevant experience and enrich the replay buffer. This design separates *coverage* from *task alignment*: unsupervised exploration and random functional reward provide broad transition support, diffusion pre-training distills this support into a reusable generative prior, and target-specific guidance steers generated samples toward the desired target distribution. By separating prior training from task-specific adaptation, UGR offers a potential route to reusing generative replay across downstream tasks.

We formulate UGR using a transfer-score perspective for diffusion models. Building on this general perspective, we instantiate source and downstream distributions over reward-annotated RL transitions and develop a practical utility-learning procedure for adapting a pre-trained exploratory prior. The resulting guidance is used to generate synthetic replay for downstream online policy learning. Consequently, learning a target-specific diffusion model is reduced to estimating this density-ratio correction, which can be efficiently optimized with a contrastive learning objective and interpreted as energy-based guidance during reverse diffusion. Compared with training a separate diffusion model for each target task, UGR reuses a single source generator and only learns a lightweight target-specific utility or guidance model. This enables efficient target-guided synthetic replay generation while preserving the broad coverage acquired from unsupervised exploration.

Overall, our contributions can be summarized as follows: (i) We introduce **U**nsupervised **G**enerative **R**eplay (UGR). This diffusion-based generative replay framework distills diverse unsupervised exploration trajectories into a reusable generative prior for online RL, thereby separating exploratory transition modeling from downstream task adaptation. (ii) We develop a utility-learning procedure that adapts the pretrained diffusion prior to downstream tasks using limited online experience, without fine-tuning a separate diffusion model using limited online data. (iii) We evaluate UGR on challenging continuous-control tasks on 3 domains from DMC suite. Empirical results demonstrate the effectiveness of UGR for sample-efficient RL.

2 Background

2.1 RL and Unsupervised Exploratory Data

RL considers Markov decision processes (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. Here $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, respectively. During interaction, the action $a \in \mathcal{A}$ leads to a next state $s' \sim \mathcal{P}(\cdot|s, a)$ and an extrinsic task reward $\mathcal{R}(s, a)$. γ is the discount factor. For a given policy $\pi(a|s)$, we define the discount state distribution of π at state s as $d_\pi(s) = (1-\gamma)\sum_{t=0}^{\infty}[\gamma^t \mathcal{P}(s^t = s)]$. The objective of RL is to learn a policy π that maximizes the expected cumulative reward $\mathcal{R}$:

$$J(\pi) \triangleq \mathbb{E}_{\tau\sim\mathcal{M},\pi}[\mathcal{R}(\tau)] = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_\pi, a\sim\pi}[\mathcal{R}(s, a)]. \tag{1}$$

In URL, the agent is encouraged to explore the environment $\mathcal{M}$ broadly, producing diverse behavioral trajectories that cover a wide range of states, actions, and transitions. Importantly, these trajectories are not associated with extrinsic task reward $\mathcal{R}$. Instead, URL requires designing intrinsic rewards $\mathcal{R}_{\text{int}}$ to guide policies to maximize the state entropy $\mathcal{H}(d_\pi(\cdot))$. This makes them a task-agnostic source of experience: while they capture rich information about the environment dynamics and feasible behaviors, they remain flexible enough to be relabeled, reweighted, or guided toward different target objectives during online learning.

2.2 Diffusion Models

Diffusion models (Ho et al., 2020; Karras et al., 2022) are a class of generative models inspired by non-equilibrium thermodynamics. Consider a data distribution $p(\mathbf{x})$ with standard deviation σ_{data}, diffusion models gradually add i.i.d. Gaussian noise of standard deviation σ on the base distribution from time 0 to K and obtain noised distributions $p(\mathbf{x}; \sigma)$. The forward noising process is defined by a sequence of noised distributions following a fixed noise schedule $\sigma_0 = \sigma_{\max} > \sigma_1 > \cdots > \sigma_N = 0$ so that at each noise level, $\mathbf{x}^k \sim p(\mathbf{x}^k; \sigma_k)$. When $\sigma_{\max} \gg \sigma_{\text{data}}$, the final noised distribution $p(\mathbf{x}^K; \sigma_{\max})$ is essentially indistinguishable from random noise. The diffusion model is trained to iteratively denoise samples from a Gaussian distribution and ultimately recover the target distribution, which is formally named the reverse process. Karras et al. (2022) consider this process

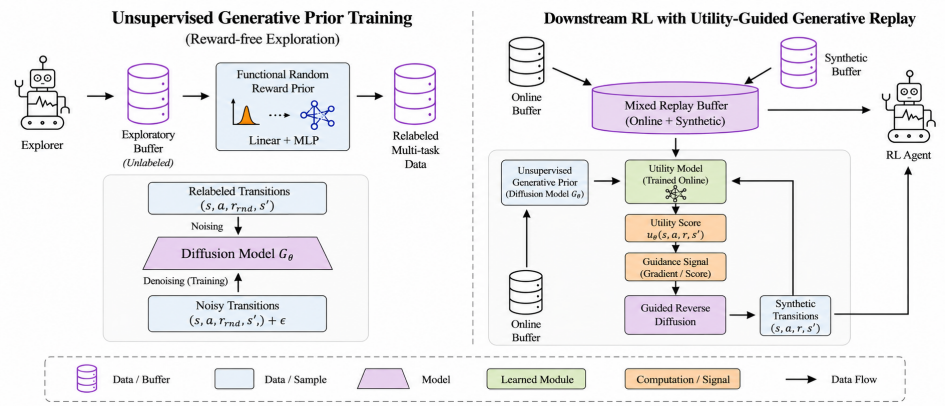


Figure 1: **Overview of UGR.** UGR first relabels exploratory transitions with sampled reward functions and trains a diffusion model on the resulting data. During online learning, it reuses this generative prior and learns a task-specific utility model from online experience to guide synthetic replay generation. The generated transitions are combined with online data to train the policy.

as a probability-flow ODE and formulate as below:

$$\mathrm{d}\mathbf{x} = -\dot{\sigma}(k)\sigma(k)\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma(k))\mathrm{d}k, \tag{2}$$

where $\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma(k))$ denotes the score function, which points towards the data for a given noise level, and the dot indicates a time derivative. The denoiser $G_\theta(\mathbf{x}_t; \sigma)$ is trained on an L2 denoising minimization objective:

$$\mathcal{L}(G_\theta; \sigma) = \mathbb{E}_{\mathbf{x}\sim p, \epsilon\sim\mathcal{N}(0,\sigma^2 I)} \|G_\theta(\mathbf{x} + \epsilon; \sigma) - \mathbf{x}\|_2^2, \tag{3}$$

and the score can be calculated by $\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma) = (G_\theta(\mathbf{x}; \sigma) - \mathbf{x})/\sigma^2$. In this paper, we sample data by solving Eq. 2 with the learned denoising network.

2.3 Problem Formulation

We consider online RL with access to reward-free exploratory data collected before task-specific learning. The exploratory data provide broad coverage of transitions, but do not contain reward signals for the specific task. Our goal is to utilize such task-agnostic experience to improve online data efficiency when the specific reward function cannot be queried on these data.

Formally, each task is modeled as an MDP $\mathcal{M}_\omega = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R_\omega, \gamma)$, where R_ω is the task-specific reward function. Before online learning, the agent is given a reward-free exploratory dataset $\mathcal{D}^{\text{expl}} = \{(s, a, s')\}_N$, collected under the same transition dynamics but without task-specific reward labels. During online learning, the agent interacts with the environment and observes rewards only for its own online transitions, forming a limited replay $\mathcal{D}^{\text{on}} = \{(s, a, r_\omega, s')\}_N$. Importantly, although the agent can observe rewards from online interactions, it cannot query the real R_ω. Therefore, the exploratory dataset cannot be directly relabeled with the reward. The problem is to use the reward-free dataset $\mathcal{D}^{\text{expl}}$, together with limited online experience $\mathcal{D}^{\text{on}}$, to improve policy learning.

3 Unsupervised Generative Replay Process

Our method, Unsupervised Generative Replay (UGR), accelerates online RL by turning reward-free exploratory data into task-relevant synthetic replay. As illustrated in Fig. 1, UGR first relabels exploratory transitions with sampled reward functions and trains a diffusion model on the resulting data. During online learning, it reuses this generative prior and learns a task-specific utility model from online experience to guide synthetic replay generation. The generated transitions are combined with online data to train the policy.

3.1 METHODOLOGY FORMULATION

Suppose that all downstream tasks share the same environment transition dynamics and differ in their reward functions. Let $z = (s, a, s')$ denote a transition, and let $\mu_{\text{expl}}(z)$ be the transition distribution represented by the pre-collected reward-free dataset $\mathcal{D}^{\text{expl}}$. During pre-training, we sample transitions from $\mathcal{D}^{\text{expl}}$ and annotate them with reward functions sampled from a task-agnostic prior Π_S. Writing the reward-annotated transition as $\mathbf{x} = (z, r)$, the resulting source distribution is,

$$p_S(\mathbf{x}) = \mu_{\text{expl}}(z) \int \delta(r - \mathcal{R}_\xi(z))\, d\Pi_S(\xi). \tag{4}$$

In practice, we approximate p_S by repeatedly sampling z from $\mathcal{D}^{\text{expl}}$, sampling $\xi \sim \Pi_S$, and setting $r = \mathcal{R}_\xi(z)$. The reward functions used in this step provide synthetic labels and do not induce separate transition-occupancy distributions. For a downstream task ω, let $\mu_\omega(z)$ denote the online experience distribution. The agent observes the task reward $r = \mathcal{R}_\omega(z)$ for its online experience, but cannot query $\mathcal{R}_\omega$ on transitions in $\mathcal{D}^{\text{expl}}$. Then the corresponding online distribution is,

$$q_\omega(\mathbf{x}) = \mu_\omega(z)\delta(r - \mathcal{R}_\omega(z))\,. \tag{5}$$

Thus, the source distribution p_S consists of exploratory transitions annotated with rewards sampled from the reward prior, whereas the target distribution q_ω consists of reward-annotated transitions collected online for task ω. We train an unconditional diffusion model on samples from p_S to obtain a reusable generative prior. Let $\mathbf{x}^k$ denote a sample at diffusion step k. After training, the model estimates the source-domain noisy score:

$$G_\theta(\mathbf{x}^k, k) \approx \nabla_{\mathbf{x}^k} \log p_{S,k}(\mathbf{x}^k). \tag{6}$$

For a downstream task ω, our goal is to adapt this prior toward the target distribution q_ω. Rather than training a separate diffusion model for each task, we estimate the target-to-source density ratio

$$g_\omega(\mathbf{x}) = q_\omega(\mathbf{x})/p_S(\mathbf{x}). \tag{7}$$

Density-ratio guidance requires support overlap between the target and source distributions over $x = (z, r)$, i.e., $q_\omega \ll p_S$. We assume this condition for the downstream tasks considered and use broad exploratory data with diverse reward annotations to promote coverage of task-relevant regions. Under this assumption and a shared forward noising kernel, the transfer-score identity relates the target noisy score to the pretrained source score and a task-specific correction. We state this identity formally in Thm. 1.

Theorem 1 (Transfer score identity)**.** *Let p_S and q_ω be source and target distributions satisfying $q_\omega \ll p_S$. Suppose that both distributions are perturbed using the same forward noising kernel $K_k(\mathbf{x}^k|\mathbf{x}^0)$. Then the target noisy score satisfies*

$$\nabla_{\mathbf{x}^k} \log q_{\omega,k}(\mathbf{x}^k) = \nabla_{\mathbf{x}^k} \log p_{S,k}(\mathbf{x}^k) + \nabla_{\mathbf{x}^k} \log \mathbb{E}_{p_S(\mathbf{x}^0|\mathbf{x}^k)} \left[q_\omega(\mathbf{x}^0)/p_S(\mathbf{x}^0)\right]. \tag{8}$$

We include the derivation for completeness in Appendix A. Thm. 1 is the theoretical starting point for our method. In UGR, the source score is estimated by a diffusion model pre-trained on reward-relabeled exploratory transitions. The task-dependent correction is approximated using a parametric density-ratio estimator and utility model, which we use to guide generation. UGR constructs the target score by combining a fixed unsupervised generative prior with a task-specific utility model. The prior diffusion model is trained once on the exploratory dataset relabeled by functional random rewards, while the utility model is trained online to steer generation toward the target distribution. This design avoids task-specific diffusion training on limited online data.

3.2 UNSUPERVISED GENERATIVE PRIOR

To train the diffusion model as an *unsupervised generative prior*, we relabel reward-free exploratory data with a diverse prior over rewards. The choice of this prior is important: fully random rewards may induce unstructured signals, while appropriately structured priors can encourage better generalization. We therefore seek a domain-agnostic reward prior that remains broad enough to cover plausible downstream tasks (Frans et al., 2024; Wolpert & Macready, 1997).

In our implementation, we instantiate this prior with a mixture of simple yet expressive random reward functions, including random linear functions and two-layer MLPs. Linear rewards provide

low-complexity signals that capture broad directional preferences, whereas MLP-based rewards introduce nonlinear, more localized objectives. Relabeling exploratory trajectories with these functions $\mathcal{R}^{\text{func}}$ yields a diverse multi-task distribution over reward-annotated transitions. We then train the diffusion model on the relabeled data using the standard denoising objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}^{\text{expl}},\mathcal{R}^{\text{func}},k,\epsilon}\big[\|\epsilon_\theta(\mathbf{x}^k, k) - \epsilon\|^2\big], \tag{9}$$

where $\mathcal{D}^{\text{expl}}$ denotes the exploratory dataset, and $\mathbf{x}^k$ is the noisy version of the relabeled transition at diffusion step k. Training on this relabeled exploratory dataset enables the diffusion model to distill feasible behaviors, environment dynamics, and diverse task signals into a single reusable model. We refer to the model as an *unsupervised generative prior*, since it is learned without access to downstream tasks and can later be guided toward task-specific replay distributions.

3.3 Utility Model in Downstream Tasks

The key challenge in downstream deployment is to guide the pre-trained unsupervised generative prior toward the target-task replay distribution. Given the multi-task source distribution $p_S(X)$ and the target-task distribution $q_\omega(\mathbf{x})$, the ideal guidance is determined by the density ratio $q_\omega(X)/p_S(X)$. To obtain a practical guidance model, we adopt an energy-based formulation and use contrastive energy prediction (CEP) (Lu et al., 2023a) to learn its noisy-time counterpart. We rewrite the density ratio as a Boltzmann distribution,

$$g_\omega(\mathbf{x}) = q_\omega(\mathbf{x})/p_S(\mathbf{x}) = \exp(-\mathcal{E}_\omega(\mathbf{x}))/Z_\omega. \tag{10}$$

Therefore, the energy model is defined by

$$\mathcal{E}_\omega(\mathbf{x}) = -\log[q_\omega(\mathbf{x})/p_S(\mathbf{x})] + C, \tag{11}$$

where C is a constant. Follow Eq. 11 we have $q_\omega(\mathbf{x}) \propto p_S(\mathbf{x})\exp(-\mathcal{E}_\omega(\mathbf{x}))$. For the time step t in the reverse process, the energy model is approximated by:

$$\mathcal{E}_{\omega,k}(\mathbf{x}^k) = -\log \mathbb{E}_{p_S(\mathbf{x}^0|\mathbf{x}^k)}\left[\exp(-\mathcal{E}_\omega(\mathbf{x}^0))\right]. \tag{12}$$

Then we have $q_{\omega,k}(\mathbf{x}^k) \propto p_{S,k}(\mathbf{x}^k)\exp(-\mathcal{E}_{\omega,k}(\mathbf{x}^k))$ and rewrite the score in the target domain,

$$G_{\theta^*}(\mathbf{x}^k, k) = \nabla_{\mathbf{x}^k}\log p_{S,k}(\mathbf{x}^k) - \nabla_{\mathbf{x}^k}\mathcal{E}_{\omega,k}(\mathbf{x}^k). \tag{13}$$

The pre-trained diffusion model preserves feasible transition structure, while the utility model biases generation toward the target distribution. Concretely, for M samples $\mathbf{x}^0_{1:M}$ with their noisy version $\mathbf{x}^k_{1:M}$ at step k in the diffusion process, $\epsilon^{1:M} \sim \mathcal{N}(0, \sigma^2 I)$ are M i.i.d. Gaussian samples, f_ψ is trained to minimize the distance between the distribution of the target energy function,

$$\mathcal{L}(\psi) = \mathbb{E}_{k,\mathbf{x}^0_{1:M},\epsilon^{1:M}\sim\mathcal{N}(0,\sigma^2 I)}\Big[-\sum_{i=1}^{M} \underbrace{e^{-\beta\mathcal{E}(\mathbf{x}_i^0)}}_{\text{softenergylabel}} \log \underbrace{\frac{e^{f_\psi(\mathbf{x}_i^k,k)}}{\sum_{j=1}^M e^{f_\psi(\mathbf{x}_j^k,k)}}}_{\text{predictedlabel}}\Big], \tag{14}$$

Based on Theorem 3.2 in CEP, the gradient of f_ψ is equal to the real intermediate energy gradient of $\mathcal{E}_\omega$ under infinite samples. However, the unnormalized term $\exp(-\beta\mathcal{E}(\mathbf{x}_i^0))$ may cause numerical instability, especially when $\mathcal{E}(\mathbf{x}^0)$ contains sharp spikes that are amplified by the exponential. Following Lu et al. (2023a), we use a self-normalized energy label by normalizing the energy values across the M samples, and define the optimization problem as:

$$\mathcal{L}(\psi) = \mathbb{E}_{k,\mathbf{x}^0_{1:M},\epsilon^{1:M}\sim\mathcal{N}(0,\sigma^2 I)}\Big[-\sum_{i=1}^{M} \underbrace{\frac{e^{-\beta\mathcal{E}(\mathbf{x}_i^0)}}{\sum_{j=1}^M e^{-\beta\mathcal{E}(\mathbf{x}_j^0)}}}_{\text{self-normalizedenergylabel}} \log \underbrace{\frac{e^{f_\psi(\mathbf{x}_i^k,k)}}{\sum_{j=1}^M e^{f_\psi(\mathbf{x}_j^k,k)}}}_{\text{predictedlabel}}\Big]. \tag{15}$$

Using the energy model definitions in Eqs. 11 and 12, we can replace the energy with the density ratio q_ω/p_S. We estimate the target-to-source density ratio using a binary classifier $c_q(x) = P(y = 1|x)$, where $y = 1$ denotes a target sample and $y = 0$ denotes a source sample. Under balanced sampling of the two classes, Bayes' rule gives,

$$\hat{g}_\omega(\mathbf{x}) \triangleq q_\omega(\mathbf{x})/p_S(\mathbf{x}) = c_q(\mathbf{x})/(1 - c_q(\mathbf{x})). \tag{16}$$

Equivalently, the classifier logit estimates the log density ratio,

$$\ell_\omega(\mathbf{x}) \triangleq \texttt{logit}(c_q(\mathbf{x})) = \log \hat{g}_\omega(\mathbf{x}). \tag{17}$$

We use this logit to construct the self-normalized weights in the clean space. In particular, by applying a temperature β to the classifier logit, the practical objective for the utility model f_ψ is,

$$\mathcal{L}(\psi) = \mathbb{E}_{k,x_0^{1:M},\epsilon^{1:M}\sim\mathcal{N}(0,\sigma^2 I)}\Big[-\sum_{i=1}^{M} \underbrace{\frac{e^{\beta\ell_\omega(\mathbf{x}_i)}}{\sum_{j=1}^{M} e^{\beta\ell_\omega(\mathbf{x}_j)}}}_{\mathrm{self-normalized energy label}} \log \underbrace{\frac{e^{f_\psi(\mathbf{x}_i^k,k)}}{\sum_{j=1}^{M} e^{f_\psi(\mathbf{x}_j^k,k)}}}_{\mathrm{predicted label}}\Big]. \tag{18}$$

This utility model serves as a guidance function during diffusion sampling, steering the unsupervised generative prior toward the task-specific replay distribution.

3.4 Noisy-dependent Cycle Regularization

A limitation of training the utility model on clean labels is the mismatch between the training target and the quantity required during reverse diffusion. Specifically, given a source sample $\mathbf{x}^0 \sim p_S$ and its noisy version $\mathbf{x}^k$, fitting

$$f_\psi(\mathbf{x}^k, k) \propto -\mathcal{E}_\omega(\mathbf{x}^0) = \log[q_\omega(\mathbf{x}^0)/p_S(\mathbf{x}^0)] \tag{19}$$

uses a clean-space density ratio as supervision for a noisy-space input. However, the guided reverse process requires the marginal density ratio at diffusion step k:

$$-\mathcal{E}_{\omega,k}(\mathbf{x}^k) = \log[q_{\omega,k}(\mathbf{x}^k)/p_{S,k}(\mathbf{x}^k)]. \tag{20}$$

These two quantities are generally different, since a noisy sample $\mathbf{x}^k$ may correspond to multiple possible clean ancestors with different clean density ratios. To reduce this mismatch and improve the utility model's target adaptation, we introduce a noisy-dependent cycle regularization. We additionally align the utility model with the noisy-time marginal density ratio. For the source and target noisy marginals $p_k(\mathbf{x}^k)$ and $q_k(\mathbf{x}^k)$, we have:

$$\mathbb{E}_{p_S(\mathbf{x}^0|\mathbf{x}^k)}\left[q_\omega(\mathbf{x}^0)/p_S(\mathbf{x}^0)\right] = q_{\omega,k}(\mathbf{x}^k)/p_{S,k}(\mathbf{x}^k). \tag{21}$$

This identity suggests that the utility model can be regularized by matching the noisy-time density ratio. Specifically, given source and target samples, both perturbed to diffusion step k, we train a noisy classifier $c_{\omega,k}(\mathbf{x}^k)$ by BCE loss to distinguish noisy samples from source and target distributions. The noisy-time density ratio can be estimated as,

$$q_{\omega,k}(\mathbf{x}^k)/p_{S,k}(\mathbf{x}^k) = c_{\omega,k}(\mathbf{x}^k)/(1 - c_{\omega,k}(\mathbf{x}^k)). \tag{22}$$

Similar to the self-normalized weights in the clean space, we have the noisy log density-ratio,

$$\ell_{\omega,k}(\mathbf{x}^k) \triangleq \texttt{logit}(c_{\omega,k}(\mathbf{x}^k)) = \log q_{\omega,k}(\mathbf{x}^k)/p_{S,k}(\mathbf{x}^k). \tag{23}$$

We then use this estimated ratio to regularize the guidance network f_ψ. Specifically, $f_\psi(\mathbf{x}^k, k)$ is encouraged to match the noisy-time density ratio:

$$\mathcal{L}_{\mathrm{cycle}}(\psi) = \mathbb{E}_{k,x_0^{1:M},\epsilon^{1:M}\sim\mathcal{N}(0,\sigma^2 I)}\Big[-\sum_{i=1}^{M} \underbrace{\frac{e^{\beta\ell_{\omega,k}(\mathbf{x}_i^k)}}{\sum_{j=1}^{M} e^{\beta\ell_{\omega,k}(\mathbf{x}_j^k)}}}_{\mathrm{self-normalized energy label}} \underbrace{\log \frac{e^{f_\psi(\mathbf{x}_i^k,k)}}{\sum_{j=1}^{M} e^{f_\psi(\mathbf{x}_j^k,k)}}}_{\mathrm{predicted label}}\Big]. \tag{24}$$

and the utility model is trained to approximate both clean and noisy samples' density ratio,

$$\mathcal{L}_{\mathrm{utility}}(\psi) = \mathcal{L}_{\mathrm{clean}}(\psi) + \mathcal{L}_{\mathrm{cycle}}(\psi), \tag{25}$$

where $\mathcal{L}_{\mathrm{clean}}(\psi)$ is Eq. 18. This cycle regularization improves the consistency of guidance across clean and noisy diffusion spaces. In practice, this alignment stabilizes guidance at high noise levels, reduces conflicting clean-label supervision, and produces synthetic replay that better matches the online replay distribution. Therefore, we use the following guided score estimate,

$$\hat{G}_{\theta,\psi}(\mathbf{x}^k, k) \triangleq \nabla_{\mathbf{x}^k} \log p_{S,k}(\mathbf{x}^k) + \eta \nabla_{\mathbf{x}^k} f_\psi(\mathbf{x}^k, k). \tag{26}$$

Here, η is the guidance scale. Since the utility model is trained with β-scaled density-ratio weights, $\hat{G}_{\theta,\psi}$ serves as an approximate target-score estimate. We use it to generate task-specific synthetic transitions, which are combined with online samples to train the RL agent.

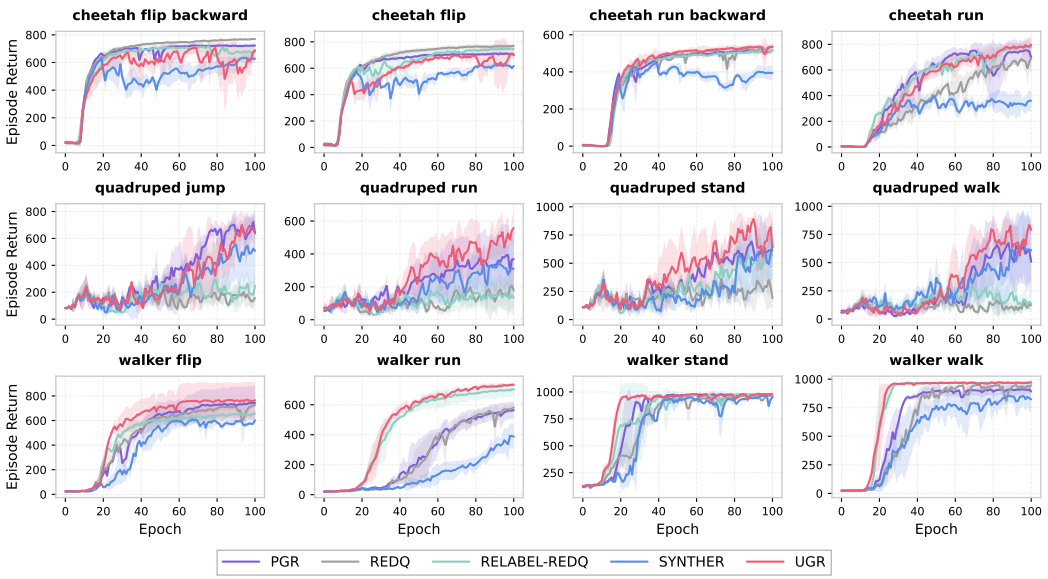


Figure 2: Training curves on DMC suite. The solid lines depict the mean performance, while the shaded areas represent the confidence intervals over three seeds.

3.5 Framework Summary

Algorithm 1 summarizes the overall UGR pipeline. UGR consists of two phases. In the pre-training phase, we train a diffusion model on exploratory data using Eq. 9, yielding an unsupervised generative prior. In the downstream phase, we learn a utility model with Eq. 25 and use it to guide the pre-trained prior through the practical score in Eq. 26. The resulting guided diffusion model generates task-relevant synthetic replay, which is mixed with real online samples to form the policy training batch. Since UGR only augments the replay, it is plug-and-play and does not require modifying the backbone RL algorithm.

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluate our method on a diverse benchmark of multiple challenging locomotion tasks drawn from the DeepMind Control Suite (DMC) (Tassa et al., 2018). The DMC tasks feature diverse tasks inspired by ExORL (Yarats et al., 2022), including 3 domains: Walker, Cheetah, and Quadruped. Each domain involves 4 different tasks.

Baselines. We use a widely adopted RL algorithm as the primary baseline for UGR: **REDQ** (Chen et al., 2021), a stochastic policy optimization method that enables efficient online learning through entropy-regularized exploration and a high update-to-data (UTD) ratio. We also consider a variant, denoted as **Relabel-REDQ**, which trains a reward model on online replay to relabel the exploratory dataset for policy learning. To further assess the effectiveness of UGR, we compare against representative DM-based replay methods, including **SynthER** (Lu et al., 2023b) and **PGR** (Wang et al., 2025). SynthER trains an unconditional DM to approximate the observed buffer, while PGR follows a CFG paradigm to synthesize high-curiosity data from the observed buffer.

Implementation details. In this paper, the total training step size for all experiments is set at 100K, with the results of all experiments averaged over 3 random seeds. We use the dataset provided by ExORL (Yarats et al., 2022) using Proto (Yarats et al., 2021). The architecture of the diffusion model

Table 1: Average returns on 12 DMC tasks after 100K environment steps (3 seeds, 1 std. dev. err.). We highlight the best scores in **bold** and shade the UGR score that is better than those of other diffusion-based methods.

Task	REDQ	Relabel-REDQ	SynthER	PGR	UGR (Ours)
Walker Stand	974.8±2.2	984.7±6.9	972.5±3.3	981.9±6.7	**988.1±2.0**
Walker Walk	961.5±4.4	**977.1±3.5**	872.2±95.3	916.7±53.7	974.7±1.0
Walker Run	592.4±33.4	710.9±64.2	431.8±103.1	568.9±69.2	**740.3±10.3**
Walker Flip	730.1±20.9	663.0±27.4	620.3±24.3	752.0±166.2	**777.5±177.6**
Cheetah Run	713.4±44.6	797.4±29.1	417.9±81.0	771.7±100.1	**829.7±32.5**
Cheetah Flip	**771.9±20.0**	754.2±22.5	639.6±8.0	717.5±13.9	726.4±16.1
Cheetah Run Backward	**545.6±1.1**	519.9±5.6	411.2±41.1	520.6±16.7	541.4±14.9
Cheetah Flip Backward	**772.4±2.0**	687.9±74.1	649.4±38.5	725.6±4.9	712.1±21.1
Quadruped Stand	487.3±274.7	772.6±307.7	720.7±324.6	758.6±131.4	**865.4±169.5**
Quadruped Jump	309.2±166.0	400.2±75.1	614.8±236.3	761.7±11.6	**798.1±64.5**
Quadruped Walk	255.3±14.9	311.1±194.2	725.6±322.4	757.6±315.9	**922.8±15.8**
Quadruped Run	281.9±193.6	217.1±73.9	390.0±123.5	423.3±209.5	**608.1±109.4**
DMC Suite Average	616.3 (+0%)	649.7 (+5.4%)	622.2 (+1.0%)	721.3 (+17.0%)	**790.4** (+28.3%)

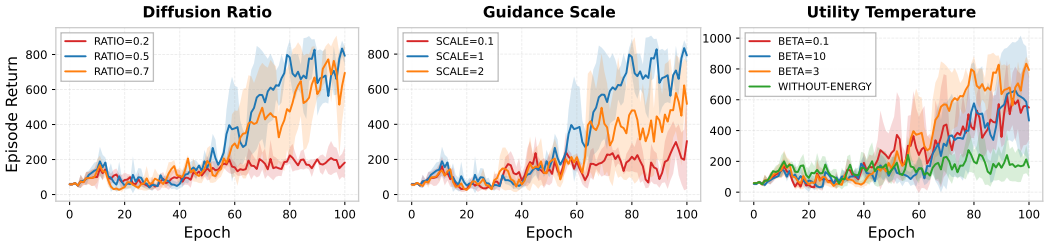


Figure 3: Ablation results on the Quadruped Walk task. The solid lines depict the mean performance, while the shaded areas represent the confidence intervals over three seeds.

G_θ is aligned with Wang et al. (2025). All hyperparameters are aligned with standard implementations, and the configuration details are documented in Appendix C.

4.2 Experimental Results

Figure 2 shows the training curves, and Table 1 reports the corresponding numerical results. Under the limited 100K online interactions, UGR achieves the highest mean return across the 3 domains, improving over REDQ by $+28.3\%$ and over the strongest diffusion-based baseline, PGR, by $+9.6\%$.

UGR obtains the best score on 8 out of 12 tasks and consistently improves over diffusion-based replay baselines on most environments. The gains are especially clear on challenging Quadruped tasks, where UGR outperforms PGR by large margins on Stand, Jump, Walk, and Run. These results suggest that the unsupervised generative prior provides broad transition coverage, while utility guidance effectively steers synthetic replay toward task-relevant regions.

The learning curves further show that UGR achieves higher returns under the evaluated online interaction budget. Compared with SynthER, which learns generative replay from the online buffer, UGR benefits from pre-training on reward-free exploratory data. Compared with PGR, UGR uses utility guidance to generate replay that is more aligned with current policy learning. Although UGR is not uniformly best on every individual task, its strong aggregate performance supports the effectiveness of combining unsupervised generative pre-training with target-aware replay generation.

4.3 Ablation Study

Synthetic replay ratio. The amount of synthetic replay needs to be balanced with real online samples. As shown in Fig. 3 (**Left**), using a small ratio (0.2) brings only limited gains, while increasing it to 0.5 yields the best learning speed and final return. A larger ratio (0.7) degrades

performance, suggesting that excessive generated data can bias policy updates away from the online distribution. This indicates that UGR benefits most from a moderate synthetic replay ratio.

Guidance scale. The effectiveness of utility-guided generation also depends on the strength of the guidance signal. We present the results with different guidance scales in Fig. 3 (**Middle**). With weak guidance (0.1), the generated samples remain close to the pre-trained prior and are not sufficiently adapted to the current task. A moderate scale (1.0) achieves the strongest performance, whereas a larger scale (2.0) leads to lower final returns. These results suggest that utility guidance should be strong enough to induce task alignment, but not so strong as to distort the learned prior.

Utility temperature β. The utility temperature controls the sharpness of the energy-based guidance target. Based on the results in Fig. 3 (**Right**), a small value ($\beta = 0.1$) produces weak task preference and leads to slower improvement, while an overly large value ($\beta = 10$) makes guidance more aggressive and less stable. The moderate setting ($\beta = 3$) achieves the best performance, suggesting that calibrated utility weighting is important for selecting useful synthetic transitions. Removing the energy guidance substantially degrades performance, confirming that the pre-trained diffusion prior alone is insufficient for generating task-relevant replay.

5 Related Work

Diffusion models have shown strong capacity in modeling complex, high-dimensional distributions (Ho et al., 2020; Saharia et al., 2022; Nichol et al., 2022; Nichol & Dhariwal, 2021; Song et al., 2023), and have recently been extended to reinforcement learning. In offline RL, they have been used for expressive policy modeling, action generation, trajectory optimization, multi-task decision making, and preference alignment (Janner et al., 2022; Ajay et al., 2023; Ren et al., 2025; Wang et al., 2023; Lu et al., 2023a; He et al., 2024; Jain & Ravanbakhsh, 2024; Mao et al., 2024; He et al., 2023; Dong et al., 2024). These methods typically treat diffusion models as policy or trajectory models, where generation is directly tied to action selection, planning, or value-improving behavior synthesis.

Another line of work uses diffusion models as synthetic data generators for RL (Lu et al., 2023b; Lee et al., 2024; Li et al., 2024; Jackson et al., 2024; Liu et al., 2024). Lu et al. (2023b) trains a diffusion model on the observed replay buffer to generate additional transitions for policy learning. Lee et al. (2024) and Huang et al. (2024) apply partial noising to trajectories and use diffusion models to synthesize high-quality trajectory data. Related methods exploit diffusion models for trajectory stitching, transition modeling, and planning, enabling the generation of trajectories or transitions not explicitly present in the dataset (Ghugare et al., 2024; Li et al., 2024; Yang & Wang, 2025; Yuan et al., 2025; Rigter et al., 2024; Ding et al., 2024). In exploration-driven settings, Wang et al. (2026) and Ying et al. (2026) use diffusion models to estimate visited state distributions, while Wang et al. (2025) adopts conditioned diffusion models to generate high-curiosity replay.

UGR is closely related to diffusion-based replay generation, but differs in its reusable pre-training paradigm. Existing methods usually fit the task-specific replay or train a diffusion model for a particular online distribution, which can suffer from limited early online coverage. In contrast, UGR learns an unsupervised generative prior from reward-free exploratory data, and adapts it through a lightweight utility model. This separates broad transition coverage from target-task alignment, enabling task-relevant synthetic replay without task-specific diffusion training.

6 Conclusion

In this work, we address the challenge of improving data efficiency in online RL under a fixed interaction budget by leveraging unsupervised exploration data as a reusable source of synthetic experience. We present Unsupervised Generative Replay (UGR), which learns a diffusion prior from reward-free exploratory data relabeled with functional rewards. During online learning, a lightweight utility model adapts the prior to generate task-relevant synthetic replay without training a separate diffusion model for each task. Experimental results demonstrate the promise of this approach and suggest that separating prior training from task adaptation can support generative replay across diverse downstream tasks.

7 AI USE STATEMENT

In this work, we used generative AI tools to assist in the writing of proofs, assist with translation, and formulate mathematical claims. Additionally, we used generative AI tools for modifying framework figures and editing code. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B. Tenenbaum, Tommi S. Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=sP1fo2K9DFG.

Xinyue Chen, Che Wang, Zijian Zhou, and Keith Ross. Randomized ensembled double q-learning: Learning fast without a model, 2021. URL https://arxiv.org/abs/2101.05982.

Zihan Ding, Amy Zhang, Yuandong Tian, and Qinqing Zheng. Diffusion world model: Future modeling beyond step-by-step rollout for offline reinforcement learning, 2024. URL https://arxiv.org/abs/2402.03570.

Zibin Dong, Yifu Yuan, Jianye HAO, Fei Ni, Yao Mu, YAN ZHENG, Yujing Hu, Tangjie Lv, Changjie Fan, and Zhipeng Hu. Aligndiff: Aligning diverse human preferences via behavior-customisable diffusion model. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=bxfKIYfHyx.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function, 2018. URL https://arxiv.org/abs/1802.06070.

Kevin Frans, Seohong Park, Pieter Abbeel, and Sergey Levine. Unsupervised zero-shot reinforcement learning via functional reward encodings, 2024. URL https://arxiv.org/abs/2402.17135.

Hongfan Gao, Wangmeng Shen, Xiangfei Qiu, Ronghui Xu, Bin Yang, and Jilin Hu. Ssd-ts: Exploring the potential of linear state space models for diffusion models in time series imputation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '25, pp. 649–660. ACM, August 2025. doi: 10.1145/3711896.3737135. URL http://dx.doi.org/10.1145/3711896.3737135.

Raj Ghugare, Matthieu Geist, Glen Berseth, and Benjamin Eysenbach. Closing the gap between TD learning and supervised learning - a generalisation point of view. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=qg5JENs0N4.

Haoran He, Chenjia Bai, Kang Xu, Zhuoran Yang, Weinan Zhang, Dong Wang, Bin Zhao, and Xuelong Li. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=fAdMly4ki5.

Longxiang He, Li Shen, Linrui Zhang, Junbo Tan, and Xueqian Wang. Diffcps: Diffusion model based constrained policy search for offline reinforcement learning, 2024. URL https://arxiv.org/abs/2310.05333.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. URL https://arxiv.org/abs/2006.11239.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models, 2022. URL https://arxiv.org/abs/2204.03458.

Renming Huang, Yunqiang Pei, Guoqing Wang, Yangming Zhang, Yang Yang, Peng Wang, and Hengtao Shen. Diffusion models as optimizers for efficient planning in offline rl, 2024. URL https://arxiv.org/abs/2407.16142.

Xiao Huang, Xu Liu, Enze Zhang, Tong Yu, and Shuai Li. Offline-to-online reinforcement learning with classifier-free diffusion generation. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=4JbQK1qGpA.

Matthew Thomas Jackson, Michael Tryfan Matthews, Cong Lu, Benjamin Ellis, Shimon Whiteson, and Jakob Foerster. Policy-guided diffusion, 2024. URL https://arxiv.org/abs/2404.06356.

Vineet Jain and Siamak Ravanbakhsh. Learning to reach goals via diffusion. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=3JhmHCVPa8.

Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis, 2022. URL https://arxiv.org/abs/2205.09991.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models, 2022. URL https://arxiv.org/abs/2206.00364.

Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. Urlb: Unsupervised reinforcement learning benchmark, 2021. URL https://arxiv.org/abs/2110.15191.

Jaewoo Lee, Sujin Yun, Taeyoung Yun, and Jinkyoo Park. GTA: Generative trajectory augmentation with guidance for offline reinforcement learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=kZpNDbZrzy.

Guanghe Li, Yixiang Shan, Zhengbang Zhu, Ting Long, and Weinan Zhang. DiffStitch: Boosting offline reinforcement learning with diffusion-based trajectory stitching. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 28597–28609. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/li24bf.html.

Xu-Hui Liu, Tian-Shuo Liu, Shengyi Jiang, Ruifeng Chen, Zhilong Zhang, Xinwei Chen, and Yang Yu. Energy-guided diffusion sampling for offline-to-online reinforcement learning, 2024. URL https://arxiv.org/abs/2407.12448.

Cheng Lu, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning, 2023a. URL https://arxiv.org/abs/2304.12824.

Cong Lu, Philip J. Ball, Yee Whye Teh, and Jack Parker-Holder. Synthetic experience replay. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b. URL https://openreview.net/forum?id=6jNQ1AY1Uf.

Liyuan Mao, Haoran Xu, Xianyuan Zhan, Weinan Zhang, and Amy Zhang. Diffusion-dice: In-sample diffusion guidance for offline reinforcement learning, 2024. URL https://arxiv.org/abs/2407.20109.

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models, 2022. URL https://arxiv.org/abs/2112.10741.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models, 2021. URL https://openreview.net/forum?id=-NEXDKk8gZ.

Yidong Ouyang, Liyan Xie, Hongyuan Zha, and Guang Cheng. Transfer learning for diffusion models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=6emETARnWi.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022. URL https://arxiv.org/abs/2204.06125.

Allen Z. Ren, Justin Lidard, Lars Lien Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=mEpqHvbD2h.

Marc Rigter, Jun Yamada, and Ingmar Posner. World models via policy-guided trajectory diffusion. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=9CcgO0LhKG.

Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation, 2023. URL https://arxiv.org/abs/2208.12242.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022. URL https://arxiv.org/abs/2205.11487.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021. URL https://arxiv.org/abs/2011.13456.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models, 2023. URL https://arxiv.org/abs/2303.01469.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation, 2021. URL https://arxiv.org/abs/2107.03502.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. Deepmind control suite, 2018. URL https://arxiv.org/abs/1801.00690.

Likun Wang, Xiangteng Zhang, Yinuo Wang, Guojian Zhan, Wenxuan Wang, Haoyu Gao, Jingliang Duan, and Shengbo Eben Li. Off-policy reinforcement learning with model-based exploration augmentation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=JGkZgEEjiM.

Renhao Wang, Kevin Frans, Pieter Abbeel, Sergey Levine, and Alexei A Efros. Prioritized generative replay. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=5IkDAfabuo.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=AHvFDPi-FA.

D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997. doi: 10.1109/4235.585893.

Qianlan Yang and Yu-Xiong Wang. RTDiff: Reverse trajectory synthesis via diffusion for offline reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=0FK6tzqV76.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Reinforcement learning with prototypical representations, 2021. URL https://arxiv.org/abs/2102.11271.

Denis Yarats, David Brandfonbrener, Hao Liu, Michael Laskin, Pieter Abbeel, Alessandro Lazaric, and Lerrel Pinto. Don't change the algorithm, change the data: Exploratory data for offline reinforcement learning, 2022. URL https://arxiv.org/abs/2201.13425.

Chengyang Ying, Huayu Chen, Xinning Zhou, Zhongkai Hao, Hang Su, and Jun Zhu. Exploratory diffusion model for unsupervised reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=k0Kb1ynFbt.

Lei Yuan, Yuqi Bian, Lihe Li, Ziqian Zhang, Cong Guan, and Yang Yu. Efficient multi-agent offline coordination via diffusion-based trajectory stitching. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=EpnZEzYDUT.

A THEORETICAL DETAILS

In this section, we provide the proof of Thm. 1 and derive the identity underlying the noisy-time cycle regularization.

A.1 PROOF OF THM. 1

We provide a direct proof of the target-score decomposition following Ouyang et al. (2024). Let $p_S(\mathbf{x}^0)$ and $q_\omega(\mathbf{x}^0)$ denote the source and target clean data distributions, respectively. Assume $q_\omega \ll p_S$, and let both domains share the same forward diffusion kernel $K_k(\mathbf{x}^k|\mathbf{x}^0) = p_{S,k}(\mathbf{x}^k|\mathbf{x}^0) = q_{\omega,k}(\mathbf{x}^k|\mathbf{x}^0)$. The noisy marginals are formulated as,

$$p_{S,k}(\mathbf{x}^k) = \int K_k(\mathbf{x}^k|\mathbf{x}^0)p_S(\mathbf{x}^0)d\mathbf{x}^0, \quad q_{\omega,k}(\mathbf{x}^k) = \int K_k(\mathbf{x}^k|\mathbf{x}^0)q_\omega(\mathbf{x}^0)d\mathbf{x}^0. \tag{27}$$

Using the density ratio $q_\omega(\mathbf{x}^0)/p_S(\mathbf{x}^0)$, we can rewrite the target noisy marginal as,

$$q_{\omega,k}(\mathbf{x}^k) = \int K_k(\mathbf{x}^k|\mathbf{x}^0)p_S(\mathbf{x}^0)\frac{q_\omega(\mathbf{x}^0)}{p_S(\mathbf{x}^0)}d\mathbf{x}^0 = p_{S,k}(\mathbf{x}^k)\mathbb{E}_{p(\mathbf{x}^0|\mathbf{x}^k)}\left[\frac{q_\omega(\mathbf{x}^0)}{p_S(\mathbf{x}^0)}\right], \tag{28}$$

where $p(\mathbf{x}^0|\mathbf{x}^k)$ is the source posterior induced by the forward diffusion process. Taking $\nabla_{\mathbf{x}^k}\log$ on both sides gives

$$\nabla_{\mathbf{x}^k}\log q_{\omega,k}(\mathbf{x}^k) = \nabla_{\mathbf{x}^k}\log p_{S,k}(\mathbf{x}^k) + \nabla_{\mathbf{x}^k}\log\mathbb{E}_{p(\mathbf{x}^0|\mathbf{x}^k)}\left[\frac{q_\omega(\mathbf{x}^0)}{p_S(\mathbf{x}^0)}\right]. \tag{29}$$

Since the optimal target-domain score estimator satisfies

$$G_{\theta^\star}(\mathbf{x}^k, k) = \nabla_{\mathbf{x}^k}\log q_{\omega,k}(\mathbf{x}^k), \tag{30}$$

we obtain:

$$G_{\theta^\star}(\mathbf{x}^k, k) = \nabla_{\mathbf{x}^k}\log p_{S,k}(\mathbf{x}^k) + \nabla_{\mathbf{x}^k}\log\mathbb{E}_{p(\mathbf{x}^0|\mathbf{x}^k)}\left[\frac{q_\omega(\mathbf{x}^0)}{p_S(\mathbf{x}^0)}\right], \tag{31}$$

which completes the proof of Thm. 1.

A.2 DERIVATION OF THE NOISY-DEPENDENT CYCLE REGULARIZATION

We further derive the identity used by the noisy-dependent cycle regularization. Let $p_S(\mathbf{x}^0)$ be the source distribution and $q_\omega(\mathbf{x}^0)$ be the target-task distribution. Under the same forward diffusion kernel $K_k(\mathbf{x}^k|\mathbf{x}^0)$, we have

$$p_{S,k}(\mathbf{x}^k) = \int K_k(\mathbf{x}^k|\mathbf{x}^0)p_S(\mathbf{x}^0)d\mathbf{x}^0, \quad q_{\omega,k}(\mathbf{x}^k) = \int K_k(\mathbf{x}^k|\mathbf{x}^0)q_\omega(\mathbf{x}^0)d\mathbf{x}^0. \tag{32}$$

Then,

$$\begin{aligned}\mathbb{E}_{p_S(\mathbf{x}^0|\mathbf{x}^k)}\left[\frac{q_\omega(\mathbf{x}^0)}{p_S(\mathbf{x}^0)}\right] &= \int \frac{q_\omega(\mathbf{x}^0)}{p_S(\mathbf{x}^0)}p_S(\mathbf{x}^0|\mathbf{x}^k)d\mathbf{x}^0 \\ &= \frac{1}{p_{S,k}(\mathbf{x}^k)}\int q_\omega(\mathbf{x}^0)K_k(\mathbf{x}^k|\mathbf{x}^0)d\mathbf{x}^0 \\ &= \frac{q_{\omega,k}(\mathbf{x}^k)}{p_{S,k}(\mathbf{x}^k)}.\end{aligned} \tag{33}$$

Thus, the conditional expectation of the clean density ratio under the source posterior equals the noisy marginal density ratio at diffusion step k. This derivation reveals a key insight: the forward diffusion process does not discard the distribution shift between source and target; rather, the importance weight propagates along the Markov chain in the sense of conditional expectation. Consequently, we may flexibly choose any intermediate time step k to estimate the density ratio, balancing estimation accuracy against the smoothness of the diffusion state without altering the underlying transfer objective.

B PSEUDOCODE

Algorithm 1 Unsupervised Generative Replay (UGR)

Require: diffusion model (DM) ϵ_θ, classifier c_q, noisy classifier c_ω, utility model f_ψ, agent π_ϕ and Q_ϕ, reward priors $\mathcal{R}^{\text{func}}$, exploratory dataset $\mathcal{D}^{\text{expl}}$, online buffer $\mathcal{D}^{\text{on}}$, synthetic buffer $\mathcal{D}^{\text{syn}}$, temperature β, generation frequency N, DM training steps U, RL steps T.

```
// Diffusion Model Pre-training
for update step = 1, 2, 3, ..., U do
    Sample transitions (s, a, s') from D^expl.
    Sample a reward function R' ~ R^func.
    Relabel the unsupervised transitions x = (s, a, r', s') with r' = R'(s, a, s').
    Update ε_θ via optimizing with Eq. 9 with x.
end for
// Online Training
for online training step = 1, 2, 3, ..., T do
    Collect transitions (s, a, r, s') with π_φ in the environment and add to D^on.
    if steps meets data generating frequency N then
        Update c_q and c_ω via BCE loss with relabeled and online data from D^expl and D^on.
        Update f_ψ via optimizing Eq. 25 with relabeled data from D^expl ∪ D^on.
        Generate samples by Eq. 26 and add them to D^syn.
    end if
    Update agent on samples from D^on ∪ D^syn.
end for
```

C EXPERIMENTAL DETAILS

This section provides additional experimental details, including the evaluated domains and tasks, baseline implementations, functional reward prior design, hyperparameter settings, supplementary ablations, and computing resources.

C.1 DOMAIN AND TASKS

We conduct our experiments on three locomotion domains from the ExORL benchmark (Yarats et al., 2022), which is built on top of the DeepMind Control Suite (Tassa et al., 2018). The ExORL benchmark provides offline datasets collected by unsupervised reinforcement learning agents across multiple environments, each equipped with a set of downstream test-time tasks.

Walker. The Walker domain features a two-legged robot tasked with locomotion starting from a bent-kneed position. The state space is 24-dimensional, and the action space is 6-dimensional, consisting of joint torques, velocities, and positions. Following the ExORL protocol, we evaluate on four tasks: Stand, Walk, Run, and Flip. The reward function for Stand encourages straightened legs and an upright torso; Walk and Run are supersets of Stand that additionally incentivize small and large degrees of forward velocity, respectively; and Flip encourages angular velocity of the torso after standing. All rewards are dense.

Cheetah. The Cheetah domain consists of a bipedal running robot. Following the ExORL evaluation suite, we consider four test-time tasks: Run, Run Backward, Flip, and Flip Backward. These tasks require the agent to achieve forward or backward locomotion at varying speeds.

Quadruped. The Quadruped domain involves a four-legged robot navigating within a 3D environment. The state space is 78-dimensional, and the action space is 12-dimensional, comprising joint torques, velocities, and positions. We evaluate on four tasks: Stand, Walk, Run, and Jump. The Stand task incentivizes maintaining a minimum torso height and straightened legs; Walk and Run extend Stand with rewards for forward velocity at different scales; Jump incorporates a term

encouraging vertical displacement; and Flip encourages angular velocity of the torso after standing. Rewards are dense across all tasks.

C.2 Baselines and Implementations

Our RL backbone builds upon the REDQ (Randomized Ensembled Double Q-learning) implementation used in both PGR (Prioritized Generative Replay) and SynthER (Synthetic Experience Replay). We adopt the same diffusion model architecture and hyperparameters as PGR to ensure fair comparison. All methods share the same underlying REDQ algorithm, with differences lying in how they augment or prioritize the replay buffer.

REDQ. REDQ is a model-free off-policy algorithm that achieves high sample efficiency through a high update-to-data ratio (Chen et al., 2021). To enable this aggressive update scheme while maintaining stability, it employs an ensemble of Q-networks with random subset selection for target computation. In our experiments, we use the same REDQ implementation and hyperparameter settings as PGR, including network architecture, learning rates, ensemble size, target update intervals, and all other training configurations. We train REDQ exclusively on online environment interactions without any generative augmentation, serving as our primary model-free baseline.

Relabel-REDQ. Relabel-REDQ serves as a baseline for reward labeling on reward-free unsupervised datasets. It trains a reward model from online environment interactions to fit the reward function, then uses this learned reward model to annotate transitions from the reward-free exploratory dataset. These relabeled transitions are subsequently added to the training batch alongside online data when updating the policy. This baseline isolates the effect of our generative approach from simple reward relabeling.

SynthER. SynthER is a diffusion-based approach that flexibly upsamples an agent's collected experience by generating synthetic transitions (Lu et al., 2023b). It trains an unconditional diffusion model on the replay buffer and generates additional synthetic transitions to augment the training data, effectively increasing the update-to-data ratio. SynthER can be readily integrated into any algorithm that uses experience replay without requiring algorithmic changes. We implement SynthER using the same diffusion architecture and REDQ backbone as our method.

PGR. PGR proposes a prioritized, parametric memory system that uses generative models to capture online experience (Wang et al., 2025). Unlike SynthER's unconditional generation, PGR employs a conditional diffusion model and guides generation via "relevance functions" (e.g., curiosity or TD-error) that push generated transitions toward more useful regions of the agent's experience. This paradigm enables both (1) densification of past experience through the generative model's generalization capacity and (2) guidance toward more relevant transitions. We use the same diffusion architecture and hyperparameter settings as PGR where applicable, ensuring that performance differences are not due to architectural changes.

C.3 Functional Random Rewards

We employ the same family of functional random rewards following Frans et al. (2024) to annotate our exploratory dataset. Specifically, we generate a diverse set of reward functions including random MLP mappings and linear functions following their exact parameterization and sampling procedure.

Random linear rewards are generated by sampling a weight vector uniformly from $[-1, 1]$. Each dimension is independently masked out with probability 0.9, encouraging sparse and simpler reward functions.

Random MLP functions are generated using a neural network of size $(\mathrm{inputdim}, 32, 1)$. Parameters are sampled using a normal distribution scaled by the average dimension of the layer. A `tanh` activation is used between the two layers. The final output is clipped between -1 and 1.

Table 2: Hyperparameters used for UGR in our experiments.

Hyperparameter	**Setting**
Network Type (Denoising)	Residual MLP
Denoising Network Depth	6
Denoising Steps	128
Denoising Network Learning Rate	3×10^{-4}
Denoising Network Hidden Dimension	1024 units
Denoising Network Batch Size	256
Denoising Network Activation Function	ReLU
Denoising Network Optimizer	Adam
Learning Rate Schedule (Denoising Network)	Cosine Annealing
Training Epochs (Denoising Network)	$50,000$
Update Frequency (Denoising Network)	Every $10,000$ steps
Replay Buffer Warm Up Step	$5,000$
Online Steps	1×10^5
Clean Classifier c_q	4-Layers MLP
Noisy Classifier c_ω	5-Layers MLP
Utility Model f_ψ	5-Layers MLP
Hidden Size for c_q, c_ω, f_ψ	256
Learning Rate for c_q, c_ω, f_ψ	3×10^{-4}
Optimizer for c_q, c_ω, f_ψ	Adam
Energy Temperature β	3.0
Guidance Scale η	1.0
Diffusion Ratio λ	0.5
UTD Ratio	20
Energy Label Group M	16

C.4 Hyperparameters

For the energy temperature hyperparameter β, we adopt task-specific configurations: we set $\beta = 5$ on the Quadruped Run and Cheetah Run Backward tasks, $\beta = 10$ on Quadruped Jump and Cheetah Flip, and $\beta = 3$ for all remaining tasks.

C.5 Additional Ablation Studies

Energy learning objective. We study the learning objective of the utility model in Fig. 4. Here, **MSE_UGR** directly regresses the density-ratio target with an MSE loss, **UGR** uses the proposed objective in Eq. 25, and **Without-energy** removes utility guidance. Across both tasks, removing the energy guidance results in clearly worse performance, confirming that the pretrained generative prior alone is insufficient to produce task-relevant replay. Direct MSE regression improves over the no-energy variant, but remains less effective than UGR. This suggests that the density ratio is useful, but directly fitting its absolute value can be sensitive to scale, noise, and sharp ratio variations. In contrast, UGR learns a relative energy ranking within sampled groups, providing a more stable and calibrated guidance signal for diffusion sampling. These results highlight that UGR benefits not only from target-aware generation, but also from the contrastive objective for the utility model.

Reward prior design. We study the effect of the functional random reward prior used for exploratory relabeling. In Fig. 5, **Linear** uses only random linear rewards, **MLP** uses only random nonlinear rewards, and **UGR** uses the full mixture of both. On both tasks, the mixed reward prior in UGR achieves the strongest performance, with especially clear gains on the more challenging Quadruped Walk task. Using only one of the rewards leads to weaker improvement, suggesting that either choice alone provides insufficient task diversity for pretraining the generative prior. The results indicate that linear and nonlinear random rewards play complementary roles: linear rewards provide simple global task structure, while MLP rewards introduce richer nonlinear objectives. Their

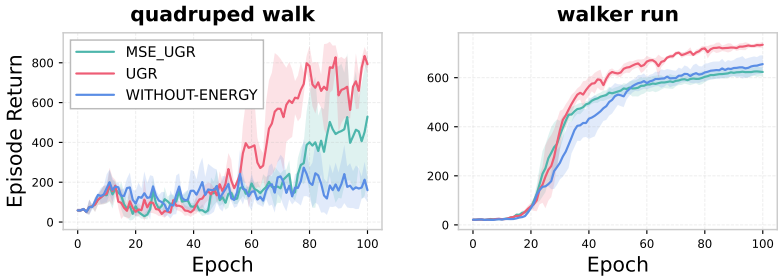


Figure 4: Ablation results on the Quadruped Walk and Walker Run tasks to study the energy learning objective. The solid lines depict the mean performance, while the shaded areas represent the confidence intervals over three seeds.

combination yields a more expressive reward prior distribution, which in turn produces a stronger generative prior for downstream utility-guided replay.

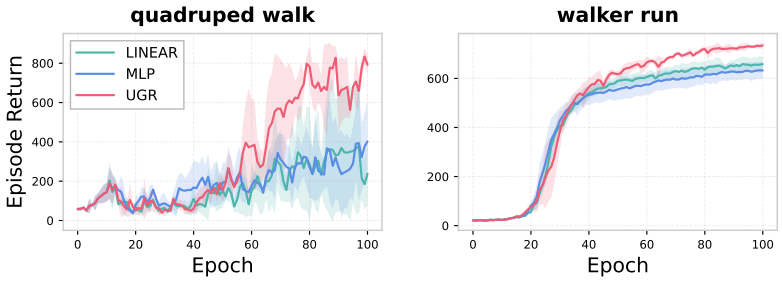


Figure 5: Ablation results on the Quadruped Walk and Walker Run tasks for reward prior design.

Cycle regularization. We then investigate the effect of noisy-dependent cycle regularization for training the utility model. In Fig. 6, **Clean** uses only clean-sample density-ratio supervision, **Noisy** uses only noisy-time density-ratio supervision, and **UGR** combines both. On both tasks, the full UGR consistently outperforms the two partial variants by a large margin, especially on the harder Quadruped Walk task. These results suggest that the two forms of supervision play complementary roles: clean density-ratio signals provide task-level semantic alignment, while noisy-time supervision improves consistency with the diffusion sampling process. Their combination yields a more reliable utility model and, in turn, more effective task-relevant synthetic replay.

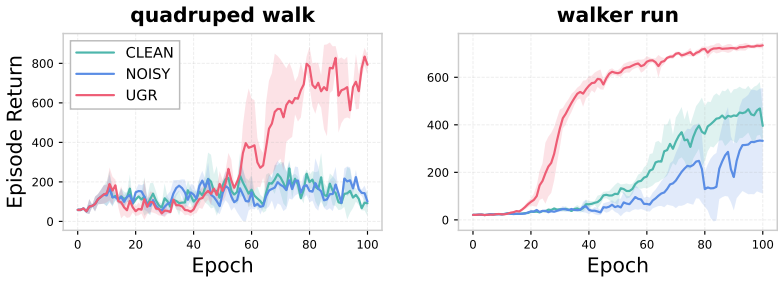


Figure 6: Ablation results on the Quadruped Walk and Walker Run tasks for noisy-time cycle regularization.

UTD ratio. We next examine how effectively different methods exploit synthetic replay under varying update-to-data (UTD) ratios. As shown in Fig. 7, increasing UTD generally improves policy learning, but the gains are substantially more pronounced for UGR than for PGR. With a small UTD,

both methods make limited progress, suggesting that insufficient policy updates cannot fully exploit the information provided by synthetic replay. As UTD increases, UGR achieves faster learning and higher final returns on both tasks. These results indicate that UGR produces replay samples that are not only more task-relevant, but also more useful for repeated policy optimization. In other words, UGR can better support high-UTD learning, enabling the agent to extract more value from each unit of online experience.

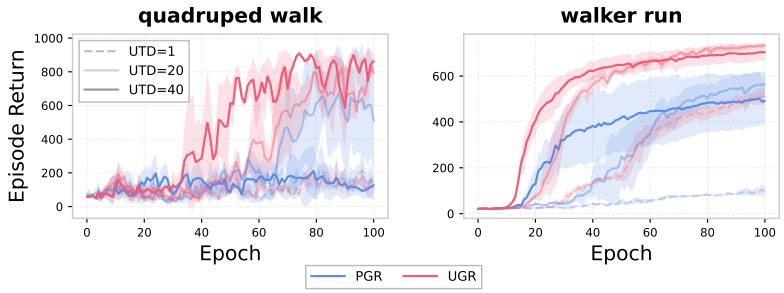


Figure 7: Ablation results on the Quadruped Walk and Walker Run tasks under different UTD ratios.

C.6 Computing Resources

All experiments are conducted on a server with Ubuntu 18.04 LTS, a single NVIDIA GeForce RTX 3090 GPU with 24GB VRAM, and a 32-core CPU. The GPU is used for diffusion model pre-training, guided synthetic replay generation, and policy network updates, while CPU workers handle DeepMind Control environment rollouts and data loading.