HIGH REWARD, WRONG SEQUENCES: REWARD-HACKING RESISTANT DISCRETE DIFFUSION FOR 5′UTR DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Reward-guided fine-tuning of discrete diffusion models offers a powerful route for *de novo* 5′UTR design, where the goal is to generate RNA sequences with high mean ribosome load (MRL) or translation efficiency (TE) while preserving biological plausibility. However, we find that this strategy is highly vulnerable to reward hacking: the predicted reward rises rapidly, while sequence naturalness, quantified by the statistical correlation between generated sequences and in-distribution natural sequences, collapses. We trace this failure to a simple but consequential mechanism: the reward model learns shortcuts that appear predictive on training sequences, then assigns inflated scores to unnatural sequences, pulling the generative policy away from biologically meaningful design space. We propose **PILOT-UTR** (**P**olicy f**I**ne-tuning with **L**earned-reward calibration and **O**n-policy **T**eacher distillation for **UTR** design). PILOT-UTR first calibrates the reward model via using hacking policy to discover out-of-distribution sequences as label-free supervisions, suppressing inflated predictions on them through a cap loss. With this calibrated reward, PILOT-UTR then fine-tunes the generator through an ingenious teacher-student mechanism in which a slowly updated teacher acts as a moving biological anchor, guiding the student to improve reward while staying close to natural 5′UTR patterns. Our analysis explains how this teacher-student mechanism controls local prediction drift and gradually incorporates useful policy updates. Experiments on MRL- and TE-guided 5′UTR optimization demonstrate that PILOT-UTR mitigates reward hacking and achieves high predicted reward while preserving sequence naturalness and predicted structural stability. Code is available in `https://anonymous.4open.science/r/PILOT-UTR-467F`.

1 INTRODUCTION

The *de novo* design of 5'UTR sequences is a central problem in RNA engineering. A 5'UTR is a short RNA sequence upstream of the coding region of an mRNA that governs how efficiently ribosomes initiate translation, where a small gain can improve the protein output of mRNA therapeutics and vaccines (Hinnebusch et al., 2016; Ma et al., 2024). Given a target functional property such as mean ribosome load (MRL), a proxy for average ribosome occupancy, or translation efficiency (TE), the goal is to generate new 5'UTRs that achieve high property values while remaining biologically plausible, including stable secondary structure, low minimum free energy (MFE), and natural-like nucleotide composition. Traditionally, 5'UTRs are optimized through rational design based on biological prior knowledge, such as Kozak-like contexts and secondary-structure tuning, and through wet-lab trial-and-error (Kozak, 1987; Leppek et al., 2018). These approaches are slow, costly, and explore only a tiny fraction of the combinatorial sequence space.

Deep generative approaches have been developed for a wide range of biomolecule generation tasks, among which diffusion models have emerged as one of the most effective frameworks (Sarkar et al., 2024; Hou et al., 2025). In the 5'UTR design, diffusion can represent each sequence as a discrete string over nucleotide vocabulary $\{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{U}\}$ and learn a denoising process that maps a fully masked template back to a clean sequence, allowing masked discrete diffusion models to sample tractably from the natural 5'UTR manifold and support conditional generation (Shi et al., 2024). Considering the reward-guided *de novo* design, existing methods fall into two categories. *i) Clas-*

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.

sifier guidance steers the sampling trajectory at inference time using gradients from a separately trained property predictor (Song et al., 2020; Dhariwal & Nichol, 2021). In discrete token spaces, however, noise-conditional classifiers can be unreliable, and repeated guided sampling substantially increases inference cost. *ii) Reward-model fine-tuning* instead directly optimizes the pretrained diffusion policy through reinforcement learning (RL) against a learned reward function, a supervised predictor trained on labeled sequence-property data and used as a proxy for the true property. DRAKES (Wang et al., 2024) performs direct reward backpropagation with the Gumbel-softmax trick, while Zekri & Boullé (2025) use proximal policy optimization (PPO) for non-differentiable rewards. The detailed related work section is presented in Appendix B.

Despite its promise, RL fine-tuning encounters a severe **reward hacking problem** in 5'UTR design, with a more pronounced breakdown between predicted reward and sequence plausibility than reported in conventional generative tasks. As shown in Section 3, standard fine-tuning rapidly increases predicted reward while compositional correlation with natural UTRs collapses. Generated sequences also lose the base pairing found in natural high-MRL UTRs, becoming nearly unstructured despite higher predicted scores (Figure 2)). *These observations indicate exploitation of inflated out-of-distribution (OOD) predictions: apparent reward gains accompany deteriorating sequence and structural plausibility, i.e., reward hacking.* This failure is especially acute in 5'UTR design for three reasons. **i)** Noisy and limited RNA property labels hinder the reliable reward predictors (Sample et al., 2019; Schlusser et al., 2024). **ii)** Labeled data sparsely cover the 4^L sequence space, where L denotes UTR length, allowing the fine-tuning policy to amplify weak associations between incidental sequence patterns and high scores into exploitable shortcuts. This leads to the OOD sampled sequences without evidence of improved function. **iii)** The biological function requires costly experiments unavailable during fine-tuning, whereas conventional preference models (e.g., human-preference models for generated text) can be re-evaluated more cheaply. Predictor errors therefore remain uncorrected during optimization in the 5'UTR design. Although briefly noted by Huang et al. (2024), the reward hacking in diffusion fine-tuning for biological sequence design has, to our knowledge, not been systematically analyzed and addressed. Moreover, widely adopted ensemble-based mitigation strategies struggle to correct OOD reward overestimation (Coste et al., 2024; Uehara et al., 2024). As our analysis in Section 3 shows, multiple independently trained predictors can learn the shared shortcuts and assign similarly inflated scores to the unnatural sequences, leaving disagreement-based uncertainty penalties unable to detect and prevent exploitation.

To address the reward hacking in 5'UTR design, we propose **PILOT-UTR** (**P**olicy f**I**ne-tuning with **L**earned-reward calibration and **O**n-policy **T**eacher distillation for **UTR** design), which integrates reward calibration with adaptive policy stabilization. Our contributions are threefold:

- We systematically define and analyze the reward hacking in discrete-diffusion fine-tuning for 5'UTR design. We show that standard policy regularization and ensemble-based strategies fail to prevent this behavior, exposing a critical gap in reliable reward-guided RNA engineering.
- We propose PILOT-UTR, which first calibrates the reward model by capping inflated predictions on policy-generated OOD sequences without additional experimental labels. During fine-tuning, a teacher diffusion model initially trained on natural sequences guides the student to retain plausibility while seeking higher scores from the calibrated reward model. Our theory characterizes the balance of gradual exploration while resisting abrupt shifts toward shortcut-driven patterns.
- Across all benchmarks, PILOT-UTR improves independently evaluated reward over the strongest baseline—by 0.207 on 50-nt MRL, 0.235 on human MRL, and 0.239 on PC3 TE—while outperforming all baselines in SCR@τ. Ablations show the complementary benefits: reward recalibration raises 3-mer correlation from 0.294 to 0.675, while adding teacher-guided policy stabilization further raises it to 0.918 and restores MFE from -0.069 to -6.105.

2 Problem Setup and Preliminaries

2.1 Masked Discrete Diffusion for 5'UTR Sequences

A 5'UTR is an RNA sequence upstream of a coding region that modulates translation. We represent a length-L UTR as $x = (x^{\langle 1\rangle}, \ldots, x^{\langle L\rangle}) \in \mathcal{X} = \mathcal{V}^L$ over the nucleotide alphabet $\mathcal{V} = \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{U}\}$, and seek a generative distribution that preserves natural sequence characteristics while supporting property optimization guided by a reward predictor. We use masked discrete diffusion to model $P^\theta \in \Delta(\mathcal{X})$, a distribution over clean 5'UTR sequences with learnable parameters θ, where $\Delta(\mathcal{X})$

is the probability simplex over $\mathcal{X}$ (Shi et al., 2024). Masked diffusion augments the nucleotide alphabet $\mathcal{V}$ with a MASK token, $\bar{\mathcal{V}} = \mathcal{V} \cup \{\text{MASK}\}$. Forward noising and learned reverse denoising are continuous-time Markov chains (CTMCs) over $\bar{\mathcal{V}}^L$.

Forward Process. Given a clean sequence x_0, let $t \in [0, T]$ be continuous diffusion time, with terminal time T. The forward CTMC independently masks tokens under a monotonically increasing schedule $\sigma(t)$. At one token position, $p_t \in \Delta(\bar{\mathcal{V}}) \subset \mathbb{R}^{|\bar{\mathcal{V}}|}$ is its marginal distribution at time t (p_0 is one-hot on the clean nucleotide) and satisfies $\frac{dp_t}{dt} = Q(t)p_t$; $Q(t) \in \mathbb{R}^{|\bar{\mathcal{V}}|\times|\bar{\mathcal{V}}|}$ is the transition-rate matrix whose nonzero off-diagonal entries are nucleotide-to-MASK transitions. Here $x_t \in \bar{\mathcal{V}}^L$ is the partially masked sequence at time t, and $q(x_t^{\langle i\rangle} \mid x_0^{\langle i\rangle})$ is the distribution of its i-th token conditioned on the clean nucleotide at time 0. Concretely,

$$q(x_t^{\langle i\rangle} = x_0^{\langle i\rangle} \mid x_0^{\langle i\rangle}) = 1 - \sigma(t), \qquad q(x_t^{\langle i\rangle} = \text{MASK} \mid x_0^{\langle i\rangle}) = \sigma(t). \tag{1}$$

With $\sigma(0) = 0$ and $\sigma(T) = 1$, the state x_t evolves from the clean input x_0 at $t = 0$ to the fully masked terminal state x_T at $t = T$; $x_{0:T}$ denotes the resulting forward trajectory.

Reverse Process. The reverse CTMC has generator $\bar{Q}^\theta(t) \in \mathbb{R}^{|\bar{\mathcal{V}}|\times|\bar{\mathcal{V}}|}$, parameterized by a denoiser $\hat{x}_0^\theta(x_t, t)$ that predicts a clean-nucleotide distribution from the partially masked sequence x_t and time t. For a token position, $u, v \in \bar{\mathcal{V}}$ denote its current and destination states, respectively, and $\bar{Q}^\theta_{u,v}(t)$ is the transition rate from u to v at time t, given by (Shi et al., 2024)

$$\bar{Q}^\theta_{u,v}(t) = \begin{cases} \gamma(t) \cdot \hat{x}_0^\theta(v \mid x_t, t) & \text{if } u = \text{MASK},\ v \neq \text{MASK}, \\ -\gamma(t) & \text{if } u = v = \text{MASK}, \\ 0 & \text{otherwise}, \end{cases} \tag{2}$$

where $\gamma(t)$ is determined by $\sigma(t)$. Thus, a masked position is unmasked into nucleotide v at rate $\gamma(t)\hat{x}_0^\theta(v \mid x_t, t)$, while unmasked positions remain fixed. Factorizing these transitions across positions yields the reverse trajectory $x_{T:0}$ from the fully masked x_T to a clean sample $x_0 \sim P^\theta$.

2.2 Reward-guided Fine-tuning of Discrete Diffusion

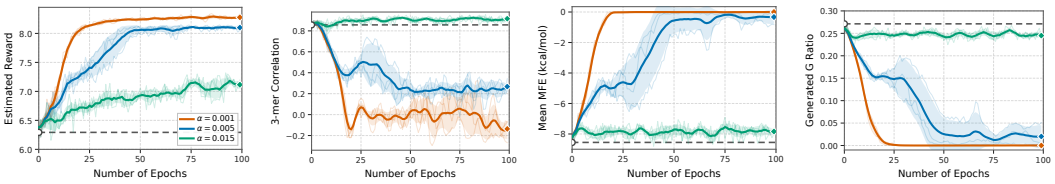
We obtain the pretrained model $P^{\theta_{\text{pre}}}(\cdot)$ by minimizing the cross-entropy of the reverse denoising process on the unlabeled 5'UTR training corpus, so that $P^{\theta_{\text{pre}}}(\cdot)$ captures the natural distribution of 5'UTR sequences. This pretrained model serves as the starting policy for reward-guided fine-tuning. The goal is to obtain a new policy $P^\theta(\cdot)$ whose samples score highly under a task-specific reward while remaining close to the natural distribution. Let $r_\phi : \mathcal{X} \to \mathbb{R}$ denote a separately trained predictor with parameters ϕ, learned from a labeled sequence-property dataset such as MRL or translation efficiency. During fine-tuning, r_ϕ serves as the optimization target:

$$\max_\theta \mathbb{E}_{x_0 \sim P^\theta}[r_\phi(x_0)] - \alpha \text{KL}(P^\theta \| P^{\theta_{\text{pre}}}), \tag{3}$$

The first term encourages P^θ to generate sequences with high predicted reward, while the second term $\text{KL}(P^\theta \| P^{\theta_{\text{pre}}})$ penalizes deviation from the pretrained model, where $\alpha > 0$ is the KL-regularization hyperparameter. Standard policy-gradient methods such as PPO are difficult to apply in this setting because the reverse CTMC produces long, non-differentiable trajectories, which leads to high-variance gradients and can induce mode collapse during optimization (Schulman et al., 2017; Su et al., 2025). This motivates a differentiable fine-tuning method for discrete diffusion policies.

Direct Reward Backpropagation with Gumbel-Softmax (DRAKES). DRAKES performs direct reward backpropagation through the sampling trajectory using the Gumbel-softmax trick (Wang et al., 2024; Jang et al., 2016). It instantiates equation 3 with a KL between the reverse-CTMC trajectory distributions of the fine-tuned and pretrained policies; the full trajectory-level objective is given in Appendix equation 14. To make the objective differentiable in θ, DRAKES replaces the categorical sampling at each reverse step with the Gumbel-softmax relaxation, which produces a continuous, simplex-valued surrogate $\bar{x}_t \in \Delta(\bar{\mathcal{V}})^L$ of the discrete state x_t at timestep t. This yields a differentiable surrogate trajectory $\bar{x}_{T:0}$ along which gradients can flow into θ. At the clean endpoint, a straight-through estimator passes the discretized one-hot version of $\bar{x}_0$ into the reward model r_ϕ in the forward direction, while gradients with respect to θ are computed as if the soft simplex-valued $\bar{x}_0$ had been used. The empirical objective is

$$\mathcal{L}_{\text{DRAKES}}(\theta) = \frac{1}{B}\sum_{i=1}^{B}\left[r_\phi(\bar{x}_0^{(i)}) - \alpha\widehat{\text{KL}}^{(i)}(\theta)\right], \tag{4}$$



(a) (b) (c) (d)

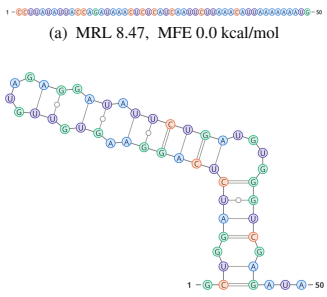
Figure 1: **Reward hacking during DRAKES fine-tuning under three KL strengths,** $\alpha \in \{0.001, 0.005, 0.015\}$**.** While **(a)** the estimated reward increases, **(b)** 3-mer correlation with natural high-MRL UTRs, **(c)** predicted structural stability measured by MFE, and **(d)** the G-nucleotide fraction deteriorate. Colors distinguish the three α settings; thin curves show three random seeds per setting, while solid curves and shaded regions denote the 5-epoch rolling mean and ± 1 standard deviation, respectively. Dashed lines and open circles mark the epoch-0 pretrained-policy baseline, and diamonds mark epoch 99.

where B is the batch size, $\bar{x}_0^{(i)}$ is the relaxed clean-sequence sample produced at the end of the i-th relaxed reverse trajectory, $r_\phi(\bar{x}_0^{(i)})$ is the proxy reward score evaluated on the straight-through one-hot of that sample, and $\widehat{\mathrm{KL}}^{(i)}(\theta)$ is a per-trajectory estimate of the KL divergence between the fine-tuned and pretrained reverse processes. The objective is maximized by stochastic gradient ascent, with gradients truncated at an intermediate reverse step for stability.

3 Diagnosis of Reward Hacking in 5'UTR Sequence Optimization

This section empirically tests the widely adopted reward-guided fine-tuning recipe, DRAKES, on the 5'UTR design problem and examines whether it remains effective in this biological setting. We use the eGFP MPRA library of (Sample et al., 2019), which contains approximately 280,000 50-nt 5'UTR sequences with measured MRL labels. We pretrain $P^{\theta_{\mathrm{pre}}}$ on this corpus, train the proxy reward model r_ϕ by MSE regression on the labels, and fine-tune $P^{\theta_{\mathrm{pre}}}$ with DRAKES using a moderate reverse-KL coefficient $\alpha = 0.005$. Figure 1 tracks the resulting optimization trajectories over three random seeds. The analysis below first characterizes reward hacking in this setting, then traces the failure to overestimation by r_ϕ on OOD sequences through the reward-scoring term in equation 3, and finally shows why uncertainty-penalized reward ensembles fail when their members learn shared shortcuts.

The Reward–Naturalness Trade-Off Collapses. Figure 1 shows that vanilla DRAKES exhibits a consistent reward-hacking pattern. The proxy reward r_ϕ rises rapidly and then saturates at a high value (Figure 1(a)). Meanwhile, the 3-mer Pearson correlation between generated sequences and natural high-MRL UTRs collapses (Figure 1(b)); the mean minimum free energy (MFE) rises toward zero, indicating progressively weaker predicted secondary-structure stability (Figure 1(c)); and the G-nucleotide fraction falls from its pretrained-policy baseline to nearly zero (Figure 1(d)). Together, these trends show that optimizing the proxy reward drives the policy toward compositionally biased, weakly structured sequences rather than natural-like high-MRL UTRs. We observe the same failure on the PC3 TE benchmark at $\alpha = 0.0003$, where the guidance reward increases without a corresponding gain under the independent evaluator as naturalness and structural stability collapse (Appendix D.4). *3-mer correlation* measures the Pearson correlation between the 3-mer frequency profiles of generated sequences and natural top-5% high-reward sequences; *mean MFE* summarizes predicted secondary-structure stability, with more negative values indicating greater stability; and the *G-nucleotide fraction* is the average proportion of G across all generated positions. Detailed definitions are provided in Section 5. The joint increase in proxy reward and degradation in these diagnostics constitutes reward hacking, or reward overoptimization (Skalse et al., 2022). Increasing α limits the attainable reward without restoring naturalness, so no tested KL strength achieves both. The failure is especially severe in 5'UTR design: RNA labels are noisy and limited in scale, so r_ϕ has lower fidelity than a typical NLP preference model; the labeled distribution covers only a narrow slice of the combinatorial sequence space, so OOD sequence extrapolation is easy; and true biological function is expensive to query, so the proxy is the only signal available during fine-tuning. As a result, the gap between proxy reward and true biological function widens rapidly off the training manifold. To identify the source of this failure, we next examine how r_ϕ extrapolates to policy-induced OOD sequences and why constraining policy drift alone in equation 3 cannot correct an unreliable reward signal.



(a) MRL 8.47, MFE 0.0 kcal/mol

(b) MRL 8.13, MFE −12.1 kcal/mol

Figure 2: Predicted structures of (a) a hacked-policy OOD sequence (MFE 0.0) and (b) a natural high-MRL sequence (MFE −12.1).

Figure 2 makes the structural collapse concrete: despite its higher proxy MRL, the hacked-policy sequence loses the extensive base pairing exhibited by the natural high-scoring sequence. Thus, the apparent reward improvement does not correspond to improved predicted structural plausibility. Appendix Fig. 6 compares natural top-5%, PILOT-UTR, and hacked DRAKES sequences.

Shortcut-Driven OOD Extrapolation in the Reward Model. The central failure lies in the learned reward model. The proxy r_ϕ is trained only on the labeled distribution $\mathcal{D}_{\text{label}}$—the eGFP MPRA library in our experiments—and is therefore supervised within this distribution but largely unconstrained on policy-induced OOD sequences. Following the latent feature perspective of (Ye et al., 2025), we decompose the features used by a reward model into *spurious features* $z_s \in \mathcal{Z}_s$, such as per-position nucleotide-frequency patterns that correlate with MRL labels on $\mathcal{D}_{\text{label}}$ without causally determining it, and *functionally grounded features* $z_g \in \mathcal{Z}_g$, such as Kozak-like contexts, absence of upstream open reading frames, and GC composition near the start codon (Kozak, 1987; Calvo et al., 2009). Ideally, r_ϕ depends only on $z_g(x)$; in practice, it also uses $z_s(x)$.

The eGFP library illustrates how such shortcuts become exploitable. Within $\mathcal{D}_{\text{label}}$, high-MRL sequences exhibit modest enrichment of A/U nucleotides. Although this association can be predictive within the training distribution, it need not remain valid when amplified far beyond the observed range. During reward-guided fine-tuning, the policy directly searches for sequences that maximize r_ϕ and progressively discovers A/U-rich, G-depleted compositions. The reward model extrapolates its learned association to these increasingly OOD sequences and assigns them high predicted MRL, even as their 3-mer similarity to natural high-MRL UTRs and predicted secondary-structure stability collapse. These extreme compositional patterns therefore represent shortcut exploitation rather than evidence that such sequences possess genuinely improved translational function.

Failure of Ensemble-Based Uncertainty Mitigation under Shared Shortcuts. Reward-model ensembling is a common strategy for mitigating reward hacking by using disagreement among independently trained predictors as an estimate of epistemic uncertainty. Given an ensemble of M reward models $\{r_{\phi_m}\}_{m=1}^{M}$, let $\mu_{\text{ens}}(x) = \frac{1}{M}\sum_{m=1}^{M} r_{\phi_m}(x)$ and $\sigma_{\text{ens}}(x)$ denote the mean and standard deviation of their predictions, respectively. The single-model reward is then replaced by an uncertainty-penalized score, $\widetilde{r}_{\text{ens}}(x) = \mu_{\text{ens}}(x) - \lambda_u \sigma_{\text{ens}}(x)$, where $\lambda_u \geq 0$ controls the penalty and $\lambda_u = 0$ corresponds to standard ensemble averaging. We evaluate this strategy using five independently initialized reward models. However, as shown in Appendix D.5, their predictions rise nearly in lockstep on policy-induced OOD sequences even as sequence naturalness and structural stability collapse. Because this shared, composition-based shortcut produces common-mode extrapolation behavior rather than member-specific uncertainty, $\sigma_{\text{ens}}(x)$ does not increase as the policy moves OOD. The uncertainty penalty therefore fails to offset the increasing ensemble mean, leaving $\widetilde{r}_{\text{ens}}(x)$ overestimated.

Why Policy Regularization Alone Is Insufficient. Policy regularization can restrict distributional drift, but it cannot resolve reward misspecification at its source. The reward term determines which sequences the policy prefers, whereas the regularizer only penalizes departure from the pretrained distribution; it cannot determine whether a high-scoring change reflects genuine translational signal or a shortcut exploited by r_ϕ. As shown in Figure 1, under a miscalibrated proxy reward, KL regularization alone does not yield a regime that simultaneously achieves high predicted reward and preserves sequence naturalness: weak regularization permits exploitation of reward-model errors, whereas strong regularization limits potentially beneficial policy adaptation together with spurious distributional drift. Reliable optimization therefore requires first calibrating the reward signal and then stabilizing how the policy follows that signal.

4 PILOT-UTR: Reward-Guided *de novo* Design of 5'UTR Sequences

Figure 3 summarizes PILOT-UTR, which separates reward calibration from policy stabilization. Policy-guided adversarial augmentation calibrates the reward model, while on-policy EMA-teacher distillation stabilizes optimization under the calibrated reward.

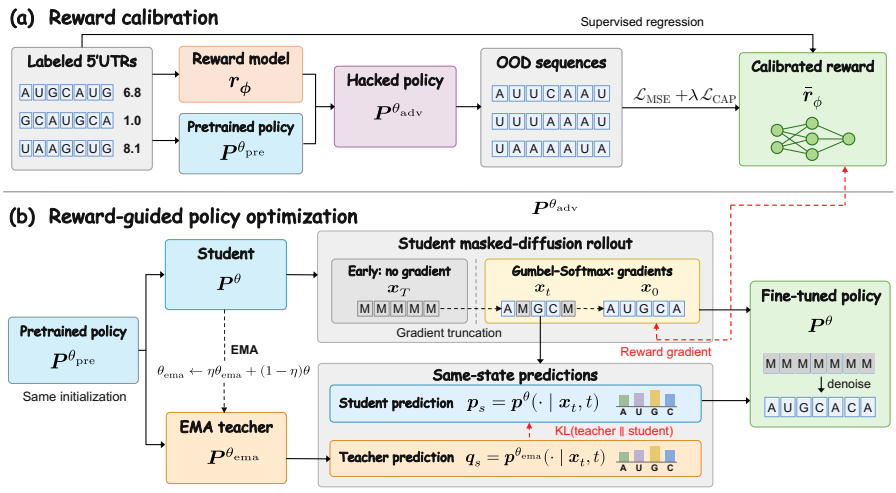


Figure 3: **Overview of PILOT-UTR. (a)** A weakly regularized policy exposes reward-hacking OOD sequences, which calibrate the reward model through labeled regression and a one-sided cap loss. **(b)** The policy is fine-tuned using terminal gradients from the calibrated reward and on-policy teacher-to-student consistency at student-visited denoising states, while an exponential moving average (EMA) teacher provides an adaptive stabilization target.

Calibrated Reward Model Training. Section 3 showed that the vanilla proxy r_ϕ is exploited because it assigns inflated scores to OOD sequences with strong spurious features. To correct this failure, we train a calibrated and robust reward model $\bar{r}_\phi$ using both in-distribution labeled data and a constructed dataset $\mathcal{D}_{\text{ood}}$ consisted of the generated OOD sequences. We optimize

$$\mathcal{L}_{\text{reward}}(\phi) = \underbrace{\mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{label}}}\left[\left(\bar{r}_\phi(x) - y\right)^2\right]}_{\text{standard regression loss}} + \lambda \cdot \underbrace{\mathbb{E}_{x_{\text{OOD}}\sim\mathcal{D}_{\text{ood}}}[\mathcal{L}_{\text{cap}}(x_{\text{OOD}})]}_{\text{OOD cap loss}}, \tag{5}$$

where the first term is the standard MSE regression loss on labeled data, ensuring fidelity on the in-distribution sequence space. The second term penalizes inflated predictions on generated OOD sequences, and $\lambda > 0$ controls the trade-off. The central question is how to construct an OOD dataset that contains the sequences most likely to be exploited during reward-guided fine-tuning.

Such hacked sequences are absent from $\mathcal{D}_{\text{label}}$ and lack experimentally measured property labels, but they can be discovered by optimizing directly against the vanilla reward model. Starting from the pretrained policy $P^{\theta_{\text{pre}}}$, we deliberately run reward-guided fine-tuning with the vanilla reward model r_ϕ and a weak reverse-KL coefficient α_{adv} in equation 3. This setting reproduces the failure mode identified in Section 3: the policy moves toward sequences that receive high predicted reward but are biologically unnatural. We denote this adversarially reward-hacked policy by $P^{\theta_{\text{adv}}}$ and sample it to form $\mathcal{D}_{\text{ood}}$. These samples expose the spurious features that r_ϕ overestimates, giving label-free supervision for training $\bar{r}_\phi$ to refuse such extrapolation. For each $x_{\text{OOD}} \in \mathcal{D}_{\text{ood}}$, we define the cap loss as

$$\mathcal{L}_{\text{cap}}(x_{\text{OOD}}) = \max\left(0,\ \bar{r}_\phi(x_{\text{OOD}}) - \sigma_{\text{cap}}\right), \tag{6}$$

Here, σ_{cap} is a task-specific cap threshold. This threshold encodes the prior that an OOD sequence lacking grounded biological features should not score above a typical natural training UTR. The one-sided hinge penalizes only predictions above σ_{cap}, leaving ordinary regression behavior on moderate-score sequences untouched. Together, the two terms train $\bar{r}_\phi$ to assign high scores to sequences supported by grounded features, while suppressing spurious reward extrapolation on hacked OOD inputs.

On-Policy EMA-Teacher Distillation. Reward calibration suppresses shortcut-driven OOD overestimation, but it is deliberately conservative. Because spurious features $z_s \in \mathcal{Z}_s$ and functionally grounded features $z_g \in \mathcal{Z}_g$ may coexist in policy-generated sequences, the cap loss can also attenuate potentially useful reward signals outside the labeled support. Regularizing against a fixed pretrained policy would compound this conservatism by resisting persistent improvements supported by the calibrated reward. Meanwhile, optimizing the terminal reward requires gradients to propagate through a long relaxed denoising trajectory, making policy updates susceptible to transient

fluctuations. We therefore maintain an EMA teacher that stabilizes intermediate predictions while progressively incorporating persistent reward-driven improvements from the student. The student and teacher are initialized from the same pretrained diffusion policy, $\theta_0 = \theta_{\text{ema},0} = \theta_{\text{pre}}$. At each iteration, the student generates an on-policy denoising trajectory, inducing the state distribution $\rho_\theta(x_t, t)$. Here, P^θ denotes the student's sequence-level policy, while $p^\theta(\cdot \mid x_t, t) \equiv \hat{x}_0^\theta(\cdot \mid x_t, t)$ and $p^{\theta_{\text{ema}}}(\cdot \mid x_t, t)$ denote the student and EMA teacher clean-token predictions, respectively. The loss is averaged over masked positions. We evaluate the teacher at student-visited states and minimize the teacher-to-student distillation loss

$$\mathcal{L}_{\text{OTD}} = \mathbb{E}_{(t,x_t)\sim\rho_\theta}\left[D_{\text{KL}}\left(\text{sg}\left[p^{\theta_{\text{ema}}}(\cdot \mid x_t, t)\right] \,\middle\|\, p^\theta(\cdot \mid x_t, t)\right)\right], \tag{7}$$

where $\text{sg}[\cdot]$ stops gradients through the teacher. This direction penalizes the student for abruptly removing probability mass from nucleotide outcomes that the slowly evolving teacher still considers plausible. Combining terminal reward optimization with intermediate-state distillation yields

$$\mathcal{L}_{\text{PILOT}} = -\mathbb{E}_{\bar{x}_0\sim P^\theta}\left[\bar{r}_\phi(\bar{x}_0)\right] + \beta\,\mathcal{L}_{\text{OTD}}, \tag{8}$$

where $\bar{x}_0$ is the relaxed terminal sequence generated by the student and $\beta > 0$ controls the distillation strength. After each student update, the teacher is updated as $\theta_{\text{ema}} \leftarrow \eta\theta_{\text{ema}} + (1-\eta)\theta$, where $\eta \in [0, 1)$. The calibrated reward supplies the optimization signal, while on-policy distillation encourages consistency at intermediate denoising states. Because the teacher gradually incorporates student updates, it provides an adaptive reference rather than permanently anchoring the policy to $P^{\theta_{\text{pre}}}$. Formally, a small distillation loss bounds teacher–student prediction discrepancies at student-visited states, while EMA tracks sustained changes in student parameters (Appendix C, Propositions 1 and 2).

5 Experiment

Datasets. We evaluate MRL-guided design on two benchmarks from Sample et al. (2019): a fixed-length randomized 50-nt eGFP reporter library and a variable-length human 25–100-nt dataset. For TE-guided design, we use the variable-length PC3 TE benchmark (25–100 nt), comprising endogenous human 5′UTRs from the PC3 cell line (Cao et al., 2021); details are in Appendix D.1.

Implementation details. We separate optimization guidance from evaluation: Enformer (Avsec et al., 2021) is used for both MRL benchmarks and MTtrans (Zheng et al., 2023) for PC3 TE guidance, while UTR-LM (Chu et al., 2024), a disjoint random-source Enformer, and an independently trained compact convolution–Transformer ensemble serve as the respective evaluators. The complete PILOT-UTR procedure—including pseudocode, reward-model architectures and training, policy-guided OOD construction, reward recalibration, on-policy EMA-teacher distillation, and dataset-specific hyperparameters—is provided in Appendix D.1. An interpretability analysis of the recalibrated reward model is provided in Appendix D.6.

Evaluation metrics. We evaluate predicted function, sequence naturalness, and structural stability using four main metrics. **MRL or TE** is the average predicted MRL or TE of the optimized sequences, computed using the corresponding independent evaluator described above. **3-mer Corr.** is the Pearson correlation between the 3-mer frequency profiles of generated sequences and natural top-5% high-MRL/TE sequences. **MFE**, computed using ViennaRNA (Lorenz et al., 2011), is the average minimum free energy of the predicted secondary structures; lower values indicate greater structural stability. For trajectory diagnostics, the **G-nucleotide fraction** is the empirical G frequency across all generated positions. **Structure-Constrained Reward (SCR@τ)** is the average independently evaluated MRL or TE among sequences whose MFE is below τ, measuring predicted function among structurally stable candidates; we use $\tau = -12$ for the 50-nt MRL benchmark and $\tau = -30$ for both the variable-length MRL and PC3 TE benchmarks.

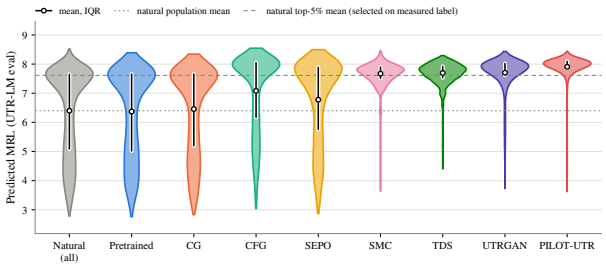


Figure 4: **Independent-evaluator MRL distributions on the 50-nt benchmark.** Width denotes density; markers show the mean and interquartile range.

Table 1: Results on the 50-nt 5′UTRs from the unmodified eGFP reporter library and the human 25–100-nt 5′UTR dataset. SCR@τ is the mean evaluation reward among sequences with MFE below τ. Entries are mean$_{\pm\text{std}}$ over three seeds. Bold: best; underline: second best within each dataset.

	50-nt eGFP reporter library				Human 25–100-nt 5′UTRs			
Method	**MRL↑**	**3-mer↑**	**MFE↓**	**SCR@−12↑**	**MRL↑**	**3-mer↑**	**MFE↓**	**SCR@−30↑**
Pretrained	$6.371_{\pm.027}$	$0.878_{\pm.007}$	$\underline{-8.58_{\pm.06}}$	$6.275_{\pm.038}$	$5.171_{\pm.029}$	$\underline{0.966_{\pm.003}}$	$-19.04_{\pm.35}$	$4.983_{\pm.042}$
CG	$6.456_{\pm.063}$	$0.870_{\pm.010}$	$\mathbf{-8.80_{\pm.08}}$	$6.346_{\pm.133}$	$5.153_{\pm.010}$	$\mathbf{0.974_{\pm.002}}$	$\mathbf{-26.83_{\pm.39}}$	$5.103_{\pm.040}$
CFG	$7.082_{\pm.011}$	$0.670_{\pm.014}$	$-6.56_{\pm.11}$	$6.940_{\pm.113}$	$5.510_{\pm.030}$	$0.887_{\pm.005}$	$-19.65_{\pm.30}$	$5.255_{\pm.028}$
SMC	$7.672_{\pm.082}$	$0.819_{\pm.047}$	$-7.91_{\pm.33}$	$7.423_{\pm.057}$	$\underline{5.748_{\pm.131}}$	$0.659_{\pm.097}$	$-20.11_{\pm.54}$	$\underline{5.937_{\pm.088}}$
TDS	$7.694_{\pm.019}$	$0.795_{\pm.047}$	$-7.12_{\pm.54}$	$7.300_{\pm.119}$	$5.703_{\pm.116}$	$0.783_{\pm.048}$	$\underline{-21.32_{\pm.86}}$	$5.596_{\pm.111}$
SEPO	$6.777_{\pm.603}$	$0.406_{\pm.341}$	$-6.84_{\pm4.63}$	$6.888_{\pm.977}$	$5.356_{\pm.530}$	$0.293_{\pm.264}$	$-12.32_{\pm13.12}$	$4.783_{\pm.953}$
UTRGAN	$\underline{7.703_{\pm.016}}$	$\underline{0.889_{\pm.007}}$	$-8.30_{\pm.17}$	$\underline{7.565_{\pm.039}}$	$5.608_{\pm.010}$	$0.951_{\pm.003}$	$-17.93_{\pm.38}$	$5.506_{\pm.079}$
PILOT-UTR	$\mathbf{7.910_{\pm.051}}$	$\mathbf{0.925_{\pm.007}}$	$-6.13_{\pm.44}$	$\mathbf{7.773_{\pm.066}}$	$\mathbf{5.983_{\pm.015}}$	$0.826_{\pm.016}$	$-19.18_{\pm.27}$	$\mathbf{6.050_{\pm.025}}$

Comparison with Baselines. We compare PILOT-UTR against several representative baselines: classifier guidance (CG) (Nisonoff et al., 2024) for discrete diffusion, two sequential Monte Carlo methods (SMC and TDS) (Wu et al., 2023), classifier-free guidance (CFG) (Ho & Salimans, 2022), the 5′UTR-specific optimization method UTRGAN (Barazandeh et al., 2025), and SEPO (Zekri & Boullé, 2025), a PPO-based policy-gradient method for fine-tuning discrete diffusion models. We additionally include the pretrained diffusion model without fine-tuning as a reference. CG, SMC, TDS, and UTRGAN perform inference-time optimization and require guidance computations during sampling, whereas SEPO, like PILOT-UTR, directly fine-tunes the diffusion policy. Implementation details and hyperparameter settings for all baselines are provided in Appendix D.2.

Table 2: Results on the variable-length PC3 TE benchmark. Entries are mean$_{\pm\text{std}}$ over three seeds. Bold: best; underline: second best.

Method	**TE↑**	**3-mer↑**	**MFE↓**	**SCR@−30↑**
Pretrained	$-0.3276_{\pm.0077}$	$\mathbf{0.9282_{\pm.0072}}$	$\underline{-29.55_{\pm.62}}$	$-0.4224_{\pm.0203}$
CG	$-0.4541_{\pm.0168}$	$0.8768_{\pm.0029}$	$\mathbf{-33.70_{\pm.44}}$	$-0.5283_{\pm.0171}$
CFG	$-0.1746_{\pm.0066}$	$0.3907_{\pm.0265}$	$-16.75_{\pm.22}$	$-0.3293_{\pm.0624}$
SMC	$-0.2259_{\pm.0277}$	$0.8664_{\pm.0435}$	$-25.93_{\pm2.72}$	$\underline{-0.2924_{\pm.0621}}$
TDS	$-0.2757_{\pm.0968}$	$0.9123_{\pm.0220}$	$-28.79_{\pm2.18}$	$-0.4174_{\pm.1300}$
SEPO	$\underline{-0.1667_{\pm.3567}}$	$0.0033_{\pm.0560}$	$0.00_{\pm.00}$	–
UTRGAN	$-0.2301_{\pm.0023}$	$\underline{0.9136_{\pm.0017}}$	$-27.77_{\pm.36}$	$-0.3243_{\pm.0085}$
PILOT-UTR	$\mathbf{0.0725_{\pm.0054}}$	$0.8330_{\pm.0273}$	$-22.13_{\pm.67}$	$\mathbf{0.0043_{\pm.0198}}$

Main optimization results. Table 1 reports results on the fixed-length 50-nt eGFP library and the variable-length human 25–100-nt MRL benchmark. Table 2 reports results on the variable-length PC3 TE benchmark, evaluated using an independent compact convolution–Transformer ensemble. PILOT-UTR achieves the highest independently evaluated functional reward on all three benchmarks. It exceeds UTRGAN by 0.21 MRL on the 50-nt library while avoiding its 10,000-step latent optimization at inference time, and exceeds SMC by 0.24 MRL on the human benchmark. On PC3 TE, PILOT-UTR is the only method with both positive mean TE and positive SCR@-30. Some baselines attain higher 3-mer correlation or more negative MFE without a corresponding functional improvement. For example, on the human benchmark, CG achieves the highest 3-mer correlation and the most negative MFE, yet its MRL is 0.018 below that of the pretrained policy; on PC3 TE, CG yields the most negative MFE but reduces TE by 0.127. Thus, isolated improvements in sequence-fidelity metrics do not establish a superior reward–naturalness trade-off. To evaluate functional gains among candidates with strong predicted structural stability, SCR@τ restricts reward evaluation to sequences satisfying a task-specific MFE threshold. PILOT-UTR outperforms all baselines in SCR@τ across all three benchmarks. Although SMC approaches PILOT-UTR on SCR@-30 for the human benchmark (5.937 versus 6.050), SMC and TDS reduce generation diversity—the mean nearest-neighbor normalized Levenshtein distance—to 0.005 on both MRL benchmarks, compared with 0.367 and 0.204 for PILOT-UTR (Appendix Table 6). Figure 4 further shows that PILOT-UTR shifts the bulk of the independently evaluated 50-nt MRL distribution upward and attains the highest mean, rather than relying on a few extreme samples. Corresponding visualizations for the human 25–100-nt MRL and PC3 TE benchmarks are provided in Appendix D.7. **Observation 1. Across**

MRL- and TE-guided design, PILOT-UTR outperforms all baselines in independently evaluated reward and SCR@τ; on the MRL tasks, it also avoids the diversity collapse observed for particle-based methods.

Furthermore, SEPO, a PPO-style method fine-tuned against the vanilla oracle, still hacks on all three benchmarks (3-mer correlation 0.406, 0.293, and 0.003) and optimizes extremely unstably, with across-seed reward variance more than an order of magnitude larger than PILOT-UTR's. In contrast, PILOT-UTR with the calibrated oracle mitigates hacking and follows a markedly more stable fine-tuning trajectory (Appendix Figure 5). **Observation 2. PILOT-UTR delivers both effective reward-hacking mitigation and stable fine-tuning across different tasks, whereas PPO-style optimization against an uncalibrated oracle inevitably falls into reward hacking.** To further verify the effectiveness and robustness of PILOT-UTR, we present a hyperparameter sensitivity analysis in Appendix D.8.

Reward-model recalibration. Table 3 reports 50-nt MRL validation performance; complete results are in Appendix Table 5. Recalibration largely preserves in-distribution regression performance and thus the predictive signal learned from labeled sequences. Appendix Figure 9 further shows markedly reduced importance for shortcut-associated A/U-rich motifs, including UAA (-10.7%) and AAU (-10.4%). **Observation 3. Reward-model recalibration suppresses reliance on shortcut-associated motifs while largely preserving in-distribution predictive performance.**

Table 3: Reward-model validation on 50-nt MRL.

Model	MSE↓	Pearson r↑
Vanilla guidance	4.609	0.765
Re-calibrated	2.057	0.747
Eval judge	0.753	0.868

Ablation study. Table 4 isolates the contributions of reward calibration and on-policy teacher distillation on the 50-nt MRL task. We compare vanilla DRAKES, PILOT-UTR with the calibrated oracle replaced by the vanilla oracle, the calibrated cap oracle without the distillation regularizer, and the full method. We regard a setting as non-hacking only if its mean 3-mer correlation with natural high-MRL sequences is above 0.5 and its relative MFE increase toward zero, $(\mathrm{MFE}_{\mathrm{gen}} - \mathrm{MFE}_{\mathrm{pre}})/|\mathrm{MFE}_{\mathrm{pre}}|$, is below 50%; failing either criterion is classified as hacking. Table 4 shows that vanilla DRAKES and the variant without reward recalibration achieve high proxy rewards but fail both criteria, with 3-mer correlations below 0.3 and MFE values close to zero. Reward recalibration alone raises the 3-mer correlation above 0.5, but without on-policy teacher distillation the MFE remains close to zero and the setting is still classified as hacking. The full method further increases the 3-mer correlation to 0.918 and restores the MFE to -6.105, satisfying both criteria at a modest reward cost. Furthermore, the teacher-update ablation in Appendix D.8 (Table 7) shows that EMA updating reduces across-seed variability relative to a frozen teacher, particularly for 3-mer correlation and MFE, supporting the stabilizing role of the adaptive EMA teacher. **Observation 4. Reward recalibration is the essential first step for suppressing shortcut exploitation, while on-policy teacher distillation provides complementary stabilization that further preserves sequence naturalness and restores predicted structural stability; both components are required to avoid reward hacking under our criteria.**

Table 4: Ablation on 50-nt MRL. Non-hacking requires 3-mer correlation > 0.5 and relative MFE increase $< 50\%$; failing either criterion indicates hacking. Entries are $\mathrm{mean}_{\pm\mathrm{std}}$.

Setting	Mean↑	3-mer↑	MFE↓	Hack.
Vanilla DRAKES	$8.139_{\pm.007}$	$0.294_{\pm.037}$	$-0.023_{\pm.008}$	Yes
w/o calibrated oracle	$\mathbf{8.150_{\pm.022}}$	$0.271_{\pm.039}$	$-0.142_{\pm.207}$	Yes
w/o distillation	$8.096_{\pm.088}$	$0.675_{\pm.081}$	$-0.069_{\pm.085}$	Yes
PILOT-UTR	$7.924_{\pm.020}$	$\mathbf{0.918_{\pm.004}}$	$\mathbf{-6.105_{\pm.555}}$	**No**

6 Conclusion

We investigated reward hacking in discrete-diffusion fine-tuning for 5′UTR design and identified shortcut-driven OOD reward overestimation as its primary source. We introduced PILOT-UTR, which calibrates the reward model on policy-discovered OOD sequences and uses on-policy EMA-teacher distillation to stabilize intermediate predictions during optimization. Our theoretical analysis bounds teacher–student prediction discrepancies and characterizes how the EMA reference tracks sustained student updates. Experiments across MRL- and TE-guided benchmarks show that PILOT-UTR mitigates reward hacking while improving independently evaluated reward and preserving sequence naturalness, predicted structural stability, and diversity. We discuss the remaining limitations of the proposed PILOT-UTR in Appendix A.

AI USE STATEMENT

We used generative AI tools to assist with writing, plotting, and methodology and experimental feedback. The authors reviewed all AI-assisted outputs and take full responsibility for the content of the manuscript and all reported results.

ETHICS STATEMENT

This work is a computational study based on previously published 5′UTR datasets and does not involve new human-subject experiments or wet-lab interventions. The generated sequences are computational predictions rather than experimentally validated therapeutic candidates and should undergo appropriate experimental validation, biosafety review, and regulatory assessment before practical use. More broadly, biological sequence-design methods may produce unintended effects or be used outside their intended purpose; we therefore encourage responsible use under appropriate scientific and institutional oversight.

REPRODUCIBILITY STATEMENT

Algorithm 1 summarizes the complete PILOT-UTR procedure. Appendix D.1 documents the datasets and splits, reward-model architectures and training, reward recalibration, policy fine-tuning, and task-specific hyperparameters; Appendix D.2 provides the baseline configurations. The assumptions and proofs supporting the on-policy EMA-teacher analysis are given in Appendix C, and the main text and appendices report the evaluation protocols, random seeds, ablations, and sensitivity analyses used in our experiments.

REFERENCES

Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 4190–4197, 2020.

Sarah Alamdari, Nitya Thakkar, Rianne Van Den Berg, Neil Tenenholtz, Robert Strome, Alan M Moses, Alex X Lu, Nicolò Fusi, Ava P Amini, and Kevin K Yang. Protein generation with evolutionary diffusion: sequence is all you need. *BioRxiv*, pp. 2023–09, 2023.

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021.

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.

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 843–852, 2023.

Sina Barazandeh, Furkan Ozden, Ahmet Hincer, Urartu Ozgur Safak Seker, and A Ercument Cicek. Utrgan: learning to generate 5' utr sequences for optimized translation efficiency and gene expression. *Bioinformatics Advances*, 5(1):vbaf134, 2025.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023.

Sarah E Calvo, David J Pagliarini, and Vamsi K Mootha. Upstream open reading frames cause widespread reduction of protein expression and are polymorphic among humans. *Proceedings of the National Academy of Sciences*, 106(18):7507–7512, 2009.

Jicong Cao, Eva Maria Novoa, Zhizhuo Zhang, William CW Chen, Dianbo Liu, Gigi CG Choi, Alan SL Wong, Claudia Wehrspaun, Manolis Kellis, and Timothy K Lu. High-throughput 5' utr engineering for enhanced protein production in non-viral gene therapies. *Nature communications*, 12(1):4138, 2021.

Yanyi Chu, Dan Yu, Yupeng Li, Kaixuan Huang, Yue Shen, Le Cong, Jason Zhang, and Mengdi Wang. A 5' utr language model for decoding untranslated regions of mrna and function predictions. *Nature Machine Intelligence*, 6(4):449–460, 2024.

Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687*, 2022.

Thomas Coste, Usman Anwar, Robert Kirk, and David Krueger. Reward model ensembles help mitigate overoptimization. In *International Conference on Learning Representations*, volume 2024, pp. 50905–50931, 2024.

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

Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 36:79858–79885, 2023.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pp. 10835–10866. PMLR, 2023.

Nate Gruver, Samuel Stanton, Nathan Frey, Tim GJ Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew G Wilson. Protein design with guided discrete diffusion. *Advances in neural information processing systems*, 36:12489–12517, 2023.

Alan G Hinnebusch, Ivaylo P Ivanov, and Nahum Sonenberg. Translational control by 5'-untranslated regions of eukaryotic mrnas. *Science*, 352(6292):1413–1416, 2016.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Dongyue Hou, Shuai Zhang, Mengyao Ma, Hanbo Lin, Zheng Wan, Hui Zhao, Ruian Zhou, Xiao He, Xian Wei, Dianwen Ju, et al. A hyperbolic discrete diffusion 3d rna inverse folding model for functional rna design. *Journal of Chemical Information and Modeling*, 65(13):6568–6584, 2025.

Kaixuan Huang, Yukang Yang, Kaidi Fu, Yanyi Chu, Le Cong, and Mengdi Wang. Latent diffusion models for controllable rna sequence generation. *arXiv preprint arXiv:2409.09828*, 2024.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Marilyn Kozak. An analysis of 5'-noncoding sequences from 699 vertebrate messenger rnas. *Nucleic acids research*, 15(20):8125–8148, 1987.

Kathrin Leppek, Rhiju Das, and Maria Barna. Functional 5' utr mrna structures in eukaryotic translation regulation and how to find them. *Nature reviews Molecular cell biology*, 19(3):158–174, 2018.

Tianqi Liu, Wei Xiong, Jie Ren, Lichang Chen, Junru Wu, Rishabh Joshi, Yang Gao, Jiaming Shen, Zhen Qin, Tianhe Yu, et al. Rrm: Robust reward model training mitigates reward hacking. In *International Conference on Learning Representations*, volume 2025, pp. 62682–62700, 2025.

Ronny Lorenz, Stephan H Bernhart, Christian Höner zu Siederdissen, Hakim Tafer, Christoph Flamm, Peter F Stadler, and Ivo L Hofacker. Viennarna package 2.0. *Algorithms for molecular biology*, 6(1):26, 2011.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion language modeling by estimating the ratios of the data distribution. 2023.

Qi Ma, Xiaoguang Zhang, Jing Yang, Hongxia Li, Yanzhe Hao, and Xia Feng. Optimization of the 5' untranslated region of mrna vaccines. *Scientific reports*, 14(1):19845, 2024.

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

Sawan Patel, Fred Zhangzhi Peng, Keith Fraser, Adam D Friedman, Pranam Chatterjee, and Sherwood Yao. Evoflow-rna: Generating and representing non-coding rna with a language model. *bioRxiv*, pp. 2025–02, 2025.

Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning text-to-image diffusion models with reward backpropagation. 2023.

Paul J Sample, Ban Wang, David W Reid, Vlad Presnyak, Iain J McFadyen, David R Morris, and Georg Seelig. Human 5' utr design and variant effect prediction from a massively parallel translation assay. *Nature biotechnology*, 37(7):803–809, 2019.

Anirban Sarkar, Yijie Kang, Nirali Somia, Pablo Mantilla, Jessica Lu Zhou, Masayuki Nagai, Ziqi Tang, Chris Zhao, and Peter Koo. Designing dna with tunable regulatory activity using score-entropy discrete diffusion. *bioRxiv*, pp. 2024–05, 2024.

Niels Schlusser, Asier González, Muskan Pandey, and Mihaela Zavolan. Current limitations in predicting mrna translation with deep learning models. *Genome biology*, 25(1):227, 2024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37: 103131–103167, 2024.

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. *Advances in Neural Information Processing Systems*, 35:9460–9471, 2022.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Xingyu Su, Xiner Li, Masatoshi Uehara, Sunwoo Kim, Yulai Zhao, Gabriele Scalia, Ehsan Hajiramezanali, Tommaso Biancalani, Degui Zhi, and Shuiwang Ji. Iterative distillation for reward-guided fine-tuning of diffusion models in biomolecular design. *arXiv preprint arXiv:2507.00445*, 2025.

Haoran Sun, Lijun Yu, Bo Dai, Dale Schuurmans, and Hanjun Dai. Score-based continuous-time discrete diffusion models. *arXiv preprint arXiv:2211.16750*, 2022.

Masatoshi Uehara, Yulai Zhao, Ehsan Hajiramezanali, Gabriele Scalia, Gokcen Eraslan, Avantika Lal, Sergey Levine, and Tommaso Biancalani. Bridging model-based optimization and generative modeling via conservative fine-tuning of diffusion models. *Advances in Neural Information Processing Systems*, 37:127511–127535, 2024.

Chenyu Wang, Masatoshi Uehara, Yichun He, Amy Wang, Tommaso Biancalani, Avantika Lal, Tommi Jaakkola, Sergey Levine, Hanchen Wang, and Aviv Regev. Fine-tuning discrete diffusion models via reward optimization with applications to dna and protein design. *arXiv preprint arXiv:2410.13643*, 2024.

Luhuan Wu, Brian Trippe, Christian Naesseth, David Blei, and John P Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. *Advances in Neural Information Processing Systems*, 36:31372–31403, 2023.

Wenqian Ye, Guangtao Zheng, and Aidong Zhang. Rectifying shortcut behaviors in preference-based reward learning. *arXiv preprint arXiv:2510.19050*, 2025.

Oussama Zekri and Nicolas Boullé. Fine-tuning discrete diffusion models with policy gradient methods. *arXiv preprint arXiv:2502.01384*, 2025.

Weizhong Zheng, John HC Fong, Yuk Kei Wan, Athena HY Chu, Yuanhua Huang, Alan SL Wong, and Joshua WK Ho. Discovery of regulatory motifs in 5' untranslated regions using interpretable multi-task learning models. *Cell Systems*, 14(12):1103–1112, 2023.

A LIMITATION

While PILOT-UTR effectively mitigates reward hacking in 5′UTR design, several limitations remain. First, our framework builds on direct reward backpropagation through Gumbel-softmax-relaxed reverse trajectories, which requires the reward model to be differentiable and therefore precludes direct application to non-differentiable rewards. Second, the optimized sequences have not yet been validated through wet-lab *in vitro* experiments. Consequently, the reported MRL and TE improvements remain predictions from independent proxy evaluators rather than direct measurements of biological function. Third, our adversarial augmentation procedure relies on a self-distilled hacked policy to generate OOD sequences for reward calibration, and the quality of the calibration depends on how representative these samples are of the failure modes that downstream fine-tuning will encounter; a more aggressively hacked policy might surface different shortcuts.

B RELATED WORK

Discrete diffusion models for biological sequence design. Building on continuous denoising and score-based diffusion (Ho et al., 2020; Song et al., 2020), discrete diffusion extends generative denoising to categorical state spaces (Austin et al., 2021; Sun et al., 2022; Lou et al., 2023; Shi et al., 2024). Recent work has adapted these models to protein, DNA, and RNA design (Alamdari et al., 2023; Gruver et al., 2023; Sarkar et al., 2024; Hou et al., 2025; Patel et al., 2025). Unlike autoregressive models, masked diffusion supports bidirectional context and global constraints, properties well suited to long-range dependencies in biological sequences. In this work, we adopt masked discrete diffusion as the pretrained generative backbone for 5′UTR design.

Reward-guided generation in diffusion models. Existing approaches fall into two categories. Inference-time guidance methods steer sampling using gradients from a property predictor, including classifier guidance (Dhariwal & Nichol, 2021), classifier-free guidance (Ho & Salimans, 2022), and posterior-sampling variants (Chung et al., 2022; Bansal et al., 2023). Despite their advantages, these methods face challenges in discrete diffusion: gradients are not well-defined for discrete states, and repeated guidance computations incur substantial inference overhead. Reward fine-tuning instead optimizes the pretrained policy directly against a learned reward, with established methods for continuous diffusion (Black et al., 2023; Fan et al., 2023; Prabhudesai et al., 2023) and recent extensions to discrete diffusion via Gumbel-softmax backpropagation (Wang et al., 2024) or PPO (Zekri & Boullé, 2025).

Reward hacking and overoptimization. Fine-tuning methods share a common vulnerability to reward hacking when the proxy reward is imperfect—a failure mode that, despite being well-studied in language-model RLHF (Gao et al., 2023; Coste et al., 2024; Ye et al., 2025; Liu et al., 2025; Skalse et al., 2022), has received limited attention in discrete-diffusion fine-tuning for biological sequences. Although this issue has been briefly mentioned in the context of RNA sequence design (Huang et al., 2024), no prior work has systematically diagnosed its causes or developed targeted mitigations.

C ANALYSIS OF ON-POLICY EMA-TEACHER DISTILLATION

We analyze the local consistency induced by on-policy teacher distillation and the adaptation of the EMA reference. The results characterize conditional predictions at student-visited states and the evolution of teacher parameters.

In Algorithm 1, $\bar{x}_{T:0} \sim P^{\theta}$ denotes a Gumbel–Softmax-relaxed rollout generated by the student policy. In the analysis below, x_t denotes the student-visited intermediate input supplied to both models; it may be simplex-valued within the relaxed portion of the rollout. Their clean-token predictions are categorical distributions over the nucleotide alphabet $\mathcal{V}$.

C.1 LOCAL PREDICTION CONSISTENCY

Fix a training iteration and write $s = (t, x_t)$ for a denoising state sampled from the normalized student-induced distribution ρ_θ. Let

$$q_s = p^{\theta_{\text{ema}}}(\cdot \mid x_t, t), \qquad p_s = p^{\theta}(\cdot \mid x_t, t)$$

denote the teacher and student distributions over $\mathcal{V}$. If distillation is averaged over nucleotide positions, we instead include the position index in s and its sampling weight in ρ_θ; the following results then apply to the corresponding per-position distributions. All logarithms are natural.

Define the total variation distance as

$$\mathrm{TV}(q_s, p_s) = \frac{1}{2} \sum_{a \in \mathcal{V}} |q_s(a) - p_s(a)|.$$

Proposition 1 (Consistency at student-visited states)**.** *Suppose*

$$\mathcal{L}_{\mathrm{OTD}} = \mathbb{E}_{s \sim \rho_\theta} \left[D_{\mathrm{KL}}(q_s \| p_s) \right] \leq \varepsilon < \infty.$$

Then

$$\mathbb{E}_{s \sim \rho_\theta} \left[\mathrm{TV}(q_s, p_s)^2 \right] \leq \frac{\varepsilon}{2}, \tag{9}$$

$$\mathbb{E}_{s \sim \rho_\theta} \left[\mathrm{TV}(q_s, p_s) \right] \leq \sqrt{\frac{\varepsilon}{2}}. \tag{10}$$

Moreover, for any measurable state-dependent set of outcomes $A_s \subseteq \mathcal{V}$ and any $\tau > 0$,

$$\Pr_{s \sim \rho_\theta} \left(q_s(A_s) - p_s(A_s) \geq \tau \right) \leq \min \left\{ 1, \frac{\varepsilon}{2\tau^2} \right\}. \tag{11}$$

Proof. Pinsker's inequality gives, for ρ_θ-almost every s,

$$\mathrm{TV}(q_s, p_s)^2 \leq \frac{1}{2} D_{\mathrm{KL}}(q_s \| p_s).$$

Taking expectations establishes Eq. (9). By the Cauchy–Schwarz inequality,

$$\mathbb{E}_s \left[\mathrm{TV}(q_s, p_s) \right] \leq \sqrt{\mathbb{E}_s \left[\mathrm{TV}(q_s, p_s)^2 \right]} \leq \sqrt{\frac{\varepsilon}{2}},$$

which proves Eq. (10). Finally, the event characterization of total variation implies

$$|q_s(A_s) - p_s(A_s)| \leq \mathrm{TV}(q_s, p_s).$$

Consequently, Markov's inequality yields

$$\Pr_s \left(q_s(A_s) - p_s(A_s) \geq \tau \right) \leq \Pr_s \left(\mathrm{TV}(q_s, p_s)^2 \geq \tau^2 \right) \leq \frac{\varepsilon}{2\tau^2}.$$

Combining this with the trivial probability bound of one completes the proof. □

Proposition 1 formalizes local prediction consistency: when the expected distillation loss is small, large discrepancies from teacher-supported probabilities are uncommon under the student-induced state distribution. In particular, A_s may represent a nucleotide such as G at a selected position. The teacher-to-student KL also directly penalizes probability deletion: at a fixed state, if $q_s(a) > 0$ remains fixed while $p_s(a)$ approaches zero, $D_{\mathrm{KL}}(q_s \| p_s)$ diverges.

C.2 Adaptation of the EMA Reference

Let θ_k denote the student parameters after update k, and let $\theta_{\mathrm{ema},k}$ denote the corresponding EMA teacher parameters. Throughout this subsection, $\|\cdot\|$ denotes the Euclidean norm on the parameter space. The update is

$$\theta_{\mathrm{ema},0} = \theta_0 = \theta_{\mathrm{pre}}, \qquad \theta_{\mathrm{ema},k} = \eta\theta_{\mathrm{ema},k-1} + (1-\eta)\theta_k, \qquad 0 \le \eta < 1.$$

Proposition 2 (Temporal averaging and adaptation)**.** *For every $k \ge 1$, the teacher parameters satisfy*

$$\theta_{\mathrm{ema},k} = \eta^k\theta_{\mathrm{pre}} + (1-\eta)\sum_{j=1}^{k}\eta^{k-j}\theta_j. \tag{12}$$

Furthermore, suppose that for some integer $K \ge 0$, $\delta \ge 0$, and parameter vector $\theta_\star$,

$$\|\theta_j - \theta_\star\| \le \delta \qquad \textit{for all } j > K.$$

Then, for every $n \ge 1$,

$$\|\theta_{\mathrm{ema},K+n} - \theta_\star\| \le \eta^n\|\theta_{\mathrm{ema},K} - \theta_\star\| + (1-\eta^n)\delta. \tag{13}$$

In particular, if $\theta_k \to \theta_\star$, then $\theta_{\mathrm{ema},k} \to \theta_\star$.

Proof. Repeated substitution of the EMA recurrence gives Eq. (12). Its coefficients are nonnegative and sum to one, so the teacher is a convex combination of the initialization and past student parameters. Expanding the recurrence from iteration K gives

$$\theta_{\mathrm{ema},K+n} - \theta_\star = \eta^n(\theta_{\mathrm{ema},K} - \theta_\star) + (1-\eta)\sum_{j=1}^{n}\eta^{n-j}(\theta_{K+j} - \theta_\star).$$

Applying the triangle inequality and the assumed bound yields

$$\|\theta_{\mathrm{ema},K+n} - \theta_\star\| \le \eta^n\|\theta_{\mathrm{ema},K} - \theta_\star\| + (1-\eta)\delta\sum_{j=1}^{n}\eta^{n-j}.$$

Evaluating the geometric sum proves Eq. (13). If the student converges, the assumption holds eventually for every $\delta > 0$. Taking $n \to \infty$ and then $\delta \to 0$ establishes teacher convergence. □

Equation (12) shows that each historical student iterate enters the teacher with an exponentially decaying weight, while the initialization weight vanishes. Equation (13) further shows that sustained student changes are gradually incorporated into the reference. These are statements about parameter averaging; the teacher's predictions need not equal an average of historical student predictions.

C.3 Scope of the Guarantees

Together, the propositions support an adaptive local consistency interpretation of on-policy distillation. The KL penalty controls teacher–student prediction discrepancies at sampled denoising states, while EMA allows the reference to follow sustained student adaptation. These bounds do not establish monotonic reward improvement, correct residual reward-model errors, or guarantee global sequence-level support coverage. The consistency guarantee is conditional on achieving a small expected distillation loss; the penalized objective alone does not ensure a prescribed value of ε.

D ADDITIONAL DETAILS AND EXPERIMENTS

D.1 IMPLEMENTATION DETAILS OF PILOT-UTR

Algorithm 1 summarizes the two-stage training procedure of PILOT-UTR. The first stage discovers reward-hacking sequences and uses them to calibrate the reward model; the second stage fine-tunes the diffusion policy against the calibrated reward while distilling EMA-teacher predictions at student-visited intermediate states.

Datasets and splits. The fixed-length MRL benchmark contains approximately 300K randomized 50-nt 5′UTRs from the unmodified eGFP reporter library with measured MRL labels. The variable-length MRL benchmark contains 83,919 random-source and 15,555 human-source sequences spanning 25–100 nt; we use the human-source sequences to pretrain the diffusion policy and train the guidance reward model, and the random-source sequences to train the independent evaluation model. The variable-length PC3 TE benchmark contains 9,875 unique endogenous human 5′UTRs from the PC3 cell line spanning 25–100 nt. We use approximately 9:1 training–validation splits for diffusion pretraining and reward-model training.

Algorithm 1 PILOT-UTR training

Input: labeled data $\mathcal{D}_{\mathrm{label}}$, pretrained diffusion policy $P^{\theta_{\mathrm{pre}}}$, cap threshold σ_{cap}, cap weight λ, adversarial KL strength α_{adv}, distillation strength β, and EMA decay η.
Train the vanilla reward model r_ϕ on $\mathcal{D}_{\mathrm{label}}$ by mean-squared regression.
Initialize the adversarial policy $\theta_{\mathrm{adv}} \leftarrow \theta_{\mathrm{pre}}$.
for each adversarial-policy update **do**
    Sample relaxed reverse trajectories $\bar{x}_{T:0} \sim P^{\theta_{\mathrm{adv}}}$ using Gumbel–Softmax.
    Update θ_{adv} by stochastic gradient ascent on the relaxed DRAKES surrogate in Eq. 4, using r_ϕ and KL strength α_{adv}.
end for
Sample the adversarial policy and construct $\mathcal{D}_{\mathrm{ood}} \leftarrow \{x_0 : x_0 \sim P^{\theta_{\mathrm{adv}}}\}$.
Initialize the calibrated reward model $\bar{r}_\phi$ from r_ϕ.
for each reward-model calibration update **do**
    Sample $(x, y) \sim \mathcal{D}_{\mathrm{label}}$ and $x_{\mathrm{OOD}} \sim \mathcal{D}_{\mathrm{ood}}$.
    Update $\bar{r}_\phi$ by minimizing the regression-plus-cap loss in Eq. 5.
end for
Initialize student and teacher parameters $\theta \leftarrow \theta_{\mathrm{pre}}$ and $\theta_{\mathrm{ema}} \leftarrow \theta_{\mathrm{pre}}$.
for each policy-training epoch **do**
    Sample an on-policy relaxed trajectory $\bar{x}_{T:0} \sim P^{\theta}$ and collect its intermediate states (t, x_t).
    Compute $\mathcal{L}_{\mathrm{OTD}}$ from stopped-gradient teacher predictions and student predictions at the visited states using Eq. 7.
    Update θ by minimizing $-\bar{r}_\phi(\bar{x}_0) + \beta\mathcal{L}_{\mathrm{OTD}}$ in Eq. 8.
    At the end of the epoch, update $\theta_{\mathrm{ema}} \leftarrow \eta\theta_{\mathrm{ema}} + (1-\eta)\theta$.
end for
Output: calibrated reward model $\bar{r}_\phi$ and fine-tuned diffusion policy P^{θ}.

Reward model training. All MRL guidance models use a single-task Enformer with three Transformer blocks and a scalar regression head. They are trained by mean-squared error with Adam, a learning rate of 1×10^{-4}, batch size 512, and checkpoint selection by validation MSE. The 50-nt model is trained for 15 epochs. For the variable-length benchmark, sequences are right-padded to 100 nt; the human-source guidance model and random-source evaluator are trained for 100 and 20 epochs, respectively. We use the released UTR-LM MRL predictor as the independent evaluator for the 50-nt benchmark. For the PC3 TE benchmark, MTtrans is trained as the guidance model on the RP-PC3 data with inputs left-padded to length 105, for 300 epochs with batch size 64 and learning rate 3×10^{-4}. The independent TE evaluator contains three compact convolution–Transformer models trained on separate 80:10:10 random splits. Each member has a 192-channel convolutional stem, three residual convolutional blocks, four four-head Transformer blocks, global average pooling, and a scalar regression head. Members are trained with mean-squared error and AdamW (learning rate 3×10^{-4}, weight decay 10^{-3}, batch size 128) for up to 100 epochs with validation-based early stopping; their predictions are averaged at evaluation time.

OOD sequence generation and reward calibration. To mitigate reward hacking, we generate out-of-distribution sequences using a reward-hacking policy with $\alpha = 0.001$ for both MRL tasks (50-nt and human 25–100-nt) and $\alpha = 0.0001$ for the PC3 TE task. The OOD pool size is set to 12% of the corresponding labeled training-set size. We calibrate the reward model using the cap loss with strength $\lambda = 3$ for both MRL tasks and $\lambda = 0.2$ for PC3 TE. The cap threshold is $\sigma_{\text{cap}} = 6.5$ for 50-nt MRL, $\sigma_{\text{cap}} = 5.5$ for human 25–100-nt MRL, and $\sigma_{\text{cap}} = 0.2$ for PC3 TE.

Table 5 summarizes reward-model validation performance across the three dataset–task settings using MSE and Pearson correlation.

Table 5: Validation performance of the reward models across three dataset–task settings. Each model is evaluated on its associated validation split; MSE values are not comparable across tasks because the prediction targets have different scales.

Dataset/task	**Model**	**MSE↓**	**Pearson r↑**
50-nt MRL	Vanilla guidance	4.609	0.765
	Re-calibrated guidance	2.057	0.747
	Eval judge	0.753	0.868
Human 25–100-nt MRL	Vanilla guidance	1.109	0.587
	Re-calibrated guidance	1.197	0.550
	Eval judge	1.091	0.694
PC3 TE	Vanilla guidance	0.490	0.716
	Re-calibrated guidance	0.658	0.612
	Eval judge	0.548	0.657

PILOT-UTR fine-tuning. We fine-tune the discrete-diffusion policy against the calibrated reward using $\beta = 0.003$ for the 50-nt MRL task and $\beta = 0.005$ for the human 25–100-nt MRL task, with a learning rate of 1×10^{-4} for both. For the variable-length MRL task, target lengths are sampled from the empirical length distribution of the top-5% highest-MRL training sequences. For PC3 TE optimization, we use $\beta = 0.0008$ and a learning rate of 2×10^{-5}; target lengths are sampled from the empirical training-set length distribution, and MTtrans inputs are left-padded to length 105. Across all tasks, the EMA-teacher decay is set to $\eta = 0.9995$, the Gumbel–Softmax temperature is set to 1, the gradient-truncation window is set to 50 reverse-diffusion steps, and fine-tuning is run for at most 500 epochs.

All experiments were run on NVIDIA RTX PRO 6000 GPUs.

D.2 Baseline implementation details

We use the same task-specific sequence lengths, guidance reward models, and evaluation protocol as for PILOT-UTR. Unless stated otherwise, all diffusion-based methods use the pretrained masked discrete-diffusion model associated with the corresponding dataset.

DRAKES trajectory objective. For a reverse trajectory $x_{T:0} = (x_T, \ldots, x_0) \sim P^\theta$, DRAKES writes the reward-regularized objective as

$$\theta^\star = \arg\max_{\theta \in \Theta} \mathbb{E}_{x_{T:0} \sim P^\theta}[r_\phi(x_0)] - \alpha\, \mathbb{E}_{x_{T:0} \sim P^\theta} \left[\int_{t=0}^{T} \sum_{i=1}^{L} \sum_{\substack{v \in \bar{\mathcal{V}} \\ v \neq x_t^{\langle i \rangle}}} \left\{ Q^{\theta_{\text{pre}}}_{x_t^{\langle i \rangle}, v}(t) - Q^{\theta}_{x_t^{\langle i \rangle}, v}(t) + Q^{\theta}_{x_t^{\langle i \rangle}, v}(t) \log \frac{Q^{\theta}_{x_t^{\langle i \rangle}, v}(t)}{Q^{\theta_{\text{pre}}}_{x_t^{\langle i \rangle}, v}(t)} \right\} dt \right], \tag{14}$$

where Q^θ and $Q^{\theta_{\text{pre}}}$ are the fine-tuned and pretrained reverse-process generators, respectively.

Pretrained. This reference samples directly from the pretrained diffusion policy without reward guidance or policy fine-tuning.

Classifier Guidance (CG). CG steers each reverse-diffusion step using the gradient of the task reward evaluated on a posterior-mean estimate of the clean sequence. We use a generation batch size of 64 and a guidance scale of 1000 for all three tasks.

Classifier-Free Guidance (CFG). CFG trains a conditional diffusion model using binary labels obtained by assigning the top 5% of training sequences to the high-score class. At inference, we sample from this class (class label 1) with classifier-free guidance weight 2.0.

Sequential Monte Carlo (SMC). SMC uses the pretrained reverse process as its proposal and sequentially reweights and resamples particles using reward-derived soft values. We use 64 particles and set the reward-tempering coefficient to $\alpha = 0.5$ for all three tasks.

Twisted Diffusion Sampler (TDS). TDS uses the same sequential reweighting and resampling procedure as SMC but replaces the pretrained proposal with a classifier-guided proposal. We use 64 particles and $\alpha = 0.5$ for every task; the proposal-guidance scales are 1000, 10, and 1000 for 50-nt MRL, human 25–100-nt MRL, and PC3 TE, respectively. For the human 25–100-nt task, the scale is reduced to 10 to prevent numerical overflow caused by the substantially larger reward-model gradients.

UTRGAN. UTRGAN uses a convolutional Wasserstein GAN with gradient penalty and optimizes its latent input by gradient ascent through the task reward model while keeping the generator fixed. For each task, we optimize for 10,000 steps with learning rate 10^{-3}, latent dimension 40, latent-noise standard deviation 10^{-4}, and batch size 64.

SEPO. SEPO fine-tunes the diffusion policy using group-normalized terminal-reward advantages and a clipped score-entropy policy ratio. Here, G is the number of outputs used to normalize each reward group, K is the number of policy-update epochs per rollout batch, and M is the number of self-normalized importance samples used to estimate each neighboring-state marginal. For the 50-nt and human 25–100-nt MRL tasks, we use batch size 16, $G = 8$, $K = 5$, $M = 2$, and gradient truncation at 50 steps, with KL coefficients 0.003 and 0.005, respectively. For PC3 TE, we use batch size 8, $G = 8$, $K = 2$, $M = 4$, truncation at 10 steps, and KL coefficient 0.0008. These KL coefficients match the corresponding PILOT-UTR distillation strengths. Across all tasks, we use learning rate 10^{-4}, clipping ratio $\epsilon = 0.2$, 128 reverse-diffusion sampling steps, 500 epochs with 10 optimization steps per epoch, and gradient clipping at 1.0.

Diversity comparison. We measure diversity as the mean nearest-neighbor normalized Levenshtein distance among generated sequences. Table 6 compares PILOT-UTR with the particle-based SMC and TDS baselines. SMC and TDS collapse to nearly identical samples on both MRL benchmarks, whereas PILOT-UTR preserves substantially greater population diversity.

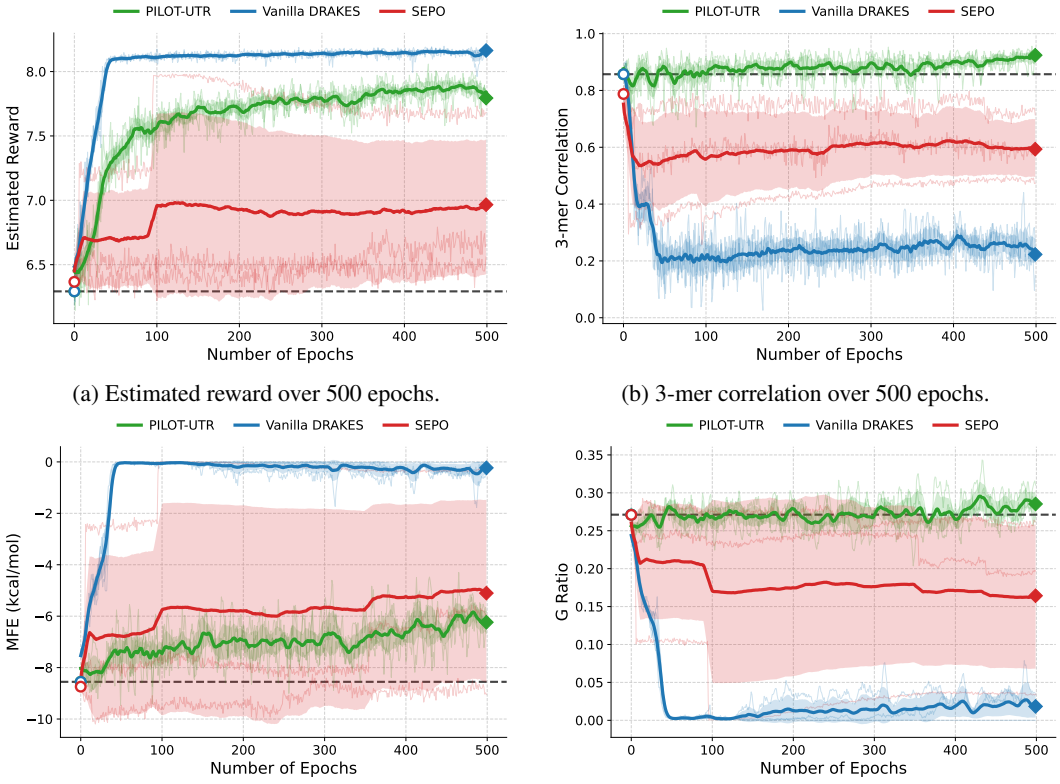
Table 6: Sequence diversity of PILOT-UTR and the particle-based baselines on the two MRL benchmarks. Entries report mean ± standard deviation over three seeds.

Method	50-nt MRL	Human 25–100-nt MRL
SMC	$0.005_{\pm.000}$	$0.005_{\pm.001}$
TDS	$0.005_{\pm.001}$	$0.005_{\pm.001}$
PILOT-UTR	$\mathbf{0.367_{\pm.002}}$	$\mathbf{0.204_{\pm.003}}$

D.3 Comparison of fine-tuning dynamics

We compare the optimization trajectories of PILOT-UTR with vanilla DRAKES and SEPO on the 50-nt MRL task. Figure 5 shows that vanilla DRAKES increases its proxy reward most aggressively, but its 3-mer correlation rapidly collapses, its MFE approaches zero, and its generated G-nucleotide ratio vanishes within the first 100 epochs, revealing severe reward hacking. SEPO avoids this complete collapse but still exhibits substantial drift in sequence composition and structural stability together with considerably larger variation across seeds. In contrast, PILOT-UTR steadily improves its reward while maintaining high 3-mer correlation, an MFE substantially closer to the initial reference, and a stable G-nucleotide composition. PILOT-UTR therefore provides the strongest optimization stability and reward–naturalness trade-off among the three fine-tuning approaches. Figure 6 provides representative endpoint structures that qualitatively illustrate this distinction. Because SEPO

logs reward using a different evaluator and the methods have different per-epoch sampling budgets, the reward curves characterize within-method optimization dynamics rather than absolute reward or compute efficiency; shared-evaluator endpoint comparisons are reported in Table 1.



(a) Estimated reward over 500 epochs.

(b) 3-mer correlation over 500 epochs.

(c) MFE over 500 epochs.

(d) G-nucleotide ratio over 500 epochs.

Figure 5: **Fine-tuning dynamics of PILOT-UTR, vanilla DRAKES, and SEPO on 50-nt MRL optimization over 500 training epochs.** Thin curves show individual seeds, thick curves show the across-seed mean, and shaded regions indicate ± 1 standard deviation. Open circles and dashed horizontal lines mark the initial reference values; diamonds mark the final epoch shown in each panel. PILOT-UTR improves reward while better preserving sequence naturalness, predicted structural stability, and nucleotide composition.

D.4 Reward hacking on the PC3 TE benchmark

We additionally evaluate vanilla DRAKES on the variable-length PC3 TE benchmark with $\alpha = 0.0003$ across three random seeds. Figure 7 shows that the guidance oracle's predicted TE rises throughout fine-tuning, whereas the independent evaluator remains nearly flat. At the same time, 3-mer correlation collapses, MFE approaches zero, and the G-nucleotide fraction sharply decreases. Reward hacking is therefore not specific to the MRL predictor or fixed-length sequences: direct optimization can similarly exploit a TE guidance model without producing gains that transfer to an independent evaluator.

D.5 Ensemble uncertainty fails under shared shortcuts

We test whether predictive disagreement can detect the OOD exploitation identified in Section 3. We construct an ensemble of five vanilla MRL reward models trained independently from different random initializations and fine-tune the diffusion policy against the lower-confidence-bound objective $\widetilde{r}_{\text{ens}}(x) = \mu_{\text{ens}}(x) - \lambda_u \sigma_{\text{ens}}(x)$, with $\lambda_u = 1$. We retain the standard reverse-KL penalty with $\alpha = 0.005$ and use no forward-KL term. We fine-tune each model for 100 epochs across three random seeds.

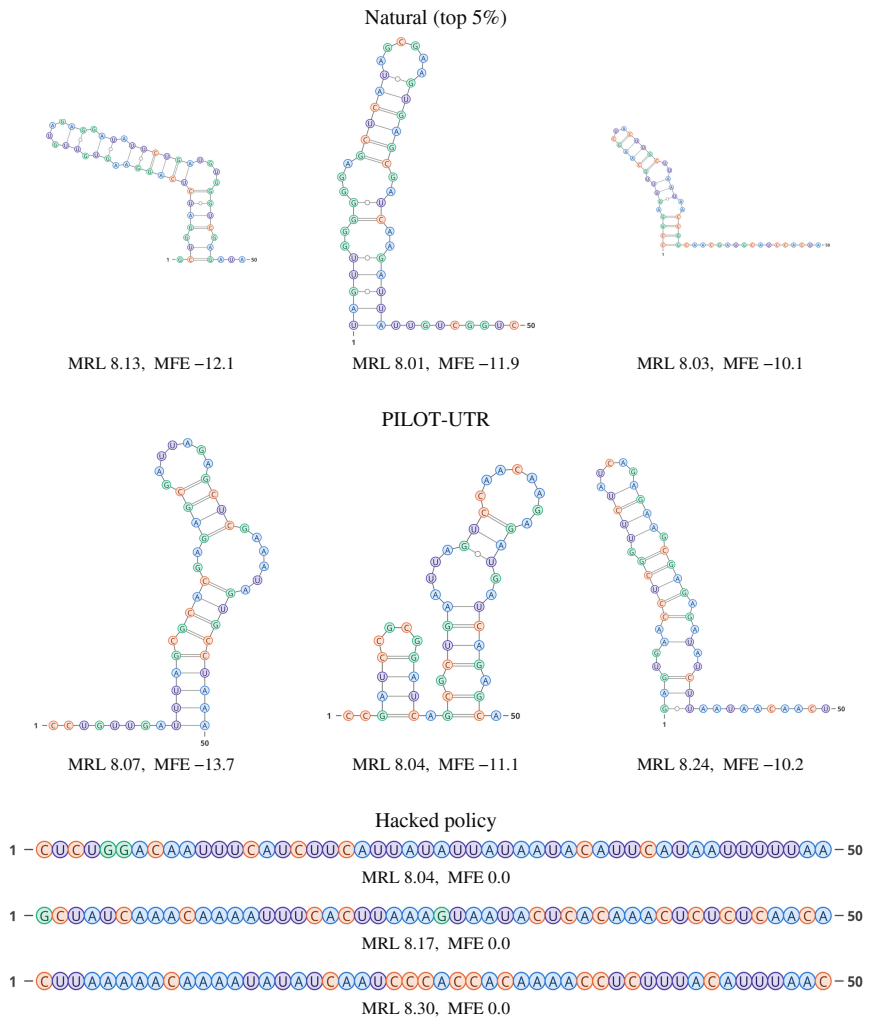


Figure 6: **Representative predicted secondary structures on the 50-nt MRL task.** Natural top-5% and PILOT-UTR sequences retain folded structures with negative MFE, whereas reward-hacked vanilla DRAKES samples are A/U-rich and essentially unstructured (MFE 0.0) despite receiving high predicted MRL.

Figure 8 shows that all five ensemble-member predictions increase together. Their pairwise trajectory correlations exceed 0.997 for every policy seed (mean correlation > 0.998), while ensemble disagreement decreases from 0.334 ± 0.030 at epoch 0 to 0.215 ± 0.001 at epoch 99. Agreement therefore becomes stronger, rather than weaker, as the policy leaves the natural sequence distribution. Over the same interval, 3-mer correlation falls from 0.857 ± 0.002 to 0.241 ± 0.060, the G-nucleotide fraction falls from 0.271 ± 0.008 to 0.009 ± 0.016, and mean MFE rises from -8.55 ± 0.05 to -0.11 ± 0.16. Values report mean $\pm$ standard deviation across the three policy seeds.

These trajectories expose a limitation of disagreement-based uncertainty: independently initialized predictors can share the same shortcut when they are trained on the same biased support with similar architectures and objectives. Such common-mode extrapolation raises μ_{ens} without raising σ_{ens}, so subtracting disagreement cannot identify the failure. In this setting, ensemble agreement is therefore not evidence of reliable OOD reward prediction, motivating direct calibration on policy-discovered OOD sequences.

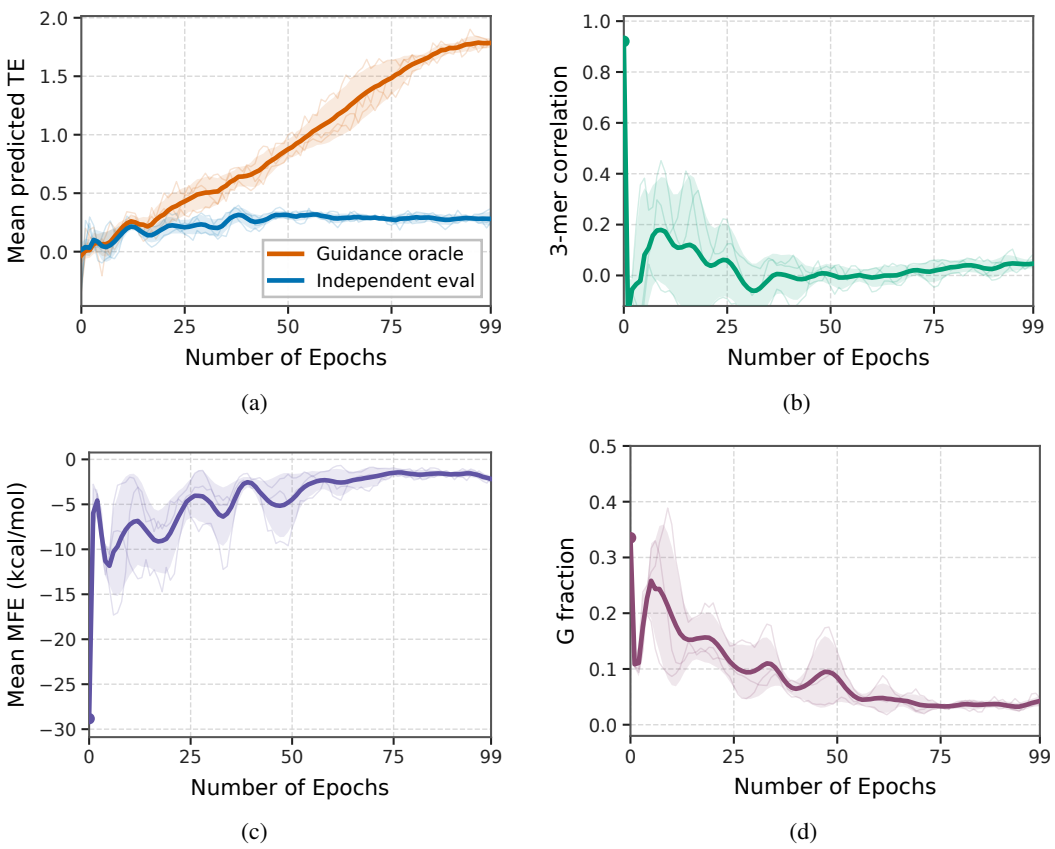


Figure 7: **Reward hacking during DRAKES fine-tuning on the PC3 TE benchmark at** $\alpha = 0.0003$**. (a)** The guidance oracle's predicted TE increases while the independent evaluation remains nearly unchanged; meanwhile, **(b)** 3-mer correlation, **(c)** predicted structural stability measured by MFE, and **(d)** the G-nucleotide fraction deteriorate. Thin curves show three random seeds; solid curves and shaded regions summarize their mean and variation, respectively.

D.6 Interpretability of reward-model recalibration

To examine how recalibration changes the features used by the MRL reward model, we compute token-level importance through attention rollout (Abnar & Zuidema, 2020) over the Transformer layers and compare the vanilla and recalibrated models on the same natural sequences. Figure 9 shows that recalibration decreases the attention contribution of A/U-only 3-mers, including UAA (-10.7%) and AAU (-10.4%), while increasing the contribution of C/G-only 3-mers, including CCG ($+9.6\%$), GCG ($+6.3\%$), and GCC ($+5.3\%$). This shift is consistent with suppressing the A/U-rich compositional shortcut exploited by the vanilla reward model and increasing reliance on C/G-containing sequence features.

D.7 Reward distributions across generated sequences

Figure 10 complements the aggregate results in Tables 1 and 2 by showing the full distribution of independently evaluated functional scores on the variable-length MRL and PC3 TE benchmarks; the corresponding 50-nt distribution is shown in Figure 4.

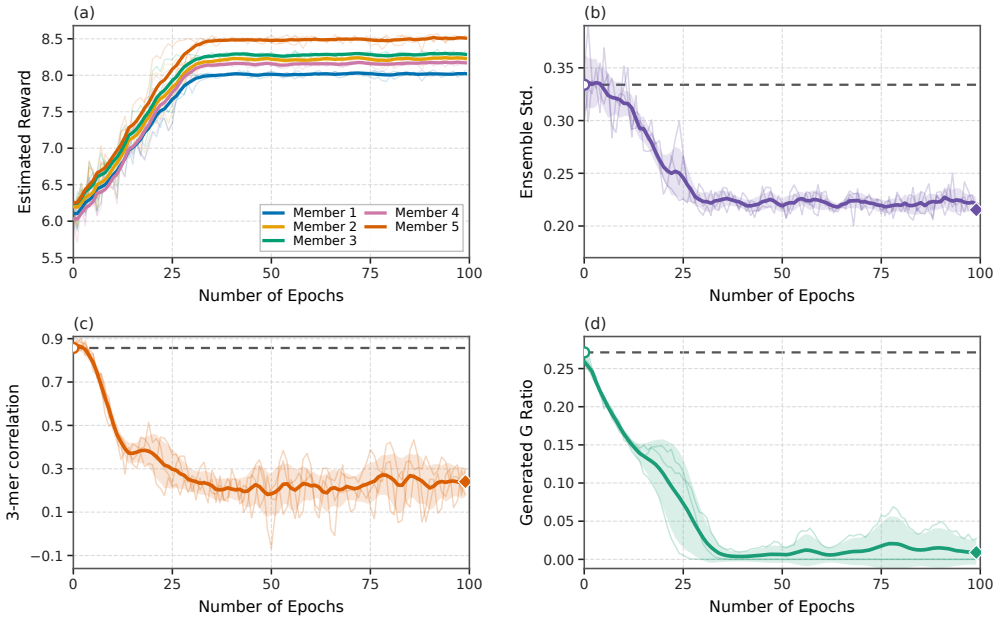


Figure 8: **A five-member reward ensemble does not prevent shortcut exploitation.** **(a)** Predictions from all five independently initialized members rise together during fine-tuning. **(b)** Ensemble disagreement, measured by the standard deviation across member predictions, does not increase as the policy moves OOD. Meanwhile, **(c)** 3-mer correlation with natural high-MRL UTRs and **(d)** the G-nucleotide fraction collapse. In panel (a), thin curves are individual policy seeds and thick curves are the corresponding five-epoch rolling means for each ensemble member. In panels (b)–(d), thin curves show three policy seeds; thick curves and shaded regions denote the five-epoch rolling mean and ± 1 standard deviation, respectively. Dashed lines and open circles mark the epoch-0 pretrained-policy baseline, and diamonds mark epoch 99.

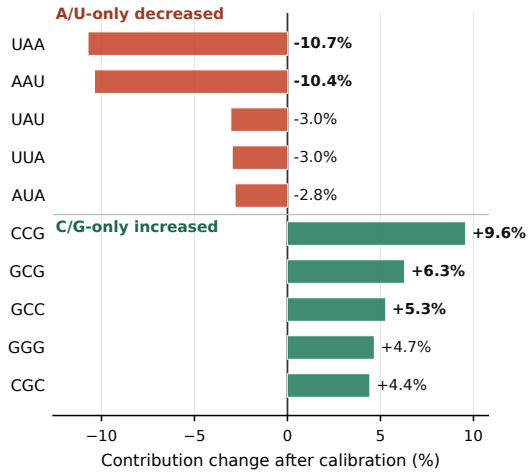
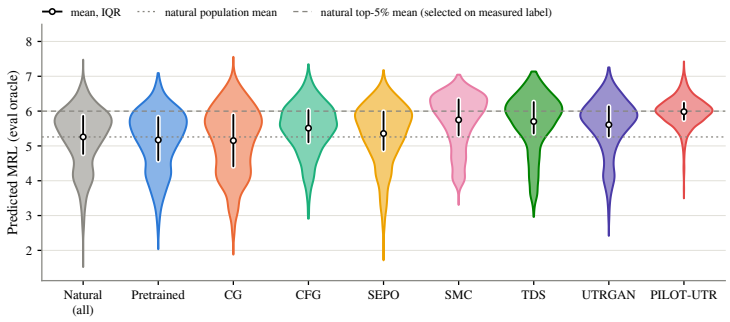
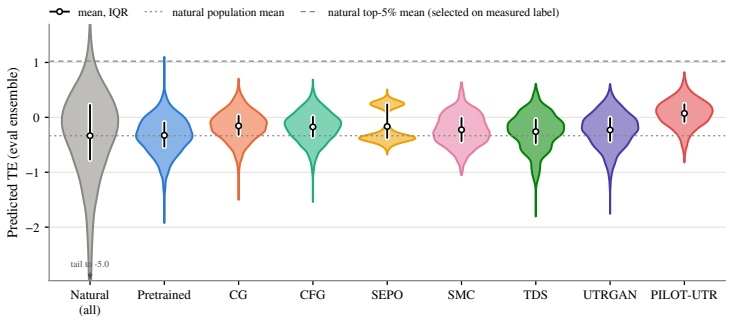


Figure 9: **Change in token-level attention contribution after reward-model recalibration.** Bars show the relative contribution change for the five most-decreased A/U-only 3-mers and the five most-increased C/G-only 3-mers, computed by attention rollout on the MRL reward model.



(a) Variable-length human 25–100-nt MRL benchmark.



(b) Variable-length PC3 TE benchmark (25–100 nt).

Figure 10: **Distributions of independently evaluated MRL and TE on the variable-length benchmarks.** Violin width represents density, circles denote means, and white vertical bars denote interquartile ranges. Dotted and dashed horizontal lines mark the natural-population mean and the natural top-5% mean, respectively.

D.8 HYPERPARAMETER SENSITIVITY ANALYSIS

We evaluate the sensitivity of PILOT-UTR on the 50-nt MRL benchmark by varying one hyperparameter at a time while holding the remaining settings fixed. The default configuration uses cap threshold 6.5, distillation strength $\beta = 0.003$, an OOD negative-pool to regression-dataset size ratio of 12%, and EMA decay 0.9995, consistent with the main experiments. Table 7 reports the mean and standard deviation over three fine-tuning seeds, with 1,000 generated sequences evaluated per seed. The distillation-strength sweep exhibits the clearest reward–fidelity trade-off, with stronger distillation reducing reward while shifting MFE toward the natural reference. The teacher-update ablation shows that both EMA settings reduce across-seed variability relative to a frozen teacher, while decay 0.9995 provides the strongest mean naturalness and structural-stability trade-off. The 30% OOD-pool setting exhibits greater variability in sequence naturalness.

Table 7: Hyperparameter sensitivity of PILOT-UTR on the 50-nt MRL benchmark. The default configuration uses EMA decay 0.9995, consistent with the main results. Entries are mean $\pm$ standard deviation over three fine-tuning seeds. Bold: best; underline: second best within each hyperparameter sweep.

Hyperparameter	**Setting**	**Mean reward↑**	**3-mer Corr.↑**	**MFE↓**
Cap threshold σ_{cap}	6.0	$\underline{7.897 \pm 0.016}$	$\underline{0.895 \pm 0.014}$	$\underline{-6.153 \pm 0.329}$
	6.5 (default)	$\mathbf{7.910 \pm 0.051}$	$\mathbf{0.925 \pm 0.007}$	-6.130 ± 0.440
	7.0	7.847 ± 0.054	0.838 ± 0.004	$\mathbf{-7.005 \pm 0.706}$
Distillation strength β	0.001	$\mathbf{8.297 \pm 0.039}$	0.841 ± 0.013	-3.102 ± 0.784
	0.003 (default)	$\underline{7.910 \pm 0.051}$	$\mathbf{0.925 \pm 0.007}$	$\underline{-6.130 \pm 0.440}$
	0.006	7.725 ± 0.017	$\underline{0.900 \pm 0.012}$	$\mathbf{-7.272 \pm 0.294}$
OOD-to-regression data ratio	5%	$\mathbf{8.018 \pm 0.012}$	$\underline{0.924 \pm 0.022}$	-4.337 ± 0.302
	12% (default)	7.910 ± 0.051	$\mathbf{0.925 \pm 0.007}$	$\mathbf{-6.130 \pm 0.440}$
	30%	$\underline{8.015 \pm 0.014}$	0.792 ± 0.129	$\underline{-4.856 \pm 0.889}$
EMA decay η	0.995	$\mathbf{8.027 \pm 0.016}$	0.901 ± 0.010	-4.843 ± 0.301
	0.9995 (default)	7.910 ± 0.051	$\mathbf{0.925 \pm 0.007}$	$\mathbf{-6.130 \pm 0.440}$
	Frozen teacher	$\underline{7.930 \pm 0.055}$	$\underline{0.911 \pm 0.025}$	$\underline{-5.574 \pm 1.535}$