DP-DNA: DUAL-POLICY REINFORCEMENT LEARNING FOR DNA DISCRETE DIFFUSION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) provides a powerful framework for steering generative models toward task-specific objectives and has shown promising results in regulatory DNA design. However, fine-tuning discrete diffusion models with downstream rewards can lead to undesirable distributional degeneration, where improved predicted activity is accompanied by reduced pattern diversity, distorted sequence statistics, and decreased biological naturalness. In this work, we first systematically characterize this failure mode and find that its severity is strongly generation-order dependent: after RL, different unmasking schedules produce markedly different degrees of sequence collapse. Motivated by this observation, we formulate discrete diffusion generation as two coupled decisions: *where to generate* and *what to generate*. We introduce a dual-policy RL framework DP-DNA, which optimizes both a generation-order planner and a content generator with a joint order–content GSPO objective. A free-generation stream trains both policies to generate DNA with high functional reward, while a real-sequence stream teaches them to reproduce the sequence patterns of real DNA. Across HepG2, K562, and SK-N-SH enhancer design, our method achieves high predicted activity while producing more natural sequence distributions, supporting dual-policy learning for function-directed DNA generation.

1 INTRODUCTION

Reinforcement learning (RL) has become a powerful mechanism for steering generative models toward task-specific objectives beyond maximum-likelihood training. Its impact is now evident in large-scale language-model post-training, where policy optimization can substantially improve reasoning and task performance (Schulman et al., 2017; Guo et al., 2025; Zheng et al., 2025). Its success has also motivated reward-driven biological sequence design, where a learned predictor guides a generator toward desired functions (Yang et al., 2025; Chen et al., 2025; Wang et al., 2025). DNA discrete diffusion language models (DLMs) offer a promising foundation: their bidirectional context and flexible unmasking process support sequence construction beyond a fixed causal order (Yang et al., 2026; Song et al., 2025). Combining these models with RL could therefore enable flexible, function-directed regulatory DNA design.

However, optimizing a predictive reward can also favor sequences that exploit the scorer's preferences while departing from realistic DNA statistics (Wang et al., 2025; Chen et al., 2025). We investigate this behavior in RL-tuned D3LM and observe rising reward alongside excessive `GC` enrichment and reduced 3-mer entropy relative to reward-matched real sequences. Crucially, this collapse is *generation-order dependent*. Using high-activity real DNA, we construct partially revealed contexts with different position-selection strategies and evaluate their completions under a common decoder. After RL, the resulting reward curves diverge sharply, and higher-reward continuations often exhibit stronger compositional distortion and lower pattern diversity. These findings identify generation order as an important part of the reward–naturalness trade-off: *which positions are resolved first influences both functional optimization and sequence realism*.

Motivated by these observations, we introduce DP-DNA, a *dual-policy RL framework* that jointly learns *where to generate* and *what to generate*. A context-dependent planner and a content generator share a joint order–content Group Sequence Policy Optimization (GSPO) objective (Zheng et al., 2025). Two complementary streams connect functional optimization with real-sequence structure.

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

The *free-generation stream* updates both policies using downstream reward and a frozen content reference. The *real-sequence stream* couples a learned corruption model with content reconstruction and forward–reverse rate matching, adapting LoMDM's order-aware modeling to downstream RL (Hong et al., 2026b). The planner therefore learns task-specific priorities while receiving a separate signal for real-DNA compatibility; the generator learns both high-reward outputs and natural sequence patterns.

Our contributions are threefold:

- We characterize RL-induced distributional collapse in DNA diffusion models and establish its connection to generation order through reward-matched controls and real-context reveal experiments.
- We propose DP-DNA, which jointly optimizes a position planner and a content generator with joint order–content GSPO and complementary free-generation and real-sequence training signals.
- Experiments on HepG2, K562, and SK-N-SH enhancer design show strong activity, motif agreement, and genomic-model likelihood; ablations and order analyses characterize the contributions of the two streams and adaptive planning.

2 PRELIMINARIES

2.1 DISCRETE DIFFUSION LANGUAGE MODELS

Let $x = (x_1, \ldots, x_L) \in \mathcal{V}^L$ denote a tokenized DNA sequence and $[\mathtt{M}]$ a mask token. We use D3LM as the discrete diffusion backbone throughout this work (Yang et al., 2026). Given a clean sequence x, D3LM samples a corruption level $u \in (0, 1]$ and independently masks each position with probability u, producing x_u. A bidirectional Transformer then predicts the original tokens at masked positions using

$$\mathcal{L}_{\mathrm{D3LM}}(\theta) = -\mathbb{E}_{x,u,x_u}\left[\frac{1}{u}\sum_{i=1}^{L}\mathbb{I}[x_{u,i} = [\mathtt{M}]]\log p_\theta(x_i \mid x_u, i)\right]. \tag{1}$$

Training over different masking ratios allows the model to denoise from partially observed contexts and ultimately generate from a fully masked sequence.

Generation progressively resolves masked positions. Let s_t be the current partially generated sequence and $\mathcal{M}(s_t)$ its unresolved positions. At each event, a position scheduler g selects where to generate and D3LM predicts what to generate:

$$i_t \sim \underbrace{g(\cdot \mid s_t)}_{\text{order / where}}, \qquad v_t \sim \underbrace{p_\theta(\cdot \mid s_t, i_t)}_{\text{content / what}}, \qquad s_{t+1} = \mathrm{Write}(s_t, i_t, v_t). \tag{2}$$

Masked diffusion does not prescribe a unique order. A random scheduler uses $g_{\mathrm{rand}}(i \mid s) = 1/|\mathcal{M}(s)|$, while confidence-based decoding prioritizes positions with high model confidence, e.g., $\max_v p_\theta(v \mid s, i)$ (Yang et al., 2026; Hong et al., 2026a). For a sampled trajectory $\tau = (i_{1:T}, v_{1:T})$, its path probability is

$$Q_{\theta,g}(\tau) = \prod_{t=1}^{T} g(i_t \mid s_t)\, p_\theta(v_t \mid s_t, i_t). \tag{3}$$

This factorization separates generation order from token generation and will be used throughout our analysis and method.

2.2 REINFORCEMENT LEARNING FOR SEQUENCE GENERATION

Let c denote a design condition, such as a target cell type, and let $Q_\Omega(\tau \mid c)$ be a generative policy over complete trajectories with endpoint $x = f(\tau)$. Reinforcement learning adapts this policy to maximize a terminal reward $R_c(x)$, commonly with a reference-policy regularizer:

$$\max_{\Omega} \quad \mathbb{E}_{\tau \sim Q_\Omega(\cdot|c)}[R_c(f(\tau))] - \beta D_{\mathrm{KL}}(Q_\Omega \,\|\, Q_0). \tag{4}$$

For regulatory DNA design, R_c can be provided by a frozen sequence-to-function predictor. Following DRAKES (Wang et al., 2025), we use a fine-tuned Enformer-based model to score the predicted regulatory activity of each generated sequence.

We optimize sequence-level rewards with Group Sequence Policy Optimization (GSPO) (Zheng et al., 2025). Let Ω^- be the behavior policy that generates a group of trajectories $\{\tau_i\}_{i=1}^G$, with group-relative advantages A_i computed from their terminal rewards. GSPO assigns each trajectory a single length-normalized likelihood ratio

$$\rho_i(\Omega) = \exp\left[\frac{1}{T_i}\sum_{t=1}^{T_i}\log\frac{\pi_\Omega(a_{it} \mid s_{it}, c)}{\pi_{\Omega^-}(a_{it} \mid s_{it}, c)}\right], \tag{5}$$

and optimizes

$$\mathcal{L}_{\text{GSPO}} = -\frac{1}{G}\sum_{i=1}^{G}\min(\rho_i A_i, \text{clip}(\rho_i, 1-\epsilon, 1+\epsilon)A_i)\,. \tag{6}$$

For D3LM, the GSPO "sequence" is the sampled generation path. Under Eq. (2), each action is $a_t = (i_t, v_t)$ and $\pi(a_t \mid s_t) = g(i_t \mid s_t)p_\theta(v_t \mid s_t, i_t)$. The ratio in Eq. (5) is therefore obtained by replaying the same recorded states and actions under the current and behavior policies. With a fixed random scheduler, the position terms cancel and the ratio depends only on token probabilities; with a learned scheduler, its position probabilities must be included in the same path-level ratio. This sampled-path ratio is distinct from the marginal likelihood ratio of the final DNA sequence, which would require summing over all compatible generation orders.

3 Understanding the Behavior of RL in DNA Design

We examine two connected aspects of RL-based DNA design: changes in sequence content during reward optimization, and the role of generation order in shaping those changes.

3.1 Reward Optimization Can Induce Sequence Collapse

Regulatory DNA design commonly uses a trained neural predictor to score generated sequences. RL can consequently amplify sequence patterns favored by the scorer, making it important to examine the distribution underlying reward improvements (Wang et al., 2025; Chen et al., 2025). We track D3LM over 100 GSPO iterations for HepG2 design, using three independent training seeds. At each checkpoint, we generate sequences and average their scores under the frozen Enformer-based reward model. We also measure GC fraction—the proportion of G and C bases—and 3-mer entropy, which summarizes local-pattern diversity. Real sequences matched by the same predictor's reward provide a reference (Appendix A.2).

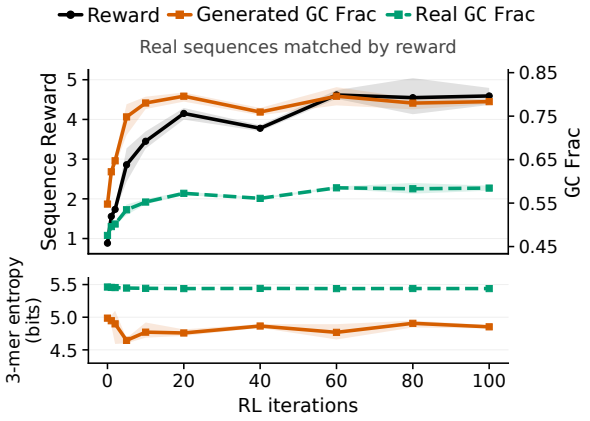
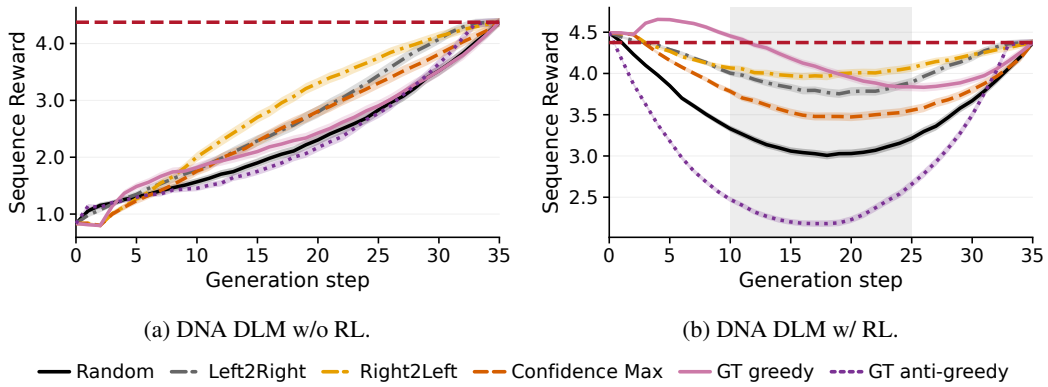


Figure 1: **Reward gains accompany compositional collapse.** D3LM during GSPO training: mean reward and generated/reward-matched real GC fractions (top), and 3-mer entropy (bottom). Shading shows 95% bootstrap intervals across three RL runs.

Figure 1 shows reward and GC enrichment rising together. Mean reward increases from 0.88 to 4.59, while generated GC fraction reaches 78.3%, compared with 58.5% in reward-matched real DNA. Generated 3-mer entropy is also lower: 4.85 versus 5.44 bits. At the content level, RL reward improvements are accompanied by compositional skew and reduced local-pattern diversity. The matched reference associates this degeneration with scorer-driven optimization, rather than the sequence statistics of naturally high-scoring DNA.



(a) DNA DLM w/o RL. (b) DNA DLM w/ RL.

Figure 2: **RL strengthens D3LM's dependence on reveal order.** Mean continuation reward from strategy-specific real contexts, with pointwise uncertainty bands and three independent RL training runs. The dashed line denotes the real targets' mean reward; the shaded region marks steps 10–25.

3.2 Generation order modulates collapse

Comparing real-context reveal strategies. To investigate the factors driving collapse, we construct partially revealed contexts from held-out DNA with high measured HepG2 activity. For each target, a strategy repeatedly selects a masked position and reveals the corresponding *true token*. Different reveal counts represent successive generation stages; for D3LM, a 200-base sequence comprises 35 tokens. The resulting contexts therefore contain genuine sequence information selected in different orders.

We compare six strategies. *Random* chooses uniformly among masked positions. *Left2Right* reveals positions from the left end to the right, while *Right2Left* proceeds from the right end to the left. *Confidence Max* selects the position with the highest top-token probability. To probe contrasting extremes of target-token predictability, we also use the real sequence as an oracle: *GT greedy* selects the position whose true token has the highest model probability, and *GT anti-greedy* selects the lowest. Their mathematical definitions are in Appendix A.3.

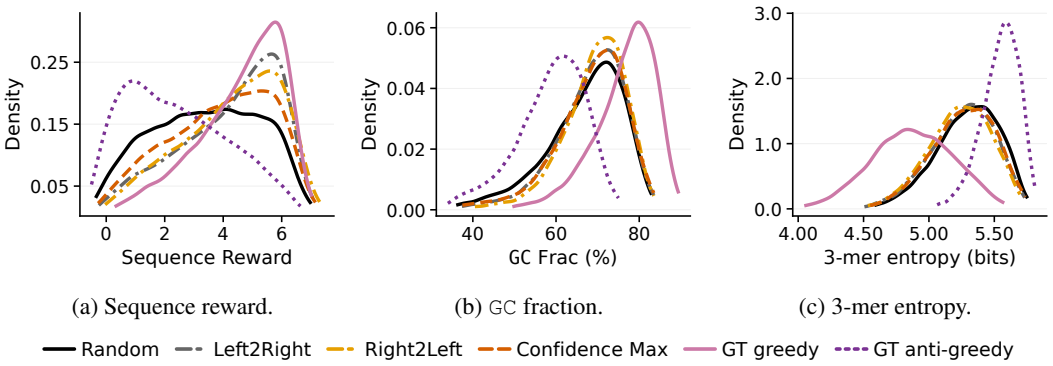
For each strategy-specific context s_t^g, we use a common random-order decoder to sample multiple completions. Their average reward estimates the continuation value:

$$\widehat{V}_\theta(s_t^g) = \frac{1}{K}\sum_{k=1}^{K} R_c\big(f(\tau^{(k)})\big), \qquad \tau^{(k)} \sim Q_{\theta,g_{\text{rand}}}(\cdot \mid s_t^g, c). \tag{7}$$

This protocol compares how different reveal orders shape the real context and its subsequent generation, using D3LM before and after RL.

RL amplifies order dependence. Before RL, all strategies broadly gain reward as more high-activity real tokens are revealed (Figure 2a). After RL, their trajectories separate sharply, especially at steps 10–25 (Figure 2b). The mean across-strategy reward spread over this interval increases from 0.95 to 1.82. After RL, the model becomes substantially more sensitive to which real tokens are revealed early. Generation order thus provides an important degree of control over the behavior of the fine-tuned model.

Order sensitivity is linked to collapse. We next examine the sequence distributions behind these reward differences. We select contexts at step 13, where the post-RL curves show strong separation, and draw repeated completions under the common decoder. Figure 3 shows that higher-reward strategies generally produce more GC-enriched sequences with lower 3-mer entropy. For example, GT greedy yields mean reward 4.57, GC fraction 76.4%, and entropy 4.87 bits, compared with 2.42, 58.8%, and 5.54 bits for GT anti-greedy. At the order level, position selection shapes both functional reward and the severity of sequence collapse. SLM also exhibits strong order-dependent behavior (Appendix A.5). Together, the analyses motivate learning order and content jointly: free-generation rewards encourage task-relevant decisions, real-sequence learning anchors denoising dynamics to



(a) Sequence reward. (b) GC fraction. (c) 3-mer entropy.

Figure 3: **Reveal order changes the reward–collapse trade-off.** Completion distributions for RL-tuned D3LM from real contexts at step 13. Higher-reward strategies generally exhibit greater GC enrichment and lower local-pattern diversity.

natural DNA, and a content-reference penalty constrains reward-driven drift. These complementary signals form the two-stream framework developed next.

4 DP-DNA: Dual-Policy Reinforcement Learning

Section 3 identifies two coupled challenges: reward-driven updates can distort sequence content, while generation order can further amplify this distortion. We therefore introduce a dual-policy framework that jointly learns *where to generate* and *what to generate*, rather than optimizing content under a fixed generation schedule. Specifically, an order policy g_ψ selects the next masked position, while a content policy p_θ predicts the token at that position. DP-DNA trains the two policies through complementary streams: the free-generation stream optimizes task utility over jointly sampled order–content trajectories, while the real-sequence stream trains the generator to reconstruct natural sequence content and the planner to follow real-data denoising dynamics. (Figure 4).

4.1 A dual-policy generator with learned corruption

Reverse generation process. The content generator p_θ is a bidirectional D3LM. The order planner g_ψ takes the generator's contextual features as input through a lightweight Transformer head,

$$\mathbf{r}_\psi(s) = \mathrm{RateHead}_\psi(\mathrm{sg}[h_\theta(s)]), \tag{8}$$

where sg denotes stop-gradient. Thus, the planner can use the generator's representation of the current context without back-propagating planner gradients through the generator. Normalizing the positive reveal rates over masked positions gives

$$g_\psi(i \mid s) = \frac{r_{\psi,i}(s)}{\sum_{j\in\mathcal{M}(s)} r_{\psi,j}(s)}, \qquad i \in \mathcal{M}(s). \tag{9}$$

The planner selects *where* to generate and p_θ determines *what* to generate. For reveal order $\omega = (i_1, \ldots, i_T)$, their joint path probability is

$$Q_{\theta,\psi}(x, \omega \mid c) = \underbrace{\prod_{t=1}^{T} g_\psi(i_t \mid s_t)}_{a_\psi(\omega \| x, c)} \underbrace{\prod_{t=1}^{T} p_\theta(x_{i_t} \mid s_t, i_t)}_{b_\theta(x;\omega,c)}. \tag{10}$$

Forward corruption process. To learn generation orders compatible with real DNA, we introduce a training-only corruption head ϕ, following LoMDM (Hong et al., 2026b). It takes detached D3LM features of a complete real sequence and predicts position-dependent corruption rates,

$$\mathbf{e}_\phi(x) = \mathrm{RateHead}_\phi(\mathrm{sg}[h_\theta(x)]). \tag{11}$$

At noise level $u \in (0, 1]$, these rates define

$$q_\phi(m \mid x, u) = \prod_{i=1}^{L} \mathrm{Bernoulli}\left(m_i; u^{e_{\phi,i}(x)}\right), \qquad z = \mathrm{Mask}(x, m), \tag{12}$$

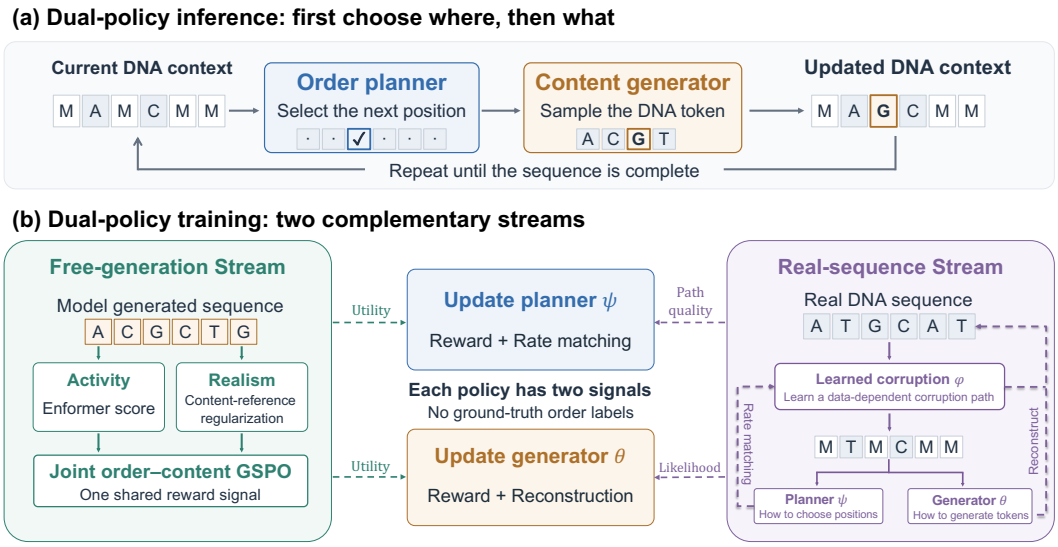


Figure 4: **Dual-policy generation with two complementary training streams.** Free-generation rewards update both policies through joint order–content GSPO and content-reference regularization. Real DNA passes through learned corruption ϕ to train content reconstruction and forward–reverse rate matching, providing the planner with a second learning signal.

where $m_i = 1$ denotes a masked position. The resulting real-DNA contexts are used below to train content reconstruction and align the forward corruption rates e_ϕ with the reverse reveal rates r_ψ. The corruption head ϕ is used only during training; generation uses p_θ and g_ψ.

4.2 Two-stream training

Free-generation stream: functional optimization. A behavior snapshot $\Omega^- = (\theta^-, \psi^-)$ samples a group of G trajectories. Each completed sequence receives target-cell reward $R_c(x)$, with content-reference regularization against a frozen domain-adapted content model p_{ref}:

$$K^-(\tau) = \sum_{t=1}^{T} \log \frac{p_{\theta^-}(x_{i_t} \mid s_t, i_t)}{p_{\text{ref}}(x_{i_t} \mid s_t, i_t)}, \qquad U^-(\tau) = R_c(x) - \beta K^-(\tau). \tag{13}$$

The leave-one-out advantage $A_n^G = U_n^- - (G-1)^{-1} \sum_{k \neq n} U_k^-$ encourages improvement relative to the other samples. The content-reference penalty βK^- compares token probabilities at the same sampled states and selected positions. It enters $\mathcal{L}_{\text{free}}$ through these utility-based advantages, which are held fixed during replay; order remains free to adapt through both streams.

We apply GSPO (Zheng et al., 2025) to the joint sampled path, replaying its recorded states and actions:

$$\rho_n^G = \exp\left[\frac{\log Q_{\theta,\psi}(\tau_n \mid c) - \log Q_{\Omega^-}(\tau_n \mid c)}{T}\right], \qquad \mathcal{L}_{\text{free}} = -\frac{T}{G} \sum_{n=1}^{G} \Phi(\rho_n^G, A_n^G), \tag{14}$$

where $\Phi(\rho, A) = \min\{\rho A, \text{clip}(\rho, 1-\epsilon, 1+\epsilon)A\}$. Order and content share one advantage, one path ratio, and one clipping decision. A successful trajectory therefore reinforces both its position and token choices.

Real-sequence stream: reconstruction and order alignment. For real DNA $x \sim p_D(\cdot \mid c)$ and $u \sim \text{Uniform}(0,1)$, learned corruption supplies z. Write $e_i = e_{\phi,i}(x)$, $r_i = r_{\psi,i}(z)$, and $\ell_i = -\log p_\theta(x_i \mid z, i)$. We adapt the LoMDM objective (Hong et al., 2026b):

$$\mathcal{L}_{\text{real}} = \mathbb{E}_{x,u,\, m \sim q_\phi}\left[\sum_{i=1}^{L} \frac{m_i}{u}\left\{\underbrace{e_i \ell_i}_{\text{content reconstruction}} + \underbrace{e_i \log(e_i/r_i) - e_i + r_i}_{\text{rate matching}}\right\}\right]. \tag{15}$$

Reconstruction teaches the generator to recover natural sequence content; rate matching trains the planner to follow the learned real-data denoising dynamics. The corruption head learns from both terms, including the effect of its masking distribution. Together, these mechanisms give the planner a real-DNA training signal complementary to functional reward.

Combined optimization. The final training loss is

$$\mathcal{L}(\theta, \psi, \phi) = \mathcal{L}_{\text{free}} + \lambda_{\text{real}} \mathcal{L}_{\text{real}}, \qquad \lambda_{\text{real}} = 0.2. \tag{16}$$

We perform two updates per rollout group, holding the free-path advantages fixed and computing the real-data objective under the same parameters. The planner thus learns task-specific orders from RL and real-sequence compatibility from rate matching, while the generator receives both reward and reconstruction supervision. More details of training/inference algorithms are given in Appendix C.

5 Experiments

5.1 Experimental settings

Datasets and evaluation. We study enhancer design for HepG2, K562, and SK-N-SH using the Gosai massively parallel reporter assay collection (Gosai et al., 2024), following the DRAKES benchmark (Wang et al., 2025). The dataset contains approximately 700,000 200-bp sequences with activity measurements in all three cell types. We evaluate four aspects: *Property* (Pred-Activity and ATAC-Acc), *Novelty* (Authenticity and Rarity), *Composition* (3-mer Corr and JASPAR Corr), and *Realism* (App-Log-Lik and Evo 2 log-likelihood (Brixi et al., 2026)). The RL reward predictor and evaluation oracle are separately trained models; their training-data separation and roles are detailed in Appendix D.1. Metric definitions are given in Appendix D.4.

Baselines. We compare with MDLM (Sahoo et al., 2024), MDLM+CG (Nisonoff et al., 2025; Wang et al., 2025), DNA-Diff (DaSilva et al., 2026), DRAKES (Wang et al., 2025), SVDD (Li et al., 2024), and D3LM+GSPO covering pretrained, guided, and reward-optimized DNA generation.

Implementation details. DP-DNA uses a domain-adapted D3LM with separate corruption and reveal-rate heads, trained independently for each target cell. Sequence-only domain adaptation constructs the frozen content reference, while separate pre-adaptation initializes the trainable model (Appendix E). We use groups of 32 free trajectories, two GSPO updates per group, and the two-stream objective described in Section 4. Table 1 reports three independent replicates of 640 generated sequences; for trainable methods, these correspond to independent training runs with different random seeds. Full architecture and optimization details are provided in Appendix D.2.

5.2 Performance on regulatory DNA design

DP-DNA achieves strong predicted functional performance while maintaining favorable sequence quality (Table 1). Across the 24 dataset–metric combinations, it ranks first in 19 and second in 4, including the highest ATAC-Acc on K562 and SK-N-SH and the highest Pred-Activity on HepG2 and SK-N-SH. It also shows consistently strong novelty, composition, and realism metrics.

Compared with D3LM+GSPO, which applies GSPO directly to D3LM with a fixed uniform generation order, DP-DNA performs better in 23 of the 24 dataset–metric combinations. Details of D3LM+GSPO model are given in Appendix D.3. These results show that DP-DNA improves both predicted function and sequence quality beyond reward optimization alone.

5.3 Ablations and learned-order analysis

To examine the contributions of the proposed training strategy and model design across different cell types, we conduct ablation studies on the enhancer-design task. Results for HepG2 and SK-N-SH are shown in Table 2, with K562 results in Appendix D.6.

Table 1: **Benchmark on three enhancer-design tasks across four evaluation aspects.** Results are reported as mean ± standard deviation across three independent runs; for trainable methods, these correspond to independent training runs with different random seeds. Bold and underline indicate the highest and second-highest means. Metric definitions are given in Appendix D.4.

Method	Property		Novelty		Composition		Realism	
	Pred-Activity ↑	ATAC-Acc (%) ↑	Authen-ticity ↑	Rarity (IQR) ↑	3-mer Corr ↑	JASPAR Corr ↑	App-Log-Lik ↑	Evo 2 log-lik (nats/bp) ↑
HepG2								
MDLM	0.156 ±0.025	2.290 ±0.59	0.7771 ±0.0326	-0.1494 ±0.0325	0.004 ±0.019	0.3371 ±0.0560	-263.9 ±0.4	-1.3333 ±0.0022
MDLM+CG	4.517 ±0.151	48.28 ±2.03	0.8094 ±0.0135	0.2121 ±0.0159	0.855 ±0.008	0.9161 ±0.0038	-265.8 ±0.2	-1.3417 ±0.0013
DNA-Diff	0.338 ±0.029	3.594 ±0.156	0.7682 ±0.0133	-0.1494 ±0.0325	0.505 ±0.011	0.7862 ±0.0070	-264.3 ±0.1	-1.3372 ±0.0007
DRAKES	5.658 ±0.068	**91.82** ±0.477	0.8693 ±0.0119	1.1978 ±0.0789	0.842 ±0.007	0.9063 ±0.0039	-267.2 ±0.1	-1.3386 ±0.0006
SVDD	6.830 ±0.024	46.25 ±1.88	0.8937 ±0.0083	0.8408 ±0.0264	0.386 ±0.042	0.4503 ±0.0188	-253.5 ±0.02	-1.2700 ±0.0002
D3LM+GSPO	4.671 ±0.211	54.27 ±5.99	0.8118 ±0.0136	1.2891 ±0.065	0.883 ±0.010	0.8643 ±0.0054	-259.6 ±1.69	-1.3318 ±0.0103
DP-DNA	**7.132** ±0.019	89.59 ±0.81	**0.8991** ±0.0124	**1.4884** ±0.0001	**0.897** ±0.0004	**0.9279** ±0.0018	**-253.0** ±0.5	**-1.2444** ±0.0029
K562								
MDLM	0.106 ±0.029	1.98 ±0.09	0.7771 ±0.0326	-0.0336 ±0.0291	-0.024 ±0.019	0.3371 ±0.0560	-263.6 ±0.7	-1.3333 ±0.0022
MDLM+CG	6.433 ±0.052	91.09 ±1.02	0.8104 ±0.0063	0.4132 ±0.0282	0.923 ±0.008	0.8933 ±0.0009	-263.9 ±0.4	-1.3318 ±0.0012
DNA-Diff	0.163 ±0.013	1.667 ±0.705	0.7948 ±0.0186	0.0167 ±0.0290	-0.217 ±0.013	0.2388 ±0.0097	-265.7 ±0.1	-1.3397 ±0.0011
DRAKES	7.803 ±0.009	97.86 ±0.090	0.8964 ±0.0063	1.3652 ±0.4015	0.714 ±0.013	0.8325 ±0.0081	-272.3 ±0.4	-1.3410 ±0.0014
SVDD	**9.479** ±0.056	96.51 ±0.48	0.8484 ±0.0311	0.7670 ±0.0275	**0.950** ±0.001	0.8677 ±0.0065	-253.2 ±0.4	-1.2710 ±0.0016
D3LM+GSPO	6.045 ±0.156	65.03 ±3.00	0.8892 ±0.0116	1.2822 ±0.033	0.897 ±0.002	0.9074 ±0.0021	-262.8 ±1.26	-1.3117 ±0.0099
DP-DNA	8.402 ±0.019	**99.17** ±0.18	**0.8979** ±0.0059	**1.3890** ±0.0793	0.872 ±0.001	**0.9122** ±0.0027	**-252.1** ±0.3	**-1.2638** ±0.0004
SK-N-SH								
MDLM	0.112 ±0.026	2.14 ±0.50	0.7771 ±0.0326	-0.0884 ±0.0307	-0.079 ±0.020	0.3371 ±0.0560	-264.0 ±0.6	-1.3333 ±0.0022
MDLM+CG	5.046 ±0.101	47.66 ±2.60	0.8281 ±0.0095	0.2273 ±0.0602	0.796 ±0.012	0.9079 ±0.0074	-266.9 ±0.7	-1.3463 ±0.0018
DNA-Diff	0.402 ±0.027	3.385 ±0.238	0.8021 ±0.0106	-0.0177 ±0.0306	0.077 ±0.027	0.5239 ±0.0184	-267.4 ±0.1	-1.3401 ±0.0022
DRAKES	7.995 ±0.072	98.32 ±0.502	0.7042 ±0.0104	1.1400 ±0.1738	0.674 ±0.009	0.8683 ±0.0082	-276.8 ±0.4	-1.3681 ±0.0004
SVDD	7.989 ±0.040	34.64 ±1.02	0.8682 ±0.0206	1.2505 ±0.0562	-0.054 ±0.006	0.1629 ±0.0100	**-248.9** ±0.6	**-1.2471** ±0.0011
D3LM+GSPO	3.410 ±0.657	67.59 ±10.33	0.8182 ±0.0077	0.5760 ±0.093	0.905 ±0.022	0.8933 ±0.0034	-271.20 ±2.04	-1.3619 ±0.0121
DP-DNA	**8.732** ±0.008	**98.91** ±0.24	**0.8880** ±0.0050	**1.2791** ±0.0574	**0.956** ±0.003	**0.9351** ±0.0017	-261.6 ±0.2	-1.3094 ±0.0011

Table 2: **Ablation study on HepG2 and SK-N-SH enhancer-design tasks.** Differences relative to the full DP-DNA model are shown in parentheses. Metric definitions are given in Appendix D.4.

Ablation	HepG2				SK-N-SH			
	Pred-Activity ↑	ATAC-Acc (%) ↑	3-mer Corr ↑	Evo 2 LL ↑	Pred-Activity ↑	ATAC-Acc (%) ↑	3-mer Corr ↑	Evo 2 LL ↑
Training								
w/o free-generation stream	0.1305 (-7.002)	1.771 (-87.82)	-0.025 (-0.922)	-1.330 (-0.086)	0.1441 (-8.588)	1.302 (-97.61)	-0.207 (-1.163)	-1.327 (-0.018)
w/o real-sequence stream	7.016 (-0.116)	90.36 (+0.77)	0.872 (-0.025)	-1.385 (-0.141)	8.672 (-0.06)	99.22 (+0.31)	0.938 (-0.018)	-1.430 (-0.121)
Model Arch								
w/o planner	7.001 (-0.131)	87.72 (-1.87)	0.880 (-0.017)	-1.393 (-0.149)	8.586 (-0.146)	97.71 (-1.20)	0.955 (-0.001)	-1.436 (-0.127)
w/o generator-feature input	6.798 (-0.334)	88.75 (-0.84)	0.901 (+0.004)	-1.250 (-0.006)	8.522 (-0.21)	97.76 (-1.15)	0.954 (-0.002)	-1.330 (-0.021)
w/o corruption model ϕ	6.753 (-0.379)	89.53 (-0.06)	0.890 (-0.007)	-1.257 (-0.013)	8.602 (-0.13)	98.33 (-0.58)	0.939 (-0.017)	-1.325 (-0.016)
DP-DNA	**7.132**	**89.59**	**0.897**	**-1.244**	**8.732**	**98.91**	**0.956**	**-1.309**

The two streams serve complementary roles. Removing the free-generation stream nearly eliminates functional adaptation, causing large drops in both Pred-Activity and accessibility. Since this ablation receives no downstream reward signal, the substantial loss in functional performance further demonstrates the effectiveness of RL-based task adaptation. In contrast, removing the real-sequence stream largely preserves accessibility but reduces Evo 2 likelihood and 3-mer correlation. The two streams therefore provide complementary signals: reward-driven exploration promotes function, while real-sequence learning helps preserve natural sequence structure.

Learned order and corruption contribute to functional design. Table 2 shows that removing the planner reduces Pred-Activity and Evo 2 likelihood on both HepG2 and SK-N-SH. Removing the generator-derived features from the planner causes larger activity drops, indicating that context-dependent information is important for learning effective generation schedules. Removing learned corruption also lowers activity and degrades several sequence-quality metrics, supporting the joint learning of generation order and real-data corruption.

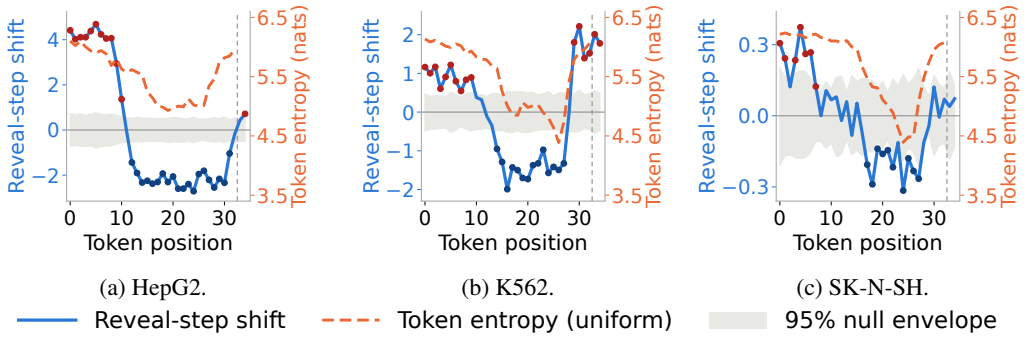


Figure 5: **Learned position preferences in DP-DNA.** Blue shows the average reveal-step shift at each position relative to uniform decoding, with negative values indicating earlier decoding under the learned planner. Orange shows the empirical token (6-mer) entropy at each position. Gray bands denote 95% sign-flip null envelopes.

What does the planner prioritize? Figure 5 reveals structured, cell-dependent generation preferences. HepG2 preferentially reveals the central-to-right region, K562 emphasizes a central block, and SK-N-SH shows a weaker but related central preference. The learned policy is therefore not reducible to a fixed directional order.

Reveal-step shifts are also positively correlated with position-wise empirical token entropy across all three cell types. Lower-entropy positions tend to be generated earlier, while more variable positions are delayed. The planner therefore learns task-dependent context construction associated with local sequence variability, consistent with the context-planner ablation above.

5.4 Mitigating sequence collapse during RL

We revisit the diagnostics from Figure 1 to test whether DP-DNA improves reward without the same compositional collapse. We apply the same HepG2 diagnostic protocol and reward-matched reference construction as in Figure 1 to DP-DNA, replacing only the vanilla GSPO training procedure with DP-DNA. Figure 6 shows that DP-DNA reaches substantially higher reward while keeping its GC fraction close to the reward-matched real reference and maintaining 3-mer entropy near the real-data level.

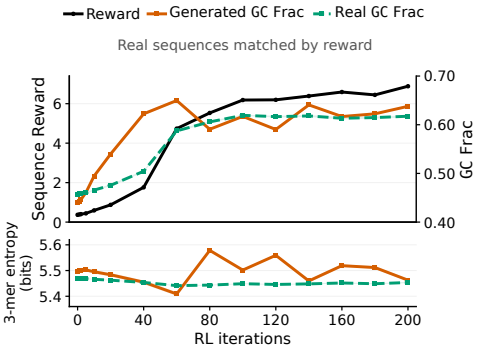


Figure 6: **DP-DNA improves reward while preserving local-pattern diversity.** Reward, GC fraction, and 3-mer entropy during HepG2 training; real sequences are reward matched.

At iteration 100, DP-DNA reaches a mean reward of 6.18 with 61.7% GC and 3-mer entropy of 5.50 bits. In contrast, the conventional RL model in Figure 1 shows pronounced GC enrichment and lower entropy at a lower reward. The same pattern remains at later training stages. DP-DNA therefore achieves higher reward with reduced compositional collapse and preserved local-pattern (3-mer) diversity, mitigating RL-induced sequence collapse.

6 Conclusion

We investigated how RL affects DNA discrete diffusion generation and identified a connection between sequence collapse and generation order. Motivated by this finding, we introduced DP-DNA, which jointly learns what and where to generate through complementary free-generation and real-sequence streams. Free-generation stream promotes functional design under content-reference regularization, while the real-sequence stream encourages generation orders compatible with real DNA. Across three enhancer-design tasks, DP-DNA achieves strong predicted activity while preserving favorable sequence-quality characteristics. Ablations and diagnostic analyses support the roles of adaptive planning and the two training streams, and show that reward can be improved with markedly less GC distortion and preserved 3-mer diversity. Together, these results demonstrate the value of task-adaptive generation order for RL-based DNA design.

Reproducibility Statement

To support reproducibility, Appendix A describes the experimental procedures and training setup used in the behavioral analyses. Appendix C provides the optimization objectives, gradient estimators, and training and inference algorithms. Appendix D documents the data, reward and evaluation models, baseline configurations, evaluation metrics, ablations, and diagnostic experiments, with architecture and optimization details given in Appendix D.2. Appendix E describes domain adaptation, pre-adaptation, and reference selection. We will publicly release the code for model training, evaluation, and analysis.

LLM Usage Disclosure

We used large language models to assist with refining the methodological formulation and mathematical derivations, providing feedback on experimental design, and interpreting results. LLM-based agents also assisted with developing experimental and analysis code. Additional uses included drafting and organizing the manuscript, translating and editing research-related text, identifying and summarizing relevant literature, preparing and revising scientific figures and tables, and formatting references. The authors are responsible for checking the AI-assisted mathematical arguments, code, and results against the underlying derivations, implementations, and experiment records. We take responsibility for the final manuscript and research artifacts, including all AI-assisted text, claims, code, and figures.

References

Ahmed Alaa, Boris Van Breugel, Evgeny S. Saveliev, and Mihaela van der Schaar. How faithful is your synthetic data? Sample-level metrics for evaluating and auditing generative models. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 290–306. PMLR, 2022. URL https://proceedings.mlr.press/v162/alaa22a.html.

Alan N. Amin, Nate Gruver, and Andrew Gordon Wilson. Why masking diffusion works: Condition on the jump schedule for improved discrete diffusion. In *Advances in Neural Information Processing Systems*, volume 38, pp. 33783–33813, 2025. doi: 10.52202/085713-1135. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/3086ede0108b174087add7f1ce55f827-Abstract-Conference.html.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pp. 17981–17993, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/958c530554f78bcd8e97125b70e6973d-Abstract.html.

Pavel Avdeyev, Chenlai Shi, Yuhao Tan, Kseniia Dudnyk, and Jian Zhou. Dirichlet diffusion score model for biological sequence generation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 1276–1301. PMLR, 2023. URL https://proceedings.mlr.press/v202/avdeyev23a.html.

Žiga Avsec, Vikram Agarwal, Daniel Visentin, Joseph R. Ledsam, Agnieszka Grabska-Barwinska, Kyle R. Taylor, Yannis Assael, John Jumper, Pushmeet Kohli, and David R. Kelley. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature Methods*, 18 (10):1196–1203, 2021. doi: 10.1038/s41592-021-01252-x. URL https://www.nature.com/articles/s41592-021-01252-x.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, pp. 4965–4987, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/14f75513f0f1ca01de1e826b52e6b840-Abstract-Conference.html.

Garyk Brixi, Matthew G. Durrant, Jerome Ku, Mohsen Naghipourfar, Michael Poli, Gwanggyu Sun, Greg Brockman, Daniel Chang, Alison Fanton, Gabriel A. Gonzalez, Samuel H. King, David B. Li, Aditi T. Merchant, Eric Nguyen, Chiara Ricci-Tam, David W. Romero, Jonathan C. Schmok, Ali Taghibakhshi, Anton Vorontsov, Brandon Yang, Myra Deng, Liv Gorton, Nam Nguyen, Nicholas K. Wang, Michael T. Pearce, Elana Simon, Etowah Adams, Zachary J. Amador, Euan A. Ashley, Stephen A. Baccus, Haoyu Dai, Steven Dillmann, Stefano Ermon, Daniel Guo, Michael H. Herschl, Rajesh Ilango, Ken Janik, Amy X. Lu, Reshma Mehta, Mohammad R. K. Mofrad, Madelena Y. Ng, Jaspreet Pannu, Christopher Ré, John St. John, Jeremy Sullivan, Joseph Tey, Ben Viggiano, Kevin Zhu, Greg Zynda, Daniel Balsam, Patrick Collison, Anthony B. Costa, Tina Hernandez-Boussard, Eric Ho, Ming-Yu Liu, Thomas McGrath, Kimberly Powell, Sudarshan Pinglay, Dave P. Burke, Hani Goodarzi, Patrick D. Hsu, and Brian L. Hie. Genome modelling and design across all domains of life with Evo 2. *Nature*, 652(8112):1349–1361, 2026. doi: 10.1038/s41586-026-10176-5. URL https://www.nature.com/articles/s41586-026-10176-5.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Tom Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 28266–28279. Curran Associates, Inc., 2022. doi: 10.52202/068431-2049. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/b5b528767aa35f5b1a60fe0aaeca0563-Abstract-Conference.html.

Hanqun Cao, Haosen Shi, Chenyu Wang, Sinno Jialin Pan, and Pheng-Ann Heng. GLID^{2}E: A gradient-free lightweight fine-tune approach for discrete biological sequence design. In *Advances in Neural Information Processing Systems*, volume 38, pp. 139875–139898, 2025. doi: 10.52202/085713-4672. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/cc84bfabe6389d8883fc2071c848f62a-Abstract-Conference.html.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. MaskGIT: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11315–11325, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Chang_MaskGIT_Masked_Generative_Image_Transformer_CVPR_2022_paper.html.

Xingyu Chen, Shihao Ma, Runsheng Lin, Jiecong Lin, and Bo Wang. Ctrl-DNA: Controllable cell-type-specific regulatory DNA design via constrained RL. In *Advances in Neural Information Processing Systems*, volume 38, pp. 14037–14065, 2025. doi: 10.52202/085713-0472. URL https://papers.neurips.cc/paper_files/paper/2025/hash/14c018d2e72c521605b0567029ef0efb-Abstract-Conference.html.

Lucas Ferreira DaSilva, Simon Senan, Judith F. Kribelbauer-Swietek, Zain Munir Patel, Lithin Karmel Louis, Aniketh Janardhan Reddy, Sameer Gabbita, Jonathan D. Rosen, Zach Nussbaum, César Miguel Valdez Córdova, Aaron Wenteler, Noah Weber, Tin M. Tunjic, Martino Mansoldo, Talha Ahmad Khan, Gue-Ho Hwang, Vincent Gardeux, David T. Humphreys, Cameron Smith, Matei Bejan, Peter Bromley, Will Connell, Bart Deplancke, Michael I. Love, Emily S. Wong, Wouter Meuleman, and Luca Pinello. Designing synthetic regulatory elements using the generative AI framework DNA-Diffusion. *Nature Genetics*, 58(1):180–194, 2026. doi: 10.1038/s41588-025-02441-6. URL https://www.nature.com/articles/s41588-025-02441-6.

Bernardo P. de Almeida, Franziska Reiter, Michaela Pagani, and Alexander Stark. DeepSTARR predicts enhancer activity from DNA sequence and enables the de novo design of synthetic enhancers. *Nature Genetics*, 54(5):613–624, 2022. doi: 10.1038/s41588-022-01048-5. URL https://www.nature.com/articles/s41588-022-01048-5.

Sager J. Gosai, Rodrigo I. Castro, Natalia Fuentes, John C. Butts, Kousuke Mouri, Michael Alasoadura, Susan Kales, Thanh Thanh L. Nguyen, Ramil R. Noche, Arya S. Rao, Mary T. Joy, Pardis C. Sabeti, Steven K. Reilly, and Ryan Tewhey. Machine-guided design of cell-type-targeting cis-regulatory elements. *Nature*, 634(8036):1211–1220, 2024. doi: 10.1038/s41586-024-08070-z. URL https://www.nature.com/articles/s41586-024-08070-z.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://www.nature.com/articles/s41586-025-09422-z.

Jiyeon Han, Hwanil Choi, Yunjey Choi, Junho Kim, Jung-Woo Ha, and Jaesik Choi. Rarity score: A new metric to evaluate the uncommonness of synthesized images. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=JTGimap_-F.

Chunsan Hong, Seonho An, Min-Soo Kim, and Jong Chul Ye. Improving discrete diffusion unmasking policies beyond explicit reference policies. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=on6cb46OhD.

Chunsan Hong, Sanghyun Lee, and Jong Chul Ye. Unifying masked diffusion models with various generation orders and beyond. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026b. URL https://arxiv.org/abs/2602.02112.

Zemin Huang, Zhiyang Chen, Zijun Wang, Tiancheng Li, and Guo-Jun Qi. Reinforcing the diffusion chain of lateral thought with diffusion language models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-5105. URL https://papers.nips.cc/paper_files/paper/2025/hash/dffd1c523512e557f4e75e8309049213-Abstract-Conference.html.

Natasha Jaques, Shixiang Gu, Dzmitry Bahdanau, José Miguel Hernández-Lobato, Richard E. Turner, and Douglas Eck. Sequence tutor: Conservative fine-tuning of sequence generation models with KL-control. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1645–1654. PMLR, 2017. URL https://proceedings.mlr.press/v70/jaques17a.html.

Nathan Killoran, Leo J. Lee, Andrew Delong, David Duvenaud, and Brendan J. Frey. Generating and designing DNA with deep generative models. *arXiv preprint arXiv:1712.06148*, 2017. doi: 10.48550/arXiv.1712.06148. URL https://arxiv.org/abs/1712.06148.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham M. Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 30749–30768. PMLR, 2025. URL https://proceedings.mlr.press/v267/kim25ah.html.

Avantika Lal, David Garfield, Tommaso Biancalani, and Gokcen Eraslan. Designing realistic regulatory DNA with autoregressive language models. *Genome Research*, 34(9):1411–1420, 2024. doi: 10.1101/gr.279142.124. URL https://genome.cshlp.org/content/34/9/1411.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. Diffusion-LM improves controllable text generation. In *Advances in Neural Information Processing Systems*, volume 35, pp. 4328–4343. Curran Associates, Inc., 2022. doi: 10.52202/068431-0313. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/1be5bc25d50895ee656b8c2d9eb89d6a-Abstract-Conference.html.

Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gokcen Eraslan, Surag Nair, Tommaso Biancalani, Shuiwang Ji, Aviv Regev, Sergey Levine, and Masatoshi Uehara. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. *arXiv preprint arXiv:2408.08252*, 2024. doi: 10.48550/arXiv.2408.08252. URL https://arxiv.org/abs/2408.08252.

Johannes Linder and Georg Seelig. Fast activation maximization for molecular sequence design. *BMC Bioinformatics*, 22(1):510, 2021. doi: 10.1186/s12859-021-04437-5. URL https://link.springer.com/article/10.1186/s12859-021-04437-5.

Johannes Linder, Nicholas Bogard, Alexander B. Rosenberg, and Georg Seelig. A generative neural network for maximizing fitness and diversity of synthetic DNA and protein sequences. *Cell Systems*, 11(1):49–62.e16, 2020. doi: 10.1016/j.cels.2020.05.007. URL https://www.cell.com/cell-systems/fulltext/S2405-4712(20)30192-7.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32819–32848. PMLR, 2024. URL https://proceedings.mlr.press/v235/lou24a.html.

Zanlin Ni, Shenzhi Wang, Yang Yue, Tianyu Yu, Weilin Zhao, Yeguo Hua, Tianyi Chen, Jun Song, Cheng Yu, Bo Zheng, and Gao Huang. The flexibility trap: Rethinking the value of arbitrary order in diffusion language models. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2601.15165.

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. In *Advances in Neural Information Processing Systems*, volume 38, pp. 50608–50646, 2025. doi: 10.52202/085713-1689. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/48b383b24230e0e6e649d9c98dae4d8c-Abstract-Conference.html.

Hunter Nisonoff, Junhao Xiong, Stephan Allenspach, and Jennifer Listgarten. Unlocking guidance for discrete state-space diffusion and flow models. In *The Thirteenth International Conference on Learning Representations*, pp. 36052–36106, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/597254dc45be8c166d3ccf0ba2d56325-Abstract-Conference.html.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. doi: 10.52202/068431-2011. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

Fred Zhangzhi Peng, Zachary Bezemek, Sawan Patel, Jarrid Rector-Brooks, Sherwood Yao, Avishek Joey Bose, Alexander Tong, and Pranam Chatterjee. Path planning for masked diffusion model sampling. *arXiv preprint arXiv:2502.03540*, 2025. doi: 10.48550/arXiv.2502.03540. URL https://arxiv.org/abs/2502.03540.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 130136–130184, 2024. doi: 10.52202/079017-4135. URL https://papers.nips.cc/paper_files/paper/2024/hash/eb0b13cc515724ab8015bc978fdde0ad-Abstract-Conference.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. doi: 10.48550/arXiv.1707.06347. URL https://arxiv.org/abs/1707.06347.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In *Advances in Neural Information Processing Systems*, volume 37, pp. 103131–103167. Curran Associates, Inc., 2024. doi: 10.52202/079017-3277. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/bad233b9849f019aead5e5cc60cef70f-Abstract-Conference.html.

Yuxuan Song, Zhe Zhang, Yu Pei, Jingjing Gong, Qiying Yu, Zheng Zhang, Mingxuan Wang, Hao Zhou, Jingjing Liu, and Wei-Ying Ma. Shortlisting model: A streamlined simplex diffusion for discrete variable generation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 46781–46812, 2025. doi: 10.52202/085713-1558. URL https://papers.neurips.cc/paper_files/paper/2025/hash/42c4ca570eec68981c20094de1cc760e-Abstract-Conference.html.

Hannes Stark, Bowen Jing, Chenyu Wang, Gabriele Corso, Bonnie Berger, Regina Barzilay, and Tommi Jaakkola. Dirichlet flow matching with applications to DNA sequence design. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 46495–46513. PMLR, 2024. URL https://proceedings.mlr.press/v235/stark24b.html.

Sophia Tang, Yinuo Zhang, Alexander Tong, and Pranam Chatterjee. Gumbel-softmax flow matching with straight-through guidance for controllable biological sequence generation. *arXiv preprint arXiv:2503.17361*, 2025. doi: 10.48550/arXiv.2503.17361. URL https://arxiv.org/abs/2503.17361.

Eeshit Dhaval Vaishnav, Carl G. de Boer, Jennifer Molinet, Moran Yassour, Lin Fan, Xian Adiconis, Dawn A. Thompson, Joshua Z. Levin, Francisco A. Cubillos, and Aviv Regev. The evolution, evolvability and engineering of gene regulatory DNA. *Nature*, 603(7901):455–463, 2022. doi: 10.1038/s41586-022-04506-6. URL https://www.nature.com/articles/s41586-022-04506-6.

Chenyu Wang, Masatoshi Uehara, Yichun He, Amy Wang, Avantika Lal, Tommi Jaakkola, Sergey Levine, Aviv Regev, Hanchen Wang, and Tommaso Biancalani. Fine-tuning discrete diffusion models via reward optimization with applications to DNA and protein design. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=G328D1xt4W.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL https://link.springer.com/article/10.1007/BF00992696.

Zhao Yang, Bing Su, Chuan Cao, and Ji-Rong Wen. Regulatory DNA sequence design with reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=F4IMiNhim1.

Zhao Yang, Hengchang Liu, Chuan Cao, and Bing Su. D3LM: A discrete DNA diffusion language model for bidirectional DNA understanding and generation. In *MLGenX 2026 Workshop*, 2026. URL https://openreview.net/forum?id=J2FJJ72Psj.

Oussama Zekri and Nicolas Boullé. Fine-tuning discrete diffusion models with policy gradient methods. In *Advances in Neural Information Processing Systems*, volume 38, pp. 152868–152906, 2025. doi: 10.52202/085713-5112. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/e053a2ffa2a7f5156e2b9a852fde6d45-Abstract-Conference.html.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025. doi: 10.48550/arXiv.2507.18071. URL https://arxiv.org/abs/2507.18071.

APPENDIX CONTENTS

A Additional Details of the Behavioral Analysis **17**
A.1 Experimental setting and RL training 17
A.2 Sequence statistics and reward-matched references 17
A.3 Reveal strategies and conditional completion 17
A.4 Generation order for the ShortListing Model 18
A.5 Results for SLM 18

B Related Work **19**
B.1 Reward-based adaptation of generative models 19
B.2 Discrete generative models for regulatory DNA design 20
B.3 Generation order in masked diffusion models 20

C Two-Stream Optimization and Algorithms **21**
C.1 Policy parameterization and gradient decomposition 21
C.2 Learned masking and the two-sample estimator 22
C.3 Training and inference 22

D Experimental Details and Evaluation Protocol **23**
D.1 Data, reward predictor, and evaluation oracle 23
D.2 Architecture and training configuration 24
D.3 D3LM+GSPO Baseline 24
D.4 Evaluation metrics 25
D.5 MDLM baseline across cell-specific tasks 27
D.6 Additional Ablation Results on K562 27
D.7 Learned-order diagnostic 28

E Sequence Domain Adaptation and Pre-Adaptation **29**
E.1 Domain adaptation and reference selection 29
E.2 Uniform-corruption pre-adaptation 29

A Additional Details of the Behavioral Analysis

A.1 Experimental setting and RL training

The analysis follows the 200-base Gosai enhancer-design setting of DRAKES (Wang et al., 2025), targeting HepG2. A frozen Enformer-based predictor provides the terminal sequence reward. D3LM starts from the released D3LM-scratch checkpoint (Yang et al., 2026); SLM starts from a Gosai-adapted prior obtained after 5,000 reward-free adaptation steps and validation-based selection (Song et al., 2025). Both models are fine-tuned with GSPO using three independent training seeds.

D3LM encodes each sequence as 33 non-overlapping 6-mer tokens followed by two single-base tokens, giving 35 generation events. At each event, an unresolved position is chosen uniformly and a token is sampled from its legal vocabulary at temperature 1. SLM uses the candidate-removal sampler defined in Appendix A.4. Each RL iteration collects a fresh group of sequences, computes group-normalized reward advantages, and replays the recorded paths for the GSPO update.

A.2 Sequence statistics and reward-matched references

For a decoded sequence $b = (b_1, \ldots, b_n)$, $n = 200$, over $\mathcal{B} = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$, we compute

$$\mathtt{GC}\ \mathrm{Frac}(b) = \frac{1}{n}\sum_{j=1}^{n} \mathbb{I}[b_j \in \{\mathtt{G}, \mathtt{C}\}], \qquad H_3(b) = -\sum_{u \in \mathcal{B}^3} \widehat{p}_b(u) \log_2 \widehat{p}_b(u), \tag{17}$$

where $\widehat{p}_b(u)$ is the frequency of u among the overlapping three-base windows. The 3-mer entropy H_3 is measured in bits and summarizes local-pattern diversity.

Checkpoint statistics use 2,048 generated sequences per training seed. For each generated sequence x, we select up to 32 neighbors from 40,000 held-out real sequences satisfying $|R_c(z) - R_c(x)| \leq 0.15$, using the same reward predictor. Reference statistics are averaged over matched neighbors and then generated sequences. Training-curve bands are 95% hierarchical bootstrap intervals over training runs and sequence draws. All sequence statistics are evaluated on decoded bases.

A.3 Reveal strategies and conditional completion

For each high-activity real target x, a reveal strategy constructs contexts recursively:

$$i_{t+1} \sim g(\cdot \mid s_t^g), \qquad s_{t+1}^g = \mathrm{Write}(s_t^g, i_{t+1}, x_{i_{t+1}}), \tag{18}$$

starting from a fully masked sequence. Table 3 defines the six position rules. The adaptive rules are recomputed after each reveal. Oracle-guided GT greedy and GT anti-greedy probe contrasting extremes of true-token predictability, helping characterize how reveal order influences the resulting completions.

Table 3: Position policies used to construct real contexts. For deterministic rules, $g(i \mid s) = \mathbb{I}[i = j_g(s)]$; GT rules additionally use the real target x as an oracle.

Strategy	Position selection
Random	$g(i \mid s) = 1/\lvert\mathcal{M}(s)\rvert,\ i \in \mathcal{M}(s)$
Left2Right	$j_g(s) = \min \mathcal{M}(s)$
Right2Left	$j_g(s) = \max \mathcal{M}(s)$
Confidence Max	$j_g(s) = \arg\max_{i \in \mathcal{M}(s)} \max_v p_\theta(v \mid s, i)$
GT greedy	$j_g(s, x) = \arg\max_{i \in \mathcal{M}(s)} \log p_\theta(x_i \mid s, i)$
GT anti-greedy	$j_g(s, x) = \arg\min_{i \in \mathcal{M}(s)} \log p_\theta(x_i \mid s, i)$

The D3LM value curves use 1,024 held-out high-activity targets. At each reveal level, we sample 16 completions under the same random-order decoder and estimate their mean reward using Eq. (7). Post-RL results combine three independently trained models. Shading in the value curves represents pointwise mean ± 1.96 standard errors over evaluated contexts. For the distribution plots, we choose step 13, where the strategies show the strongest reward separation, to examine their distinct generation behavior. We use 512 targets and 32 completions per context and training seed.

A.4 Generation order for the ShortListing Model

SLM progressively prunes candidate sets (Song et al., 2025). Let $C_{j,e} \subseteq \mathcal{B}$ be the candidates at base position j after removal event e, initially $C_{j,0} = \mathcal{B}$. We define generation order by the events at which positions commit to singleton sets:

$$e_j = \min\{e : |C_{j,e}| = 1\}, \qquad \omega = \operatorname{argsort}_{j=1}^{200} e_j. \tag{19}$$

Each event removes one candidate, giving 600 removals per sequence. With $k_j = |C_j|$ and model probabilities normalized over C_j, the position and removal rules are

$$g_{\text{haz}}(j \mid s) = \frac{k_j - 1}{\sum_m (k_m - 1)}, \tag{20}$$

$$q_\theta(\text{remove } b \mid s, j) = (1 - \zeta)\frac{1 - p_\theta(b \mid s, j)}{k_j - 1} + \frac{\zeta}{k_j}, \quad b \in C_j,\ k_j > 1, \tag{21}$$

with numerical-support mixture $\zeta = 10^{-6}$. Singleton positions remain fixed.

In the real-context experiment, selecting position j commits it to the true base by setting $C_j = \{x_j\}$. After k commitments, the remaining positions retain all four candidates. The strategies in Table 3 act on uncommitted positions, with Confidence Max and the GT strategies using model base probabilities. Every context is then completed using the same candidate-count hazard and removal rules above. This makes singleton-commitment order the SLM counterpart of D3LM's unmasking order.

A.5 Results for SLM

RL-induced degeneration. Figure 7 shows reward rising from 0.49 to 5.88 and GC fraction from 45.8% to 77.6%, compared with 61.5% for reward-matched real DNA at the endpoint. Generated 3-mer entropy falls from 5.48 to 4.95 bits, while the matched reference remains near 5.44 bits. **RL-induced compositional skew and reduced pattern diversity extend to the candidate-pruning model family.**

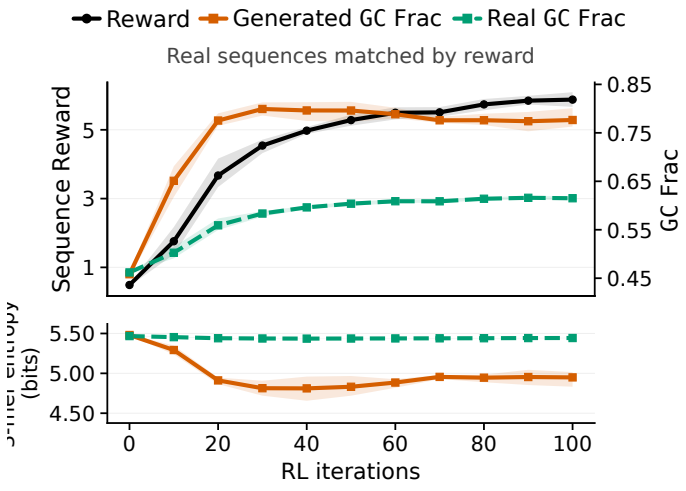
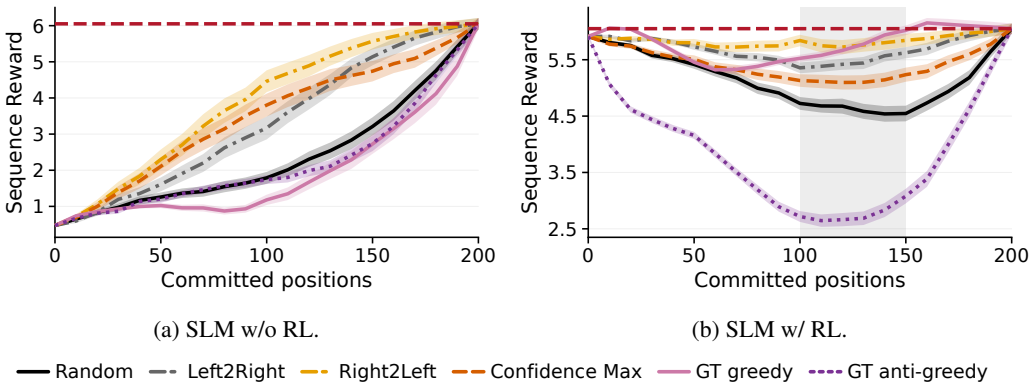


Figure 7: **SLM also develops compositional collapse during RL.** Reward, generated and reward-matched real GC fractions, and 3-mer entropy across three GSPO runs. Shading shows 95% bootstrap intervals.

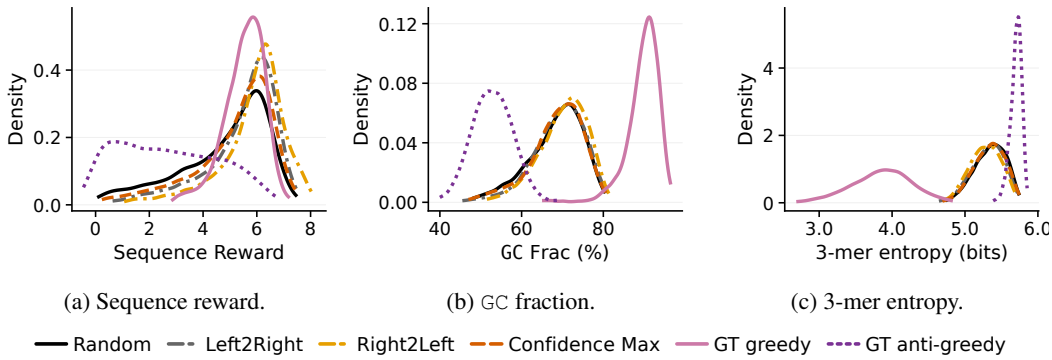
Order dependence before and after RL. We use 128 high-activity real targets, evaluating contexts at 10-position intervals with 16 completions per context. Figure 8 shows large differences in expected reward across commitment strategies, both before and after RL. **Different singleton-commitment orders substantially change SLM's generation behavior.**



(a) SLM w/o RL. (b) SLM w/ RL.

Figure 8: **SLM exhibits strong commitment-order dependence.** Real contexts constructed with different strategies are completed by a common hazard decoder. Shading shows pointwise uncertainty bands; post-RL results combine three training runs. The dashed line indicates the targets' mean reward.

Order-dependent collapse. We inspect the completion distributions at $k = 100$, where the strategies show strong separation, drawing 32 completions per context and training seed. Figure 9 reveals distinct reward–composition profiles. GT greedy yields mean reward 5.52, GC fraction 89.6%, and entropy 3.86 bits; GT anti-greedy yields 2.72, 52.9%, and 5.70 bits. Right2Left produces another profile, with reward 5.84, GC fraction 70.1%, and entropy 5.28 bits. **SLM independently confirms that generation order strongly shapes both predicted function and sequence statistics.** Together with D3LM, these results motivate jointly adapting order and content under functional and realism objectives.



(a) Sequence reward. (b) GC fraction. (c) 3-mer entropy.

Figure 9: **Commitment order changes SLM's reward–collapse trade-off.** Completion distributions from real contexts with 100 committed positions, using the same hazard decoder across reveal strategies.

B Related Work

B.1 Reward-based adaptation of generative models

Score-function estimators such as REINFORCE provide a foundation for optimizing stochastic generators using sampled rewards (Williams, 1992). Sequence Tutor applies KL-control to improve generated sequences while retaining proximity to a pretrained sequence model (Jaques et al., 2017), and InstructGPT uses a learned preference reward to adapt language models with policy optimization (Ouyang et al., 2022). For diffusion models, DDPO treats the denoising trajectory as a sequential decision process and optimizes downstream rewards with policy gradients (Black et al., 2024). SEPO develops policy-gradient fine-tuning for discrete diffusion with non-differentiable rewards, including applications to DNA and text (Zekri & Boullé, 2025). These studies establish reward-driven adaptation and reference regularization as general tools. Our work focuses on order-aware

DNA post-training, combining task-reward optimization with real-sequence supervision to guide both position selection and sequence content.

B.2 Discrete generative models for regulatory DNA design

Predictor-guided sequence design has developed alongside generative modeling. Early work combined generative adversarial networks with activation maximization to generate DNA with desired predicted properties (Killoran et al., 2017). Deep Exploration Networks jointly optimized predicted fitness and sequence diversity (Linder et al., 2020), while Fast SeqProp improved gradient-based sequence optimization through normalized sequence logits and straight-through sampling (Linder & Seelig, 2021). Sequence-to-function models have also supported experimentally grounded regulatory design: DeepSTARR learned enhancer activity and motif-context dependencies and enabled the design of synthetic enhancers (de Almeida et al., 2022), and large-scale yeast promoter measurements supported models of regulatory fitness landscapes and expression engineering (Vaishnav et al., 2022). These studies motivate evaluating designed sequences along both functional and distributional dimensions.

Non-autoregressive generative models are appealing for DNA because they can exploit bidirectional context without imposing a fixed left-to-right factorization. Diffusion-LM demonstrates controllable text generation through denoising continuous token embeddings (Li et al., 2022). In discrete state spaces, general foundations include structured noising in D3PM (Austin et al., 2021), continuous-time Markov-chain formulations (Campbell et al., 2022), and density-ratio estimation in SEDD (Lou et al., 2024). Masked-denoising approaches include MDLM, MD4, and LLaDA; MD4 also studies state-dependent masking schedules (Sahoo et al., 2024; Shi et al., 2024; Nie et al., 2025). DDSM introduced diffusion over the probability simplex for de novo promoter generation (Avdeyev et al., 2023), while Dirichlet Flow Matching extended simplex-based generation with guided sampling for promoter and enhancer design (Stark et al., 2024). STGFlow combines Gumbel-Softmax flow matching with straight-through guidance for controllable biological sequence generation (Tang et al., 2025). SLM simplified this paradigm by progressively pruning candidate nucleotides on simplex centroids and achieved strong results on both promoter and enhancer generation (Song et al., 2025). More recently, D3LM reformulated a bidirectional genomic language model as a masked discrete diffusion model, combining DNA representation learning with iterative generation (Yang et al., 2026). Together, these works demonstrate the promise of discrete, non-autoregressive modeling for regulatory DNA, but mainly focus on generative modeling or conditional guidance rather than downstream RL adaptation.

On the autoregressive side, regLM learns label-conditioned regulatory sequence distributions (Lal et al., 2024). Reward optimization provides a complementary route from generation to functional design. TACO and Ctrl-DNA apply RL to autoregressive genomic language models, with Ctrl-DNA additionally enforcing off-target and TFBS-based constraints (Yang et al., 2025; Chen et al., 2025). DRAKES instead optimizes pretrained discrete diffusion models with downstream rewards while regularizing toward the pretrained distribution, and demonstrates enhancer-activity optimization (Wang et al., 2025). GLID^{2}E uses policy-based RL with likelihood clipping and reward shaping for lightweight discrete biological sequence fine-tuning (Cao et al., 2025). These DNA-design approaches primarily adapt sequence content under predefined generation procedures. Our framework additionally learns which position to generate next, with functional reward and real-sequence supervision guiding the joint adaptation of order and content.

B.3 Generation order in masked diffusion models

Generation order is increasingly recognized as an important component of masked diffusion. Earlier iterative masked generation also used model confidence to select which tokens to retain and which to resample, as exemplified by MaskGIT for image tokens (Chang et al., 2022). SCUD separates transition timing from destination prediction by conditioning on the known jump schedule (Amin et al., 2025), motivating distinct treatment of timing and content. P2 explicitly separates planning from denoising during sampling and allows revision of previously unmasked tokens (Peng et al., 2025). Kim et al. (2025) show that different unmasking orders can substantially change inference difficulty.

Recent work learns generation order through either reward-based or likelihood-based objectives. UPO replaces heuristic position selection with an RL-trained scheduler while freezing the underlying diffusion model (Hong et al., 2026a). In its masked-diffusion formulation, DCoLT uses a Plackett–Luce unmasking policy and jointly trains it with the token generator using outcome-based RL for reasoning tasks (Huang et al., 2025). LoMDM learns a context-dependent order jointly with the denoising model during likelihood-based pretraining, using learned corruption and forward–reverse rate matching (Hong et al., 2026b). During RL post-training, JustGRPO further shows that unrestricted arbitrary-order decoding is not always beneficial and instead uses a fixed left-to-right rollout order for reasoning tasks (Ni et al., 2026).

Building on these advances, DP-DNA studies order-aware RL from two complementary perspectives: optimizing functional reward and encouraging compatibility with real DNA. Motivated by the order-sensitive distributional degeneration observed during DNA reward optimization, we combine joint order–content GSPO with a real-sequence training stream adapted from LoMDM (Hong et al., 2026b). The free-generation stream updates both policies using task reward under content-reference regularization. The real-sequence stream uses learned corruption to train content reconstruction and forward–reverse rate matching, giving the generator a reconstruction signal and the planner a real-data-derived order signal. This combination trains the planner to select positions that support functional optimization while encouraging generation dynamics compatible with real DNA.

C Two-Stream Optimization and Algorithms

C.1 Policy parameterization and gradient decomposition

The content model has parameters θ, the reverse planner ψ, and the training-only corruption head ϕ. The two rate heads have the same architecture with independent parameters. For head output h_i, the positive rates are parameterized as

$$\mathrm{rate}_i = 0.7 + 0.65 \left(\sigma(h_i) - \frac{1}{L} \sum_{j=1}^{L} \sigma(h_j) \right). \tag{22}$$

Their sequence-wide mean is 0.7, with rates in $(0.05, 1.35)$. The reverse head normalizes these rates over currently masked positions, as in Eq. (9). The forward head instead specifies the position-dependent corruption probabilities in Eq. (12).

Joint order–content GSPO. For one stored free trajectory, define its current log-probability $\ell_n^Q = \log a_\psi + \log b_\theta$. Since $\rho_n^G = \exp[(\ell_n^Q - \ell_n^-)/T]$, differentiating the free loss gives

$$w_n = \frac{1}{G} \rho_n^G \, \partial_\rho \Phi(\rho_n^G, A_n^G), \qquad \begin{aligned} \nabla_\theta \mathcal{L}_{\mathrm{free}} &= -\sum_n w_n \sum_t \nabla_\theta \log p_\theta(x_{n,i_t} \mid s_{nt}, i_t), \\ \nabla_\psi \mathcal{L}_{\mathrm{free}} &= -\sum_n w_n \sum_t \nabla_\psi \log g_\psi(i_t \mid s_{nt}), \\ \nabla_\phi \mathcal{L}_{\mathrm{free}} &= 0. \end{aligned} \tag{23}$$

Both policies receive the same trajectory coefficient. On an active unclipped branch, $w_n = \rho_n^G A_n^G / G$; on a flat clipped branch it is zero. The factor T in Eq. (14) preserves the sequence-sum score-gradient scale: at the behavior parameters, the coefficient is A_n^G/G. All behavior log-probabilities and advantages are held fixed during the two replay updates.

These derivatives use the head-input stop-gradient convention: rate heads receive detached backbone features. The order loss updates the reverse head, while the token loss updates the backbone; current features are recomputed at every update. The reference contribution in Eq. (13) compares token probabilities on sampled contexts. The joint GSPO ratio nevertheless includes both token and position probabilities.

Real-data gradients. Let $F(m; x, u)$ be the integrand in Eq. (15), and let $\ell_i = -\log p_\theta(x_i \mid z, i)$. With detached head inputs, the explicit content and reverse-head contributions are

$$\nabla_\theta \mathcal{L}_{\text{real}} = \mathbb{E}\left[\sum_i \frac{m_i e_i}{u} \nabla_\theta \ell_i\right], \tag{24}$$

$$\nabla_\psi \mathcal{L}_{\text{real}} = \mathbb{E}\left[\sum_i \frac{m_i}{u}\left(1 - \frac{e_i}{r_i}\right) \nabla_\psi r_i\right]. \tag{25}$$

The rate-matching term is nonnegative and minimized at $r_i = e_i$. The corruption head also affects the sampling law:

$$\nabla_\phi \mathcal{L}_{\text{real}} = \mathbb{E}\left[\sum_i \frac{m_i}{u}\{\ell_i + \log(e_i/r_i)\}\nabla_\phi e_i + F(m; x, u)\nabla_\phi \log q_\phi(m \mid x, u)\right]. \tag{26}$$

Consequently, generator updates combine token-level reward credit with natural-sequence reconstruction, planner updates combine trajectory-level reward credit with rate matching, and corruption-head updates adapt the real-data masking distribution. This gives the decomposition

$$\begin{aligned} \nabla_\theta \mathcal{L} &= \nabla_\theta \mathcal{L}_{\text{free}} + \lambda_{\text{real}} \nabla_\theta \mathcal{L}_{\text{real}}, \\ \nabla_\psi \mathcal{L} &= \nabla_\psi \mathcal{L}_{\text{free}} + \lambda_{\text{real}} \nabla_\psi \mathcal{L}_{\text{real}}, \\ \nabla_\phi \mathcal{L} &= \lambda_{\text{real}} \nabla_\phi \mathcal{L}_{\text{real}}. \end{aligned} \tag{27}$$

C.2 Learned masking and the two-sample estimator

We use the rate-based LoMDM formulation of Hong et al. (2026b), with content reconstruction and forward–reverse rate matching. The corruption head reads complete training sequences, whereas the reverse head reads only a masked context. The forward head is used during training; generation uses g_ψ and p_θ.

For efficient integration over noise levels, set $\zeta = -\log u$. Then du/u becomes $d\zeta$ (with reversed limits), and the masking probability is $\exp(-e_i\zeta)$. Draw $\zeta \geq 0$ from

$$\nu(\zeta) = \tfrac{1}{2}(0.7)e^{-0.7\zeta} + \tfrac{1}{2}(0.025)e^{-0.025\zeta}. \tag{28}$$

The corresponding importance-weighted integrand is

$$\widetilde{F}(m; x, \zeta) = \frac{1}{\nu(\zeta)} \sum_i m_i \left[e_i \ell_i + e_i \log(e_i/r_i) - e_i + r_i\right]. \tag{29}$$

The batch allocates half its time samples to each proposal component. At each (x, ζ), sample two independent masks, with integrands F_1, F_2 and mask log-probabilities b_1, b_2. The real-stream gradient is estimated by differentiating

$$\widehat{\mathcal{L}}_{\text{real}} = \text{mean}_{x,\zeta}\left[\frac{F_1 + F_2}{2} + \frac{b_1 - b_2}{2}\,\text{sg}(F_1 - F_2)\right]. \tag{30}$$

Here sg stops gradients. The second term estimates the masking-distribution contribution in Eq. (26) using the other independent mask as a baseline. This real-stream score estimator is separate from the joint order–content GSPO used on freely generated DNA.

Each rollout group's two updates reuse the real sequences and base random draws, but evaluate the current rates, masked contexts, and losses anew. Each update uses $\lambda_{\text{real}} = 0.2$ on the sequence-sum objective. The reverse and forward heads are optimized jointly with the content model.

C.3 Training and inference

Algorithm 1 Two-stream joint content–order GSPO training

Require: Generator p_θ, reverse head ψ, corruption head ϕ, frozen p_{ref}, real data p_D, reward R_c
Require: Group size G, real batch size B_D, clipping ϵ, weights $\beta, \lambda_{\mathrm{real}}$
for each rollout group **do**
  Sample G complete paths using g_ψ and p_θ
  Store states, positions, tokens, and behavior order/content log-probabilities
  Score each endpoint with R_c and compute reference penalties K^-
  Compute and freeze leave-one-out advantages of $R_c - \beta K^-$
  Sample B_D real sequences from the target-cell TRAIN pool
  for each of two optimization epochs **do**
    Recompute current token and order probabilities on the stored free paths
    Compute ρ^G and $\mathcal{L}_{\mathrm{free}}$ using Eq. (14)
    Apply the corruption head to real sequences; sample two masked views per sequence
    Evaluate real-token reconstruction and forward–reverse rate matching
    Form $\widehat{\mathcal{L}}_{\mathrm{real}}$ with Eq. (30)
    Update θ, ψ, ϕ using $\mathcal{L}_{\mathrm{free}} + \lambda_{\mathrm{real}} \widehat{\mathcal{L}}_{\mathrm{real}}$
  end for
end for

Algorithm 2 Inference with the learned dual policy

Require: Content generator p_θ, reverse head ψ, condition c, legal token supports
Initialize s with every generation position masked
while $\mathcal{M}(s) \neq \emptyset$ **do**
  Compute content probabilities and reverse rates from the current context
  Sample $i \sim g_\psi(\cdot \mid s)$ over unresolved positions
  Sample $v \sim p_\theta(\cdot \mid s, i)$ on the legal token support
  $s \leftarrow \mathrm{Write}(s, i, v)$
end while
return decoded DNA sequence

D Experimental Details and Evaluation Protocol

D.1 Data, reward predictor, and evaluation oracle

The Gosai collection contains approximately 700,000 200-base regulatory sequences with MPRA measurements in HepG2, K562, and SK-N-SH (Gosai et al., 2024). Each cell defines a functional design objective on the same sequence collection. The data specification uses 652,505 TRAIN sequences for reward-free adaptation and a fixed 4,096-sequence validation set.

Reward predictor versus evaluation oracle. We follow the distinction between optimization and evaluation in DRAKES (Wang et al., 2025, Section 6.2 and Appendix F.2). DRAKES trains two activity predictors with the same Enformer architecture and pretrained initialization (Avsec et al., 2021), but fits them separately on disjoint chromosome-based subsets of the Gosai MPRA data. Each subset contains sequences from approximately half of the 23 human chromosomes. The FT predictor supplies optimization rewards, whereas the Eval predictor is reserved for evaluating generated sequences. Thus, their supervised training data and fitted parameters differ, while their architecture and activity-prediction task are shared. In our experiments, the frozen FT predictor supplies R_c, and the separately trained Eval oracle supplies E_c for Pred-Activity; the Eval oracle is not queried during RL. ATAC-Acc uses an additional chromatin-accessibility classifier and is separate from both activity predictors.

For the real-sequence stream, we select TRAIN sequences whose measured target-cell activity exceeds its TRAIN median and whose two non-target activities fall below their respective TRAIN medians. This gives 31,902 HepG2, 51,061 K562, and 43,373 SK-N-SH sequences. The same TRAIN-derived thresholds define corresponding validation pools of 208, 327, and 268 sequences. Real training batches are sampled with replacement from the target-cell pool.

D.2 ARCHITECTURE AND TRAINING CONFIGURATION

A 200-base sequence is represented by 33 nonoverlapping six-mer tokens and two terminal single-base tokens. The content model is a 12-layer bidirectional Transformer with hidden dimension 512 and 16 attention heads. Position-specific vocabulary support admits 4,096 six-mers at the first 33 positions and A/C/G/T at the final two.

Both rate heads take detached backbone representations, project them to 128 dimensions, add trainable position embeddings, and apply one Transformer block with four heads and feed-forward dimension 256. A small output network and Eq. (22) produce positive rates. The forward head sees complete real DNA; the reverse head sees the current masked context. The two heads have independent parameters, and each target cell has its own content model and heads.

The configuration initializes the actor and rate heads from 20,000 reward-free uniform-corruption adaptation updates (batch size 64), with actor 10^{-4} and a 500-update warmup. The frozen content reference is the separately selected 10,000-step domain-adapted D3LM checkpoint. During downstream optimization, both rate heads are learned and the content reference stays fixed. Appendix E details these two preparation stages.

Table 4: *Two-stream GSPO configuration specified for* DP-DNA. *Losses use sequence-sum normalization.*

Setting	Value
Generator / generation length	D3LM; 35 token commitments / 200 bases
Forward / reverse rate heads	Independent one-layer Transformers; width 128
Rate mean / centered scale	0.7 / 0.65
Free-generation group size	32
Real-sequence batch / masks per sequence	64 / 2
GSPO updates per rollout group / clipping	2 / 0.005
Advantage	Leave-one-out utility; fixed during replay
Content-reference penalty β	0.05
Real-sequence weight λ_{real}	0.2 per update, fixed
Optimizer	Adam; $(0.9, 0.999)$; $\epsilon = 10^{-8}$
Generator / rate-head learning rates	3×10^{-5} / 10^{-4}
Gradient norm limits	1 for generator; 1 for the combined heads
Temperature / nucleus parameter	1 / 1

The real loss is evaluated once per GSPO update. The two updates share one batch of free trajectories and fixed leave-one-out advantages, while current probabilities and real-data losses are recomputed. Position probabilities are included in the joint GSPO ratio. The reference penalty acts on generated content probabilities.

D.3 D3LM+GSPO BASELINE

To isolate the effect of applying reward optimization directly to the D3LM backbone, we additionally train a D3LM+GSPO baseline. This baseline starts from the uniform-corruption pre-adaptation D3LM checkpoint and applies GSPO directly to the content model.

Model and generation process. The baseline uses the same D3LM sequence representation as our method: each 200-bp sequence is represented by 33 non-overlapping 6-mer tokens followed by two single-base tokens, giving $T = 35$ generation events. The backbone is the released 12-layer bidirectional D3LM Transformer with hidden dimension 512 and 16 attention heads. All model parameters are fine-tuned.

At generation time, the next unresolved position is sampled uniformly:

$$g_{\mathrm{uni}}(i \mid s) = \frac{1}{|\mathcal{M}(s)|}, \qquad i \in \mathcal{M}(s), \tag{31}$$

and D3LM samples the token at that position from its masked conditional distribution. The position policy is fixed and contains no trainable parameters.

Table 5: **Training configuration for the D3LM+GSPO baseline.**

Setting	Value
Initialization	Uniform-corruption pre-adaptation D3LM
Generation order	Fixed uniform
Training iterations	1,000
Rollout group size	32
Minibatch size	16
Optimization epochs / group	2
Optimizer	Adam
Learning rate	3×10^{-5}
GSPO clipping ϵ	0.005
Content KL coefficient β	0.05
Order KL coefficient	0
Gradient-norm limit	1.0
Sampling temperature	1.0
Top-p	1.0

GSPO optimization. For a sampled trajectory $\tau = (i_{1:T}, v_{1:T})$, GSPO uses a single length-normalized likelihood ratio for the complete generation path. Because both the current and behavior policies use the same fixed uniform position scheduler, the position terms cancel exactly, and the ratio reduces to the content-policy contribution:

$$\rho(\theta) = \exp\left[\frac{1}{T}\sum_{t=1}^{T}\log\frac{p_\theta(v_t \mid s_t, i_t)}{p_{\theta^-}(v_t \mid s_t, i_t)}\right]. \tag{32}$$

The resulting sequence-level ratio is optimized with the standard clipped GSPO surrogate. Terminal rewards are standardized within each rollout group of 32 sequences; no critic or value network is used.

We additionally regularize the content policy toward a frozen copy of the domain-adapted D3LM model using a conditional KL penalty with coefficient $\beta = 0.05$. Since the generation-order policy is fixed, no order-policy KL term is used.

Training configuration. A separate model is trained for each of HepG2, K562, and SK-N-SH using three independent random seeds. The main benchmark uses the 1,000-iteration setting, corresponding to 32,000 reward queries per training run. We use Adam with learning rate 3×10^{-5}, group size 32, minibatch size 16, two optimization epochs per rollout group, GSPO clipping threshold $\epsilon = 0.005$, and global gradient-norm clipping at 1. Sampling uses temperature 1 and no nucleus truncation ($p = 1$).

The baseline uses the same frozen fine-tuning activity predictor described in Appendix D.1 as its terminal reward. The separately trained evaluation oracle is never queried during optimization and is used only for post-training evaluation. Thus, D3LM+GSPO provides a reward-optimized D3LM reference without the learned-order and real-sequence mechanisms introduced by DP-DNA.

D.4 Evaluation metrics

Metrics are computed on decoded sequences $y = (y_1, \ldots, y_N)$, with $N = 200$ and bases in $\{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{T}\}$. Let $\mathcal{X}$ denote one generation replicate of 640 sequences.

Pred-Activity and ATAC-Acc. Pred-Activity is $\mathrm{median}_{y\in\mathcal{X}} E_c(y)$, where E_c is the target-cell evaluation predictor. ATAC-Acc is the percentage of sequences with cell-matched accessibility probability above 0.5:

$$\mathrm{ATAC\text{-}Acc}(\mathcal{X}) = \frac{100}{|\mathcal{X}|}\sum_{y\in\mathcal{X}}\mathbb{I}[P_{\mathrm{ATAC},c}(\mathrm{open} \mid y) > 0.5]\,. \tag{33}$$

These two statistics follow the definitions accompanying the updated benchmark table and Wang et al. (2025).

3-mer Corr. We count overlapping three-base substrings in each sequence and aggregate over the generated set. Let $n_3(\mathcal{X}) \in \mathbb{R}^{64}$ be the resulting count vector. The score is

$$3\text{mer Corr}(\mathcal{X}) = \text{Corr}_{\text{Pearson}}(n_3(\mathcal{X}), n_3(\mathcal{R}_c)), \tag{34}$$

where $\mathcal{R}_c$ contains the top 0.1% most active TRAIN sequences for cell c. This distribution-level correlation is distinct from the within-sequence 3-mer entropy used in the collapse analysis.

JASPAR Corr. We use the DRAKES motif-scanning and population-count convention. For generated and real-reference motif-count vectors $m(\mathcal{X})$ and $m(\mathcal{R})$, respectively,

$$\text{JASPAR Corr}(\mathcal{X}) = \text{Corr}_{\text{Spearman}}(m(\mathcal{X}), m(\mathcal{R})). \tag{35}$$

The native metric retains its fixed top-0.1% reference. Duplicate motif names are pooled before pairwise-complete correlation. Here motif agreement is an evaluation measure, not a training loss.

Authenticity and Rarity. Let $h_4(y)$ be the frozen 4-mer feature representation and z_j the real-reference features. Set $\nu(j) = \min_{k \neq j} \|z_j - z_k\|$ and $j^*(y) = \arg\min_j \|h_4(y) - z_j\|$. The non-memorization score follows the authenticity criterion of Alaa et al. (2022):

$$\text{Authenticity}(\mathcal{X}) = \frac{1}{|\mathcal{X}|} \sum_{y \in \mathcal{X}} \mathbb{I}\big[\|h_4(y) - z_{j^*(y)}\| > \nu(j^*(y))\big]. \tag{36}$$

Following Han et al. (2023), for Rarity let r_j be the distance to reference point z_j's fifth nearest reference neighbor. Supported generated features lie in $\mathcal{S} = \bigcup_j B(z_j, r_j)$, with raw rarity $\text{rar}(z) = \min_{j:\|z - z_j\| \leq r_j} r_j$. We report

$$\text{Rarity}_{\text{IQR}} = \frac{\text{median}(\text{rar}_{\text{gen}}) - \text{median}(\text{rar}_{\text{Train}})}{Q_{.75}(\text{rar}_{\text{Train}}) - Q_{.25}(\text{rar}_{\text{Train}})}. \tag{37}$$

Generated rarity is evaluated on the supported subset; reference features and neighborhoods are fixed across methods.

App-Log-Lik. Following Wang et al. (2025), we compute an evidence lower bound under the frozen pretrained masked-diffusion prior using 128 denoising steps, then report the median over $\mathcal{X}$. This measures typicality under that prior; the values are sequence-level log-likelihood bounds rather than the per-base autoregressive scores below.

Evo 2 log-likelihood. We additionally score sequences with the pretrained Evo 2 7B genomic language model (Brixi et al., 2026). The supplied scorer evaluates the forward sequence and its reverse complement y^{RC}, averaging their per-position log-probabilities. With no prepended start token, the first base is context for the remaining $N - 1$ predictions:

$$\ell_{\rightarrow}(y) = \frac{1}{N-1} \sum_{j=2}^{N} \log p_{\text{Evo2}}(y_j \mid y_{1:j-1}), \qquad \ell_{\text{Evo2}}(y) = \frac{\ell_{\rightarrow}(y) + \ell_{\rightarrow}(y^{\text{RC}})}{2}. \tag{38}$$

The units are nats per scored base (199 scored bases for a 200-base fragment). Fragments are scored without genomic flanks. The scorer exposes both the mean and median of ℓ_{Evo2} across sequences; Table 1 uses the median specified in its reporting protocol. Larger values indicate greater likelihood under Evo 2, complementary to the task-specific activity and accessibility metrics.

Replicate aggregation. The main table reports means and sample standard deviations across three training replicates of 640 sequences. Per-replicate metrics are computed before this aggregation. These repetitions quantify the stability of model performance. The ablation experiments follow the same protocol, with each model trained independently three times and the average performance reported as the final result.

Table 6: **Ablation study on the K562 enhancer-design task.** *Differences relative to the full* DP-DNA *model are shown in parentheses.*

Ablation	K562			
	Pred-Activity ↑	**ATAC-Acc (%) ↑**	**3-mer Corr ↑**	**Evo 2 LL ↑**
Training				
w/o free-generation stream	0.081 (-8.321)	1.30 (-97.87)	-0.224 (-1.096)	-1.3342 (-0.0704)
w/o real-sequence stream	8.761 (+0.359)	99.84 (+0.67)	0.860 (-0.012)	-1.3913 (-0.1275)
Model Arch				
w/o planner	8.160 (-0.242)	97.02 (-2.15)	0.870 (-0.002)	-1.3781 (-0.1143)
w/o generator-feature input	8.432 (+0.030)	98.23 (-0.94)	0.861 (-0.011)	-1.2672 (-0.0034)
w/o corruption model ϕ	8.208 (-0.194)	98.20 (-0.97)	0.892 (+0.020)	-1.2687 (-0.0049)
DP-DNA	**8.402**	**99.17**	**0.872**	**-1.2638**

D.5 MDLM BASELINE ACROSS CELL-SPECIFIC TASKS

MDLM is used as an unconditional generative baseline. Unlike the task-specific methods, it is not conditioned on the target cell type and is not further fine-tuned for HepG2, K562, or SK-N-SH. We therefore use the same pretrained MDLM checkpoint and decoding procedure for all three enhancer-design tasks. Consequently, for matched sampling seeds, MDLM produces the same generated sequences across cell types; more generally, all three tasks sample from the same unconditional sequence distribution.

The generated sequences are subsequently evaluated with the corresponding cell-specific evaluators and references. Cell-dependent metrics, such as Pred-Activity and ATAC-Acc, can therefore differ across tasks even though the underlying MDLM samples are unchanged. In contrast, evaluation statistics that depend only on the generated sequences, or otherwise use a cell-independent reference, are identical when computed on the same generated set. Repeated MDLM values across cell types thus reflect the task-agnostic nature of this baseline rather than separate cell-specific training runs.

D.6 ADDITIONAL ABLATION RESULTS ON K562

To further examine whether the effects of the main design components extend across cell types, we evaluate the same five ablations on the K562 enhancer-design task. Table 6 reports their effects on predicted activity, accessibility, sequence composition, and genomic-model likelihood.

The K562 results further support the importance of adaptive generation order while also revealing target-dependent differences in the relationship between functional optimization and sequence-distribution agreement.

Training streams. The two training streams show the clearest separation of roles. Removing the free-generation stream nearly eliminates the gains from task-specific adaptation, reducing Pred-Activity from 8.402 to 0.081 and ATAC-Acc from 99.17% to 1.30%, together with a large degradation in 3-mer correlation (-1.096). This stream is therefore essential for learning the downstream design objective. Removing the real-sequence stream produces a different pattern: Pred-Activity increases to 8.761 and ATAC-Acc to 99.84%, whereas Evo 2 likelihood decreases by 0.1275 nats/bp and 3-mer correlation by 0.012. On K562, the real-sequence stream therefore provides a realism-oriented regularizing signal to the learned generation process, with improved sequence-distribution agreement accompanied by a modest reduction in the functional metrics. The magnitude of the Evo 2 decrease closely matches those observed on HepG2 (-0.141) and SK-N-SH (-0.121), indicating a consistent contribution of the real-sequence stream to genomic-model likelihood across all three targets. On K562, this decrease is also larger than that caused by removing the free-generation

stream (-0.0704), suggesting that the real-sequence stream makes the larger contribution to Evo 2 likelihood in this setting.

Model components. Removing the planner reduces Pred-Activity by 0.242 and ATAC-Acc by 2.15 percentage points, while also decreasing 3-mer correlation slightly (-0.002) and Evo 2 likelihood (-0.1143). Learned position selection therefore contributes to both functional performance and sequence-distribution agreement on K562. The Pred-Activity effect is directionally consistent with HepG2 (-0.131) and SK-N-SH (-0.146).

The planner inputs show a more nuanced, target-dependent pattern. Removing generator-derived features marginally increases Pred-Activity ($+0.030$), but decreases ATAC-Acc by 0.94 percentage points, 3-mer correlation by 0.011, and Evo 2 likelihood by 0.0034. Thus, unlike on HepG2 and SK-N-SH, where removing these features reduces Pred-Activity by 0.334 and 0.210, respectively, their inclusion on K562 trades a small amount of Pred-Activity for improvements in the other three metrics. Similarly, removing the learned corruption model ϕ reduces Pred-Activity, ATAC-Acc, and Evo 2 likelihood, while increasing 3-mer correlation. The learned corruption model therefore does not improve every distributional statistic independently, but affects functional and sequence-distribution metrics in different directions.

Summary. Taken together with the HepG2 and SK-N-SH ablations, these results show that the free-generation stream is consistently essential for learning the downstream design objective, whereas the real-sequence stream consistently improves Evo 2 likelihood across all three targets. Its effect on Pred-Activity is more target dependent: removing the stream reduces activity on HepG2 and SK-N-SH but increases it on K562. The architectural components likewise show some target-specific behavior, most clearly for the generator-feature input, whose removal lowers activity on HepG2 and SK-N-SH but slightly increases it on K562. Thus, while the main components make consistent contributions to functional adaptation and sequence-distribution regularization, the precise trade-off between these objectives depends on the target cell type.

D.7 Learned-order diagnostic

For each target cell, we use 640 learned-order and 640 uniform-order generations from checkpoints of each cell. Let $r_{n,j}^{\text{learned}}$ and $r_{n,j}^{\text{uniform}}$ be the zero-based reveal ranks (0–34) of position j. The plotted shift is

$$\Delta_j = \frac{1}{640} \sum_{n=1}^{640} \left(r_{n,j}^{\text{learned}} - r_{n,j}^{\text{uniform}} \right). \tag{39}$$

Negative shifts indicate earlier commitment under the planner. The gray band is the archived pointwise 95% sign-flip null envelope; dots identify shifts outside it.

To characterize position-wise sequence complexity, we estimate the six-mer (token) frequency $f_j(w)$ across the uniform-order sequences and calculate

$$H_j = -\sum_{w} f_j(w) \log f_j(w), \qquad j = 0, \ldots, 32. \tag{40}$$

Natural logarithms give nats. This is empirical token diversity across sequences, distinct from the model's conditional predictive entropy and from within-sequence 3-mer entropy. Positions 33–34 contain single-base suffix tokens, so no six-mer entropy is assigned to them. Both decoders use the same model parameters; only the position-selection rule changes.

Across the 33 six-mer positions, the Pearson correlations between Δ_j and H_j are 0.7772, 0.8900, and 0.8235 for HepG2, K562, and SK-N-SH. The positive association indicates that lower-entropy positions tend to be advanced relative to higher-entropy positions, with cell-dependent effect sizes.

E SEQUENCE DOMAIN ADAPTATION AND PRE-ADAPTATION

We use two reward-free preparation runs, initialized independently from the released D3LM-scratch weights (Yang et al., 2026). Domain adaptation constructs the frozen content reference, while uniform-corruption pre-adaptation initializes the trainable model. Neither stage uses activity labels or reward/evaluation predictors.

E.1 DOMAIN ADAPTATION AND REFERENCE SELECTION

The content reference is adapted to Gosai enhancer sequences (Gosai et al., 2024), using 652,505 training sequences and 4,096 validation sequences. The validation set excludes the three cell-specific benchmark reference sets. Each $N = 200$-base sequence contains $L = 35$ tokens: 33 six-mers and two single-base tokens. We sample K uniformly from $\{1, \dots, L\}$ and a uniformly random K-position mask M, minimizing

$$\mathcal{L}_{\mathrm{DA}}(\theta) = \mathbb{E}_{x,K,M}\left[\frac{L}{NK}\sum_{i\in M} -\log p_\theta(x_i \mid \mathrm{Mask}(x, M), i)\right]. \tag{41}$$

Only the content model is updated. Training uses AdamW with batch size 32, learning rate 10^{-4}, momentum parameters $(0.9, 0.95)$, weight decay 0.01, and gradient-norm clipping at 1. Reference selection uses fixed-mask validation reconstruction loss alone; the selected 10,000-update checkpoint is frozen as p_{ref} throughout RL.

E.2 UNIFORM-CORRUPTION PRE-ADAPTATION

A separate run uses the real-sequence reconstruction formulation in Section 4.2, with corruption rates fixed at $e_i = 0.7$, rate matching disabled, and reconstruction loss normalized by $L = 35$. We use 20,000 updates, batch size 64, generator learning rate 10^{-4}, and a 500-update linear warmup.

The checkpoint includes the generator and two 128-dimensional rate heads. Zero-initialized head outputs yield constant rates of 0.7 and a uniform initial position policy; task-dependent corruption and reveal priorities are learned during RL. The trainable modules (θ, ϕ, ψ) are loaded from this checkpoint, while p_{ref} comes from the independently selected domain-adaptation checkpoint. The RL optimizer is initialized afresh.