Search-Augmented Masked Diffusion Models for Constrained Generation

Huu Binh Ta[*]
University of Virginia
nzj6jt@virginia.edu

Michael Cardei[*]
University of Virginia
ntr2rm@virginia.edu

Alvaro Velasquez
University of Colorado at Boulder
alvaro.velasquez@colorado.edu

Ferdinando Fioretto[†]
University of Virginia
fioretto@virginia.edu

Abstract

Discrete diffusion models generate sequences by iteratively denoising samples corrupted by categorical noise, and have been shown to be a powerful alternative to autoregressive decoding for structured and symbolic generation. However, standard training targets a likelihood-based objective that primarily matches the data distribution and provides no native mechanism for enforcing hard constraints or optimizing non-differentiable properties at inference time. This work addresses this limitation and introduces *SearchDiff*, a training-free neurosymbolic inference framework that integrates informed search directly into the reverse denoising process. At each denoising step, the model predictions define a proposal set that is optimized under a user-specified property satisfaction, yielding a modified reverse transition that steers sampling toward probable and feasible solutions. Experiments in biological design and symbolic reasoning illustrate that *SearchDiff* substantially improves constraint satisfaction and property adherence, while consistently outperforming discrete diffusion and autoregressive baselines.

1 Introduction

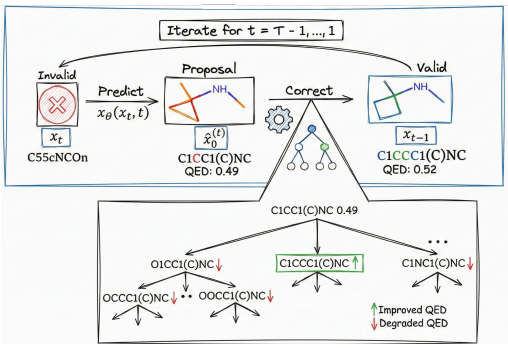

Figure 1: Illustration of the *SearchDiff* method on QED-optimized molecular generation.

Discrete diffusion has emerged as a powerful paradigm for generative modeling of discrete sequences and attracted significant interest due to their ability to model complex dependencies and global structure [1, 2]. Unlike autoregressive models, which are constrained by a fixed generation order, discrete diffusion treat sequence generation as a global denoising process, enabling joint modeling of long-range token dependencies. This property makes these models especially suitable for scientific and symbolic domains, in which target properties (e.g., drug-likeness, biological activity) and constraints (e.g., chemical validity, logical consistency) often depend on the entire sequence structure and can be assessed only when this global view is available [3, 4, 5]. However, while constraints are key in these domains, standard discrete

[*]Equal and main contribution.
[†]Contact author.

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

diffusion models are trained to optimize a likelihood-based objective that encourages matching the training data distribution, without a native mechanism for enforcing hard constraints or optimizing non-differentiable properties at inference time.

To mitigate this issue, recent work on controllable discrete diffusion incorporates guidance into the generation process. Nevertheless, common methods require additional training or rely on approximations that have been shown to introduce biases and degrade generation quality [4, 5, 6]. Moreover, fine-tuning a generative model for each new constraint fundamentally alters the original model and is impractical in settings where constraints vary across tasks or application contexts. These limitations highlight the *need for a training-free, inference-time mechanism that can enforce a diverse body of constraints while preserving the generative strengths of discrete diffusion models*.

This work addresses this need and introduces a novel integration of neurosymbolic search into discrete diffusion models. The proposed method, *Search-Augmented Masked Diffusion* (SearchDiff), builds on the insight that the denoising dynamics of masked diffusion expose a natural interface for sequence-level refinement. SearchDiff executes these refinements by embedding a local search step into each reverse denoising iteration, using the model predicted distribution to first generate high-quality proposals, and then navigate the discrete space toward feasible regions. This integration enables several key properties: *first*, it naturally exploits the non-determinism of parallel denoising with that of classical search methods, enabling global sequence-level refinements; *second*, it enables the use of non-differentiable constraints, allowing it to accommodate complex functions such as chemical simulators or symbolic solvers; third, being training-free, it directly leverages the model's learned inductive biases during the proposal and search process; and *finally,* by enforcing domain constraints at inference time, it allows the model to sample solutions that are feasible, but low-support under the base model. Figure 1 illustrates an example of this procedure steering the denoising trajectory towards maximizing QED (drug-likeness score) through iterative structural refinements.

Contributions. This paper makes three key contributions: **(1)** It introduces *SearchDiff*, a training-free inference method that embeds informed discrete search into each reverse denoising step of discrete diffusion to enforce sequence-level constraints. **(2)** It formalizes constrained discrete diffusion via constraint-violation functions and defines search-guided reverse transitions that incorporate constraint and property feedback while preserving the model's inductive biases. This enables non-differentiable constraints (e.g., simulators or symbolic solvers) to be enforced at inference time without model modification. **(3)** Finally, it evaluates the method on five tasks spanning biological sequence generation (molecules, peptides, and transfer RNA) and constrained symbolic reasoning (Sudoku and Boolean satisfiability), showing large and consistent gains in both constraint satisfaction and property adherence over prior methods. For molecular generation, SearchDiff achieves up to a fourfold increase in the fraction of samples satisfying Synthetic Accessibility relative to standard diffusion while also attaining the highest average drug-likeness; for Boolean SAT, it raises accuracy from 9.6% to 76.0% (nearly $8\times$) and outperforms strong diffusion and autoregressive baselines.

2 Related work

The need for controllable generation in discrete diffusion has led to the development of guidance methods to bias the learned distribution towards desired properties. For example, in the context of protein design, [7] introduces gradient-guided sampling by updating the denoiser's continuous hidden states and enabling gradient-based steering despite the non-differentiability of discrete tokens. [4] later provided a principled derivation of classifier-free and classifier-based guidance for discrete diffusion. Projected diffusion methods [5, 8] instead enforce feasibility by casting each denoising step as an optimization problem that projects the predicted categorical distribution onto a target set. However, these approaches typically rely on smooth gradient signals or additional training.

Because SearchDiff performs search at inference time, it is also related to search-based controllable decoding for autoregressive models. In the autoregressive setting, controllable decoding may also be approached as a search problem; for example, lexically constrained beam search enforces required words or phrases by tracking constraint coverage during expansion, as in Grid Beam Search [9] and Dynamic Beam Allocation [10]. For richer structure, NeuroLogic Decoding injects predicate-logic lexical constraints into beam search [11], and subsequent work adds lookahead heuristics to better satisfy logical and structural requirements [12]. Related approaches also reweight next-token scores using learned predictors to steer generation toward target attributes without modifying the base model

[13]. However, because these methods commit to a fixed left-to-right order, their search operates on partial sequences and must predict global satisfaction from incomplete prefixes, which can cause sequential error accumulation: early token choices may irreversibly steer decoding into regions where the final constraint is difficult, or impossible, to meet. Masked diffusion avoids this prefix limitation because each step operates on a specified full sequence. [14] and [15] rely on high-volume sampling and path-steering to select candidates based on predicted rewards of noisy intermediate states. However, this dependency on the pre-trained model's existing support limits their ability to explore rare regions, making the discovery of high-value candidates significantly less efficient.

In contrast, SearchDiff leverages the parallel denoising dynamics of masked diffusion to support global sequence refinement throughout the reverse process. The integration of discrete search within each denoising step is key to align sampling with user-defined constraints using black-box and non-differentiable constraint-violation functions, while preserving the diffusion model's learned proposal distribution and requiring neither additional training nor gradient access.

3 Preliminaries

Consider a sequence $\boldsymbol{x} = (x^1, \ldots, x^L)$ of L discrete tokens, each taking value from a vocabulary $\mathcal{V}$. Discrete diffusion models define a Markov noising process that progressively corrupts an initial sequence $\boldsymbol{x}_0 = \boldsymbol{x}$ into a simple prior distribution $\boldsymbol{x}_T$. This time-indexed transformation $\boldsymbol{x}_0 \to \boldsymbol{x}_T$ is called the *forward process* and is parameterized by a set of transition probabilities $q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})$, for $t = 1, \ldots, T$. Then a parametrized reverse process $p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)$ is learned to invert the corruption $\boldsymbol{x}_T \to \boldsymbol{x}_0$ and recover samples from the original data distribution.

In masked diffusion models [16, 1], the forward process consists of an absorbing noising process in which tokens are independently replaced by a special `[MASK]` token. Technically, the various sequences $\boldsymbol{x}_t$, for $t = 1, \ldots, T$, are latent variables and represented as categorical distributions over $\mathcal{V}$; when masked, the corresponding token is represented by a one-hot vector for the `[MASK]` token, denoted `M` in this paper. Formally, the forward noising distribution from $\boldsymbol{x}_0$ to $\boldsymbol{x}_T$ is defined as:

$$q(\boldsymbol{x}_t^i \mid \boldsymbol{x}_{t-1}^i) = \mathrm{Cat}\big(\boldsymbol{x}_t^i;\ \alpha_{t|t-1}\boldsymbol{x}_{t-1}^i + (1 - \alpha_{t|t-1})\mathtt{M}\big)\,, \tag{1}$$

for each position $i \in [L]$ of the sequence where $\alpha_{t|t-1} := \frac{\alpha_t}{\alpha_{t-1}}$. Therein, $\{\alpha_t\}_{t=1}^T$ is a monotonically decreasing noise schedule with $\alpha_0 = 1$ and $\alpha_T = 0$ and $\mathrm{Cat}(\cdot; \boldsymbol{p})$ denotes the categorical distribution with probability vector $\boldsymbol{p}$. Essentially, for each position i, if the token is not already masked at time $t-1$, it is preserved with probability $\alpha_{t|t-1}$ and replaced by `[MASK]` with probability $1 - \alpha_{t|t-1}$; once masked, it remains masked.

During generation, $\boldsymbol{x}_0$ is unknown, therefore, the reverse process is parameterized by a neural denoiser $\boldsymbol{x}_\theta(\boldsymbol{x}_t, t)$, typically implemented as a larger transformer network, that predicts a distribution over clean tokens, approximating $\boldsymbol{x}_0$. The reverse transitions from $\boldsymbol{x}_T \to \boldsymbol{x}_0$ are then given by:

$$p_\theta(x_{t-1}^i \,|\, \boldsymbol{x}_t) = \begin{cases} \mathrm{Cat}(x_{t-1}^i; x_t^i), & \text{if } x_t^i \neq \mathtt{M}, \\ \mathrm{Cat}\big(x_{t-1}^i; \gamma_t \mathtt{M} + \eta_t \boldsymbol{x}_\theta^i(x_t, t)\big)\,, & \text{if } x_t^i = \mathtt{M} \end{cases} \tag{2}$$

for all $t = T, \ldots, 1$ and positions $i \in [L]$. Therein, the coefficients $\gamma_t = \frac{1-\alpha_{t-1}}{1-\alpha_t}$ and $\eta_t = \frac{\alpha_{t-1}-\alpha_t}{1-\alpha_t}$ control the contribution of the denoiser prediction and the masking token in the reverse transition.

Once trained, sampling proceeds by initializing $\boldsymbol{x}_T$ as the fully masked sequence and iteratively denoises each step $T \to 0$ according to Equation (2). The final sample $\boldsymbol{x}_0$ is then returned as the generated sequence. The task of discrete diffusion models is to maximize fidelity w.r.t. the original data distribution $p_{\text{data}}(\boldsymbol{x}_0)$, but this does not, natively, enable a notion of control, which is central for scientific applications of interest to this paper. The next section formalizes the problem of constrained generation in discrete diffusion models.

4 Problem Formulation

The key desiderate of constrained generation if to produce sequences that lie in high-density regions of the data distribution and satisfy user-defined constraints $\mathcal{C}$. The idealized objective is:

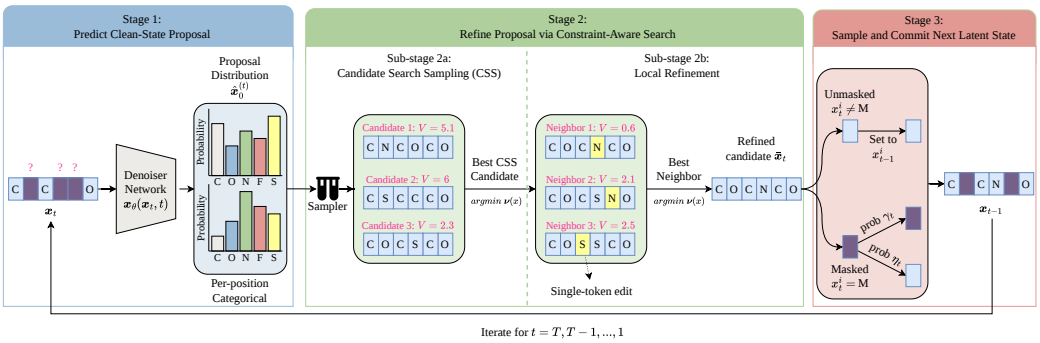

Figure 2: Illustration of a single SearchDiff denoising step: **(1)** The denoiser predicts a proposal distribution $\hat{\boldsymbol{x}}_0^{(t)}$. **(2)** The search operator refines this into a low-violation candidate $\bar{\boldsymbol{x}}_t$ via (2a) CSS and (2b) a local refinement step. **(3)** The modified reverse kernel used $\bar{\boldsymbol{x}}_t$ to deterministically update unmasked tokens and stochastically unmask a subset of masked tokens, yielding the next latent $\boldsymbol{x}_{t-1}$.

Given context $\boldsymbol{s}$, the sampler should produce samples $\boldsymbol{x}$ from the constrained target distribution:

$$p_\theta^{\mathcal{C}}(\boldsymbol{x}|\boldsymbol{s}) \propto p_\theta(\boldsymbol{x}|\boldsymbol{s})\mathbb{I}[\boldsymbol{x} \in \mathcal{C}], \tag{3}$$

where $p_\theta(\boldsymbol{x}|\boldsymbol{s})$ is the base discrete diffusion model's learned distribution and $\mathcal{C}$ is the feasible region induced by a set of constraints.

These constraints can encode both discrete logical requirements (e.g., structural validity) and quantitative conditions (e.g., property scores exceeding a threshold), and the setting assumes access to a violation vector that quantifies the extent to which a candidate sequence violates the constraints. Let $\nu : \mathcal{V}^L \to \mathbb{R}_+^K$ be a vector-valued violation map, where K is the number of constraints, $\nu_i(\boldsymbol{x})$ measures the violation of constraint i and $\nu_i(\boldsymbol{x}) = 0$ indicates satisfaction. A sequence is feasible if and only if it incurs zero violation across all constraints, that is, $\boldsymbol{x} \in \mathcal{C}$ if $\nu(\boldsymbol{x}) = \mathbf{0}$.

Next we devise a training-free control mechanism for masked diffusion that induces a constraint-aware distribution $p_\theta^{\mathcal{C}}$ satisfying two desiderata: **(1)** *generative fidelity* and **(2)** *constraint feasibility*.

5 SearchDiff

Method overview. SearchDiff augments masked discrete diffusion with an inference-time search step that reduces constraint violation. Starting from the fully masked sequence $\boldsymbol{x}_T = (\texttt{M}, \ldots, \texttt{M})$, the reverse chain iterates for $t = T, \ldots, 1$; at each step, given the current latent state $\boldsymbol{x}_t \in \Delta^{|\mathcal{V}| \times L}$, where Δ denotes the probability simplex, the method performs the following three operations:

1. **Clean-state proposal.** The denoiser produces a per-position categorical distribution over the clean sequence,
$$\hat{\boldsymbol{x}}_0^{(t)} \;=\; \boldsymbol{x}_\theta(\boldsymbol{x}_t, t) \in \Delta^{|\mathcal{V}| \times L}, \tag{4}$$
which serves as a proposal prior that is then used to generate clean sequences consistent with $\boldsymbol{x}_t$.
2. **Refine the proposal via constraint-aware search.** Using the denoiser output $\hat{\boldsymbol{x}}_0^{(t)}$ and the current diffusion context $(\boldsymbol{x}_t, t)$, SearchDiff invokes a discrete search operator that returns a refined discrete candidate $\bar{\boldsymbol{x}}_t$:
$$\bar{\boldsymbol{x}}_t \;\leftarrow\; \textsc{Search}(\hat{\boldsymbol{x}}_0^{(t)}, \boldsymbol{x}_t, t;\, \mathcal{C}). \tag{5}$$
The operator seeks to reduce constraint violation while remaining close to the original proposal. Importantly, it relies only on constraint evaluations in $\mathcal{C}$, without any gradient assumption.
3. **Sample and commit the next latent state.** Finally, it advances the diffusion chain using a modified reverse update: tokens that are already unmasked are set deterministically, while each currently masked position samples a binary unmasking decision. If a position is chosen to unmask, its value is set to the token proposed by $\bar{x}_t$; otherwise it remains `[MASK]`.

The full sampling trajectory can be summarized as:

$$\boldsymbol{x}_t \xrightarrow{\boldsymbol{x}_\theta(\cdot,t)} \hat{\boldsymbol{x}}_0^{(t)} \xrightarrow{\textsc{Search}(\cdot;\mathcal{C})} \bar{\boldsymbol{x}}_t \xrightarrow{\bar{p}_\theta(\boldsymbol{x}_{t-1}|\bar{\boldsymbol{x}}_t,x_t)} \boldsymbol{x}_{t-1}$$

for each $t = T, \ldots, 1$. A single search-augmented denoising step is illustrated in Figure 2.

5.1 Search-augmented denoising (Stage 2)

The denoiser $\boldsymbol{x}_\theta$ defines a proposal distribution over clean sequences at each reverse step t, but it does not, on its own, enforce user-defined constraints. SearchDiff addresses this by interleaving discrete search moves with the denoiser prediction and the reverse transition. At step t, the search has two goals: **(1)** to produce a discrete candidate $\bar{\boldsymbol{x}}_t$ with low constraint violation, and **(2)** to stay compatible with the denoiser's proposal so that the reverse transition follows the learned diffusion dynamics.

Search problem. At timestep t, SearchDiff defines a step-conditioned search problem

$$\mathcal{S}_t \;=\; \langle \mathcal{X}_t,\; \mathcal{A}_t,\; \mathrm{Succ}_t,\; \boldsymbol{\nu} \rangle,$$

where the diffusion context $(\boldsymbol{x}_t, t)$ and denoiser proposal $\hat{\boldsymbol{x}}_0^{(t)}$ are treated as fixed inputs. Therein, $\mathcal{X}_t$ is the state space of admissible candidates, $\mathcal{A}_t$ is the action space, Succ_t is the successor function, and $\boldsymbol{\nu} : \mathcal{V}^L \to \mathbb{R}_+^K$ is a vector-value violation map (as defined below). The output of the search is a discrete candidate $\bar{\boldsymbol{x}}_t \in \mathcal{V}^L$ that is used as input for the modified reverse kernel $\bar{p}_\theta$.

Search space. At reverse step t, the search operator optimizes over fully specified sequences $\boldsymbol{x} \in \mathcal{V}^L$, notably not incorporating the `[MASK]` token as unmasking is determined by the diffusion process rather than produced by search moves. The search is guided by the current diffusion context $(\boldsymbol{x}_t, t)$ and the denoiser proposal $\hat{\boldsymbol{x}}_0^{(t)}$, and returns a discrete candidate $\bar{\boldsymbol{x}}_t$ that reduces constraint violation while remaining compatible with the model's proposal distribution.

Actions and successor function. Actions correspond to single-token edits. Given a candidate $\boldsymbol{x} \in \mathcal{X}_t$, an action selects a position i and a token $v \in \mathcal{V}$ and assigns $x^i \leftarrow v$:

$$\mathcal{A}_t(\boldsymbol{x}) \;=\; \{(i, v) : v \in \mathcal{V},\; \mathrm{Succ}_t(\boldsymbol{x}, (i, v)) \in \mathcal{X}_t\}. \tag{6}$$

where the successor function applies the edit and returns the resulting candidate,

$$\mathrm{Succ}_t(\boldsymbol{x}, (i, v)) \;=\; \boldsymbol{x}' \quad \text{with} \quad x'^i = v,\;\; x'^j = x^j \;\forall j \neq i,$$

and rejects edits that violate admissibility.

Objective function. Consider a constraint set $\mathcal{C} = \{C_1, \ldots, C_K\}$; SearchDiff evaluates constraints through continuous violation functions ν_k, each associated to a constraint C_k, as introduced in Section 4. These violations define a vector-valued map:

$$\boldsymbol{\nu}(\boldsymbol{x}) = (\nu_1(\boldsymbol{x}), \ldots, \nu_K(\boldsymbol{x})) \in \mathbb{R}_+^K,$$

with $\boldsymbol{\nu}(\boldsymbol{x}) = \mathbf{0}$ iff $\boldsymbol{x} \in \mathcal{C}$. To compare candidates, the search operator is equipped with a total preorder $\preceq_{\mathcal{C}}$ on $\mathbb{R}_+^K$ that encodes how violations across constraints should be prioritized. The search step then computes

$$\bar{\boldsymbol{x}}_t \in \operatorname*{arg\,min}_{\boldsymbol{x} \in \mathcal{X}_t} \preceq_{\mathcal{C}} \boldsymbol{\nu}(\boldsymbol{x}), \tag{7}$$

where $\arg\min_{\preceq_{\mathcal{C}}}$ denotes any minimizer under $\preceq_{\mathcal{C}}$. A canonical instantiation of $\preceq_{\mathcal{C}}$ is a lexicographyic order. The consistency requirement is enforced by the admissible set $\mathcal{X}_t$ and the denoiser output $\hat{\boldsymbol{x}}_0^{(t)}$ is used as a proposal prior that biases the search toward candidates that remain plausible under the learned reverse dynamics.

5.2 SEARCH Initialization and local refinement

We instantiate SEARCH as a two-part procedure:

1. Obtain an initial candidate from the denoiser proposal through *Candidate Search Sampling* (CSS).
2. Apply local search refinements that decrease the violation function $\boldsymbol{\nu}(\cdot)$.

Candidate Search Sampling. Note that, the inputs to the search operator include the denoiser output $\hat{\boldsymbol{x}}_0^{(t)}$, which defines a distribution over clean sequences at step t. We leverage this distribution to

construct a strong initial candidate via *Candidate Search Sampling* (CSS). Let $p_t(\cdot \mid \hat{\boldsymbol{x}}_0^{(t)}, \boldsymbol{x}_t)$ be the proposal distribution over discrete candidates obtained by sampling tokens from $\hat{\boldsymbol{x}}_0^{(t)}$. Given a sampling budget M, CSS draws $\boldsymbol{x}_t^{(1)}, \ldots, \boldsymbol{x}_t^{(M)} \sim p_t(\cdot \mid \hat{\boldsymbol{x}}_0^{(t)}, \boldsymbol{x}_t)$, and selects the least-violating sample:

$$\boldsymbol{x}_t^{\text{css}} \in \underset{j \in \{1,\ldots,M\}}{\arg\min} \preceq_{\mathcal{C}} \boldsymbol{\nu}(\boldsymbol{x}_t^{(j)}). \tag{8}$$

The selected $\boldsymbol{x}_t^{\text{css}}$ is then used as the starting point for the search refinement; In Section 6 ablative results over CSS and refinement steps will illustrate their effects in isolation.

Local search refinement. Starting from $\boldsymbol{x}_t^{\text{css}}$, the search process proceeds by performing iterative improvements. To ensure the problem complexity remains manageable, the process is instantiated using local search over single-token edits defined by the admissible action set (6). Let the step-conditioned Hamming-1 neighborhood be defined as

$$\mathcal{N}_t(\boldsymbol{x}) \;:=\; \Big\{ \mathrm{Succ}_t(\boldsymbol{x}, (i, v)) : \; (i, v) \in \mathcal{A}_t(\boldsymbol{x}) \Big\}. \tag{9}$$

The search operator then selects the best neighbor: $\bar{\boldsymbol{x}}_t \in \arg\min_{\boldsymbol{x}' \in \mathcal{N}_t(\boldsymbol{x}^{(r)})} \preceq_{\mathcal{C}} \boldsymbol{\nu}(\boldsymbol{x}')$.

5.3 Modified reverse kernel (Stage 3)

Given the search-refined discrete proposal $\bar{\boldsymbol{x}}_t$, we advance the diffusion chain using a search-guided reverse update by sampling $\boldsymbol{x}_{t-1}$. This operates by stochastically unmasking positions. Concretely, the per-position reverse transition is

$$\bar{p}_\theta(x_{t-1}^i \mid \bar{\boldsymbol{x}}_t, \boldsymbol{x}_t) = \begin{cases} \mathrm{Cat}(x_{t-1}^i; \bar{x}_t^i), & \text{if } x_t^i \neq \mathtt{M}, \\ \mathrm{Cat}\Big(x_{t-1}^i; \gamma_t \mathtt{M} + \eta_t \, \delta_{\bar{x}_t^i}\Big), & \text{if } x_t^i = \mathtt{M}, \end{cases} \tag{10}$$

where $\delta_{\bar{x}_t^i}$ is a one-hot distribution that assigns probability 1 to the token chosen by search at position i. A pseudocode summary of SearchDiff is provided in Algorithm 1 (Appendix A).

6 Experiment

This section evaluates the performance of SearchDiff across a range of experimental settings, in the domains of biological sequence generation and symbolic reasoning SearchDiff is compared against state-of-the-art diffusion and autoregressive models of similar size.

6.1 Biological Sequence Generation

This section considers three inverse design biological sequence tasks: molecules, peptides, and transfer ribonucleic acid (tRNA). They share a common sequence-generation setup but differ in constraint structure and strictness. We compare SearchDiff against standard MDLM [1]; guidance methods (CBG, CFG) for MDLM and UDLM as proposed by [4]; autoregressive approaches (FUDGE [13], PPLM [17], and CFG [18]); CDD [5]; an adapted TreeG-SC [15] and a Monte-Carlo-based method named MCTG [19] implemented within the MDLM framework. Following prior work [4], we report mean and variance over five runs, each generating 1024 sequences.

Table 1: Performance of different approaches for QED maximization task in molecular generation.

Approach	Admissible Mol [↑]	Mean QED [↑]
AR FUDGE	53.0 ± 3.5	0.61 ± 0.00
AR PPLM	142.0 ± 14.2	0.45 ± 0.00
AR CFG	79.4 ± 6.4	0.60 ± 0.00
MDLM	397.8 ± 12.9	0.48 ± 0.01
MDLM CBG	116.6 ± 8.9	0.58 ± 0.00
UDLM CBG	63.8 ± 8.1	0.61 ± 0.00
MDLM CFG	95.8 ± 9.0	0.60 ± 0.01
UDLM CFG	64.0 ± 5.1	0.62 ± 0.00
MCTG	597.4 ± 12.4	0.62 ± 0.00
SearchDiff$_{\text{max. QED}}$	497.6 ± 4.5	$\mathbf{0.77 \pm 0.00}$
SearchDiff$_{\text{QED}\geq 0.7}$	$\mathbf{833.4 \pm 10.6}$	0.72 ± 0.00

6.1.1 Small Molecules

Settings. Generating small molecules that are both chemically plausible and exhibit desirable properties is central to early-stage drug discovery, where invalid structures or hard-to-synthesize candidates can invalidate otherwise promising leads. We study constrained generation of SMILES strings under two property constraints: synthetic accessibility (SA), measuring ease of synthesis, and

drug-likeness (QED), measuring agreement with common physicochemical profiles of successful drugs. In addition, we report standard quality metrics for *validity*, *novelty* (relative to the training set), and *uniqueness* within the generated batch. Both SA and QED are computed using non-differentiable black-box functions from `RDKit`[3] (see Appendix C.1.1). Following the setup in prior work [4, 5], we use a 92.4M-parameter Transformer diffusion model trained on QM9 [20]; at inference, generation starts from an all-`[MASK]` sequence and runs 32 denoising steps to full unmasking, after which special tokens are replaced to obtain clean SMILES strings.

Results. Table 1 reports the number of admissible molecules, defined as valid, novel, and unique molecules, and the mean QED under QED maximization. SearchDiff produces 497.6 ± 4.5 admissible molecules, which is more than 3.5 times higher than the best-performing AR baseline (AR PPLM at 142.0 ± 14.2) and nearly 5 times higher than vanilla MDLM. It also achieves the highest QED, with mean 0.77 versus 0.62 for the next best method (UDLM CFG). Since greedy QED maximization in SearchDiff leads to repeated samples, when setting the QED constraint to ≥ 0.7, SearchDiff discovers even more novel valid molecules while still outperforming all baselines in terms of mean QED. Notably, TreeG-SC struggles to balance validity and the SA constraint (Table 2). In contrast, SearchDiff preserves both: it produces 474.2 admissible molecules, with 99% of them meeting the strict SA ≤ 4 threshold, while maintaining a strong mean QED of 0.73, higher than all baselines.

Table 2: Performance of different approaches on molecular generation under (SA ≤ 4, QED maximization) constraint for molecular generation. CDD results reported from [5].

Approach	Admissible Molecules [↑]	SA ≤ 4 [↑]	Mean QED [↑]
MDLM	397.8 ± 12.9	119.2 ± 8.4	0.48 ± 0.01
CDD	31	31	0.61
TreeG-SC	190.6 ± 16.8	150.2 ± 17.0	0.48 ± 0.00
SearchDiff	$\mathbf{474.2 \pm 8.7}$	$\mathbf{470.2 \pm 7.8}$	$\mathbf{0.73 \pm 0.00}$

SearchDiff furthermore demonstrates strong out-of-distribution (OOD) discovery results which is quantified by the number of novel molecules whose QED exceeds the maximum QED in the training set. As shown in Figure 3, LS is once again the key driver: note how standard decoding and the CBG produce no OOD candidates (0.0 ± 0.0), and TreeG-SC remains negligible (0.6 ± 0.5), whereas LS alone generates 212.6 ± 11.7 OOD molecules by applying local neighbor edits rather than following the model's global positional statistics. This refinement remains compatible with CSS: *the combined LS+CSS variant reaches an average of* 332.4 *molecules with higher QED than the training dataset maximum value*; thus, LS is crucial for exploring chemical regions not observed during training, or generating OOD samples.

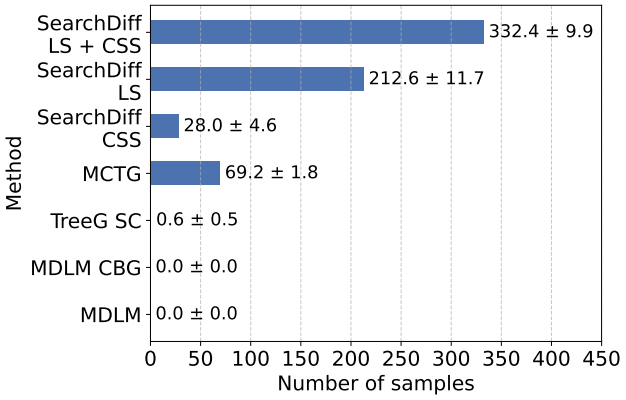

Figure 3: Number of generated molecules with higher QED compared to the highest QED value in the training dataset for different methods.

6.1.2 Peptides

Settings. Peptide generation [21] aims to design short amino-acid sequences with properties that make them viable candidates for downstream synthesis and biological activity, where even small violations of biophysical requirements can render a sequence ineffective. This setting imposes three constraints for antimicrobial and host-defense peptides [22]: sequence length between 10 and 50 amino acids, net positive charge in $[+2, +9]$ to promote interaction with negatively charged membranes, and hydrophobic residue fraction at least 30% to support membrane insertion. These properties are calculated using non-differentiable black-box functions provided by `Biopython`[4] library (see Appendix C.1.2 for details). The base model uses the same architecture as in the molecular setting [4]; generation runs for 32 denoising steps with a maximum sequence length of 50.

[3] `https://www.rdkit.org/docs/api-docs.html`
[4] `https://biopython.org/docs/1.76/api/Bio.SeqUtils.ProtParam.html`

Table 3: Performance on peptide and tRNA generation (unique, novel molecules under constraints): **Peptide**: constraints C1: $10 \leq$ length ≤ 50; C2: hydrophobic ratio $\geq 30\%$; C3: $2 \leq$ net charge ≤ 9; **tRNA**: C1: $7 \leq$ acceptor stem length ≤ 9; C2: TΨC loop; C3: D loop features a 4 to 6 base-pair stem, 5-base-pair stem enclosing the anticodon triplet, variable loop varies widely in length (3 to 21 bases).

Approach	**All constraints** [↑]	**Novel & unique** [↑]	**C1** $10 \leq$ Len ≤ 50 [↑]	**C2** Hydrophobic $\geq 30\%$ [↑]	**C3** $2 \leq$ Charge ≤ 9 [↑]
Peptide					
MDLM	530.8 ± 9.9	679.2 ± 8.2	742.8 ± 7.2	549.0 ± 16.3	624.4 ± 9.4
MDLM CBG	471.6 ± 19.6	934.0 ± 6.4	712.0 ± 18.7	806.8 ± 5.6	572.2 ± 16.6
UDLM CBG	625.6 ± 12.6	795.4 ± 17.4	726.4 ± 12.2	743.8 ± 14.8	685.4 ± 16.5
TreeG-SC	867.2 ± 4.5	979.2 ± 4.2	979.2 ± 4.2	977.0 ± 4.8	867.6 ± 4.2
SearchDiff	$\mathbf{1024 \pm 0.0}$	$\mathbf{1024 \pm 0.0}$	$\mathbf{1024 \pm 0.0}$	$\mathbf{1024 \pm 0.0}$	$\mathbf{1024 \pm 0.0}$
Approach	**All constraints** [↑]	**Novel & unique** [↑]	**C1** Acceptor stem [↑]	**C2** TΨC loop [↑]	**C3** D/anticodon/variable [↑]
tRNA					
MDLM	25.4 ± 5.5	$\mathbf{1005.2 \pm 4.1}$	445.4 ± 16.0	807.6 ± 10.1	46.2 ± 4.7
MDLM CBG	17.2 ± 4.8	942.0 ± 4.7	374.8 ± 13.1	744.8 ± 12.5	31.4 ± 6.5
UDLM CBG	20.8 ± 4.3	252.4 ± 11.8	199.4 ± 14.4	225.0 ± 8.6	34.6 ± 3.2
TreeG-SC	25.8 ± 1.4	730.4 ± 11.2	599.4 ± 9.9	694.2 ± 11.5	31.4 ± 2.2
SearchDiff	$\mathbf{719.0 \pm 105.1}$	966.6 ± 46.2	$\mathbf{964.6 \pm 45.8}$	$\mathbf{955.8 \pm 51.4}$	$\mathbf{726.0 \pm 102.3}$

Results. As shown in Table 3, SearchDiff attains a 0% constraint violation rate, generating 1024 peptides that are simultaneously valid, unique, novel, and satisfy *all peptide constraints*. This is in direct contrast to state of the art baselines such as TreeG-SC and UDLM CBG, which may perform well on individual criteria, but do not reliably satisfy the full constraint set. SearchDiff's strong performance across metrics follows from directly optimizing a black-box, gradient-free constraint evaluator through discrete edits during denoising, whereas gradient-based guidance relies on differentiable surrogates and vanilla MDLM sampling has no mechanism to repair violations.

6.1.3 Transfer ribonucleic acid

Settings. Finally, we study structure-aware design for transfer RNA (tRNA). tRNAs are central adaptor molecules in translation function depends on conserved motifs and a characteristic stem–loop secondary structure. The task considers constraints that encode the core tRNA components: an acceptor stem with a conserved CCA tail [23, 24], the D loop [23], the anticodon stem–loop responsible for codon recognition [23], the TΨC loop containing the conserved TΨC motif [25], and the variable loop whose length distinguishes different tRNA classes [26, 27]. These properties are calculated using non-differentiable black-box functions provided by `ViennaRNA`[5] (see Appendix C.1.3 for details). Importantly, these structural constraints ensure that generated sequences remain biologically plausible. We train the same base model as in the molecular setting [4] and generate with 32 denoising steps and maximum length 110. However, compared to peptides and molecules, tRNA generation poses a substantially greater challenge due to the larger number of tokens that must be predicted at each denoising step under tightly coupled structural constraints.

Results. Table 3 tabulates the results for all models. Note that the base model MDLM performs rather poorly on this task while other baselines offer only limited gains. In contrast, SearchDiff yields a substantial improvement in joint feasibility achieving a $27\times$ increase in the "all constraints" metric relative to the best-performing baseline. The dominant failure mode across methods is constraint **C3**, which requires satisfying coupled structural features such as specific base-paired stems and variable-loop length, and thus depends on long-range dependencies that are both rare (OOD) and sharply defined. SearchDiff instead attains 726.0 on this constraint. Observing the produced traces we notice that local search consistently steers intermediate denoising

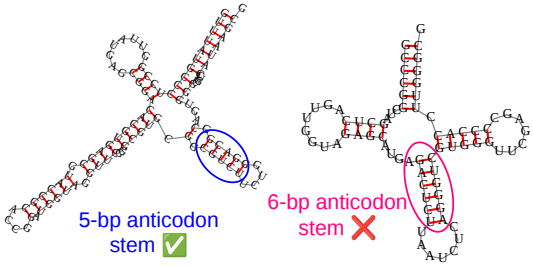

Figure 4: tRNAs generated by SearchDiff (left) and MDLM (right).

[5] `https://www.tbi.univie.ac.at/RNA`

states toward sequences whose edits repair structural violations and thus recover feasible folding configurations in the high-dimensional discrete space. An example of a valid tRNA generated by SearchDiff and an invalid tRNA generated by MDLM is shown in Figure 4.

6.2 Symbolic Reasoning Tasks

Settings. Next, this section evaluates SearchDiff on two symbolic logic benchmarks, Sudoku and Boolean satisfiability, to assess constrained generation in regimes where feasibility is binary and constraints are global, non-differentiable, and tightly coupled. In both tasks, a single token error can simultaneously violate multiple invariants, so standard stochastic unmasking often fails to reach a feasible region. The settings use the same masked discrete diffusion model (MDM) backbone for both problems, with a GPT-2 style Transformer denoiser (causal mask removed for bidirectional conditioning) trained with the standard masked diffusion objective [28]. For Sudoku, we represent a 9×9 board as a length-81 token sequence and format each instance as `PUZZLE[SEP]SOLUTION`, where `PUZZLE` fixes the givens and `SOLUTION` is generated by initializing target positions to `[MASK]` and denoising for T steps. For Boolean SAT, we consider random 3-SAT with $n = 7$ variables, each with 45 clauses, and format each instance as `FORMULA[SEP]ASSIGNMENT`, where the model generates the length-n assignment conditioned on the formula [28]. In both cases, SearchDiff uses a black-box violation evaluator to guide discrete local edits: for Sudoku, it counts duplicates across rows, columns, and 3×3 subgrids; for SAT, it counts unsatisfied clauses. The neighborhood is defined by single-token replacements (cell value changes for Sudoku; truth-value flips for SAT), holding the conditioning prefix fixed. Additional experimental details are available in Appendix C.2.

Results. Table 4 illustrates accuracy for Sudoku (left) and Boolean SAT (right). For the Sudoku task, SearchDiff improves over vanilla MDM decoding (96.5% vs. 93.2%), while applying search only at the final step results in a weaker performance (92.2%) than the vanilla MDM, indicating that incorporating search throughout the denoising trajectory is preferable to post-hoc repair.

Table 4: Accuracy (%) across Sudoku (left) and Boolean SAT (right) with 7 variables. For MDM, we report Vanilla sampling, SearchDiff, and search only on the last denoising step.

Model	**Size**	**Accuracy (%) [↑]**	
		Sudoku	**Boolean SAT**
Random Init + H1-LS	NA	0	45.8
AR (GPT-2)	6M	31.25	86.13
AR (GPT-2)	85M	26.56	93.97
AR (GPT-2)	303M	23.21	**94.42**
MDM (Vanilla)	6M	93.2	9.6
MDM (Last_Step)	6M	92.2	11.4
MDM (SearchDiff)	6M	**96.5**	76.0

The Boolean SAT application is particularly revealing as the generation target is an assignment that must satisfy a set of coupled, global constraints, and there is essentially no notion of "almost correct": flipping a single variable can simultaneously break many clauses, while the model receives no incremental signal about which bit caused the failure. In other words, the feasible set is a sparse, combinatorial subset of $\{0, 1\}^n$ and satisfaction is a discontinuous predicate, so stochastic unmasking can easily drift into large infeasible regions and has little structure to follow back toward feasibility. This makes the gap substantially larger: vanilla MDM achieves only 9.6% accuracy, while SearchDiff reaches 76.0% (a $\sim 7.9\times$ improvement), with last-step-only search again providing only marginal gains (11.4%). Notably, SearchDiff attains these results with a 6M-parameter diffusion model; larger GPT-2 autoregressive baselines do not close the gap on Sudoku and, more importantly, do not provide a comparable mechanism for enforcing hard constraints during sampling. We further consider a baseline that starts from a random initialization and applies only greedy Hamming-1 local search, without using the MDM denoiser. The results indicate that local search alone does not account for the observed gains. Instead, the learned MDM proposal acts as an informative prior, improving both the quality of the initial states and the subsequent denoising trajectory. Additional results across denoising step counts are reported in Appendix B.2.

7 Conclusion

This work introduced SearchDiff, a training-free, search-augmented framework for precise constrained generation with discrete diffusion models. The proposed framework interleaves candidate search sampling and iterative local refinement directly into the reverse denoising process of masked

diffusion models. This results in a process that steers the pre-trained model distribution toward feasibility on the discrete constraint manifold, without incurring the overhead of additional training. The performance of SearchDiff was examined across a range of scientific and reasoning benchmarks, spanning biological sequence design and symbolic reasoning. Across all experiments, SearchDiff consistently improves constraint satisfaction and data faithfulness, while remaining computationally practical for discovery workflows. More broadly, these results suggest that trajectory-level, inference-time optimization may be a principled route to making discrete generative models more reliable and controllable under global, non-differentiable constraints in both scientific and reasoning tasks.

Acknowledgments and Disclosure of Funding

This work was partially supported by NSF grants CAREER-2401285, RI-2533631, RI-2334936, the LaCross Institute Fellowship and DARPA under Contract No. #HR0011252E005. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No 2234693. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The authors also acknowledge the Research Computing at the University of Virginia. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or DARPA.

The authors declare no competing interests.

References

[1] Subham Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

[2] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces, 2023.

[3] Seul Lee, Karsten Kreis, Srimukh Prasad Veccham, Meng Liu, Danny Reidenbach, Yuxing Peng, Saee Paliwal, Weili Nie, and Arash Vahdat. Genmol: A drug discovery generalist with discrete diffusion. *arXiv preprint arXiv:2501.06158*, 2025.

[4] Yair Schiff, Subham Sekhar Sahoo, Hao Phung, Guanghan Wang, Sam Boshar, Hugo Dallatorre, Bernardo P de Almeida, Alexander Rush, Thomas Pierrot, and Volodymyr Kuleshov. Simple guidance mechanisms for discrete diffusion models. *arXiv preprint arXiv:2412.10193*, 2024.

[5] Michael Cardei, Jacob K Christopher, Thomas Hartvigsen, Brian R Bartoldson, Bhavya Kailkhura, and Ferdinando Fioretto. Constrained discrete diffusion. *arXiv preprint arXiv:2503.09790*, 2025.

[6] Hunter Nisonoff, Junhao Xiong, Stephan Allenspach, and Jennifer Listgarten. Unlocking guidance for discrete state-space diffusion and flow models. *arXiv preprint arXiv:2406.01572*, 2024.

[7] Nate Gruver, Samuel Stanton, Nathan C. Frey, Tim G. J. Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew Gordon Wilson. Protein design with guided discrete diffusion, 2023.

[8] Jacob K. Christopher, Michael Cardei, Jinhao Liang, and Ferdinando Fioretto. Neuro-symbolic generative diffusion models for physically grounded, robust, and safe generation, 2025.

[9] Chris Hokamp and Qun Liu. Lexically constrained decoding for sequence generation using grid beam search, 2017.

[10] Matt Post and David Vilar. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation, 2018.

[11] Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Neurologic decoding: (un)supervised neural text generation with predicate logic constraints, 2021.

[12] Ximing Lu, Sean Welleck, Peter West, Liwei Jiang, Jungo Kasai, Daniel Khashabi, Ronan Le Bras, Lianhui Qin, Youngjae Yu, Rowan Zellers, et al. Neurologic a* esque decoding: Constrained text generation with lookahead heuristics. *arXiv preprint arXiv:2112.08726*, 2021.

[13] Kevin Yang and Dan Klein. Fudge: Controlled text generation with future discriminators. *arXiv preprint arXiv:2104.05218*, 2021.

[14] Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gokcen Eraslan, Surag Nair, Tommaso Biancalani, Shuiwang Ji, Aviv Regev, Sergey Levine, et al. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. *arXiv preprint arXiv:2408.08252*, 2024.

[15] Yingqing Guo, Yukang Yang, Hui Yuan, and Mengdi Wang. Training-free guidance beyond differentiability: Scalable path steering with tree search in diffusion and flow models. *arXiv preprint arXiv:2502.11420*, 2025.

[16] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models, 2025.

[17] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*, 2019.

[18] Guillaume Sanchez, Honglu Fan, Alexander Spangher, Elad Levi, Pawan Sasanka Ammanamanchi, and Stella Biderman. Stay on topic with classifier-free guidance. *arXiv preprint arXiv:2306.17806*, 2023.

[19] Sophia Tang, Yinuo Zhang, and Pranam Chatterjee. Peptune: De novo generation of therapeutic peptides with multi-objective-guided discrete diffusion. *ArXiv*, pages arXiv–2412, 2025.

[20] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

[21] Fangping Wan, Daphne Kontogiorgos-Heintz, and Cesar de la Fuente-Nunez. Deep generative models for peptide design. *Digital Discovery*, 1(3):195–208, 2022.

[22] Robert EW Hancock and Hans-Georg Sahl. Antimicrobial and host-defense peptides as new anti-infective therapeutic strategies. *Nature biotechnology*, 24(12):1551–1557, 2006.

[23] Yuzuru Itoh, Shun-ichi Sekine, Shiro Suetsugu, and Shigeyuki Yokoyama. Tertiary structure of bacterial selenocysteine trna. *Nucleic acids research*, 41(13):6729–6738, 2013.

[24] Martina Jahn, M John Rogers, and Dieter Söll. Anticodon and acceptor stem nucleotides in trnagln are major recognition elements for e. coli glutaminyl-trna synthetase. *Nature*, 352(6332):258–260, 1991.

[25] Clarence W Chan, Bhaskar Chetnani, and Alfonso Mondragón. Structure and function of the t-loop structural motif in noncoding rnas. *Wiley Interdisciplinary Reviews: RNA*, 4(5):507–522, 2013.

[26] Arjun Prabhakar, Natalie Krahn, Jingji Zhang, Oscar Vargas-Rodriguez, Miri Krupkin, Ziao Fu, Francisco J Acosta-Reyes, Xueliang Ge, Junhong Choi, Ana Crnković, et al. Uncovering translation roadblocks during the development of a synthetic trna. *Nucleic acids research*, 50(18):10201–10211, 2022.

[27] T Brennan and M Sundaralingam. Structure, of transfer rna molecules containing the long variable loop. *Nucleic Acids Research*, 3(11):3235–3252, 1976.

[28] Jiacheng Ye, Jiahui Gao, Shansan Gong, Lin Zheng, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Beyond autoregression: Discrete diffusion for complex reasoning and planning. *arXiv preprint arXiv:2410.14157*, 2024.

[29] Jacob Witten and Zack Witten. Deep learning regression model for antimicrobial peptide design. *BioRxiv*, page 692681, 2019.

[30] Alexander Rich and Uttam L RajBhandary. Transfer rna: molecular structure, sequence, and properties. *Annual review of biochemistry*, 45(1):805–860, 1976.

[31] Patricia P Chan and Todd M Lowe. Gtrnadb 2.0: an expanded database of transfer rna genes identified in complete and draft genomes. *Nucleic acids research*, 44(D1):D184–D189, 2016.

A SearchDiff Pseudocode

Algorithm 1 SearchDiff

Input: Pre-trained MDM denoiser $\boldsymbol{x}_\theta$, constraint-violation function $V(\cdot)$, steps T, CSS budget M, refinement rounds R.
Output: Clean discrete sequence $\boldsymbol{x}_0$

$\boldsymbol{x}_T \leftarrow (\mathtt{M}, \ldots, \mathtt{M})$ {Initialize fully masked}
for $t = T, T-1, \ldots, 1$ **do**
  $\hat{\boldsymbol{x}}_0^{(t)} \leftarrow \boldsymbol{x}_\theta(\boldsymbol{x}_t, t)$
  Search Initialization via CSS:
  for $j = 1, \ldots, M$ **do**
    Sample $\boldsymbol{x}_t^{(j)} \sim p_t(\cdot \mid \hat{\boldsymbol{x}}_0^{(t)}, \boldsymbol{x}_t)$ {Clamp positions with $x_t^i \neq \mathtt{M}$}
  end for
  $\boldsymbol{x}_t^{\text{css}} \leftarrow \arg\min_{j \in \{1,\ldots,M\}} \preceq_{\mathcal{C}} \boldsymbol{\nu}(\boldsymbol{x}_t^{(j)})$
  Local search refinement:
  $\boldsymbol{x}^{(0)} \leftarrow \boldsymbol{x}_t^{\text{css}}$
  $\boldsymbol{x}^* \leftarrow \arg\min_{\tilde{\boldsymbol{x}} \in \mathcal{N}_t(\boldsymbol{x}_t^{css})} \preceq_{\mathcal{C}} \boldsymbol{\nu}(\tilde{\boldsymbol{x}})$ {$\mathcal{N}_t(\cdot)$ is the Hamming-1 neighborhood}
  if $\boldsymbol{\nu}(\boldsymbol{x}^*) \prec_{\mathcal{C}} \boldsymbol{\nu}(\boldsymbol{x}_t^{css})$ **then**
    $\bar{\boldsymbol{x}}_t \leftarrow \boldsymbol{x}^*$ {Greedy improvement}
  else
    $\bar{\boldsymbol{x}}_t \leftarrow \boldsymbol{x}_t^{css}$ {Keep CSS initialization}
  end if
  Modified reverse kernel:
  for $i = 1, \ldots, L$ **do**
    if $x_t^i \neq \mathtt{M}$ **then**
      $x_{t-1}^i \leftarrow \bar{x}_t^i$
    else
      Sample $x_{t-1}^i \sim \mathrm{Cat}\Big(\gamma_t\, \mathtt{M} + \eta_t\, \delta_{\bar{x}_t^i}\Big)$
    end if
  end for
  $\boldsymbol{x}_{t-1} \leftarrow (x_{t-1}^1, \ldots, x_{t-1}^L)$
end for
return $\boldsymbol{x}_0$

Algorithm 1 (SearchDiff) augments the standard reverse chain of a masked discrete diffusion model (MDM) with an inference-time search step that reduces a user-specified constraint-violation score. Our code is publicly available at `https://github.com/Tahuubinh/searchdiff`.

The inputs are a pre-trained denoiser $\boldsymbol{x}_\theta$, a vector-value violation map $\boldsymbol{\nu}(\cdot)$ that evaluates a fully discrete candidate sequence, the number of reverse steps T, a candidate search sampling (CSS) budget M, and a maximum number of local-refinement rounds R. The algorithm returns the final denoised sequence $\boldsymbol{x}_0$.

The procedure initializes the chain at the fully masked state $\boldsymbol{x}_T = (\mathtt{M}, \ldots, \mathtt{M})$ and iterates backward for $t = T, T-1, \ldots, 1$. At each step, the denoiser produces a clean prediction $\hat{\boldsymbol{x}}_0^{(t)} \leftarrow x_\theta(\boldsymbol{x}_t, t)$, which serves as a guide for proposing improved intermediate states at level t. SearchDiff then performs search initialization via CSS: it draws M candidates $\boldsymbol{x}_t^{(j)} \sim p_t(\cdot \mid \hat{\boldsymbol{x}}_0^{(t)}, \boldsymbol{x}_t)$ from the model's reverse-time proposal distribution at step t. Importantly, already-unmasked positions are clamped (positions with $x_t^i \neq \mathtt{M}$ are preserved), so CSS only explores values for the currently masked coordinates. Among the M candidates, the algorithm selects the lowest-violation sample

$$\boldsymbol{x}_t^{\text{css}} \leftarrow \arg \min_{j \in \{1,\ldots,M\}} V(\boldsymbol{x}_t^{(j)}),$$

which provides a "best-of-M" initialization for refinement.

Next, SearchDiff performs a single greedy local search refinement starting from $\boldsymbol{x}_t^{\text{css}}$. It evaluates the step-conditioned Hamming-1 neighborhood $\mathcal{N}_t(\boldsymbol{x}_t^{\text{css}})$, comprising all sequences that differ from $\boldsymbol{x}_t^{\text{css}}$

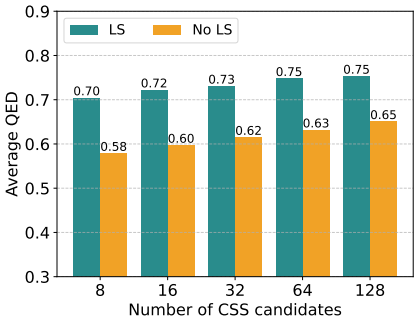

Figure 5: Average QED by LS and no LS approaches with different numbers of CSS candidates.

Table 6: Performance under increasing SA thresholds (comparison with CDD) on the QM9 dataset. CDD results reported from [5].

Method	SA satisfied	Mean QED
SA ≤ 3.0		
CDD	36	0.63
SearchDiff	391.8 ± 6.1	0.66 ± 0.01
SA ≤ 3.5		
CDD	22	0.62
SearchDiff	448.6 ± 9.0	0.70 ± 0.00
SA ≤ 4.5		
CDD	33	0.58
SearchDiff	484.8 ± 6.3	0.75 ± 0.00

at a single editable position. Then, it selects the neighbor with the minimal violation:

$$\boldsymbol{x}' \leftarrow \arg \min_{\tilde{\boldsymbol{x}} \in \mathcal{N}_t(\boldsymbol{x}^{(r)})} \preceq_{\mathcal{C}} \boldsymbol{\nu}(\tilde{\boldsymbol{x}}).$$

If this move strictly reduces the violation, that is $\boldsymbol{\nu}(\boldsymbol{x}') \prec_{\mathcal{C}} \boldsymbol{\nu}(\boldsymbol{x}^{(r)})$, the algorithm accepts it and continues; otherwise it terminates. The refined sequence after the last accepted move is denoted $\bar{\boldsymbol{x}}_t$.

Finally, SearchDiff uses $\bar{\boldsymbol{x}}_t$ to define a modified reverse kernel that preserves feasibility improvements while maintaining the diffusion-style stochastic unmasking. For each position $i \in 1, \ldots, L$, if the coordinate was already fixed at time t (i.e., $x_t^i \neq \mathtt{M}$), SearchDiff deterministically carries forward the refined value, $x_{t-1}^i \leftarrow \bar{x}_t^i$. If instead the coordinate is still masked, it samples from a two-component categorical distribution that either keeps the mask (with weight γ_t) or commits to the refined token $\bar{x}_t^i$ (with weight η_t), namely

$$x_{t-1}^i \sim \mathrm{Cat}\Big(\gamma_t\, \mathtt{M} + \eta_t\, \delta_{\bar{x}_t^i}\Big).$$

This step implements controlled unmasking guided by the refined candidate, ensuring that improvements found by CSS and local search influence the subsequent reverse trajectory rather than being applied only as a post-hoc repair. Repeating this process down to $t = 1$ yields the final discrete output $\boldsymbol{x}_0$.

B Additional Results

B.1 Biological Sequence Generation

As shown in Figure 5, SearchDiff exhibits a distinct test-time scaling phenomenon, where both variants show incremental performance gains as the number of CSS candidates increases. Specifically, the *No LS* variant scales from an average QED of 0.58 at 8 candidates to 0.65 at 128 candidates, while the *LS* variant improves from 0.70 to 0.75. This behavior indicates that the framework effectively leverages increased computational budget at inference time to explore a wider search space, consistently identifying higher-quality candidates across the discrete manifold.

Table 5: Time per molecular generation of different SearchDiff's variations on QM9 dataset constraints (SA ≤ 4, maximize QED).

Method	Setting	Time (s)
MDLM	–	0.67
MDLM CBG	–	1.28
TreeG-SC	–	32.12
MCTG	–	29.86
SearchDiff		
Local search: No	32 CSS candidates	1.88
	128 CSS candidates	6.00
Local search: All steps	1 CSS candidate	3.66
	32 CSS candidates	10.68

To further investigate the flexibility of SearchDiff, we evaluate its performance against CDD across varying synthetic accessibility (SA) thresholds. As shown in Table 6, SearchDiff demonstrates a remarkable ability to adapt to both strict and relaxed constraints. While CDD remains limited to a very small set of satisfied

Table 7: Impact of the number of local search steps per denoising step on SearchDiff performance.

Num. LS per denoising step	Admissible Molecules	SA $\leq$ 4	Mean QED
1	474.2 ± 8.7	470.2 ± 7.8	0.73 ± 0.00
2	$\mathbf{480.0 \pm 8.6}$	$\mathbf{476.8 \pm 7.2}$	0.73 ± 0.00
3	477.0 ± 8.4	473.6 ± 6.9	$\mathbf{0.74 \pm 0.00}$

Table 8: Efficiency comparison in generating OOD samples of SearchDiff compared to rejection sampling.

Method	OOD samples found	Avg. time per OOD sample
MDM + rejection	4.4 ± 0.8 per 32768 samples	4989.67s
SearchDiff	332.4 ± 9.9 per 1024 samples	32.90s

samples (ranging from 22 to 36), SearchDiff consistently produces hundreds of satisfying molecules. SearchDiff also dominates CDD in terms of maximizing QED.

As shown in Table 5, all SearchDiff variations generate molecules faster compared to TreeG-SC while achieving superior results in configurations incorporating Local Search (LS), as evidenced in Table 10. Although utilising LS leads to increased computational time, it proves more efficient than high-volume sampling; notably, the variation with 128 CSS candidates without LS is both slower and performs worse than variants using LS. While SearchDiff is still slower than vanilla MDLM, it remains a highly flexible, training-free method that can steer pre-trained models across diverse scientific use cases without the prohibitive costs of retraining or fine-tuning.

As shown in Table 7, we evaluate the sensitivity of SearchDiff to the number of local search steps performed at each denoising iteration. By default, SearchDiff employs a single LS step ($R = 1$) to maintain computational efficiency. Our results indicate that while increasing R to 2 or 3 provides marginal gains in mean QED and constraint satisfaction, the performance remains remarkably stable. This suggests that the continuous guidance provided by interleaving search throughout the entire diffusion trajectory is more critical than the intensity of local refinement within any single step. Furthermore, the slight decrease in admissible molecules and SA-compliant samples observed at $R = 3$ compared to $R = 2$ suggests a trade-off between optimization intensity and generative diversity. As the number of local search steps increases, the search operator may become overly greedy, causing distinct denoising trajectories to collapse into a narrow set of high-scoring local optima.

As shown in Table 8, we evaluate the efficiency of SearchDiff in discovering out-of-distribution (OOD) molecules (defined as novel molecules with a QED score exceeding the training set maximum) compared to a standard MDM baseline using rejection sampling. SearchDiff identifies an average of 332.4 ± 9.9 OOD samples per 1,024 generated sequences, whereas the rejection-based baseline finds only 4.4 ± 0.8 candidates even when the sample pool is expanded to 32768. This disparity highlights SearchDiff's ability to overcome the "distributional trap" by utilizing search-augmented steering to actively explore rare regions of the chemical manifold that are typically inaccessible through global positional statistics. Most notably, the computational efficiency gains are profound: SearchDiff requires only 32.90 seconds per OOD molecule, compared to 4989.67 seconds for rejection sampling, representing a speedup of roughly $150\times$. These results underscore that SearchDiff is a highly practical engine for scientific discovery, enabling the identification of high-value candidates in the "tails" of the training distribution without the prohibitive time costs of exhaustive sampling.

Table 9 illustrates the trade-off between strict lexicographical priority and compensatory weighted sum scalarization. The "SA-first lexicographic" rule achieves significantly higher adherence to the higher-priority requirements, with 470.2 samples satisfying $SA \leq 4$ compared to 439.2 for the weighted sum approach. While the weighted sum yields a higher Mean QED (0.76 vs. 0.73), this improvement stems from the compensatory nature of scalarization, where high property scores can mask violations in primary structural constraints.

As shown in Table 9, we compare our default lexicographical ranking against several weighted-sum scalarization baselines, where SA scores were scaled to the $(0, 1)$ range (same as QED). The results illustrate a clear distinction between the non-compensatory nature of lexicographical optimization and

Table 9: Performance of SearchDiff with different ranking rule.

Ranking rule	SA ≤ 4	Mean QED
SA-first lexicographic	**470.2** $\pm$ **7.8**	0.73 $\pm$ 0.00
SA weight = 0.75, QED weight = 0.25	467.4 $\pm$ 9.4	0.75 $\pm$ 0.00
SA weight = 0.5, QED weight = 0.5	439.2 $\pm$ 11.6	0.76 $\pm$ 0.00
SA weight = 0.25, QED weight = 0.75	348.6 $\pm$ 10.8	**0.76** $\pm$ **0.00**

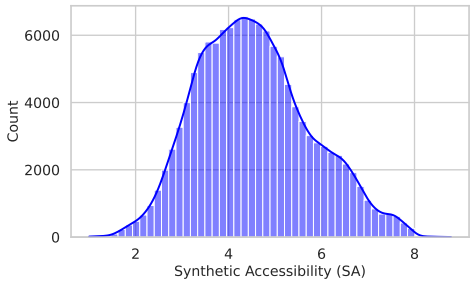

Figure 6: Distribution of Synthetic Accessibility (SA) scores in the training dataset.

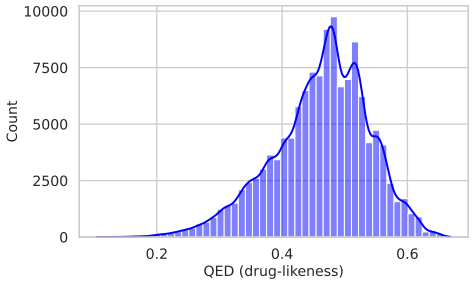

Figure 7: Distribution of QED (drug-likeness) scores in the training dataset.

the compensatory behavior of weighted sums. While increasing the weight for QED in a scalarized objective allows for marginal gains in mean QED (up to 0.76), it leads to a catastrophic collapse in structural feasibility, with the number of molecules satisfying $SA \leq 4$ dropping from 470.2 to 348.6. The lexicographical approach enforces a strict priority that guarantees the considered task of maximizing QED while satisfying SA ≤ 4. This underscores the versatility of the SearchDiff, which can be adapted to either strict hierarchical priorities or standard weighted-sum objectives depending on the specific requirements of the task.

Next, we isolate the contributions of CSS and LS on the task of enforcing SA $\leq$ 4 while maximizing QED (Table 10). Notice that increasing the CSS candidate budget improves results across settings, is the main driver of the gains and cannot be replicated by scaling CSS alone. In particular, SearchDiff with a single CSS candidate plus LS achieves 263.6 molecules with SA $\leq$ 4, substantially outperforming the variant with 128 CSS candidates but no LS (145.0). Applying search throughout the trajectory (all steps) further improves performance relative to post-hoc refinement applied only at the final step until convergence.

As shown in Table 12, we also consider the settings of constraints: SA ≤ 3 and QED ≥ 0.6 [3]. The dramatic disparity in results shown in this table reveals that existing methods are predominantly confined to the high-density regions of the training data. For complex multi-objective tasks where $SA \leq 3$ and $QED \geq 0.6$, the feasible molecules are often located in rare or sparse regions of the chemical manifold, as shown in Figure 6 and Figure 7. MDLM and TreeG-SC primarily struggle to reach these solutions that reside in the "tails" of the training distribution. SearchDiff effectively overcomes this "distributional trap" by utilizing search-augmented steering to actively explore the discrete manifold. This capability enables it to successfully navigate into these rare feasible regions and identify hundreds of valid, unique, and novel molecules where traditional baselines find virtually none.

Table 10: Ablation of SearchDiff search steps on QM9 dataset under constraints (SA ≤ 4, maximize QED). Local search is either deactivated (top), active only in the last step, and run until convergence (middle), or active in each denoising step.

Local search	**# CSS cand.**	**SA** ≤ 4 **[↑]**	**QED [↑]**
Not active	1	110.0	0.48
	32	143.2	0.62
	128	145.0	0.65
Last step until convergence	1	419.2	0.66
	32	456.2	0.72
	128	500.2	0.73
All steps	1	433.8	0.67
	32	470.2	0.73
	128	**514.2**	**0.75**

To provide another standardized comparison under an identical search budget, we conducted an additional experiment comparing SearchDiff against Random-Init + Local Search, Rejection Sampling, and MCTG. This experiment was evaluated over 5 independent runs, with each run generating 256 molecules for the joint constraints $SA \leq 3$ and $QED \geq 0.6$. A single search budget unit (an evaluation call) is defined as one joint evaluation of $SA + QED$. We specifically configured the three baseline algorithms so that their allocated search budgets are at least equal to the worst-case number of evaluation calls of SearchDiff. We report the number of satisfied admissible molecules, search budget, and model calls (forward passes) in Table 11. Random-Init + Local Search completely fails to generate a single satisfying sample across all five runs. Rejection Sampling and MCTG both yield worse performance despite requiring a drastically higher number of model calls compared to SearchDiff. This massive discrepancy in forward passes makes executing these two baseline algorithms substantially slower in practice. These results clearly illustrate that SearchDiff's efficiency stems from effectively integrating search within the diffusion trajectory, rather than merely relying on a large evaluation budget.

Table 11: Standardized search budget comparison for the joint constraints (SA ≤ 3 and QED ≥ 0.6). The allocated search budgets for the baselines match or exceed the worst-case evaluation calls of SearchDiff.

Method	**Satisfied Admissible Mol [↑]**	**Search Budget [↓]**	**Model Calls [↓]**
Random-Init + Local Search	0	21794	N/A
Rejection Sampling	78.8 ± 4.1	21567	690144
MCTG	138.2 ± 8.1	21760	653141
SearchDiff (Ours)	$\mathbf{170.4 \pm 4.6}$	**21567**	**32**

As shown in Figure 8, the number of molecules satisfying the stringent combined constraints of $SA \leq 3$ and $QED \geq 0.6$ underscores the test-time scaling capabilities of SearchDiff. Across all tested candidate counts, the *LS* variants consistently and significantly outperform the *No-LS* counterparts, achieving nearly triple the number of satisfied molecules even at the lowest search breadth. These results confirm that while increasing the number of CSS candidates provides a steady improvement in results, the discrete local refinement provided by LS is the fundamental driver for successfully navigating complex, multi-dimensional constraint boundaries that are otherwise inaccessible through global sampling alone.

As shown in Table 13, we evaluate the performance of SearchDiff across various QED constraint thresholds, revealing a clear trade-off between optimization intensity and sample diversity. As the QED threshold becomes more stringent, the mean QED naturally rises from 0.68 to 0.75, while the number of admissible molecules (valid, unique, and novel) decreases from 833.4 $\pm$ 10.6 to 756.2 $\pm$ 13.9. The "Maximize QED" configuration, which utilizes an ideal target of $QED \geq 1.0$, achieves the highest overall quality with a mean QED of 0.77 but results in the lowest diversity with 497.6 admissible molecules. This trend indicates that more aggressive optimization targets lead to greedier search behavior, where the operator collapses diverse denoising trajectories into a narrower set of high-performing local optima.

Table 12: Performance of different approaches on molecular generation in terms of number of valid, unique, and novel molecules satisfied these constraints: SA ≤ 3 and QED ≥ 0.6.

Method	**All constraints**	**SA ≤ 3**	**QED ≥ 0.6**
MDLM	1.2 ± 0.7	19.6 ± 4.2	4.8 ± 1.9
TreeG-SC	2.4 ± 1.6	66.6 ± 5.5	4.0 ± 1.8
MCTG	233.6 ± 5.4	597.4 ± 12.4	233.6 ± 5.4
SearchDiff	$\mathbf{560.6 \pm 14.3}$	$\mathbf{682.8 \pm 9.0}$	$\mathbf{566.4 \pm 16.8}$

Table 13: Performance of SearchDiff with different QED constraints in the QED maximization task.

QED constraint	**Admissible Mol [↑]**	**Mean QED [↑]**
QED ≥ 0.65	$\mathbf{892.0 \pm 16.9}$	0.68 ± 0.00
QED ≥ 0.7	833.4 ± 10.6	0.72 ± 0.00
QED ≥ 0.75	756.2 ± 13.9	0.75 ± 0.00
Maximize QED	497.6 ± 4.5	$\mathbf{0.77 \pm 0.00}$

Table 14: Comparison of diversity scores (higher is more diverse) across approaches and data modalities (Tanimoto score for molecules; normalized edit distance for peptides and tRNA).

Model	Molecules	Peptides	tRNA
Standard	0.0717 ± 0.0585	0.8582 ± 0.0924	0.4517 ± 0.0971
CBG	0.1293 ± 0.0684	0.8584 ± 0.0908	0.4966 ± 0.0462
TreeG-SC	0.1465 ± 0.2017	0.7308 ± 0.1149	0.3736 ± 0.1508
SearchDiff	0.1186 ± 0.0733	0.8115 ± 0.0608	0.5242 ± 0.0373

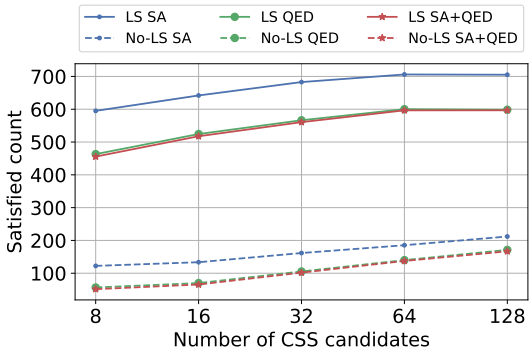

Figure 8: Number of satisfied molecules with different variants of SearchDiff and different numbers of CSS candidates versus other methods with constraints (SA ≤ 3, QED ≥ 0.6).

Table 15: Time per sequence generation of different methods for peptide and tRNA generation.

Experiment	**Method**	**Time (s)**
Peptides	MDLM	0.74
	MDLM CBG	1.66
	TreeG-SC	21.09
	SearchDiff	1.11
tRNA	MDLM	0.82
	MDLM CBG	2.49
	TreeG-SC	52.93
	SearchDiff	7.72

Table 14 presents the pairwise diversity analysis across all experimental domains, using Tanimoto similarity for molecules (where lower scores indicate higher diversity) and normalized edit distance for peptides and tRNA (where higher scores indicate higher diversity). The results demonstrate that SearchDiff maintains competitive diversity across all domains while simultaneously achieving significantly stronger constrained generation performance. Notably, SearchDiff remains more diverse than stronger constrained baselines such as CBG (in the molecules and tRNA domains) and TreeG-SC. This indicates that the gains in feasibility and property optimization provided by our search-augmented steering do not come at the cost of severe mode collapse, effectively preserving the generative variety and structural richness of the underlying masked diffusion model.

As shown in Table 15, SearchDiff achieves remarkable efficiency across both Peptides and tRNA generation tasks. In the Peptides experiment, SearchDiff (1.11s) introduces only a marginal overhead compared to the unconstrained MDLM (0.74s) and, notably, performs faster than the guided MDLM CBG baseline (1.66s). Compared to the grammar-based TreeG-SC (21.09s), SearchDiff is approximately 19$\times$ faster while providing superior constraint satisfaction. For the more complex tRNA domain, although the latency increases to 7.72s (reflecting the computational cost of the unified folding simulation required for structural verification) SearchDiff still maintains a nearly 7$\times$ speed advantage over TreeG-SC (52.93s). These results demonstrate that our training-free framework is highly practical for large-scale scientific discovery where both structural integrity and inference speed are paramount.

B.2 Symbolic Reasoning

Across both symbolic reasoning tasks, as observed in Table 17, varying the number of denoising steps T highlights that SearchDiff's improvements are consistent across different amount of denoising steps in the reverse diffusion chains. For Sudoku, vanilla MDM improves steadily as T increases (69.3$\rightarrow$93.2), while SearchDiff remains consistently higher at every T (91.4$\rightarrow$96.5), with the largest absolute gain occurring with smaller

Table 16: Improvement rate across denoising steps.

Step	**Improvement Rate (%)**
20	9.04 ± 0.46
19	5.76 ± 0.69
18	4.52 ± 0.67
17	3.18 ± 0.99
15	2.02 ± 0.53
10	1.02 ± 0.16
5	0.30 ± 0.10
1	0.06 ± 0.05

Table 17: Accuracy (%) across tasks. For MDM, we report Vanilla sampling, SearchDiff, and SearchDiff applied to the last step under denoising steps $T \in \{1, 5, 10, 15, 20\}$.

Model	Time Steps	Sudoku (Acc.%)			Boolean SAT (Acc.%, 7 vars)		
		Vanilla	SearchDiff	Last_Step	Vanilla	SearchDiff	Last_Step
MDM (6M)	T=1	69.3	91.4	-	9.4	71.5	-
	T=5	89.3	95.6	89.0	9.6	72.6	17.6
	T=10	91.9	95.9	92.1	9.5	72.6	11.7
	T=15	93.4	96.7	92.4	9.5	73.8	11.5
	T=20	93.2	96.5	92.2	9.6	76.0	11.4
AR (GPT-2)	6M	31.25	–	–	86.13	–	–
AR (GPT-2)	85M	26.56	–	–	93.97	–	–
AR (GPT-2)	303M	23.21	–	–	94.42	–	–

T value. This indicates that search-augmented refinement is particularly effective when the diffusion process has limited opportunity to correct early mistakes. For Boolean SAT, vanilla MDM stays consistently low across all T (9.4–9.6), suggesting that additional denoising steps alone do not help the sampler locate the sparse feasible set; in contrast, SearchDiff achieves strong accuracy even at T=1 (71.5) and improves modestly with more steps (up to 76.0 at T=20), indicating that multi-step denoising mainly provides incremental benefit once search supplies a feasibility-driven direction. Finally, applying search only at the last step results in weaker performance across T on both tasks, reinforcing that distributing search throughout the denoising trajectory is more effective than post-hoc repair. We also examine, in Table 16, the full denoising trajectory by tracking the number of corrections made at each step. The correction rate declines nearly monotonically over time, suggesting that early intermediate states provide the greatest opportunity for corrective intervention, while later states are comparatively stable. We will include this diagnostic in the final paper, noting that the exact shape of the correction profile is task-dependent.

C Experimental Settings

This section provides the detailed experimental settings for results shown in Section 6 and Section B.

C.1 Biological Sequence Generation

Hardware Configuration All experiments for biological sequence generation were conducted on NVIDIA A16 GPUs.

C.1.1 Molecular Generation

Molecular generation refers to the use of computational models to create novel molecular structures with desired chemical and biological properties. This field plays a crucial role in drug discovery and material science by accelerating the design of compounds that meet specific criteria, such as synthetic asccessibility (SA) and drug-likeness (QED) [5, 4].

We use the same settings as existing papers [5, 4]. We generate SMILES sequences using a base Transformer model of 92.4 million parameters, which was trained on the QM9 [20] dataset. At inference time, the diffusion sampler initializes the target SMILES sequence to [MASK] and iteratively denoises over 32 steps until all target positions are unmasked. Given the fully unmasked sequences, all special tokens are replaced to achieve the clean SMILES representations. At each diffusion step, the predicted clean sequence is post-processed to remove unmatched branch parentheses and invalid ring indices before evaluation. This study uses the QM9 dataset [20], which consists of small organic molecules with up to 9 heavy atoms.

This study focuses on generating molecules that satisfy key constraints, including chemical validity, novelty, SA, and QED. We use SA and QED as the evaluators of SearchDiff. At each diffusion step,

we predict a clean SMILES sequence and apply a post-processing step that removes unmatched branch parentheses and invalid ring indices before feeding the sequence to the evaluators. For neighboring search, we consider all valid atom types supported in QM9. Ensuring these constraints is essential for producing molecules that are not only chemically valid but also novel, practically synthesizable, and pharmaceutically relevant. Our experimental setting is based on [4]. Since reporting the mean and standard deviation of the run-level mean QED over 5 runs, the standard deviation can be really small (for example, the standard deviation for mean QED of SearchDiff is 0.004855293615777466). For the level of diversity, we report the Tanimoto similarity as reported in Table 14.

Search Operator implementation. To steer the denoising trajectory toward the feasible manifold, our search operator evaluates and ranks candidates based on their proximity to satisfying domain-specific constraints. Specifically, for each candidate molecule $\boldsymbol{x}$, we compute its Synthetic Accessibility score and Quantitative Estimate of Drug-likeness. We quantify the distance of these properties to the satisfaction region; for instance, the SA violation is defined as $\max(0, SA(\boldsymbol{x}) - \tau)$, where τ is the threshold for synthesizability. In our implementation, we adopt a selection strategy that prioritizes SA satisfaction over QED optimization. When comparing two candidate molecules, the operator first selects the one with the lower SA violation; QED values are only considered as a secondary tie-breaker once a baseline level of synthetic accessibility is achieved. This prioritization ensures that SearchDiff fundamentally secures structural validity and chemical feasibility before attempting to refine more complex medicinal properties. To compute SA and QED scores, this study utilizes the widely adopted `RDKit`[6] library, which offers reliable implementations of these metrics. Ensuring these constraints is essential for producing molecules that are not only chemically valid but also novel, practically synthesizable, and pharmaceutically relevant.

C.1.2 Peptides Generation

Peptide generation [21] involves designing and synthesizing short chains of amino acids with specific biological or chemical properties. Computational peptide generation aims to create novel peptide sequences that satisfy functional constraints such as antimicrobial activity, stability, or binding affinity. These peptides have broad applications in therapeutics, diagnostics, and biotechnology. Generative models help explore the vast combinatorial space of possible sequences to identify candidates with desirable characteristics efficiently.

The constraints chosen for peptide generation: short sequences of 10 to 50 amino acids, an overall positive charge typically between +2 and +9, and a substantial proportion ($\geq$30%) of hydrophobic residues; are grounded in well-established biological principles of antimicrobial and host-defense peptides [22]. These short cationic amphiphilic peptides play crucial roles in innate immune defense across nearly all life forms, acting as natural antibiotics with broad-spectrum antimicrobial activities. Their positive charge facilitates interaction with negatively charged microbial membranes, while the hydrophobic residues enable the formation of amphipathic structures critical for membrane disruption and antimicrobial function. By imposing these constraints, the generated peptides are more likely to adopt structural and functional properties characteristic of effective antimicrobial agents, enhancing their potential therapeutic relevance. Additionally, these properties have been shown to influence peptide stability, specificity, and resistance to degradation, all important factors for developing novel peptide-based drugs.

This study employs the GRAMPA dataset [29] to evaluate generation capabilities on peptide sequences. This is a robust database containing sequences and experimental Minimum Inhibitory Concentration (MIC) values of antimicrobial peptides (AMPs).

Search Operator implementation. The search operator steers the denoising trajectory by evaluating candidates based on their biological viability and functional stability. We quantify the distance to the feasible set using structural constraints, such as net charge or hydrophobic ratio, alongside the normalized folding energy (ΔG) as the primary property for optimization. For a candidate sequence $\boldsymbol{x}$, we calculate a violation score for each predefined constraint; for example, if a specific net charge is required, the violation is defined as $|charge(\boldsymbol{x}) - target|$. The calculations were performed using `Biopython`[7] library.

[6] `https://www.rdkit.org/docs/api-docs.html`
[7] `https://biopython.org/docs/1.76/api/Bio.SeqUtils.ProtParam.html`

C.1.3 Transfer Ribonucleic Acid Generation

Transfer ribonucleic acid [30] (tRNA) generation involves creating synthetic or predicted tRNA sequences that mimic natural tRNAs' essential roles in protein synthesis. In designing constraints for tRNA sequence generation, it is essential to capture key structural elements critical for tRNA function. The acceptor stem typically consists of 7 to 9 base pairs formed by the pairing of the 5' and 3' terminal nucleotides, with the 3' end containing the conserved CCA tail for amino acid attachment [23, 24]. This stem may include non-canonical base pairs, reflecting natural structural flexibility. The D loop features a 4 to 6 base-pair stem and commonly contains dihydrouridine modifications [23]. The anticodon loop, crucial for codon recognition, generally consists of a 5-base-pair stem enclosing the anticodon triplet [23]. The TΨC loop derives its name from the characteristic inclusion of pseudouridine (Ψ), a chemically modified form of uridine, within its sequence. This nucleotide frequently appears as part of a conserved motif written as 5'-TΨCGA-3', in which the ribothymidine forms a stabilizing base pair with the adjacent adenine [25]. Additionally, the variable loop, located between the anticodon and TΨC loops, varies widely in length (3 to 21 bases) and can form a rigid arm in some tRNAs, differentiating class I and class II tRNAs [26, 27]. Incorporating these structural features as constraints ensures generated tRNA sequences maintain biological plausibility and functionality.

We trained the base model with the same network architecture as the molecular settings. The number of denoising steps remains 32, but the maximum sequence length is now 110. This makes the problem pose a significantly higher challenge than peptides and molecules due to the high average number of tokens to be unmasked per denoising step. For RNA modeling, this study curates a dataset of mammal tRNA sequences obtained from the GtRNAdb [31] database.

Search Operator implementation. The search operator is designed to navigate the discrete manifold by prioritizing the preservation of the canonical secondary structure. We characterize the feasible set through a combination of structural constraints. For each candidate sequence, the operator calculates a violation score based on its sequence-structure consistency; for instance, a violation occurs if a predicted sequence fails to form the requisite hydrogen bonds necessary for stable folding. Specifically, we group the D-loop (4-6 base-pair stem), the 5-base-pair anticodon stem, and the variable loop (3-21 bases) into a single collective fold constraint. The rationale for this grouping is that all these constraints are derived from the simulated secondary structure, which must be computed before these specific topological values can be evaluated. By unifying them into a single operator, we achieve significant computational efficiency, as the secondary structure is simulated only once per candidate instead of three separate, redundant, and time-consuming simulations. The properties of tRNA sequences are computed using the `ViennaRNA`[8] package.

C.2 Symbolic Reasoning

C.2.1 Sudoku

Hardware Configuration All experiments for symbolic reasoning were conducted on NVIDIA A6000 GPUs.

Sudoku Search Operator. Search is guided by a simple constraint-only evaluator that measures how far a candidate grid is from satisfying Sudoku rules. Given a predicted 9×9 board, the evaluator computes a violation score by counting duplicate digits in each of the 27 Sudoku units (9 rows, 9 columns, and 9 3×3 subgrids). The total violation score is the sum across all units, where lower is better and a score of 0 indicates no duplicates anywhere. The search space consists of all candidate fillings of the Sudoku board on the `SOLUTION` side of the sequence. We consider local, token-level edits to the candidate grid: an action is a single-cell replacement that changes the digit at an editable position to an alternative value in $\{1, \ldots, 9\}$.

C.2.2 Boolean Satisfiability Problem

SAT Search Operator. Search is guided by a clause-based evaluator computed from the predicted assignment. Given a decoded candidate, it will parse the CNF clauses and evaluate each clause under the assignment. A clause satisfied if at least one of its literals evaluates to `True`; it is violated if

[8] `https://www.tbi.univie.ac.at/RNA`

all literals evaluate to `False`. The primary penalty is the number of violated clauses where lower is better. To prevent degenerate candidates it additionally penalizes undecided clauses and heavily penalize unassigned variables when present.

D Limitations

The inclusion of local search and candidate search sampling at each denoising step introduces additional computational overhead. However, this trade-off is justified by significantly higher performance in constraint satisfaction. Additionally, SearchDiff exhibits lower sample diversity compared to the standard MDLM. This can be attributed to the narrowing of the feasible target region during the search process. Nevertheless, SearchDiff still achieves competitive diversity scores relative to other controllable generation approaches.

The availability of a constraint evaluator is an explicit assumption of the problem studied in this work. SearchDiff is intended for applications where constraints can be evaluated but cannot necessarily be differentiated, learned, or incorporated during training. RDKit, Biopython, ViennaRNA, Sudoku rules, and SAT clause evaluation illustrate this setting. This method is not directly applicable when no informative evaluator is available.

E Impact Statement

This paper introduces SearchDiff to advance the field of constrained discrete diffusion. While our work has potential applications in scientific domains like biological design and symbolic reasoning, it represents a foundational research contribution to generative modeling, and we believe no specific societal consequences require unique highlighting.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the paper's contributions and scope. The paper clearly outlines its main claims and evaluation across various biological and symbolic reasoning tasks. The claims match with all the reported results.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Our paper discusses the limitations of the proposed work, primarily concerning the computational trade-offs and generative diversity. Table 5, Table 14, Table 15 show the related results. We additionally discuss this in Appendix D.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: The paper does not include formal theoretical results, theorems, or proofs. It focuses on introducing an empirical, inference-time framework.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: The paper provides detailed experimental settings, hyperparameters, and descriptions necessary to replicate the results. It outlines the datasets and evaluation metrics used and clearly describes the algorithm, including pseudocode and the methodological steps.

Guidelines:

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

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [No]

   Justification: We will release our source code and how to run the code upon acceptance.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: The paper specifies the experimental details necessary to understand the results in Section 6 and Section C.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: The paper reports the experimental results accompanied by standard deviations and report the number of runs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Hardware usage for the experiments is mentioned in Section C. This work is provides a genreal framework and is not specific to the type of compute workers or memory.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in the paper conforms in every respect with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

Justification: This paper includes a brief Impact Statement at Section E that identifies SearchDiff as a foundational research contribution. We acknowledge potential applications in scientific and reasoning domains while concluding that no specific societal consequences require highlighting.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

Answer: [N/A]

Justification: The paper poses no mentioned risks.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: All existing assets are properly credited through citations to their original publications. Furthermore, URLs for the software documentation are provided in the footnotes and appendix to ensure transparency.

Guidelines:

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

13. **New assets**

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

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

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

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

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

    Answer: [N/A]

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

    Guidelines:

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

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

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

    Answer: [N/A]

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

Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: This paper does not use LLMs as an important, original, or non-standard component of the core methods.

    Guidelines:

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