Derivative-Free Guidance in Continuous and Discrete Diffusion Models with Soft Value-based Decoding

Xiner Li [1,4] **Yulai Zhao** [2] **Chenyu Wang** [3] **Gabriele Scalia** [4]
Gokcen Eraslan [4] **Surag Nair** [4] **Tommaso Biancalani** [4] **Shuiwang Ji** [1]
Aviv Regev [4*] **Sergey Levine** [5*] **Masatoshi Uehara** [6*]
[1]Texas A&M University [2]Princeton University [3]MIT [4]Genentech
[5]UC Berkeley [6]EvolutionaryScale

Abstract

Diffusion models excel at capturing the natural design spaces of images, molecules, and biological sequences. However, for many applications, rather than merely generating designs that are natural, we aim to optimize downstream reward functions while preserving the naturalness of these design spaces. Existing methods for achieving this goal often require "differentiable" proxy models (*e.g.*, classifier guidance) or computationally-expensive fine-tuning of diffusion models (*e.g.*, classifier-free guidance, RL-based fine-tuning). Here, we propose a new method, **S**oft Value-based **D**ecoding in **D**iffusion models (**SVDD**), to address these challenges. **SVDD** is an iterative sampling method that integrates soft value functions, which looks ahead to how intermediate noisy states lead to high rewards in the future, into the standard inference procedure of pre-trained diffusion models. Notably, **SVDD** avoids fine-tuning generative models and eliminates the need to construct differentiable models. This enables us to (1) directly use non-differentiable features/reward feedback, commonly used in many scientific domains, and (2) apply our method to recent discrete diffusion models in a principled way. Finally, we demonstrate the effectiveness of **SVDD** across several domains, including image generation, molecule generation (optimization of docking scores, QED, SA), and DNA/RNA generation (optimization of activity levels). The code is available at https://github.com/masa-ue/SVDD.

1 Introduction

Diffusion models have gained popularity as powerful generative models. Their applications extend beyond image generation to include natural language generation (Sahoo et al., 2024; Shi et al., 2024; Lou et al., 2023), molecule generation (Jo et al., 2022; Vignac et al., 2022), and biological (DNA, RNA, protein) sequence generation (Avdeyev et al., 2023; Stark et al., 2024). In each of these domains, diffusion models have been shown to be very effective at capturing complex natural distributions. However, in practice, we might not only want to generate *realistic* samples, but to produce samples that optimize specific downstream reward functions while preserving naturalness by leveraging pre-trained models. For example, in computer vision, we might aim to generate natural images with high aesthetic and alignment scores (Black et al., 2023; Fan et al., 2023). In drug discovery, we may seek to generate valid molecules with high QED/SA/docking scores (Lee et al., 2023; Jin et al., 2018) or RNAs (such as mRNA vaccines (Cheng et al., 2023)) with high translational efficiency and stability (Castillo-Hair & Seelig, 2021; Asrani et al., 2018), and regulatory DNAs that drives high cell-specificity of expression (Gosai et al., 2023; Taskiran et al., 2024; Lal et al., 2024).

*Corresponding authors: regev.aviv@gene.com, svlevine@eecs.berkeley.edu, ueharamasatoshi136@gmail.com

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

The optimization of downstream reward functions using pre-trained diffusion models has been approached in various ways. In our work, we focus on non-fine-tuning-based methods because fine-tuning generative models (e.g., when using classifier-free guidance (Ho et al., 2020) or RL-based fine-tuning (Black et al., 2023; Fan et al., 2023; Clark et al., 2023; Prabhudesai et al., 2023)) often becomes computationally intensive, especially as pre-trained generative models grow larger in the era of "foundation models". Although classifier guidance and its variants (*e.g.*, Dhariwal & Nichol (2021); Song et al. (2020b); Chung et al. (2022); Bansal et al. (2023); Ho et al. (2022)) have shown some success as non-fine-tuning methods in these settings, they face significant challenges. First, as they would require constructing *differentiable* proxy models, they cannot directly incorporate useful *non-differentiable* features (*e.g.*, molecular/protein descriptors (van Westen et al., 2013; Ghiringhelli et al., 2015; Gainza et al., 2020)) or non-differentiable reward feedback (*e.g.*, physics-based simulations such as Vina and Rosetta (Trott & Olson, 2010; Alhossary et al., 2015; Alford et al., 2017)), which are particularly important in molecule design to optimize docking scores, stability, etc. This limitation also hinders the principled application of current classifier guidance methods to recently-developed discrete diffusion models (Austin et al., 2021; Campbell et al., 2022; Lou et al., 2023) (*i.e.*, without transforming the discrete space into the Euclidean space).

To tackle these challenges, we propose a novel method, **S**oft Value-based **D**ecoding in **D**iffusion models (**SVDD**), for optimizing downstream reward functions in diffusion models (Figure 1). Inspired by recent literature on RL-based fine-tuning, we first introduce soft value functions that serve as look-ahead functions, indicating how intermediate noisy samples lead to high rewards in the *future* of the diffusion denoising process. After learning (or approximating) these value functions, we present a new inference-time technique, **SVDD**, which obtains multiple noisy states from the policy (*i.e.*, denoising map) of pre-trained diffusion models and selects the sample with the highest value function at each time step. Notably, **SVDD** does not require any additional learning as long as we have access to the reward feedback by utilizing the characteristics of diffusion models (*i.e.,* the forward process in diffusion models to directly map t to 0 in terms of expectation in Figure 1). Finally, to fully leverage test-time compute, we introduce a more computationally efficient variant, **SVDD**-R, which halts unpromising samples early and reallocates inference-time resources to more promising ones.

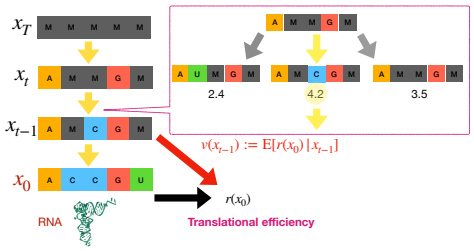


Figure 1: Summary of **SVDD**. v denotes value functions that predict reward $r(x_0)$ (at time 0) from states at time $t-1$. **SVDD** involves two steps: (1) generating multiple noisy states from pre-trained models, and (2) selecting the state with the highest value according to the value function.

Our novel technique for optimizing reward functions in pre-trained diffusion models makes two contributions. First, it eliminates the need to construct differentiable proxy models. This allows for the use of non-differentiable reward feedback, which is common in scientific fields, and makes our method applicable to recent discrete diffusion (Shi et al., 2024; Sahoo et al., 2024) models in a principled manner. Second, it avoids the need to fine-tune the generative model itself. This addresses the high computational cost associated with fine-tuning diffusion models. We demonstrate the effectiveness of **SVDD** across diverse domains, including image generation, molecule generation (optimization of docking scores, QED), and DNA/RNA generation (optimization of activity levels).

2 Related Works

We outline methods relevant to our goal. We defer other relevant works (e.g., decoding in LLMs, discrete diffusion models) to Section B due to space constraints.

Classifier guidance (Dhariwal & Nichol, 2021; Song et al., 2020a). It has been widely used to condition pre-trained diffusion models without fine-tuning. Although these methods do not originally focus on optimizing reward functions, they can be applied for this purpose (Uehara et al., 2024a, Section 6.2). In this approach, an additional derivative of a certain value function is incorporated into the drift term (mean) of pre-trained diffusion models during inference. Subsequent variants (*e.g.*,

Chung et al. (2022); Ho et al. (2022); Guo et al. (2024); Yu et al. (2023)) have been proposed to simplify the learning of value functions. However, these methods require constructing *differentiable* models, which limits their applicability to non-differentiable features/reward feedbacks commonly encountered in scientific domains as mentioned in Section 1. Additionally, this approach cannot be directly extended to discrete diffusion models (e.g., (Lou et al., 2023; Shi et al., 2024; Sahoo et al., 2024)) in a principle way. Our approach aims to address these challenges.

Note a notable exception of classifier guidance tailored to discrete diffusion models has been recently proposed by Nisonoff et al. (2024); Schiff et al. (2024). However, **SVDD** can be applied to both continuous and discrete diffusion models in a unified manner. Furthermore, their practical method requires the differentiability of proxy models, unlike **SVDD**. We compare its performance with our method in Section 6. We provide further details in Section E.

SMC-Based Methods. Several SMC-based (sequential Monte-Carlo) guidance methods, which also do not need non-differentiable reward feedback, have been proposed in recent work (Wu et al., 2024; Trippe et al., 2022; Dou & Song, 2024; Phillips et al., 2024; Cardoso et al., 2023), primarily targeting conditioning problems. In contrast, our focus is on reward maximization. While there are concurrent efforts exploring reward-guided generation (Kim et al., 2025; Singhal et al., 2025; Ma et al., 2025; Guo et al., 2025), our approach differs in key ways. Most notably, our algorithm is an instantiation of nested importance sampling (nested-IS) SMC (Naesseth et al., 2019, Algorithm 5, 13), whereas other concurrent works typically employ standard sequential Monte Carlo. Additionally, we introduce a novel technique for reward maximization in Section 5 that integrates ideas from both SMC and nested-IS SMC organically.

Fine-tuning in diffusion models. For more details, refer to Section B. In essence, we *do not* intend to claim that our approach is superior, as both methods have their respective advantages.

3 Preliminaries and Goal

We describe the standard method for training diffusion models and outline the objective of our work: optimizing downstream reward functions given pre-trained diffusion models.

3.1 Diffusion Models

In diffusion models (Ho et al., 2020; Song et al., 2020a), our goal is to learn a sampler $p(x) \in \Delta(\mathcal{X})$ given data consisting of $x \in \mathcal{X}$. The training process for a standard diffusion model is summarized as follows. First, we introduce a (fixed) forward process $q_t : \mathcal{X} \to \Delta(\mathcal{X})$. Next, we aim to learn a backward process: $\{p_t\}$ where each p_t is $\mathcal{X} \to \Delta(\mathcal{X})$ so that the distributions induced by the forward process and backward process match marginally. For this purpose, by parametrizing the backward processes with $\theta \in \mathbb{R}^d$, we typically use the loss derived from the variational lower bound of the negative log-likelihood (*i.e.*, ELBO).

Here are two examples of concrete parameterizations. Let $\alpha_t \in \mathbb{R}$ be a noise schedule.

Example 1 (Continuous space)**.** *When $\mathcal{X}$ is Euclidean space, we typically use the Gaussian distribution $q_t(\cdot \mid x) = \mathcal{N}(\sqrt{\alpha_t}x, (1-\alpha_t))$. Then, the backward process can be parameterized as the Gaussian distribution whose mean is*

$$\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\hat{x}_0(x_t;\theta)}{1-\bar{\alpha}_t},$$

where $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$. Here, $\hat{x}_0(x_t;\theta)$ is a neural network that predicts x_0 from x_t ($\mathbb{E}_q[x_0|x_t]$).

Example 2 (Discrete space in Sahoo et al. (2024); Shi et al. (2024))**.** *Let $\mathcal{X}$ be a space of one-hot column vectors $\{x \in \{0,1\}^K : \sum_{i=1}^K x_i = 1\}$, and $\mathrm{Categorical}(\pi)$ be the categorical distribution over K classes with probabilities given by $\pi \in \Delta^K$ where Δ^K denotes the K-simplex. A typical choice is $q_t(\cdot \mid x) = \mathrm{Categorical}(\alpha_t x + (1-\alpha_t)\mathbf{m})$ where $\mathbf{m} = [0, \cdots, 0, \mathrm{Mask}]$. Then, the backward process can be parameterized as*

$$\begin{cases} I(\cdot = x_t), & \text{if } x_t \neq \mathbf{m}, \\ \mathrm{Categorical}\left(\frac{(1-\bar{\alpha}_{t-1})\mathbf{m} + (\bar{\alpha}_{t-1}-\bar{\alpha}_t)\hat{x}_0(x_t;\theta)}{1-\bar{\alpha}_t}\right), & \text{if } x_t = \mathbf{m}, \end{cases}$$

where $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$. Here, $\hat{x}_0(x_t;\theta)$ is a neural network that predicts x_0 from x_t.

After learning the backward process, we can sample from a distribution that emulates training data (*i.e.*, $p(x)$) by sequentially sampling $\{p_t(\cdot \mid x_{t+1})\}_{t=T}^0$ from $t = T$ to $t = 0$.

Notation. The notation δ_a denotes a Dirac delta distribution centered at a. The notation $\propto$ indicates that the distribution is equal up to a normalizing constant. With slight abuse of notation, we often denote $p_T(\cdot|\cdot,\cdot)$ by $p_T(\cdot)$.

3.2 Objective: Generating Samples with High Rewards While Preserving Naturalness

We consider a scenario where we have a pre-trained diffusion model, which is trained using the loss function explained in Section 3.1. These pre-trained models are typically designed to excel at characterizing the natural design space (*e.g.*, image space, biological sequence space, or chemical space) by emulating the extensive training dataset. Our work focuses on obtaining samples that also optimize downstream reward functions $r : \mathcal{X} \to \mathbb{R}$ (*e.g.*, Quantitative Estimate of Druglikeness (QED) and Synthetic Accessibility (SA) in molecule generation), while maintaining the naturalness by leveraging pre-trained diffusion models. We formalize this goal as follows.

Given a pre-trained model $\{p_t^{\mathrm{pre}}\}_{t=T}^0$, we denote the induced distribution by $p^{\mathrm{pre}} \in \Delta(\mathcal{X})$ (*i.e.*, $p^{\mathrm{pre}}(x_0) = \int \{\prod_{t=T+1}^1 p_{t-1}^{\mathrm{pre}}(x_{t-1}|x_t)\} dx_{1:T}$). We aim to sample from the following distribution:

$$p^{(\alpha)}(x) := \underset{p \in [\Delta(\mathcal{X})]}{\operatorname{argmax}} \underbrace{\mathbb{E}_{x\sim p(\cdot)}[r(x)]}_{\textbf{term (a)}} - \alpha \underbrace{\mathrm{KL}(p(\cdot) \| p^{\mathrm{pre}}(\cdot))}_{\textbf{term(b)}} \propto \exp(r(x)/\alpha) p^{\mathrm{pre}}(x).$$

Here, term (a) is introduced to optimize the reward function, while term (b) is used to maintain the naturalness of the generated samples. Note these targets are widely employed in generative models, such as RLHF in LLMs (Ziegler et al., 2019; Ouyang et al., 2022). Notably, we generally focus on regimes where α is small, as our primary objective is to generate high-reward samples. In the extreme scenario, when $\alpha \to 0$, this reduces to $\operatorname{argmax}_{x \in \mathrm{Supp}(p^{pre})} r(x)$.

Existing methods. Several existing approaches target this goal (or its variant), including classifier guidance, fine-tuning (RL-based or classifier-free), and Best-of-N. In our work, we focus on non-fine-tuning-based methods; specifically, we aim to address the limitations of these methods: the requirement for differentiable proxy models in classifier guidance and the inefficiency of Best-of-N.

Finally, we note that all results discussed in this paper can be easily extended to cases where the pre-trained model is a conditional diffusion model. For example, in our image experiments (Section 6), the pre-trained model is a conditional diffusion model conditioned on text (e.g., Stable Diffusion).

4 Soft Value-Based Decoding in Diffusion Models

We present the motivation behind developing our new algorithm. We then introduce **SVDD**, which satisfies desired properties, *i.e.*, the lack of need for fine-tuning or constructing differentiable models.

4.1 Key Observation

We introduce several key concepts. First, we define the *soft value function* $t \in [T+1, \cdots, 1]$:

$$v_{t-1}(\cdot) := \alpha \log \mathbb{E}_{x_0 \sim p^{\mathrm{pre}}(x_0|x_{t-1})} \left[\exp\left(\frac{r(x_0)}{\alpha} \right) | x_{t-1} = \cdot \right], \tag{1}$$

where $\mathbb{E}_{\{p^{\mathrm{pre}}\}}[\cdot]$ is induced by $\{p_t^{\mathrm{pre}}(\cdot|x_{t+1})\}_{t=T}^0$. This value function represents the expected future reward at $t = 0$ from the intermediate noisy state at $t - 1$.

Next, we define the following *soft optimal policy* (denoising process) $p_{t-1}^{\star,\alpha} : \mathcal{X} \to \Delta(\mathcal{X})$ weighted by value functions $v_{t-1} : \mathcal{X} \to \mathbb{R}$:

$$p_{t-1}^{\star,\alpha}(\cdot|x_t) = \frac{p_{t-1}^{\mathrm{pre}}(\cdot|x_t) \exp(v_{t-1}(\cdot)/\alpha)}{\int p_{t-1}^{\mathrm{pre}}(x|x_t) \exp(v_{t-1}(x)/\alpha) dx}.$$

Here, v_t are soft value functions and $p_t^{\star,\alpha}$ are soft optimal policies, because they literally correspond, respectively, to soft value functions and soft optimal policies, where we embed diffusion models into entropy-regularized MDPs (Geist et al., 2019), as demonstrated in Uehara et al. (2024c).

With this preparation in mind, we utilize the following key observation:

Theorem 1 (From Theorem 1 in Uehara et al. (2024c))**.** *The distribution induced by* $\{p_t^{\star,\alpha}(\cdot|x_{t+1})\}_{t=T}^0$ *is the target distribution* $p^{(\alpha)}(x)$*, i.e.,*

$$p^{(\alpha)}(x_0) = \int \left\{ \prod_{t=T+1}^{1} p_{t-1}^{\star,\alpha}(x_{t-1}|x_t) \right\} dx_{1:T}.$$

While Uehara et al. (2024c) presents this theorem, they use it primarily to interpret RL-based fine-tuning methods in Fan et al. (2023); Black et al. (2023). In contrast, our work explores how to convert this into a new fine-tuning-free optimization algorithm.

Our motivation for a new algorithm. Theorem 1 states that if we can hypothetically sample from $\{p_t^{\star,\alpha}(\cdot \mid x_{t+1})\}_{t=T}^0$, we can sample from the target distribution $p^{(\alpha)}$. However, there are two challenges in sampling from each $p_{t-1}^{\star,\alpha}$: (1) the soft-value function v_{t-1} in $p_{t-1}^{\star,\alpha}$ is unknown, and (2) it is unnormalized (*i.e.*, calculating the normalizing constant is hard).

We address the first challenge in Section C. Assuming the first challenge is resolved, we consider how to tackle the second challenge. A natural approach is to use importance sampling (IS):

$$p_{t-1}^{\star,\alpha}(\cdot|x_t, c) \approx \sum_{m=1}^{M} \frac{w_{t-1}^{\langle m \rangle}}{\sum_{j=1}^{M} w_{t-1}^{\langle j \rangle}} \delta_{x_{t-1}^{\langle m \rangle}},\ \{x_{t-1}^{\langle m \rangle}\}_{m=1}^{M} \sim p_{t-1}^{\mathrm{pre}}(\cdot \mid x_t),$$

where $w_{t-1}^{\langle m \rangle} := \exp(v_{t-1}(x_{t-1}^{\langle m \rangle})/\alpha)$. Thus, we can approximately sample from $p_{t-1}^{\star,\alpha}(\cdot|x_t)$ by obtaining multiple (M) samples from pre-trained diffusion models and selecting the sample based on an index, which is determined by sampling from the categorical distribution with mean $\{w_{t-1}^{\langle m \rangle} / \sum_j w_{t-1}^{\langle j \rangle}\}_{m=1}^{\langle M \rangle}$.

Note that Best-of-N, which generates multiple samples and selects the highest reward sample, is technically considered IS, where the proposal distribution is the entire $p^{\mathrm{pre}}(x_0) = \int \prod_t \{p_t^{\mathrm{pre}}(x_{t-1} \mid x_t)\} dx_{1:T}$. However, the use of importance sampling in our algorithm differs significantly, as we apply it at each time step to approximate each soft-optimal policy.

4.2 Inference-Time Algorithm

Algorithm 1 SVDD (**S**oft **V**alue-Based **D**ecoding in **D**iffusion Models)

1: **Require**: Estimated soft value function $\{\hat{v}_t\}_{t=T}^0$ (refer to Algorithm 3 or Algorithm 4 in Section C), pre-trained diffusion models $\{p_t^{\mathrm{pre}}\}_{t=T}^0$, hyperparameter $\alpha \in \mathbb{R}$, Batch size N, IS size M.
2: **for** $t \in [T+1, \cdots, 1]$ **do**
3: For each $i \in [1, \cdots, N]$, get M samples from pre-trained polices $\{x_{t-1}^{\langle i,m \rangle}\}_{m=1}^{M} \sim p_{t-1}^{\mathrm{pre}}(\cdot|x_t^{\langle i \rangle})$, and for each m, calculate $w_{t-1}^{\langle i,m \rangle} := \exp(\hat{v}_{t-1}(x_{t-1}^{\langle i,m \rangle})/\alpha)$
4: $x_{t-1}^{\langle i \rangle} \leftarrow x_{t-1}^{\langle i, \zeta_{t-1} \rangle}$ after selecting an index: $\zeta_{t-1} \sim \mathrm{Categorical}\left(\left\{ \frac{w_{t-1}^{\langle i,m \rangle}}{\sum_{j=1}^{M} w_{t-1}^{\langle i,j \rangle}} \right\}_{m=1}^{M} \right)$,
5: **end for**
6: **Output**: x_0

Now, by leveraging the observation in Algorithm 1, we introduce our algorithm. Our algorithm is an iterative sampling method that integrates soft value functions into the standard inference procedure of pre-trained diffusion models. Each step is designed to approximately sample from a value-weighted policy $\{p_t^{\star,\alpha}(\cdot|x_{t+1}\}_{t=T}^0$.

Finally, we note several key points.

- In practice, we typically recommend a low value for α, as our focus is on optimizing the reward.
- Performance improves with increasing M, albeit at the cost of computational time. This reflects a test-time scaling behavior, where improved performance is achieved through increased computational resources. We present empirical results for varying values of M in Section 6.

- Line 1 can be computed in parallel at the cost of additional memory (scaled by M). If Line 1 is not computed in parallel, the computational time in SVDD would be approximately M times that of the standard inference procedure. We will empirically demonstrate it in Section 6.
- A proposal distribution different from p_{t-1}^{pre} in line 1 can be applied (see Section F). For instance, classifier guidance or its variants may be used to obtain better proposal distributions than those from the pure pre-trained model. For further details, refer to Section F.

4.3 Approximation of Soft Value Functions

A key remaining question is how to obtain the soft value function. Based on the definition of (1), we propose two general approaches: the Monte Carlo regression method and the posterior mean approximation method.

- **Monte Carlo approach.** We approximate a value function via supervised learning based on (1).
- **Posterior mean approximation approach.** We use $r(\hat{x}_0(x_t))$, where $\hat{x}_0(x_t)$ is a predictor in pre-trained diffusion models from x_t to x_0. This approach is motivated by the observation that the expectation in (1) can be approximated using a Dirac delta distribution centered at its mean. Similar approximations have been widely adopted in the context of classifier guidance, such as DPS (Chung et al., 2022), universal guidance (Bansal et al., 2023) with substantial empirical success.

Notably, the latter requires no additional training and involves only a single forward pass through the pre-trained diffusion model; thus, we adopt it as our default choice. In practice, we observe that both approaches yield comparable performance. For a more detailed discussion, see Section C.

4.4 Relation with SMC

SVDD differs from a direct instantiation of standard SMC like Wu et al. (2024), as **SVDD** more closely resembles the nested SMC algorithm. This distinction between standard and nested SMC has also been recognized in the computational statistics literature (Naesseth et al., 2019, Algorithms 5 and 13). Notably, while standard SMC requires computing the ratio of approximated value functions (i.e., $\exp(\hat{v}_{t-1}(x_{t-1}/\alpha)/\exp(\hat{v}_t(x_t)/\alpha))$, which is often unstable in practice, our method avoids this step, leading to improved robustness. For more details, refer to Section D.

4.5 Potential Limitation

Our approach requires more computational resources (if not parallelized) or memory (if parallelized), approximately M times more than standard inference methods, as noted in Section 4.2. Taking this aspect into account, we compare **SVDD**, with baselines such as best-of-N in our experimental section (Section 6). For gradient-based approaches like classifier guidance and DPS, while a direct comparison with **SVDD** is challenging, it is important to note that these methods also incur additional computational and memory complexity due to the backward pass, which **SVDD** avoids.

5 A More Efficient SVDD via Replacement of Non-Promising Samples

As mentioned in Section 4.2, computational efficiency is crucial in our context, as the performance of inference-time techniques is expected to scale with the amount of computation at inference. Here, to further improve reward maximization in a computationally efficient manner, we introduce an enhanced variant of **SVDD**.

The core idea is as follows. In Algorithm 1, each sample $x^{[i]}$ ($i = 1, \ldots, N$) is propagated from $t = T$ to $t = 0$. However, during intermediate steps, it often becomes apparent that some samples are unlikely to yield high rewards. In such cases, continuing to sample them is inefficient. Instead, we propose reallocating computation by terminating unpromising samples early and redistributing resources to more promising ones within the same batch. This intuition is implemented through a resampling step based on value functions, which serve as look-ahead estimators, as illustrated in Figure 2. The detailed procedure is presented in Algorithm 2, where resampling is guided by exponentially weighted value functions.

Notably, this step is computationally inexpensive, as it requires no additional forward passes through neural networks. In Section 6, we empirically evaluate the effectiveness of this component.

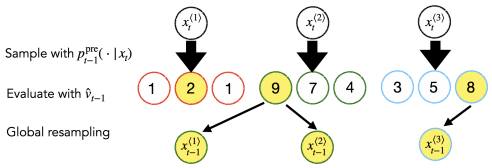


Figure 2: Illustration of Algorithm 2 where $M = 3$ and $N = 3$. After conducting weighted local sampling from pre-trained models, we perform a global resampling step.

Note that this algorithm is an improved version of Algorithm 1, incorporating ideas from the standard SMC implementation discussed in Remark 4.4. In standard SMC, global resampling is also used to select promising samples based. Similarly, our method introduces interaction among samples within a batch. However, unlike standard SMC, we first significantly improve each sample individually—following the primary step of **SVDD** in Algorithm 1—before performing global resampling.

Algorithm 2 SVDD with **R**eplacement of Non-Promising Samples

1: **Require**: Estimated soft value function $\{\hat{v}_t\}_{t=T}^0$ (refer to Algorithm 3 or Algorithm 4), pre-trained diffusion models $\{p_t^{\text{pre}}\}_{t=T}^0$, hyperparameter $\alpha \in \mathbb{R}$, Batch size N, IS size M.
2: **for** $t \in [T+1, \cdots, 1]$ **do**
3: Obtain $x_{t-1}^{\langle i, \zeta_{t-1} \rangle}$ following Line 3, 4 in Algorithm 1 and set it as $\bar{x}_{t-1}^{\langle i \rangle}$
4: **Global resampling**: choose

$$\{x_{t-1}^{\langle i \rangle}\}_{i=1}^N \leftarrow \{\bar{x}_{t-1}^{\langle \zeta_{t-1}^{\langle i \rangle} \rangle}\}_{i=1}^N, \quad \zeta_{t-1}^{\langle i \rangle} \sim \text{Categorical}\left(\frac{\exp(v(\bar{x}_{t-1}^{\langle i \rangle})/\alpha)}{\sum_j \exp(v(\bar{x}_{t-1}^{\langle j \rangle})/\alpha)}\right) \tag{2}$$

5: **end for**
6: **Output**: x_0

6 Experiments

We conduct experiments to assess the performance of our algorithm relative to baselines and its sensitivity to various hyperparameters. We start by outlining the experimental setup, and then present the results. The code is in Section G.

6.1 Settings

Methods to compare. We compare **SVDD** to several representative methods capable of performing reward maximization during inference, discussed in Section 2. Note for all methods relevant to α, we generally use the same set of α. For example, we use $\alpha = 0.01$ for biological sequences and images and $\alpha = 0.05$ for molecules.

- **Pre-trained models:** We generate samples using pre-trained models.
- **Best-of-N:** We generate samples from pre-trained models and select the top $1/N$ samples. This selection is made to ensure that the computational time during inference is approximately equivalent to that of **SVDD**.
- **DPS (Chung et al., 2022):** It is a widely used training-free version of classifier guidance. For discrete diffusion, we combine it with the state-of-the-art approach (Nisonoff et al., 2024).
- **SMC-Based Methods (e.g., Wu et al. (2024)):** Methods, which do not require differentiable models, like **SVDD**. Further details are in Appendix D.
- **SVDD (Ours):** We implement Algorithm 1 with the posterior mean approximation of soft-value functions. We generally set $M = 20$ for images and $M = 10$ for other domains. Recall M is the duplication size in the IS part.
- **SVDD-R (Ours):** We implement Algorithm 2, which is an improved version of **SVDD** with replacement of non-promising samples.

Table 1: Top 10 and 50 quantiles of the generated samples (512) in terms of rewards (with 95% confidence intervals) and metrics quantifying naturalness (CLIP score and LL). **SVDD** consistently outperforms the baselines in terms of rewards, while reasonably maintaining naturalness. The top two methods are highlighted in **bold**.

Domain	Metric	Pre-Train	Best-N	DPS	SMC	**SVDD**	**SVDD-R**
Image:Compress	50% Quantile	-101.4 ± 0.22	-71.2 ± 0.46	-58.5 ± 0.39	-57.1 ± 0.4	**-49.7** ± 0.21	**-23.4** ± 0.05
	10% Quantile	-78.6 ± 0.13	-57.3 ± 0.28	-53.0 ± 0.21	-43.1 ± 0.24	**-36.5** ± 0.15	**-22.0** ± 0.05
	CLIP score	27.5 ± 0.2	29.5 ± 0.2	27.2 ± 0.0	28.3 ± 0.1	28.9 ± 0.1	61.8 ± 2.8
Image: Aesthetic	50% Quantile	5.658 ± 0.003	6.11 ± 0.007	5.60 ± 0.01	5.99 ± 0.004	**6.12** ± 0.005	**6.49** ± 0.10
	10% Quantile	5.98 ± 0.002	6.34 ± 0.004	5.99 ± 0.005	6.24 ± 0.003	**6.45** ± 0.003	**6.63** ± 0.06
	CLIP score	27.5 ± 0.2	27.2 ± 0.3	27.4 ± 0.4	29.7 ± 0.1	29.0 ± 0.3	11.5 ± 2.7
Molecule: QED	50% Quantile	0.656 ± 0.008	0.835 ± 0.009	0.679 ± 0.024	0.667 ± 0.016	**0.848** ± 0.014	**0.924** ± 0.008
	10% Quantile	0.812 ± 0.005	0.902 ± 0.006	0.842 ± 0.014	0.722 ± 0.009	**0.928** ± 0.008	**0.961** ± 0.005
	LL	-961 ± 52	-944 ± 28	-972 ± 36	-1092 ± 19	-1032 ± 22	-548 ± 50
Molecule: SA	50% Quantile	0.652 ± 0.007	0.834 ± 0.014	0.693 ± 0.022	0.786 ± 0.004	**0.925** ± 0.016	**0.992** ± 0.003
	10% Quantile	0.803 ± 0.004	0.941 ± 0.008	0.844 ± 0.013	0.796 ± 0.003	**1.000** ± 0.010	**1.000** ± 0.000
	LL	-970 ± 65	-947 ± 54	-916 ± 71	-1053 ± 31	-1040 ± 27	-938 ± 32
Molecule: Docking parp1	50% Quantile	7.15 ± 0.52	10.00 ± 0.17	7.35 ± 0.43	6.90 ± 0.60	**11.40** ± 0.22	**13.61** ± 0.20
	10% Quantile	8.59 ± 0.31	10.67 ± 0.10	9.31 ± 0.26	9.37 ± 0.36	**12.41** ± 0.13	**13.98** ± 0.08
	LL	-973 ± 30	-954 ± 23	-949 ± 56	-962 ± 37	-987 ± 28	-643 ± 35
Enhancers	50% Quantile	0.121 ± 0.033	1.807 ± 0.214	3.782 ± 0.299	4.28 ± 0.02	**5.353** ± 0.231	**6.782** ± 0.205
	10% Quantile	1.396 ± 0.020	3.449 ± 0.128	4.879 ± 0.179	5.95 ± 0.01	**6.980** ± 0.138	**8.020** ± 0.167
	LL	-259.8 ± 0.6	-262.5 ± 0.7	-257.2 ± 2.3	-287.1 ± 1.1	-247.3 ± 4.5	-249.6 ± 3.4
5'UTR	50% Quantile	0.406 ± 0.028	0.912 ± 0.023	0.426 ± 0.073	0.76 ± 0.02	**1.214** ± 0.016	**1.549** ± 0.008
	10% Quantile	0.869 ± 0.017	1.064 ± 0.014	0.981 ± 0.044	0.91 ± 0.01	**1.383** ± 0.010	**1.668** ± 0.006
	LL	-67.9 ± 0.3	- 68.2 ± 0.5	-71.6 ± 0.3	-68.7 ± 0.7	-65.8 ± 0.6	-67.0 ± 0.8

Datasets and reward models. We provide details on the pre-trained diffusion models and downstream reward functions used. For further information, refer to Section G.

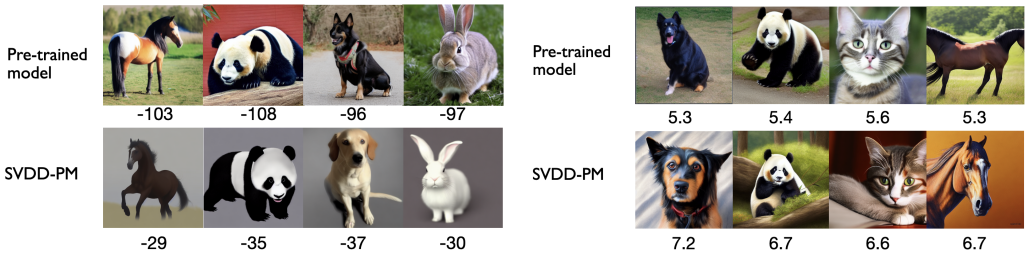
- **Images**: We use Stable Diffusion v1.5 as the pre-trained diffusion model ($T = 50$). For downstream reward functions, we use compressibility and aesthetic scores (LAION Aesthetic Predictor V2 in Schuhmann (2022)), as employed by Black et al. (2023); Fan et al. (2023). Compressibility is a *non-differentiable reward feedback*.
- **Molecules**: We use GDSS (Jo et al., 2022), trained on ZINC-250k (Irwin & Shoichet, 2005), as the pre-trained diffusion model ($T = 1000$). For downstream reward functions, we use drug-likeness (QED) and synthetic accessibility (SA) calculated by RDKit, as well as docking score (DS) calculated by QuickVina 2 (Alhossary et al., 2015), which are all *non-differentiable feedback*. Here, we renormalize SA to $(10 - \mathrm{SA})/9$ and docking score to $\max(-\mathrm{DS}, 0)$, so that a higher value indicates better performance. The docking scores measure binding affinity regarding four target proteins: Parp1, 5ht1b, braf, and jak2 following Yang et al. (2021). These tasks are critical for drug discovery. We add results for further targets in Section H.
- **DNAs (enhancers) and RNAs (5'Untranslated regions (UTRs))**: We use the discrete diffusion model (Sahoo et al., 2024), trained on datasets from Gosai et al. (2023) for enhancers, and from Sample et al. (2019) for 5'UTRs, as our pre-trained diffusion model ($T = 128$). For the reward functions, we use an Enformer model (Avsec et al., 2021) to predict activity of enhancers in the HepG2 cell line, and a ConvGRU model that predicts the mean ribosomal load (MRL) of 5'UTRs measured by polysome profiling, respectively (Sample et al., 2019). These tasks are highly relevant for cell and RNA therapies, respectively (Taskiran et al., 2024; Castillo-Hair & Seelig, 2021).

Metrics. We report the top 10% quantile and median of rewards from the generated designs, as these are the primary metrics in our context. As secondary metrics, we also report measures of the "naturalness" of the samples: CLIP score (Hessel et al., 2021) for images and (estimated) log-likelihood (LL) based on the ELBO of the pre-trained diffusion model in other domains. For additional metrics (e.g., diversity), see Section H.1.

6.2 Results

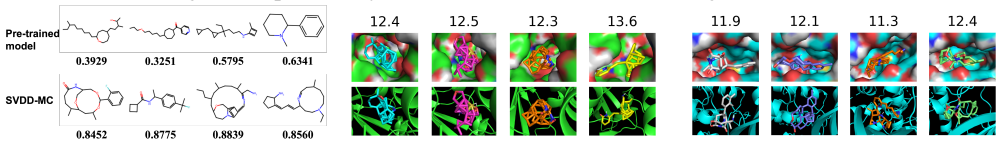
In this section, we first compare our methods with the baselines. Then, we present several ablation studies within our algorithm.

Firstly, we compare the baselines with our two proposed methods regarding rewards r and naturalness (Table 1). Overall, **SVDD** outperforms the baseline methods (**Best-of-N**, **DPS**, and **SMC**) in terms



(a) Images: compressibility

(b) Images: aesthetic scores

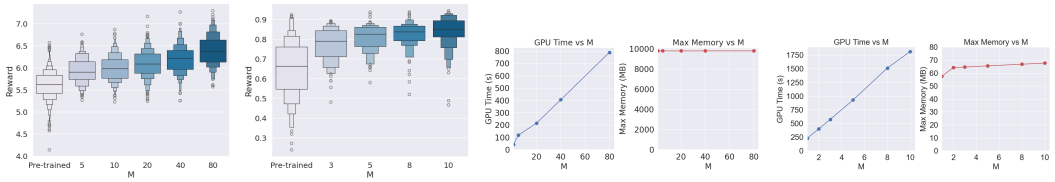


(c) Molecules: QED scores

(d) Molecules (by SVDD): high docking scores to Parp1

(e) Molecules (by SVDD): high docking scores to Jak2

Figure 3: Generated samples from **SVDD**. For more samples, refer to Section H.6. Note that the surfaces and ribbons in (e)-(f) (such as the green objects in (e)) are representations of the target proteins, while the generated molecules are displayed in the center.



(a) Performance of **SVDD** as M varies for image generation while optimizing aesthetic score.

(b) Performance of **SVDD** as M varies for molecule generation while optimizing the QED score.

(c) GPU time and max memory of **SVDD** as M varies for image generation while optimizing aesthetic score..

(d) GPU time and max memory of **SVDD** as M varies for molecule generation while optimizing the QED score.

Figure 4: Ablation studies with respect to M for **SVDD**. Figures (a) and (b) indicate that the performance increases as M increases. Figure (c) and (d) indicate that the computational time does not scale linearly with M, whereas memory usage scales linearly.

of rewards, as evidenced by higher quantiles. Additionally, our method maintains reasonably high naturalness metrics, which can also be visually verified in the images and molecular domains shown in Figure 3. This suggests that **SVDD** can generate high-reward, natural-like samples that **Best-of-N**, **DPS**, and **SMC** often struggle to generate or, in some cases, nearly fail to generate.

Remark 1 (Additional metrics)**.** *For performance on additional metrics, such as diversity, refer to Section H.1. For more quantitative metrics that describe naturalness in molecules, refer to Section H.3.*

Ablation studies in terms of the duplication size M (test-time scaling). We assessed the performance of **SVDD**(when calculating value functions in Line 1 in a non-parallel manner) along with the computational and memory complexity as M varies. First, across all domains, the performance gradually improves as M increases (Figure 4a and 4b). Second, computational complexity increases linearly with M, while memory complexity remains nearly constant (Figure 4c and 4d). This behavior is expected, as previously noted in Section 4.2. The comparison with **Best-of-N** in Table 1 is made with this consideration in mind.

Ablation studies in terms of α. We assess the performance of our algorithms by varying α in Figure 5. Here, as expected, we observe a clear trend where reducing α increases the rewards.

Note choosing a low α generally reduces diversity, as high-reward samples are more limited compared to low-reward samples. However, even with low α, we find that diversity is reasonably retained without mode collapse, as evidenced both visually in Figure 3 and quantitatively in Section H.1.

Effectiveness of replacement of non-promising samples. Finally, we compare the performance of SVDD and SVDD-R (SVDD augmented with the global resampling module). As shown, SVDD-R consistently achieves higher reward performance across all evaluated tasks. These results demonstrate that SVDD-R, by replacing non-promising samples, more effectively maximizes rewards than SVDD, without incurring additional computational costs.

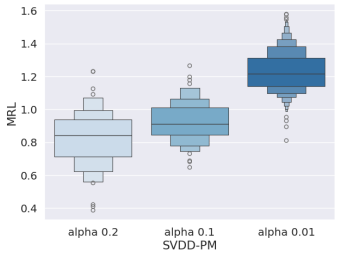


Figure 5: The performance in terms of rewards as α varies for 5'UTR design while optimizing MRL. For results in more domains, refer to Section H.5.

7 Conclusion

We propose a novel inference-time algorithm, **SVDD**, for optimizing downstream reward functions in pre-trained diffusion models that eliminate the need to construct differentiable proxy models. Future works include more applications, such as protein sequence optimization (Watson et al., 2023) and 3D molecule generation (Xu et al., 2023).

Acknowledgement

We appreciate the feedback from Christian A. Naesseth. He pointed out the connection between our work and the twisted diffusion sampler, and noted our algorithm is close to an instantiation of nested-IS SMC in our context (Naesseth et al., 2019, 2015).

References

Alford, R. F., Leaver-Fay, A., Jeliazkov, J. R., O'Meara, M. J., DiMaio, F. P., Park, H., Shapovalov, M. V., Renfrew, P. D., Mulligan, V. K., Kappel, K., et al. The rosetta all-atom energy function for macromolecular modeling and design. *Journal of chemical theory and computation*, 13(6): 3031–3048, 2017.

Alhossary, A., Handoko, S. D., Mu, Y., and Kwoh, C.-K. Fast, accurate, and reliable molecular docking with quickvina 2. *Bioinformatics*, 31(13):2214–2216, 2015.

Asrani, K. H., Farelli, J. D., Stahley, M. R., Miller, R. L., Cheng, C. J., Subramanian, R. R., and Brown, J. M. Optimization of mrna untranslated regions for improved expression of therapeutic mrna. *RNA biology*, 15(6):756–762, 2018.

Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and Van Den Berg, R. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

Avdeyev, P., Shi, C., Tan, Y., Dudnyk, K., and Zhou, J. Dirichlet diffusion score model for biological sequence generation. *arXiv preprint arXiv:2305.10699*, 2023.

Avsec, Ž., Agarwal, V., Visentin, D., Ledsam, J. R., Grabska-Barwinska, A., Taylor, K. R., Assael, Y., Jumper, J., Kohli, P., and Kelley, D. R. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature methods*, 18(10):1196–1203, 2021.

Bansal, A., Chu, H.-M., Schwarzschild, A., Sengupta, S., Goldblum, M., Geiping, J., and Goldstein, T. Universal guidance for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 843–852, 2023.

Black, K., Janner, M., Du, Y., Kostrikov, I., and Levine, S. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023.

Campbell, A., Benton, J., De Bortoli, V., Rainforth, T., Deligiannidis, G., and Doucet, A. A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*, 35:28266–28279, 2022.

Campbell, A., Yim, J., Barzilay, R., Rainforth, T., and Jaakkola, T. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. *arXiv preprint arXiv:2402.04997*, 2024.

Cardoso, G., Idrissi, Y. J. E., Corff, S. L., and Moulines, E. Monte carlo guided diffusion for bayesian linear inverse problems. *arXiv preprint arXiv:2308.07983*, 2023.

Castillo-Hair, S. M. and Seelig, G. Machine learning for designing next-generation mrna therapeutics. *Accounts of Chemical Research*, 55(1):24–34, 2021.

Cheng, F., Wang, Y., Bai, Y., Liang, Z., Mao, Q., Liu, D., Wu, X., and Xu, M. Research advances on the stability of mrna vaccines. *Viruses*, 15(3):668, 2023.

Chorowski, J. and Jaitly, N. Towards better decoding and language model integration in sequence to sequence models. *arXiv preprint arXiv:1612.02695*, 2016.

Chung, H., Kim, J., Mccann, M. T., Klasky, M. L., and Ye, J. C. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687*, 2022.

Clark, K., Vicol, P., Swersky, K., and Fleet, D. J. Directly fine-tuning diffusion models on differentiable rewards. *arXiv preprint arXiv:2309.17400*, 2023.

Dathathri, S., Madotto, A., Lan, J., Hung, J., Frank, E., Molino, P., Yosinski, J., and Liu, R. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*, 2019.

Dey, R. and Salem, F. M. Gate-variants of gated recurrent unit (gru) neural networks. In *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, pp. 1597–1600. IEEE, 2017.

Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Dong, H., Xiong, W., Goyal, D., Pan, R., Diao, S., Zhang, J., Shum, K., and Zhang, T. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*, 2023.

Dou, Z. and Song, Y. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. In *The Twelfth International Conference on Learning Representations*, 2024.

Fan, Y., Watkins, O., Du, Y., Liu, H., Ryu, M., Boutilier, C., Abbeel, P., Ghavamzadeh, M., Lee, K., and Lee, K. DPOK: Reinforcement learning for fine-tuning text-to-image diffusion models. *arXiv preprint arXiv:2305.16381*, 2023.

Ferreira DaSilva, L., Senan, S., Patel, Z. M., Reddy, A. J., Gabbita, S., Nussbaum, Z., Cordova, C. M. V., Wenteler, A., Weber, N., Tunjic, T. M., et al. Dna-diffusion: Leveraging generative models for controlling chromatin accessibility and gene expression via synthetic regulatory elements. *bioRxiv*, pp. 2024–02, 2024.

Gainza, P., Sverrisson, F., Monti, F., Rodola, E., Boscaini, D., Bronstein, M. M., and Correia, B. E. Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. *Nature Methods*, 17(2):184–192, 2020.

Geist, M., Scherrer, B., and Pietquin, O. A theory of regularized markov decision processes. In *International Conference on Machine Learning*, pp. 2160–2169. PMLR, 2019.

Ghiringhelli, L. M., Vybiral, J., Levchenko, S. V., Draxl, C., and Scheffler, M. Big data of materials science: critical role of the descriptor. *Physical review letters*, 114(10):105503, 2015.

Gosai, S. J., Castro, R. I., Fuentes, N., Butts, J. C., Kales, S., Noche, R. R., Mouri, K., Sabeti, P. C., Reilly, S. K., and Tewhey, R. Machine-guided design of synthetic cell type-specific cis-regulatory elements. *bioRxiv*, 2023.

Guo, Y., Yuan, H., Yang, Y., Chen, M., and Wang, M. Gradient guidance for diffusion models: An optimization perspective. *arXiv preprint arXiv:2404.14743*, 2024.

Guo, Y., Yang, Y., Yuan, H., and Wang, M. Training-free guidance beyond differentiability: Scalable path steering with tree search in diffusion and flow models. *arXiv preprint arXiv:2502.11420*, 2025.

Han, S., Shenfeld, I., Srivastava, A., Kim, Y., and Agrawal, P. Value augmented sampling for language model alignment and personalization. *arXiv preprint arXiv:2405.06639*, 2024.

Hessel, J., Holtzman, A., Forbes, M., Bras, R. L., and Choi, Y. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021.

Ho, J. and Salimans, T. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., and Fleet, D. J. Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646, 2022.

Inoue, F., Kreimer, A., Ashuach, T., Ahituv, N., and Yosef, N. Identification and massively parallel characterization of regulatory elements driving neural induction. *Cell stem cell*, 25(5):713–727, 2019.

Irwin, J. J. and Shoichet, B. K. ZINC- a free database of commercially available compounds for virtual screening. *Journal of chemical information and modeling*, 45(1):177–182, 2005.

Jin, W., Barzilay, R., and Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pp. 2323–2332. PMLR, 2018.

Jo, J., Lee, S., and Hwang, S. J. Score-based generative modeling of graphs via the system of stochastic differential equations. In *International Conference on Machine Learning*, pp. 10362–10383. PMLR, 2022.

Kim, S., Kim, M., and Park, D. Test-time alignment of diffusion models without reward over-optimization. In *The Thirteenth International Conference on Learning Representations*, 2025.

Lal, A., Garfield, D., Biancalani, T., and Eraslan, G. reglm: Designing realistic regulatory dna with autoregressive language models. *bioRxiv*, pp. 2024–02, 2024.

Landrum, G. et al. Rdkit: Open-source cheminformatics software, 2016. *URL http://www. rdkit. org/, https://github. com/rdkit/rdkit*, 2016.

Leblond, R., Alayrac, J.-B., Sifre, L., Pislar, M., Lespiau, J.-B., Antonoglou, I., Simonyan, K., and Vinyals, O. Machine translation decoding beyond beam search. *arXiv preprint arXiv:2104.05336*, 2021.

Lee, S., Jo, J., and Hwang, S. J. Exploring chemical space with score-based out-of-distribution generation. In *International Conference on Machine Learning*, pp. 18872–18892. PMLR, 2023.

Lew, A. K., Zhi-Xuan, T., Grand, G., and Mansinghka, V. K. Sequential monte carlo steering of large language models using probabilistic programs. *arXiv preprint arXiv:2306.03081*, 2023.

Lou, A., Meng, C., and Ermon, S. Discrete diffusion language modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

Ma, N., Tong, S., Jia, H., Hu, H., Su, Y.-C., Zhang, M., Yang, X., Li, Y., Jaakkola, T., Jia, X., et al. Inference-time scaling for diffusion models beyond scaling denoising steps. *arXiv preprint arXiv:2501.09732*, 2025.

Mudgal, S., Lee, J., Ganapathy, H., Li, Y., Wang, T., Huang, Y., Chen, Z., Cheng, H.-T., Collins, M., Strohman, T., et al. Controlled decoding from language models. *arXiv preprint arXiv:2310.17022*, 2023.

Naesseth, C., Lindsten, F., and Schon, T. Nested sequential monte carlo methods. In *International Conference on Machine Learning*, pp. 1292–1301. PMLR, 2015.

Naesseth, C. A., Lindsten, F., Schön, T. B., et al. Elements of sequential monte carlo. *Foundations and Trends® in Machine Learning*, 12(3):307–392, 2019.

Nisonoff, H., Xiong, J., Allenspach, S., and Listgarten, J. Unlocking guidance for discrete state-space diffusion and flow models. *arXiv preprint arXiv:2406.01572*, 2024.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Phillips, A., Dau, H.-D., Hutchinson, M. J., De Bortoli, V., Deligiannidis, G., and Doucet, A. Particle denoising diffusion sampler. *arXiv preprint arXiv:2402.06320*, 2024.

Prabhudesai, M., Goyal, A., Pathak, D., and Fragkiadaki, K. Aligning text-to-image diffusion models with reward backpropagation. *arXiv preprint arXiv:2310.03739*, 2023.

Qin, L., Welleck, S., Khashabi, D., and Choi, Y. Cold decoding: Energy-based constrained text generation with langevin dynamics. *Advances in Neural Information Processing Systems*, 35: 9538–9551, 2022.

Sahoo, S. S., Arriola, M., Schiff, Y., Gokaslan, A., Marroquin, E., Chiu, J. T., Rush, A., and Kuleshov, V. Simple and effective masked diffusion language models. *arXiv preprint arXiv:2406.07524*, 2024.

Sample, P. J., Wang, B., Reid, D. W., Presnyak, V., McFadyen, I. J., Morris, D. R., and Seelig, G. Human 5'utr design and variant effect prediction from a massively parallel translation assay. *Nature biotechnology*, 37(7):803–809, 2019.

Sarkar, A., Tang, Z., Zhao, C., and Koo, P. Designing dna with tunable regulatory activity using discrete diffusion. *bioRxiv*, pp. 2024–05, 2024.

Schiff, Y., Sahoo, S. S., Phung, H., Wang, G., Boshar, S., Dalla-torre, H., de Almeida, B. P., Rush, A., Pierrot, T., and Kuleshov, V. Simple guidance mechanisms for discrete diffusion models. *arXiv preprint arXiv:2412.10193*, 2024.

Schuhmann, C. LAION aesthetics, Aug 2022. URL `https://laion.ai/blog/laion-aesthetics/`.

Shi, J., Han, K., Wang, Z., Doucet, A., and Titsias, M. K. Simplified and generalized masked diffusion for discrete data. *arXiv preprint arXiv:2406.04329*, 2024.

Singhal, R., Horvitz, Z., Teehan, R., Ren, M., Yu, Z., McKeown, K., and Ranganath, R. A general framework for inference-time scaling and steering of diffusion models. *arXiv preprint arXiv:2501.06848*, 2025.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Stark, H., Jing, B., Wang, C., Corso, G., Berger, B., Barzilay, R., and Jaakkola, T. Dirichlet flow matching with applications to dna sequence design. *arXiv preprint arXiv:2402.05841*, 2024.

Taskiran, I. I., Spanier, K. I., Dickmänken, H., Kempynck, N., Pančíková, A., Ekşi, E. C., Hulselmans, G., Ismail, J. N., Theunis, K., Vandepoel, R., et al. Cell-type-directed design of synthetic enhancers. *Nature*, 626(7997):212–220, 2024.

Trippe, B. L., Yim, J., Tischer, D., Baker, D., Broderick, T., Barzilay, R., and Jaakkola, T. Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem. *arXiv preprint arXiv:2206.04119*, 2022.

Trott, O. and Olson, A. J. Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry*, 31(2):455–461, 2010.

Uehara, M., Zhao, Y., Biancalani, T., and Levine, S. Understanding reinforcement learning-based fine-tuning of diffusion models: A tutorial and review. *arXiv preprint arXiv:2407.13734*, 2024a.

Uehara, M., Zhao, Y., Black, K., Hajiramezanali, E., Scalia, G., Diamant, N. L., Tseng, A. M., Biancalani, T., and Levine, S. Fine-tuning of continuous-time diffusion models as entropy-regularized control. *arXiv preprint arXiv:2402.15194*, 2024b.

Uehara, M., Zhao, Y., Hajiramezanali, E., Scalia, G., Eraslan, G., Lal, A., Levine, S., and Biancalani, T. Bridging model-based optimization and generative modeling via conservative fine-tuning of diffusion models. *arXiv preprint arXiv:2405.19673*, 2024c.

van Westen, G. J., Swier, R. F., Cortes-Ciriano, I., Wegner, J. K., Overington, J. P., IJzerman, A. P., van Vlijmen, H. W., and Bender, A. Benchmarking of protein descriptor sets in proteochemometric modeling (part 2): modeling performance of 13 amino acid descriptor sets. *Journal of cheminformatics*, 5:1–20, 2013.

Venkatraman, S., Jain, M., Scimeca, L., Kim, M., Sendera, M., Hasan, M., Rowe, L., Mittal, S., Lemos, P., Bengio, E., et al. Amortizing intractable inference in diffusion models for vision, language, and control. *arXiv preprint arXiv:2405.20971*, 2024.

Vignac, C., Krawczuk, I., Siraudin, A., Wang, B., Cevher, V., and Frossard, P. Digress: Discrete denoising diffusion for graph generation. *arXiv preprint arXiv:2209.14734*, 2022.

Wallace, B., Dang, M., Rafailov, R., Zhou, L., Lou, A., Purushwalkam, S., Ermon, S., Xiong, C., Joty, S., and Naik, N. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8228–8238, 2024.

Watson, J. L., Juergens, D., Bennett, N. R., Trippe, B. L., Yim, J., Eisenach, H. E., Ahern, W., Borst, A. J., Ragotte, R. J., Milles, L. F., et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

Wu, L., Trippe, B., Naesseth, C., Blei, D., and Cunningham, J. P. Practical and asymptotically exact conditional sampling in diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Xu, M., Powers, A. S., Dror, R. O., Ermon, S., and Leskovec, J. Geometric latent diffusion models for 3d molecule generation. In *International Conference on Machine Learning*, pp. 38592–38610. PMLR, 2023.

Yang, K. and Klein, D. Fudge: Controlled text generation with future discriminators. *arXiv preprint arXiv:2104.05218*, 2021.

Yang, S., Hwang, D., Lee, S., Ryu, S., and Hwang, S. J. Hit and lead discovery with explorative rl and fragment-based molecule generation. *Advances in Neural Information Processing Systems*, 34: 7924–7936, 2021.

Yu, J., Wang, Y., Zhao, C., Ghanem, B., and Zhang, J. Freedom: Training-free energy-guided conditional diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 23174–23184, 2023.

Zhao, S., Brekelmans, R., Makhzani, A., and Grosse, R. Probabilistic inference in language models via twisted sequential monte carlo. *arXiv preprint arXiv:2404.17546*, 2024.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., and Irving, G. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

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: Yes, in the abstract and introduction we accurately claim our contributions and scope, and we support this with both empirical results and theoretical insights.

   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: Yes, we provide the limitations in Section 4.5.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: Yes, theoretical assumptions and proofs are clearly stated.

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 code through anonymous link as well as implementation details in Appendix G.

   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 code through anonymous link in Appendix G.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide implementation details in Appendix G.

   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: Standard deviations across tasks and metrics are provided in Section 6 and Appendices.

   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 provide details for hardware and computational requirements Appendix G.2.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in the paper conforms 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: The broader impacts are discussed in Appendix A.

    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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper does not release any new data or pre-train 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: Yes, citations are included and existing licences are discussed in Appendix G.3.

    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: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

    Answer: [NA]

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

    Guidelines:

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

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

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

    Answer: [NA]

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

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [NA]

Justification: The method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Broader Impact

This paper presents work whose goal is to advance the field of Deep Learning, particularly diffusion models. While this research primarily contributes to technical advancements in generative modeling, it has potential implications in domains such as drug discovery and biomolecular engineering. We acknowledge that generative models, particularly those optimized for specific reward functions, could be misused if not carefully applied. However, our work is intended for general applications, and we emphasize the importance of responsible deployment and alignment with ethical guidelines in generative AI. Overall, our contributions align with the broader goal of machine learning methodologies, and we do not foresee any immediate ethical concerns beyond those generally associated with generative models.

B Further Related Works

Decoding in autoregressive models with rewards. The decoding strategy, which dictates how sentences are generated from the model, is a critical component of text generation in autoregressive language models (Wu et al., 2016; Chorowski & Jaitly, 2016; Leblond et al., 2021). Recent studies have explored inference-time techniques for optimizing downstream reward functions Dathathri et al. (2019); Yang & Klein (2021); Qin et al. (2022); Mudgal et al. (2023); Zhao et al. (2024); Han et al. (2024). While there are similarities between these works and ours, to the best of our knowledge, no prior work has extended such methodologies to diffusion models. Furthermore, our approach leverages characteristics unique to diffusion models that are not present in autoregressive models such as **SVDD**-PM.

Discrete diffusion models. Based on seminal works Austin et al. (2021); Campbell et al. (2022), recent work on masked diffusion models (Lou et al., 2023; Shi et al., 2024; Sahoo et al., 2024) has demonstrated their strong performance in natural language generation. Additionally, they have been applied to biological sequence generation (*e.g.*, DNA, protein sequences in Campbell et al. (2024); Sarkar et al. (2024)). In these cases, the use of diffusion models over autoregressive models is particularly apt, given that many biological sequences ultimately adopt complex three-dimensional structures. Despite its significance, it cannot be integrated with standard classifier guidance, because adding a continuous gradient to a discrete objective is not inherently valid. Unlike standard classifier guidance, **SVDD** can be seamlessly applied to discrete diffusion models.

Fine-tuning of diffusion models. Several methods exist for fine-tuning generative models to optimize downstream reward functions, such as classifier-free guidance (Ho & Salimans, 2022) and RL-based fine-tuning (Fan et al., 2023; Black et al., 2023) or its variants (Dong et al., 2023; Wallace et al., 2024; Venkatraman et al., 2024). However, these approaches often come with caveats, including high computational costs and the risk of easily forgetting pre-trained models. In our work, we propose an inference-time technique that eliminates the need for fine-tuning generative models.

We acknowledge that if significant changes to the pre-trained models are desired, we *acknowledge* that RL-based fine-tuning (Black et al., 2023; Fan et al., 2023) could be more effective than **SVDD** for this purpose in certain scenarios, such as image examples. However, this proximity to pre-trained models could also be advantageous in the sense that it is robust against reward optimization, which conventional fine-tuning methods often suffer from by exploiting these out-of-distribution regions (Uehara et al., 2024c). Lastly, in cases where reward backpropagation (Prabhudesai et al., 2023; Clark et al., 2023; Uehara et al., 2024b) is not applicable, particularly in scientific domains for RL-based fine-tuning, we may need to rely on PPO. However, PPO is often mode-seeking and unstable, highlighting the challenges of RL-based fine-tuning in certain scenarios.

C Approximation of Soft Value Functions

Next, we describe how to get soft value functions $v_t(x)$ in practice. We propose two main approaches: a Monte Carlo regression and a posterior mean approximation approach.

C.1 Monte Carlo regression

Based on the definition of soft value functions in (1), we can formulate it as a regression problem, as described in Algorithm 3. Combining this with Algorithm 1, we refer to the entire optimization approach as **SVDD**-MC.

Algorithm 3 Value Function Estimation Using Monte Carlo Regression

1: **Require**: Pre-trained diffusion models, reward $r : \mathcal{X} \to \mathbb{R}$, function class $\Phi : \mathcal{X} \times [0, T] \to \mathbb{R}$.
2: Collect datasets $\{x_T^{(s)}, \cdots, x_0^{(s)}\}_{s=1}^S$ by rolling-out $\{p_t^{\mathrm{pre}}\}_{t=T}^0$ from $t = T$ to $t = 0$.
3: $\hat{v}' = \mathrm{argmin}_{f \in \Phi} \sum_{t=0}^T \sum_{s=1}^S \{\exp(r(x_0^{(s)})/\alpha) - \exp(f(x_t^{(s)}, t)/\alpha)\}^2$.
4: **Output**: $\hat{v}'$

C.2 Posterior Mean Approximation

Here, we perform the following approximation:

$$\begin{aligned} v_t(x) &:= \alpha \log \mathbb{E}_{x_0 \sim p^{\mathrm{pre}}(x_0|x_t)}[\exp(r(x_0)/\alpha)|x_t] \\ &\approx \alpha \log(\exp(r(\hat{x}_0(x_t))/\alpha) = r(\hat{x}_0(x_t)), \end{aligned}$$

where we replace $x_0 \sim p^{\mathrm{pre}}(x_0|x_t)$ with the Dirac delta distribution with its estimated mean $\hat{x}_0(x_t)$ obtained during pre-training in Section 3.1. Here, recall that $\hat{x}_0(x_t)$ is the approximation of $\mathbb{E}_{x_0 \sim p^{\mathrm{pre}}(x_t)}[x_0|x_t]$. In this way, we can use $r(\hat{x}_0(x_t))$ as the estimated value function (Algorithm 4). The advantage of this approach is that no additional training is required as long as we have r. When combined with Algorithm 1, we refer to the entire approach as **SVDD**-PM. Note that this is our default choice.

It is worthwhile to note in the context of classifier guidance, essentially equivalent approximations have been used (*e.g.*, DPS in Chung et al. (2022), reconstruction guidance (Ho et al., 2022), and universal guidance (Bansal et al., 2023)). In our context, we similarly empirically demonstrate that even with this approximation, we can effectively optimize the reward, as shown in Section 6. We also examine the empirical quality of this approximation in Section H.2.

Algorithm 4 Value Function Estimation using Posterior Mean Approximation

1: **Require**: Pre-trained diffusion models, reward $r : \mathcal{X} \to \mathbb{R}$
2: Set $\hat{v}^\diamond(\cdot, t) := r(\hat{x}_0(x_t = \cdot), t)$
3: **Output**: $\hat{v}^\diamond$

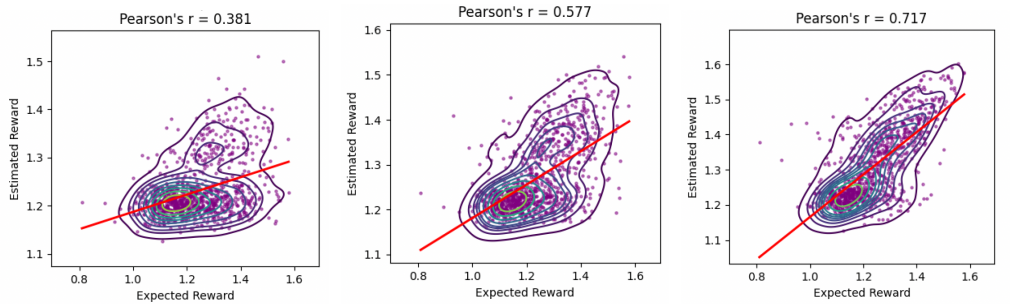
Quality of value function approximation in practice. We visualize the performance of two algorithms for value function estimation (MC and PM, as described in Algorithm 3 and Algorithm 4) in Figure 6. We make two observations. First, the performance (measured by Pearson's correlation) is generally similar, regardless of whether we use MC or PM. Second, the approximation quality improves as the time step in v_t approaches 0. This is expected, as the sample becomes progressively less noisy. For results in more domains, refer to Section H.2.

C.3 Comparison of Performance Between SVDD-MC and SVDD-PM

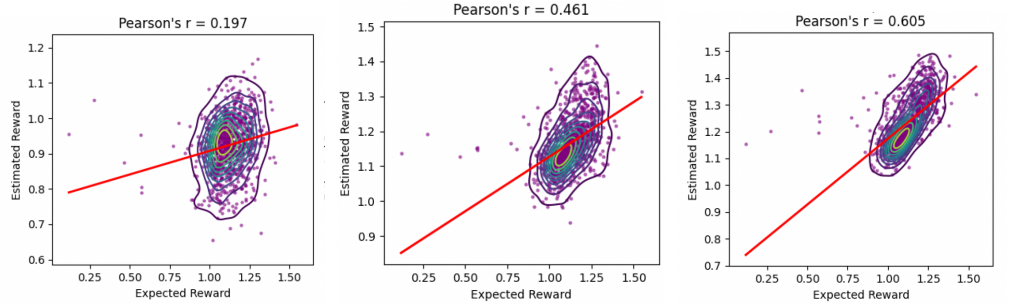
The relative performance of our **SVDD**-MC and **SVDD**-PM appears to be domain-dependent. Generally, **SVDD**-PM may be more robust since it does not require additional learning (*i.e.*, it directly utilizes reward feedback). The performance of **SVDD**-MC depends on the success of value function learning. Regarding its quality, refer to Section H.2.

D Difference Between SVDD and "Standard" SMC-Methods

In this section, we compare our algorithm with the SMC-based methods (Wu et al., 2024; Trippe et al., 2022; Dou & Song, 2024; Phillips et al., 2024; Cardoso et al., 2023) for guidance. While they originally aim to solve a conditioning problem, which is different from reward maximization, their algorithms can be converted to reward maximization. First, we will explain this converted algorithm.



(a) PM when $t = 96$ (b) PM when $t = 64$ (c) PM when $t = 16$



(d) MC when $t = 96$ (e) MC when $t = 64$ (f) MC when $t = 16$

Figure 6: The performance of PM and MC for value function approximations at each time step (Y-axis: predicted, X-axis: actual) for 5'UTR design is shown. Here, the total time step size T is 128. As the time step t approaches 0, the performance improves.

Table 2: Comparison between **SVDD**-MC and **SVDD**-PM ↑ indicates higher values correspond to better performance while ↓ indicates lower for better.

Method	Image Compressibility		Image Aesthetic Score		Enhancer HepG2		5'UTR MRL	
	Compressibility↑	Quality↓	Aesthetic ↑	Quality↓	HepG2↑	NLL↓	MRL↑	NLL↓
SVDD-PM	-49.7 ± 0.21	28.9 ± 0.1	6.12 ± 0.01	29.0 ± 0.3	5.353 ± 0.231	-247.3 ± 4.5	1.214 ± 0.016	-65.8 ± 0.6
SVDD-MC	-56.3 ± 0.29	11.4 ± 7.4	6.49 ± 0.10	11.5 ± 2.7	6.782 ± 0.205	-249.6 ± 3.4	1.549 ± 0.006	-67.0 ± 0.8
Method	Molecule QED		Molecule SA		Molecule Docking - Parp1			
	QED↑	NLL↓	SA↑	NLL↓	Docking Score↑	NLL↓		
SVDD-PM	0.848 ± 0.014	-1032 ± 22	0.925 ± 0.016	-1040 ± 27	11.40 ± 0.22	-987 ± 28		
SVDD-MC	0.924 ± 0.008	-548 ± 50	0.992 ± 0.003	-938 ± 32	13.61 ± 0.20	-643 ± 35		

We then elaborate on the differences between **SVDD**, and them in the context of reward maximization. Notably, our SVDD is an instantiation of nested-IS SMC (Naesseth et al., 2019, Algorithm 5) in the literature on computational statistics, whereas these SMC-based Methods rely on standard sequential Monte Carlo.

D.1 SMC-Based Methods

The complete algorithm of TDS in our setting is summarized in Algorithm 5. Since our notation and their notations are slightly different, we first provide a brief overview. It consists of two steps. Since our algorithm is iterative, at time point t, consider we have N samples (particles) $\{x_t^{[i]}\}_{i=1}^N$.

IS step (line 5). We generate a set of samples $\{x_{t-1}^{[i]}\}_{i=1}^N$ following a policy from a pre-trained model $p_{t-1}^{\mathrm{pre}}(\cdot|\cdot)$. In other words,

$$\forall i \in [1, \cdots, N]; x_{t-1}^{[i]} \sim p_{t-1}^{\mathrm{pre}}(\cdot|x_t^{[i]}).$$

Now, we denote the importance weight for the next particle x_{t-1} given the current particle x_t as $w(x_{t-1}, x_t)$, expressed as

$$w(x_{t-1}, x_t) := \frac{\exp(v_{t-1}(x_{t-1})/\alpha)}{\int \exp(v_{t-1}(x_{t-1})/\alpha) p_{t-1}^{\mathrm{pre}}(x_{t-1}|x_t) dx_{t-1}} = \frac{\exp(v_{t-1}(x_{t-1})/\alpha)}{\exp(v_t(x_t)/\alpha)},$$

Algorithm 5 Guidance with "Standard" SMC (for reward maximization)

1: **Require**: Estimated value functions $\{\hat{v}_t(x)\}_{t=T}^0$, pre-trained diffusion models $\{p_t^{\mathrm{pre}}\}_{t=T}^0$, hyperparameter $\alpha \in \mathbb{R}$, Batch size N
2: **for** $t \in [T+1, \cdots, 0]$ **do**
3: **IS step:**
4:

$$i \in [1, \cdots, N]; x_{t-1}^{[i]} \sim p_{t-1}^{\mathrm{pre}}(\cdot|x_t^{[i]}), w_{t-1}^{[i]} := \frac{\exp(\hat{v}_{t-1}(x_{t-1}^{[i]})/\alpha)}{\exp(\hat{v}_t(x_t^{[i]})/\alpha)}$$

5: **Selection step:** select new indices with replacement
6: $\{x_{t-1}^{[i]}\}_{i=1}^N \leftarrow \{x_{t-1}^{\zeta_{t-1}^{[i]}}\}_{i=1}^N, \quad \{\zeta_{t-1}^{[i]}\}_{i=1}^N \sim \mathrm{Cat}\left(\left\{\frac{w_{t-1}^{[i]}}{\sum_{j=1}^N w_{t-1}^{[j]}}\right\}_{i=1}^N\right)$
7: **end for**
8: **Output**: x_0

and define

$$\forall i \in [1, \cdots, N]; \quad w_{t-1}^{[i]} := w(x_{t-1}^{[i]}, x_t^{[i]}).$$

Note here we have used the soft Bellman equation:

$$\exp(v_t(x_t)/\alpha) = \int \exp(v_{t-1}(x_{t-1})/\alpha) p_{t-1}^{\mathrm{pre}}(x_{t-1}|x_t) dx_{t-1}.$$

Hence, by denoting the target marginal distribution at $t-1$, we have the following approximation:

$$p_{t-1}^{\mathrm{tar}} \underbrace{\approx}_{\mathrm{IS}} \sum_{i=1}^N \frac{w_{t-1}^{[i]}}{\sum_{j=1}^N w_{t-1}^{[j]}} \delta_{x_{t-1}^{[i]}}.$$

Selection step (line 5). Finally, we consider a resampling step. The resampling indices are determined by the following:

$$\{\zeta_{t-1}^{[i]}\}_{i=1}^N \sim \mathrm{Cat}\left(\left\{\frac{w_{t-1}^{[i]}}{\sum_{j=1}^N w_{t-1}^{[j]}}\right\}_{i=1}^N\right).$$

To summarize, we conduct

$$p_{t-1}^{\mathrm{tar}} \underbrace{\approx}_{\mathrm{IS}} \sum_{i=1}^N \frac{w_{t-1}^{[i]}}{\sum_{j=1}^N w_{t-1}^{[j]}} \delta_{x_{t-1}^{[i]}} \underbrace{\approx}_{\mathrm{Resampling}} \frac{1}{N} \sum_{i=1}^N \delta_{x_{t-1}^{\zeta_{t-1}^{[i]}}}.$$

Finally, we give several important remarks.

- In SMC, resampling is performed across the *entire* batch. However, in the algorithm, sampling is done within a single batch. Therefore, the algorithms differ significantly. We will discuss the implications in the next section.
- All of existing works Wu et al. (2024); Cardoso et al. (2023); Phillips et al. (2024); Dou & Song (2024) actually consider a scenario where the reward r is a classifier. In Algorithm 5, we tailor the algorithm for reward maximization. Vice verisa, our **SVDD** can also operate effectively when r is a classifier.
- In Wu et al. (2024); Cardoso et al. (2023); Phillips et al. (2024), the proposal distribution is not limited to the pre-trained model. Likewise, in our **SVDD**, we can select an arbitrary proposal distribution, as discussed in Section F.
- In the context of autoregressive (language) models, Zhao et al. (2024); Lew et al. (2023) proposed a similar algorithm.

D.2 Comparison of SVDD with "Standard" SMC-Based Methods (SSM) for Reward Maximization

We now compare our **SVDD** with "standard" SMC-Based Methods (SSM). Here, we write a batch size of SVDD in G. Importantly, we note that our implementation is analogous to nested-IS SMC in the literature in computational statistics; hence, many differences between nested-IS SMC (Naesseth et al., 2019, 2015) and pure SMC in computational statistics are translated here.

We first reconsider the fundamental assumptions of each algorithm. SVDD's performance, in terms of rewards, depends on the size of M but is independent of the batch size G. In contrast, the performance of SSM depends on the batch size N. With this in mind, we compare the advantages of SVDD over SSM from various perspectives.

In SSM, the "ratio" is approximated. In SVDD, we approximate each $\exp(v_{t-1}(x_{t-1})/\alpha)$ as a weight. However, in standard SMC, the ratio is approximated as a weight:

$$\frac{\exp(v_{t-1}(x_{t-1})/\alpha)}{\exp(v_t(x_t)/\alpha)}.$$

The key difference is that in SSM, both the numerator and the denominator are approximated, which could lead to greater error propagation.

Ease of parallelization in SVDD. SVDD is significantly easier to parallelize across multiple nodes. In contrast, SSM requires interaction between nodes for parallelization. This advantage is also well-documented in the context of nested-IS SMC versus standard SMC (Naesseth et al., 2019).

Relation between Blocked SMC and SVDD. When using SMC ($M = K$) and SVDD ($N = 1$, $M = K$) with $\alpha = 0$, the two algorithms become theoretically equivalent. However, in practice, we avoid setting $\alpha = 0$ to preserve diversity among samples. As a result, the two algorithms exhibit different behaviors.

E Comparison against DG (Nisonoff et al., 2024) and DiGress (Vignac et al., 2022) in Discrete Diffusion Models

Our method is closely related to guidance methods used in DG (Nisonoff et al., 2024) and DiGress (Vignac et al., 2022). However, we emphasize that our approach is more general, as it operates on any domain, including continuous spaces including Riemannian spaces and discrete spaces, in a unified manner. In this sense, a strict comparison is not feasible. With this in mind, we provide a comparison focusing on cases where all domains are discrete.

Comparison with Nisonoff et al. (2024). In this continuous framework, following the notation in (Lou et al., 2023), they propose the use of the following rate matrix:

$$Q^{\star}_{x,y}(t) = Q^{\mathrm{pre}}_{x,y}(t)\frac{\exp(v_t(y)/\alpha)}{\exp(v_t(x)/\alpha)}.$$

where $Q^{\mathrm{pre}}_{x,y}(t)$ is a rate matrix in the pre-trained model. This suggests that, with standard discretization, the optimal policy at each time step is:

$$p(x_{t+\delta t} = y | x_t = x) = \mathrm{I}(x \neq y) + Q^{\star}_{x,y}(t)(\delta t) \tag{3}$$

where δt is step size. Asymptotically, this is equivalent to sampling from the optimal policy in Theorem 1, as we will show in Remark 2. However, as Nisonoff et al. (2024) note, sampling from the optimal policy requires $O(KL)$ computation, where K is the vocabulary size and L is the sequence length, which is computationally expensive for large K and L. To address this issue, they propose using a Taylor approximation by computing the gradient once. However, this is a heuristic in the sense that there is no theoretical guarantee for this approximation. In contrast, we avoid this computational overhead in a different manner, i.e., through importance sampling and resampling. Our algorithm has an asymptotic guarantee as M goes to infinity. Empirically, we have compared two methods in Section 6.

Remark 2 (Asymptotic equivalence between formula in Nisonoff et al. (2024) and Theorem 1)**.** *The informal reasoning is as follows. Recall that the pre-trained policy can be written as*

$$p(x_{t+\delta t} = y | x_t = x) = \mathrm{I}(x \neq y) + Q^{\mathrm{pre}}_{x,y}(t)(\delta t).$$

Then, Theorem 1 states that the optimal policy is

$$\frac{\{\mathrm{I}(x \neq y) + Q^{\mathrm{pre}}_{x,y}(t)(\delta t)\} \exp(v_t(y)/\alpha)}{\sum_z \{\mathrm{I}(x \neq z) + Q^{\mathrm{pre}}_{x,z}(t)(\delta t)\} \exp(v_t(z)/\alpha)}.$$

Now, we have

$$\begin{aligned}
&\frac{\{\mathrm{I}(x \neq y) + Q^{\mathrm{pre}}_{x,y}(t)(\delta t)\} \exp(v_t(y)/\alpha)}{\sum_z \{I(x \neq z) + Q^{\mathrm{pre}}_{x,z}(t)(\delta t)\} \exp(v_t(z)/\alpha)} \\
&= \frac{\mathrm{I}(x \neq y) + Q^{\mathrm{pre}}_{x,y}(t)(\delta t)\} \exp(v_t(y)/\alpha)}{\exp(v_t(x)/\alpha)} \times \{1 + O(\delta t)\} \\
&\approx \frac{\{\mathrm{I}(x \neq y) + Q^{\mathrm{pre}}_{x,y}(t)(\delta t)\} \exp(v_t(y)/\alpha)}{\exp(v_t(x)/\alpha)} = \mathrm{I}(x \neq y) + \frac{Q^{\mathrm{pre}}_{x,y}(t)(\delta t) \exp(v_t(y)/\alpha)}{\exp(v_t(x)/\alpha)}.
\end{aligned}$$

Thus, this recovers the formula (3)*.*

Comparison with DiGress in Vignac et al. (2022). Vignac et al. (2022) proposed a diffusion model for graphs where each sampling and denoising step operates directly on the discrete structure, avoiding continuous relaxation. They discuss how to implement guidance by treating rewards as a classifier. To bypass the exponential computational cost of sampling from the optimal policy ($p^{(\alpha)}$ in Theorem 1), they employ a Taylor expansion. While it requires the calculation of gradients for value functions, then it mitigates the exponential blow-up in computational time. In contrast, we avoid this computational blow-up through importance sampling (IS) and resampling. A detailed empirical comparison between our method and theirs is left for future work.

Remark 3. *Note that in our molecule generation experiment in Section 6, we use GDSS (Jo et al., 2022), which operates in continuous space and differs from DiGress.*

F Extension with Arbitrary Proposal Distribution

Here, we describe the algorithm where the proposal distribution is not necessarily derived from the policy of the pre-trained model, as summarized in Algorithm 6. Essentially, we only adjust the importance weight. In practice, we can use the gradient of a differentiable proxy model, such as DPS, as the proposal distribution q_{t-1}. Even if the differentiable proxy (value function) models are not highly accurate, our method will still perform effectively since other value function models $\hat{v}_{t-1}$ can be non-differentiable.

Algorithm 6 SVDD (**S**oft **V**alue-Based **D**ecoding in **D**iffusion Models)

1: **Require**: Estimated soft value function $\{\hat{v}_t\}_{t=T}^0$ (refer to Algorithm 3 or Algorithm 4), pre-trained diffusion models $\{p_t^{\mathrm{pre}}\}_{t=T}^0$, hyperparameter $\alpha \in \mathbb{R}$, proposal distribution $\{q_t\}_{t=T}^0$
2: **for** $t \in [T+1, \cdots, 1]$ **do**
3: Get M samples from pre-trained polices $\{x_{t-1}^{\langle m \rangle}\}_{m=1}^M \sim q_{t-1}(\cdot|x_t)$, and for each m, and calculate

$$w_{t-1}^{\langle m \rangle} := \exp(\hat{v}_{t-1}(x_{t-1}^{\langle m \rangle})/\alpha) \times \frac{p_{t-1}^{\mathrm{pre}}(x_{t-1}^{\langle m \rangle}|x_t)}{q_{t-1}(x_{t-1}^{\langle m \rangle}|x_t)}.$$

4: $x_{t-1} \leftarrow x_{t-1}^{\langle \zeta_{t-1} \rangle}$ after selecting an index: $\zeta_{t-1} \sim \mathrm{Cat}\left(\left\{ \frac{w_{t-1}^{\langle m \rangle}}{\sum_{j=1}^M w_{t-1}^{\langle j \rangle}} \right\}_{m=1}^M \right)$,
5: **end for**
6: **Output**: x_0

G Additional Experimental Details

We further add additional experimental details. To ensure the reproducibility of this work, we provide details for the experiments in this section. We also provide an anonymous code link containing the implementation of our method and baselines: `https://anonymous.4open.science/r/SVDD/`.

G.1 Additional Setups for Experiments

G.1.1 Settings

Images. We define compressibility score as the negative file size in kilobytes (kb) of the image after JPEG compression following (Black et al., 2023). We define aesthetic scorer implemented as a linear MLP on top of the CLIP embeddings, which is trained on more than 400k human evaluations. As pre-trained models, we use Stable Diffusion, which is a common text-to-image diffusion model. As prompts to condition, we use animal prompts following (Black et al., 2023) such as [Dog, Cat, Panda, Rabbit, Horse,...].

Molecules. We calculate QED and SA scores using the RDKit (Landrum et al., 2016) library. We use the docking program QuickVina 2 (Alhossary et al., 2015) to compute the docking scores following Yang et al. (2021), with exhaustiveness as 1. Note that the docking scores are initially negative values, while we reverse it to be positive and then clip the values to be above 0, *i.e.*. We compute DS regarding four proteins, parp1 (Poly [ADP-ribose] polymerase-1), 5ht1b (5-hydroxytryptamine receptor 1B), braf (Serine/threonine-protein kinase B-raf), and jak2 (Tyrosine-protein kinase JAK2), which are target proteins that have the highest AUROC scores of protein-ligand binding affinities for DUD-E ligands approximated with AutoDock Vina.

DNA, RNA sequences. We examine two publicly available large datasets: enhancers ($n \approx 700k$) (Gosai et al., 2023) and UTRs ($n \approx 300k$) (Sample et al., 2019), with activity levels measured by massively parallel reporter assays (MPRA) (Inoue et al., 2019). These datasets have been widely used for sequence optimization in DNA and RNA engineering, particularly in advancing cell and RNA therapies (Castillo-Hair & Seelig, 2021; Lal et al., 2024; Ferreira DaSilva et al., 2024; Uehara et al., 2024c). In the Enhancers dataset, each x is a DNA sequence of length 200, while $y \in \mathbb{R}$ is the measured activity in the Hep cell line. In the 5'UTRs dataset, x is a 5'UTR RNA sequence of length 50, and $y \in \mathbb{R}$ is the mean ribosomal load (MRL) measured by polysome profiling.

G.1.2 Baselines and Proposals

We will explain in more detail how to implement baselines and our proposal. We use A100 GPUs for all the tasks.

SVDD-MC. In SVDD-MC, we require value function models. For images, we use standard CNNs for this purpose, with the same architecture as the reward model. For molecular tasks, we use a Graph Isomorphism Network (GIN) model (Xu et al., 2018) as the value function model. Notably, this model is not differentiable w.r.t. inputs. For GIN, we use mean global pooling and the RELU activation function, and the dimension of the hidden layer is 300. The number of convolutional layers in the GIN model is selected from the set $\{3, 5\}$; and we select the maximum number of iterations from $\{300, 500, 1000\}$, the initial learning rate from $\{$1e-3, 3e-3, 5e-3, 1e-4$\}$, and the batch size from $\{32, 64, 128\}$. For the Enhancer task, we use the Enformer model (Avsec et al., 2021) as the value function model. The Enformer trunk has 7 convolutional layers, each having 1536 channels. as well as 11 transformer layers, with 8 attention heads and a key length of 64. Dropout regularization is applied across the attention mechanism, with an attention dropout rate of 0.05, positional dropout of 0.01, and feedforward dropout of 0.4. The convolutional head for final prediction has 2*1536 input channels and uses average pooling, without an activation function. The model is trained using a batch size selected from $\{32, 64, 128, 256\}$, the learning rate from $\{$1e-4, 5e-4, 1e-3$\}$, and the maximum number of iterations from $\{$5k, 10k, 20k$\}$. For the 5'UTR task, we adopt the ConvGRU model (Dey & Salem, 2017). The ConvGRU trunk has a stem input with 4 channels and a convolutional stem that outputs 64 channels using a kernel size of 15. The model contains 6 convolutional layers, each initialized with 64 channels and a kernel size of 5. The convolutional layers use ReLU as the

activation function, and a residual connection is applied across layers. Batch normalization is applied to both the convolutional and GRU layers. A single GRU layer with dropout of 0.1 is added after the convolutional layers. The convolutional head for final prediction uses 64 input channels and average pooling, without batch normalization. For training, the batch size is selected from {16, 32, 64, 128}, the learning rate from {1e-4, 2e-4, 5e-4}, and the maximum number of iterations from {2k, 5k, 10k}. All function models are trained to converge in the learning process using MSE loss.

SVDD-PM (default version of SVDD). For this proposal, we directly use the reward feedback to evaluate. We remark when the reward feedback is also learned from offline data, technically, it would be better to use techniques mitigating over-optimization as discussed in Uehara et al. (2024c). However, since this point is tangential in our work, we don't do it.

DPS. We require differentiable models. For this task, for images, enhancers, and 5'UTRs, we use the same method as SVDD-MC. For molecules, we follow the implementation in Lee et al. (2023), and we use the same GNN model as the reward model. Note that we cannot compute derivatives with respect to adjacency matrices when using the GNN model.

SMC. For value function models, we use the same method as **SVDD**-PM.

SVDD-R. In (2), we discard suboptimal samples of a certain percentage and resample from high-quality samples . We sample by probability of exponential tiling. The percentage of samples replaced is a hyperparameter selected from {0.03, 0.04, 0.05} in (2).

G.2 Software and Hardware

Our implementation is under the architecture of PyTorch (Paszke et al., 2019). The deployment environments are Ubuntu 20.04 with 48 Intel(R) Xeon(R) Silver, 4214R CPU @ 2.40GHz, 755GB RAM, and graphics cards NVIDIA RTX 2080Ti. Each of our experiments is conducted on a single NVIDIA RTX 2080Ti or RTX A6000 GPU.

G.3 Licenses

The pretrained models and datasets for image tasks are under MIT license. The dataset for molecular tasks is under Database Contents License (DbCL) v1.0. The dataset for DNA task is covered under AGPL-3.0 license. The dataset for RNA tasks is under GPL-3.0 license. We follow the regulations for all licenses.

H Additional Experimental Results

H.1 Analysis in terms of Diversity

We measure the target reward as well as naturalness and diversity metrics for comprehensive evaluation. A higher diversity score indicates greater variability in generated samples, ensuring broader exploration of the data space.

- For images, we measure the diversity using cosine similarities of CLIP embeddings.
- For biological sequences, we measure diversity using the pairwise cosine distance in a one-hot encoded representation subtracted by 1 to capture structural variations.
- For molecular diversity, we employ the Tanimoto similarity on molecular Morgan fingerprints (ECFP), where diversity is quantified as the average pairwise similarity among generated molecules, subtracted by 1. This captures structural diversity by assessing the uniqueness of molecular substructures.

We report the diversity performances of different methods in Table 3 on page 31. As shown, baselines such as SMC, which rely on resampling strategies, show a marked drop in diversity, reinforcing the theoretical analyses. In contrast, **SVDD** exhibits a balance, ensuring a reasonable level of diversity while significantly enhancing reward.

Table 3: Diversity performance of different methods. Higher values correspond to better performance.

Task	Pre-Trained	Best-N	DPS	SMC	**SVDD-MC**	**SVDD-PM**	**SVDD-R**
Images: compressibility	0.30	0.33	0.34	0.12	0.36	0.31	0.17
Images: aesthetics	0.34	0.30	0.28	0.11	0.28	0.32	0.20
Molecule: QED	0.874	0.880	0.899	0.252	0.868	0.860	0.616
Molecule: SA	0.840	0.845	0.892	0.689	0.866	0.860	0.446
Molecule: Docking - Parp1	0.780	0.795	0.888	0.510	0.902	0.895	0.674
DNA Enhancers HepG2	0.722	0.710	0.748	0.579	0.724	0.715	0.570
5'UTR MRL	0.740	0.730	0.745	0.556	0.638	0.634	0.523

H.2 Analysis of Approximation Quality in Value Function Estimation

To show the quality of our value functions, we evaluate the effectiveness of our proposed value estimation methods for predicting the final reward of diffusion-generated samples at intermediate time steps. We assess the accuracy of our intermediate state value predictions by computing the Pearson correlation coefficient between the estimated reward and the actual reward obtained at x_0. We visualize this relationship using scatter density plots across several time steps. For sampled trajectories, we estimate rewards at various intermediate diffusion steps and compare them against the ground-truth reward from the oracle. Higher values of Pearson correlation indicate better alignment between the estimated and actual rewards.

Here are our observations.

- Figures 7 and 8 show the results using the posterior mean approximation (PM). During early diffusion steps, the estimated rewards show a weaker correlation with the final reward, which is expected. In later diffusion steps the correlation improves noticeably, and at late diffusion steps, the estimated rewards achieve a high correlation with the final reward of nearly perfect, confirming that as the diffusion progresses, our value estimation method accurately predicts the final reward.
- Figures 9 and 10 show the results using the Monte Carlo regression (MC). The correlation trends are generally similar with PM, while the MC estimation is more stable, especially in mid diffusion steps. The density of points along the red diagonal line suggests that the estimated values are well-calibrated. The correlation supports the effectiveness of using these value functions for intermediate state selection.

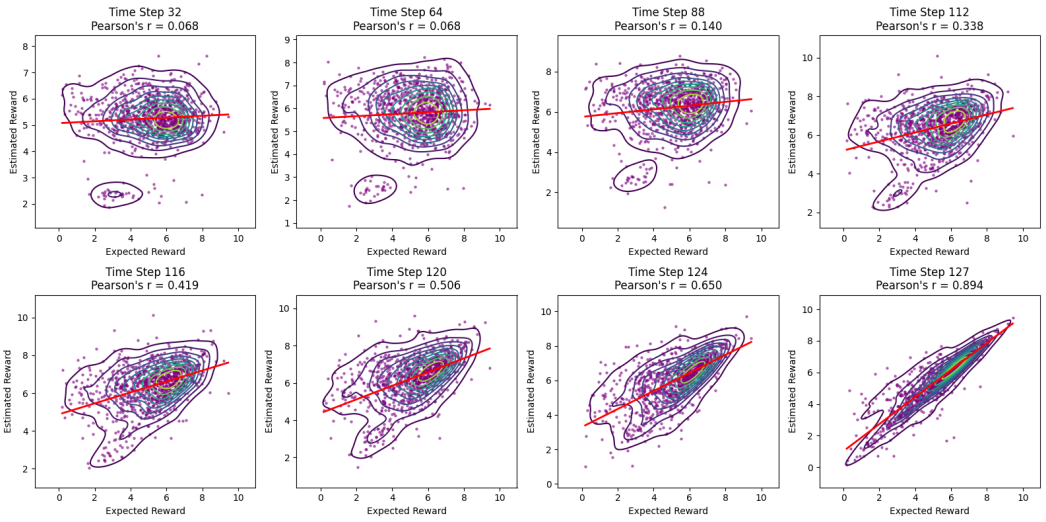


Figure 7: Scatter density plots between estimated reward (PM) and ground truth reward for DNA Enhancer task.

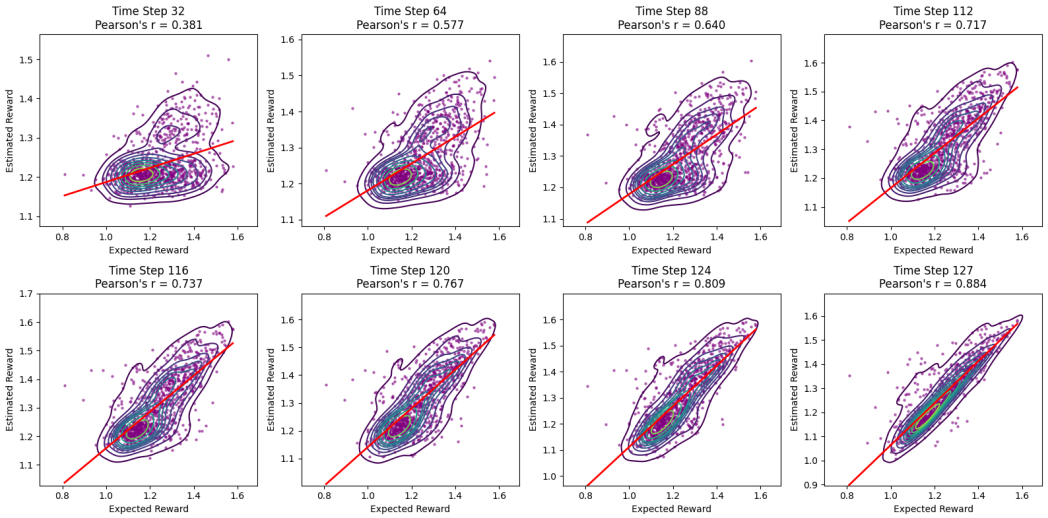


Figure 8: Scatter density plots between estimated reward (PM) and ground truth reward for 5'UTR MRL task.

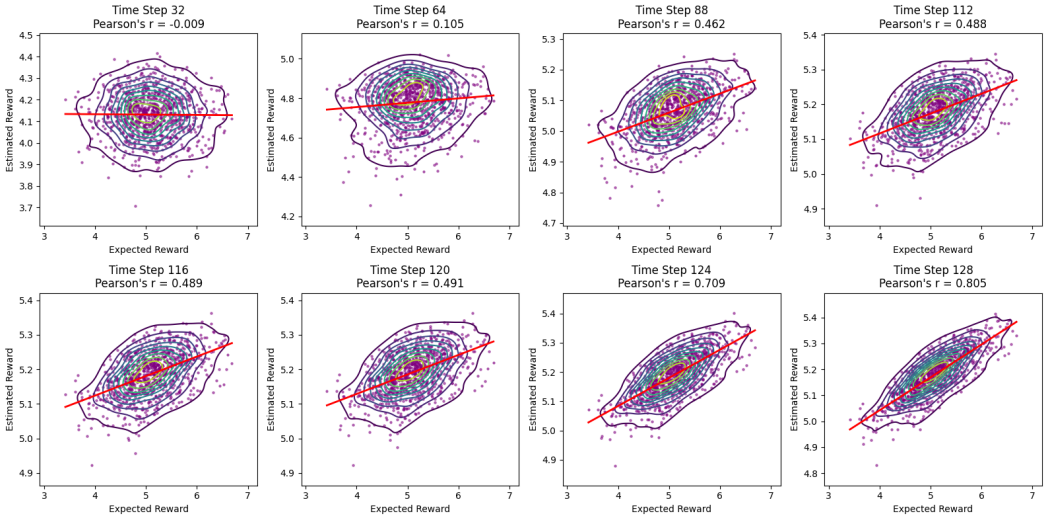


Figure 9: Scatter density plots between estimated reward (MC) and ground truth reward for DNA Enhancer task.

H.3 Validity Metrics for Molecule Generation

To evaluate the validity of our method in molecule generation, we report several key metrics that capture different aspects of molecule quality and diversity in Table 4 on page 32.

Table 4: Comparison of the generated molecules across various metrics. The best values for each metric are highlighted in **bold**.

Method	Valid↑	Unique↑	Novelty↑	FCD ↓	SNN ↑	Frag ↑	Scaf ↑	NSPDK MMD ↓	Mol Stable ↑	Atm Stable ↑
Pre-trained	**1.000**	**1.000**	**1.000**	12.979	0.414	0.513	**1.000**	**0.038**	0.320	**0.917**
DPS	**1.000**	**1.000**	**1.000**	13.230	0.389	0.388	**1.000**	0.040	0.310	0.878
SMC	**1.000**	0.406	**1.000**	22.710	0.225	0.068	**1.000**	0.285	0.000	**0.968**
SVDD-PM	**1.000**	**1.000**	**1.000**	**12.278**	**0.428**	**0.622**	**1.000**	0.052	**0.478**	0.9095
SVDD-MC	**1.000**	**1.000**	**1.000**	14.765	0.349	0.478	**1.000**	0.063	0.375	0.932

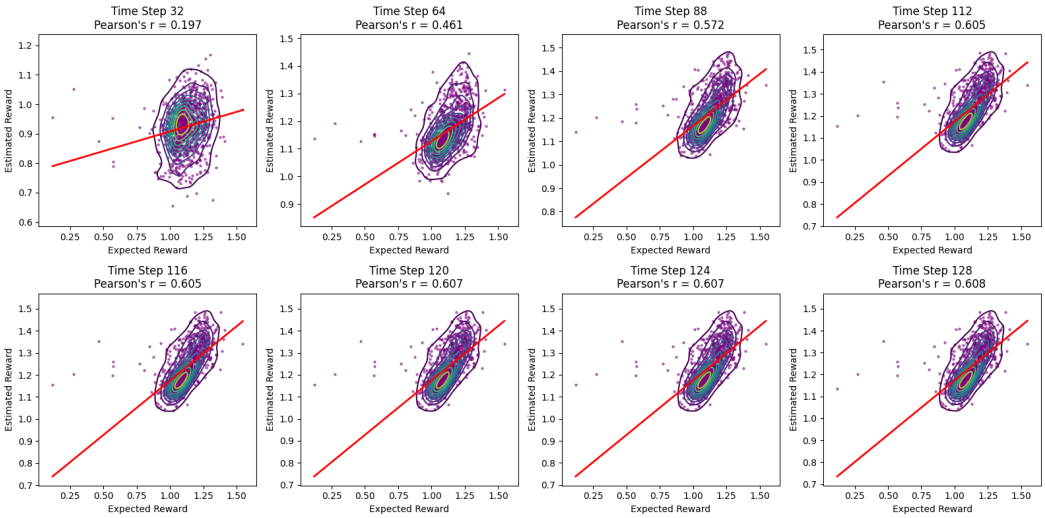


Figure 10: Scatter density plots between estimated reward (MC) and ground truth reward for 5'UTR MRL task.

The validity of a molecule indicates its adherence to chemical rules, defined by whether it can be successfully converted to SMILES strings by RDKit. Uniqueness refers to the proportion of generated molecules that are distinct by SMILES string. Novelty measures the percentage of the generated molecules that are not present in the training set. Fréchet ChemNet Distance (FCD) measures the similarity between the generated molecules and the test set. The Similarity to Nearest Neighbors (SNN) metric evaluates how similar the generated molecules are to their nearest neighbors in the test set. Fragment similarity measures the similarity of molecular fragments between generated molecules and the test set. Scaffold similarity assesses the resemblance of the molecular scaffolds in the generated set to those in the test set. The neighborhood subgraph pairwise distance kernel Maximum Mean Discrepancy (NSPDK MMD) quantifies the difference in the distribution of graph substructures between generated molecules and the test set considering node and edge features. Atom stability measures the percentage of atoms with correct bond valencies. Molecule stability measures the fraction of generated molecules that are chemically stable, *i.e.*, whose all atoms have correct bond valencies. Specifically, atom and molecule stability are calculated using conformers generated by RDKit and optimized with UFF (Universal Force Field) and MMFF (Merck Molecular Force Field).

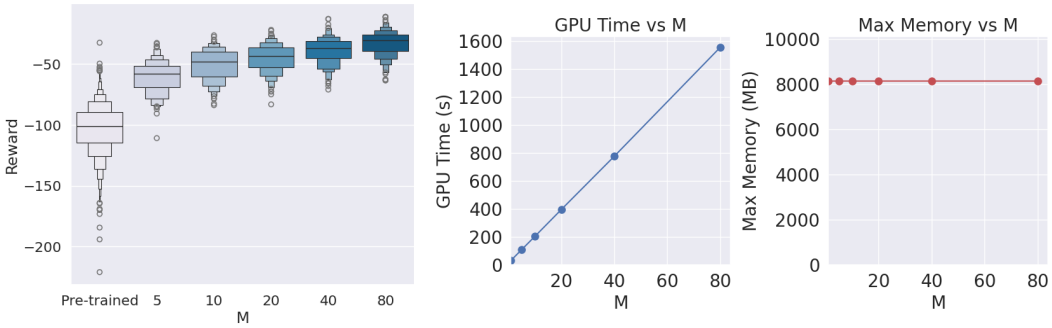
We compare the metrics using 512 molecules generated from the pre-trained GDSS model and from different methods optimizing QED, as shown in Table 4 on page 32. Overall, our method achieves comparable performances with the pre-trained model on all metrics, maintaining high validity, novelty, and uniqueness while outperforming on several metrics such as FCD, SNN, fragment similarity and molecule stability. Pre-trained performs consistently well across all metrics, particularly in SNN and atomic stability. However, it does not optimize specific molecular properties as effectively as the other methods. SMC performs poorly in fragment similarity, NSPDK MMD, and molecular stability, indicating that it generates unrealistic molecules. These results indicate that our approach can generate a diverse set of novel molecules that are chemically plausible and relevant.

H.4 Further Ablation Studies in terms of M

We provide several more ablation studies regarding M on top of the results in the main text, as plotted in Figure 11. The results are consistent with what we have observed in Figure 4.

H.5 Further Ablation Studies in terms of α

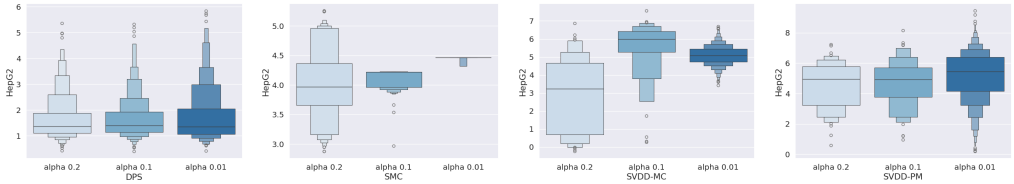
The hyperprameter α is a crucial hyperparameter that controls the balance between exploitation and exploration in state selection based on value estimation. A smaller α biases the selection towards higher-reward samples, whereas a larger α allows for a broader exploration of potential high-reward candidates. In the extreme case of $\alpha = 0$, only the highest-reward trajectory is selected



(a) Performance of **SVDD** as M varies.

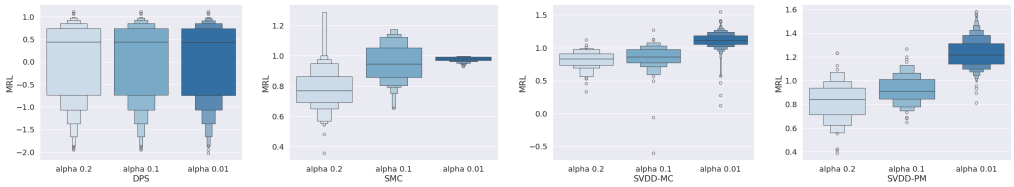
(b) GPU time and max memory of **SVDD** as M varies.

Figure 11: Abeltation Studies (for image generation while optimizing the compressibility)



(a) DPS (b) SMC (c) SVDD-MC (d) SVDD-PM

Figure 12: Reward performances of different methods as α varies for Enhancer design while optimizing cell-specificity.



(a) DPS (b) SMC (c) SVDD-MC (d) SVDD-PM

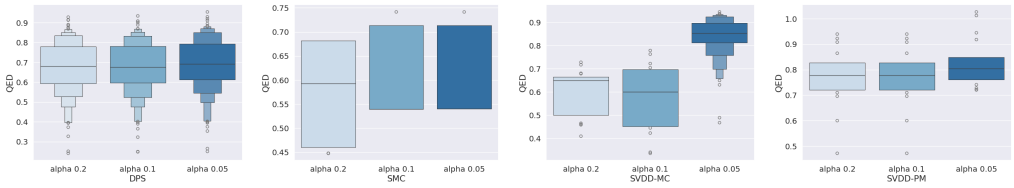
Figure 13: Reward performances of different methods as α varies for 5'UTR design while optimizing MRL.

deterministically. We analyze the effect of varying α across multiple tasks. We conduct experiments with multiple α values, ranging from a more explorative setting ($\alpha = 0.2$) to a more exploitative regime ($\alpha = 0.01$ for biological sequences and $\alpha = 0.05$ for molecules). Note that we study relatively small α value cases, since our major target is reward optimization. Results are shown in Figures 12, 13, 14, 15.

We observe that reducing α leads to an increase in the mean reward for all methods. This is expected, as smaller α prioritizes states with high estimated rewards. However, some variances of the reward distributions also increase, particularly for DPS and SVDD-PM, indicating that extreme exploitation results in more volatile sample quality. SMC struggles to maintain performance across all α values, indicating that its value estimation is less robust in this setting. Notably, SVDD-PM achieves relatively high rewards while maintaining consistency, highlighting the feature of a more balanced value function.

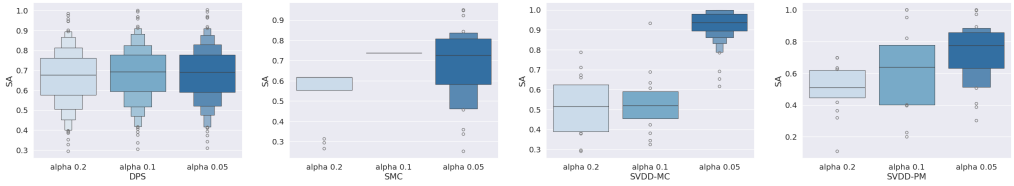
H.6 Further Visualization of Generated Samples

We provide additional generated samples in this section. Figure 16 and Figure 17 show comparisons of generated images from baseline methods and **SVDD** with different M values regarding compressibility and aesthetic score, respectively. Figure 18 and Figure 19 present the comparisons of visualized molecules generated from the baseline model and **SVDD** regarding QED and SA, respectively. The visualizations validate the strong performances of **SVDD**. Given that **SVDD** can achieve optimal SA for many molecules, we also visualize some molecules with optimal SA generated by **SVDD**, as



(a) DPS (b) SMC (c) SVDD-MC (d) SVDD-PM

Figure 14: Reward performances of different methods as α varies for molecule design while optimizing QED.



(a) DPS (b) SMC (c) SVDD-MC (d) SVDD-PM

Figure 15: Reward performances of different methods as α varies for molecule design while optimizing SA.

shown in Figure 20. In Figure 21, Figure 22, Figure 23, and Figure 24 we visualizes the docking of **SVDD**-generated molecular ligands to proteins parp1, 5ht1b, jak2, and braf, respectively. Docking scores presented above each column quantify the binding affinity of the ligand-protein interaction, while the figures include various representations and perspectives of the ligand-protein complexes.

We aim to provide a complete picture of how each ligand is situated within both the local binding environment and the larger structural framework of the protein. First rows show close-up views of the ligand bound to the protein surface, displaying the topography and electrostatic properties of the protein's binding pocket and providing insight into the complementarity between the ligand and the pocket's surface. The second rows display distant views of the protein using the surface representation, offering a broader perspective on the ligand's spatial orientation within the global protein structure. The third rows provide close-up views of the ligand interaction using a ribbon diagram, which represents the protein's secondary structure, such as alpha-helices and beta-sheets, to highlight the specific regions of the protein involved in binding. The fourth rows show distant views of the entire protein structure in the ribbon diagram, with ligands displayed within the context of the protein's full tertiary structure. Ligands generally fit snugly within the protein pocket, as evidenced by the close-up views in both the surface and ribbon diagrams, which show minimal steric clashes and strong surface complementarity.

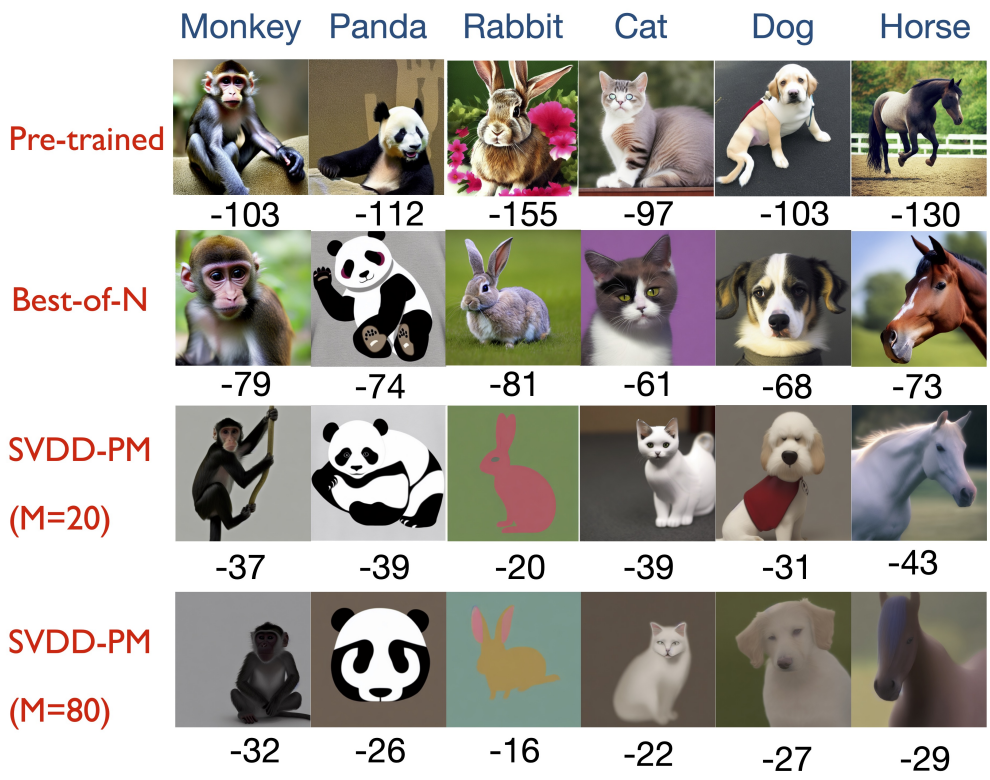


Figure 16: Additional generated samples (Domain: images, Reward: Compressibility)

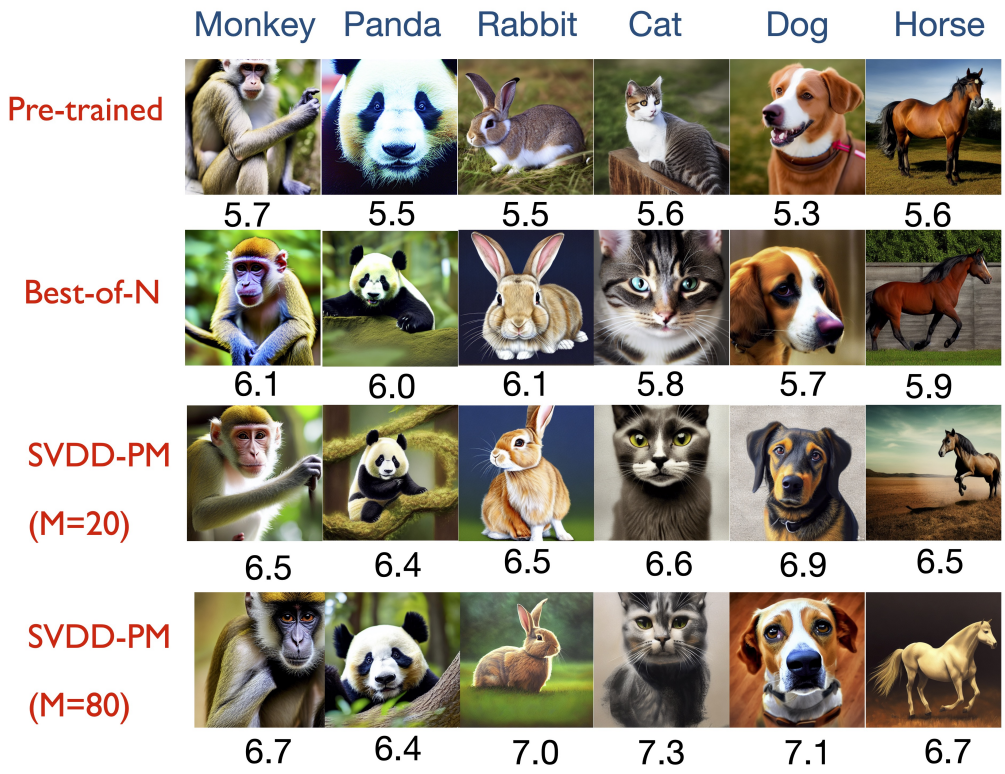


Figure 17: Additional generated samples (Domain: Images, Reward: Aesthetic score)

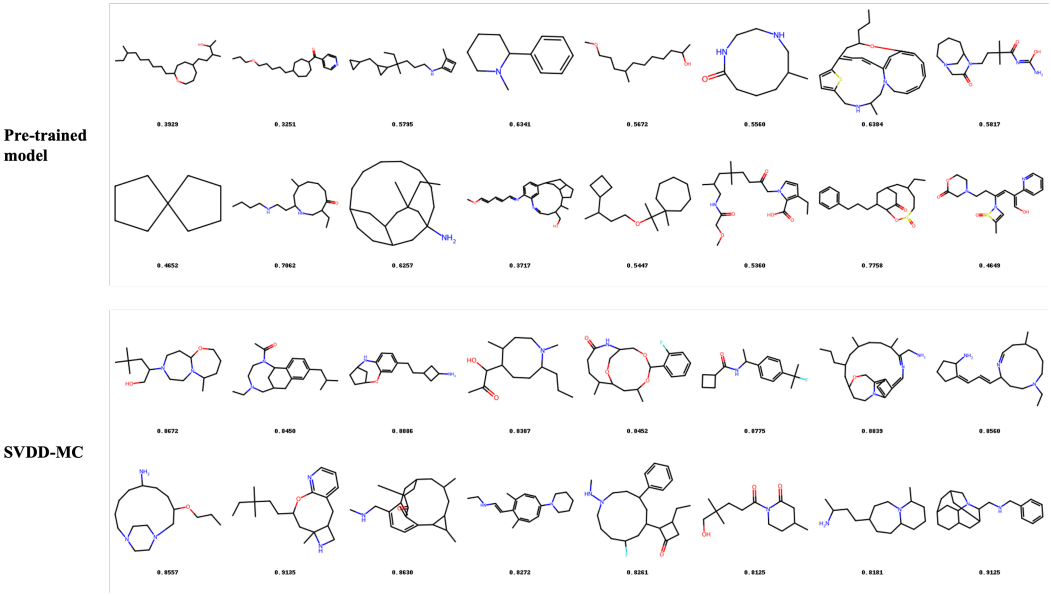


Figure 18: Additional generated samples (Domain: Molecules, Reward: QED score)

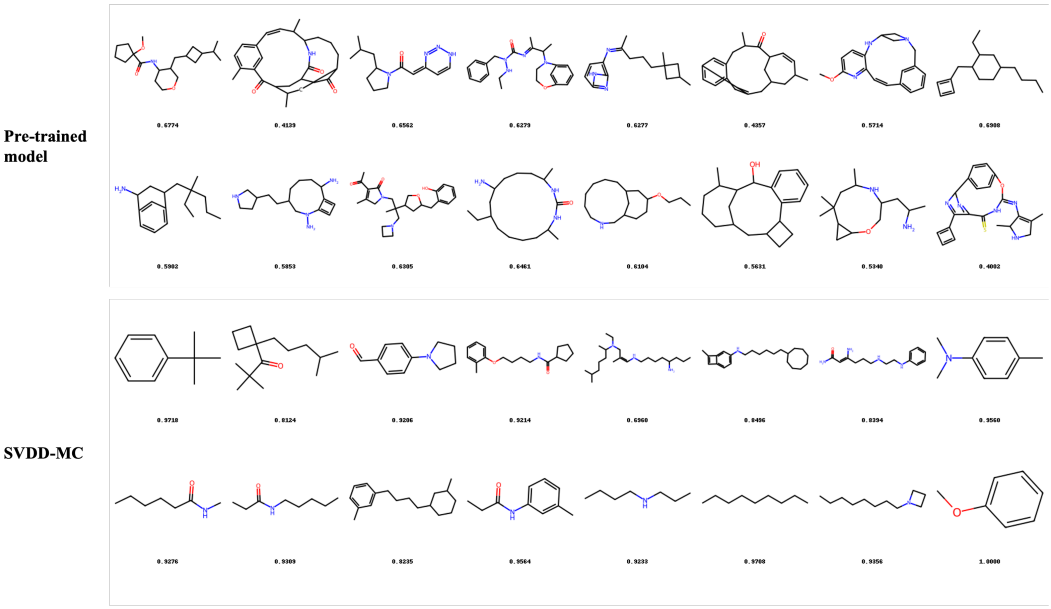


Figure 19: Additional generated samples (Domain: Molecules, Reward: SA score, normalized as $(10 - SA)/9$)

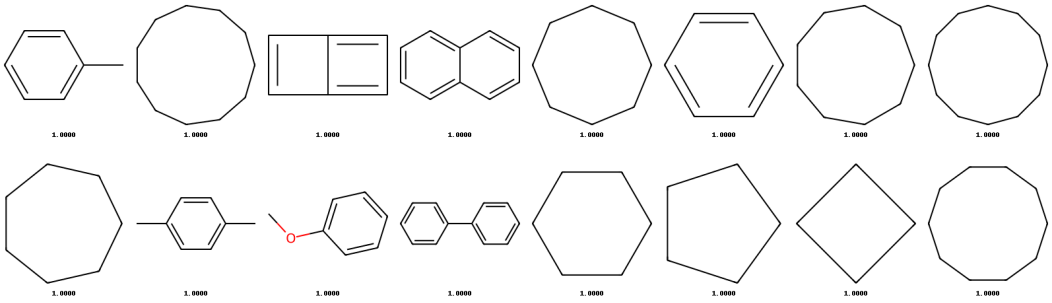


Figure 20: Additional generated samples from **SVDD**-MC (Domain: Molecules, Reward: SA score = 1.0 (normalized as $(10 - SA)/9$))

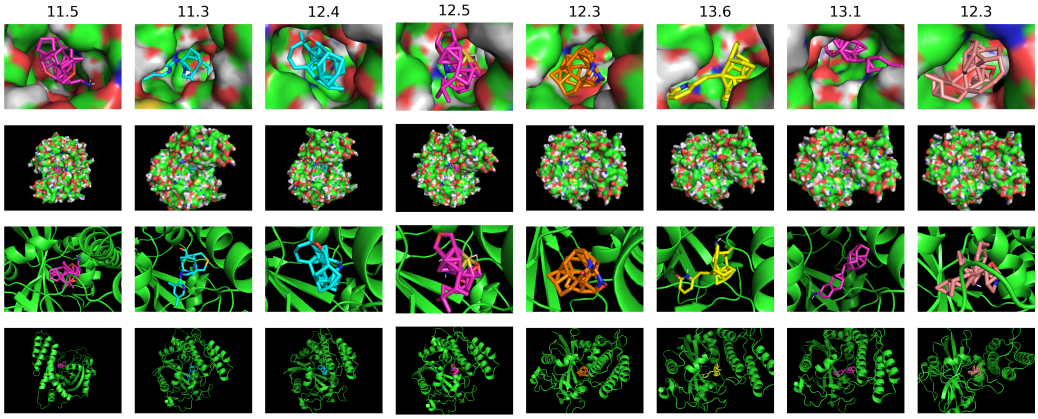


Figure 21: Additional generated samples from **SVDD** (Domain: Molecules, Reward: Docking score - parp1 (normalized as $max(-DS, 0)$))

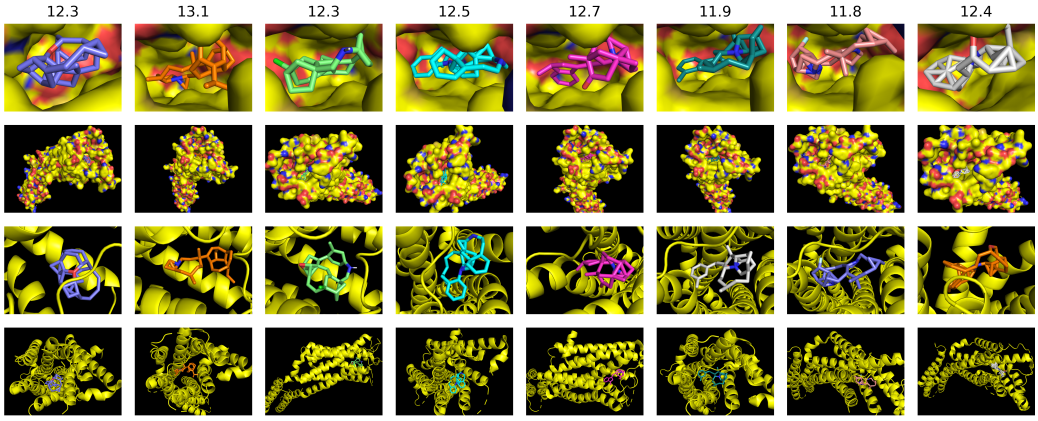


Figure 22: Additional generated samples from **SVDD** (Domain: Molecules, Reward: Docking score - 5ht1b (normalized as $max(-DS, 0)$))

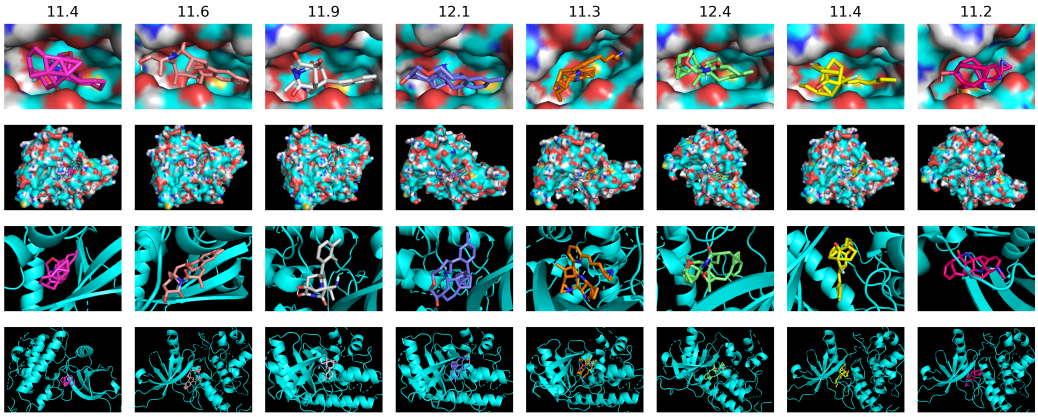


Figure 23: Additional generated samples from **SVDD** (Domain: Molecules, Reward: Docking score - jak2 (normalized as $max(-DS, 0)$))

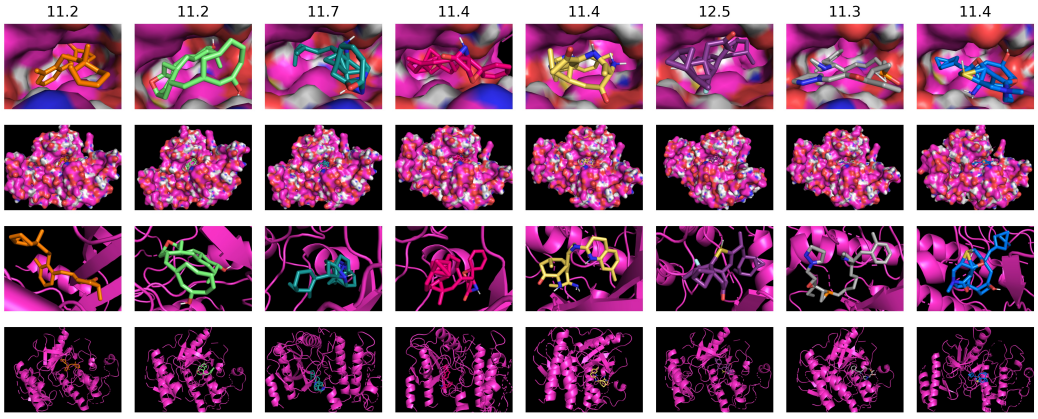


Figure 24: Additional generated samples from **SVDD** (Domain: Molecules, Reward: Docking score - braf (normalized as $max(-DS, 0)$))