REGULATORY DNA SEQUENCE DESIGN WITH REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Cis-regulatory elements (CREs), such as promoters and enhancers, are relatively short DNA sequences that directly regulate the expression of specific genes. The fitness of CRE, i.e., its functionality to enhance gene expression, highly depend on its nucleotide sequence, especially the composition of some special motifs known as transcription factor binding sites (TFBSs). Designing CREs to optimize their fitness is crucial for therapeutic and bioengineering applications. Existing CRE design methods often rely on simple strategies, such as iteratively introducing random mutations and selecting variants with high fitness from a large number of candidates through an oracle, i.e., a pre-trained gene expression prediction model. Due to the vast search space and lack of prior biological knowledge guidance, these methods are prone to getting trapped in local optima and tend to produce CREs with low diversity. In this paper, we propose the first method that leverages reinforcement learning (RL) to fine-tune a pre-trained autoregressive (AR) generative model for designing high-fitness cell-type-specific CREs while maintaining sequence diversity. We employ prior knowledge of CRE regulatory mechanisms to guide the optimization by incorporating the role of TFBSs into the RL process. In this way, our method encourages the removal of repressor motifs and the addition of activator motifs. We evaluate our method on enhancer design tasks for three distinct human cell types and promoter design tasks in two different yeast media conditions, demonstrating its effectiveness and robustness in generating high-fitness CREs.

1 INTRODUCTION

Cis-regulatory elements (CREs), such as promoters and enhancers, are short functional DNA sequences that regulate gene expression in a cell-type-specific manner. Promoters determine when and where a gene is activated, while enhancers boost gene expression levels. Over the past decade, millions of putative CREs have been identified, but these naturally evolved sequences only represent a small fraction of the possible genetic landscape and are not necessarily optimal for specific expression outcomes. It is crucial to design synthetic CREs with desired fitness (measured by their ability to enhance gene expression) as they have broad applications in areas such as gene therapy (Boye et al., 2013), synthetic biology (Shao et al., 2024), precision medicine (Collins & Varmus, 2015), and agricultural biotechnology (Gao, 2018).

Previous attempts to explore alternative CREs have relied heavily on directed evolution, which involves iterative cycles of mutation and selection in wet-lab settings (Wittkopp & Kalay, 2012; Heinz et al., 2015). This approach is sub-optimal due to the vastness of the DNA sequence space and the significant time and cost required for experimental validation. For example, a 200 base pair (bp) DNA sequence can have up to 2.58×10^{120} possible combinations (Gosai et al., 2023), far exceeding the number of atoms in the observable universe. Thus, efficient computational algorithms are needed to narrow down the design space and prioritize candidates for wet-lab testing.

Advances in high-throughput sequencing technologies, such as massively parallel reporter assays (MPRAs) (de Boer et al., 2020; Vaishnav et al., 2022), have enabled the screening of large libraries of random DNA sequences and the measurement of their activity in specific cell types. Based on these data, two categories of deep learning approaches for CRE modeling have been developed. One category focuses on training predictive models (Avsec et al., 2021; Mallet & Vert, 2021) to estimate

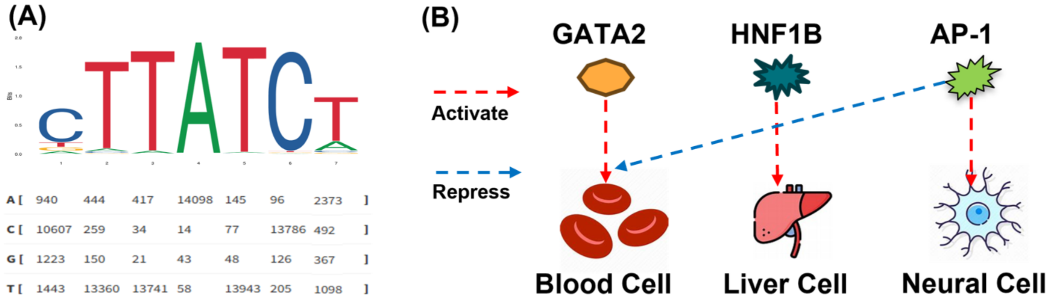
(A)
A [940 444 417 14098 145 96 2373]
C [10607 259 34 14 77 13786 492]
G [1223 150 21 43 48 126 367]
T [1443 13360 13741 58 13943 205 1098]
(B)
GATA2
HNF1B
AP-1
Activate
Repress
Blood Cell
Liver Cell
Neural Cell

Figure 1: (A) TFBS are commonly represented as frequency matrices, indicating the probability of each nucleotide appearing at specific positions within the binding site. (B) GATA2 and HNF1B specifically activate gene expression in blood cells and liver cells, respectively, while REST specifically represses gene expression in neural cells.

the fitness of CREs based on their sequences. The other category builds conditional generative models (Avdeyev et al., 2023b; Li et al., 2024b; Avdeyev et al., 2023a; Lal et al., 2024) to model the conditional distribution of CREs. However, these approaches cannot directly design new CREs with desired properties.

Recent studies have begun using fitness prediction models as oracles to guide CRE optimization, enabling the exploration of sequences that outperform naturally occurring ones (Vaishnav et al., 2022; de Almeida et al., 2024). These methods typically rely on straightforward optimization approaches, such as genetic algorithms or greedy-based directed evolution, which involve two iterative steps: randomly mutating sequences selected in the previous step to form candidates and selecting those with high fitness through an oracle. The entire search space of all possible candidates is vast, but the exploration in each step is performed by heuristic random mutations. Neither empirically learned policies nor any prior biological knowledge are used to guide exploration. As a consequence, these methods are prone to getting trapped in local optima and the produced CREs tend to lack diversity and interpretability.

Inspired by the success of using Reinforcement Learning (RL) for finetuning autoregressive (AR) generative language models (Brown et al., 2020; Ouyang et al., 2022; Liu et al., 2024), we propose the first RL for AR model-based method to design cell-type-specific CREs. We pretrain state-of-the-art (SOTA) AR DNA generative models HyenaDNA (Nguyen et al., 2024b; Lal et al., 2024) on CREs to capture their authentic distribution, ensuring the generation of realistic and diverse CRE sequences. During RL finetuning, we treat the current AR model as the policy network, and utilize the fitness predicted by an oracle as the reward signal. This allows us to update the model parameters to generate CRE sequences that not only maintain diversity but also exhibit high fitness.

Model	yeast		human		
	complex	defined	hepg2	k562	sknsh
Enformer (Sequence Feature)	0.87	0.91	0.83	0.85	0.85
LightGBM (TFBS Frequency Feature)	0.63	0.65	0.65	0.65	0.66

Table 1: **Pearson correlation coefficient of the Oracle on the test set**. Enformer is a SOTA DNA backbone model that uses DNA sequences as input, while LightGBM is a simple decision tree model that uses TFBS occurrence frequencies as input.

Additionally, we incorporate domain knowledge of CREs into our RL process. The regulatory syntax of CREs is largely dictated by the transcription factors (TFs) that bind to them (Gosai et al., 2023; de Almeida et al., 2024; Lal et al., 2024; Zhang et al., 2023). TFs are proteins that directly influence gene expression by binding to specific sequence motifs within CREs, known as TF binding sites (TFBSs), and modulating transcriptional activity. For instance, Fig. 1(A) shows the motif pattern recognized by the GATA2 TF. Furthermore, the effects of TFs can vary widely depending on the cell type. As shown in Fig. 1 (B), GATA2 and HNF1B are TFs that specifically activate gene expression in blood cells and liver cells (Lal et al., 2024), respectively, while REST acts as a repressor of gene expression in neural cells (Zullo et al., 2019), illustrating the cell-type-specific nature of TF activity.

The effect of a TF can be broken down into its intrinsic role as an activator or repressor (referred to as its ”vocabulary”) and its interactions with other TFs (such as composition and arrangement). We found that simply using the frequency of TFBS occurrences within a sequence as features can achieve reasonably good fitness prediction performance when trained with a decision tree model LightGBM (Ke et al., 2017). As shown in Tab. 1, the current SOTA DNA model, Enformer, achieves a Pearson correlation of 0.83 on the test set for predicting fitness in the HepG2 cell line using sequence data as input. In contrast, using only simple TFBS frequency features—without any explicit sequence information—achieved a Pearson correlation of 0.65. This demonstrates that even without leveraging sequence details, TF frequency alone can capture a significant portion of the predictive power. Furthermore, we use the trained LightGBM (Ke et al., 2017) model to infer whether each TFBS feature promotes or represses fitness, which allows us to explicitly incorporate TFBS domain knowledge into our RL process. We name our proposed method **TACO**: **T**FBS-**A**ware ***C**is-Regulatory* Element **O**ptimization, which integrates RL finetuning of AR models with domain knowledge of TFBSs to enhance CRE optimization.

Our main contributions are as follows:

- We are the first to introduce the RL paradigm to AR DNA models for CRE design, allowing the generated sequences to not only maintain high diversity but also explore those with higher functional performance.
- We incorporate key TFBS information by inferring their regulatory roles and integrating their impact directly into the generation process, allowing for joint data-driven and knowledge-driven exploration guidance.
- We evaluate our approach on real-world datasets, including yeast promoter designs from two media and human enhancer designs from three cell lines.

2 RELATED WORK

Conditional DNA Generative Models. DDSM (Avdeyev et al., 2023a) was the first to apply diffusion models to DNA design. By leveraging classifier-free guidance Ho & Salimans (2022), the model conditioned DNA sequences on promoter expression levels. Following this, several works have employed diffusion models for CRE design Li et al. (2024b); DaSilva et al. (2024); Sarkar et al. (2024); Avdeyev et al. (2023b). In addition to diffusion models, RegLM (Lal et al., 2024) utilized prefix-tuning on the AR DNA language model HyenaDNA (Nguyen et al., 2024b), incorporating tokens that encode expression strength to fine-tune the model specifically for CRE design. However, these generative methods are designed to fit existing data distributions, limiting their ability to design sequences that have yet to be explored by humans.

DNA Sequence Optimization. Early DNA optimization methods (Jain et al., 2022; Angermueller et al., 2019; Zeng et al., 2024) primarily focused on optimizing short TFBS motifs (6-8bp). With the availability of larger CRE fitness datasets, Vaishnav et al. (2022) applied genetic algorithms to design CREs. Recent works, such as Gosai et al. (2023), explored greedy approaches like AdaLead (Sinai et al., 2020), simulated annealing (Van Laarhoven et al., 1987), and gradient-based SeqProp (Linder & Seelig, 2021). Similarly, Taskiran et al. (2024) combined greedy strategies with directed evolution. However, these methods often start from random sequences, generating biologically irrelevant sequences, or begin with observed high-fitness sequences, leading to local optima and limited diversity. In contrast, we initialize optimization with a pretrained generative model and refine it using RL, addressing both issues.

Motif-based Machine Learning. Motifs are often regarded as small, critical elements in scientific data, such as functional groups in molecules or TFBS in DNA sequences. In machine learning, explicitly modeling these motifs can provide significant benefits. For example, motifs have been successfully used in molecular optimization (Jin et al., 2020; Chen et al., 2021), molecular generation Geng et al., molecular property prediction (Zhang et al., 2021), and DNA language models (An et al., 2022). In the context of DNA CREs, TFBS are widely considered the most important motifs. TFBS typically exhibit cell-type specificity, i.e., the same TFBS may play different roles in different cell types. Our approach is inspired by de Almeida et al. (2024), who observed that during direct evolution guided by an oracle, there is a tendency to first remove repressor TFBS and subsequently add enhancer TFBS to optimize the sequences.

Algorithm 1 TACO: RL-Based Fine-tuning for Autoregressive DNA Models

```
Require: Low-fitness dataset $\mathcal{D}^*$, TFBS vocabulary $\mathcal{T}$, Oracle $q_\theta$, Pretrained AR model $\pi_\theta$, Number of Optimization Rounds $E$
Preprocessing:
Train LightGBM model on TFBS frequency features $\mathbf{h}(x)$ from dataset $\mathcal{D}^*$
Compute SHAP values $\phi_i(x)$ for each TFBS $t_i$
Update TFBS rewards $r_{\text{TFBS}}(t)$ based on equation 6
for round $e = 1$ to $E$ do
    Sample a batch of sequences $\{x_i\}$ from policy $\pi_\theta$
    for each sequence $x_i$ do
        for time step $t = 1$ to $L$ do
            Generate nucleotide $a_t$ using $\pi_\theta(a_t|a_{<t})$
            Observe state $s_t = (a_1, \dots, a_{t-1})$
            if $a_t$ results in TFBS $t \in \mathcal{T}$ then
                Assign reward $r(s_t, a_t) \leftarrow r_{\text{TFBS}}(t)$
            else
                Assign reward $r(s_t, a_t) \leftarrow 0$
            end if
        end for
        Obtain fitness reward $r_{\text{fitness}}$ from oracle $q_\theta(x_i)$
        Compute total reward $R \leftarrow \sum_{t=1}^{L} r(s_t, a_t) + r_{\text{fitness}}$
    end for
    Update policy $\pi_\theta$ using REINFORCE:
        $\theta \leftarrow \theta + \alpha \nabla_\theta \mathbb{E}_{\pi_\theta}[R \log \pi_\theta(a_t|s_t)]$
end for
```

3 METHOD

3.1 PROBLEM FORMULATION

We define a DNA sequence $x = (x_1, \cdots, x_L)$ as a string of nucleotides with length L, where $x_i \in \mathcal{V}$ is the nucleotide at the i-th position, and $\mathcal{V}$ is the vocabulary of 4 nucleotides (A, T, C, G). In our CRE optimization task, we assume the availability of a large-scale dataset of CRE sequences with fitness measurements $\mathcal{D} = \{(x^1, f(x^1)), \cdots, (x^N, f(x^N))\}$ to train an ideal *in-silico* oracle q_θ, where N is the number of sequences in the dataset and $f(x)$ represents the fitness measurement for sequence x. Here, we use the term ***fitness*** to denote the desired regulatory activity of a CRE sequence. We follow the setting used in protein optimization (Kirjner et al., 2023; Lee et al., 2024) by sampling a set of low-fitness sequences $\mathcal{D}^*$ from $\mathcal{D}$, which includes only sequences with fitness values below a certain percentile of $\mathcal{D}^*$. This approach helps avoid generating sequences with fitness values outside the observed range, thereby ensuring the reliability of oracle predictions.Note that the dataset $\mathcal{D}$ is cell-type-specific, meaning that each sequence's fitness value corresponds to its regulatory activity within a specific cell type.

3.2 OVERVIEW

Our method consists of two main components. The first component involves fine-tuning an AR generative model, pretrained on CRE sequences, using RL (see Fig. 2). The second component is a data-driven approach to infer the role of TFBSs in a cell-type-specific manner within the given dataset (see Fig. 3). The inferred roles are then incorporated into the RL process to guide sequence generation. The complete algorithmic workflow is presented in Alg. 1.

3.3 RL-BASED FINETUNING FOR AUTOREGRESSIVE DNA MODELS

Pretraining AR Model. First, we pretrain an AR model on the low-fitness dataset $\mathcal{D}^*$ following (Lal et al., 2024), using the HyenaDNA architecture (Nguyen et al., 2024b) (More details in Appendix D), which achieves strong performance on DNA tasks by maintaining both linear com-

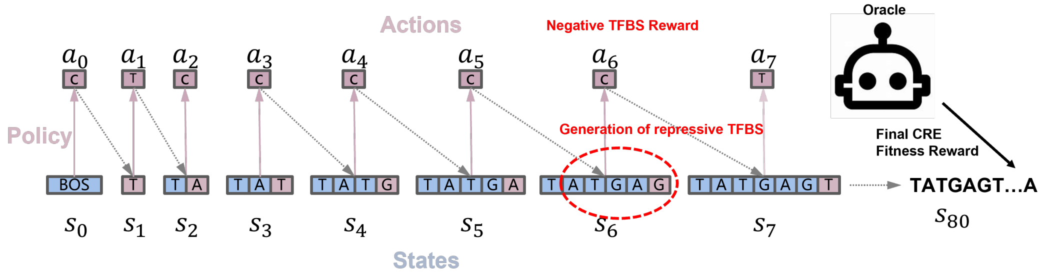

Figure 2: **The autoregressive generation of a DNA sequence.** An AR model for sequence generation can be viewed as an RL policy, where the actions a_t represent the next nucleotides to be appended to the sequence, and the state is the concatenation of all actions taken up to time $t-1$. If an action generates a TFBS that is known to be repressive, a negative reward is given. Conversely, generating a TFBS with activating properties results in a positive reward. The final sequence is evaluated using an oracle to obtain a fitness reward.

plexity and high performance. The pretrained AR model, denoted as π_θ, is trained to predict the probability distribution of the next nucleotide given the preceding sequence:

$$\min_\theta \mathbb{E}_{x\sim\mathcal{D}^*}\left[\sum_{t=1}^{L} -\log p_\theta(a_t = A_t \mid A_{t-1}, \cdots, A_0)\right], \tag{1}$$

where A_t represents the nucleotide at position t, which corresponds to the action a_t taken by the model. This alignment ensures that the notation for nucleotides is consistent with the actions in the RL setting. Pretraining on $\mathcal{D}^*$ helps the policy learn to generate sequences that already resemble the true CRE distribution (Jin et al., 2020; Chen et al., 2021), providing a good initialization for RL finetuning and promoting diversity in the generated sequences. Moreover, using the generative model as the policy ensures that the generated CREs maintain high diversity throughout the optimization process.

RL-Based Finetuning for AR DNA Models. Next, we formulate the RL finetuning process as a Markov Decision Process (MDP), as illustrated in Fig. 2. In this formulation, the states s_t correspond to the partial sequences generated up to time step t, while the actions a_t represent the nucleotides selected by the policy π_θ. The reward $r(s_t, a_t)$ is defined as a combination of two types of rewards: TFBS reward r_{TFBS} and fitness reward r_{fitness}, as shown in equation 2:

$$r(s_t, a_t) = \begin{cases} r_{\text{fitness}}, & \text{if } t = T, \\ r_{\text{TFBS}}(t), & \text{if } a_t \text{ results in a TFBS } t \in \mathcal{T}, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Here, r_{fitness} is applied when t is the final time step of the episode ($t = T$), and represents the fitness value of the generated sequence as evaluated by the oracle. On the other hand, r_{TFBS} is a reward applied whenever a TFBS $t \in \mathcal{T} = \{t_1, t_2, t_3, \ldots, t_n\}$ is identified in the sequence after selecting a_t. Details on how TFBSs are identified can be found in Appendix E. The specific values of $r_{\text{TFBS}}(t)$ are discussed in Subsec. 3.4. Negative rewards are assigned for generating repressive TFBSs, while positive rewards are given for generating activating TFBSs, as shown in Fig. 2. The overall objective is to maximize the expected cumulative reward:

$$\max_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=1}^{T} r(s_t, a_t)\right] \tag{3}$$

where $J(\theta)$ represents the expected cumulative reward, T is the length of the episode, and $r(s_t, a_t)$ is the reward at each time step. This setup ensures that the AR model can learn to generate DNA sequences with the desired regulatory properties by leveraging both sequence structure and domain-specific knowledge of TFBS vocabulary.

RL Implementation Details. To optimize the policy π_θ, we employ the REINFORCE algorithm (Williams, 1992). Similar to previous studies in molecule optimization (Ghugare et al., 2024), we observed that REINFORCE achieves better results than PPO (Schulman et al., 2017) for DNA sequence generation tasks. Additionally, we leverage a hill climbing replay buffer (Blaschke et al., 2020), which stores and samples high-fitness sequences during training to further guide exploration. We also apply $-\frac{1}{\log \pi(a|s)}$ regularization, which penalizes sequences with high likelihood, thereby encouraging the model to explore sequences with lower likelihood. This combination of techniques enables the model to balance exploration and exploitation effectively, leading to improved performance on complex DNA optimization tasks.

3.4 INFERENCE OF TFBS REGULATORY ROLES

As illustrated in Fig. 3, our approach to inferring TFBS regulatory roles consists of two steps. First, we train a decision tree-based fitness prediction model using TFBS frequency features as input. Second, we leverage model interpretability techniques to determine the regulatory impact of each TFBS feature.

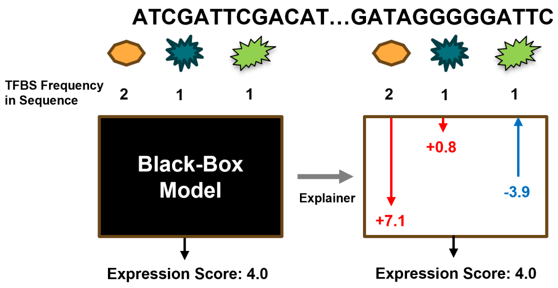

Figure 3: TFBS Regulatory Analysis. (A) TFBS frequency in the sequence. (B) SHAP values show positive and negative contributions to the expression score, highlighting cell-type-specific regulatory roles.

To infer the regulatory impact of each TFBS, we first define the TFBS frequency feature of a sequence x as a vector $\mathbf{h}(x) = [\mathbf{h}_1(x), \mathbf{h}_2(x), \ldots, \mathbf{h}_n(x)]$, where $\mathbf{h}_i(x)$ denotes the frequency of the i-th TFBS in sequence x. This feature vector represents the occurrence pattern of TFBSs within the sequence, making it suitable for tabular data modeling. Details on extracting TFBS features by scanning the sequence can be found in Appendix E. Given the tabular nature of this data, we employ LightGBM (Ke et al., 2017), a tree-based model known for its interpretability and performance on tabular datasets, to fit the fitness values of sequences. LightGBM is chosen because decision tree models, in general, offer better interpretability by breaking down the contribution of each feature in a clear, hierarchical manner.

The LightGBM model is trained to map the TFBS frequency features to the corresponding fitness values of sequences, using the objective function:

$$\min_{\gamma} \sum_{(\mathbf{h}(x), u(x)) \in \mathcal{D}^*} d\left(u(x), \hat{u}(\mathbf{h}(x); \gamma)\right), \tag{4}$$

where $u(x)$ is the true fitness value of sequence x, $\hat{u}(\mathbf{h}(x); \gamma)$ is the fitness value predicted by the LightGBM model parameterized by γ using the TFBS frequency feature vector $\mathbf{h}(x)$. The term $d\left(u(x), \hat{u}(\mathbf{h}(x); \gamma)\right)$ represents a distance metric measuring the discrepancy between the true and predicted fitness values. A detailed discussion on the choice of d can be found in Appendix F.

After training, we evaluate the model's performance using the Pearson correlation coefficient between the true and predicted fitness values, as shown in Tab. 1. This evaluation metric helps us quantify how well the LightGBM model captures the relationship between TFBS frequencies and fitness values.

Based on the trained LightGBM model, we use SHAP values (Lundberg, 2017) to interpret the impact of each TFBS on the predicted fitness. SHAP values provide a theoretically grounded approach to attribute the prediction of a model to its input features by calculating the contribution of each feature (in our case, each TFBS) to the prediction. The SHAP value for the i-th TFBS in sequence x, denoted as $\phi_i(x)$, is computed as:

$$\phi_i(x) = \sum_{S \subseteq \{1,\ldots,n\} \setminus \{i\}} \frac{|S|!(n - |S| - 1)!}{n!} \left(f(S \cup \{i\}) - f(S) \right), \tag{5}$$

where S is a subset of features not containing i, $f(S \cup \{i\})$ is the model prediction when feature i is included, and $f(S)$ is the prediction when feature i is excluded. This equation ensures that SHAP values fairly distribute the impact of each feature according to its contribution.

To infer the reward $r_{\text{TFBS}}(t)$ for each TFBS $t \in \mathcal{T} = \{t_1, t_2, t_3, \ldots, t_n\}$, we compute the mean SHAP value of t over the entire dataset. If the mean SHAP value does not significantly differ from zero (p-value > 0.05, determined by hypothesis testing), we set the reward of t to zero:

$$r_{\text{TFBS}}(t) = \begin{cases} \alpha \cdot \mu_\phi(t), & \text{if } p\text{-value} < 0.05, \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

where α is a tunable hyperparameter, and $\mu_\phi(t)$ is the mean SHAP value of TFBS t across the dataset. This approach ensures that only statistically significant TFBSs contribute to the reward, and α controls the magnitude of the reward.

4 EXPERIMENT

4.1 EXPERIMENT SETUP

Datasets and Oracles. We conduct experiments on both yeast promoter and human enhancer datasets. The yeast promoter dataset includes two types of growth media: *complex* (de Boer et al., 2020) and *defined* (Vaishnav et al., 2022). The human enhancer dataset consists of three cell lines: HepG2, K562, and SK-N-SH (Gosai et al., 2023). All paired CRE sequences and their corresponding fitness measurements were obtained from massively parallel reporter assays (MPRAs) (Sharon et al., 2012). Our dataset partitioning strategy is based on prior work in DNA sequence generation models (Lal et al., 2024), while our multi-round optimization settings follow methodologies used in protein sequence optimization (Lee et al., 2024). The DNA sequence length in the yeast promoter dataset is 80, while it is 200 for the human enhancer dataset.

Each dataset represents a cell-type-specific scenario due to distinct TF effect vocabularies and regulatory landscapes. To simulate optimization from low-fitness CREs, we employ fitness predictors trained on the complete dataset D as oracles (Lal et al., 2024). These oracles guide the optimization process of an AR model that is pretrained on a subset of sequences, D^*, within a specified fitness range. We partition each dataset into three subsets—*easy*, *medium*, and *hard*—based on their fitness values. Detailed partitioning strategies are provided in Appendix B. We set the maximum number of optimization iterations to 100, with up to 256 oracle calls allowed per iteration.

Baselines. We compare our method, TACO, against several established optimization approaches, including Bayesian optimization as implemented in the FLEXS benchmark (Sinai et al., 2020), and evolutionary algorithms such as AdaLead (Sinai et al., 2020) and PEX (Anand & Achim, 2022), as well as covariance matrix adaptation evolution strategy (CMAES) (Hansen) using one-hot encoding. Additionally, we adapte the SOTA protein optimization method LatProtRL (Lee et al., 2024) for CRE optimization. Given the lack of a powerful backbone model like ESM (Jain et al., 2022) in the DNA domain, we remove the ESM-based latent vector encoding from LatProtRL and refer to the resulting model as DNARL. DNARL can be viewed as a sequence mutation-based PPO algorithm (Schulman et al., 2017) enhanced with a replay buffer mechanism.

Evaluation Metrics We employ three evaluation metrics: Top, Medium, and Diversity. *Top* is defined as the mean fitness value of the top 6 sequences in the optimized set $\mathcal{G}^* = \{g_1^*, \cdots, g_K^*\}$, highlighting the highest-performing sequences in terms of fitness. *Medium* refers to the median fitness value of all $K = 128$ generated sequences, providing an overall measure of fitness across the entire set. *Diversity* is calculated as the median pairwise distance between every pair of sequences

Yeast Promoter (Complex)

Method	easy			medium			hard		
	Top	Medium	Diversity	Top	Medium	Diversity	Top	Medium	Diversity
PEX	1	1	8.6 ± 1.14	1	1	8.4 ± 1.95	1	1	9.8 ± 1.48
AdaLead	1	1	8.8 ± 1.3	1	1	9.0 ± 1.58	1	1	7.6 ± 0.89
BO	1	1	23.4 ± 1.52	1	1	22.6 ± 1.34	1	1	25.0 ± 5.57
CMAES	1	0.78 ± 0.13	30.2 ± 2.68	1	0.85 ± 0.02	29.4 ± 1.52	1	0.79 ± 0.09	30.0 ± 2.5
DNARL	1	1	8.6 ± 2.14	1	1	10.2 ± 1.14	1	1	7.7 ± 0.48
TACO	1	1	**52.2** ± 1.92	1	1	**48.8** ± 5.36	1	1	**52.8** ± 2.77

Yeast Promoter (Defined)

Method	easy			medium			hard		
	Top	Medium	Diversity	Top	Medium	Diversity	Top	Medium	Diversity
PEX	1	1	9.2 ± 0.84	1	1	9.2 ± 1.79	1	1	9.8 ± 2.59
AdaLead	1	1	8.0 ± 2.35	1	1	7.0 ± 1.0	1	1	6.4 ± 0.55
BO	1	1	23.0 ± 1.58	1	1	22.8 ± 2.28	1	1	23.0 ± 1.87
CMAES	1	0.26 ± 0.36	30.0 ± 2.92	1	0.48 ± 0.17	29.8 ± 1.3	1	0.44 ± 0.33	30.4 ± 2.3
DNARL	1	1	11.6 ± 3.04	1	1	18.5 ± 3.0	1	1	10.2 + 1.14
TACO	1	1	**43.2** ± 2.77	1	1	**47.0** ± 4.64	1	1	**49.6** ± 3.65

Table 2: Performance comparison of different algorithms on yeast promoter datasets.

in $\mathcal{G}^*$, reflecting the variability among the generated sequences and ensuring that the optimization process does not converge to a single solution. These metrics are consistent with those used in LatProtRL (Lee et al., 2024), except for the *Novelty* metric. We omit *Novelty* because, unlike proteins, DNA sequences lack well-defined structural constraints, making novelty values disproportionately high and less meaningful. For further details, refer to Appendix G.

Implementation Details. We base the architecture of AR model, i.e., the policy network, on HyenaDNA-1M[1]. We pre-train all initial policies on the subset D^* (Lal et al., 2024). We conduct all experiments on a single NVIDIA A100 GPU. During optimization, we set the learning rate to 5e-4 for the yeast task and 1e-4 for the human task. We set the hyperparameter α, which controls the strength of the TFBS reward in equation 6, to 0.01. We min-max normalize all reported fitness values and the rewards used for updating the policy, while the oracles are trained on the original fitness values.

4.2 FITNESS OPTIMIZATION

We report the meand and standard deviation of the evaluation metrics of 5 runs with different random seeds.

Yeast Promoters. As shown in Tab. 2, optimizing yeast promoters is relatively easy, with most methods successfully generating sequences that surpass the maximum fitness values observed in the dataset. For sequences with fitness values exceeding the maximum, we report the result as 1. Among the baselines, only CMAES fails to fully optimize to the maximum fitness value, but it demonstrates good performance in terms of diversity. Our method not only achieves the maximum fitness but also exhibits the highest diversity compared to other approaches.

Human Enhancers. Optimizing cell-type-specific human enhancers is a more challenging task. As shown in Tab. 4, the 90th percentile min-max normalized fitness values for HepG2, K562, and SK-N-SH in the real dataset D are 0.4547, 0.4541, and 0.4453, respectively. In Tab. 3, our TACO method demonstrates superior performance compared to the baselines. For the HepG2 cell line, PEX achieves the highest fitness score, but its diversity is typically below 20. In contrast, TACO attains state-of-the-art fitness for K562 and SK-N-SH cell lines while maintaining significantly higher diversity across all datasets (over 1/3 higher than CMAES, which has the highest diversity among baselines).

Cell Line	75th Percentile	90th Percentile
HepG2	0.3994	0.4547
K562	0.3975	0.4541
SK-N-SH	0.3986	0.4453

Table 4: Enhancer fitness.

---

[1] `https://huggingface.co/LongSafari/hyenadna-large-1m-seqlen-hf`

Method	HepG2-easy			HepG2-medium			HepG2-hard		
	Top	Medium	Diversity	Top	Medium	Diversity	Top	Medium	Diversity
PEX	**0.93** ± 0.02	**0.89** ± 0.01	20.2 ± 6.57	**0.89** ± 0.04	**0.86** ± 0.04	19.2 ± 7.12	**0.85** ± 0.04	**0.82** ± 0.02	16.0 ± 2.65
AdaLead	0.76 ± 0.0	0.75 ± 0.0	5.2 ± 0.45	0.75 ± 0.03	0.74 ± 0.03	12.4 ± 4.04	0.74 ± 0.02	0.73 ± 0.02	8.0 ± 1.87
BO	0.66 ± 0.06	0.6 ± 0.09	41.6 ± 8.91	0.63 ± 0.05	0.58 ± 0.05	42.0 ± 7.81	0.68 ± 0.04	0.63 ± 0.08	39.8 ± 5.07
CMAES	0.61 ± 0.06	0.42 ± 0.04	77.4 ± 4.04	0.67 ± 0.02	0.43 ± 0.03	75.0 ± 3.24	0.69 ± 0.03	0.43 ± 0.02	77.2 ± 5.17
DNARL	0.79 ± 0.07	0.71 ± 0.02	12.2 ± 0.08	0.63 ± 0.14	0.84 ± 0.09	7.32 ± 0.01	0.76 ± 0.04	0.72 ± 0.01	20.0 ± 3.42
TACO	0.78 ± 0.01	0.75 ± 0.01	**131.8** ± 2.39	0.76 ± 0.01	0.73 ± 0.01	**139.4** ± 7.13	0.76 ± 0.01	0.74 ± 0.01	**131.8** ± 4.27

Method	K562-easy			K562-medium			K562-hard		
	Top	Medium	Diversity	Top	Medium	Diversity	Top	Medium	Diversity
PEX	**0.95** ± 0.01	**0.93** ± 0.01	21.8 ± 9.68	0.94 ± 0.01	**0.92** ± 0.01	14.6 ± 1.82	**0.95** ± 0.01	**0.92** ± 0.02	15.9 ± 1.34
AdaLead	0.85 ± 0.01	0.84 ± 0.01	7.0 ± 1.0	0.85 ± 0.01	0.84 ± 0.01	9.0 ± 1.87	0.85 ± 0.01	0.84 ± 0.01	8.8 ± 1.64
BO	0.7 ± 0.13	0.65 ± 0.12	41.6 ± 5.32	0.76 ± 0.05	0.7 ± 0.05	39.6 ± 5.55	0.74 ± 0.03	0.7 ± 0.04	37.0 ± 6.52
CMAES	0.7 ± 0.05	0.42 ± 0.02	78.8 ± 4.09	0.79 ± 0.03	0.5 ± 0.03	76.0 ± 3.24	0.73 ± 0.05	0.47 ± 0.05	76.8 ± 4.55
DNARL	0.89 ± 0.04	0.87 ± 0.01	23.3 ± 3.72	0.90 ± 0.02	0.86 ± 0.01	26.3 ± 1.88	0.89 ± 0.01	0.87 ± 0.02	17.5 ± 3.33
TACO	0.93 ± 0.0	0.91 ± 0.01	**124.6** ± 3.51	0.92 ± 0.01	0.9 ± 0.02	**126.0** ± 1.58	0.93 ± 0.01	0.91 ± 0.01	**125.6** ± 2.88

Method	SK-N-SH easy			SK-N-SH medium			SK-N-SH hard		
	Top	Medium	Diversity	Top	Medium	Diversity	Top	Medium	Diversity
PEX	0.9 ± 0.01	0.86 ± 0.03	22.2 ± 5.93	**0.92** ± 0.02	**0.88** ± 0.01	23.8 ± 7.85	0.9 ± 0.02	0.86 ± 0.03	23.0 ± 2.74
AdaLead	0.84 ± 0.08	0.82 ± 0.08	7.4 ± 1.52	0.81 ± 0.06	0.8 ± 0.06	9.4 ± 3.05	0.79 ± 0.05	0.78 ± 0.05	14.4 ± 4.45
BO	0.68 ± 0.07	0.62 ± 0.07	39.8 ± 7.89	0.71 ± 0.08	0.64 ± 0.1	40.4 ± 4.83	0.71 ± 0.06	0.63 ± 0.04	39.9 ± 6.6
CMAES	0.73 ± 0.04	0.45 ± 0.02	77.0 ± 3.39	0.74 ± 0.01	0.45 ± 0.03	76.0 ± 3.81	0.74 ± 0.02	0.44 ± 0.03	76.0 ± 3.54
DNARL	0.83 ± 0.21	0.80 ± 0.06	35.42 ± 2.99	0.83 ± 0.01	0.81 ± 0.01	28.8 ± 1.93	0.82 ± 0.01	0.81 ± 0.01	18.7 ± 3.21
TACO	**0.91** ± 0.01	**0.87** ± 0.02	**133.8** ± 4.27	0.9 ± 0.01	0.86 ± 0.01	**135.0** ± 2.12	**0.92** ± 0.0	**0.88** ± 0.01	**137.4** ± 1.14

Table 3: Performance comparison of different algorithms on human enhancer datasets across three different cell lines.

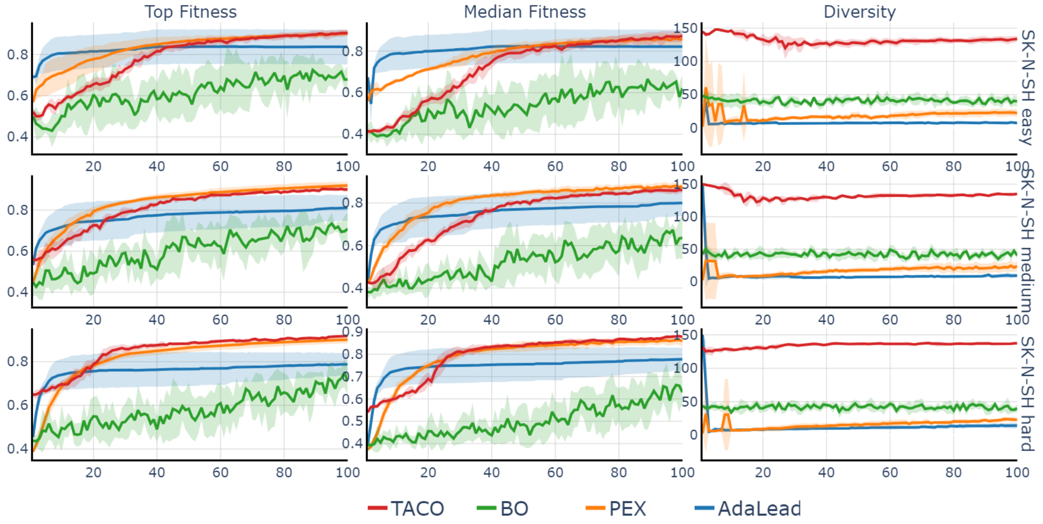

Figure 4: **Evaluation metric by optimization round** for TACO, BO, PEX and Adalead. Shaded regions indicate the standard deviation of 5 runs. The x-axis indicates the number of rounds.

Evaluation by Optimization Round. As shown in Fig. 5, we present the evaluation results after each round of optimization. We observe that AdaLead, a greedy-based algorithm, quickly finds relatively high-fitness sequences at the initial stages. However, its diversity drops rapidly, causing the fitness to plateau and get stuck in local optima. In contrast, PEX demonstrates a steady increase in fitness, but it consistently maintains a low diversity throughout. Only TACO not only achieves a

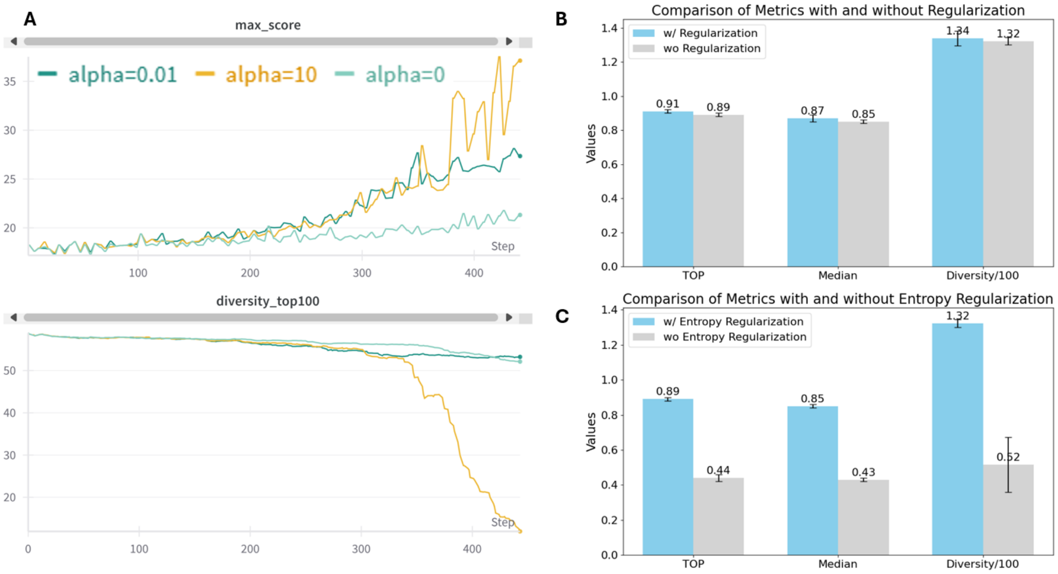

Figure 5: **Ablation Study**. (A) Impact of the TFBS reward. Increasing α leads to higher fitness values but at the cost of reduced diversity.The fitness values presented here are not normalized. (B) Effect of using hill-climb replay buffer (storing past high-fitness experiences). The use of the replay buffer significantly improves maximum fitness values. (C) Effect of entropy regularization. Entropy regularization encourages the exploration of less probable actions.

stable increase in fitness but also maintains high diversity due to its AR model finetuning paradigm, which effectively balances fitness and diversity throughout the optimization process.

4.3 ABLATION STUDY

TFBS Reward. We validate the effect of r_{TFBS} on the complete yeast complex dataset. As shown in Fig. 5(A), applying r_{TFBS} ($\alpha > 0$) allows the policy to explore higher fitness regions. At $\alpha = 0.01$, the optimized fitness shows a noticeable improvement compared to optimization without domain knowledge, without compromising sequence diversity. As α increases further to 10, we observe even higher fitness values but at the expense of a significant drop in diversity. Given that these results have already reached the fitness limits of the yeast complex, further wet-lab experiments are required to better understand the role of TFBS rewards in the RL process.

RL Design Choices. We evaluate two main components of our RL framework: the hill-climb replay buffer and entropy regularization. First, we test the effect of the hill-climb replay buffer, which stores past experiences with high fitness values (Fig. 5(B)). We find that incorporating a replay buffer significantly enhances the maximum fitness values explored, consistent with observations from prior studies (Lee et al., 2024; Ghugare et al., 2024). Next, we test the impact of entropy regularization (Fig. 5(C)) and find it effective in encouraging exploration of less probable actions, leading to improved diversity.

5 CONCLUSION

Designing CREs is a highly impactful task, and the increasing availability of fitness data makes it increasingly feasible. Current methods often rely on basic optimization strategies such as genetic algorithms and directed evolution, which, while effective, lack the ability to leverage advanced optimization techniques. To address this limitation, we propose TACO, an RL-based approach that fine-tunes an AR generative model, achieving both high fitness and diversity in CRE design. By incorporating TFBS domain knowledge, TACO offers a promising direction for further advancements in machine-learning-guided CRE optimization.

REFERENCES

Weizhi An, Yuzhi Guo, Yatao Bian, Hehuan Ma, Jinyu Yang, Chunyuan Li, and Junzhou Huang. Modna: motif-oriented pre-training for dna language model. In *Proceedings of the 13th ACM international conference on bioinformatics, computational biology and health informatics*, pp. 1–5, 2022.

Namrata Anand and Tudor Achim. Protein structure and sequence generation with equivariant denoising diffusion probabilistic models. *arXiv preprint arXiv:2205.15019*, 2022.

Christof Angermueller, David Dohan, David Belanger, Ramya Deshpande, Kevin Murphy, and Lucy Colwell. Model-based reinforcement learning for biological sequence design. In *International conference on learning representations*, 2019.

Pavel Avdeyev, Chenlai Shi, Yuhao Tan, Kseniia Dudnyk, and Jian Zhou. Dirichlet diffusion score model for biological sequence generation. In *International Conference on Machine Learning*, pp. 1276–1301. PMLR, 2023a.

Pavel Avdeyev, Chenlai Shi, Yuhao Tan, Kseniia Dudnyk, and Jian Zhou. Dirichlet diffusion score model for biological sequence generation. In *International Conference on Machine Learning*, pp. 1276–1301. PMLR, 2023b.

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.

Timothy L Bailey, James Johnson, Charles E Grant, and William S Noble. The meme suite. *Nucleic acids research*, 43(W1):W39–W49, 2015.

Thomas Blaschke, Josep Arús-Pous, Hongming Chen, Christian Margreitter, Christian Tyrchan, Ola Engkvist, Kostas Papadopoulos, and Atanas Patronov. Reinvent 2.0: an ai tool for de novo drug design. *Journal of chemical information and modeling*, 60(12):5918–5922, 2020.

Shannon E Boye, Sanford L Boye, Alfred S Lewin, and William W Hauswirth. A comprehensive review of retinal gene therapy. *Molecular therapy*, 21(3):509–519, 2013.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

Binghong Chen, Tianzhe Wang, Chengtao Li, Hanjun Dai, and Le Song. Molecule optimization by explainable evolution. In *International conference on learning representation (ICLR)*, 2021.

Francis S Collins and Harold Varmus. A new initiative on precision medicine. *New England journal of medicine*, 372(9):793–795, 2015.

Lucas Ferreira DaSilva, Simon Senan, Zain Munir Patel, Aniketh Janardhan Reddy, Sameer Gabbita, Zach Nussbaum, César Miguel Valdez Córdova, Aaron Wenteler, Noah Weber, Tin M Tunjic, et al. Dna-diffusion: Leveraging generative models for controlling chromatin accessibility and gene expression via synthetic regulatory elements. *bioRxiv*, 2024.

Bernardo P de Almeida, Christoph Schaub, Michaela Pagani, Stefano Secchia, Eileen EM Furlong, and Alexander Stark. Targeted design of synthetic enhancers for selected tissues in the drosophila embryo. *Nature*, 626(7997):207–211, 2024.

Carl G de Boer, Eeshit Dhaval Vaishnav, Ronen Sadeh, Esteban Luis Abeyta, Nir Friedman, and Aviv Regev. Deciphering eukaryotic gene-regulatory logic with 100 million random promoters. *Nature biotechnology*, 38(1):56–65, 2020.

Oriol Fornes, Jaime A Castro-Mondragon, Aziz Khan, Robin Van der Lee, Xi Zhang, Phillip A Richmond, Bhavi P Modi, Solenne Correard, Marius Gheorghe, Damir Baranašić, et al. Jaspar 2020: update of the open-access database of transcription factor binding profiles. *Nucleic acids research*, 48(D1):D87–D92, 2020.

Caixia Gao. The future of crispr technologies in agriculture. *Nature Reviews Molecular Cell Biology*, 19(5):275–276, 2018.

Zijie Geng, Shufang Xie, Yingce Xia, Lijun Wu, Tao Qin, Jie Wang, Yongdong Zhang, Feng Wu, and Tie-Yan Liu. De novo molecular generation via connection-aware motif mining. In *The Eleventh International Conference on Learning Representations*.

Raj Ghugare, Santiago Miret, Adriana Hugessen, Mariano Phielipp, and Glen Berseth. Searching for high-value molecules using reinforcement learning and transformers. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=nqlymMx42E`.

Sager J Gosai, Rodrigo I Castro, Natalia Fuentes, John C Butts, Susan Kales, Ramil R Noche, Kousuke Mouri, Pardis C Sabeti, Steven K Reilly, and Ryan Tewhey. Machine-guided design of synthetic cell type-specific cis-regulatory elements. *bioRxiv*, 2023.

Nikolaus Hansen. Covariance matrix adaptation evolution strategy (cma-es). Technical report, Technical report.

Sven Heinz, Casey E Romanoski, Christopher Benner, and Christopher K Glass. The selection and function of cell type-specific enhancers. *Nature reviews Molecular cell biology*, 16(3):144–154, 2015.

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

Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, Jarrid Rector-Brooks, Bonaventure FP Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, Dinghuai Zhang, et al. Biological sequence design with gflownets. In *International Conference on Machine Learning*, pp. 9786–9801. PMLR, 2022.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Multi-objective molecule generation using interpretable substructures. In *International conference on machine learning*, pp. 4849–4859. PMLR, 2020.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

Andrew Kirjner, Jason Yim, Raman Samusevich, Shahar Bracha, Tommi S Jaakkola, Regina Barzilay, and Ila R Fiete. Improving protein optimization with smoothed fitness landscapes. In *The Twelfth International Conference on Learning Representations*, 2023.

Avantika Lal, David Garfield, Tommaso Biancalani, and Gokcen Eraslan. reglm: Designing realistic regulatory dna with autoregressive language models. In *International Conference on Research in Computational Molecular Biology*, pp. 332–335. Springer, 2024.

Minji Lee, Luiz Felipe Vecchietti, Hyunkyu Jung, Hyun Joo Ro, Meeyoung Cha, and Ho Min Kim. Robust optimization in protein fitness landscapes using reinforcement learning in latent space. In *Forty-first International Conference on Machine Learning*, 2024. URL `https://openreview.net/forum?id=0zbxwvJqwf`.

Siyuan Li, Zedong Wang, Zicheng Liu, Di Wu, Cheng Tan, Jiangbin Zheng, Yufei Huang, and Stan Z. Li. VQDNA: Unleashing the power of vector quantization for multi-species genomic sequence modeling. In *Forty-first International Conference on Machine Learning*, 2024a. URL `https://openreview.net/forum?id=BOunbuapcv`.

Zehui Li, Yuhao Ni, William AV Beardall, Guoxuan Xia, Akashaditya Das, Guy-Bart Stan, and Yiren Zhao. Discdiff: Latent diffusion model for dna sequence generation. *arXiv preprint arXiv:2402.06079*, 2024b.

Johannes Linder and Georg Seelig. Fast activation maximization for molecular sequence design. *BMC bioinformatics*, 22:1–20, 2021.

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. Statistical rejection sampling improves preference optimization. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=xbjSwwrQOe`.

Scott Lundberg. A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*, 2017.

Vincent Mallet and Jean-Philippe Vert. Reverse-complement equivariant networks for dna sequences. *Advances in neural information processing systems*, 34:13511–13523, 2021.

Eric Nguyen, Michael Poli, Matthew G Durrant, Armin W Thomas, Brian Kang, Jeremy Sullivan, Madelena Y Ng, Ashley Lewis, Aman Patel, Aaron Lou, et al. Sequence modeling and design from molecular to genome scale with evo. *BioRxiv*, pp. 2024–02, 2024a.

Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Michael Wornow, Callum Birch-Sykes, Stefano Massaroli, Aman Patel, Clayton Rabideau, Yoshua Bengio, et al. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. *Advances in neural information processing systems*, 36, 2024b.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Anirban Sarkar, Ziqi Tang, Chris Zhao, and Peter Koo. Designing dna with tunable regulatory activity using discrete diffusion. *bioRxiv*, pp. 2024–05, 2024.

Yair Schiff, Chia Hsiang Kao, Aaron Gokaslan, Tri Dao, Albert Gu, and Volodymyr Kuleshov. Caduceus: Bi-directional equivariant long-range DNA sequence modeling. In *ICML 2024 Workshop on Efficient and Accessible Foundation Models for Biological Discovery*, 2024. URL `https://openreview.net/forum?id=aWSki2rtiA`.

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

Jiawei Shao, Xinyuan Qiu, Lihang Zhang, Shichao Li, Shuai Xue, Yaqing Si, Yilin Li, Jian Jiang, Yuhang Wu, Qiqi Xiong, et al. Multi-layered computational gene networks by engineered tristate logics. *Cell*, 2024.

Eilon Sharon, Yael Kalma, Ayala Sharp, Tali Raveh-Sadka, Michal Levo, Danny Zeevi, Leeat Keren, Zohar Yakhini, Adina Weinberger, and Eran Segal. Inferring gene regulatory logic from high-throughput measurements of thousands of systematically designed promoters. *Nature biotechnology*, 30(6):521–530, 2012.

Sam Sinai, Richard Wang, Alexander Whatley, Stewart Slocum, Elina Locane, and Eric D Kelsic. Adalead: A simple and robust adaptive greedy search algorithm for sequence design. *arXiv preprint arXiv:2010.02141*, 2020.

Ibrahim I Taskiran, Katina I Spanier, Hannah Dickmänken, Niklas Kempynck, Alexandra Pančíková, Eren Can Ekşi, Gert Hulselmans, Joy N Ismail, Koen Theunis, Roel Vandepoel, et al. Cell-type-directed design of synthetic enhancers. *Nature*, 626(7997):212–220, 2024.

Masatoshi Uehara, Yulai Zhao, Ehsan Hajiramezanali, Gabriele Scalia, Gökcen Eraslan, Avantika Lal, Sergey Levine, and Tommaso Biancalani. Bridging model-based optimization and generative modeling via conservative fine-tuning of diffusion models. *arXiv preprint arXiv:2405.19673*, 2024.

Eeshit Dhaval Vaishnav, Carl G de Boer, Jennifer Molinet, Moran Yassour, Lin Fan, Xian Adiconis, Dawn A Thompson, Joshua Z Levin, Francisco A Cubillos, and Aviv Regev. The evolution, evolvability and engineering of gene regulatory dna. *Nature*, 603(7901):455–463, 2022.

Peter JM Van Laarhoven, Emile HL Aarts, Peter JM van Laarhoven, and Emile HL Aarts. *Simulated annealing*. Springer, 1987.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Patricia J Wittkopp and Gizem Kalay. Cis-regulatory elements: molecular mechanisms and evolutionary processes underlying divergence. *Nature Reviews Genetics*, 13(1):59–69, 2012.

Xi Zeng, Xiaotian Hao, Hongyao Tang, Zhentao Tang, Shaoqing Jiao, Dazhi Lu, and Jiajie Peng. Designing biological sequences without prior knowledge using evolutionary reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 383–391, 2024.

Pengcheng Zhang, Haochen Wang, Hanwen Xu, Lei Wei, Liyang Liu, Zhirui Hu, and Xiaowo Wang. Deep flanking sequence engineering for efficient promoter design using deepseed. *Nature communications*, 14(1):6309, 2023.

Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Chee-Kong Lee. Motif-based graph self-supervised learning for molecular property prediction. *Advances in Neural Information Processing Systems*, 34:15870–15882, 2021.

Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana V Davuluri, and Han Liu. DNABERT-2: Efficient foundation model and benchmark for multi-species genomes. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=oMLQB4EZE1`.

Joseph M Zullo, Derek Drake, Liviu Aron, Patrick O'Hern, Sameer C Dhamne, Noah Davidsohn, Chai-An Mao, William H Klein, Alexander Rotenberg, David A Bennett, et al. Regulation of lifespan by neural excitation and rest. *Nature*, 574(7778):359–364, 2019.

APPENDIX

A PRELIMINARY ON CREs

What are CREs? CREs are non-coding DNA sequences that regulate the expression of nearby genes by modulating the binding of TFs and RNA polymerase. The two main types of CREs are promoters, which initiate and maintain mRNA transcription, and enhancers, which are distal elements that interact with promoters to increase gene expression. CREs play a crucial role in establishing specific gene expression profiles across different cell types, influencing cellular identity and function.

Why are CREs cell-type specific? The cell-type specificity of CREs arises from differential TF binding. TF binding is influenced by several factors, including DNA sequence composition, local chromatin structure, and interactions with other proteins and cofactors. Human cells express around 1,500 to 2,000 different TFs, and their expression patterns vary across cell types. Each cell type thus has a unique set of active CREs that drive the expression of genes necessary for its specific functions. For example, a CRE active in liver cells (hepatocytes) might bind liver-specific TFs such as HNF4A, whereas in neurons, the same CRE might be inactive due to the absence of these TFs.

How are designed CREs utilized? Designed CREs can be used in both *in-vivo* and *in-vitro* settings depending on the application.*In-vivo*, CREs are often delivered using viral vectors, such as adenoviruses or adeno-associated viruses (AAVs), which facilitate the incorporation of synthetic CREs into the target cell's genome. This method is particularly useful for gene therapy, where precise control over gene expression is crucial for therapeutic efficacy and safety. *In-vitro*, CREs are typically introduced into cultured cells using plasmids or CRISPR-based methods, allowing researchers to test the functionality and regulatory impact of the synthetic CREs under controlled conditions. This approach is invaluable for high-throughput screening of CRE designs and optimization of regulatory elements before moving to *in-vivo* applications.

Applications and Future Prospects. Designing synthetic CREs with precise, cell-type-specific regulatory functions has significant potential in both basic research and therapeutic applications. In gene therapy, cell-type-specific CREs can be used to target therapeutic gene expression to specific tissues, minimizing off-target effects and toxicity. In industrial biotechnology, engineered CREs can optimize protein production in desired cell lines. Recent advances in deep learning and generative models have shown promise in predicting and generating CREs with desired regulatory profiles, opening new avenues for programmable gene regulation.

B DETAILS OF DATASETS

Existing CRE fitness datasets are generated through Massively Parallel Reporter Assays (MPRAs), which allow for high-throughput measurements of regulatory sequences in in vitro settings. The yeast promoter dataset includes results from two different media conditions: *complex* and *defined*. The human enhancer dataset, on the other hand, consists of data from three distinct human cell lines: HepG2 (a liver cell line), K562 (an erythrocyte cell line), and SK-N-SH (a neuroblastoma cell line).

We adopt the dataset splits proposed by RegLM (Lal et al., 2024) and use their defined training set as our full dataset, denoted as $\mathcal{D}$. To simulate a progression from low-fitness to high-fitness sequences, we further partition $\mathcal{D}$ into a subset $\mathcal{D}^*$ for finetuning and evaluation. Each dataset represents a cell-type-specific scenario due to distinct TF effect vocabularies and regulatory landscapes.

Our partitioning scheme follows the same approach as RegLM. Specifically, we define three difficulty levels—*hard*, *medium*, and *easy*—based on fitness percentiles of 20-40, 40-60, and 60-80, respectively, in both media conditions for the yeast dataset. Since yeast is a single-cell organism, we ensure that the fitness levels are consistent across both media. For the human enhancer datasets, we define the *hard* fitness range as values below 0.2, the *medium* range as values between 0.2 and 0.75, and the *easy* range as values between 0.75 and 2.5. These ranges are selected to maintain fitness values below 0.2 in other cell lines, thereby simulating a cell-type-specific regulatory scenario.

C ENFORMER SERVES AS ORACLE

Enformer (Avsec et al., 2021) is a hybrid architecture that combines CNNs and Transformers, achieving state-of-the-art (SOTA) performance across a range of DNA regulatory prediction tasks. In our study, all CRE fitness prediction oracles are based on the Enformer architecture (Lal et al., 2024; Uehara et al., 2024). The primary distinction lies in the output: while the original Enformer model predicts 5,313 human chromatin profiles, we modify it to predict a single scalar value representing CRE fitness.

The oracle model for the human enhancer datasets retains the same number of parameters as the original Enformer. In contrast, for the yeast promoter datasets, we reduce the model size due to the simpler nature of yeast promoter sequences. Specific architectural configurations are listed in Tab. 5. In this study, we directly utilize the oracle weights provided by RegLM (Lal et al., 2024) for consistency.

Model	**Dimension**	**Depth**	**Number of Downsamples**
Human Enhancer	1536	11	7
Yeast Promoter	384	1	3

Table 5: Oracle model parameters for human and yeast datasets.

D DETAILS OF AR GENERATIVE MODELS

Over the past year, there has been significant growth in the development of DNA language models, with many new models emerging. However, most of these models, such as Caduceus (Schiff et al., 2024), DNABert2 (Zhou et al., 2024), and VQDNA (Li et al., 2024a), are based on BERT-style pretraining and lack the capability to generate DNA sequences. Among them, HyenaDNA (Nguyen et al., 2024b) is the only GPT-style DNA language model. Unlike traditional Transformer-based architectures, HyenaDNA leverages a state space model (SSM), which provides linear computational complexity, making it suitable for handling long DNA sequences with complex dependencies.

Subsequent work based on HyenaDNA, such as Evo (Nguyen et al., 2024a), has demonstrated the powerful DNA sequence generation capabilities of this architecture. Additionally, RegLM (Lal et al., 2024) has explored conditional DNA generation by employing a prefix-tuning strategy, where a customized token is used as the prefix of the DNA sequence to guide the subsequent generation process. This approach has enabled RegLM to effectively model context-dependent DNA sequence generation.

E TFBS SCAN AND FREQUENCY FEATURE PREPROCESSING

The Jaspar database (Fornes et al., 2020) provides detailed annotations of TFBSs. Each TFBS t_i corresponds to a transcription factor that binds to it, regulating gene expression. Instead of representing t_i as a fixed sequence, it is described by a position frequency matrix $\mathbf{M}_i \in \mathbb{R}^{L_i \times 4}$, where L_i is the length of the TFBS, and the four columns correspond to the nucleotides $\{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}$. The matrix encodes the likelihood of each nucleotide appearing at each position in the TFBS, making it possible to capture variations in TF binding.

We utilize FIMO (Find Individual Motif Occurrences) (Bailey et al., 2015) to scan each sequence for potential TFBSs. Given a sequence x and a matrix $\mathbf{M}_i$, FIMO evaluates each subsequence x_j in x by calculating a probabilistic score:

$$\mathrm{score}(x_j, \mathbf{M}_i) = \prod_{k=1}^{L_i} P(n_k \mid \mathbf{M}_i[k]), \tag{7}$$

where $P(n_k \mid \mathbf{M}_i[k])$ represents the probability of nucleotide n_k occurring at position k in the matrix $\mathbf{M}_i$. FIMO identifies the subsequences with the highest scores as potential occurrences of the TFBS.

For each sequence x, FIMO outputs a frequency feature vector $\mathbf{h}(x) = [\mathbf{h}_1(x), \mathbf{h}_2(x), \ldots, \mathbf{h}_n(x)]$, where $\mathbf{h}_i(x)$ denotes the frequency of the i-th TFBS in sequence x. This frequency feature vector is then used as input for the downstream prediction model. The use of frequency-based features, as opposed to binary indicators, captures the varying levels of TFBS occurrences in the sequence, allowing for a more nuanced understanding of the regulatory role of each TFBS. Given this tabular representation, we employ LightGBM (Ke et al., 2017), a tree-based model known for its interpretability and effectiveness on tabular datasets, to predict the fitness values of sequences.

F DETAILS OF LIGHTGBM

We utilized LightGBM (Ke et al., 2017) to train models that directly predict CRE fitness based on TFBS frequency features, enabling us to infer the cell type-specific roles of individual TFBSs. For each dataset, we independently trained a LightGBM regression model. The specific parameters used in our model are listed in Table 6.

Parameter	**Value**
Objective	Regression
Metric	MAE
Boosting Type	GBDT
Number of Leaves	63
Learning Rate	0.05
Feature Fraction	0.7
Seed	Random State

Table 6: Hyperparameters used for training the LightGBM regression model.

Metric	**yeast**		**human**		
	complex	**defined**	**hepg2**	**k562**	**sknsh**
MAE	0.63	0.65	0.65	0.65	0.66
RMSE	0.63	0.64	0.56	0.57	0.58

Table 7: Ablation study comparing different metrics on CRE fitness prediction for yeast and human datasets.

We experimented with various metrics corresponding to the distance metric d in Equation equation 4, specifically testing `rmse` and `mae` as well as different learning rates $\{0.01, 0.05\}$ and number of leaves $\{31, 63\}$. Our results indicate that only the metric has a significant impact on the final performance. The ablation results are summarized in Table 7.

$$d_{\text{MAE}} = \frac{1}{n}\sum_{i=1}^{n}\left|f(x_i) - \hat{f}(h(x_i);\theta)\right| \tag{8}$$

$$d_{\text{RMSE}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\Big(f(x_i) - \hat{f}(h(x_i);\theta)\Big)^2} \tag{9}$$

Our experiments demonstrate that the MAE metric yields better performance across all cell types, as shown in Table 7. Therefore, we selected MAE as the final evaluation metric.

G DNA SEQUENCE PLAUSIBILITY

Unlike molecules and proteins (Uehara et al., 2024), which inherently possess well-defined physical and chemical properties, DNA sequences lack such structural constraints. For example, molecular structures are subject to physical properties like bond angles and energy states, while protein sequences are evaluated based on their 3D folding stability and interactions, making it straightforward to filter out physically implausible designs. Therefore, in molecule and protein design, oracle-predicted fitness is often supplemented with physical property constraints to ensure the plausibility of generated candidates. This helps exclude a significant number of physically infeasible structures, enhancing the relevance of the optimization process.

However, DNA sequences pose a unique challenge in this regard. Unlike molecules or proteins, DNA's plausibility cannot be easily assessed through physical properties, as its functional attributes are primarily determined by its interaction with transcription factors and other regulatory proteins in a context-specific manner. Furthermore, current MPRA (massively parallel reporter assay) datasets are typically generated from random sequences, meaning there is no inherent concept of ”plausibility” in the data itself. Consequently, the lack of well-defined constraints in DNA sequences makes it difficult to develop a robust metric for evaluating their plausibility.

Our observations further highlight this challenge. In our experiments, we found that the novelty values of generated DNA sequences were disproportionately high compared to the initial low-fitness sequences, making the novelty metric less informative. This behavior suggests that DNA sequences tend to diverge significantly from their starting points during optimization, regardless of their biological relevance or plausibility. Due to these limitations, we exclude the *Novelty* metric and instead focus on evaluating the generated sequences using *Fitness* and *Diversity* metrics, which better capture the optimization objectives for CRE design.

H LIMITATIONS

Our ultimate goal is to optimize CREs with higher fitness values than those currently observed. However, the reliability of such optimized CREs is limited by the fact that our oracles are trained on existing real-world datasets. As a result, predictions for CREs with fitness values beyond the training data range may be less accurate. Currently, our primary *in-silico* experiments simulate an optimization setting that starts from low-fitness CREs, following the strategy proposed in (Lee et al., 2024). Previous studies, such as Vaishnav et al. (2022); de Almeida et al. (2024), have successfully designed CREs using simple optimization methods and validated them *in vivo*, demonstrating high fitness and cell-type specificity in real-world scenarios. Our work serves as a complementary effort to these studies by providing advanced algorithmic strategies for CRE optimization. In the future, we hope to conduct *in vivo* experiments to validate the performance of more sophisticated CRE optimization algorithms.