DUO: Diffusion Models for Universal Offline Black-Box Optimization

Anonymous Author(s)
Affiliation
Address
email

Abstract

Offline black-box optimization aims to find high-performing designs from a fixed dataset without online evaluations, with applications spanning protein design, materials discovery, and robot learning. Most existing methods are typically designed for a single task with fixed dimensionality, leaving universal offline optimization—learning one shared model across heterogeneous search spaces, mixed variable types, and scarce-data transfer settings—largely unresolved. In this paper, we study universal offline optimization from a generative perspective for the first time and propose DUO (**D**iffusion model for **U**niversal offline black-box **O**ptimization), which bridges universal representation learning with trajectory-level diffusion modeling. DUO uses a Transformer-based variational autoencoder to embed both continuous and discrete designs into a shared latent space, avoiding task-specific architectures for incompatible native domains. Within this unified space, we synthesize optimization-oriented trajectories and train a conditional diffusion model, with task-level semantics injected through frozen text-metadata embeddings and classifier-free guidance. A cross-entropy consistency term further aligns continuous training with discrete evaluation objectives. Evaluated on the Design-Bench and SOO-Bench benchmarks, DUO demonstrates strong performance across diverse continuous and discrete tasks under multitask joint training. Our experiments highlight its robust zero-shot and few-shot transfer capabilities, suggesting that metadata-aware latent trajectory diffusion provides a highly effective framework for universal offline black-box optimization.

1 Introduction

Black-box optimization (BBO) seeks high-performing designs when the objective is unknown and can only be observed through input–output evaluations. Such problems arise in high-cost domains including biological engineering, materials screening, robot learning, and chip parameter tuning [33, 13]. Classical tools such as Bayesian optimization [8] remain effective when the optimizer can query the objective repeatedly, but this assumption breaks down in wet-lab pipelines, large-scale simulations, and other settings where each measurement is expensive. In these regimes one must instead optimize from a *fixed* historical dataset with no online queries during training, the setting known as offline BBO [33].

Most offline BBO methods, however, follow a single-task regime: each model is tied to one search space, one dimensionality, and one task-specific dataset. This is restrictive because real tasks often provide only a small number of labeled examples, or even metadata alone, while related tasks may share useful structure [21]. Universal offline BBO addresses this gap by training one shared model across heterogeneous objectives, dimensions, and variable types, typically by serializing designs and conditioning on natural-language metadata so that one predictor can cover many tasks [30, 29]. The

Submitted to ICML 2026 Workshop on Decision-Making from Offline Datasets to Online Adaptation: Black-Box Optimization to Reinforcement Learning. Do not distribute.

result is cross-task knowledge transfer in zero-shot and few-shot regimes that single-task models cannot reach.

Yet the optimization interface in these universal methods remains forward-oriented: the model predicts rewards, and a separate inner-loop search or ranking step is responsible for actually improving designs. UniSO [30] is representative of this pattern—string representations and metadata-guided representation learning unify different search spaces, but optimization still consists of fitting a reward predictor on offline tuples and then searching in model space. Such an interface inherits the well-known fragility of offline reward models on out-of-distribution designs, and the search step itself is hard to share or amortize across tasks whose coordinates and step sizes have no common meaning. A generative *backward* approach offers a natural alternative: directly learn how designs move toward higher scores, rather than scoring them and searching afterwards. GTG [43] demonstrates the strength of this idea in the single-task setting by constructing pseudo optimization paths and modeling them with diffusion.

Extending trajectory generation to universal offline BBO, however, cannot be achieved by simply pooling trajectories from different tasks. GTG-style models are trained and sampled in one native design space of fixed shape, whereas universal BBO must handle spaces that differ in variable type, dimensionality, and objective semantics. Because these spaces share no common coordinate system, neighborhoods, noise perturbations, and improvement directions are not comparable across tasks. A universal backward optimizer must therefore (i) expose a common interface for heterogeneous designs, (ii) build improvement trajectories on which "moving forward" has consistent meaning, and (iii) condition generation on the task semantics that define what "improvement" is.

We instantiate these requirements in **DUO** (**D**iffusion model for **U**niversal offline black-box **O**ptimization), which addresses these limitations with three matching components: a Transformer-based VAE that maps mixed continuous and discrete designs into a shared latent space, synthetic improvement trajectories constructed in that latent space, and a metadata-conditioned 1D U-Net diffusion model that generates better designs backward along these paths. DUO thus replaces the forward-regression-plus-search interface used by universal forward methods such as UniSO [30] with a single generative backbone, and lifts the single-task trajectory paradigm of GTG [43] to heterogeneous multitask settings.

Our contributions are threefold. **(1) A backward formulation of universal offline BBO:** we reframe universal offline BBO as a backward generative problem over improvement trajectories, removing the reward-predictor-plus-inner-search interface that current universal methods inherit from single-task offline BBO. **(2) Metadata-aware latent trajectory diffusion:** we realize this formulation with a Transformer-based VAE shared latent interface, a conditional 1D U-Net over synthetic improvement trajectories, frozen text-metadata guidance [10], and a grouped cross-entropy term for discrete tasks. **(3) Empirical gains over forward baselines:** on Design-Bench and SOO-Bench, DUO improves over the representative universal forward optimizer UniSO and the single-task trajectory baseline GTG under multitask training, while retaining zero-shot and few-shot transfer ability on held-out tasks.

2 Related Work

Offline black-box optimization. Offline BBO seeks high-performing designs from a fixed dataset without online queries [33]. Existing methods are commonly divided into *forward* approaches, which fit a surrogate objective and then search against it, and *backward* approaches, which directly learn to generate promising designs [35]. Representative forward methods include COMs [32], RoMA [40], ICT [41], and RaM-ListNet [31]; representative backward methods include MINs [17], DDOM [15], and RGD [5]. These methods have shown strong single-task results but are tied to one fixed search space and cannot directly reuse experience from other tasks.

Universal and pretrained optimizers. Universal offline BBO asks one model to operate across tasks with different dimensions, variable types, objectives, and metadata [30]. UniSO [30] is the closest universal offline baseline: it serializes heterogeneous designs, attaches natural-language metadata, and trains token-targeted or numeric-targeted regressors with semantic alignment. Related pretrained optimizers and predictors, including ExPT [21], POM [18], OptFormer [6], and OmniPred [29], further show that synthetic diversity, serialized histories, and language-model embeddings can improve

transfer. However, most universal methods are forward types: they learn a reward predictor or ranking interface and still require a separate search procedure. In particular, UniSO covers heterogeneous multitask learning with metadata but offers no trajectory- or generation-based optimization interface; DUO instead keeps the universal, metadata-aware training setting and replaces forward regression with a backward generative model of improvement trajectories.

Trajectory-based generative optimization. Trajectory-level methods model how designs move toward better regions rather than only modeling isolated high-score samples. BONET [16] sorts offline examples by score, stitches synthetic optimization paths, and trains an autoregressive Transformer. GTG [43] improves this idea with diffusion and neighborhood-based trajectory construction, reducing autoregressive error accumulation and providing a strong single-task trajectory baseline. Trajectory diffusion is also well established in offline RL [12, 2, 19]. The key limitation for universal BBO is geometric: these models are normally trained in one native design space of fixed shape. BONET and GTG motivate trajectory-centric generation but remain single-task and lack any mechanism for joint multitask training, heterogeneous design spaces, or task-metadata conditioning. DUO extends the trajectory diffusion paradigm by first mapping heterogeneous designs into a shared latent interface and then conditioning the denoising process on task metadata, thereby combining trajectory-based backward generation with the multitask, metadata-aware setting that current universal forward optimizers occupy.

Latent and metadata-conditioned generative interfaces. DUO combines two auxiliary ideas that make universal trajectory diffusion practical. First, latent generative modeling uses a compressed representation as the common optimization space, as in latent diffusion [27], latent Bayesian optimization [34], latent energy-based optimization [39], and transfer optimization across spaces [7]. Second, language-model features provide task-level semantics for optimization, either as Bayesian-optimization features [23, 22], multitask prediction inputs [29], or fully agentic optimization interfaces [20]. DUO uses these ideas in a narrower way: a Transformer–VAE supplies the shared latent space, while frozen metadata embeddings steer diffusion through classifier-free guidance [10]; the optimization backbone remains a trajectory denoiser rather than an LLM or a forward surrogate.

3 DUO

In this section, we introduce DUO, a diffusion-based framework for universal offline black-box optimization. The main difficulty is not merely how to condition a model on different tasks. A more basic question is: *what should a single diffusion model denoise when different tasks have different dimensions, variable types, and native geometries?* Directly defining one diffusion process over raw designs is ill-posed: a robot morphology vector, a material descriptor, a DNA sequence, and a trajectory-planning vector do not share a common coordinate system.

DUO addresses this issue in three steps, as shown in Figure 1. First, it maps heterogeneous designs into a shared latent interface with a Transformer–VAE. Second, it constructs synthetic improvement trajectories in this latent space, so that the model learns how designs move from lower-score to higher-score regions rather than only modeling isolated good samples. Third, it trains a metadata-conditioned diffusion model over these latent trajectories and decodes the generated latents back to task-native designs.

3.1 Universal offline optimization setting

We consider a set of training tasks $\mathcal{T}_{\mathrm{tr}}$. Each task $\tau \in \mathcal{T}_{\mathrm{tr}}$ has a search space $\mathcal{X}_\tau$, an unknown objective function $f_\tau : \mathcal{X}_\tau \to \mathbb{R}$, and a fixed offline dataset

$$\mathcal{D}_\tau = \{(\boldsymbol{x}_i^\tau, y_i^\tau)\}_{i=1}^{N_\tau}, \qquad y_i^\tau = f_\tau(\boldsymbol{x}_i^\tau). \tag{1}$$

No online query to f_τ is allowed during training. The search spaces $\mathcal{X}_\tau$ may be continuous, discrete, or mixed, and their dimensionalities can be different across tasks. Each task also comes with metadata m_τ, such as the task name, variable type, description, and optimization objective.

The goal is to train one shared generator that can output high-scoring candidates for both seen and unseen tasks. At deployment time, a new task may provide only metadata $m_{\tau_{\mathrm{new}}}$, corresponding to the zero-shot case, or metadata plus a small dataset $\mathcal{D}_{\tau_{\mathrm{new}}}$, corresponding to the few-shot case.

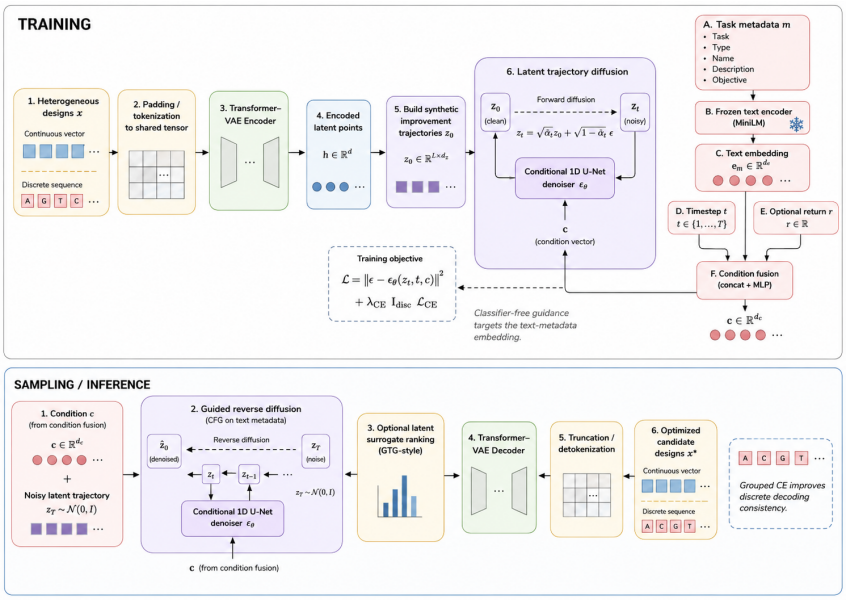

Figure 1: Overview of DUO. **Training:** Heterogeneous continuous/discrete designs are tokenized into masked tensors, encoded by a Transformer–VAE, and stitched into score-improving latent trajectories. A metadata-conditioned 1D U-Net denoises these trajectories, with optional return conditioning and grouped CE for discrete variables. **Sampling:** Guided reverse diffusion generates latent trajectories, which are optionally reranked by an offline surrogate and decoded back to task-native designs.

In the few-shot case, DUO can optionally finetune the shared parameters, while keeping the same architecture and latent interface.

3.2 A shared latent interface via VAE

A natural but problematic solution is to pad all raw designs to the same length and train diffusion directly in that padded space. This makes the tensors compatible, but it does not make the geometry meaningful. For example, Euclidean distance between two padded DNA one-hot tensors is not comparable to Euclidean distance between two robot morphology vectors. A universal denoiser trained in such a raw space would therefore mix incompatible notions of locality and smoothness.

DUO instead learns a shared latent interface. For each task, we first convert a native design $\boldsymbol{x}_i^\tau$ into a unified tensor

$$\boldsymbol{u}_i^\tau = \Pi_\tau(\boldsymbol{x}_i^\tau) \in \mathbb{R}^{L \times C}, \tag{2}$$

where Π_τ denotes task-specific preprocessing. Continuous vectors are normalized and padded or truncated to length L with masks. Discrete designs are represented by grouped one-hot tensors, where each group corresponds to a categorical decision such as a sequence position. The mask records which entries are valid and which entries are introduced only for padding.

A Transformer–VAE then maps $\boldsymbol{u}_i^\tau$ into a latent vector:

$$q_\phi(\boldsymbol{z} \mid \boldsymbol{u}) = \mathcal{N}\big(\boldsymbol{\mu}_\phi(\boldsymbol{u}), \mathrm{diag}(\boldsymbol{\sigma}_\phi^2(\boldsymbol{u}))\big), \qquad \boldsymbol{z} = \boldsymbol{\mu}_\phi(\boldsymbol{u}) + \boldsymbol{\sigma}_\phi(\boldsymbol{u}) \odot \boldsymbol{\xi}, \quad \boldsymbol{\xi} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{3}$$

The decoder reconstructs the unified tensor from the latent code:

$$\hat{\boldsymbol{u}} = \mathrm{Dec}_\psi(\boldsymbol{z}). \tag{4}$$

The VAE is trained over all tasks with a masked reconstruction objective and a KL regularizer:

$$\mathcal{L}_{\mathrm{VAE}} = \mathbb{E}_{\tau,i}\left[\ell_{\mathrm{rec}}\big(\mathrm{Dec}_\psi(\boldsymbol{z}_i^\tau), \boldsymbol{u}_i^\tau\big) + \beta D_{\mathrm{KL}}\big(q_\phi(\boldsymbol{z} \mid \boldsymbol{u}_i^\tau) \,\|\, \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})\big)\right). \tag{5}$$

Here ℓ_{rec} is evaluated only on valid, non-padded entries. After training, each offline design is represented by a latent point $\boldsymbol{z}_i^\tau = \mathrm{Enc}_\phi(\boldsymbol{u}_i^\tau)$. This gives DUO a common object to model: every task now provides latent points of the same dimension.

3.3 Constructing latent improvement trajectories

DUO does not train diffusion on independent latent points. Instead, it trains on short trajectories that imitate the behavior of an optimizer moving toward better regions. This trajectory view is useful because offline BBO cares about improvement, not just reconstruction of the offline data distribution.

For each task τ, we first encode all offline designs into latent points $\{(\boldsymbol{z}_i^\tau, y_i^\tau)\}_{i=1}^{N_\tau}$. We then build synthetic trajectories of fixed length H by chaining local score-improving transitions. Let $\mathcal{N}_\kappa(i)$ be the set of κ nearest neighbors of $\boldsymbol{z}_i^\tau$ in the latent space. A candidate successor of i must be both close in latent space and better in objective value:

$$\mathcal{S}_\tau(i) = \left\{ j \in \mathcal{N}_\kappa(i) : \tilde{y}_j^\tau > \tilde{y}_i^\tau + \epsilon_{\text{traj}} \right\}, \tag{6}$$

where $\tilde{y}$ denotes the normalized score and ϵ_{traj} filters out transitions with negligible improvement. Starting from an offline point, we repeatedly sample a successor from $\mathcal{S}_\tau(i)$ and append it to the path. This produces a latent trajectory

$$\boldsymbol{Z}_0^\tau = [\boldsymbol{z}_{i_1}^\tau, \boldsymbol{z}_{i_2}^\tau, \ldots, \boldsymbol{z}_{i_H}^\tau] \in \mathbb{R}^{H \times d_z}, \qquad \tilde{y}_{i_1}^\tau \le \tilde{y}_{i_2}^\tau \le \cdots \le \tilde{y}_{i_H}^\tau. \tag{7}$$

The important point is that every trajectory is constructed within one task, but after VAE encoding all such trajectories have the same shape. Thus, the diffusion model can be trained jointly across tasks without requiring a native-space diffusion process for each task.

3.4 Metadata-conditioned latent diffusion

Given the latent trajectories, DUO trains a conditional diffusion model to generate new improvement paths. For a clean latent trajectory $\boldsymbol{Z}_0$, the forward diffusion process adds Gaussian noise:

$$\boldsymbol{Z}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{Z}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{8}$$

A conditional 1D U-Net $\boldsymbol{\epsilon}_\theta$ predicts the noise from $(\boldsymbol{Z}_t, t)$ and task-level condition $\boldsymbol{c}_\tau$.

The condition $\boldsymbol{c}_\tau$ is built from natural-language metadata. For each task, we write a short text prompt containing fields such as `Task`, `Type`, `Name`, `Description`, and `Objective`. This prompt is encoded by the frozen `sentence-transformers/all-MiniLM-L6-v2` model, producing a 384-dimensional metadata vector:

$$\boldsymbol{e}_\tau^{\text{text}} = E_{\text{text}}(m_\tau). \tag{9}$$

We project this vector to the U-Net width and fuse it with the diffusion timestep embedding and optional return information:

$$\boldsymbol{c}_\tau = g_\eta\left(W_m \boldsymbol{e}_\tau^{\text{text}}, \boldsymbol{e}_t, \boldsymbol{e}_r\right). \tag{10}$$

Here $\boldsymbol{e}_r$ is used only when return conditioning is enabled. The diffusion loss is

$$\mathcal{L}_{\text{diff}}^{(\tau)} = \mathbb{E}_{\boldsymbol{Z}_0^\tau, t, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{Z}_t^\tau, t, \boldsymbol{c}_\tau)\|_2^2 \right]. \tag{11}$$

To make metadata useful at sampling time, DUO uses classifier-free guidance on the text-metadata channel. During training, the metadata embedding is randomly dropped with probability p_{drop}. During sampling, the conditional and text-dropped predictions are combined as

$$\tilde{\boldsymbol{\epsilon}}_\theta = (1+s)\boldsymbol{\epsilon}_\theta(\boldsymbol{Z}_t, t, \boldsymbol{c}_\tau) - s\boldsymbol{\epsilon}_\theta(\boldsymbol{Z}_t, t, \boldsymbol{c}_\varnothing), \qquad s \ge 0. \tag{12}$$

In this default form, $\boldsymbol{c}_\varnothing$ removes the text-metadata vector while keeping the diffusion timestep and other non-text conditioning signals unchanged. This makes the guidance weight s directly control how strongly the generated trajectory follows the task semantics.

3.5 Discrete consistency

Latent diffusion is trained with an MSE noise-prediction objective, but this objective alone is not always aligned with discrete evaluation. For example, in a DNA task, each sequence position must decode to exactly one nucleotide. Two relaxed vectors can be close under MSE while still producing different categorical decisions after argmax decoding. This mismatch is especially harmful when the benchmark oracle evaluates the final discrete sequence rather than the relaxed tensor.

Algorithm 1 DUO training and sampling

Require: Offline datasets $\{\mathcal{D}_\tau\}_{\tau\in\mathcal{T}_{\text{tr}}}$, metadata $\{m_\tau\}_{\tau\in\mathcal{T}_{\text{tr}}}$, trajectory length H, diffusion steps T, CE weight λ_{CE}, guidance scale s.

Ensure: Generated candidates for each target task.

1: Convert each native design $\boldsymbol{x}_i^\tau$ into a unified masked tensor $\boldsymbol{u}_i^\tau = \Pi_\tau(\boldsymbol{x}_i^\tau)$.
2: Train the Transformer–VAE with $\mathcal{L}_{\text{VAE}}$ and encode all offline designs into latent points $\boldsymbol{z}_i^\tau$.
3: For each task, construct synthetic improvement trajectories $\boldsymbol{Z}_0^\tau \in \mathbb{R}^{H\times d_z}$ by chaining local score-improving latent points.
4: Encode metadata m_τ with the frozen text encoder and build conditions $\boldsymbol{c}_\tau$.
5: Train the conditional 1D U-Net denoiser with $\mathcal{L}_{\text{DUO}}$.
6: **for** each target task τ **do**
7: Sample latent trajectories from Gaussian noise using guided reverse diffusion.
8: Decode generated latent points into native designs.
9: Optionally rank the decoded candidates with an offline-trained surrogate.
10: Return the top candidates.
11: **end for**

DUO therefore adds a grouped cross-entropy consistency term for discrete tasks. From the predicted noise, we recover the predicted clean latent trajectory

$$\hat{\boldsymbol{Z}}_0 = \frac{\boldsymbol{Z}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\boldsymbol{Z}_t, t, \boldsymbol{c}_\tau)}{\sqrt{\bar{\alpha}_t}}. \tag{13}$$

We then decode $\hat{\boldsymbol{Z}}_0$ into grouped logits $\hat{\boldsymbol{U}} = \text{Dec}_\psi(\hat{\boldsymbol{Z}}_0)$ and compare them with the original grouped one-hot tensor $\boldsymbol{U}_0$. For a discrete task with groups $\mathcal{G}_\tau$, the auxiliary loss is

$$\mathcal{L}_{\text{CE}}^{(\tau)} = -\frac{1}{H|\mathcal{G}_\tau|}\sum_{h=1}^{H}\sum_{g\in\mathcal{G}_\tau}\sum_{c=1}^{C_g} U_{0,h,g,c}\log\frac{\exp(\hat{U}_{h,g,c})}{\sum_{c'=1}^{C_g}\exp(\hat{U}_{h,g,c'})}. \tag{14}$$

The full training objective is

$$\mathcal{L}_{\text{DUO}} = \mathbb{E}_{\tau\sim\mathcal{T}_{\text{tr}}}\left[\mathcal{L}_{\text{diff}}^{(\tau)} + \lambda_{\text{CE}}\mathbb{I}[\tau\in\mathcal{M}_{\text{disc}}]\mathcal{L}_{\text{CE}}^{(\tau)}\right], \tag{15}$$

where $\mathcal{M}_{\text{disc}}$ denotes the set of discrete tasks. The CE term is inactive for purely continuous tasks.

3.6 Sampling and candidate selection

At inference time, DUO starts from Gaussian noise $\boldsymbol{Z}_T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and runs the guided reverse diffusion process using Eq. (12). This yields a generated latent trajectory

$$\hat{\boldsymbol{Z}}_0 = [\hat{\boldsymbol{z}}_1, \hat{\boldsymbol{z}}_2, \ldots, \hat{\boldsymbol{z}}_H]. \tag{16}$$

Each latent point is decoded by the VAE decoder and mapped back to the native task space:

$$\hat{\boldsymbol{x}}_h = \Pi_\tau^{-1}\left(\text{Dec}_\psi(\hat{\boldsymbol{z}}_h)\right), \qquad h = 1, \ldots, H. \tag{17}$$

For continuous tasks, Π_τ^{-1} removes padding and reverses normalization. For discrete tasks, it applies group-wise argmax or the benchmark-specified discretization rule.

Following the GTG evaluation pipeline, DUO may optionally train a lightweight task-specific surrogate on the offline dataset and use it only as a final filter over generated candidates. This step is not part of the core generative model: it is used to rank a finite candidate pool under matched evaluation budgets. Without this filter, DUO simply returns decoded samples from the metadata-conditioned latent diffusion model.

3.7 Algorithm summary

Algorithm 1 summarizes the full DUO pipeline. Line 1 standardizes heterogeneous raw designs into a common masked tensor format. Line 2 learns the shared Transformer–VAE interface and stores each offline design as a fixed-dimensional latent point. Line 3 turns these points into short score-improving

trajectories, which provide the training data for diffusion. Line 4 converts task metadata into condition vectors through the frozen text encoder and condition-fusion module. Line 5 trains the conditional 1D U-Net with the DUO objective in Eq. (15). Lines 6–10 describe inference: for each target task, DUO samples latent trajectories by guided reverse diffusion, decodes the generated latents into native designs, optionally reranks them with an offline surrogate, and returns the top candidates.

4 Experiments

Our experiments investigate that whether one offline generator can be trained once on heterogeneous tasks and still produce high-scoring designs without any online feedback. We evaluate this question on Design-Bench [33] and SOO-Bench [25], following the universal multitask offline protocol [30]: nine shared training tasks, held-out tasks for transfer, and no oracle queries during learning.

4.1 Main results

Table 1 reports maximum rewards on the nine multitask training tasks (absolute simulator units; see caption). DUO achieves the best mean rank among UniSO, single-task GTG, and itself, with the clearest margins on continuous robotics and several GTOPX missions; UniSO[1] remains competitive on some DNA and trajectory rows, so the comparison is best read task-by-task.

Table 1: Multitask training corpus at **raw** scale (mean±std): UniSO [30], single-task GTG [43], and our method—multitask DUO with full text-metadata embeddings. $\mathcal{D}$(best) is the best logged reward. Mean rank averages per-task ranks among UniSO, GTG, and DUO (lower is better). Ablations and single-task latent DUO are in Table 5.

Task	$\mathcal{D}$(best)	UniSO	GTG	DUO
Ant	165.326	**455.658 ± 39.188**	452.330 ± 61.502	**565.873 ± 13.651**
D'Kitty	199.363	222.007 ± 33.677	**264.857 ± 17.571**	**312.350 ± 6.353**
Superconductor	74.000	82.642 ± 3.467	**89.802 ± 6.763**	**112.880 ± 8.088**
TF Bind 8	0.439	0.857 ± 0.069	**0.934 ± 0.030**	**0.933 ± 0.055**
TF Bind 10	0.005	**0.944 ± 0.794**	0.632 ± 0.106	**0.955 ± 0.516**
GTOPX 2	-196.663	-90.106 ± 8.223	**-72.163 ± 8.495**	**-44.477 ± 4.761**
GTOPX 3	-152.364	**-44.427 ± 13.456**	-53.866 ± 9.877	**-29.310 ± 2.237**
GTOPX 4	-216.709	-74.779 ± 11.032	**-72.749 ± 6.632**	**-51.019 ± 6.536**
GTOPX 6	-111.666	**-48.244 ± 5.766**	-55.834 ± 9.655	**-33.449 ± 12.583**
Mean rank	/	2.556 ± 0.497	**2.333 ± 0.667**	**1.111 ± 0.314**

Table 2 evaluates the same multitask DUO model (full text-metadata conditioning, our method) on the five Design-Bench tasks following [33, 30]; preprocessing is as in the table caption. We list list Design-Bench baselines together with offline BBO methods since 2024 onward and recompute **Avg. Rank** within this subset. The row $\mathcal{D}$(best) is the best objective observed in each task's offline training log without a learned generative policy. Competing methods report mean±std when their releases include multiple runs.

Taken together, the two views support the same conclusion: DUO improves over the universal baseline UniSO overall (better mean rank on the training corpus and the best average rank in Table 2) and over representative single-task trajectory diffusion (GTG) and classic Design-Bench optimizers, while staying competitive with the strongest recent entries (e.g., LTR on DNA). Task-specific noise remains, especially on DNA binding, where Appendix H studies the cross-entropy auxiliary. Section 4.3 and Appendix I add an illustrative reverse-diffusion diagnostic (Superconductor, seed 0) with full mean/top-8/max panels. The full Design-Bench leaderboard is Table 6.

4.2 Zero-shot and few-shot transfer on held-out tasks

We evaluate whether metadata and shared latents transfer to held-out control-style tasks: LunarLander, Rover, and RobotPush [4, 36, 24]. Table 3 reports maximum episode reward under zero-shot and

[1] Unless stated otherwise, we write **UniSO** for the universal offline optimizer of Tan et al. [30]; Tables use the results of *Improved UniSO-T*, which is the best-performing model reported in that framework.

Table 2: Design-Bench (rescaled): Design-Bench baselines [33] plus learned optimizers from 2024 onward. Publication venues for all methods appear in Table 6. **Avg. Rank** reports the mean per-task rank divided by the number of competing methods in this table (16 learned optimizers; $\mathcal{D}$(best) excluded). Evaluation follows Appendix D.

Method	Ant	D'Kitty	Superconductor	TF-Bind-8	TF-Bind-10	Avg. Rank
$\mathcal{D}$(best)	0.565	0.884	0.400	0.439	0.467	/
BO-qEI	0.812 ± 0.000	0.896 ± 0.000	0.382 ± 0.013	0.802 ± 0.081	0.628 ± 0.036	12.0 / 16
CMA-ES	**1.712 ± 0.754**	0.725 ± 0.002	0.463 ± 0.042	0.944 ± 0.017	0.641 ± 0.036	8.2 / 16
REINFORCE	0.248 ± 0.039	0.541 ± 0.196	0.478 ± 0.017	0.935 ± 0.049	**0.673 ± 0.074**	10.0 / 16
Grad. Ascent	0.273 ± 0.023	0.853 ± 0.018	0.510 ± 0.028	0.969 ± 0.021	0.646 ± 0.037	8.2 / 16
Grad. Ascent Mean	0.306 ± 0.053	0.875 ± 0.024	0.508 ± 0.019	**0.985 ± 0.008**	0.633 ± 0.030	7.6 / 16
Grad. Ascent Min	0.282 ± 0.033	0.884 ± 0.018	0.514 ± 0.020	0.979 ± 0.014	0.632 ± 0.027	7.8 / 16
PGS	0.715 ± 0.046	0.954 ± 0.022	0.444 ± 0.020	0.889 ± 0.061	0.634 ± 0.040	9.6 / 16
FGM	0.923 ± 0.023	0.944 ± 0.014	0.481 ± 0.024	0.811 ± 0.079	0.611 ± 0.008	9.8 / 16
MATCH-OPT	0.933 ± 0.016	0.952 ± 0.008	0.504 ± 0.021	0.824 ± 0.067	0.655 ± 0.050	6.8 / 16
GTG	0.855 ± 0.044	0.942 ± 0.017	0.480 ± 0.055	0.910 ± 0.040	0.619 ± 0.029	9.4 / 16
GABO	0.038 ± 0.012	0.719 ± 0.001	0.374 ± 0.020	0.926 ± 0.038	0.619 ± 0.043	14.2 / 16
LTR	0.949 ± 0.025	0.962 ± 0.015	**0.517 ± 0.029**	**0.981 ± 0.012**	0.670 ± 0.035	**3.0 / 16**
ROOT	0.958 ± 0.012	0.971 ± 0.005	0.451 ± 0.032	0.977 ± 0.015	0.653 ± 0.030	5.4 / 16
DynAMO-Adam	0.113 ± 0.085	0.789 ± 0.059	0.413 ± 0.106	0.719 ± 0.142	0.556 ± 0.090	14.8 / 16
UniSO	0.850 ± 0.062	0.915 ± 0.015	0.489 ± 0.062	0.947 ± 0.036	**0.673 ± 0.136**	6.4 / 16
DUO	**0.975 ± 0.014**	**0.977 ± 0.005**	**0.610 ± 0.044**	0.933 ± 0.055	**0.706 ± 0.129**	**2.8 / 16**

few-shot protocols (mean±std), alongside offline references $\mathcal{D}_{\text{all}}$(best) and $\mathcal{D}_{\text{fs}}$(best) and UniSO with metadata for direct comparison [30].

Table 3: Held-out control tasks: maximum episode reward (mean±std). UniSO uses the same metadata convention as [30]. $\mathcal{D}_{\text{all}}$(best) and $\mathcal{D}_{\text{fs}}$(best) are the best logged rewards on the merged offline dataset and on the few-shot evaluation pool, respectively.

Task	$\mathcal{D}_{\text{all}}$(best)	$\mathcal{D}_{\text{fs}}$(best)	UniSO zero-shot	UniSO few-shot	DUO zero-shot	DUO few-shot
RobotPush	9.764	-4.989	3.171 ± 0.984	**7.067 ± 0.169**	2.715 ± 1.602	**15.109 ± 4.495**
Rover	1.216	-32.127	-8.888 ± 2.119	-8.239 ± 1.270	**33.489 ± 0.016**	**39.600 ± 3.183**
LunarLander	280.180	-236.793	31.186 ± 27.971	**248.573 ± 45.386**	-216.627 ± 4.602	**81.193 ± 44.657**
Mean rank	/	/	3.333 ± 0.577	**2.000 ± 1.000**	3.333 ± 1.155	**1.333 ± 0.577**

Few-shot DUO leads RobotPush and Rover here, ahead of UniSO few-shot and the offline-pool references—consistent with trajectory diffusion exploiting a small extra log when metadata stay informative. LunarLander remains hard for all methods, suggesting a train–test dynamics gap beyond missing text. Zero-shot orderings vary by task, so we emphasize comparisons to offline pools and a matched universal baseline rather than a single headline.

4.3 Qualitative visualization along reverse diffusion

We visualize oracle **max** along reverse diffusion: at each sampling step we oracle-evaluate the current partial trajectory and take the maximum objective over trajectory positions (axes in Appendix I). Figure 2 overlays four regimes on one protocol (**Superconductor**, evaluation seed 0).

ST (single-task DUO without text-metadata embeddings) and **ST+text** (single-task with frozen text-metadata embeddings) curves remain close throughout, which matches what we expect under *single-task* training: without multitask data there is little cross-task manifold structure or transferable metadata knowledge to exploit, so adding diffusion conditioning on text alone barely moves the objective. Our full model **MT+text** (our default setting) clearly separates above **ST+text**, which supports the claim that *multitask* training is doing real work. **MT+text** also tracks well above **MT+label** (multitask with task-ID conditioning only), indicating that rich textual metadata is substantially more useful than task-id conditioning alone. The oracle-**max** trace often peaks in the middle of reverse diffusion and weakens toward the final step because we deliberately maximize over *intermediate* chain states where stochasticity is higher—spuriously high spot maxima are easier—whereas the terminal decode is what we actually deploy; a mid-chain peak therefore does *not* imply that the intermediate

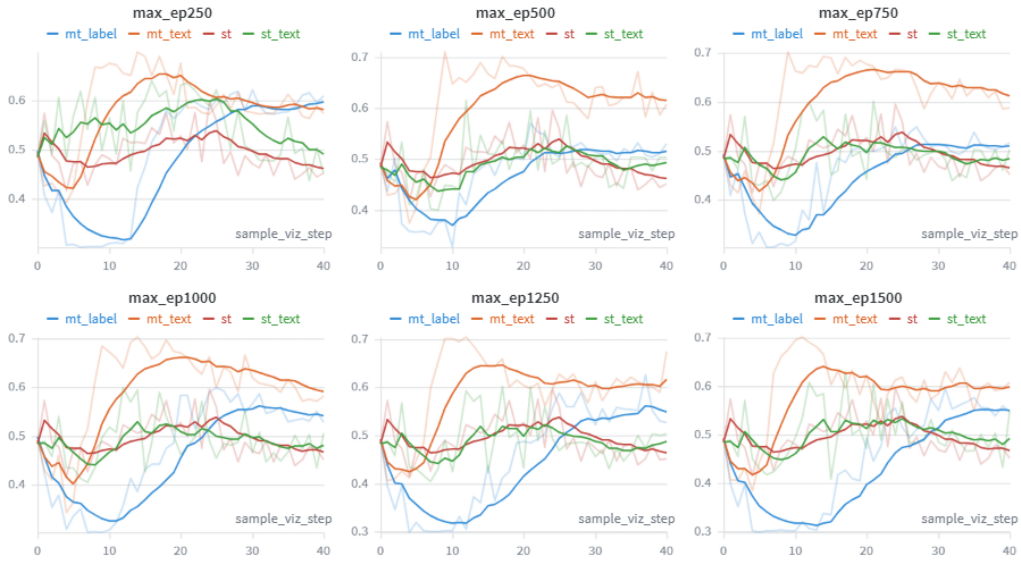

Figure 2: **Inference-time diagnostic.** Oracle **max** vs. reverse-diffusion step on **Superconductor** (evaluation seed 0). Six checkpoints in a 3×2 grid (early to late training, roughly 250–1500 epochs; learning rate 2×10^{-4}).

latent is the better model output. Vertically across checkpoints, the earliest row corresponds to $\sim$250 epochs and is still under-trained; rows in the $\sim$500–1000 epoch band look strongest, while $\sim$1500 epochs shows a mild decline consistent with overfitting to the finite offline log.

4.4 Additional experiments

We include three auxiliary analyses to clarify which parts of DUO matter beyond the headline reward tables. First, Appendix G sweeps the classifier-free guidance weight on text metadata; moderate guidance generally balances continuous robotics and DNA objectives, while overly large weights can hurt some GTOPX tasks, consistent with guidance pushing samples away from the offline support. Second, Appendix H shows that the cross-entropy auxiliary mainly improves the discrete TF Bind tasks, with smaller changes on continuous and GTOPX rows, supporting its role as a targeted correction for discrete decoding rather than a global regularizer. Finally, Appendix I expands Figure 2 with mean and top-8 oracle traces, which helps distinguish broad trajectory improvement from isolated high-scoring spikes; Appendix E lists the trajectory-construction hyperparameters used throughout.

5 Conclusion

In this paper, we present DUO, coupling metadata-aware multitask conditioning with latent trajectory diffusion behind a shared Transformer–VAE bottleneck. Empirically it compares favorably to UniSO and strong single-task optimizers on the benchmarks we report, suggesting that universal offline optimization benefits from combining semantic task conditioning with latent trajectory diffusion rather than relying only on forward reward prediction followed by search. One limitation is that our current evaluation still focuses on a moderate collection of Design-Bench and SOO-Bench tasks rather than Vizier-scale suites [6], and the approach depends on informative task metadata and diffusion sampling. Natural next steps include broader prompt studies, comparisons with diffusion–LLM hybrids [42], stronger discrete encoders, multi-objective extensions [38], improved samplers [44], and distillation for deployment.

References

[1] Michael Ahn, Henry Zhu, Kristian Hartikainen, Hugo Ponte, Abhishek Gupta, Sergey Levine, and Vikash Kumar. ROBEL: Robotics Benchmarks for Learning with Low-Cost Robots, 2019.

[2] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is Conditional Generative Modeling all you need for Decision-Making?, 2023.

[3] Luis A. Barrera, Anastasia Vedenko, Jesse V. Kurland, Julia M. Rogers, Stephen S. Gisselbrecht, Elizabeth J. Rossin, Jaie Woodard, Luca Mariani, Kian Hong Kock, Sachi Inukai, Trevor Siggers, Leila Shokri, Raluca Gordân, Nidhi Sahni, Chris Cotsapas, Tong Hao, Song Yi, Manolis Kellis, Mark J. Daly, Marc Vidal, David E. Hill, and Martha L. Bulyk. Survey of variation in human transcription factors reveals prevalent DNA binding changes. *Science*, 351(6280):1450–1454, 2016. doi: 10.1126/science.aad2257.

[4] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. https://arxiv.org/abs/1606.01540v1, 2016.

[5] Can Sam Chen, Christopher Beckham, Zixuan Liu, Xue Liu, and Christopher Pal. Robust Guided Diffusion for Offline Black-Box Optimization, 2024.

[6] Yutian Chen, Xingyou Song, Chansoo Lee, Zi Wang, Qiuyi Zhang, David Dohan, Kazuya Kawakami, Greg Kochanski, Arnaud Doucet, Marc'aurelio Ranzato, Sagi Perel, and Nando de Freitas. Towards Learning Universal Hyperparameter Optimizers with Transformers, 2022.

[7] Zhou Fan, Xinran Han, and Zi Wang. Transfer Learning for Bayesian Optimization on Heterogeneous Search Spaces, 2023.

[8] Roman Garnett. *Bayesian Optimization*. Cambridge University Press, 2023. ISBN 978-1-108-62355-1.

[9] Kam Hamidieh. A Data-Driven Statistical Model for Predicting the Critical Temperature of a Superconductor, 2018.

[10] Jonathan Ho and Tim Salimans. Classifier-Free Diffusion Guidance, 2022.

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models, 2020.

[12] Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with Diffusion for Flexible Behavior Synthesis, 2022.

[13] Minsu Kim, Jiayao Gu, Ye Yuan, Taeyoung Yun, Zixuan Liu, Yoshua Bengio, and Can Chen. Offline Model-Based Optimization: Comprehensive Review, 2026.

[14] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes, 2022.

[15] Siddarth Krishnamoorthy, Satvik Mehul Mashkaria, and Aditya Grover. Diffusion Models for Black-Box Optimization, 2023.

[16] Siddarth Krishnamoorthy, Satvik Mehul Mashkaria, and Aditya Grover. Generative Pretraining for Black-Box Optimization, 2023.

[17] Aviral Kumar and Sergey Levine. Model Inversion Networks for Model-Based Optimization, 2019.

[18] Xiaobin Li, Kai Wu, Yujian Betterest Li, Xiaoyu Zhang, Handing Wang, and Jing Liu. Pretrained Optimization Model for Zero-Shot Black Box Optimization. https://arxiv.org/abs/2405.03728v2, 2024.

[19] Zhixuan Liang, Yao Mu, Mingyu Ding, Fei Ni, Masayoshi Tomizuka, and Ping Luo. AdaptDiffuser: Diffusion Models as Adaptive Self-evolving Planners, 2023.

[20] Natalie Maus, Yimeng Zeng, Haydn Thomas Jones, Yining Huang, Gaurav Ng Goel, Alden Rose, Kyurae Kim, Hyun-Su Lee, Marcelo Der Torossian Torres, Fangping Wan, Cesar de la Fuente-Nunez, Mark Yatskar, Osbert Bastani, and Jacob R. Gardner. Purely Agentic Black-Box Optimization for Biological Design, 2026.

[21] Tung Nguyen, Sudhanshu Agrawal, and Aditya Grover. ExPT: Synthetic Pretraining for Few-Shot Experimental Design, 2023.

[22] Tung Nguyen, Qiuyi Zhang, Bangding Yang, Chansoo Lee, Jorg Bornschein, Yingjie Miao, Sagi Perel, Yutian Chen, and Xingyou Song. Predicting from Strings: Language Model Embeddings for Bayesian Optimization, 2024.

[23] Tung Nguyen, Qiuyi Zhang, Bangding Yang, Chansoo Lee, Jorg Bornschein, Yingjie Miao, Sagi Perel, Yutian Chen, and Xingyou Song. Language Model Embeddings Can Be Sufficient for Bayesian Optimization, 2025.

[24] Ian Parberry. *Introduction to Game Physics with Box2D*. CRC Press, 2013. ISBN 978-1-4665-6577-7.

[25] Hong Qian, Yiyi Zhu, Xiang Shu, Shuo Liu, Yaolin Wen, Xin An, Huakang Lu, Aimin Zhou, Ke Tang, and Yang Yu. SOO-Bench: Benchmarks for Evaluating the Stability of Offline Black-Box Optimization. In *The Thirteenth International Conference on Learning Representations*, 2024.

[26] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410.

[27] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models, 2022.

[28] Martin Schlueter, Mehdi Neshat, Mohamed Wahib, Masaharu Munetomo, and Markus Wagner. GTOPX space mission benchmarks. *SoftwareX*, 14:100666, 2021. doi: 10.1016/j.softx.2021.100666.

[29] Xingyou Song, Oscar Li, Chansoo Lee, Bangding Yang, Daiyi Peng, Sagi Perel, and Yutian Chen. OmniPred: Language Models as Universal Regressors, 2025.

[30] Rong-Xi Tan, Ming Chen, Ke Xue, Yao Wang, Yaoyuan Wang, Sheng Fu, and Chao Qian. Towards Universal Offline Black-Box Optimization via Learning Language Model Embeddings, 2025.

[31] Rong-Xi Tan, Ke Xue, Shen-Huan Lyu, Haopu Shang, Yao Wang, Yaoyuan Wang, Sheng Fu, and Chao Qian. Offline Model-Based Optimization by Learning to Rank, 2025.

[32] Brandon Trabucco, Aviral Kumar, Xinyang Geng, and Sergey Levine. Conservative Objective Models for Effective Offline Model-Based Optimization, 2021.

[33] Brandon Trabucco, Xinyang Geng, Aviral Kumar, and Sergey Levine. Design-Bench: Benchmarks for Data-Driven Offline Model-Based Optimization, 2022.

[34] Austin Tripp, Erik Daxberger, and José Miguel Hernández-Lobato. Sample-Efficient Optimization in the Latent Space of Deep Generative Models via Weighted Retraining, 2020.

[35] Masatoshi Uehara, Yulai Zhao, Ehsan Hajiramezanali, Gabriele Scalia, Gökcen Eraslan, Avantika Lal, Sergey Levine, and Tommaso Biancalani. Bridging Model-Based Optimization and Generative Modeling via Conservative Fine-Tuning of Diffusion Models, 2024.

[36] Shukuan Wang, Ke Xue, Lei Song, Xiaobin Huang, and Chao Qian. Monte Carlo Tree Search based Space Transfer for Black-box Optimization, 2024.

[37] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. https://arxiv.org/abs/2002.10957v2, 2020.

[38] Ke Xue, Rong-Xi Tan, Xiaobin Huang, and Chao Qian. Offline Multi-Objective Optimization, 2024.

[39] Peiyu Yu, Dinghuai Zhang, Hengzhi He, Xiaojian Ma, Ruiyao Miao, Yifan Lu, Yasi Zhang, Deqian Kong, Ruiqi Gao, Jianwen Xie, Guang Cheng, and Ying Nian Wu. Latent Energy-Based Odyssey: Black-Box Optimization via Expanded Exploration in the Energy-Based Latent Space, 2024.

[40] Sihyun Yu, Sungsoo Ahn, Le Song, and Jinwoo Shin. RoMA: Robust Model Adaptation for Offline Model-based Optimization, 2021.

[41] Ye Yuan, Can Chen, Zixuan Liu, Willie Neiswanger, and Xue Liu. Importance-aware Co-teaching for Offline Model-based Optimization, 2023.

[42] Ye Yuan, Can, Chen, Zipeng Sun, Dinghuai Zhang, Christopher Pal, and Xue Liu. Diffusion Large Language Models for Black-Box Optimization, 2026.

[43] Taeyoung Yun, Sujin Yun, Jaewoo Lee, and Jinkyoo Park. Guided Trajectory Generation with Diffusion Models for Offline Model-based Optimization, 2024.

[44] Taeyoung Yun, Kiyoung Om, Jaewoo Lee, Sujin Yun, and Jinkyoo Park. Posterior Inference with Diffusion Models for High-dimensional Black-box Optimization, 2025.

A Extended preliminaries

This appendix collects formal definitions and standard objectives that parallel the treatment in our thesis-style exposition, so that the main paper can stay focused on modeling choices. Readers comfortable with latent diffusion and offline RL-style notation may skim and proceed to Appendix B.

A.1 Single-task offline black-box optimization

Let $\mathcal{X}$ denote a search space and $f : \mathcal{X} \to \mathbb{R}$ an unknown objective accessible only through past evaluations. Given a static dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ with $y_i = f(x_i)$, offline black-box optimization seeks

$$x^\star \in \arg\max_{x \in \mathcal{X}} f(x) \tag{18}$$

using only information contained in $\mathcal{D}$ and inductive biases encoded in the algorithm [33]. Crucially, no additional queries $f(x)$ are permitted during learning: this models expensive simulators, wet-lab measurements, or any setting where online interaction is prohibitively costly.

A.2 Universal offline black-box optimization

Universal offline BBO [30] considers a finite suite of tasks $\mathcal{T}$. Each task $\tau \in \mathcal{T}$ is equipped with its own space $\mathcal{X}_\tau$, dimension d_τ, objective f_τ, dataset $\mathcal{D}_\tau$, and metadata m_τ (names, descriptions, goals, variable types, ...). The goal is to learn a *single* shared model that, when conditioned on m_τ, proposes high-scoring designs for every training task and—when metadata are available at deployment—generalizes to held-out tasks in zero- or few-shot regimes.

A.3 Variational autoencoders (VAE)

A VAE [14] consists of an encoder and decoder, inducing an approximate posterior $q_\phi(z|x)$ and reconstructing from latent z. A common training objective combines reconstruction and a KL penalty to a prior $p(z)$:

$$\mathcal{L}_{\mathrm{VAE}} = \mathbb{E}_{z \sim q_\phi(z|x)} \left[\|x - \mathrm{Dec}(z)\|_2^2 \right] + D_{\mathrm{KL}}\big(q_\phi(z|x) \,\|\, p(z)\big). \tag{19}$$

In DUO, the VAE maps heterogeneous native tensors into a fixed-format latent tensor so that a single diffusion backbone can be shared across tasks.

A.4 Denoising diffusion (DDPM)

DDPMs [11] define a forward noising process that gradually corrupts data x_0 into Gaussian noise over T steps. A common Gaussian parameterization is

$$q(x_t \mid x_{t-1}) = \mathcal{N}\big(\sqrt{1-\beta_t}\, x_{t-1},\, \beta_t I\big), \tag{20}$$

with variance schedule $\{\beta_t\}_{t=1}^{T}$. Let $\bar{\alpha}_t = \prod_{i=1}^{t}(1-\beta_i)$; then one may sample

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon, \quad \epsilon \sim \mathcal{N}(0, I), \tag{21}$$

which is the same reparameterization used in the latent trajectory model of Section 3. The reverse process is learned by a network ϵ_θ that predicts noise from (x_t, t) (optionally conditioned on task signals). A widely used simplified noise-prediction objective is

$$\mathcal{L}_{\mathrm{diff}} = \mathbb{E}_{t, x_0, \epsilon} \left[\|\epsilon - \epsilon_\theta(x_t, t)\|_2^2 \right], \tag{22}$$

where x_t is sampled from the forward schedule given x_0.

A.5 Classifier-free guidance (CFG)

CFG [10] trains a score model with randomly dropped conditions and at sampling time mixes conditional and unconditional predictions:

$$\hat{\epsilon} = (1+s)\, \epsilon_\theta(x_t, t, c) - s\, \epsilon_\theta(x_t, t, \emptyset), \quad s \geq 0. \tag{23}$$

In DUO's default setup, the guided component c in Eq. (23) should be read as the *textual metadata embedding*; training randomly drops this vector while keeping timestep, task index, and optional return features in place, and sampling uses a scalar weight s to interpolate between conditional and text-dropped predictions. The public codebase can also factor guidance across additional condition splits for controlled comparisons (Appendix G), but we do not treat that engineering variant as a core contribution.

A.6 Discrete cross-entropy grouping

For a length-L sequence with alphabet size C, let $\mathbf{u}_i \in \mathbb{R}^C$ be logits at position i and $\mathbf{y}_i$ the one-hot label. The auxiliary loss is

$$\mathcal{L}_{\mathrm{CE}} = -\frac{1}{L}\sum_{i=1}^{L}\sum_{c=1}^{C} y_{i,c}\log\frac{\exp(u_{i,c})}{\sum_{c'}\exp(u_{i,c'})}. \tag{24}$$

A.7 Trajectory-based generative offline optimization

Synthetic improvement trajectories. Instead of modeling only the marginal distribution of high-scoring designs, trajectory-based methods construct ordered sequences $(x^{(0)}, \ldots, x^{(H)})$ that mimic iterative improvement inside the offline log [16, 43]. The generative model is trained to reproduce the *dynamics* of moving from lower- to higher-reward regions, which often yields better optimization inductive bias than static density modeling when data are limited.

BONET and GTG. BONET [16] sorts offline points by score and samples trajectories toward high values, using an autoregressive Transformer to model the sequence. GTG [43] refines this idea by chaining neighbors in design space so that successive states resemble local optimization moves, and replaces autoregressive decoding with diffusion to mitigate compounding errors. Both demonstrate that "learning how improvement unfolds" is a strong recipe for offline BBO.

Limitations for universal settings. Existing trajectory pipelines are typically bound to a single $\mathcal{X}_\tau$ and do not natively share parameters across heterogeneous spaces, nor do they consume free-form metadata beyond coarse task IDs. DUO builds on the GTG-style trajectory viewpoint but lifts trajectories into a shared latent tensor space with Transformer–VAE encoding and metadata-conditioned diffusion, directly targeting the universal formulation in Appendix A.2.

B DUO training and sampling pseudocode

Algorithm 2 mirrors the implementation order in our training scripts: optional GTG-style [43] surrogates and the VAE are fit before trajectory construction, diffusion training consumes the latent trajectories together with metadata embeddings, and the sampling loop may close with the same surrogate ranking step as in the public GTG pipeline when we seek matched evaluation budgets. Pseudocode omits engineering details (mixed-precision flags, EMA weights, gradient clipping) that do not change the scientific narrative but are documented alongside the released code.

C Tasks, datasets, and metadata

Benchmarks and splits. We follow the unconstrained task selection commonly used in universal offline BBO work [30]: **Design-Bench** [33] provides Ant Morphology, D'Kitty Morphology, Superconductor, TF Bind 8, and TF Bind 10; **SOO-Bench** [25] provides GTOPX instances 2, 3, 4, and 6. These **nine** tasks form the default multitask *training* corpus for DUO in the main experiments (Section **??**). Together they cover continuous robotics morphologies [4, 1], a surrogate materials objective [9], discrete DNA binding design [3], and continuous trajectory planning [28].

Offline logs. Every task ships with a static design–score dataset; we never query the true simulator or oracle during DUO training, matching the offline BBO definition in Design-Bench. When a benchmark exposes extremely large logs, we follow the same fixed-size subsampling practice as in the community implementations we compare against (details mirror our thesis experiments).

Algorithm 2 DUO: diffusion models for universal offline black-box optimization (outline)

Require: Task datasets $\{\mathcal{D}_k\}$, metadata $\{m_k\}$, diffusion steps T, trajectory length H, weight λ_{CE}, text embedder E
Ensure: Per-task candidates $\{x_k^\star\}$
Train task surrogates $f_{\mathrm{surr},k}$
Fit VAE $(\mathrm{Enc}, \mathrm{Dec})$
Encode points $\mathbf{z} \leftarrow \mathrm{Enc}(\mathbf{x})$; build synthetic trajectories $\mathbf{Z}$
Embed metadata $\mathbf{c}_k \leftarrow E(m_k)$; train conditional U-Net $\boldsymbol{\epsilon}_\theta$ with loss (15)
for each task k **do**
    Sample latent trajectories with classifier-free guidance on text metadata to obtain $\{\hat{\mathbf{z}}_k\}$
    Rank with $f_{\mathrm{surr},k}$, decode top-N: $x_k^\star \leftarrow \mathrm{Dec}(\hat{\mathbf{z}}_k^\star)$
end for
return $\{x_k^\star\}$

Task metadata for conditioning. As in UniSO [30], we concatenate short textual fields—typically task name, variable type, natural-language description, and stated optimization objective—into one metadata string per task, encode it with the frozen `sentence-transformers/all-MiniLM-L6-v2` encoder (MiniLM [37]; Sentence-BERT-style training [26]; 384-dimensional outputs), and supply the projected embedding to DUO's diffusion backbone (together with task-index embeddings where used). Field-level templates and longer Chinese explanations appear in the companion thesis.

Held-out transfer tasks. For zero- and few-shot transfer we additionally evaluate LunarLander, Rover, and RobotPush [4, 36, 24], using the same offline logs and metadata conventions as in [30].

D Experimental setup details

This section expands the condensed setup in Section **??**. Every experiment respects the offline protocol: the true objective is never queried during training; all learning uses static logs $\mathcal{D}_\tau$. When we mirror the GTG [43] decoding stack, lightweight surrogates may rank candidates before oracle calls, matching that codebase rather than defining a new protocol element.

Design-Bench evaluation. Following the experimental setting of Tan et al. [30] on Design-Bench [33], for each task we independently sample 128 candidate designs from the learned policy, evaluate each with the oracle, and report the maximum objective. Mean and standard deviation aggregate evaluation seeds as in Section **??**. **Avg. Rank** is computed from per-task ranks among the listed competing methods, normalized as in each table caption. Numbers taken from prior work retain the statistics reported in the original publications.

Random seeds. All DUO runs that we train and evaluate ourselves use **eight** random seeds $\{0, 1, \ldots, 7\}$; reported means and standard deviations pool these runs. Tables that include external baselines may quote mean$\pm$std only when the original release provides multiple seeds, but our own rows always follow the eight-seed protocol above.

Multitask training corpus. Unless we explicitly ablate single-task variants, DUO is trained *once* on the **nine** tasks listed in Appendix C, using textual metadata and task-index conditioning. Any table with fewer than nine task columns still uses this same checkpoint and simply omits tasks that fall outside that evaluation protocol (e.g., standard Design-Bench scaling reports five columns). Held-out control tasks in Section 4.2 are *not* part of this nine-task fit.

Text embeddings. Runs that condition on textual task metadata use the frozen Hugging Face checkpoint `sentence-transformers/all-MiniLM-L6-v2` (a 384-dimensional MiniLM sentence encoder [37] in the Sentence-BERT lineage [26]); see Section 3 for fusion into diffusion.

E Trajectory construction hyperparameters

Synthetic trajectories are controlled by four quantities: the number of sampled paths per refresh n_{traj}, the neighborhood size k used when chaining latent states, the edge-quality threshold ε, and the fixed trajectory length $H{=}64$. Here k limits how many nearest neighbors are considered when forming a synthetic edge, and ε drops edges whose score improvement is too small so that trajectories do not wander through flat or misleading regions.

Design-Bench tasks. For Ant, D'Kitty, Superconductor, TF Bind 8, and TF Bind 10 we **reuse the trajectory hyperparameter recommendations from GTG** [43] (i.e., the same k and ε choices advocated there for single-task trajectory diffusion), so that our multitask DUO remains aligned with the best-studied single-task trajectory baseline on these domains.

SOO-Bench (GTOPX) tasks. For GTOPX 2/3/4/6 we do not rely on an external recipe; instead we selected (k, ε) via **lightweight single-task screening** over a small candidate grid on each mission, picked the setting with the best validation proxy on that task alone, and then froze those values for all multitask training and reporting (matching the procedure documented in our thesis).

Table 4 lists the resulting $(n_{\text{traj}}, k, \varepsilon)$ for every training task.

Table 4: Per-task trajectory construction hyperparameters (n_{traj}, neighborhood size k, threshold ε); latent trajectory length $H{=}64$. Design-Bench rows follow GTG-recommended (k, ε); GTOPX rows use values from single-task screening (Section text).

Task	n_{traj}	k	ε
Ant	1000	20	0.05
D'Kitty	1000	20	0.01
Superconductor	1000	20	0.05
TF Bind 8	1000	50	0.05
TF Bind 10	1000	50	0.05
GTOPX 2	1000	20	0.05
GTOPX 3	1000	10	0.01
GTOPX 4	1000	20	0.05
GTOPX 6	1000	20	0.01

F Training-corpus comparison and ablations

Section 4.1 reports aggregate multitask performance together with the Design-Bench view; this appendix gives the full training-corpus grid used for that comparison. Table 5 places the universal regressor **UniSO** and single-task trajectory diffusion **GTG** next to four **DUO** training regimes: **ST** and **ST+text** (single-task, without vs. with frozen text-metadata embeddings), and **MT+label** vs. **MT+text** (multitask with task-ID conditioning only vs. multitask with full text metadata—our default). The mean-rank row summarizes cross-task ordering among these six learned columns together with the two multitask baselines (see table caption).

The **ST** column diffuses entirely in the shared VAE latent while GTG diffuses in native coordinates; scores are often comparable and trade wins row-wise, which supports latent trajectory diffusion as a workable interface rather than a bookkeeping trick before multitask sharing. Under single-task training, **ST+text** moves only modestly relative to **ST**, consistent with Section 4.3: without multitask context, metadata has little structure to exploit and text-only conditioning rarely reshapes the objective. By contrast, **MT+text** improves markedly over **ST+text** on average and attains the best mean rank among the four DUO columns, matching the diagnostic where **MT+text** separates upward while **ST** and **ST+text** remain entangled—evidence that shared training, not text alone, drives the gain. **MT+text** also dominates **MT+label**, paralleling Section 4.3: task IDs are a weak substitute for full textual descriptions when steering a universal denoiser. Finally, relative to Section 4.1, **UniSO** and **GTG** remain strong anchors: **UniSO** stays competitive on several DNA and trajectory rows, whereas **MT+text** tends to show the clearest margins on continuous robotics and selected GTOPX tasks, so the grid is best read task-by-task rather than as a single global ordering [30].

Table 5: Multitask training corpus at **raw** simulator scale (mean±std). Baselines: UniSO [30] and single-task GTG [43]. The four DUO columns are latent trajectory diffusion ablations: *(i)* single-task without metadata embeddings; *(ii)* single-task with frozen text-metadata embeddings; *(iii)* multitask with task-ID conditioning only; *(iv)* our method (multitask with full text-metadata embeddings). $\mathcal{D}$(best) is the best logged reward. Mean rank averages per-task ranks across the six learned columns (lower is better). A compact three-way comparison (UniSO, GTG, our method) is in Table 1.

Task	$\mathcal{D}$(best)	UniSO	GTG	DUO (single-task)	DUO (single-task + metadata emb.)	DUO (multitask + task label)	DUO (multitask + metadata emb.)
Ant	165.326	455.658 ± 39.188	452.330 ± 61.502	474.207 ± 11.743	490.691 ± 17.299	**565.699 ± 8.628**	**565.873 ± 13.651**
D'Kitty	199.363	222.007 ± 33.677	264.857 ± 17.571	257.494 ± 17.376	252.830 ± 10.977	**300.777 ± 13.149**	**312.350 ± 6.353**
Superconductor	74.000	82.642 ± 3.467	89.802 ± 6.763	78.774 ± 4.163	76.990 ± 3.889	**100.741 ± 12.287**	**112.880 ± 8.088**
TF Bind 8	0.439	0.857 ± 0.069	0.934 ± 0.030	**0.953 ± 0.035**	**0.952 ± 0.038**	0.724 ± 0.073	0.933 ± 0.055
TF Bind 10	0.005	**0.944 ± 0.794**	0.632 ± 0.106	0.641 ± 0.144	0.787 ± 0.076	0.632 ± 0.171	**0.955 ± 0.516**
GTOPX 2	-196.663	-90.106 ± 8.223	-72.163 ± 8.495	**-66.664 ± 15.951**	-75.231 ± 20.343	-110.869 ± 13.664	**-44.477 ± 4.761**
GTOPX 3	-152.364	**-44.427 ± 13.456**	-53.866 ± 9.877	-47.522 ± 6.089	-51.900 ± 13.122	-51.175 ± 14.311	**-29.310 ± 2.237**
GTOPX 4	-216.709	-74.779 ± 11.032	-72.749 ± 6.632	**-57.550 ± 12.458**	-67.212 ± 11.006	-63.167 ± 13.669	**-51.019 ± 6.536**
GTOPX 6	-111.666	**-48.244 ± 5.766**	-55.834 ± 9.655	-59.971 ± 9.638	-58.092 ± 2.617	-61.664 ± 10.439	**-33.449 ± 12.583**
Mean rank	/	4.111 ± 1.691	4.222 ± 1.481	**3.333 ± 1.414**	4.000 ± 1.225	4.000 ± 1.803	**1.333 ± 1.000**

Design-Bench leaderboard (complete table). Table 6 lists all competing methods with **Source** venues; Section 4.1 uses a shorter method list in Table 2.

Table 6: **Full** Design-Bench leaderboard under standard per-task rescaling (mean±std when reported). **Source** lists the publication venue (or Design-Bench baselines [33]). **Avg. Rank**: mean per-task rank divided by the number of competing methods (26). Evaluation: Appendix D. Short leaderboard: Table 2.

Method	Source	Ant	D'Kitty	Superconductor	TF-Bind-8	TF-Bind-10	Avg. Rank
$\mathcal{D}$(best)	/	0.565	0.884	0.400	0.439	0.467	/
BO-qEI	Design-Bench baselines [33]	0.812 ± 0.000	0.896 ± 0.000	0.382 ± 0.013	0.802 ± 0.081	0.628 ± 0.036	20.4 / 26
CMA-ES		**1.712 ± 0.754**	0.725 ± 0.002	0.463 ± 0.042	0.944 ± 0.017	0.641 ± 0.036	12.8 / 26
REINFORCE		0.248 ± 0.039	0.541 ± 0.196	0.478 ± 0.017	0.935 ± 0.049	**0.673 ± 0.074**	15.7 / 26
Grad. Ascent		0.273 ± 0.023	0.853 ± 0.018	0.510 ± 0.028	0.969 ± 0.021	0.646 ± 0.037	12.8 / 26
Grad. Ascent Mean		0.306 ± 0.053	0.875 ± 0.024	0.508 ± 0.019	**0.985 ± 0.008**	0.633 ± 0.030	12.4 / 26
Grad. Ascent Min		0.282 ± 0.033	0.884 ± 0.018	0.514 ± 0.020	0.979 ± 0.014	0.632 ± 0.027	12.6 / 26
CbAS	ICML'19	0.846 ± 0.032	0.896 ± 0.009	0.421 ± 0.049	0.921 ± 0.046	0.630 ± 0.039	17.7 / 26
MINs	ICML'19	0.906 ± 0.024	0.939 ± 0.007	0.464 ± 0.023	0.910 ± 0.051	0.633 ± 0.034	14.6 / 26
DDOM	ICML'23	0.908 ± 0.024	0.930 ± 0.005	0.452 ± 0.028	0.913 ± 0.047	0.616 ± 0.018	16.4 / 26
BONET	ICML'23	0.921 ± 0.031	0.949 ± 0.016	0.390 ± 0.022	0.798 ± 0.123	0.575 ± 0.039	17.3 / 26
GTG	NeurIPS'24	0.855 ± 0.044	0.942 ± 0.017	0.480 ± 0.055	0.910 ± 0.040	0.619 ± 0.029	15.6 / 26
COMs	ICML'21	0.916 ± 0.026	0.949 ± 0.016	0.460 ± 0.040	0.953 ± 0.038	0.644 ± 0.052	10.5 / 26
RoMA	ICML'21	0.430 ± 0.048	0.767 ± 0.031	0.494 ± 0.025	0.665 ± 0.000	0.553 ± 0.000	20.9 / 26
IOM	NeurIPS'22	0.889 ± 0.034	0.928 ± 0.008	0.491 ± 0.034	0.925 ± 0.054	0.628 ± 0.036	14.5 / 26
BDI	NeurIPS'22	0.963 ± 0.000	0.941 ± 0.000	0.508 ± 0.013	0.973 ± 0.000	0.658 ± 0.000	6.5 / 26
ICT	NeurIPS'23	0.915 ± 0.024	0.947 ± 0.009	0.494 ± 0.026	0.897 ± 0.050	0.659 ± 0.024	10.6 / 26
Tri-Mentoring	NeurIPS'23	0.891 ± 0.011	0.947 ± 0.005	0.503 ± 0.013	0.956 ± 0.000	0.662 ± 0.012	8.3 / 26
PGS	AAAI'24	0.715 ± 0.046	0.954 ± 0.022	0.444 ± 0.020	0.889 ± 0.061	0.634 ± 0.040	15.4 / 26
FGM	AISTATS'24	0.923 ± 0.023	0.944 ± 0.014	0.481 ± 0.024	0.811 ± 0.079	0.611 ± 0.008	15.0 / 26
MATCH-OPT	ICML'24	0.933 ± 0.016	0.952 ± 0.008	0.504 ± 0.021	0.824 ± 0.067	0.655 ± 0.050	9.4 / 26
GABO	NeurIPS'24 Oral	0.038 ± 0.012	0.719 ± 0.001	0.374 ± 0.020	0.926 ± 0.038	0.619 ± 0.043	22.1 / 26
LTR	ICLR'25	0.949 ± 0.025	0.962 ± 0.015	**0.517 ± 0.029**	**0.981 ± 0.012**	0.670 ± 0.035	**3.2 / 26**
ROOT	NeurIPS'25 Spotlight	0.958 ± 0.012	0.971 ± 0.005	0.451 ± 0.032	0.977 ± 0.015	0.653 ± 0.030	7.8 / 26
DynAMO-Adam	ICML'25	0.113 ± 0.085	0.789 ± 0.059	0.413 ± 0.106	0.719 ± 0.142	0.556 ± 0.090	24.0 / 26
UniSO	ICML'25	0.850 ± 0.062	0.915 ± 0.015	0.489 ± 0.062	0.947 ± 0.036	**0.673 ± 0.136**	11.1 / 26
DUO	/	**0.975 ± 0.014**	**0.977 ± 0.005**	**0.610 ± 0.044**	0.933 ± 0.055	**0.706 ± 0.129**	3.4 / 26

G Additional analysis: metadata guidance strength

The main text (Section 4) reports primary benchmarks, a qualitative reverse-diffusion diagnostic, and held-out transfer; Table 5, additional discussion, and the full Design-Bench leaderboard appear in Appendix F. The discrete cross-entropy ablation is in Appendix H; full diffusion-curve panels are in Appendix I. Here we sweep the scalar classifier-free guidance weight w on the text-metadata embedding, holding the remainder of the conditioning stack to its default settings. Other condition splits can be ablated in code; those variants are not emphasized in the main narrative.

Table 7 aggregates mean ± std over all evaluation runs in the ablation. Moderate weights tend to balance continuous robotics tasks and DNA objectives; very large weights occasionally hurt GTOPX

instances, suggesting that overly aggressive semantic guidance can push sampling outside the offline data support—analogous to excessive guidance in offline RL.

Table 7: Ablation of classifier-free guidance strength on the text-metadata embedding (w): mean $\pm$ std of maximum episode reward over evaluation runs. Baselines: $\mathcal{D}$(best), UniSO, GTG (single-task).

Task	$\mathcal{D}$(best)	UniSO	GTG ST	w=0	w=1	w=2	w=4	w=8	w=16	w=32	w=64
Ant	165.326	455.658 ± 39.188	452.330 ± 61.502	**568.346 ± 10.406**	**574.388 ± 8.549**	566.263 ± 8.668	561.212 ± 5.810	566.659 ± 9.082	564.747 ± 3.107	567.857 ± 8.753	565.456 ± 7.107
D'Kitty	199.363	222.007 ± 33.677	264.857 ± 17.571	**311.384 ± 3.566**	310.822 ± 1.518	308.203 ± 8.994	310.155 ± 11.200	307.139 ± 12.211	**311.659 ± 9.848**	305.326 ± 13.748	304.468 ± 14.558
Superconductor	74.000	82.642 ± 3.467	89.802 ± 6.763	90.914 ± 3.655	90.833 ± 3.615	92.202 ± 3.700	98.833 ± 6.457	104.827 ± 7.582	**108.596 ± 12.243**	104.092 ± 8.325	**106.923 ± 7.899**
TF Bind 8	0.439	**0.857 ± 0.069**	**0.934 ± 0.030**	0.718 ± 0.095	0.718 ± 0.095	0.780 ± 0.071	0.780 ± 0.061	0.800 ± 0.091	0.717 ± 0.113	0.821 ± 0.034	0.816 ± 0.116
TF Bind 10	0.005	**0.944 ± 0.794**	0.632 ± 0.106	0.627 ± 0.227	0.627 ± 0.227	0.650 ± 0.249	0.652 ± 0.300	**0.730 ± 0.356**	0.597 ± 0.108	0.649 ± 0.082	0.663 ± 0.058
GTOPX 2	-196.663	-90.106 ± 8.223	-72.163 ± 8.495	-94.874 ± 24.696	-99.536 ± 25.617	-126.143 ± 27.620	-82.143 ± 37.817	**-48.779 ± 9.781**	**-65.063 ± 37.999**	-69.415 ± 47.726	-76.584 ± 38.446
GTOPX 3	-152.364	-44.427 ± 13.456	-53.866 ± 9.877	-85.963 ± 32.525	-86.938 ± 33.593	-65.658 ± 15.598	-48.934 ± 26.540	**-29.687 ± 1.663**	**-34.348 ± 9.069**	**-29.687 ± 1.663**	-47.244 ± 24.893
GTOPX 4	-216.709	-74.779 ± 11.032	-72.749 ± 6.632	-136.050 ± 84.354	-135.972 ± 84.455	-80.577 ± 12.241	-81.501 ± 26.936	**-49.997 ± 7.345**	**-49.997 ± 7.345**	**-49.997 ± 7.345**	**-57.299 ± 17.860**
GTOPX 6	-111.666	-48.244 ± 5.766	-55.834 ± 9.655	-67.402 ± 4.817	-69.440 ± 4.137	-65.996 ± 9.502	-48.609 ± 17.321	**-30.053 ± 1.303**	**-30.053 ± 1.303**	**-34.417 ± 10.093**	-37.325 ± 15.452
Mean rank	/	6.000 ± 3.317	6.556 ± 2.833	7.111 ± 3.008	7.444 ± 3.196	6.667 ± 1.732	6.000 ± 1.500	**2.889 ± 1.746**	4.167 ± 3.775	**3.611 ± 1.799**	4.556 ± 1.740

H Discrete cross-entropy auxiliary loss

Discrete tasks are often trained with continuous relaxations in latent space; mean-squared trajectory losses alone may not penalize one-hot decoding errors sharply enough. Table 8 compares multitask DUO with text metadata when $\lambda_{\mathrm{CE}}{=}0$ versus $\lambda_{\mathrm{CE}}{=}0.005$ in the auxiliary term (24). All entries use the same classifier-free guidance strength $w{=}8$ on the text-metadata embedding; other hyperparameters are held fixed across the two columns.

Table 8: Auxiliary cross-entropy ablation on multitask DUO with text metadata (maximum episode reward, mean$\pm$std; $\lambda_{\mathrm{CE}}{=}0$ vs. 0.005). **Bold** marks TF Bind rows: CE sharpens discrete-task performance, while co-trained continuous tasks show only small shifts relative to the reported dispersion (no consistent degradation).

Task	$\lambda_{\mathrm{CE}}{=}0$	$\lambda_{\mathrm{CE}}{=}0.005$
Ant	559.863 $\pm$ 8.221	565.873 $\pm$ 13.651
D'Kitty	313.803 $\pm$ 4.345	312.350 $\pm$ 6.353
Superconductor	108.143 $\pm$ 8.186	112.880 $\pm$ 8.088
TF Bind 8	$\mathbf{0.855 \pm 0.052}$	$\mathbf{0.933 \pm 0.055}$
TF Bind 10	$\mathbf{0.642 \pm 0.174}$	$\mathbf{0.955 \pm 0.516}$
GTOPX 2	-44.477 $\pm$ 4.761	-44.477 $\pm$ 4.761
GTOPX 3	-29.310 $\pm$ 2.237	-29.310 $\pm$ 2.237
GTOPX 4	-54.560 $\pm$ 14.187	-51.019 $\pm$ 6.536
GTOPX 6	-28.580 $\pm$ 2.489	-33.449 $\pm$ 12.583
Mean rank	1.667 $\pm$ 0.433	1.333 $\pm$ 0.433

Turning on the cross-entropy term yields clear improvements on the discrete TF Bind 8/10 rows in Table 8, while Ant, D'Kitty, Superconductor, and GTOPX entries move only modestly and remain on the same scale once standard deviations are considered—consistent with CE acting as a targeted correction for discrete decoding rather than a global regularizer that reshapes the entire multitask solution. The mean rank row shifts slightly toward $\lambda_{\mathrm{CE}}{=}0.005$, matching the view that the auxiliary loss primarily aligns latent training with discrete evaluation.

I Additional analysis: reverse-diffusion diagnostic curves

This appendix complements Section 4.3 with the full $3{\times}6$ panel of reverse-diffusion diagnostics on **Superconductor** (evaluation seed 0), using the same four regimes as Figure 2: **ST**, **ST+text**, **MT+label**, and **MT+text** (our method: multitask with full text-metadata embeddings). Figure 2 shows only the **oracle max** column; Figure 3 adds **mean** and **top-8** alongside **max** for rows R1–R6 (checkpoints in serialized export order). All runs use learning rate $2{\times}10^{-4}$. Section 4.3 interprets regime differences, checkpoint aging, and the oracle-**max** shape; here we focus on how **mean** and **top-8** should be read.

The horizontal axis counts reverse-diffusion steps from noisy latents to a denoised trajectory of horizon $H{=}64$. **Mean** averages the oracle objective over all generated trajectory positions *except* any context tokens from the conditioning stack, summarizing *global* trajectory quality rather than a

single extremum. **Top-8** averages the oracle values of the eight highest-scoring positions (top $8/H$ by oracle at the current step), which stresses whether strong objectives appear in multiple coordinates instead of one outlier slot. **Max** is the best oracle anywhere on the partial chain at the current step (the column duplicated in the main figure); we do not reinterpret its mid-chain peaks here. These traces call the *true* simulator oracle for visualization only and omit the deployment-time surrogate filter, so they are not directly comparable to proxy-ranked entries in the main tables.

When **top-8** rises in step with or ahead of **mean**, high oracle mass is spreading across the trajectory rather than concentrating in a lone spike; flat **mean** with volatile **top-8** instead suggests occasional lucky coordinates without broad improvement. Comparing the two columns across R1–R6 therefore highlights whether later checkpoints improve *distributed* quality or mainly sharpen a few positions; we defer narrative about training duration and overfitting to Section 4.3.

Figure 3: **Reverse-diffusion diagnostics (illustrative). Superconductor**, seed 0: oracle **mean**, **top-8** mean, and **max** along reverse-diffusion steps (see text). Rows R1–R6 are exported training checkpoints from early to late in dashboard order; learning rate 2×10^{-4}. Curves compare **ST**, **ST+text**, **MT+label**, and **MT+text** as defined in Figure 2.