Robust Guided Diffusion for Offline Black-box Optimization

Anonymous Author(s)
Affiliation
Address
email

Abstract

Offline black-box optimization aims to maximize a black-box function using an offline dataset of designs and their measured properties. Two main approaches have emerged: the forward approach, which learns a mapping from input to its value, thereby acting as a proxy to guide optimization, and the inverse approach, which learns a mapping from value to input for conditional generation. (a) Although proxy-free (classifier-free) diffusion shows promise in robustly modeling the inverse mapping, it lacks explicit guidance from proxies, essential for generating high-performance samples beyond the training distribution. Therefore, we propose *proxy-enhanced sampling* which utilizes the explicit guidance from a trained proxy to bolster proxy-free diffusion with enhanced sampling control. (b) Yet, the trained proxy is susceptible to out-of-distribution issues. To address this, we devise the module *diffusion-based proxy refinement*, which seamlessly integrates insights from proxy-free diffusion back into the proxy for refinement. To sum up, we propose ***R**obust **G**uided **D**iffusion for Offline Black-box Optimization* (**RGD**), combining the advantages of proxy (explicit guidance) and proxy-free diffusion (robustness) for effective conditional generation. RGD achieves state-of-the-art results on various design-bench tasks, underscoring its efficacy. Our code is here.

1 Introduction

Creating new objects to optimize specific properties is a ubiquitous challenge that spans a multitude of fields, including material science, robotic design, and genetic engineering. Traditional methods generally require interaction with a black-box function to generate new designs, a process that could be financially burdensome and potentially perilous [1, 2]. Addressing this, recent research endeavors have pivoted toward a more relevant and practical context, termed offline black-box optimization (BBO) [3, 4]. In this context, the goal is to maximize a black-box function exclusively utilizing an offline dataset of designs and their measured properties.

There are two main approaches for this task: the forward approach and the reverse approach. The forward approach entails training a deep neural network (DNN), parameterized as $\mathcal{J}_{\boldsymbol{\phi}}(\cdot)$, using the offline dataset. Once trained, the DNN acts as a proxy and provides explicit gradient guidance to enhance existing designs. However, this technique is susceptible to the out-of-distribution (OOD) issue, leading to potential overestimation of unseen designs and resulting in adversarial solutions [5].

The reverse approach aims to learn a mapping from property value to input. Inputting a high value into this mapping directly yields a high-performance design. For example, MINs [6] adopts GAN [7] to model this inverse mapping, and demonstrate some success. Recent works [4] have applied proxy-free diffusion[1] [8], parameterized by $\boldsymbol{\theta}$, to model this mapping, which proves its efficacy over

[1] *Classifier-free diffusion* is for classification and adapted to *proxy-free diffusion* to generalize to regression.

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

other generative models. Proxy-free diffusion employs a score predictor $\tilde{\boldsymbol{s}}_\theta(\cdot,\cdot,\omega)$. This represents a linear combination of conditional and unconditional scores, modulated by a strength parameter ω to balance condition and diversity in the sampling process. This guidance significantly diverges from proxy (classifier) diffusion that interprets scores as classifier gradients and thus generates adversarial solutions. Such a distinction grants proxy-free diffusion its inherent robustness in generating samples.

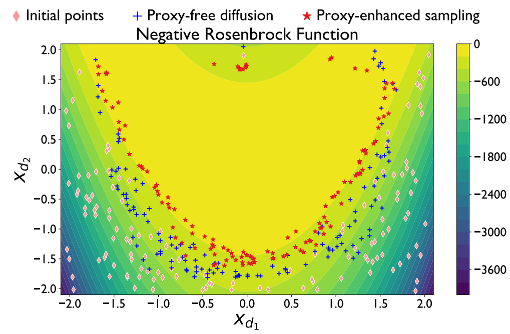

Figure 1: Motivation of explicit proxy guidance.

Nevertheless, proxy-free diffusion, initially designed for in-distribution generation, such as synthesizing specific image categories, faces limitations in offline BBO. Particularly, it struggles to generate high-performance samples that exceed the training distribution due to the lack of explicit guidance[2]. Consider, for example, the optimization of a two-dimensional variable (x_{d1}, x_{d2}) to maximize the negative Rosenbrock function [9]: $y(x_{d1}, x_{d2}) = -(1 - x_{d1})^2 - 100(x_{d2} - x_{d1}^2)^2$, as depicted in Figure 1. The objective is to steer the initial points (indicated in pink) towards the high-performance region (highlighted in yellow). While proxy-free diffusion can nudge the initial points closer to this high-performance region, the generated points (depicted in blue) fail to reach the high-performance region due to its lack of explicit proxy guidance.

To address this challenge, we introduce a *proxy-enhanced sampling* module as illustrated in Figure 2(a). It incorporates the explicit guidance from the proxy $\mathcal{J}_{\boldsymbol{\phi}}(\boldsymbol{x})$ into proxy-free diffusion to enable enhanced control over the sampling process. This module hinges on the strategic optimization of the strength parameter ω to achieve a better balance between condition and diversity, per reverse diffusion step. This incorporation not only preserves the inherent robustness of proxy-free diffusion but also leverages the explicit proxy guidance, thereby enhancing the overall conditional generation efficacy. As illustrated in Figure 1, samples (depicted in red) generated via *proxy-enhanced sampling* are more effectively guided towards, and often reach, the high-performance area (in yellow).

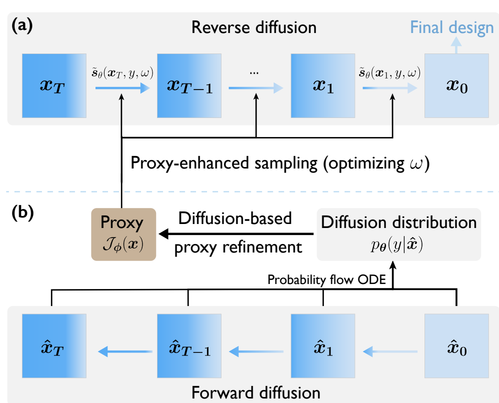

Figure 2: Overall of RGD.

Yet, the trained proxy is susceptible to out-of-distribution (OOD) issues. To address this, we devise a module *diffusion-based proxy refinement* as detailed in Figure 2(b). This module seamlessly integrates insights from proxy-free diffusion into the proxy $\mathcal{J}_{\boldsymbol{\phi}}(\boldsymbol{x})$ for refinement. Specifically, we generate a diffusion distribution $p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})$ on adversarial samples $\hat{\boldsymbol{x}}$, using the associated probability flow ODE [3]. This distribution is derived independently of a proxy, thereby exhibiting greater robustness than the proxy distribution on adversarial samples. Subsequently, we calculate the Kullback-Leibler divergence between the two distributions on adversarial samples, and use this divergence minimization as a regularization strategy to fortify the proxy's robustness and reliability.

To sum up, we propose ***R**obust **G**uided **D**iffusion for Offline Black-box Optimization* (**RGD**), a novel framework that combines the advantages of proxy (explicit guidance) and proxy-free diffusion (robustness) for effective conditional generation. Our contributions are three-fold:

- We propose a *proxy-enhanced sampling* module which incorporates proxy guidance into proxy-free diffusion to enable enhanced sampling control.
- We further develop *diffusion-based proxy refinement* which integrates insights from proxy-free diffusion back into the proxy for refinement.
- RGD delivers state-of-the-art performance on various design-bench tasks, emphasizing its efficacy.

[2] Proxy-free diffusion cannot be interpreted as a proxy and thus does not provide explicit guidance [8].
[3] Ordinary Differential Equation

2 Preliminaries

2.1 Offline Black-box Optimization

Offline black-box optimization (BBO) aims to maximize a black-box function with an offline dataset. Imagine a design space as $\mathcal{X} = \mathbb{R}^d$, where d is the design dimension. The offline BBO [3] is:

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}\in\mathcal{X}} J(\boldsymbol{x}). \tag{1}$$

In this equation, $J(\cdot)$ is the unknown objective function, and $\boldsymbol{x} \in \mathcal{X}$ is a possible design. In this context, there is an offline dataset, $\mathcal{D}$, that consists of pairs of designs and their measured properties. Specifically, each $\boldsymbol{x}$ denotes a particular design, like the size of a robot, while y indicates its related metric, such as its speed.

A common approach *gradient ascent* fits a proxy distribution $p_{\boldsymbol{\phi}}(y|\boldsymbol{x}) = \mathcal{N}(J_{\boldsymbol{\phi}}(\boldsymbol{x}), \sigma_{\boldsymbol{\phi}}(\boldsymbol{x}))$ to the offline dataset where $\boldsymbol{\phi}$ denote the proxy parameters:

$$\begin{aligned} &\arg\min_{\boldsymbol{\phi}} \mathbb{E}_{(\boldsymbol{x},y)\in\mathcal{D}}[-\log p_{\boldsymbol{\phi}}(y|\boldsymbol{x})]. \\ &= \arg\min_{\boldsymbol{\phi}} \mathbb{E}_{(\boldsymbol{x},y)\in\mathcal{D}} \log(\sqrt{2\pi}\sigma_{\boldsymbol{\phi}}(\boldsymbol{x})) + \frac{(y - J_{\boldsymbol{\phi}}(\boldsymbol{x}))^2}{2\sigma^2_{\boldsymbol{\phi}}(\boldsymbol{x})}. \end{aligned} \tag{2}$$

For the sake of consistency with terminology used in the forthcoming subsection on guided diffusion, we will refer to $p_{\boldsymbol{\phi}}(\cdot|\cdot)$ as the proxy distribution and $J_{\boldsymbol{\phi}}(\cdot)$ as the proxy. Subsequently, this approach performs gradient ascent with $J_{\boldsymbol{\phi}}(\boldsymbol{x})$, leading to high-performance designs $\boldsymbol{x}^*$:

$$\boldsymbol{x}_{\tau+1} = \boldsymbol{x}_\tau + \eta \nabla_{\boldsymbol{x}} J_{\boldsymbol{\phi}}(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}_\tau}, \quad \text{for } \tau \in [0, \mathrm{M}-1], \tag{3}$$

converging to $\boldsymbol{x}_{\mathrm{M}}$ after M steps. However, this method suffers from the out-of-distribution issue where the proxy predicts values that are notably higher than the actual values.

2.2 Diffusion Models

Diffusion models, a type of latent variable models, progressively introduce Gaussian noise to data in the forward process, while the reverse process aims to iteratively remove this noise through a learned score estimator. In this work, we utilize continuous time diffusion models governed by a stochastic differential equation (SDE), as presented in [10]. The forward SDE is formulated as:

$$d\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x}, t)dt + g(t)d\boldsymbol{w}. \tag{4}$$

where $\boldsymbol{f}(\cdot, t) : \mathbb{R}^d \to \mathbb{R}^d$ represents the drift coefficient, $g(\cdot) : \mathbb{R} \to \mathbb{R}$ denotes the diffusion coefficient and $\boldsymbol{w}$ is the standard Wiener process. This SDE transforms data distribution into noise distribution. The reverse SDE is:

$$d\boldsymbol{x} = \left[\boldsymbol{f}(\boldsymbol{x}, t) - g(t)^2 \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})\right] dt + g(t) d\bar{\boldsymbol{w}}, \tag{5}$$

with $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$ representing the score of the marginal distribution at time t, and $\bar{\boldsymbol{w}}$ symbolizing the reverse Wiener process. The score function $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$ is estimated using a time-dependent neural network $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$, enabling us to transform noise into samples. For simplicity, we will use $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t)$, implicitly including the time dependency t.

2.3 Guided Diffusion

Guided diffusion seeks to produce samples with specific desirable attributes, falling into two categories: *proxy diffusion* [11] and *proxy-free diffusion* [8]. While these were initially termed *classifier diffusion* and *classifier-free diffusion* in classification tasks, we have renamed them to *proxy diffusion* and *proxy-free diffusion*, respectively, to generalize to our regression context. Proxy diffusion combines the model's score estimate with the gradient from the proxy distribution, providing explicit guidance. However, it can be interpreted as a gradient-based adversarial attack.

Proxy-free guidance, not dependent on proxy gradients, enjoys an inherent robustness of the sampling process. Particularly, it models the score as a linear combination of an unconditional and a conditional score. A unified neural network $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, y)$ parameterizes both score types. The score $\boldsymbol{s}_\theta(\boldsymbol{x}_t, y)$

approximates the gradient of the log probability $\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|y)$, i.e., the conditional score, while $\boldsymbol{s}_\theta(\boldsymbol{x}_t)$ estimates the gradient of the log probability $\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t)$, i.e., the unconditional score. The score function follows:

$$\tilde{\boldsymbol{s}}_\theta(\boldsymbol{x}_t, y, \omega) = (1+\omega)\boldsymbol{s}_\theta(\boldsymbol{x}_t, y) - \omega \boldsymbol{s}_\theta(\boldsymbol{x}_t). \tag{6}$$

Within this context, the strength parameter ω specifies the generation's adherence to the condition y, which is set to the maximum value y_{max} in the offline dataset following [4]. Optimization of ω balances the condition and diversity. Lower ω values increase sample diversity at the expense of conformity to y, and higher values do the opposite.

3 Method

In this section, we present our method RGD, melding the strengths of proxy and proxy-free diffusion for effective conditional generation. Firstly, we describe a newly developed module termed *proxy-enhanced sampling*. It integrates explicit proxy guidance into proxy-free diffusion to enable enhanced sampling control, as detailed in Section 3.1. Subsequently, we explore *diffusion-based proxy refinement* which incorporates insights gleaned from proxy-free diffusion back into the proxy, further elaborated in Section 3.2. The overall algorithm is shown in Algorithm 1.

3.1 Proxy-enhanced Sampling

As discussed in Section 2.3, proxy-free diffusion trains an unconditional model and conditional models. Although proxy-free diffusion can generate samples aligned with most conditions, it traditionally lacks control due to the absence of an explicit proxy. This is particularly significant in offline BBO where we aim to obtain samples beyond the training distribution. Therefore, we require explicit proxy guidance to achieve enhanced sampling control. This module is outlined in Algorithm 1, Line 8- Line 16.

Algorithm 1 Robust Guided Diffusion for Offline BBO

Input: offline dataset $\mathcal{D}$, # of diffusion steps T.

1: Train proxy distribution $p_{\boldsymbol{\phi}}(y|\boldsymbol{x})$ on $\mathcal{D}$ by Eq. (2).
2: Train proxy-free diffusion model $\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, y)$ on $\mathcal{D}$.
3: `/*Diffusion-based proxy refinement*/`
4: Identify adversarial samples via grad ascent.
5: Compute diffusion distribution $p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})$ by Eq. (12).
6: Compute KL divergence loss as per Eq. (13).
7: Refine proxy distribution $p_{\boldsymbol{\phi}}(y|\boldsymbol{x})$ through Eq. (15).
8: `/*Proxy-enhanced sampling*/`
9: Begin with $\boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
10: **for** $t = T-1$ **to** 0 **do**
11: Derive the score $\tilde{\boldsymbol{s}}_\theta(\boldsymbol{x}_{t+1}, y, \omega)$ from Eq. (6).
12: Update $\boldsymbol{x}_{t+1}$ to $\boldsymbol{x}_t(\omega)$ using ω as per Eq. (7).
13: Optimize ω to $\hat{\omega}$ following Eq. (8).
14: Finalize the update of $\boldsymbol{x}_t$ with $\hat{\omega}$ via Eq. (9).
15: **end for**
16: Return $\boldsymbol{x}^* = \boldsymbol{x}_0$

Optimization of ω. Directly updating the design $\boldsymbol{x}_t$ with proxy gradient suffers from the OOD issue and determining a proper condition y necessitates the manual adjustment of multiple hyperparameters [6]. Thus, we propose to introduce proxy guidance by only optimizing the strength parameter ω within $\tilde{\boldsymbol{s}}_\theta(\boldsymbol{x}_t, y, \omega)$ in Eq. (6). As discussed in Section 2.3, the parameter ω balances the condition and diversity, and an optimized ω could achieve a better balance in the sampling process, leading to more effective generation.

Enhanced Sampling. With the score function, the update of a noisy sample $\boldsymbol{x}_{t+1}$ is computed as:

$$\boldsymbol{x}_t(\omega) = solver(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{s}}_\theta(\boldsymbol{x}_{t+1}, y, \omega)), \tag{7}$$

where the *solver* is the second-order Heun solver [12], chosen for its enhanced accuracy through a predictor-corrector method. A proxy is then trained to predict the property of noise $\boldsymbol{x}_t$ at time step t, denoted as $J_{\boldsymbol{\phi}}(\boldsymbol{x}_t, t)$. By maximizing $J_{\boldsymbol{\phi}}(\boldsymbol{x}_t(\omega), t)$ with respect to ω, we can incorporate the explicit proxy guidance into proxy-free diffusion to enable enhanced sampling control in the balance between condition and diversity. This maximization process is:

$$\hat{\omega} = \omega + \eta \frac{\partial J_{\boldsymbol{\phi}}(\boldsymbol{x}_t(\omega), t)}{\partial \omega}. \tag{8}$$

where η denotes the learning rate. We leverage the automatic differentiation capabilities of PyTorch [13] to efficiently compute the above derivatives within the context of the solver's operation. The optimized $\hat{\omega}$ then updates the noisy sample $\boldsymbol{x}_{t+1}$ through:

$$\boldsymbol{x}_t = solver(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{s}}_\theta(\boldsymbol{x}_{t+1}, y, \hat{\omega})). \tag{9}$$

This process iteratively denoises $\boldsymbol{x}_t$, utilizing it in successive steps to progressively approach $\boldsymbol{x}_0$, which represents the final high-scoring design $\boldsymbol{x}^*$.

Proxy Training. Notably, $J_{\boldsymbol{\phi}}(\boldsymbol{x}_t, t)$ can be directly derived from the proxy $J_{\boldsymbol{\phi}}(\boldsymbol{x})$, the mean of the proxy distribution $p_{\boldsymbol{\phi}}(\cdot|\boldsymbol{x})$ in Eq. (2). This distribution is trained exclusively at the initial time step $t = 0$, eliminating the need for training across time steps. To achieve this derivation, we reverse the diffusion from $\boldsymbol{x}_t$ back to $\boldsymbol{x}_0$ using the formula:

$$\boldsymbol{x}_0 = \frac{\boldsymbol{x}_t + s_{\boldsymbol{\theta}}(\boldsymbol{x}_t) \cdot \sigma(t)^2}{\mu(t)}, \tag{10}$$

where $s_{\boldsymbol{\theta}}(\boldsymbol{x}_t)$ is the estimated unconditional score at time step t, and $\sigma(t)^2$ and $\mu(t)$ are the variance and mean functions of the perturbation kernel at time t, as detailed in equations (32-33) in [10]. Consequently, we express

$$J_{\boldsymbol{\phi}}(\boldsymbol{x}_t, t) = J_{\boldsymbol{\phi}}\left(\frac{\boldsymbol{x}_t + s_{\boldsymbol{\theta}}(\boldsymbol{x}_t) \cdot \sigma(t)^2}{\mu(t)}\right). \tag{11}$$

This formulation allows for the optimization of the strength parameter ω via Eq. (8). For simplicity, we will refer to $J_{\boldsymbol{\phi}}(\cdot)$ in subsequent discussions.

3.2 Diffusion-based Proxy Refinement

In the *proxy-enhanced sampling* module, the proxy $J_{\boldsymbol{\phi}}(\cdot)$ is employed to update the parameter ω to enable enhanced control. However, $J_{\boldsymbol{\phi}}(\cdot)$ may still be prone to the OOD issue, especially on adversarial samples [5]. To address this, we refine the proxy by using insights from proxy-free diffusion. The procedure of this module is specified in Algorithm 1, Lines 3-7.

Diffusion Distribution. Adversarial samples are identified by gradient ascent on the proxy as per Eq. (3) to form the distribution $q(\boldsymbol{x})$. Consequently, these samples are vulnerable to the proxy distribution. Conversely, the proxy-free diffusion, which functions without depending on a proxy, inherently offers greater resilience against these samples, thus producing a more robust distribution. For an adversarial sample $\hat{\boldsymbol{x}} \sim q(\boldsymbol{x})$, we compute $p_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}})$, $p_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}}|y)$ via the probability flow ODE, and $p(y)$ through Gaussian kernel-density estimation. The diffusion distribution regarding y is derived as:

$$p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}}) = \frac{p_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}}|y) \cdot p(y)}{p_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}})}, \tag{12}$$

which demonstrates inherent robustness over the proxy distribution $p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})$. Yet, directly applying diffusion distribution to design optimization by gradient ascent is computationally intensive and potentially unstable due to the demands of reversing ODEs and scoring steps.

Proxy Refinement. We opt for a more feasible approach: refine the proxy distribution $p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}}) = \mathcal{N}(J_{\boldsymbol{\phi}}(\hat{\boldsymbol{x}}), \sigma_{\boldsymbol{\phi}}(\hat{\boldsymbol{x}}))$ by minimizing its distance to the diffusion distribution $p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})$. The distance is quantified by the Kullback-Leibler (KL) divergence:

$$\mathbb{E}_q[\mathcal{D}(p_{\boldsymbol{\phi}}||p_{\boldsymbol{\theta}})] = \mathbb{E}_{q(\boldsymbol{x})} \int p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}}) \log\left(\frac{p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}{p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})}\right) dy. \tag{13}$$

We avoid the parameterization trick for minimizing this divergence as it necessitates backpropagation through $p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})$, which is prohibitively expensive. Instead, for the sample $\hat{\boldsymbol{x}}$, the gradient of the KL divergence $\mathcal{D}(p_{\boldsymbol{\phi}}||p_{\boldsymbol{\theta}})$ with respect to the proxy parameters $\boldsymbol{\phi}$ is computed as:

$$\mathbb{E}_{p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}\left[\frac{d \log p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}{d\boldsymbol{\phi}}\left(1 + \log \frac{p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}{p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})}\right)\right]. \tag{14}$$

Complete derivations are in Appendix A. The KL divergence then acts as regularization in our loss $\mathcal{L}$:

$$\mathcal{L}(\boldsymbol{\phi}, \alpha) = \mathbb{E}_{\mathcal{D}}[-\log p_{\boldsymbol{\phi}}(y|\boldsymbol{x})] + \alpha \mathbb{E}_{q(\boldsymbol{x})}[\mathcal{D}(p_{\boldsymbol{\phi}}||p_{\boldsymbol{\theta}})], \tag{15}$$

where $\mathcal{D}$ is the training dataset and α is a hyperparameter. We propose to optimize α based on the validation loss via bi-level optimization as detailed in Appendix B.

4 Experiments

In this section, we conduct comprehensive experiments to evaluate our method's performance.

4.1 Benchmarks

Tasks. Our experiments encompass a variety of tasks, split into continuous and discrete categories.

The continuous category includes four tasks: **(1)** Superconductor (SuperC) [4]: The objective here is to engineer a superconductor composed of 86 continuous elements. The goal is to enhance the critical temperature using $17,010$ design samples. This task is based on the dataset from [1]. **(2)** Ant Morphology (Ant): In this task, the focus is on developing a quadrupedal ant robot, comprising 60 continuous parts, to augment its crawling velocity. It uses $10,004$ design instances from the dataset in [3, 14]. **(3)** D'Kitty Morphology (D'Kitty): Similar to Ant Morphology, this task involves the design of a quadrupedal D'Kitty robot with 56 components, aiming to improve its crawling speed with $10,004$ designs, as described in [3, 15]. **(4)** Rosenbrock (Rosen): The aim of this task is to optimize a 60-dimension continuous vector to maximize the Rosenbrock black-box function. It uses 50000 designs from the low-scoring part [9].

For the discrete category, we explore three tasks: **(1)** TF Bind 8 (TF8): The goal is to identify an 8-unit DNA sequence that maximizes binding activity. This task uses $32,898$ designs and is detailed in [16]. **(2)** TF Bind 10 (TF10): Similar to TF8, but with a 10-unit DNA sequence and a larger pool of $50,000$ samples, as described in [16]. **(3)** Neural Architecture Search (NAS): This task focuses on discovering the optimal neural network architecture to improve test accuracy on the CIFAR-10 dataset, using $1,771$ designs [17].

Evaluation. In this study, we utilize the oracle evaluation from design-bench [3]. Adhering to this established protocol, we analyze the top 128 promising designs from each method. The evaluation metric employed is the 100^{th} percentile normalized ground-truth score, calculated using the formula $y_n = \frac{y - y_{\min}}{y_{\max} - y_{\min}}$, where $y_{\min}$ and $y_{\max}$ signify the lowest and highest scores respectively in the comprehensive, yet unobserved, dataset. In addition to these scores, we provide an overview of each method's effectiveness through the mean and median rankings across all evaluated tasks. Notably, the best design discovered in the offline dataset, designated as $\mathcal{D}(\textbf{best})$, is also included for reference. For further details on the 50^{th} percentile (median) scores, please refer to Appendix C.

4.2 Comparison Methods

Our approach is evaluated against two primary groups of baseline methods: forward and inverse approaches. Forward approaches enhance existing designs through gradient ascent. This includes: **(i)** Grad: utilizes simple gradient ascent on current designs for new creations; **(ii)** ROMA [18]: implements smoothness regularization on proxies; **(iii)** COMs [5]: applies regularization to assign lower scores to adversarial designs; **(iv)** NEMO [19]: bridges the gap between proxy and actual functions using normalized maximum likelihood; **(v)** BDI [20]: utilizes both forward and inverse mappings to transfer knowledge from offline datasets to the designs; **(vi)** IOM [21]: ensures consistency between representations of training datasets and optimized designs.

Inverse approaches focus on learning a mapping from a design's property value back to its input. High property values are input into this inverse mapping to yield enhanced designs. This includes: **(i)** CbAS [22]: CbAS employs a VAE model to implicitly implement the inverse mapping. It gradually tunes its distribution toward higher scores by raising the scoring threshold. This process can be interpreted as incrementally increasing the conditional score within the inverse mapping framework. **(ii)** Autofocused CbAS (Auto.CbAS) [23]: adopts importance sampling for retraining a regression model based on CbAS. **(iii)** MIN [6]: maps scores to designs via a GAN model and explore this mapping for optimal designs. **(iv)** BONET [24]: introduces an autoregressive model for sampling high-scoring designs. **(v)** DDOM [4]: utilizes proxy-free diffusion to model the inverse mapping.

Traditional methods as detailed in [3] are also considered: **(i)** CMA-ES [25]: modifies the covariance matrix to progressively shift the distribution towards optimal designs; **(ii)** BO-qEI [26]: implements Bayesian optimization to maximize the proxy and utilizes the quasi-Expected-Improvement acquisition function for design suggestion, labeling designs using the proxy; **(iii)** REINFORCE [27]: enhances the input space distribution using the learned proxy model.

[4]Previously, the task oracle exhibited inconsistencies, producing varying outputs for identical inputs. This issue has now been rectified by the development team.

4.3 Experimental Configuration

In alignment with the experimental protocols established in [3, 20], we have tailored our training methodologies for all approaches, except where specified otherwise. For methods such as BO-qEI, CMA-ES, REINFORCE, CbAS, and Auto.CbAS that do not utilize gradient ascent, we base our approach on the findings reported in [3]. We adopted $T = 1000$ diffusion sampling steps, set the condition y to y_{max}, and initial strength ω as 2 in line with [4]. To ensure reliability and consistency in our comparative analysis, each experimental setting was replicated across 8 independent runs, unless stated otherwise, with the presentation of both mean values and standard errors. These experiments were conducted using a NVIDIA GeForce V100 GPU. We've detailed the computational overhead of our approach in Appendix D to provide a comprehensive view of its practicality.

Table 1: Results (maximum normalized score) on continuous tasks.

Method	Superconductor	Ant Morphology	D'Kitty Morphology	Rosenbrock
$\mathcal{D}$(**best**)	0.399	0.565	0.884	0.518
BO-qEI	0.402 ± 0.034	0.819 ± 0.000	0.896 ± 0.000	0.772 ± 0.012
CMA-ES	0.465 ± 0.024	**1.214 ± 0.732**	0.724 ± 0.001	0.470 ± 0.026
REINFORCE	0.481 ± 0.013	0.266 ± 0.032	0.562 ± 0.196	0.558 ± 0.013
Grad	0.490 ± 0.009	0.932 ± 0.015	0.930 ± 0.002	0.701 ± 0.092
COMs	**0.504 ± 0.022**	0.818 ± 0.017	0.905 ± 0.017	0.672 ± 0.075
ROMA	**0.507 ± 0.013**	0.898 ± 0.029	0.928 ± 0.007	0.663 ± 0.072
NEMO	0.499 ± 0.003	0.956 ± 0.013	**0.953 ± 0.010**	0.614 ± 0.000
IOM	**0.524 ± 0.022**	0.929 ± 0.037	0.936 ± 0.008	0.712 ± 0.068
BDI	**0.513 ± 0.000**	0.906 ± 0.000	0.919 ± 0.000	0.630 ± 0.000
CbAS	**0.503 ± 0.069**	0.876 ± 0.031	0.892 ± 0.008	0.702 ± 0.008
Auto.CbAS	0.421 ± 0.045	0.882 ± 0.045	0.906 ± 0.006	0.721 ± 0.007
MIN	0.499 ± 0.017	0.445 ± 0.080	0.892 ± 0.011	0.702 ± 0.074
BONET	0.422 ± 0.019	0.925 ± 0.010	0.941 ± 0.001	0.780 ± 0.009
DDOM	0.495 ± 0.012	0.940 ± 0.004	0.935 ± 0.001	**0.789 ± 0.003**
RGD	**0.515 ± 0.011**	**0.968 ± 0.006**	**0.943 ± 0.004**	**0.797 ± 0.011**

Table 2: Results (maximum normalized score) on discrete tasks & ranking on all tasks.

Method	TF Bind 8	TF Bind 10	NAS	Rank Mean	Rank Median
$\mathcal{D}$(**best**)	0.439	0.467	0.436		
BO-qEI	0.798 ± 0.083	0.652 ± 0.038	**1.079 ± 0.059**	$9.1/15$	$11/15$
CMA-ES	0.953 ± 0.022	0.670 ± 0.023	0.985 ± 0.079	$7.3/15$	$4/15$
REINFORCE	0.948 ± 0.028	0.663 ± 0.034	-1.895 ± 0.000	$11.3/15$	$14/15$
Grad	0.872 ± 0.062	0.646 ± 0.052	0.624 ± 0.102	$9.0/15$	$10/15$
COMs	0.517 ± 0.115	0.613 ± 0.003	0.783 ± 0.029	$10.3/15$	$10/15$
ROMA	0.927 ± 0.033	0.676 ± 0.029	0.927 ± 0.071	$6.1/15$	$6/15$
NEMO	0.942 ± 0.003	**0.708 ± 0.022**	0.737 ± 0.010	$5.3/15$	$5/15$
IOM	0.823 ± 0.130	0.650 ± 0.042	0.559 ± 0.081	$7.4/15$	$6/15$
BDI	0.870 ± 0.000	0.605 ± 0.000	0.722 ± 0.000	$9.6/15$	$9/15$
CbAS	0.927 ± 0.051	0.651 ± 0.060	0.683 ± 0.079	$8.7/15$	$8/15$
Auto.CbAS	0.910 ± 0.044	0.630 ± 0.045	0.506 ± 0.074	$10.3/15$	$10/15$
MIN	0.905 ± 0.052	0.616 ± 0.021	0.717 ± 0.046	$10.4/15$	$10/15$
BONET	0.913 ± 0.008	0.621 ± 0.030	0.724 ± 0.008	$7.7/15$	$8/15$
DDOM	0.957 ± 0.006	0.657 ± 0.006	0.745 ± 0.070	$4.9/15$	$5/15$
RGD	**0.974 ± 0.003**	**0.694 ± 0.018**	0.825 ± 0.063	**2.0/15**	**2/15**

4.4 Results and Analysis

In Tables 1 and 2, we showcase our experimental results for both continuous and discrete tasks. To clearly differentiate among the various approaches, distinct lines separate traditional, forward, and inverse approaches within the tables For every task, algorithms performing within a standard deviation of the highest score are emphasized by **bolding** following [5].

We make the following observations. (1) As highlighted in Table 2, RGD not only achieves the top rank but also demonstrates the best performance in six out of seven tasks, emphasizing the robustness and superiority of our method. (2) RGD outperforms the VAE-based CbAS, the GAN-based MIN and the Transformer-based BONET. This result highlights the superiority of diffusion models in modeling inverse mappings compared to other generative approaches. (3) Upon examining TF Bind 8, we observe that the average rankings for forward and inverse methods stand at 10.3 and 6.0, respectively. In contrast, for TF Bind 10, both methods have the same average ranking of 8.7, indicating no advantage. This notable advantage of inverse methods in TF Bind 8 implies that the relatively smaller design space of TF Bind 8 (4^8) facilitates easier inverse mapping, as opposed to the more complex space in TF Bind 10 (4^{10}). (4) RGD's performance is less impressive on NAS, where designs are encoded as 64-length sequences of 5-category one-hot vectors. This may stem from the design-bench's encoding not fully capturing the sequential and hierarchical aspects of network architectures, affecting the efficacy of inverse mapping modeling.

Table 3: Ablation studies on RGD.

Task	D	RGD	w/o proxy-e	w/o diffusion-b r	direct grad update
SuperC	86	$\mathbf{0.515 \pm 0.011}$	0.495 ± 0.012	$\underline{0.502 \pm 0.005}$	0.456 ± 0.002
Ant	60	$\mathbf{0.968 \pm 0.006}$	0.940 ± 0.004	$\underline{0.961 \pm 0.011}$	-0.006 ± 0.003
D'Kitty	56	$\mathbf{0.943 \pm 0.004}$	0.935 ± 0.001	$\underline{0.939 \pm 0.003}$	0.714 ± 0.001
Rosen	60	$\underline{0.797 \pm 0.011}$	0.789 ± 0.003	$\mathbf{0.813 \pm 0.005}$	0.241 ± 0.283
TF8	8	$\mathbf{0.974 \pm 0.003}$	0.957 ± 0.007	$\underline{0.960 \pm 0.006}$	0.905 ± 0.000
TF10	10	$\mathbf{0.694 \pm 0.018}$	0.657 ± 0.006	0.667 ± 0.009	$\underline{0.672 \pm 0.018}$
NAS	64	$\mathbf{0.825 \pm 0.063}$	$\underline{0.745 \pm 0.070}$	0.717 ± 0.032	0.718 ± 0.032

4.5 Ablation Studies

In this section, we present a series of ablation studies to scrutinize the individual contributions of distinct components in our methodology. We employ our proposed approach as a benchmark and methodically exclude key modules, such as the *proxy-enhanced sampling* and *diffusion-based proxy refinement*, to assess their influence on performance. These variants are denoted as *w/o proxy-e* and *w/o diffusion-b r*. Additionally, we explore the strategy of directly performing gradient ascent on the diffusion intermediate state, referred to as *direct grad update*. The results from these ablation experiments are detailed in Table 3.

Our analysis reveals that omitting either module results in a decrease in performance, thereby affirming the importance of each component. The *w/o diffusion-b r* variant generally surpasses *w/o proxy-e*, highlighting the utility of the proxy-enhanced sampling even with a basic proxy setup. Conversely, *direct grad update* tends to produce subpar results across tasks, likely attributable to the proxy's limitations in handling out-of-distribution samples, leading to suboptimal design optimizations.

To further dive into the proxy-enhanced sampling module, we visualize the strength ratio ω/ω_0—where ω_0 represents the initial strength—across diffusion steps t. This analysis is depicted in Figure 3 for two specific tasks: Ant and TF10. We observe a pattern of initial decrease followed by an increase in ω across both tasks. This pattern can be interpreted as follows: The decrease in ω facilitates the generation of a more diverse set of samples, enhancing exploratory capabilities. Subsequently, the increase in ω signifies a shift towards integrating high-performance features into the sample generation. Within this context, conditioning on the maximum y is not aimed at achieving the dataset's maximum but at enriching samples with high-scoring attributes. Overall, this adjustment of ω effectively balances between generating novel solutions and honing in on high-quality ones.

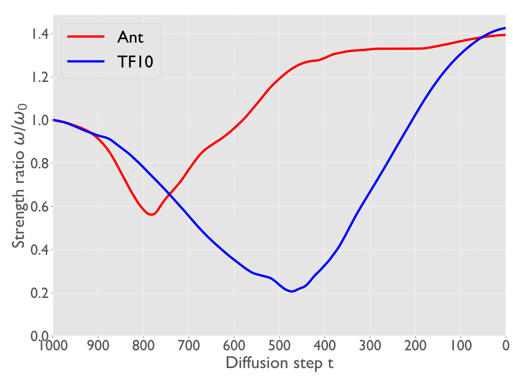

Figure 3: Dynamics of strength ratio ω/ω_0.

In addition, we visualize the proxy distribution alongside the diffusion distribution for a sample $\hat{\boldsymbol{x}}$ from the Ant task in Figure 4, to substantiate the efficacy of diffusion-based proxy refinement. The proxy distribution significantly overestimates the ground truth, whereas the diffusion distribution closely aligns with it, demonstrating the robustness of diffusion distribution. For a more quantitative analysis, we compute the expectation of both distributions and compare them with the ground truth. The mean of the diffusion distribution is calculated as $\mathbb{E}_{p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})}[y] = \mathbb{E}_{p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}\left[\frac{p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})}{p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}y\right]$.

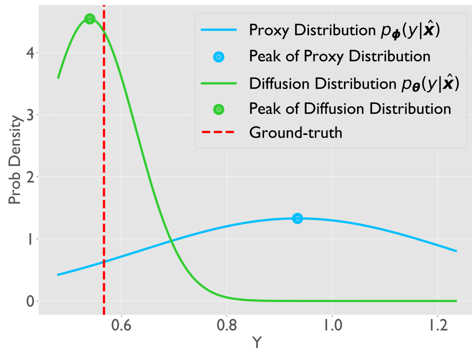

Figure 4: Proxy vs. diffusion distribution.

The MSE loss for the proxy distribution is 2.88, while for the diffusion distribution, it is 0.13 on the Ant task. Additionally, we evaluate this on the TFB10 task, where the MSE loss for the proxy distribution is 323.63 compared to 0.82 for the diffusion distribution. These results further corroborate the effectiveness of our proposed module.

Furthermore, we (1) investigate the impact of replacing our trained proxy model with alternative approaches, specifically ROMA and COMs, (2) analyze the performance with an optimized condition y and (3) explore a simple annealing approach of ω. For a comprehensive discussion on these, readers are referred to Appendix E.

4.6 Hyperparameter Sensitivity Analysis

This section investigates the sensitivity of *RGD* to various hyperparameters. Specifically, we analyze the effects of (1) the number of diffusion sampling steps T, (2) the condition y, and (3) the learning rate η of the proxy-enhanced sampling. These parameters are evaluated on two tasks: the continuous Ant task and the discrete TFB10 task. For a detailed discussion, see Appendix F.

5 Related Work

Offline black-box optimization. A recent surge in research has presented two predominant approaches for offline BBO. The forward approach deploys a DNN to fit the offline dataset, subsequently utilizing gradient ascent to enhance existing designs. Typically, these techniques, including COMs [5], ROMA [18], NEMO [19], BDI [20, 28], IOM [29] and Parallel-mentoring [30], are designed to embed prior knowledge within the surrogate model to alleviate the OOD issue. The reverse approach [6, 31] is dedicated to learning a mapping from property values back to inputs. Feeding a high value into this inverse mapping directly produces a design of elevated performance. Additionally, methods in [22, 23] progressively tailor a generative model towards the optimized design via a proxy function and BONET [24] introduces an autoregressive model trained on fixed-length trajectories to sample high-scoring designs. Recent investigations [4] have underscored the superiority of diffusion models in delineating the inverse mapping. However, research on specialized guided diffusion for offline BBO remains limited. This paper addresses this research gap.

Guided diffusion. Guided diffusion seeks to produce samples with specific desirable attributes. Contemporary research in guided diffusion primarily concentrates on enhancing the efficiency of its sampling process. [32] propose a method for distilling a classifier-free guided diffusion model into a more efficient single model that necessitates fewer steps in sampling. [33] introduce an operator splitting method to expedite classifier guidance by separating the update process into two key functions: the diffusion function and the conditioning function. Additionally, [34] presents an efficient and universal guidance mechanism that utilizes a readily available proxy to enable diffusion guidance across time steps. In this work, we explore the application of guided diffusion in offline BBO, with the goal of creating tailored algorithms to efficiently generate high-performance designs.

6 Conclusion

In conclusion, we propose ***R**obust **G**uided **D**iffusion for Offline Black-box Optimization* (**RGD**). The *proxy-enhanced sampling* module adeptly integrates proxy guidance to enable enhanced sampling control, while the *diffusion-based proxy refinement* module leverages proxy-free diffusion insights for proxy improvement. Empirical evaluations on design-bench have showcased RGD's outstanding performance, further validated by ablation studies on the contributions of these novel components. We discuss the broader impact and limitation in Appendix G.

References

[1] Kam Hamidieh. A data-driven statistical model for predicting the critical temperature of a superconductor. *Computational materials science*, 2018.

[2] Karen S Sarkisyan et al. Local fitness landscape of the green fluorescent protein. *Nature*, 2016.

[3] Brandon Trabucco, Xinyang Geng, Aviral Kumar, and Sergey Levine. Design-Bench: benchmarks for data-driven offline model-based optimization. *arXiv preprint arXiv:2202.08450*, 2022.

[4] Siddarth Krishnamoorthy, Satvik Mehul Mashkaria, and Aditya Grover. Diffusion models for black-box optimization. *Proc. Int. Conf. Machine Learning (ICML)*, 2023.

[5] Brandon Trabucco, Aviral Kumar, Xinyang Geng, and Sergey Levine. Conservative objective models for effective offline model-based optimization. In *Proc. Int. Conf. Machine Learning (ICML)*, 2021.

[6] Aviral Kumar and Sergey Levine. Model inversion networks for model-based optimization. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2020.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2014.

[8] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[9] HoHo Rosenbrock. An automatic method for finding the greatest or least value of a function. *The computer journal*, 1960.

[10] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *Proc. Int. Conf. Learning Rep. (ICLR)*, 2021.

[11] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2021.

[12] Endre Süli and David F Mayers. *An introduction to numerical analysis*. Cambridge university press, 2003.

[13] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: an imperative style, high-performance deep learning library. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2019.

[14] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[15] Michael Ahn, Henry Zhu, Kristian Hartikainen, Hugo Ponte, Abhishek Gupta, Sergey Levine, and Vikash Kumar. Robel: robotics benchmarks for learning with low-cost robots. In *Conf. on Robot Lea. (CoRL)*, 2020.

[16] Luis A Barrera et al. Survey of variation in human transcription factors reveals prevalent DNA binding changes. *Science*, 2016.

[17] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2017.

[18] Sihyun Yu, Sungsoo Ahn, Le Song, and Jinwoo Shin. Roma: robust model adaptation for offline model-based optimization. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2021.

[19] Justin Fu and Sergey Levine. Offline model-based optimization via normalized maximum likelihood estimation. *Proc. Int. Conf. Learning Rep. (ICLR)*, 2021.

[20] Can Chen, Yingxue Zhang, Jie Fu, Xue Liu, and Mark Coates. Bidirectional learning for offline infinite-width model-based optimization. In *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2022.

[21] Han Qi, Yi Su, Aviral Kumar, and Sergey Levine. Data-driven model-based optimization via invariant representation learning. In *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2022.

[22] David Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. In *Proc. Int. Conf. Machine Learning (ICML)*, 2019.

[23] Clara Fannjiang and Jennifer Listgarten. Autofocused oracles for model-based design. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2020.

[24] Satvik Mehul Mashkaria, Siddarth Krishnamoorthy, and Aditya Grover. Generative pretraining for black-box optimization. In *Proc. Int. Conf. Machine Learning (ICML)*, 2023.

[25] Nikolaus Hansen. The CMA evolution strategy: a comparing review. *Towards A New Evolutionary Computation*, 2006.

[26] James T Wilson, Riccardo Moriconi, Frank Hutter, and Marc Peter Deisenroth. The reparameterization trick for acquisition functions. *arXiv preprint arXiv:1712.00424*, 2017.

[27] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 1992.

[28] Can Chen, Yingxue Zhang, Xue Liu, and Mark Coates. Bidirectional learning for offline model-based biological sequence design. In *Proc. Int. Conf. Machine Lea. (ICML)*, 2023.

[29] Han Qi, Yi Su, Aviral Kumar, and Sergey Levine. Data-driven model-based optimization via invariant representation learning. In *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2022.

[30] Can Chen, Christopher Beckham, Zixuan Liu, Xue Liu, and Christopher Pal. Parallel-mentoring for offline model-based optimization. In *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2023.

[31] Alvin Chan, Ali Madani, Ben Krause, and Nikhil Naik. Deep extrapolation for attribute-enhanced generation. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2021.

[32] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proc. Comp. Vision. Pattern. Rec.(CVPR)*, 2023.

[33] Suttisak Wizadwongsa and Supasorn Suwajanakorn. Accelerating guided diffusion sampling with splitting numerical methods. In *Proc. Int. Conf. Learning Rep. (ICLR)*, 2023.

[34] Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *Proc. Comp. Vision. Pattern. Rec.(CVPR)*, 2023.

[35] Can Chen, Yingxue Zhang, Jie Fu, Xue Liu, and Mark Coates. Bidirectional learning for offline infinite-width model-based optimization. *Proc. Adv. Neur. Inf. Proc. Syst (NeurIPS)*, 2022.

A Derivation

This section provides a derivation of the gradient of the KL divergence. Let's consider the KL divergence term, defined as:

$$\mathcal{D}(p_{\boldsymbol{\phi}}||p_{\boldsymbol{\theta}}) = \int p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}}) \log\left(\frac{p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}{p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})}\right) dy. \tag{16}$$

The gradient with respect to the parameters $\boldsymbol{\phi}$ is computed as follows:

$$\begin{aligned}\frac{d\mathcal{D}(p_{\boldsymbol{\phi}}||p_{\boldsymbol{\theta}})}{d\boldsymbol{\phi}} &= \int \frac{dp_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}{d\boldsymbol{\phi}}\left(1+\log\frac{p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}{p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})}\right) dy \\ &= \int p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})\frac{d\log p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}{d\boldsymbol{\phi}}(1+\log\frac{p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}{p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})})\, dy \\ &= \mathbb{E}_{p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}\left[\frac{d\log p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}{d\boldsymbol{\phi}}\left(1+\log\frac{p_{\boldsymbol{\phi}}(y|\hat{\boldsymbol{x}})}{p_{\boldsymbol{\theta}}(y|\hat{\boldsymbol{x}})}\right)\right].\end{aligned} \tag{17}$$

B Hyperparameter Optimization

We propose adjusting α based on the validation loss, establishing a bi-level optimization framework:

$$\alpha^* = \arg\min_{\alpha} \mathbb{E}_{\mathcal{D}_v}[\log p_{\boldsymbol{\phi}^*(\alpha)}(y_v|\boldsymbol{x}_v)], \tag{18}$$

$$\text{s.t.} \quad \boldsymbol{\phi}^*(\alpha) = \arg\min_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}, \alpha). \tag{19}$$

Within this context, $\mathcal{D}_v$ represents the validation dataset sampled from the offline dataset. The inner optimization task, which seeks the optimal $\boldsymbol{\phi}^*(\alpha)$, is efficiently approximated via gradient descent.

C Evaluation of Median Scores

While the main text of our paper focuses on the 100^{th} percentile scores, this section provides an in-depth analysis of the 50^{th} percentile scores. These median scores, previously explored in [3], serve as an additional metric to assess the performance of our *RGD* method. The outcomes for continuous tasks are detailed in Table 5, and those pertaining to discrete tasks, along with their respective ranking statistics, are outlined in Table 6. An examination of Table 6 highlights the notable success of the *RGD* approach, as it achieves the top rank in this evaluation. This finding underscores the method's robustness and effectiveness.

D Computational Overhead

Table 4: Computational Overhead (in seconds).

Process	SuperC	Ant	D'Kitty	NAS
Proxy training	40.8	74.5	24.7	7.8
Diffusion training	405.9	767.9	251.1	56.0
Proxy-e sampling	30.0	29.7	29.6	31.5
Diffusion-b proxy r	3104.6	4036.7	2082.8	3096.2
Overall cost	3581.3	4908.8	2388.2	3191.5

In this section, we analyze the computational overhead of our method. RGD consists of two core components: proxy-enhanced sampling (*proxy-e sampling*) and diffusion-based proxy refinement (*diffusion-b proxy r*). Additionally, RGD employs a trained proxy and a proxy-free diffusion model, whose computational demands are denoted as *proxy training* and *diffusion training*, respectively.

Table 4 indicates that experiments can be completed within approximately one hour, demonstrating efficiency. The *diffusion-based proxy refinement* module is the primary contributor to the computational overhead, primarily due to the usage of a probability flow ODE for sample likelihood computation.

However, as this is a one-time process for refining the proxy, its high computational cost is offset by its non-recurring nature. In contexts such as robotics or bio-chemical research, the most time-intensive part of the production cycle is usually the evaluation of the unknown objective function. Therefore, the time differences between methods for deriving high-performance designs are less critical in actual production environments, highlighting RGD's practicality where optimization performance are prioritized over computational speed. This aligns with recent literature (A.3 Computational Complexity in [35] and A.7.5. Computational Cost in [28]) indicating that in black-box optimization scenarios, computational time is relatively minor compared to the time and resources dedicated to experimental validation phases.

Table 5: Results (median normalized score) on continuous tasks.

Method	Superconductor	Ant Morphology	D'Kitty Morphology	Rosenbrock
BO-qEI	0.300 ± 0.015	0.567 ± 0.000	$\mathbf{0.883 \pm 0.000}$	$\mathbf{0.761 \pm 0.004}$
CMA-ES	0.379 ± 0.003	-0.045 ± 0.004	0.684 ± 0.016	0.200 ± 0.000
REINFORCE	$\mathbf{0.463 \pm 0.016}$	0.138 ± 0.032	0.356 ± 0.131	0.553 ± 0.008
Grad	0.339 ± 0.013	0.532 ± 0.014	0.867 ± 0.006	0.540 ± 0.025
COMs	0.312 ± 0.018	0.568 ± 0.002	$\mathbf{0.883 \pm 0.000}$	0.419 ± 0.286
ROMA	0.364 ± 0.020	0.467 ± 0.031	0.850 ± 0.006	-0.121 ± 0.242
NEMO	0.319 ± 0.010	0.592 ± 0.001	$\mathbf{0.882 \pm 0.002}$	0.510 ± 0.000
IOM	0.343 ± 0.018	0.513 ± 0.024	0.873 ± 0.009	0.126 ± 0.443
BDI	0.412 ± 0.000	0.474 ± 0.000	0.855 ± 0.000	0.561 ± 0.000
CbAS	0.111 ± 0.017	0.384 ± 0.016	0.753 ± 0.008	0.676 ± 0.008
Auto.CbAS	0.131 ± 0.010	0.364 ± 0.014	0.736 ± 0.025	0.695 ± 0.008
MIN	0.336 ± 0.016	0.618 ± 0.040	$\mathbf{0.887 \pm 0.004}$	0.634 ± 0.082
BONET	0.319 ± 0.014	0.615 ± 0.004	$\mathbf{0.895 \pm 0.021}$	0.630 ± 0.009
DDOM	0.295 ± 0.001	0.590 ± 0.003	0.870 ± 0.001	0.640 ± 0.001
RGD	0.308 ± 0.003	$\mathbf{0.684 \pm 0.006}$	$\mathbf{0.874 \pm 0.001}$	0.644 ± 0.002

Table 6: Results (median normalized score) on discrete tasks & ranking on all tasks.

Method	TF Bind 8	TF Bind 10	NAS	Rank Mean	Rank Median
BO-qEI	0.439 ± 0.000	0.467 ± 0.000	$\mathbf{0.544 \pm 0.099}$	$6.4/15$	$7/15$
CMA-ES	0.537 ± 0.014	0.484 ± 0.014	$\mathbf{0.591 \pm 0.102}$	$8.0/15$	$5/15$
REINFORCE	0.462 ± 0.021	0.475 ± 0.008	-1.895 ± 0.000	$9.7/15$	$9/15$
Grad	0.546 ± 0.022	0.526 ± 0.029	0.443 ± 0.126	$6.6/15$	$8/15$
COMs	0.439 ± 0.000	0.467 ± 0.000	$\mathbf{0.529 \pm 0.003}$	$7.7/15$	$8/15$
ROMA	0.543 ± 0.017	0.518 ± 0.024	$\mathbf{0.529 \pm 0.008}$	$7.6/15$	$5/15$
NEMO	0.436 ± 0.016	0.453 ± 0.013	$\mathbf{0.563 \pm 0.020}$	$8.3/15$	$8/15$
IOM	0.439 ± 0.000	0.474 ± 0.014	-0.083 ± 0.012	$9.3/15$	$8/15$
BDI	0.439 ± 0.000	0.476 ± 0.000	$\mathbf{0.517 \pm 0.000}$	$7.3/15$	$8/15$
CbAS	0.428 ± 0.010	0.463 ± 0.007	0.292 ± 0.027	$11.3/15$	$12/15$
Auto.CbAS	0.419 ± 0.007	0.461 ± 0.007	0.217 ± 0.005	$11.9/15$	$13/15$
MIN	0.421 ± 0.015	0.468 ± 0.006	0.433 ± 0.000	$7.0/15$	$7/15$
BONET	0.507 ± 0.007	0.460 ± 0.013	$\mathbf{0.571 \pm 0.095}$	$5.9/15$	$6/15$
DDOM	0.553 ± 0.002	0.488 ± 0.001	0.367 ± 0.021	$6.9/15$	$5/15$
RGD	$\mathbf{0.557 \pm 0.002}$	$\mathbf{0.545 \pm 0.006}$	0.371 ± 0.019	**4.9/15**	**4/15**

E Further Ablation Studies

In this section, we extend our exploration to include alternative proxy refinement schemes, namely ROMA and COMs, to compare against our diffusion-based proxy refinement module. The objective is to assess the relative effectiveness of these schemes in the context of the Ant and TFB10 tasks. The comparative results are presented in Table 7. Our investigation reveals that proxies refined through ROMA and COMs exhibit performance akin to the vanilla proxy and they fall short of achieving the enhancements seen with our diffusion-based proxy refinement. We hypothesize that the diffusion-based proxy refinement, by aligning closely with the characteristics of the diffusion model, provides a more relevant and impactful signal. This alignment improves the proxy's ability to enhance the sampling process more effectively.

Table 7: Comparative Results of Proxy Integration with COMs, ROMA, and ours.

Method	Ant Morphology	TF Bind 10
No proxy	0.940 ± 0.004	0.657 ± 0.006
Vanilla proxy	0.961 ± 0.011	0.667 ± 0.009
COMs	0.963 ± 0.004	0.668 ± 0.003
ROMA	0.953 ± 0.003	0.667 ± 0.003
Ours	0.968 ± 0.006	0.694 ± 0.018

Additionally, we contrast our approach, which adjusts the strength parameter ω, with the MIN method that focuses on identifying an optimal condition y. The MIN strategy entails optimizing a Lagrangian objective with respect to y, a process that requires manual tuning of four hyperparameters. We adopt their methodology to determine optimal conditions y and incorporate these into the proxy-free diffusion for tasks Ant and TF10. The normalized scores for Ant and TF10 are 0.950 ± 0.017 and 0.660 ± 0.027, respectively. The outcomes fall short of those achieved by our method as detailed in Table 7. This discrepancy likely stems from the complexity involved in optimizing y, whereas dynamically adjusting ω proves to be a more efficient strategy for enhancing sampling control.

Last but not least, we explore simple annealing approaches for ω. Specifically, we test two annealing scenarios considering the default ω as 2.0: (1) a decrease from 4.0 to 0.0, and (2) an increase from 0.0 to 4.0, both modulated by a cosine function over the time step (t). We apply these strategies to the Ant Morphology and TF Bind 10 tasks, and the results are as follows:

Table 8: Results of Annealing Approaches.

Method	Ant Morphology	TF Bind 10
RGD	0.968	0.694
$\omega = 2.0$	0.940	0.657
Increase	0.948	0.654
Decrease	0.924	0.647

The empirical results across both strategies illustrate their inferior performance compared to our approach, thereby demonstrating the efficacy of our proposed method.

F Hyperparameter Sensitivity Analysis

RGD's performance is assessed under different settings of T, y, and η. We experiment with T values of 500, 750, 1000, 1250, and 1500, with the default being $T = 1000$. For the condition ratio y/y_{max}, we test values of 0.5, 1.0, 1.5, 2.0, and 2.5, considering 1.0 as the default. Similarly, for the learning rate η, we explore values of $2.5e^{-3}$, $5.0e^{-3}$, 0.01, 0.02, and 0.04, with the default set to $\eta = 0.01$. Results are normalized by comparing them with the performance obtained at default values.

As depicted in Figures 5, 6, and 7, *RGD* demonstrates considerable resilience to hyperparameter variations. The Ant task, in particular, exhibits a more marked sensitivity, with a gradual enhancement in performance as these hyperparameters are varied. The underlying reasons for this trend include: (1) An increase in the number of diffusion steps (T) enhances the overall quality of the generated samples. This improvement, in conjunction with more effective guidance from the trained proxy, leads to better results. (2) Elevating the condition (y) enables the diffusion model to extend its reach beyond the existing dataset, paving the way for superior design solutions. However, selecting an optimal y can be challenging and may, as observed in the TFB10 task, sometimes lead to suboptimal results. (3) A higher learning rate (η) integrates an enhanced guidance signal from the trained proxy, contributing to improved performances.

In contrast, the discrete nature of the TFB10 task seems to endow it with a certain robustness to variations in these hyperparameters, highlighting a distinct behavioral pattern in response to hyperparameter adjustments.

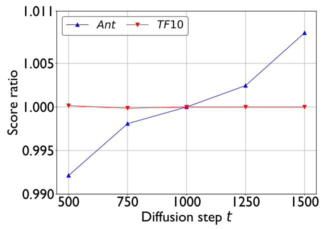

Figure 5: **The ratio of** the performance of our *RGD* method with T **to** the performance with $T = 1000$.

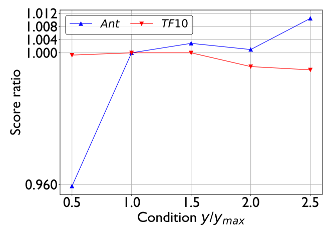

Figure 6: **The ratio of** the performance of our *RGD* method with y/y_{max} **to** the performance with 1.0.

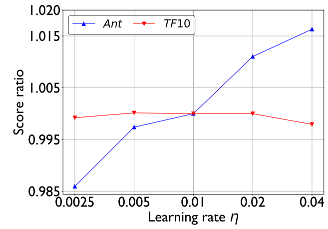

Figure 7: **The ratio of** the performance of our *RGD* method with η **to** the performance with $\eta = 0.01$.

G Broader Impact and Limitation

Broader impact. Our research has the potential to significantly accelerate advancements in fields such as new material development, biomedical innovation, and robotics technology. These advancements could lead to breakthroughs with substantial positive societal impacts. However, we recognize that, like any powerful tool, there are inherent risks associated with the misuse of this technology. One concerning possibility is the exploitation of our optimization techniques to design objects or entities for malicious purposes, including the creation of more efficient weaponry or harmful biological agents. Given these potential risks, it is imperative to enforce strict safeguards and regulatory measures, especially in areas where the misuse of technology could lead to significant ethical and societal harm. The responsible application and governance of such technologies are crucial to ensuring that they serve to benefit society as a whole.

Limitation. We recognize that the benchmarks utilized in our study may not fully capture the complexities of more advanced applications, such as protein drug design, primarily due to our current limitations in accessing wet-lab experimental setups. Moving forward, we aim to mitigate this limitation by fostering partnerships with domain experts, which will enable us to apply our method to more challenging and diverse problems. This direction not only promises to validate the efficacy of our approach in more complex scenarios but also aligns with our commitment to pushing the boundaries of what our technology can achieve.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the limitations in Appendix G.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

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

   Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

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

4. **Experimental Result Reproducibility**

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

   Answer: [Yes]

   Justification: We provide our code link in the abstract and detail our settings in Section 4.3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

Answer: [Yes]

Justification: We provide a link to our source code in the abstract and thoroughly describe our experimental settings in Section 4.3.

Guidelines:

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

6. **Experimental Setting/Details**

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

Answer: [Yes]

Justification: We detail our setting in Section 4.3 and also discuss hyperparameter sensitivity in Appendix F.

Guidelines:

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

7. **Experiment Statistical Significance**

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

Answer: [Yes]

Justification: To ensure reliability and consistency in our comparative analysis, each experimental setting was replicated across 8 independent runs, unless stated otherwise, with the presentation of both mean values and standard errors.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

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

   Answer: [Yes]

   Justification: We have discussed these in Section 4.3.

   Guidelines:

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

9. **Code Of Ethics**

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

   Answer: [Yes]

   Justification: We preserve anonymity and conform with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader Impacts**

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

   Answer: [Yes]

   Justification: We discuss both potential positive and negative impacts in Appendix G.

   Guidelines:

   - The answer NA means that there is no societal impact of the work performed.
   - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
   - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: We do not release any datasets nor pre-trained models.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We have duly credited all utilized assets and adhered to their respective licenses and terms of use.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

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

    Answer: [Yes]

    Justification: We plan to open-source our code and have ensured thorough documentation of the code.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: This paper does not engage in crowdsourcing or involve studies with human participants.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: This paper does not engage in crowdsourcing or research involving human subjects.

    Guidelines:

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