Emergent Biological Realism in RL-Trained DNA Language Models

Anonymous Authors[1]

Abstract

Reinforcement learning has driven the mass adoption of large language models by unlocking often unexpected emergent capabilities, yet this approach remains largely underexplored for generative DNA models. We investigate whether similar post-training techniques can induce emergent biological realism in DNA language models, using plasmid generation as a testbed due to plasmids' relative simplicity, well-characterized functional constraints, and ubiquity in biotechnology. Using Group Relative Policy Optimization with a reward function based on constraints from engineered biology, our model achieves 97% quality control pass rate compared to 6% for the pretrained baseline. Remarkably, beyond explicitly optimized features, the model exhibits emergent biological parallels: generated sequences match natural plasmids in thermodynamic stability, codon usage patterns, and ORF length distributions, properties not directly encoded in the reward function. These results suggest that RL post-training can steer DNA language models toward biologically coherent regions of sequence space, analogous to how such techniques unlock emergent capabilities in natural language models.

1. Introduction

Plasmids are extrachromosomal DNA sequences, often found in bacteria, capable of replication independent of a host genome (Lederberg, 1952). These genetic elements are ubiquitous in biotechnology, serving as the primary vectors for protein expression, gene editing, and emerging DNA therapeutics (Prather et al., 2003; Kutzler & Weiner, 2008). Despite their widespread utility, plasmid engineering remains a complex, high-dimensional optimization problem. Traditional workflows are cost intensive and heuristic

driven, often requiring iterative cycles of manual sequence editing and experimental validation to resolve structural instabilities (Oliveira et al., 2009; Meng & Ellis, 2015). Suboptimal plasmid architectures, plagued by incompatible regulatory elements or unstable repeat regions, frequently lead to metabolic burden, reduced expression efficiency, and manufacturing bottlenecks (Brophy & Voigt, 2011; Wu et al., 2016).

Current approaches to plasmid design rely heavily on tacit domain knowledge and piecemeal assembly of genetic parts. Designers must simultaneously optimize for competing objectives such as copy number, transcriptional output, and host viability while navigating the strict biophysical constraints of DNA folding and context dependent regulatory interactions (Deng et al., 2025; Fung et al., 2025).

The dramatic success of reinforcement learning post-training in natural language processing, such as improved reasoning, instruction following, and unexpected generalization highlights the potential of RL post training in DNA language models (Wei et al., 2022; Berti et al., 2025). We investigate this using plasmid generation as a testbed, applying Group Relative Policy Optimization to the PlasmidGPT foundation model (Shao et al., 2024a). Beyond dramatically improving quality control pass rates (97% vs. 6% baseline), we observe emergent properties not explicitly optimized: generated sequences match natural plasmids in thermodynamic stability, codon usage patterns, and ORF length distributions. These results suggest that RL post-training may unlock similar emergent capabilities in genomic models as it has in language models, steering generation toward biologically coherent sequence space regions through reward-guided optimization.

2. Background

2.1. DNA Language Models

Natural language processing and genomics have developed in parallel, often with algorithms developed for bioinformatics used in NLP and vice versa (Durbin et al., 1998). Natural language models pretrained on massive amounts of data have displayed emergent capabilities and remarkable utility by understanding the structure of language as a whole (Wei et al., 2022). These capabilities have vastly

increased the utility of language models in a wide range of tasks, and are one of the primary reasons why large language model adoption has skyrocketed in the last few years. This has inspired the application of many similar techniques to genomic language models, resulting in models capable of impressive performance on biological tasks including variant prediction (Ji et al., 2021), transcription factor binding site identification (Nguyen et al., 2023), and even whole-genome generation (Nguyen et al., 2024).

Plasmid DNA has received relatively little attention compared to other sequence types despite its importance in biomanufacturing and research. OriGen (Martinson et al., 2025) introduces a generative model to produce previously undiscovered origins of replication (ORIs) but does not model whole sequences. PlasmidGPT (Shao et al., 2024a; Cunningham et al., 2025) uses modern language modeling techniques to develop a generative model for whole plasmid sequences, and later work expands on this by synthesizing whole plasmids generated using a fine tuned version of the PlasmidGPT model (Cunningham et al., 2025). We use this as a base model and apply post-training techniques to improve results significantly.

2.2. Plasmid Design

Lab-designed plasmids are short circular DNA molecules (typically 2-15 kb) that must contain multiple functional components arranged in precise configurations. At minimum, a viable plasmid requires: (i) an origin of replication to enable autonomous replication, (ii) a selection marker (e.g., antibiotic resistance gene) for identifying successfully transformed cells, and (iii) a cloning site where genes of interest can be inserted.

The search space of valid plasmids is massive due to combinatorial explosion across components (Naseri & Koffas, 2020). There are many types of each component on the scale of hundreds to thousands per component, and additional regulatory elements (promoters, terminators, enhancers) and reporters may be required depending on the application. Multiple instances of certain components may be necessary, and the ordering and spacing of these elements significantly impacts function. Beyond sheer combinatorics, designers must navigate complex biological constraints including compatibility requirements (specific ORIs only function in certain hosts), physical stability issues (repeat regions can fold and bind to each other), and other design challenges (Meng & Ellis, 2015; Brophy & Voigt, 2011).

3. Methods

We follow an established method that starts with a base model, fine-tunes it using supervised fine-tuning (SFT), and then applies reinforcement learning (RL) to optimize for specific attributes in the output (Figure 1).

3.1. Supervised Fine-Tuning

Supervised fine-tuning (SFT) was performed on a curated corpus of *E. coli* plasmid sequences assembled from PlasmidScope and Addgene (Addgene, 2024; Cunningham et al., 2025). After deduplication and quality filtering, approximately 15k circular plasmids (≤ 30 kb) were retained, excluding linear entries, fragments, and incomplete records. Sequences were tokenized using the original PlasmidGPT byte-pair DNA tokenizer. The pretrained PlasmidGPT model was fine-tuned using an autoregressive next-token prediction objective with gradient accumulation and learning-rate warmup over three epochs.

3.2. Reinforcement Learning with GRPO

We implement a configurable reinforcement learning pipeline for plasmid design that uses Group Relative Policy Optimization (GRPO) (Shao et al., 2024b) with a domain-specific reward function described below. At each training iteration, the model generates a batch of candidate plasmids via autoregressive rollouts conditioned on short nucleotide prompts. Prompts are either stochastic (4–25 bp random seeds, excluding "ATG", used to promote rollout diversity) or structured (partial "cassette" seeds encoding canonical marker genes such as antibiotic-resistance or fluorescent reporters). Each candidate sequence is evaluated via our reward function, which captures structural plausibility, cassette organization, repeat content, and other biologically motivated constraints. GRPO is then applied to update the model parameters using these sequence-level rewards, enabling the policy to progressively shift toward generating plasmids with higher predicted validity.

3.3. Reward Function Design

The reward function scores each generated plasmid according to its structural plausibility and expected stability. It is composed of three conceptual components:

Functional annotation scoring: Lightweight annotations identify origins of replication, promoters, terminators, coding sequences (CDS), and selectable markers, which are then scored according to a configuration designed with subject matter expert (SME) input to reflect biologically reasonable quantities (e.g., exactly one origin of replication and at least one selectable marker). CDS regions are identified using Pyrodigal (Hyatt et al., 2010), a gene prediction tool that detects coding sequences based on statistical patterns rather than homology search. To encourage coherent gene cassettes, this component also includes a location-aware bonus for promoter $\rightarrow$ CDS $\rightarrow$ terminator arrangements that appear in the correct order and within a reasonable
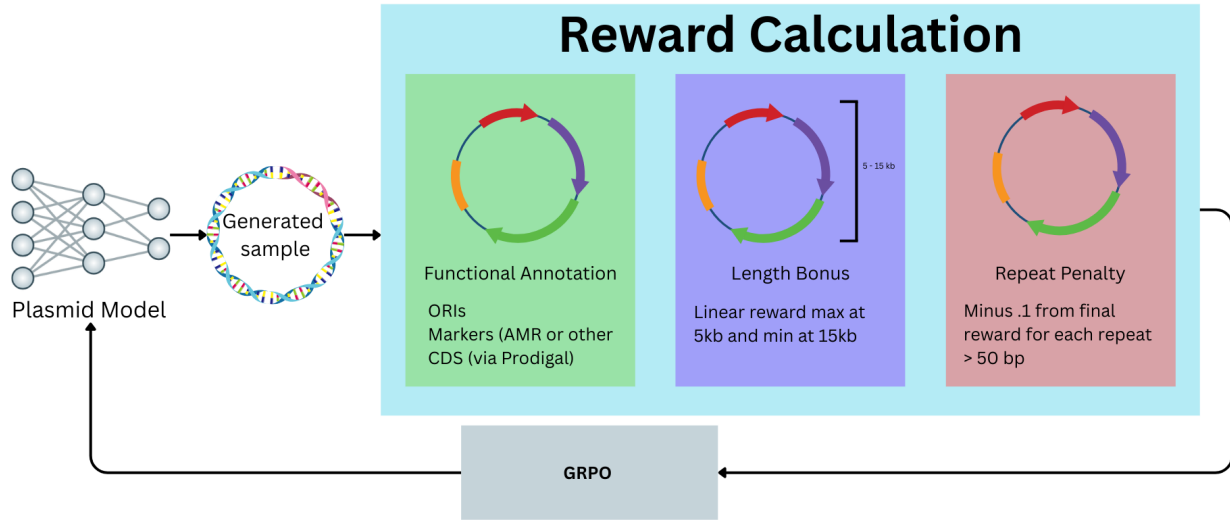
Figure 1. Overview of the Plasmid-RL training pipeline. Starting from the pretrained PlasmidGPT base model, we apply supervised fine-tuning on curated plasmid sequences, followed by reinforcement learning with Group Relative Policy Optimization using a biologically-motivated reward function that evaluates functional annotations, length constraints, and repeat content.

proximity window.

Length prior: A length prior favors plasmid sizes within typical experimental ranges (5–15 kb) preferred for plasmid construction by giving a linearly increasing reward for lengths between 5 and 15 kb with a maximum reward at 5 kb and no reward for sequences longer than 15 kb.

Repeat penalty: A repeat penalty down-weights sequences containing long exact repeats that are associated with instability or recombination, specifically penalizing .1 reward for each repeat of length 50 bp or greater.

These terms are combined into a single scalar in $[0, 1]$, yielding a fast and interpretable proxy for "plasmid-likeness" and validity during reinforcement learning.

4. Experiments

4.1. Plasmid Quality Control and Uniqueness

We evaluate three model variants on held-out prompts not seen during training. For each model, we sampled 50 rollouts with two prompts: (i) a minimal prompt (single ATG codon) to test unconditional generation capability, and (ii) a structured prompt containing a complete GFP expression cassette to test the model's ability to build around provided components. These prompts were deliberately excluded from the training corpus to ensure evaluation reflects generalization rather than memorization.

4.1.1. VALIDITY ASSESSMENT

In silico plasmid validity was assessed using a bioinformatics quality-control pipeline that leverages BLAST based tools, requiring exactly one origin of replication ($\geq$95% identity and coverage), one or two antimicrobial resistance genes ($\geq$99% identity and coverage), and no internal repeats longer than 50 bp (Altschul et al., 1990). This pipeline has been validated as a reliable proxy for experimental synthesis success (Cunningham et al., 2025). While we do not perform wet-lab validation in this work, our focus is on establishing that RL post-training can successfully navigate the plasmid design space in silico, laying groundwork for future conditional generation systems where user-specified designs can be experimentally validated.

4.1.2. UNIQUENESS ASSESSMENT

To assess whether generated plasmids represent genuinely new designs rather than minor variants of existing constructs,

3

Table 1. Novelty and diversity metrics across model variants. RL achieves dramatically higher quality (97% pass rate) with reduced diversity in samples.

MODEL	QC PASS RATE	% NOVEL	DIVERSITY
BASE	6%	95.5%	0.926
SFT	11%	100%	0.886
RL	97%	88%	0.391

we compute similarity to known sequences using the NCBI BLASTn API. Each generated plasmid is assigned to one of three categories based on identity and query-coverage thresholds: sequences with $\geq$99% identity and $\geq$95% coverage are classified as **Exists**; those with $\geq$95% identity and $\geq$80% coverage are **Similar**; and all others are classified as **Novel**.

This categorization is based on large scale data curation efforts such as PLSDB that use these thresholds of similarity to attempt to de-duplicate plasmids for not adding any additional value to a dataset (Galata et al., 2018).

4.1.3. DIVERSITY ASSESSMENT

To detect and prevent model collapse, we attempt to measure the diversity of the many samples of the model from the same prompt. Due to the lack of utility of traditional NLP metrics on this task, we use the mean Pairwise Jaccard distance of the 21-mers of each sequence. Diversity of a group of rollouts is calculated as follows:

$$D = 1 - \frac{1}{\binom{n}{2}} \sum_{i=1}^{n} \sum_{j=i+1}^{n} J(S_i, S_j)$$

where $J(S_i, S_j)$ is the Jaccard similarity between MinHash sketches of sequences i and j, and n is the number of sequences in the group.

The diversity metric (pairwise Jaccard distance) serves primarily as a model collapse detector rather than a biological validity measure. The RL model achieves 0.391 diversity compared to 0.926 for the base model, indicating that while the RL model concentrates probability mass on higher-quality regions of sequence space, it does not collapse to identical outputs. This is consistent with RL optimization finding and using conserved "successful motifs" (e.g., proven origins of replication, reliable resistance markers) while still maintaining sequence-level uniqueness.

4.1.4. RESULTS

Reinforcement learning substantially increases the probability of generating plasmids that pass our bioinformatics quality control (QC) pipeline, while supervised fine-tuning
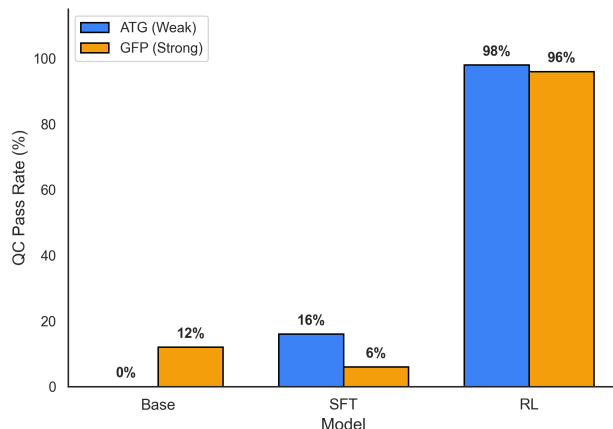

Figure 2. Summary of QC outcomes by prompt. Base model is only able to generate functional plasmids with a strong prompt. SFT allows the model to overcome this limitation on occasion but still often fails QC. Adding RL to the training process improves pass rate dramatically.

provides modest improvements. Figure 2 shows pass rates by prompt type, and Table 1 summarizes novelty and diversity metrics across models.

When prompted with the weak ATG prompt, the base model never produces a valid plasmid (0%), whereas SFT increases the pass rate to 16%, and RL further increases it to 96%. A similar trend holds with the stronger GFP-cassette prompt: the base model achieves 12%, SFT drops to 4%, but the RL-optimized model reaches 76%.

Aggregated across strong and weak prompts, the overall QC pass rate rises from 6% with the base model to 11% with the SFT model and finally to 97% with the RL model, representing more than an order-of-magnitude improvement in validity relative to the pretrained baseline.

Importantly, this increase does not come from the RL models repeating previously known, high scoring sequences. Among passing sequences, the proportion classified as novel remains substantial for all models (Figure 3). Using our novelty thresholds, the RL model produces 88% novel plasmids among its QC-passing samples, compared to 95.5% for the base model. The SFT model generated 100% novel sequences. When normalized over all 100 rollouts per model, the RL model produces 85.4% sequences that are both QC-valid and novel, compared to 11% for SFT and 6% for the base model.

Taken together, these results show that RL post-training not only dramatically improves biological plausibility but also preserves meaningful novelty relative to published plasmids. The model learns to satisfy constraints without collapsing to
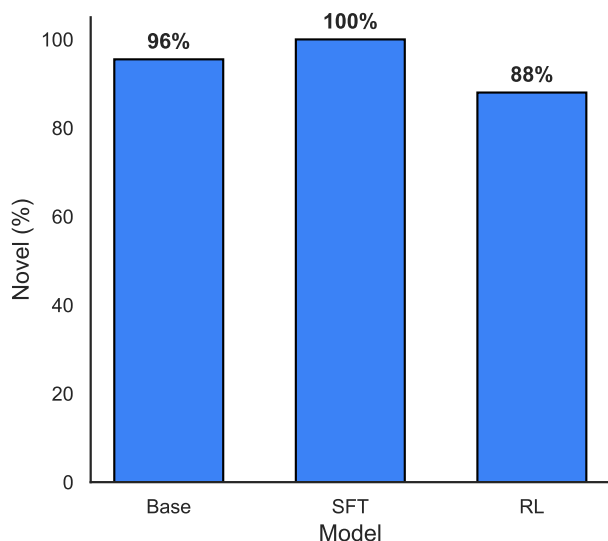
Figure 3. Summary of plasmid novelty as measured by comparison to NCBI database. SFT improves novelty of generated sequences from both base model and RL post-trained model.

MODEL	MEAN LOG-PROB	STD DEV
BASE	-12.449	6.144
RL	-11.148	2.977

memorized or trivial constructs, suggesting that sequence-level RL can push generation toward realistic design regions while still exploring new areas of plasmid space.

4.2. Distribution Comparison

We compute several statistics known to be relevant to the performance of DNA from the raw sequences of both the generated plasmids and a small subset of real plasmids used for protein expression, genome editing, and other applications. See full details of plasmids used in the appendix. We calculate the following summary statistics: sequence length distribution, GC content, longest open reading frame (ORF; calculated two ways), Jensen-Shannon divergence of the codon distribution, and Gibbs free energy (as calculated by ViennaRNA (Lorenz et al., 2011)).

Figure 4 shows that the RL post-trained model's samples much more closely match the distributions of the real plasmids than the pretrained and supervised models, even when the metric is not directly encoded by the reward function. Kolmogorov-Smirnov statistics comparing model-generated distributions to real plasmids demonstrate this quantitatively: for GC content, RL achieves a KS statistic of 0.51 (closest to real at 0.52) compared to base (0.48) and SFT (0.50); for codon usage measured by Jensen-Shannon divergence, RL achieves 0.06, identical to real plasmids and substantially better than base (0.10) and SFT (0.09). See Appendix for complete distribution comparison metrics.

Sequence length is directly encoded by the reward function,

with optimal reward determined by a parameter sweep for training stability. GC content is not directly encoded, but regions selected for by the reward function likely have GC content distributed similarly to real plasmids. Despite generally making up a smaller percentage of the whole sequence, GC content is partially encoded by the reward function, as regions with higher GC content are more likely to be rewarded.

ORF length and codon distribution are not factored into the reward function directly. We calculate ORF length two ways: (1) maximum length of codons that are not stop codons on a single strand, and (2) longest stretch of non-stop codons after the presence of a start codon on either strand. These methods are disjoint from the method used to account for ORF in the reward function, which uses Prodigal (Hyatt et al., 2010) to predict and reward correctly placed ORFs. The ORF length, measured by either method, converges closely to the distribution of the real plasmids.

The same pattern holds for codon distribution and Gibbs free energy. While not accounted for in the reward function, the RL model learns to generate tokens with a much more similar codon distribution to real plasmids than the two models that have seen the correct distribution in the training data. The similarity in the distribution of free energy measurements is particularly remarkable given that not only is free energy not optimized for directly, but no structural components are factored in at all excluding the weakly correlated repeat penalty originally included to solve recombination issues.

4.3. Held-Out Continuation

To evaluate how RL training affects nucleotide-level predictive performance, we measure how well each model can predict future bases given a real plasmid prefix. For each sequence, we provide the first 400 nucleotides as a prompt, have each model produce the next 100 nucleotides, and then compute the average log-probability of the true next 100 bases under each model. This allows us to compare the Base and RL models against real plasmids in terms of next-token prediction accuracy.

The RL model shows improved log-probability compared to the base model (Figure 5), with the standard deviation
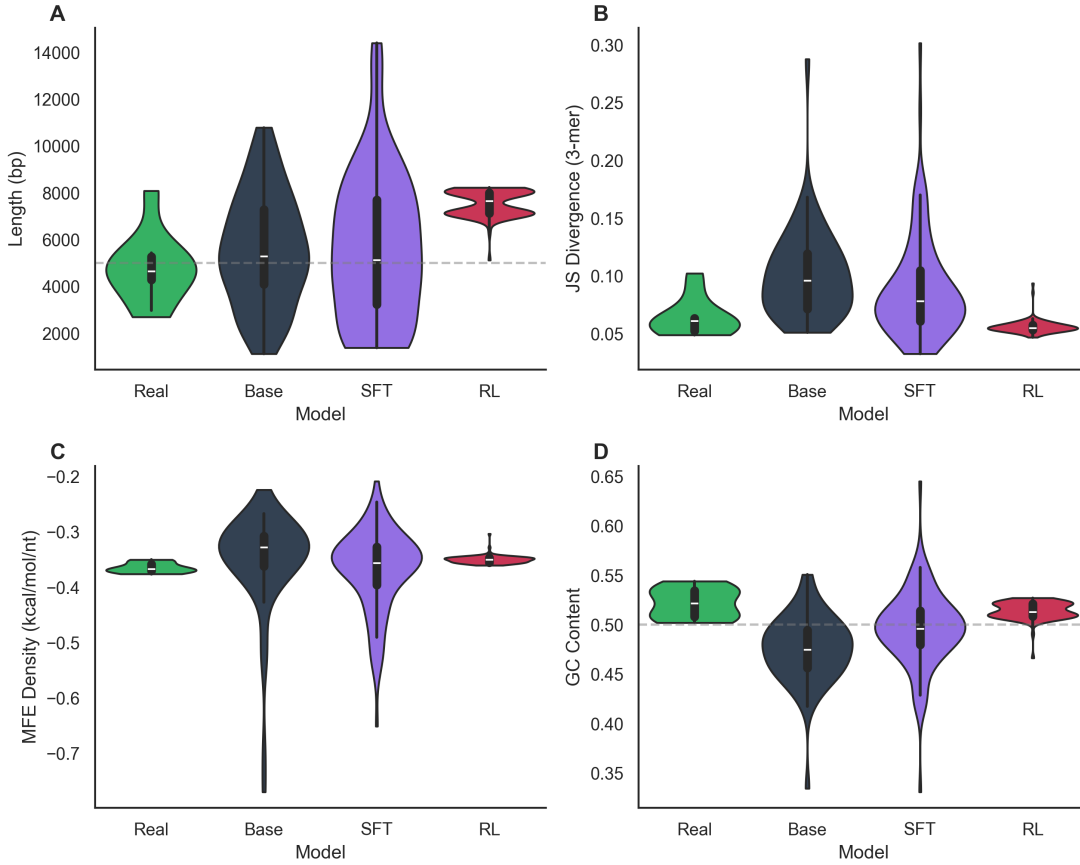
Figure 4. Distribution comparison across key biophysical metrics. The RL post-trained model (red) closely matches real plasmid distributions (green) across sequence length, GC content, ORF length, codon usage (Jensen-Shannon divergence), and thermodynamic stability (Gibbs free energy), while the base model (blue) and SFT model (purple) show substantial deviations. Notably, RL optimization produces realistic distributions even for metrics not explicitly encoded in the reward function.

shrinking substantially from 6.144 to 2.977. This improvement is unexpected, as RL generally makes language models worse at next-token prediction tasks, a phenomenon known as the "alignment tax" (Lin et al., 2023).

4.4. Coding Sequence Surprisal Analysis

The reward function evaluates CDS regions using Prodigal rather than bioinformatics token-level approaches, reducing the risk of token-level information leakage. Because Prodigal predicts coding regions based only on generic statistical patterns and not on specific plasmid sequences, the reward signal is biologically grounded but distribution-agnostic. Interestingly, the RL-trained model achieves lower surprisal on real plasmids than the pretrained model (Figure 6), suggesting that the Prodigal-based reward sharpens the model's understanding of natural plasmid structure.

5. Discussion

5.1. Summary: Emergent Biological Realism in RL-Trained Models

Our key finding is that RL post-training induces emergent biological realism in DNA language models. Beyond the dramatic improvement in quality control pass rates (97% vs. 6%), the RL model exhibits properties not explicitly encoded in the reward function: realistic thermodynamic stability (Gibbs free energy distributions matching natural plasmids), natural codon usage patterns (Jensen-Shannon divergence from real plasmids significantly lower than base model), and appropriate ORF length distributions. The emergence of biologically realistic properties, reflecting those of real plasmids, mirrors evolutionary processes that produce complex, correlated traits as byproducts of selecting for primary fitness criteria.

Furthermore, the RL model shows unexpected improvement on next-token prediction tasks, reversing the typical "align-
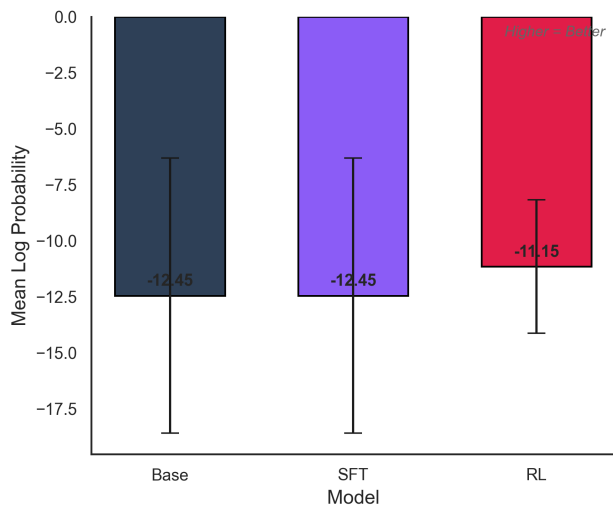
6

Figure 5. Held-out continuation performance across models. The RL model shows improved log-probability on real plasmid sequences, demonstrating better alignment with natural plasmid structure despite not being explicitly optimized for next-token prediction.

Figure 6. Coding sequence surprisal analysis. The RL model achieves lower surprisal on real plasmid coding sequences compared to the base and SFT models, indicating better capture of natural coding patterns despite using only Prodigal-based structural rewards.

ment tax" observed in language models. This suggests the model is not simply memorizing valid solutions but rather learning general principles of plasmid structure. The steering vector hypothesis offers a potential explanation: RL may be moving the model's latent space toward regions that are not only valid by our reward criteria but also broadly biologically coherent, explaining why unoptimized properties improve alongside explicitly rewarded features.

5.2. Implications for Computational Biology and Genomics

This work demonstrates that modern post-training techniques from natural language processing can transfer to DNA language models with similar success. Just as instruction tuning and RLHF unlocked the practical utility of large language models, RL post-training may represent a key step toward practical DNA generative models. The success of our approach suggests that appropriate reward function design, even without perfect biological accuracy (e.g., using lightweight bioinformatics rather than wet-lab validation), can guide models toward broadly realistic sequence spaces.

The parallel between emergent capabilities in LLMs and DNA models has broader implications: it suggests that genomic foundation models may benefit from similar training paradigms that have proven successful in other domains. This opens avenues for applying other techniques from the NLP toolkit (e.g., chain-of-thought prompting, mixture of experts, parameter-efficient fine-tuning) to genomic tasks.
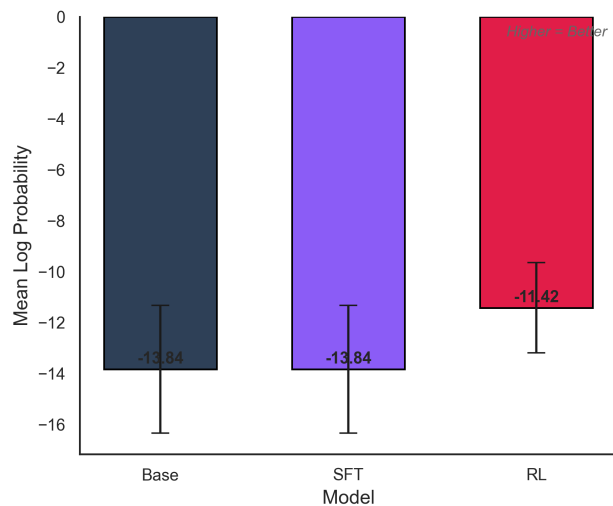
Moreover, the emergent properties we observe suggest that DNA language models may be capturing fundamental biological constraints implicit in natural sequence data, much as LLMs capture linguistic structure.

5.3. Limitations and Critiques

Bioinformatics-only evaluation: Our training and evaluation process relies almost entirely on bioinformatics, which depends on libraries of known regions. These libraries are naturally incomplete. Therefore, if our model generates a novel functional sequence (e.g., a never-before-seen ORI), it will not receive reward, sharply limiting how creative the model can be. This contrasts with other domains where RL has succeeded: natural language processing uses preference models that can evaluate novel outputs, and protein design uses biophysical simulators. Without wet-lab validation, we cannot definitively claim our sequences are experimentally viable, though our pipeline has been validated as a proxy for synthesis success.

Diversity trade-offs: The post-trained model shows decreased sequence diversity (0.391 vs. 0.926 for base model), a known effect of RL optimization. While the diversity metric primarily serves as a collapse detector rather than a biological validity measure, there is a meaningful reduction in functional diversity. For example, when sampling 500 sequences from each model, the base model uses 10 unique ORIs while the RL model uses only 7. The model converges toward conserved, reliable solutions at the expense

of exploring less common but potentially viable alternatives. This is consistent with RL finding "successful motifs" but may limit the model's utility for applications requiring high functional diversity.

Assumptions about reward function design: Our reward function design assumes that structural annotations (promoters, ORIs, etc.) and simple composition metrics can capture essential aspects of plasmid validity. While QC pass rates and wet lab validation of said QC process suggest this assumption is partially validated, we may be missing important biological constraints not easily encoded in lightweight bioinformatics. The simplification is pragmatic for efficient training but may become limiting as we scale to more complex genomic systems.

5.4. Future Work

This work further supports the ongoing hypothesis that modern language modeling techniques can be applied to DNA language models in a similar fashion to how they have been applied to natural language in the past. The real utility of natural language models came from instruction tuning them to respond to questions and follow directions. This work shows that post- training techniques can be applied to DNA language models to achieve similar results.

We hope to build out a dataset designed for conditional generation where the user can prompt the model with the specifics of the plasmid they want, and the model will develop it from there. This would enable practical use cases such as "design a plasmid for expressing protein X in E. coli with high copy number" or "create a mammalian expression vector with constitutive GFP expression."

In addition to being far more practical than unconditional generation, the conditioning will promote more diversity of samples and make evaluation of the model's performance on the task of plasmid design much more understandable.

6. Conclusion

We demonstrate that reinforcement learning can mimic distributions of real plasmids across several key features, even when not explicitly optimized for these features. Our model generates structurally valid plasmids at a 97% rate (compared to 6% for the base model), but more significantly, exhibits biological realism not directly encoded in the reward function: realistic thermodynamic stability, natural codon usage patterns, and appropriate ORF length distributions. These parallels to natural plasmids emerge from optimizing only functional annotations, length constraints, and repeat penalties.

This finding suggests that RL post-training operates similarly across modalities by steering models toward coherent, naturalistic regions of their respective sequence spaces. Just as RL unlocked reasoning and generalization in language models beyond their pretraining objectives, it guides DNA models toward biologically realistic sequences beyond their explicit rewards. The unexpected improvement in next-token prediction further supports this interpretation: the model learns general principles of plasmid structure rather than memorizing specific solutions.

These findings open several directions for future work. First, extending to conditional generation (e.g., "express protein X in E. coli with high copy number") would enable practical applications and experimental validation of model-generated sequences. Second, applying RL post-training to other genomic models could test whether emergent biological realism generalizes across applications. Finally, investigating what reward function properties induce such emergence could provide principles for designing effective biological reward functions, potentially accelerating the broader application of RL to computational biology.

Impact Statement

This paper presents work whose goal is to advance the field of computational biology and machine learning for biological sequence design. The ability to generate novel, valid plasmid sequences could accelerate research in synthetic biology, biomanufacturing, and therapeutic development. While the immediate applications are primarily beneficial for scientific research, we acknowledge the dual-use nature of synthetic biology tools. The methods described here are limited to generating plasmid sequences based on patterns in existing databases and do not enable the design of harmful organisms without substantial additional effort and expertise. We believe the benefits to research efficiency and accessibility outweigh the potential risks, particularly given existing biosafety regulations and oversight mechanisms in synthetic biology research.

Code Availability

Code for model training and evaluation will be released upon publication. Training data is derived from Plasmid-Scope and Addgene databases, available under their respective licenses.

Acknowledgements

A. Training Configuration Details

A.1. Supervised Fine-Tuning Hyperparameters

- **Batch size:** 1

- **Learning rate:** 5×10^{-5} with 500 warmup steps

- **Optimizer:** AdamW

- **Epochs:** 3

- **Gradient accumulation steps:** 8

- **Hardware:** [single/multi] NVIDIA L4 GPU(s)

A.2. Reinforcement Learning (GRPO) Hyperparameters

- **Rollout batch size:** 50

- **Policy learning rate:** 3.55×10^{-5} (found via hyperparameter sweep)

- **GRPO group size:** 16

- **Training steps:** Varies between 1000-2500 (scheduled for 2500 but rarely reached due to early convergence)

- **Convergence criteria:** Reward plateau or dip indicating collapse

- **Prompt types:** Random 4–25bp seeds (excluding ATG) and structured cassette seeds

- **Total training time:** ∼10–20 hours on NVIDIA L4 GPU

A.3. Reward Function Configuration

- **Origin of replication (exactly 1):** Weight = 1.0

- **Selectable markers (≥1):** Weight = 1.5

- **Promoter→CDS→terminator bonus:** Weight = 1.5

- **Length prior (2–15kb):** Weight = 0.6, linear with maximum reward at 5kb and minimum at 15kb, zero reward beyond 15kb

- **Repeat penalty (>50bp):** -0.1 from total reward for each repeat > 50bp

A.4. Distribution Comparison Metrics

Table 3 presents Kolmogorov-Smirnov statistics comparing the distributions of generated plasmids to real plasmids across key biophysical metrics. Lower values indicate closer match to the real plasmid distribution.

Table 3. Kolmogorov-Smirnov statistics for distribution comparison against real plasmids. RL model achieves closest match for GC content and codon usage (JS divergence).

METRIC	REAL	BASE	SFT	RL
LENGTH (BP)	4737	5548	5647	7551
GC CONTENT	0.52	0.48	0.50	0.51
JS DIVERGENCE	0.06	0.10	0.09	0.06
MFE DENSITY	-0.36	-0.35	-0.37	-0.35

A.5. Reference Plasmids for Distribution Comparison

The following common laboratory plasmids were used for distribution comparison and benchmarking analyses in Section 4.2: pUC19 (Yanisch-Perron et al., 1985), pBluescript (Short et al., 1988), pBR322 (Bolivar et al., 1977), pA-CYC184 (Chang & Cohen, 1978), pBAD24 (Guzman et al., 1994), pEGFP (Clontech Laboratories, 1999), pGEX-4T-1 (GE Healthcare, 2000), pET-28a (Novagen, 2005), pcDNA3 (Kaufman et al., 1991), and px330 (Cong et al., 2013). These plasmids represent widely-used vectors spanning diverse applications and size ranges (2.7–8.1 kb), providing a representative benchmark for evaluating the biological realism of generated sequences.

References

Addgene. Addgene: The nonprofit plasmid repository. https://www.addgene.org/, 2024. Accessed December 2024.

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.

Berti, L., Giorgi, F., and Kasneci, G. Emergent abilities in large language models: A survey, 2025. URL https://arxiv.org/abs/2503.05788.

Bolivar, F., Rodriguez, R. L., Greene, P. J., Betlach, M. C., Heyneker, H. L., Boyer, H. W., Crosa, J. H., and Falkow, S. Construction and characterization of new cloning vehicles. II. A multipurpose cloning system. *Gene*, 2(2): 95–113, 1977.

Brophy, J. A. and Voigt, C. A. Plasmid design for tunable gene expression in bacteria. *Methods in Enzymology*, 497: 371–388, 2011.

Chang, A. C. and Cohen, S. N. Construction and characterization of new cloning vehicles. V. Mobility and coding functions of pbr322 and pbr325 'derived' plasmids. *Journal of Bacteriology*, 134(3):1141–1156, 1978.

Clontech Laboratories. pEGFP vector information. Product Manual, 1999. Catalog #6077-1.

Cong, L., Ran, F. A., Cox, D., Lin, S., Barretto, R., Habib, N., Hsu, P. D., Wu, X., Jiang, W., Marraffini, L. A., et al. Multiplex genome engineering using CRISPR/Cas systems. *Science*, 339(6121):819–823, 2013.

Cunningham, A. G., Dekker, L., Shcherbakova, A., and Barnes, C. P. Generative design and construction of functional plasmids with a dna language model. *bioRxiv*, 2025. doi: 10.64898/2025.12.06.692736. URL https://www.biorxiv.org/content/early/2025/12/07/2025.12.06.692736.

Deng, Y., Maurais, H. E., Etheridge, K., and Sarpeshkar, R. Gene syntaxes modulate gene expression and circuit behavior on plasmids. *Journal of Biological Engineering*, 19(1):25, 2025. doi: 10.1186/s13036-025-00493-0. URL https://doi.org/10.1186/s13036-025-00493-0.

Durbin, R., Eddy, S. R., Krogh, A., and Mitchison, G. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998. ISBN 9780521629713.

Fung, V., Tiwade, P. B., and Fenton, O. S. Clonefast: A simple plasmid design and construction guide for labs venturing into synthetic biology. *STAR Protocols*, 6(3):104025, 2025. ISSN 2666-1667. doi: https://doi.org/10.1016/j.xpro.2025.104025. URL https://www.sciencedirect.com/science/article/pii/S2666166725004319.

Galata, V., Fehlmann, T., Backes, C., and Keller, A. Plsdb: a resource of complete bacterial plasmids. *Nucleic Acids Research*, 47(D1):D195–D202, 10 2018. ISSN 0305-1048. doi: 10.1093/nar/gky1050. URL https://doi.org/10.1093/nar/gky1050.

GE Healthcare. pGEX vectors. Product Manual, 2000. GST Gene Fusion System.

Guzman, L.-M., Belin, D., Carson, M. J., and Beckwith, J. Making transcriptional silencers work in Escherichia coli. *Molecular Microbiology*, 13(4):655–662, 1994.

Hyatt, D., Chen, G.-L., LoCascio, P. F., Land, M. L., Larimer, F. W., and Hauser, L. J. Prodigal: prokaryotic gene recognition and translation initiation site identification. *BMC Bioinformatics*, 11(1):1–11, 2010. doi: 10.1186/1471-2105-11-119.

Ji, Y., Zhou, Z., Liu, H., and Davuluri, R. V. DNABERT: pre-trained bidirectional encoder representations from transformers model for DNA-language in genome. *Bioinformatics*, 37(15):2112–2120, 2021.

Kaufman, R. J., Davies, M. V., Pathak, V. K., and Hershey, J. W. A mammalian expression vector for the expression of complete open reading frames. *Nucleic Acids Research*, 19(16):4485–4490, 1991.

Kutzler, M. A. and Weiner, D. B. Plasmid DNA vaccines: an overview. *Vaccine*, 26:S59–S75, 2008.

Lederberg, J. Cell genetics and hereditary symbiosis. *Physiological Reviews*, 32(4):403–430, 1952. Seminal paper defining the term "plasmid".

Lin, Y., Dong, H., Wang, H., Zhang, J., Liu, J., Pi, R., Pan, R., Zhang, H., Hu, W., Zhao, H., et al. Mitigating the alignment tax of RLHF. *arXiv preprint arXiv:2309.06256*, 2023.

Lorenz, R., Bernhart, S. H., Höner zu Siederdissen, C., Tafer, H., Flamm, C., Stadler, P. F., and Hofacker, I. L. Viennarna package 2.0. *Algorithms for Molecular Biology*, 6 (1):26, 2011.

Martinson, J. N. V. et al. Generating functional plasmid origins with OriGen. *Nucleic Acids Research*, 53(22): gkaf1198, 2025. doi: 10.1093/nar/gkaf1198.

Meng, F. and Ellis, T. Challenges in rational design of synthetic promoters. *New Biotechnology*, 32(3):337–344, 2015.

Naseri, G. and Koffas, M. A. G. Application of combinatorial optimization strategies in synthetic biology. *Nature Communications*, 11(1), 2020. doi: 10.1038/s41467-020-16175-y.

Nguyen, E., Poli, M., Faizi, M., Thomas, A., Birch-Sykes, C., Wornow, M., Patel, A., Rabideau, C., Massaroli, S., Bengio, Y., et al. HyenaDNA: Long-range genomic sequence modeling at single nucleotide resolution. *arXiv preprint arXiv:2306.15794*, 2023.

Nguyen, E., Poli, M., Durrant, M. G., Kang, B., Katrekar, D., Li, D. B., Bartie, A., Thomas, A. W., King, S. H., Bifulco, G., et al. Sequence modeling and design from molecular to genome scale with Evo. *bioRxiv*, 2024. doi: 10.1101/2024.02.27.582234.

Novagen. pET system manual. Product Manual, 11th Edition, 2005. TB055.

Oliveira, P. H., Prather, K. L., Prazeres, D. M., and Monteiro, G. A. Structural instability of plasmid biopharmaceuticals: challenges and implications. *Trends in Biotechnology*, 27(9):503–511, 2009.

Prather, K. J., Sagar, S., Murphy, J., and Chartrain, M. Industrial scale production of plasmid DNA for vaccine and gene therapy: plasmid design, production, and purification. *Enzyme and Microbial Technology*, 33(7):865–883, 2003.

Shao, L. et al. PlasmidGPT: a generative framework for plasmid design and annotation. *bioRxiv*, 2024a. doi: 10.1101/2024.09.30.615762. Preprint.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Zhang, M., Li, Y. K., Wu, Y., and Guo, D. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024b.

Short, J., Fernandez, J., Sorge, J., and Huse, W. Bluescript: a high copy number Escherichia coli plasmid vector. *Nucleic Acids Research*, 16(15):7583–7600, 1988.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. Emergent abilities of large language models, 2022. URL https://arxiv.org/abs/2206.07682.

Wu, G., Yan, Q., Jones, J. A., Tang, Y. J., Fong, S. S., and Koffas, M. A. Metabolic burden: cornerstones in synthetic biology and metabolic engineering applications. *Trends in Biotechnology*, 34(8):652–664, 2016.

Yanisch-Perron, C., Vieira, J., and Messing, J. New M13 vectors for cloning. *Gene*, 33(1):103–119, 1985.