Designing DNA With Tunable Regulatory Activity Using Score-Entropy Discrete Diffusion

Anirban Sarkar[1*] **Yijie Kang**[1,3] **Nirali Somia**[1] **Pablo Mantilla Puccetti**[1,2] **Jessica Zhou**[1]
Masayuki Nagai[1,2] **Ziqi Tang**[4] **Chris Zhao**[2] **Peter K Koo**[1*]
[1]Simons Center for Quantitative Biology, Cold Spring Harbor Laboratory, Cold Spring Harbor, NY, USA
[2]School of Biological Sciences, Cold Spring Harbor Laboratory, Cold Spring Harbor, NY, USA
[3]Graduate Program in Genetics, Stony Brook University, Stony Brook, NY, USA
[4]InstaDeep, Cambridge, MA, USA
{asarkar, koo}@cshl.edu

Abstract

Designing regulatory DNA sequences with precise, cell-type-specific activity is critical for applications in medicine and biotechnology, but remains challenging due to the vast combinatorial space and complex regulatory grammar governing gene expression. Recent deep generative models—including genomic language models and diffusion-based approaches—offer new tools for sequence design, yet lack systematic evaluation frameworks to assess the biological and functional fidelity of generated sequences. Here, we introduce a comprehensive computational framework for evaluating generated sequences based on their functional activity, sequence similarity, and regulatory motif composition relative to natural regulatory DNA. We further present DNA Discrete Diffusion (D3), a score-entropy discrete diffusion model for conditional generation of regulatory sequences. Benchmarking D3 on multiple functional genomics datasets, we find that D3 produces sequences nearly indistinguishable from natural DNA under our evaluation metrics. Unlike previous diffusion models, which often fail to capture the nuanced combinatorial patterns of regulatory elements, D3 effectively recapitulates cell-type-specific activity and motif organization. We also show that D3 learns informative representations even in the absence of conditioning labels, outperforming genomic language models and supervised models trained on naive one-hot encodings. D3 maintains strong performance in low-data regimes and enhances downstream supervised models when its generated sequences are used for data augmentation. Together, our work advances generative design of regulatory DNA and establishes comprehensive evaluation methods to ensure biological fidelity.

1 Introduction

Designing DNA sequences that precisely control gene expression in specific cell types is a central challenge in synthetic biology, gene therapy, and biotechnology. Regulatory DNA elements, such as promoters and enhancers, determine when, where, and how much genes are expressed [1]. The ability to engineer such elements with user-specified activity levels could enable powerful applications in therapeutics and cellular programming. However, this task is complicated by the sparse and context-dependent nature of regulatory grammars [2]. Functional motifs are embedded within large stretches of non-informative sequence, and identical sequences can yield different regulatory outputs depending on cell type and chromatin context. These properties make it difficult to model or design regulatory sequences using conventional machine learning approaches.

*Corresponding author: asarkar@cshl.edu and koo@cshl.edu

Preprint. Under review.

Recent advances in generative modeling, including diffusion models and genomic language models, offer promising tools for biological sequence generation. Yet these methods often fall short when applied to regulatory DNA [3, 4]. They struggle to capture the sparse and conditional logic underlying enhancer or promoter activity, and typically lack the ability to generate sequences with precise, tunable function across different cell types. Compounding the challenge, robust evaluation of generated regulatory sequences remains difficult: experimental validation is expensive, and existing computational benchmarks are limited in scope and biological realism.

In this work, we introduce DNA Discrete Diffusion (D3), the first application of score-entropy discrete diffusion model [5] for regulatory genomics. D3 generates regulatory DNA sequences directly in discrete nucleotide space by iteratively refining nucleotide transitions conditioned on a target regulatory activity. To evaluate the biological fidelity of generated sequences, we pair D3 with a comprehensive computational framework that benchmarks functional, sequence-level, and compositional similarity to natural regulatory DNA.

Our main contributions are:

- We adapt the first ever score-entropy discrete diffusion for regulatory DNA sequences.
- We develop an evaluation framework to benchmark functional, sequence-level, and compositional properties of synthetic regulatory DNA.
- We show D3 generates sequences nearly indistinguishable from natural sequences, outperforming prior diffusion-based and language model-based approaches.
- We demonstrate that D3 encodes informative regulatory representations even under unconditional training and maintains robust performance in limited data settings.
- We extend D3 for conditional generation using classifier-free guidance, enabling cell-type-specific regulatory sequence design with improved activity specificity.

2 Background and Challenges in Regulatory DNA Modeling

2.1 Regulatory DNA and the Sequence-to-Function Problem

Gene expression in multicellular organisms is orchestrated by regulatory DNA sequences—including promoters, enhancers, and silencers—that encode instructions for when, where, and how much genes are expressed. These non-coding elements control transcription by recruiting transcription factors (TFs), proteins that bind short sequence motifs (typically 4–19 nucleotides) to modulate gene activity [6]. Regulatory sequences function in a highly context-dependent manner: the same DNA sequence can produce different levels of gene expression depending on chromatin accessibility, TF availability, and 3D genome architecture. This unique property means that every cell type projects its specific regulatory program onto a shared genomic sequence, resulting in highly varied regulatory output across cellular contexts.

Unlike protein-coding genes, whose sequences are conserved and whose functions remain relatively stable across cell types and species, regulatory DNA evolves under weaker constraints. Functional motifs are sparsely distributed within large tracts of non-informative sequence, further compounding the modeling challenge. The regulatory code is not only highly heterogeneous but also deeply embedded in a noisy sequence background, making it difficult to decipher the underlying logic. Together, these properties—context-dependence, sparsity, and weak conservation—render the task of understanding and engineering regulatory DNA an open challenge in genomics and synthetic biology.

2.2 Data Landscape for Modeling and Design

Advances in high-throughput experimental assays have enabled large-scale measurements of regulatory DNA function. Massively parallel reporter assays (MPRAs) allow the activity of thousands of designed sequences to be measured in parallel [7]. Variants such as STARR-seq enable genome-wide enhancer activity readouts [8], while lentiMPRA enables integration-based reporter assays by stably incorporating regulatory sequences into the genome, allowing their activity to be measured across diverse cell types and chromatin contexts [9]. In addition, chromatin profiling methods such as CAGE-seq [10] and ATAC-seq [11] provide genome-wide maps of transcription start sites and chromatin accessibility, respectively. These assays form the foundation for learning sequence-to-function

relationships and benchmarking predictive models. However, these datasets can be expensive to produce and often lack sufficient representation across cell types or experimental conditions.

2.3 Computational Approaches to Regulatory DNA Design

Supervised models and oracle-based design. Deep neural networks (DNNs) have become a standard approach for predicting gene expression or chromatin accessibility from DNA sequence [12, 13]. Once trained, these models serve as *in silico* oracles for scoring synthetic sequences, reframing sequence design as an optimization problem [14]. Local search strategies—such as gradient ascent [15], simulated annealing [16], and *in silico* evolution [17]—iteratively refine candidate sequences toward higher predicted activity. Other approaches, including motif implanting [18] and reinforcement learning-based methods such as GFlowNets [19], similarly depend on oracle guidance to explore functional sequence space. While these methods vary in search strategy, they all rely on strong supervision and often get trapped in narrow optima. They also require large labeled datasets and tend to generalize poorly under covariate shifts, limiting their reliability in novel biological contexts [20–23].

Traditional deep generative approaches. Early generative approaches to regulatory sequence design were led by conditional generative adversarial networks (GANs), first introduced by [24] and later extended by [25]. However, these methods have seen limited follow-up and often underperform compared to simpler baselines. While variational autoencoders (VAEs) and normalizing flows have proven successful in other domains [26, 27], they have yet to show compelling results for regulatory DNA.

Genomic language models (gLMs). Recent efforts have proposed genomic language models, adapting masked or causal language modeling tasks to DNA sequences [28–37]. These models show promise for variant interpretation and genome annotation, under simplified benchmarks. However, comprehensive evaluations in the non-coding genome suggest that pre-trained gLMs have limited ability to learn core principles of cis-regulatory grammar [3, 38]. This limitation stems in part from a mismatch between language modeling objectives and the structure of regulatory DNA: both masked and causal language models are trained to predict every nucleotide in the sequence, implicitly treating all positions as equally informative. In regulatory DNA, however, functional information is sparse and highly context-specific—most positions are biologically neutral, while regulatory activity depends on a small number of motifs and their combinatorial interactions. As a result, these models are biased toward modeling low information backgrounds rather than learning the underlying regulatory grammar.

Diffusion models. Diffusion-based generative models have gained interest for biological sequence generation by learning to reverse a progressive noise process applied to training data, enabling the generation of realistic new sequences [39, 40]. In regulatory genomics, diffusion has been applied across various spaces: in continuous space [41], latent space [42], and over the simplex [43]. Each approach introduces unique modeling constraints. Continuous and latent-space models require quantization to map outputs back to discrete DNA, which can distort functional motifs. Simplex-based methods enforce geometric constraints that may not align with the discrete, context-dependent nature of regulatory sequence grammars.

An alternative line of work, LegNet [44], adopts a cold diffusion framework that refines sequences via iterative nucleotide substitutions. However, unlike generative diffusion models that learn to model and reverse a noise distribution over the training data, LegNet optimizes sequences to maximize a supervised predictor. As a result, it acts more like a goal-directed denoising process than a generative model of the underlying sequence distribution.

Despite their promise, existing diffusion approaches for regulatory DNA face core limitations. Models that operate in continuous or latent spaces risk distorting sparse functional motifs during quantization, while simplex-based models may impose geometric constraints misaligned with biological sequence structure. These limitations underscore the need for a generative model that operates directly in discrete nucleotide space, avoids distortion from quantization, and can model functional sequence distributions without requiring labeled data.

3 Our Approach: DNA Discrete Diffusion

To address these challenges, we adapted Score-Entropy Discrete Diffusion (SEDD) [5] to regulatory genomics, resulting in DNA Discrete Diffusion (D3). SEDD is a discrete diffusion framework originally developed for natural language generation. It operates by progressively corrupting a discrete sequence with noise and learning a score function to reverse this process, enabling generation of coherent sequences directly in discrete symbol space. Unlike autoregressive models that generate tokens one-by-one, SEDD denoises the entire sequence in parallel, allowing it to model complex, non-local dependencies. See Appendix B for technical details of the model.

A key feature of SEDD is its concrete score parameterization, which estimates the data distribution via constrained Bregman divergence. This formulation leads to more stable training and sharper generations. Additionally, SEDD optimizes a tractable evidence lower bound (ELBO), providing a stable and principled training objective without relying on high-variance score-function estimators. Its flexible conditioning mechanism supports both inpainting and attribute-guided generation, capabilities well-suited to regulatory DNA design, where one often needs to impose cell-type-specific activity constraints or structural requirements. These properties make D3 particularly effective for modeling regulatory DNA, where functional motifs are sparse and embedded within long stretches of uninformative background. Unlike reconstruction-based models that aim to predict every nucleotide, including those irrelevant to regulation, D3 learns to evolve sequences through selective, context-aware mutations.

By operating directly in discrete nucleotide space, D3 focuses learning on functionally meaningful patterns, refining regulatory motifs while tolerating neutral variation. This generative process is analogous to biological evolution: rather than reconstructing an entire sequence, D3 begins with random sequences and incrementally applies discrete nucleotide substitutions, steering the sequence toward a desired regulatory activity. Inspired by classical models of molecular evolution such as the Jukes–Cantor mutation matrix [45], these substitutions occur uniformly across nucleotides, allowing D3 to explore the sequence space broadly while maintaining tractable, evolution-like dynamics. Functional motifs emerge through the accumulation of beneficial mutations, resembling a form of selection. Because D3 denoises the entire sequence in parallel, it can model dependencies among distant motifs and capture higher-order combinatorial regulatory logic. By learning to evolve sequences under these biologically grounded constraints, D3 offers a principled generative framework for regulatory DNA design, addressing key challenges of class imbalance and context specificity.

4 Conditional Sequence Generation

To demonstrate the effectiveness of D3 in conditionally generating sequences, we benchmark its performance against previous methods using three high-quality experimental datasets: human promoters [46], fly enhancers [47], and cell-type-specific regulatory sequences [16].

4.1 Designing human promoters

Dataset. The human promoter dataset is an established benchmark for DNA generative models [46]. It consists of 100,000 promoter sequences (1024 nt long, centered on transcription start sites) with corresponding transcription initiation signal profiles measured via CAGE-seq [10] from the FANTOM database [48]. These profiles vary across genes and are shaped by underlying promoter sequence features. The modeling task is to generate promoter sequences conditioned on a target transcription initiation signal profile.

Baselines. The benchmark was originally established by DDSM [46] and extended by DFM [43]. It includes baselines established by Bit-diffusion [49] and D3PM [50]. Here we extend this benchmark by employing D3 using the same CNN specs as the original DDSM study for a fair comparison. We also include a transformer architecture that is trained within the D3 and DFM framework (details in Appendix C).

Evaluation. To assess the quality of generated regulatory sequences, we evaluated models along multiple axes of similarity between generated and real sequences, using `Sei` [51] as an oracle to provide *in silico* functional measurements. Generated sequences were conditioned on target activity values taken from held-out test sequences, allowing a matched comparison between generated and real sequences under the same functional specification.

Table 1: Performance of generated human promoter sequences.

Model	Functional similarity		Sequence similarity		Compositional similarity	
	Conditional generation fidelity (MSE ↓)	Predictive distribution shift (KS test statistic ↓)	Percent identity (Average)	k-mer spectrum shift (JS distance ↓)	Motif enrichment (Pearson's r)	Motif co-occurrence (Frobenius norm)
Bit Diffusion* (bit)	0.0414	-	-	-	-	-
BIT Diffusion* (1-hot)	0.0395	-	-	-	-	-
D3PM-Uniform*	0.0375	-	-	-	-	-
DDSM*	0.0334	-	-	-	-	-
DFM-Conv*	0.0269	0.399	7.09	0.346	0.831	23.699
DFM-Tran	0.0318	0.336	6.49	0.216	0.873	12.924
D3-Conv	0.0259	0.093	8.57	0.003	0.997	0.746
D3-Tran	0.0219	0.052	9.06	0.002	0.999	1.487
Shuffle	0.0617	0.300	7.97	0.096	0.891	2.545

We report metrics that quantify functional, sequence-level, and compositional similarity: *conditional generation fidelity*, *predictive distribution shift*, *percent identity*, *k-mer spectrum shift*, *motif enrichment*, and *motif co-occurrence*. *Conditional generation fidelity* measures how closely the oracle-predicted activity of a generated sequence matches the conditioning target according to mean-squared error (MSE). *Predictive distribution shift* compares the overall distribution of oracle-predicted activities between generated and real sequences, using the Kolmogorov–Smirnov test statistic to quantify deviations. *Sequence similarity* is evaluated by percent identity, computed as the normalized Hamming distance between generated and real sequences. This captures the extent of memorization (high identity to training data) versus novelty. We also assess *k-mer spectrum shift* by comparing k-mer frequency distributions (for $k = 6$) between generated and real sequences using Jensen–Shannon divergence. To probe compositional similarity at the motif level, we evaluate *motif enrichment* and *motif co-occurrence*. *Motif enrichment* measures the correlation between motif counts (across 879 known transcription factor motifs) in generated and real sequences, scanning sequences with position weight matrices from the JASPAR database [52] using FIMO [53]. *Motif co-occurrence* quantifies the similarity in pairwise motif co-occurrence patterns across sequences by comparing covariance matrices derived from motif counts. Formal definitions for each metric are in Appendix A.

Results. D3 with a CNN (D3-Conv) achieved lower *conditional generation fidelity* (MSE) compared to the state-of-the-art DFM with a CNN (DFM-Conv) (Table 1). A transformer architecture further improved MSE for D3 but worsened results for DFM. All generative models outperformed dinucleotide shuffled sequences, suggesting they learned useful sequence properties, albeit to varying extents. D3-based models yielded lower *predictive distribution shifts* compared to DFM-based models. All models did not memorize training sequences, indicated by low *percent identity* on par with shuffled sequences. The *k-mer spectrum shift* was substantially lower for D3-based models, while DFM-based models showed higher shift than shuffled sequences, suggesting unnatural sequence biases. Similar trends for *motif enrichment* and *motif co-occurrence* suggest that D3-generated sequences better reflect real sequence composition.

4.2 Context-specific Fly Enhancer Design

Data. The data consists of enhancer sequences with corresponding activity values in two contexts (housekeeping and developmental promoters), taken from the STARR-seq dataset [47]. The dataset contains 402,296 training, 40,570 validation, and 41,186 test sequences. Each 249 nt enhancer sequence has two scalar activity values for the developmental and housekeeping conditions. We repurposed this dataset to train and evaluate generative models for context-specific enhancer activity.

Baselines. We trained DFM-Conv, DFM-Tran, D3-Conv, and D3-Tran with conditioning using activities from the training set. Training details are in Appendix C.

Evaluation. In addition to the previous metrics, we report several complementary metrics. *Fréchet distance* quantifies the alignment between the oracle embedding distributions of generated and real sequences, using the 2-Wasserstein distance under a multivariate Gaussian assumption; lower values indicate closer alignment in embedding space. *Discriminability* measures how easily a classifier can distinguish generated from real sequences, computed as the area under the receiver operating characteristic (AUROC) curve. To assess whether generated sequences encode consistent regulatory

features, we evaluated *attribution consistency*. Attribution maps were computed using GradientSHAP [54], with 5,000 random reference sequences, followed by gradient correction [55]. For each sequence, attribution scores were aggregated across $k = 6$ nucleotide windows and embedded into an analytical geometric representation. Consistency was quantified by the Kullback–Leibler divergence between the observed attribution distribution and an uninformative prior [56], with higher divergence indicating stronger enrichment of consistent patterns. Attribution consistency was computed separately for real and generated sequences, as well as jointly by concatenating both sets. Formal definitions of all metrics and implementation details are provided in Appendix A.

We used a DeepSTARR model trained with EvoAug [57] as the oracle for this evaluation, as it achieves state-of-the-art performance on the DeepSTARR dataset. Generated sequences were conditioned on the oracle-predicted activities of held-out test sequences, producing a matched set of generated and real sequences for comparison.

Results. D3-based models outperformed DFM-based models on evaluations for functional, sequence-level, and compositional similarity (Table 2). D3-Tran achieved the lowest MSE (*conditional generation fidelity*) and KS test statistic (*predictive distribution shift*), while D3-Conv had a slightly lower Frechet distance (*representational similarity*). According to *percent identity*, no model memorized training sequences. D3-based models yielded lower *k-mer spectrum shift* and *discriminatability* compared to DFM-based models, suggesting DFM-generated sequences contain biases easily recognized by a discriminator. Compositionally, D3-generated sequences contained *motif enrichment* and *motif combinations* at similar levels as real genomic sequences. Moreover, D3-generated sequences produced attribution maps visually similar to real sequences, with short, consistent motif-like patterns. In contrast, DFM-generated sequences exhibited spurious attribution scores for arbitrary positions (Fig. 3, Appendix D). The *attribution consistency* score showed that D3-Tran captures patterns more consistently within its own set of generated sequences and in combinations with the real sequences. According to the *added information* evaluation (see Appendix A), a supervised DeepSTARR improved performance when trained with a combination of real sequences and D3-generated sequences (Figs. 4, Appendix D), with further improvements when augmenting the training data two-fold (Figs. 5, Appendix D) and remarkably, training DeepSTARR using only D3-Tran-generated sequences achieved nearly the same performance as training on the actual set (Figs. 5e, Appendix D), suggesting D3-Tran samples informative sequences (beyond the observed training sequences), providing additional information that benefits supervised modeling.

To ensure broad sampling, we also considered the *sequence diversity* and *mechanistic diversity* of sequences generated using a fixed conditioning value (see Appendix A). We found that generated sequences exhibit a diverse set of sequences (Table 4, Appendix D) represented by a diverse set of cis-regulatory mechanisms (Fig. 6, Appendix D). Finally, we assessed the ability for *task-specific design* by generating sequences conditioned with specific activity settings across tasks. Strikingly, D3-Tran-generated sequences better reflected the task conditional values compared to DFM-Conv

Table 2: Performance for fly enhancers. See Sec. A for details of each metric. Red indicates best performance.

	Functional similarity			**Sequence similarity**		
Model	Conditional generation fidelity (MSE ↓)	Predictive distribution shift (KS test statistic ↓)	Representational similarity (Frechet distance ↓)	Percent identity (Average)	k-mer spectrum shift (JS distance ↓)	Discriminatability (AUROC ↑)
DFM-Conv	2.029	0.231	1.109	38.9 ± 0.01	0.303	0.979 ± 0.014
DFM-Tran	2.182	0.212	1.059	40.4 ± 0.03	0.195	0.982 ± 0.019
D3-Conv	1.192	0.025	0.022	40.3 ± 0.02	0.004	0.516 ± 0.017
D3-Tran	1.156	0.020	0.033	40.4 ± 0.02	0.004	0.552 ± 0.034
Shuffle	1.708	0.184	0.803	40.0 ± 0.02	0.085	0.970 ± 0.003

	Compositional similarity metrics			
Model	Motif enrichment (Pearson's r ↑)	Motif co-occurrence (Frobenius norm ↓)	Attribution consistency (KLD ↑)	Attribution consistency (gen+real sequences) (KLD ↑)
DFM-Conv	0.450	100.5	0.485	0.459
DFM-Tran	0.656	104.5	0.438	0.450
D3-Conv	0.977	36.4	0.504	0.503
D3-Tran	0.984	19.3	0.525	0.514
Shuffle	0.743	81.0	0.084	0.376

(Fig. 7, Appendix D), demonstrating that D3-Tran has learned a meaningful density over sequence space, enabling the ability to sample unseen sequences with desired activity levels across tasks.

We also benchmarked D3 on a dataset of human cis-regulatory sequences with cell-type-specific activity measured via lentiMPRA in three human cell lines—K562 (erythroid precursors), HepG2 (hepatocytes), and SK-N-SH (neuroblastoma)—as described in Ref. [16]. Similarly, D3 consistently outperformed DFM across all cell types. Full details can be found in Appendix E.

5 Generating Cell-Type-Specific Regulatory Sequences

Designing enhancer sequences that drive expression in specific cell types is a central goal in synthetic biology, drug discovery, and gene therapy. Although D3 has demonstrated promise in conditional generation tasks, previous evaluations have focused on a limited number of conditions. To more broadly assess its capabilities, we benchmark D3 on its ability to generate cell-type-specific enhancers using the fly brain enhancer dataset from Ref. [58], which spans a diverse range of regulatory contexts.

This dataset contains 104,000 500-bp enhancer sequences, each annotated with binary activity labels across 81 brain cell types inferred from single-nucleus ATAC-seq [59]. The goal is to generate sequences that are active in a given cell type, capturing the specificity of cis-regulatory programs across a high-dimensional categorical space. We trained D3 on this dataset using the same 20-layer 1D convolutional architecture as DFM [43], with a batch size of 128 and early stopping based on validation performance using Fréchet Biological Distance (FBD).

Unlike previous tasks with scalar conditioning, this task involves categorical conditioning based on discrete cell-type labels. To handle this, we apply Classifier-Free Guidance (CFG), in which D3 is trained both with and without conditioning labels, enabling it to interpolate between conditional and unconditional score estimates during sampling. At inference time, we generate sequences using the adjusted score function:

$$s_\theta^{CFG}(x, t, y) = \gamma s_\theta(x, t, y) + (1 - \gamma) s_\theta(x, t, \phi),$$

where $\gamma > 1$ controls the strength of guidance. This approach allows us to modulate conditioning strength and improve the quality of cell-type-specific sequence generation.

To evaluate generation quality, we use Fréchet Biological Distance (FBD) [43], which compares real and generated sequences in the embedding space of a pretrained enhancer classifier. Lower FBD values indicate closer alignment between synthetic and natural sequences. We compute FBD between 10.4k test sequences and synthetic sequences generated under matching cell-type conditions.

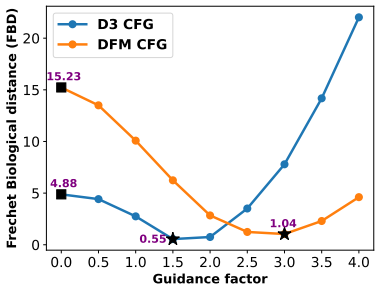

Figure 1: Performance comparison of DFM and D3 for cell type conditional enhancer design with varying CFG levels where cell types are sampled based on empirical frequency. ■ - Unconditional enhancer generation (for guidance factor 0), ⋆ - Best CFG generation.

As shown in Fig. 1, D3 achieves substantially lower FBD than DFM across both unconditional and conditional generation settings. Without CFG, D3 yields a substantially lower FBD (4.88) compared to DFM (15.23). With CFG, D3's performance further improves, reaching its best FBD (0.55) at guidance factor $\gamma = 1.5$, compared to DFM's best of 1.0 at $\gamma = 3$.

We also evaluated D3's ability to generate sequences conditioned on specific cell types by measuring how accurately the generated sequences are classified using the pretrained classifier from Stark et al (2024). We follow their setup by focusing on three representative cell types (IDs: 2, 68, and 16) and assess performance across different guidance factors. D3 consistently achieves lower FBD and higher classification probabilities than DFM (Figure 11, Appendix G), demonstrating its effectiveness in both unguided and guided generation of cell-type-specific sequences.

6 Representational Informativeness: D3 vs. genomic Language Models

Recent advances in genomic language models (gLMs) have raised expectations that unsupervised pretraining on raw DNA sequence might yield representations rich in cis-regulatory information

[60]. However, recent benchmarks have found that, when evaluated in a zero-shot setting, most pre-trained gLMs underperform on regulatory prediction tasks relative to traditional one-hot encodings, highlighting limitations in their ability to capture functional biology without downstream fine-tuning [3, 38]. Motivated by these benchmarks, we ask whether D3, trained solely with a generative objective and no functional labels, can provide more informative representations for regulatory activity prediction.

We evaluate this question using the lentiMPRA benchmark introduced by Tang et al. (2024), which assesses frozen sequence embeddings on enhancer activity prediction across two human cell lines (K562 and HepG2). The dataset contains 230 nt sequences paired with quantitative enhancer activity values [61]. The original benchmark compared embeddings from four gLMs—Nucleotide Transformer [29], DNABERT2 [30], HyenaDNA [28], and GPN [32]—using a fixed downstream CNN. It also included supervised baselines (SEI [51], Enformer [62]), a one-hot CNN, and a ResNet trained directly on one-hot sequences [63, 64]. All models followed the CNN architecture used in Ref. [3], unless specified otherwise.

To add D3 to this benchmark, we trained it in an unsupervised manner on the lentiMPRA training data and extracted frozen embeddings from the final layer of the score network at a low-noise timestep. These embeddings were then used as inputs to the same CNN. We found that low-noise embeddings from the final layer yielded the most predictive representations.

As shown in Table 3, D3 embeddings outperform all tested gLMs—including transformer-based (Nucleotide Transformer, DNABERT2), state-space (HyenaDNA, Evo2), and convolutional (GPN) models—across both cell types. D3 also exceeds the performance of supervised SEI and Enformer embeddings on K562 and trails them only slightly on HepG2. These results are particularly striking given that D3's training involved no regulatory labels, highlighting its capacity to recover biologically meaningful features from sequence alone. These results reinforce and extend the findings of [3], who observed that pre-trained gLMs largely struggle to match one-hot performance in non-coding settings. D3 breaks from this trend, providing evidence that score-based diffusion can recover functionally rich representations without fine-tuning and not requiring massive model scale. While supervised models like ResNet remain upper bounds, D3 narrows the gap using a purely unsupervised objective.

To further evaluate generalization across species and regulatory contexts, we benchmarked embeddings from D3 and Evo2 [65], a recently introduced a 7B and 40B parameter genomic language model, on the fly enhancer STARR-seq dataset. DeepSTARR models trained on the full D3 embeddings consistently outperformed those using Evo2 embeddings and one-hot inputs across both

Table 3: Performance of embeddings. Left: Performance on cell-type specific regulatory activity prediction task from lentiMPRA data using a baseline CNN trained using gLM embeddings, D3 embeddings, supervised embeddings, or one-hot seqeunces. [1]Enformer was trained using a ridge regression on the mean embeddings. [2]ResNet represents performance of more sophisticated models trained using one-hot sequences. Right: Performance on fly enhancer activity prediction task for developmental (Dev) and Housekeeping (Hk) conditions from STARR-seq data using a DeepSTARR model trained using one-hot, Evo2 and D3 embeddings.

(a) LentiMPRA (Pearson's r)

Embedding	K562	HepG2
NT (2B51000G)	0.549	0.473
DNABERT2	0.568	0.434
HyenaDNA	0.650	0.556
GPN	0.678	0.637
Evo2(7B)	0.666	0.587
Evo2(40B)	0.696	0.616
D3	0.766	0.705
SEI	0.751	0.735
Enformer[1]	0.731	0.728
One-hot	0.691	0.638
ResNet[2]	0.789	0.753
Random	0.435	0.335

(b) Fly Enhancers (STARR-seq)

Embedding	Pearson's r (↑)		Spearman's ρ (↑)	
	Dev	HK	Dev	HK
One-hot	0.664	0.741	0.635	0.560
Evo2(7B)	0.526	0.568	0.519	0.455
Evo2(40B)	0.578	0.723	0.562	0.530
D3	**0.687**	**0.785**	**0.656**	**0.603**

developmental and housekeeping enhancer prediction tasks (Table 3). Notably, despite its scale, Evo2 underperformed relative to the one-hot baseline, highlighting persistent limitations of current gLMs in modeling non-coding regulatory logic. These findings were robust across alternative probing methods (Table 6, Appendix G).

We acknowledge that while Evo2 was trained using a curriculum that incorporated cis-regulatory elements, D3 was trained in an unsupervised manner on the STARR-seq training set. It remains an open question whether Evo2, when trained from scratch on the same dataset, would achieve comparable performance. Nevertheless, Evo2 is considered a foundation model trained on genomes of diverse species across the tree of life, and thus serves as an appropriate benchmark to assess generalization and transfer capabilities in regulatory sequence modeling.

Together, these results demonstrate that D3 offers a powerful alternative to pre-trained language models for learning regulatory representations, yielding embeddings that are more predictive, transferable, and learned with dramatically fewer parameters (92M vs 7B) and no functional labels.

7 D3 Utility in Small-Data Regimes

Generative models are valuable in data-limited settings, which are common in regulatory genomics. We evaluated whether D3 can maintain generation quality when trained on small datasets and whether its synthetic sequences can improve supervised modeling via data augmentation. To simulate small-data regimes, we downsampled the training set to 10%, 25%, 50%, and 75% of its original size and trained D3 models to generate sequences conditioned on test-set activities. We found that D3 maintained high generation fidelity down to 25% of the data, with sharp performance degradation only at 10% (Fig. 10a, Appendix F), indicating strong generalization even when data is limited.

To assess the informativeness of D3-generated sequences, we trained a DeepSTARR model on real data augmented with synthetic sequences from the 25%-data D3 model. Augmenting the dataset by 2x, 3x, and 4x led to consistent gains in predictive accuracy across both enhancer tasks (Fig. 10b,c, Appendix F), demonstrating that D3-generated sequences capture informative regulatory features complementary to the original data. We also explored two strategies to improve generative performance in low-data regimes. First, we tested transfer learning by pretraining D3 on a Drosophila ATAC-seq dataset [59], using binary chromatin accessibility profiles as conditional labels. Second, we applied EvoAug [57], a two-stage evolution-inspired augmentation framework that perturbs sequences through insertions, deletions, inversions, and mutations during pretraining, followed by fine-tuning on the unperturbed data.

Among all strategies, EvoAug-pretrained models provided the most consistent improvements in downstream DeepSTARR performance, even outperforming D3 models trained solely on 25% of real data. Transfer learning from ATAC-seq showed limited gains. These results indicate that EvoAug exposes D3 to broader and more diverse sequence grammars, enabling it to generate more functionally informative sequences. Together, these findings demonstrate that D3 can produce high-quality synthetic sequences that enhance supervised learning, even in data-scarce settings. Moreover, evolution-inspired pretraining like EvoAug offers an effective strategy to improve generative capabilities when large-scale functional datasets are unavailable.

8 Limitations

While D3 demonstrates strong performance across a range of regulatory sequence design tasks, several limitations remain. First, our benchmarking focused mostly on discrete diffusion models and did not include alternative generative strategies such as masked diffusion [66, 67], latent-space diffusion models [68], or supervised sequence design approaches that directly optimize oracle predictions [16, 18]. Second, our evaluation relied entirely on in silico proxies, without experimental validation of functional activity; as a result, our metrics primarily assess alignment with supervised predictors rather than true biological efficacy. Third, D3 operates directly on raw nucleotide sequences without compression or hierarchical abstraction, limiting its scalability to longer inputs. Fourth, we did not explore architectural enhancements, which could improve D3's performance in each benchmark. More scalable designs such as state-space models or striped architectures like Evo2 may offer performance gains and improve scalability. Finally, we did not evaluate D3's behavior under out-of-distribution generation, a key consideration for real-world applications.

9 Conclusion

We introduced DNA Discrete Diffusion (D3), a generative model that adapts score-entropy discrete diffusion for designing regulatory DNA sequences with targeted functional activity. Through extensive benchmarking, we demonstrated that D3 outperforms existing methods in generating sequences that better capture the functional, sequence-level, and compositional properties of the regulatory genome, both in unsupervised and conditional generation settings. Remarkably, D3-generated sequences improved supervised modeling even in data-limited settings, underscoring their ability to decipher complex cis-regulatory mechanisms, which can be leveraged for synthetic data augmentation and transfer learning. These results demonstrate the potential of diffusion-based training to uncover complex cis-regulatory grammars without relying on labeled data. Future work should focus on improving sampling efficiency, refining diffusion dynamics, scaling to longer sequences to model distal interactions, model interpretability to gain insights into learned cis-regulatory biology, and validating predictions experimentally. Overall, D3 provides a biologically grounded and scalable approach to generative modeling, opening new avenues for targeted regulatory sequence design in biomedicine and synthetic biology.

10 Broader Impact

Understanding and engineering the regulatory genome has transformative potential across biomedicine, synthetic biology, and fundamental biology. DNA Discrete Diffusion (D3) contributes toward this goal by enabling flexible, scalable generation of regulatory DNA sequences with desired functional properties. In research contexts, D3 could accelerate the development of hypotheses about cis-regulatory logic, guide the design of high-throughput functional assays, and improve annotation of non-coding regions implicated in disease. Clinically, this capability could eventually support personalized gene therapies by enabling the design of cell-type-specific regulatory elements that drive precise expression of therapeutic genes. However, this work also raises considerations regarding dual-use and equitable access. The ability to design synthetic regulatory elements could be misused to manipulate gene expression in unintended or harmful ways. Such misuse may pose serious biosafety and biosecurity risks, raise ethical concerns around the modification of living systems, and carry the potential for unintended ecological consequences. Moreover, as generative models for biology become more powerful, there is a risk that access and benefits may be concentrated among a small number of institutions with the resources to deploy them. We encourage open-access dissemination of methods and benchmarks, transparent documentation of model limitations, and close collaboration with experimentalists to ensure safe and meaningful use. Broader societal impacts will depend on how such technologies are governed and integrated into clinical and biotechnological pipelines.

References

[1] Hannah K Long, Sara L Prescott, and Joanna Wysocka. Ever-changing landscapes: transcriptional enhancers in development and evolution. *Cell*, 167(5):1170–1187, 2016.

[2] Seungsoo Kim and Joanna Wysocka. Deciphering the multi-scale, quantitative cis-regulatory code. *Molecular cell*, 83(3):373–392, 2023.

[3] Ziqi Tang and Peter K Koo. Evaluating the representational power of pre-trained dna language models for regulatory genomics. *bioRxiv*, pages 2024–02, 2024.

[4] Aman Patel, Arpita Singhal, Austin Wang, Anusri Pampari, Maya Kasowski, and Anshul Kundaje. Dart-eval: A comprehensive dna language model evaluation benchmark on regulatory dna. *Advances in Neural Information Processing Systems*, 37:62024–62061, 2025.

[5] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion language modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

[6] François Spitz and Eileen EM Furlong. Transcription factors: from enhancer binding to developmental control. *Nature reviews genetics*, 13(9):613–626, 2012.

[7] Justin B Kinney and David M McCandlish. Massively parallel assays and quantitative sequence–function relationships. *Annual review of genomics and human genetics*, 20:99–127, 2019.

[8] Cosmas D Arnold, Daniel Gerlach, Christoph Stelzer, Łukasz M Boryń, Martina Rath, and Alexander Stark. Genome-wide quantitative enhancer activity maps identified by starr-seq. *Science*, 339(6123):1074–1077, 2013.

[9] M Grace Gordon, Fumitaka Inoue, Beth Martin, Max Schubach, Vikram Agarwal, Sean Whalen, Shiyun Feng, Jingjing Zhao, Tal Ashuach, Ryan Ziffra, et al. lentimpra and mpraflow for high-throughput functional characterization of gene regulatory elements. *Nature protocols*, 15(8):2387–2412, 2020.

[10] Toshiyuki Shiraki, Shinji Kondo, Shintaro Katayama, Kazunori Waki, Takeya Kasukawa, Hideya Kawaji, Rimantas Kodzius, Akira Watahiki, Mari Nakamura, Takahiro Arakawa, et al. Cap analysis gene expression for high-throughput analysis of transcriptional starting point and identification of promoter usage. *Proceedings of the National Academy of Sciences*, 100(26):15776–15781, 2003.

[11] Jason D Buenrostro, Paul G Giresi, Lisa C Zaba, Howard Y Chang, and William J Greenleaf. Transposition of native chromatin for fast and sensitive epigenomic profiling of open chromatin, dna-binding proteins and nucleosome position. *Nature methods*, 10(12):1213–1218, 2013.

[12] Peter K Koo and Matt Ploenzke. Deep learning for inferring transcription factor binding sites. *Current opinion in systems biology*, 19:16–23, 2020.

[13] L. Barbadilla-Martínez, N. Klaassen, B. van Steensel, et al. Predicting gene expression from dna sequence using deep learning models. *Nature Reviews Genetics*, 2025.

[14] Seppe De Winter, Vasileios Konstantakos, and Stein Aerts. Modelling and design of transcriptional enhancers. *Nature Reviews Bioengineering*, pages 1–16, 2025.

[15] Johannes Linder and Georg Seelig. Fast differentiable dna and protein sequence optimization for molecular design. *arXiv preprint arXiv:2005.11275*, 2020.

[16] 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.

[17] 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.

[18] 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.

[19] 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*, pages 9786–9801. PMLR, 2022.

[20] Alexander Sasse, Bernard Ng, Anna E Spiro, Shinya Tasaki, David A Bennett, Christopher Gaiteri, Philip L De Jager, Maria Chikina, and Sara Mostafavi. Benchmarking of deep neural networks for predicting personal gene expression from dna sequence highlights shortcomings. *Nature Genetics*, 55(12):2060–2064, 2023.

[21] Connie Huang, Richard W Shuai, Parth Baokar, Ryan Chung, Ruchir Rastogi, Pooja Kathail, and Nilah M Ioannidis. Personal transcriptome variation is poorly explained by current genomic deep learning models. *Nature Genetics*, 55(12):2056–2059, 2023.

[22] Ziqi Tang, Shushan Toneyan, and Peter K Koo. Current approaches to genomic deep learning struggle to fully capture human genetic variation. *Nature Genetics*, 55(12):2021–2022, 2023.

[23] Pooja Kathail, Richard W Shuai, Ryan Chung, Chun Jimmie Ye, Gabriel B Loeb, and Nilah M Ioannidis. Current genomic deep learning models display decreased performance in cell type-specific accessible regions. *Genome Biology*, 25(1):202, 2024.

[24] Nathan Killoran, Leo J Lee, Andrew Delong, David Duvenaud, and Brendan J Frey. Generating and designing dna with deep generative models. *arXiv preprint arXiv:1712.06148*, 2017.

[25] Jan Zrimec, Xiaozhi Fu, Azam Sheikh Muhammad, Christos Skrekas, Vykintas Jauniskis, Nora K Speicher, Christoph S Börlin, Vilhelm Verendel, Morteza Haghir Chehreghani, Devdatt Dubhashi, et al. Controlling gene expression with deep generative design of regulatory dna. *Nature communications*, 13(1):5099, 2022.

[26] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[27] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

[28] Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Callum Birch-Sykes, Michael Wornow, Aman Patel, Clayton Rabideau, Stefano Massaroli, Yoshua Bengio, et al. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. *arXiv preprint arXiv:2306.15794*, 2023.

[29] Hugo Dalla-Torre, Liam Gonzalez, Javier Mendoza-Revilla, Nicolas Lopez Carranza, Adam Henryk Grzywaczewski, Francesco Oteri, Christian Dallago, Evan Trop, Hassan Sirelkhatim, Guillaume Richard, et al. The nucleotide transformer: Building and evaluating robust foundation models for human genomics. *bioRxiv*, 2023.

[30] Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana Davuluri, and Han Liu. Dnabert-2: Efficient foundation model and benchmark for multi-species genome. *arXiv*, 2023.

[31] Veniamin Fishman, Yuri Kuratov, Maxim Petrov, Aleksei Shmelev, Denis Shepelin, Nikolay Chekanov, Olga Kardymon, and Mikhail Burtsev. Gena-lm: A family of open-source foundational models for long dna sequences. *bioRxiv*, pages 2023–06, 2023.

[32] Gonzalo Benegas, Sanjit Singh Batra, and Yun S Song. Dna language models are powerful predictors of genome-wide variant effects. *Proceedings of the National Academy of Sciences*, 120(44):e2311219120, 2023.

[33] Alexander Karollus, Johannes Hingerl, Dennis Gankin, Martin Grosshauser, Kristian Klemon, and Julien Gagneur. Species-aware dna language models capture regulatory elements and their evolution. *bioRxiv*, 2023.

[34] Yanyi Chu, Dan Yu, Yupeng Li, Kaixuan Huang, Yue Shen, Le Cong, Jason Zhang, and Mengdi Wang. A 5 utr language model for decoding untranslated regions of mrna and function predictions. *Nature Machine Intelligence*, pages 1–12, 2024.

[35] Eric Nguyen, Michael Poli, Matthew G. Durrant, Armin W. Thomas1, Brian Kang, Jeremy Sullivan, Madelena Y. Ng, Ashley Lewis, Aman Patel, Aaron Lou, Stefano Ermon, Stephen A. Baccus, Tina Hernandez-Boussard1, Christopher Re, Patrick D. Hsu, and Brian L. Hie. Sequence modeling and design from molecular to genome scale with evo. *bioRxiv*, 2024.

[36] Yair Schiff, Chia-Hsiang Kao, Aaron Gokaslan, Tri Dao, Albert Gu, and Volodymyr Kuleshov. Caduceus: Bi-directional equivariant long-range dna sequence modeling. *arXiv 2403.03234*, 2024.

[37] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[38] Aman Patel, Arpita Singhal, Austin Wang, Anusri Pampari, Maya Kasowski, and Anshul Kundaje. Dart-eval: A comprehensive dna language model evaluation benchmark on regulatory dna. *arXiv preprint arXiv:2412.05430*, 2024.

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

[40] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[41] Lucas Ferreira DaSilva, Simon Senan, Zain Munir Patel, Aniketh Janardhan Reddy, Sameer Gabbita, Zach Nussbaum, Cesar Miguel Valdez Cordova, 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*, pages 2024–02, 2024.

[42] Zehui Li, Yuhao Ni, Tim August B Huygelen, Akashaditya Das, Guoxuan Xia, Guy-Bart Stan, and Yiren Zhao. Latent diffusion model for dna sequence generation. *arXiv preprint arXiv:2310.06150*, 2023.

[43] Hannes Stark, Bowen Jing, Chenyu Wang, Gabriele Corso, Bonnie Berger, Regina Barzilay, and Tommi Jaakkola. Dirichlet flow matching with applications to dna sequence design. *arXiv preprint arXiv:2402.05841*, 2024.

[44] Dmitry Penzar, Daria Nogina, Elizaveta Noskova, Arsenii Zinkevich, Georgy Meshcheryakov, Andrey Lando, Abdul Muntakim Rafi, Carl de Boer, and Ivan V Kulakovskiy. Legnet: a best-in-class deep learning model for short dna regulatory regions. *Bioinformatics*, 39(8):btad457, 2023.

[45] Thomas H Jukes, Charles R Cantor, et al. Evolution of protein molecules. *Mammalian protein metabolism*, 3(21):132, 1969.

[46] 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*, pages 1276–1301. PMLR, 2023.

[47] Bernardo P de Almeida, Franziska Reiter, Michaela Pagani, and Alexander Stark. Deepstarr predicts enhancer activity from dna sequence and enables the de novo design of synthetic enhancers. *Nature genetics*, 54(5):613–624, 2022.

[48] AR Forrest, H Kawaji, M Rehli, JK Baillie, MJ de Hoon, V Haberle, T Lassmann, IV Kulakovskiy, M Lizio, M Itoh, et al. A promoter-level mammalian expression atlas. *Nature*, 507(7493):462–470, 2014.

[49] Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022.

[50] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

[51] Kathleen M Chen, Aaron K Wong, Olga G Troyanskaya, and Jian Zhou. A sequence-based global map of regulatory activity for deciphering human genetics. *Nature genetics*, 54(7):940–949, 2022.

[52] Ieva Rauluseviciute, Rafael Riudavets-Puig, Romain Blanc-Mathieu, Jaime A Castro-Mondragon, Katalin Ferenc, Vipin Kumar, Roza Berhanu Lemma, Jérémy Lucas, Jeanne Chèneby, Damir Baranasic, et al. Jaspar 2024: 20th anniversary of the open-access database of transcription factor binding profiles. *Nucleic Acids Research*, 52(D1):D174–D182, 2024.

[53] Charles E Grant, Timothy L Bailey, and William Stafford Noble. Fimo: scanning for occurrences of a given motif. *Bioinformatics*, 27(7):1017–1018, 2011.

[54] Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, et al. Captum: A unified and generic model interpretability library for pytorch. *arXiv preprint arXiv:2009.07896*, 2020.

[55] Antonio Majdandzic, Chandana Rajesh, and Peter K Koo. Correcting gradient-based interpretations of deep neural networks for genomics. *Genome Biology*, 24(1):109, 2023.

[56] Antonio Majdandzic, Chandana Rajesh, Ziqi Tang, Shushan Toneyan, Ethan L Labelson, Rohit K Tripathy, and Peter K Koo. Selecting deep neural networks that yield consistent attribution-based interpretations for genomics. In *Machine Learning in Computational Biology*, pages 131–149. PMLR, 2022.

[57] Nicholas Keone Lee, Ziqi Tang, Shushan Toneyan, and Peter K Koo. Evoaug: improving generalization and interpretability of genomic deep neural networks with evolution-inspired data augmentations. *Genome Biology*, 24(1):105, 2023.

[58] Jasper Janssens, Sara Aibar, Ibrahim Ihsan Taskiran, Joy N Ismail, Alicia Estacio Gomez, Gabriel Aughey, Katina I Spanier, Florian V De Rop, Carmen Bravo Gonzalez-Blas, Marc Dionne, et al. Decoding gene regulation in the fly brain. *Nature*, 601(7894):630–636, 2022.

[59] 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.

[60] Gonzalo Benegas, Chengzhong Ye, Carlos Albors, Jianan Canal Li, and Yun S Song. Genomic language models: opportunities and challenges. *Trends in Genetics*, 2025.

[61] Vikram Agarwal, Fumitaka Inoue, Max Schubach, Dmitry Penzar, Beth K Martin, Pyaree Mohan Dash, Pia Keukeleire, Zicong Zhang, Ajuni Sohota, Jingjing Zhao, et al. Massively parallel characterization of transcriptional regulatory elements. *Nature*, pages 1–10, 2025.

[62] 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.

[63] Peter K Koo, Antonio Majdandzic, Matthew Ploenzke, Praveen Anand, and Steffan B Paul. Global importance analysis: An interpretability method to quantify importance of genomic features in deep neural networks. *PLoS computational biology*, 17(5):e1008925, 2021.

[64] Shushan Toneyan, Ziqi Tang, and Peter K Koo. Evaluating deep learning for predicting epigenomic profiles. *Nature machine intelligence*, 4(12):1088–1100, 2022.

[65] Garyk Brixi, Matthew G Durrant, Jerome Ku, Michael Poli, Greg Brockman, Daniel Chang, Gabriel A Gonzalez, Samuel H King, David B Li, Aditi T Merchant, et al. Genome modeling and design across all domains of life with evo 2. *BioRxiv*, pages 2025–02, 2025.

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

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

[68] 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*, 2024.

[69] T Dao, DY Fu, S Ermon, A Rudra, and C Flashattention Ré. Fast and memory-efficient exact attention with io-awareness, 2022. *URL https://arxiv. org/abs/2205.14135*.

[70] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[71] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[72] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv 1810.04805*, 2018.

[73] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[74] Ishaan Gulrajani and Tatsunori B Hashimoto. Likelihood-based diffusion language models. *Advances in Neural Information Processing Systems*, 36, 2024.

[75] EA Feingold, PJ Good, MS Guyer, S Kamholz, L Liefer, K Wetterstrand, FS Collins, TR Gingeras, D Kampa, EA Sekinger, et al. The encode (encyclopedia of dna elements) project. *Science*, 306(5696):636–640, 2004.

[76] Peter K Koo and Matt Ploenzke. Improving representations of genomic sequence motifs in convolutional networks with exponential activations. *Nature machine intelligence*, 3(3):258–266, 2021.

[77] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv 1412.6980*, 2014.

[78] Aniketh Janardhan Reddy, Michael H Herschl, Sathvik Kolli, Amy X Lu, Xinyang Geng, Aviral Kumar, Patrick D Hsu, Sergey Levine, and Nilah M Ioannidis. Pretraining strategies for effective promoter-driven gene expression prediction. *bioRxiv*, 2023.

A Evaluation Framework for Regulatory DNA Sequences

We evaluate generated regulatory DNA sequences along three key dimensions (Fig. 2): (1) *functional similarity*, (2) *sequence similarity*, and (3) *compositional similarity*. Let x_{real} denote a real sequence from the dataset and x_{gen} a generated sequence. An oracle model f_θ (a pretrained deep neural network) provides an *in silico* prediction $\hat{y} = f_\theta(x)$ estimating functional activity for any input sequence x. We denote the conditional target activity as y_{cond}, and true experimental measurements (if available) as y_{exp}. Since experimental assays are costly, we rely on the oracle to approximate y_{exp}.

Functional similarity evaluates whether generated sequences achieve functional activities similar to real sequences or the conditioning target y_{cond}. Sequence similarity quantifies novelty and memorization relative to training data. Compositional similarity assesses whether generated sequences recapitulate key motif content and regulatory features found in real genomic sequences.

A.1 Functional Similarity

Conditional generation fidelity measures how well generated sequences achieve the conditioning target. Given oracle predictions $\hat{y}_i = f_\theta(x_{\text{gen},i})$ for each generated sequence $x_{\text{gen},i}$, fidelity is computed as:

$$\text{Fidelity} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_{\text{cond}})^2$$

where N is the number of generated sequences. Lower values indicate higher fidelity to y_{cond}.

Fréchet distance compares the distribution of oracle-predicted embeddings between real and generated sequences. Let $\mu_{\text{gen}}, \Sigma_{\text{gen}}$ denote the mean and covariance of oracle embeddings for x_{gen}, and $\mu_{\text{real}}, \Sigma_{\text{real}}$ for x_{real}. Assuming multivariate normality, Fréchet distance is:

$$d_F^2 = \|\mu_{\text{gen}} - \mu_{\text{real}}\|_2^2 + \text{Tr}\left(\Sigma_{\text{gen}} + \Sigma_{\text{real}} - 2(\Sigma_{\text{gen}}\Sigma_{\text{real}})^{1/2}\right)$$

Lower values indicate closer alignment of generated and real data in oracle embedding space.

Predictive distribution shift uses the Kolmogorov–Smirnov (KS) statistic to compare the empirical cumulative distribution functions (ECDFs) of oracle predictions for x_{gen} and x_{real}:

$$D_{KS} = \sup_y |F_{\text{gen}}(y) - F_{\text{real}}(y)|$$

Lower D_{KS} implies more similar predictive distributions.

A.2 Sequence Similarity

Percent identity quantifies similarity using normalized Hamming distance. For each generated sequence $x_{\text{gen},i}$, percent identity against all real sequences is:

$$\text{PID}(x_{\text{gen},i}, x_{\text{real}}) = 1 - \frac{d_H(x_{\text{gen},i}, x_{\text{real}})}{L}$$

where d_H is Hamming distance and L is sequence length. We report the average maximum percent identity over all $x_{\text{gen},i}$ relative to real sequences (memorization) and within generated sequences (diversity).

k-mer spectrum shift compares k-mer frequency distributions between x_{gen} and x_{real} via Jensen–Shannon divergence:

$$D_{JS}(P_{\text{gen}} \parallel P_{\text{real}}) = \frac{1}{2}D_{KL}(P_{\text{gen}} \parallel M) + \frac{1}{2}D_{KL}(P_{\text{real}} \parallel M)$$

where $M = \frac{1}{2}(P_{\text{gen}} + P_{\text{real}})$. We compute D_{JS} for $k = 1, \ldots, 7$.

Discriminability quantifies the ability to distinguish generated sequences $x_{\text{gen}} \sim p_{\text{gen}}$ from real sequences $x_{\text{real}} \sim p_{\text{real}}$ by training a binary classifier $h : x \to \{0, 1\}$ to predict whether a sequence is generated ($y = 1$) or real ($y = 0$). Discriminability is measured by AUROC:

$$\text{AUROC} = \text{AUC}(h(x), y)$$

An AUROC near 0.5 indicates indistinguishability; an AUROC near 1 indicates separability.

A.3 Compositional Similarity

Motif enrichment. Let $M = \{m_1, \ldots, m_K\}$ denote a set of K known motifs (e.g., $K = 879$ from JASPAR). For each sequence x, we count motif occurrences $c_k(x)$ using a motif scanner (FIMO). Aggregate counts:

$$C_{\text{gen}} = \frac{1}{N_{\text{gen}}} \sum_{i=1}^{N_{\text{gen}}} c(x_{\text{gen},i}), \quad C_{\text{real}} = \frac{1}{N_{\text{real}}} \sum_{j=1}^{N_{\text{real}}} c(x_{\text{real},j})$$

Compute Pearson correlation:

$$r = \text{corr}(C_{\text{gen}}, C_{\text{real}})$$

Higher r indicates similar motif prevalence.

Motif co-occurrence. Define binary motif presence vectors $b(x) \in \{0,1\}^K$ where $b_k(x) = 1$ if motif m_k occurs in x. Stack vectors:

$$B_{\text{gen}} \in \{0,1\}^{N_{\text{gen}} \times K}, \quad B_{\text{real}} \in \{0,1\}^{N_{\text{real}} \times K}$$

Compute sample covariance matrices:

$$\Sigma_{\text{gen}} = \frac{1}{N_{\text{gen}} - 1} B_{\text{gen}}^\top B_{\text{gen}}, \quad \Sigma_{\text{real}} = \frac{1}{N_{\text{real}} - 1} B_{\text{real}}^\top B_{\text{real}}$$

Quantify co-occurrence similarity by Frobenius norm:

$$d_F = \|\Sigma_{\text{gen}} - \Sigma_{\text{real}}\|_F$$

Lower d_F indicates similar motif co-occurrence patterns.

Attribution consistency. Attribution consistency evaluates whether highly attributed positions exhibit consistent motif-like patterns. Attribution maps $A(x) \in \mathbb{R}^L$ are computed for each sequence using GradientSHAP with R random references, followed by gradient correction. For each x, aggregate attribution scores over k-mers to define normalized attribution distribution $P_A(k)$ over $K = 4^k$ possible k-mers. Consistency is quantified by KL divergence relative to uniform $U(k) = 1/K$:

$$D_{KL}(P_A \parallel U) = \sum_w P_A(w) \log \frac{P_A(w)}{U(w)}$$

We compute D_{KL} for generated, real, and concatenated sets. Higher D_{KL} indicates stronger enrichment of consistent sequence patterns in highly attributed regions.

A.4 Other Evaluations

Informativeness measures whether generated sequences improve predictive power. Let $D_{\text{train}}^{(\alpha)} \subseteq D_{\text{real}}$ be a real training subset (fraction α) and D_{gen} the generated set. Define hybrid dataset:

$$D_\alpha = D_{\text{gen}} \cup D_{\text{train}}^{(\alpha)}$$

Train a supervised model f_θ on D_α, evaluate on validation set D_{val}. Measure validation metric $M(D_\alpha)$ (e.g., R^2, AUROC, MSE):

$$\Delta M(\alpha) = M(D_\alpha) - M(D_{\text{train}}^{(\alpha)})$$

Positive $\Delta M(\alpha)$ indicates added predictive information from generated sequences.

Conditional sampling diversity measures pairwise percent identity among sequences generated at fixed y_{cond}.

Mechanistic diversity measures pairwise Euclidean distances between attribution maps $A(x_{\text{gen}})$ at fixed y_{cond}.

Task-specific design evaluates whether sequences conditioned on y_{cond} achieve desired functional activities by comparing oracle-predicted distributions $\hat{y}$.

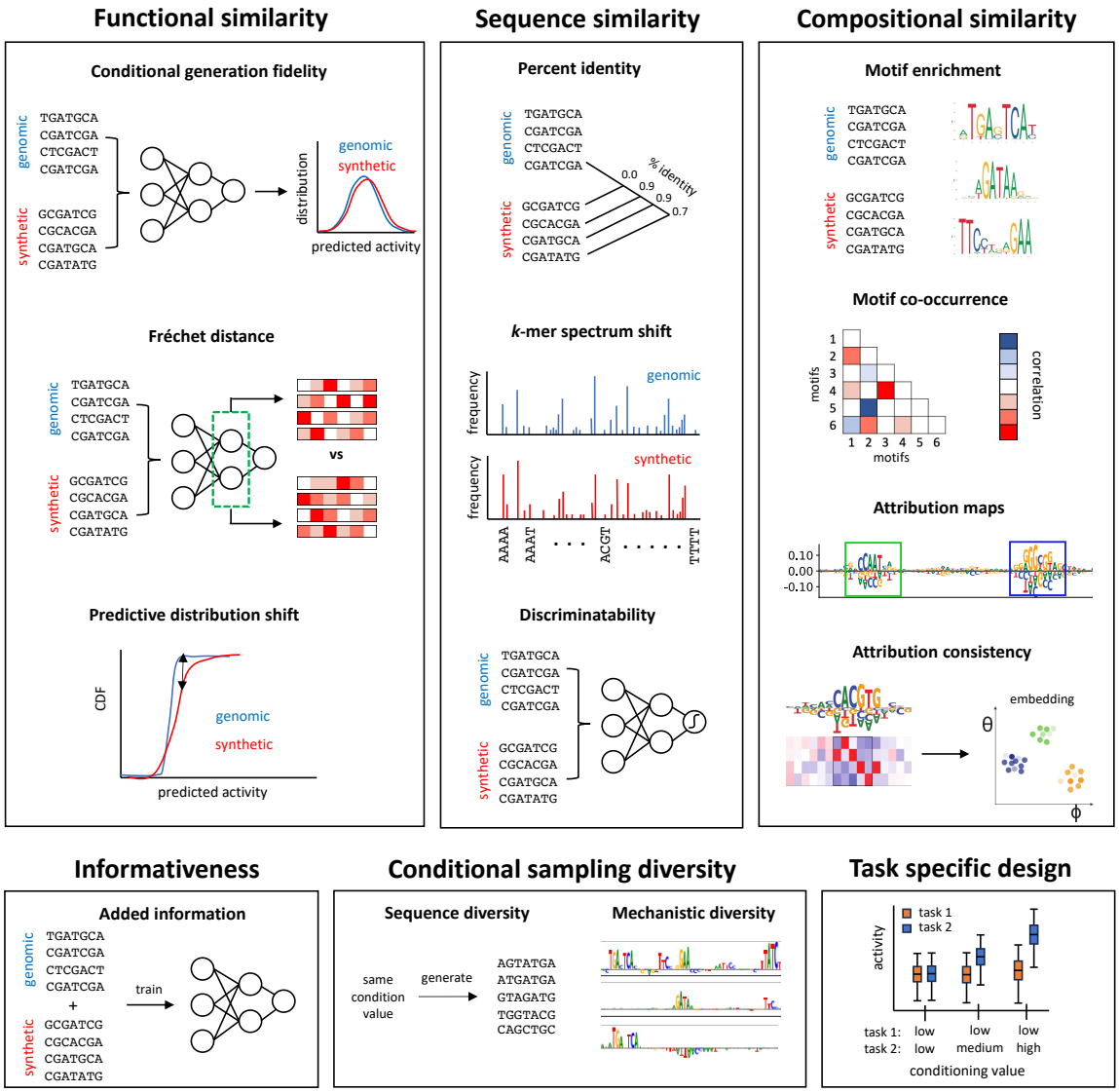

Figure 2: Schematic of evaluation framework.

B DNA Discrete Diffusion

We introduce DNA Discrete Diffusion (D3), a score entropy discrete diffusion model designed for the conditional generation of regulatory DNA sequences with specified functional activity levels. D3 adapts the Score Entropy Discrete Diffusion (SEDD) framework to operate directly within the discrete nucleotide space ({A,C,G,T}), which is particularly suitable for modeling regulatory DNA characterized by sparse functional motifs within noninformative regions.

B.1 Continuous Time Framework

The D3 model employs a continuous-time Markov process. Here, the transitions can occur at any time during $t = 0$ to $t = T$, which is more flexible compared to the discrete-time scenario. This process starts with an initial distribution q_0 and is driven by a transition rate matrix $\mathcal{R}_t \in \mathbb{R}^{S \times S}$ that defines how the state changes during the process. The corresponding discrete diffusion process can be denoted by a linear ordinary differential $\frac{dq_t}{dt} = \mathcal{R}_t q_t$ where q_t represents the family of distributions that evolves with time t that starts with $q_0 \approx q_{data}$. The transition rate matrix $\mathcal{R}_t$ should have non-negative non-diagonal entries and columns summed to zero that restrict p_t to gaining or losing any mass during the process. $\mathcal{R}_t$ can be analytically chosen as $\mathcal{R}_t = \beta(t)\mathcal{R}_b$ with a time-dependent scalar factor β_t and time independent base rate matrix $\mathcal{R}_b$ such that p_t approaches a stationary distribution p_{ref} as $t \to \infty$.

$\mathcal{R}_t$ can also represent infinitesimal transition probability and corresponding small Δt Euler steps for the process at any time points $t - \Delta t$ and t as below:

$$q_{t|t-\Delta t}(x|\bar{x}) = \delta_{x,\bar{x}} + \mathcal{R}_t(\bar{x}, x)\Delta t + \mathcal{O}(\Delta t) \tag{1}$$

The reverse process starts from the marginal $q_T(x_T)$ and can lead to data distribution $p_{data}(x_0)$ with suitable transition rate matrix $\bar{\mathcal{R}}_t \in \mathbb{R}^{S \times S}$ as:

$$q_{t|t+\Delta t}(\bar{x}|x) = \delta_{\bar{x},x} + \bar{\mathcal{R}}_t(x, \bar{x})\Delta t + \mathcal{O}(\Delta t) \tag{2}$$

Also the reverse transition matrix satisfies

$$\bar{\mathcal{R}}_t(x, \bar{x}) = \frac{q_t(\bar{x})}{q_t(x)}\mathcal{R}_t(\bar{x}, x) \tag{3}$$

and the reverse process ODE can be given by

$$\frac{dq_{T-t}}{dt} = \bar{\mathcal{R}}_{T-t} q_{T-t} \tag{4}$$

The ratios $\frac{q_t(\bar{x})}{q_t(x)}$ are collectively known as concrete score, but are generally intractable. Their behavioral similarity to score function $\nabla \log q(x)$ for continuous distributions instigates learning the ratios for discrete data points (such as genomics data that we are trying to model).

B.2 Use of Score Entropy for Discrete Diffusion

Concrete score matching generalizes the standard Fisher divergence to learn a neural network $s_\theta(x, t)$ that can approximate the density ratios $\frac{q_t(\bar{x})}{q_t(x)}$ but does not perform well due to the inability of l_2 loss to penalize the non-positive values. On the other hand, score entropy is defined as Bregman divergence $D_F(s_\theta(x), \frac{q(\bar{x})}{q(x)})$ that enjoys convexity and non-negativity properties in general. Specifically, we consider divergence associated with $F(p)$ as $\sum p \log p$, i.e., the negative entropy function that inspired the name. Score entropy loss ($\mathcal{L}_{SE}$) is defined as:

$$\mathcal{L}_{SE} = \mathbb{E}_{x \sim q}\left[\sum_{\bar{x} \neq x} w_{x\bar{x}} \left\{ s_\theta(x)_{\bar{x}} - \frac{q(\bar{x})}{q(x)} \log s_\theta(x)_{\bar{x}} + K\left(\frac{q(\bar{x})}{q(x)}\right)\right\}\right], \tag{5}$$

where $K(l) = l(\log l - 1)$. For any fully supported distribution q and positive weights $w_{x\bar{x}}$, the optimal θ^* minimizing $\mathcal{L}_{SE}$ for all $x, \bar{x}$ pairs satisfies $s_{\theta^*}(x)_{\bar{x}} = \frac{q(\bar{x})}{q(x)}$, with the minimum value

being 0 at θ^*. This can be shown by taking derivative of $\mathcal{L}_{SE}$ (without considering $K(.)$ as it doesn't depend on θ) w.r.t. s_θ and setting it to 0 [5].

Unfortunately, the score entropy term contains the unknown $\frac{q_t(\bar{x})}{q_t(x)}$ term, which requires planning for an alternate solution to tractably calculate $\mathcal{L}_{SE}$. Following Theorem 3.4 in [5], we continue with calculating denoising score entropy as an equivalent (up to a constant independent of θ) of $\mathcal{L}_{SE}$. If q is a perturbation of a base density q_0 by a transition kernel $q(.|.)$ such as $q(x) = \sum_{x_0} q(x|x_0)q_0(x_0)$, $\mathcal{L}_{DSE}$ is given by

$$\mathcal{L}_{DSE} = \mathbb{E}_{x_0 \sim q_0, x \sim q(.|x_0)}[\sum_{\bar{x} \neq x} w_{x\bar{x}}(s_\theta(x)_{\bar{x}} - \frac{q(\bar{x}|x_0)}{q(x|x_0)} \log s_\theta(x)_{\bar{x}})] \tag{6}$$

Access to the perturbation kernel makes $\mathcal{L}_{DSE}$ scalable as the hope is to train a model $s_\theta(x)$ that can provide any $s_\theta(x)_{\bar{x}}$. We will discuss different perturbation techniques in the following subsection. Particularly, defining an ELBO and likelihood-based training requires defining a time-dependent score network $s_\theta(., t)$ with the parameterized reverse matrix as $\bar{\mathcal{R}}_t(\bar{x}, x) = s_\theta(x, t)\mathcal{R}_t(x, \bar{x})$ which is found by replacing the density ratios with scores in Equation 3. Also, the parameterized densities q_t^θ satisfy reverse process ODE in Equation 4. The log-likelihood of data points was derived for discrete diffusion models with a specific form for score entropy in [5] as

$$-log q_0^\theta(x_0) \leq \mathcal{L}_{DWDSE}(x_0) + D_{KL}(q_{T|0}(.|x_0)||p_{ref}) \tag{7}$$

where $q_T^\theta = p_{ref}$ for the parameterized densities q_t^θ, and $\mathcal{L}_{DWDSE}(x_0)$ is the diffusion weighted denoising score entropy for x_0, which is given by

$$\int_0^T \mathbb{E}_{x_t \sim q_{t|0}(.|x_0)} \sum_{\bar{x} \neq x_t} \mathcal{R}_t(\bar{x}_t, t) \left(s_\theta(x_t, t)_{\bar{x}} - \frac{q_{t|0}(\bar{x}|x_0)}{q_{t|0}(x|x_0)} \log s_\theta(x_t, t)_{\bar{x}} + K\left(\frac{q_{t|0}(\bar{x}|x_0)}{q_{t|0}(x|x_0)})\right)\right) dt \tag{8}$$

We follow the likelihood based training off of this upper bound with specific choices for forward transition kernel that allow easy calculation of $\mathcal{L}_{DWDSE}$ which we discuss now in detail.

B.3 Choice of Forward Process

Modeling highly complex genomics sequences of the form $\mathbf{x} = x^1...x^d$ with a high value of d through score entropy incurs real scaling challenges. Our state factorizes into sequences $\mathcal{X} = \{A, C, G, T\}^d$ such that each position of the sequence represents one nucleotide i.e. $x^i \in \{A, C, G, T\}$, which is synonymous to tokens for language models or image pixels for computer vision tasks. $\mathcal{R}_t$ should be able to take any sequence to any other sequence which requires exponential time and space, specifically $\mathcal{O}(4^d)$. Here, we consider a sparse structured matrix $\mathcal{R}_t^N$ that can perturb every nucleotide independently, which is given by

$$\mathcal{R}_t(x^1...x^i...x^d, x^1...\bar{x}^i...x^d) = \mathcal{R}_t^N(x^i, \bar{x}^i) \tag{9}$$

Specifically, we model all the density ratios for sequences with a Hamming distance of 1, which drastically reduces the span of perturbations to only sequences with one nucleotide change. This leads to the score network to model a seq-to-seq map with $\mathcal{O}(4d)$ time as below:

$$(s_\theta(x^1...x^i...x^d, t))_{i, x \to \bar{x}} \approx \frac{q_t(x^1...\bar{x}^i...x^d)}{q(x^1...x^i...x^d)} \tag{10}$$

As we consider only one token change, the calculation for the forward transition $q_{t|0}^{seq}(.|.)$ can be done in a single step by decomposing it to independent token perturbation:

$$q_{t|0}^{seq}(.|x) = \prod_{i=1}^{d} q_{t|0}^N(\bar{x}^i, x^i) \tag{11}$$

Each $q_{t|0}^N(\bar{x}^i, x^i)$ can be calculated with a noise level β and a pre-decided transition matrix $\mathcal{R}^N$ as $\mathcal{R}_t^N = \beta(t)\mathcal{R}^N$. Considering $\bar{\beta}_t$ as the cumulative noise $\int_0^t \beta(t)dt$, we have $q_{t|0}^N(.|x) = x$-th column of $exp(\bar{\beta}(t)\mathcal{R}^N)$. More specifically, that implies

$$q_{t|0}^N(\bar{x}_t = j|x_0 = i) = exp(\bar{\beta}(t)\mathcal{R}^N)(j, i) \tag{12}$$

where (j, i) represents jth row,ith column entry of the matrix $exp(\bar{\beta}(t)\mathcal{R}^N)$. With this formulation, We follow the diffusion-weighted denoising score entropy loss $\mathcal{L}_{DWDSE}$ from [5] for our model training.

Storing all possible edge weights $\mathcal{R}^N(i, j)$ requires a high cost in terms of space and time. This issue can be addressed by considering special structures for transition matrix [50]. In particular, a uniform stationary base rate matrix $\mathcal{R} = 11^\top - \mathcal{S}\mathcal{I}$ can be a natural choice for categorical discrete distributions where $11^\top$ is a matrix of all ones, $\mathcal{I}$ is identity matrix. For our specific scenario, the token size is 4, and hence the corresponding time-dependent transition matrix is given by

$$\begin{bmatrix} \frac{1}{4}+\frac{3}{4}e^{-4t} & \frac{1}{4}-\frac{1}{4}e^{-4t} & \frac{1}{4}-\frac{1}{4}e^{-4t} & \frac{1}{4}-\frac{1}{4}e^{-4t} \\ \frac{1}{4}-\frac{1}{4}e^{-4t} & \frac{1}{4}+\frac{3}{4}e^{-4t} & \frac{1}{4}-\frac{1}{4}e^{-4t} & \frac{1}{4}-\frac{1}{4}e^{-4t} \\ \frac{1}{4}-\frac{1}{4}e^{-4t} & \frac{1}{4}-\frac{1}{4}e^{-4t} & \frac{1}{4}+\frac{3}{4}e^{-4t} & \frac{1}{4}-\frac{1}{4}e^{-4t} \\ \frac{1}{4}-\frac{1}{4}e^{-4t} & \frac{1}{4}-\frac{1}{4}e^{-4t} & \frac{1}{4}-\frac{1}{4}e^{-4t} & \frac{1}{4}+\frac{3}{4}e^{-4t} \end{bmatrix}$$

This structured matrix perturbs the input sequence according to the uniformly sampled time t, with the idea that all the matrix entries tend to $\frac{1}{4}$ uniformly when $t \to \infty$. This formulation explained above enables us to calculate all values of $\mathcal{L}_{DWDSE}$ quickly, which makes the training process much faster. Specifically, $\mathcal{L}_{DWDSE}$ can be calculated as below:

$$\mathcal{L}_{DWDSE} = \beta(t) \sum_{i=1}^{d} \sum_{y=1}^{4} (1 - \delta_{\bar{x}_t^i}(y))(s_\theta(\bar{x}_t, t)_{i,y} - \frac{q_{t|0}(y|x_0^i)}{q_{t|0}(\bar{x}_t^i|x_0^i)} \log s_\theta(\bar{x}_t, t)_{i,y} \tag{13}$$

where $\beta(t)$ is noise schedule (with total noise $\bar{\beta}(t)$, and δ is Kronecker delta. While other options are available, such as absorbing state process with a stationary matrix of all zeros[5], our experiments with uniform rate matrix showed exciting results, and we leave other options for future work.

B.4 Simulating Generative Reverse Process

The parametric generative reverse process following continuous time framework with rate matrix $\bar{\mathcal{R}}$ can be employed by taking a small Euler step shown in Eq. 2. While this process can be simulated using Gillespie's Algorithm, this is impractical for large sequence lengths as stepping through each transition individually will allow only one dimension change for each backward step. Another popular approximate simulation method, tau-leaping, can be employed that applies all the transitions simultaneously. Particularly, a sequence $x_{t-\Delta t}$ can be constructed from x_t by sampling each token $x_{t-\Delta t}^i$ independently [5] by below probability:

$$q^i(x_{t-\Delta t}^i|x_t^i) = \delta_{x_t^i}(x_{t-\Delta t}^i) + \Delta t \bar{\mathcal{R}}_t(x_t^i, x_{t-\Delta t}^i) s_\theta(x_t, t)_{i, x_{t-\Delta t}^i} \tag{14}$$

The optimal denoising can be achieved only by knowing all density ratios $\frac{q_t(\bar{x})}{q_t(x)}$. If q_t follows the diffusion ODE $\frac{dq_t}{dt} = \mathcal{R}_t q_t$, the true denoiser can be represented as a discrete version of Tweedie's theorem, as shown in [5]

$$q_{0|t}(x_0|x_t) = \left(exp(-t\mathcal{R}) \left[\frac{q_t(i)}{q_t(x_t)} \right]_{i=1}^{4} \right)_{x_0} exp(t\mathcal{R}(x_t, x_0) \tag{15}$$

As we only know the density rations for sequences with 1 Hamming distance away, a Tweedie denoiser analog of tau-leaping to construct transition densities can be obtained by replacing the token transition probabilities as below:

$$q^i(x_{t-\Delta t}^i|x_t^i) = \left(exp(-\beta(t)^{\Delta t}\mathcal{R}) s_\theta(x_t, t)_i \right)_{x_{t-\Delta t}^i} exp(-\beta(t)^{\Delta t}\mathcal{R}(x_t^i, x_{t-\Delta t}^i) \tag{16}$$

where $\beta(t)^{\Delta t} = (\bar{\beta}(t) - \bar{\beta}(t - \Delta t))$. Now $q^i(x_{t-\Delta t}^i|x_t^i)$ may need clamping for minimum 0 and normalizing the sum to 1, and $x_{t-\Delta t}$ can be obtained by sampling i-th token as $x_{t-\Delta t}^i \sim q^i(x_{t-\Delta t}^i)$.

Conditional generation is achieved by providing the conditioning signal (for example, target activity levels as scalar values) as an additional input to the score network $s_\theta(x, t)$, which enables it to learn condition-specific density ratios. However, D3 supports sequence generation with categorical conditioning (for example, discrete cell-type labels) using classifier-free guidance (CFG), as demonstrated in Section 5.

C Experiment details

C.1 Diffusion Details

We provide all our code at `https://anonymous.4open.science/r/D3-DNA-Discrete-Diffusion-3E42/` for reference to the reviewers. Please note that the training and sampling code is primarily influenced by the code shared with [5] with changes required for DNA sequence modeling.

Method Details. We follow the diffusion training closely from [5] with the geometric noise distribution $\bar{\beta}(t) = \beta_{min}^{1-t}\beta_{max}^{t}$, where $\bar{\beta}$ moves from $\beta_{min} = 0$ ($1e^{-3}$ to be specific) to $\beta_{max} = 1$ with increasing time step t. The noise schedules are chosen such that the prior loss $\mathcal{D}_{KL}(q_{T|0}(.|x_0)||p_{ref}$ are negligible and $q_{\bar{\beta}(0)}$ remains close to q_{data}. The uniform transition matrix is scaled down by $\frac{1}{N}$ ($N = 4$ here), and we consider p_{ref} as uniform.

Convolution Model Details. We use the same architecture for D3-Conv as the architecture used in Refs. [43, 46] for human promoter sequences. In place of the time step embedding with Gaussian random feature projection, we use similar embedding to parameterize the network with the total noise level instead of the time t. Here, the results (MSE) are compared with other previous methods, for which we referred to [43]. We follow a similar setup for DeepSTARR and MPRA datasets, too, for D3-Conv. As DFM [43] and other previous methods did not experiment with these two datasets, we trained the DFM-conv model (with a similar setup provided for the promoter) for proper performance comparison. Please note that we only considered DFM for comparison as their results are closest to D3-conv.

Transformer Model Details. We follow the similar transformer architecture for D3-Tran as used for SEDD small [5]. We used flash attention [69] with fused kernels and adaLN-zero time information network from [70] with 128 hidden dimensions, which is based on incorporating time conditioning into a standard encoder-only transformer architecture [71, 72]. A small change on this setup is adding rotary positional embedding [73] for performance improvement. D3-Tran has 12 transformer blocks and 12 attention heads per block. Training D3-Tran with this transformer architecture yielded even better performance than D3-Conv for all three datasets, i.e., promoter, DeepSTARR, and MPRA, as explained in Section 4. Surprisingly, the performance of DFM with the transformer degraded more than its convolution counterpart, and this trend remained the same for all three datasets. We do not rule out the possibility of some minor architectural or methodical change to improve the performance of DFM for transformers, which may be explored in the near future.

Toward a more complete performance comparison of D3 with DFM, we trained DFM with the same transformer architecture with the only change of time step embedding (with Gaussian random feature projection) instead of embedding on noise level as already mentioned for convolution model details. For a fair comparison, we did not change any other part of the transformer architecture. We followed a similar overall training setup to train DFM as explained in [43].

Training Details. All of our models were trained with a learning rate 3×10^{-4}, a batch size of 128 for convolution architecture and 256 for transformer architecture. Following [5], our gradient norm is clipped to 1 and a linear warmup schedule for the first 2000 iterations. We set the training for 500000 steps with a validation step after every 50000 training step. The best models were found after different training steps for different datasets and methods. For example, DFM yielded the best model after 345730 steps, which is 400000 steps for D3 for convolution architecture with DeepSTARR. We employ an exponential moving average of 0.9999 for all our D3-Conv and D3-Tran training. All the training was performed on a single node with 4 A100 80GB GPUs.

Hyperparameter Search. As mentioned in Ref. [5], no hyperparameter or architecture search was performed for our models. The convolution architecture was chosen from the previous works [43, 46] and the transformer architecture from DDiT [70], with the rotary embedding as was used in [74]. The learning rate of 3×10^{-4} and the EMA of 0.9999 were selected as a standard choice from the previous well-known diffusion training recipes.

Conditioning Details. As mentioned in the main paper, we consider datasets with continuous-valued vectors as conditioning signals. Therefore, we send these signals with the true sequences to the score function, which is trained to learn density ratios based on the signals. For the promoter dataset, we have transcription initiation signal profiles, a vector of the same length as the sequences, i.e., 1024. Here, we concat (similar functionality as *torch.cat* for PyTorch framework) the signal profiles with the sequences before forwarding to the score function. As the fly enhancer sequences have corresponding activity values in 2 contexts (housekeeping and developmental gene promoters), we employ a 2-layered fully connected network that outputs an embedding (of the same length as sequence, i.e., 249) from the 2 activity values. Then, we concat the embeddings with the sequences and follow the same training and sampling process. For cell-type specific regulatory sequences, there are three cell-type specific activity levels. Hence, we use a similar 2-layered fully connected network to find an embedding (of length 200 to match the sequence length) that we concat to the sequences.

C.2 Oracle details

Human promoters. The oracle was Sei [51], a multi-task CNN that predicts 21,907 chromatin profiles across >1,300 cell lines/tissues from DNA sequences. Active promoter predictions were obtained from the H3K4me3 chromatin mark prediction. The Sei oracle and weights were acquired from the DDSM repository (`https://github.com/jzhoulab/ddsm`).

Fly enhancers. The DeepSTARR dataset [47] is a multi-task regression problem that predicts enhancer activity for two promoters, representing a developmental and housekeeping gene in *D. melanogaster* S2 cells. The input consists of genomic sequences with a length of 249 nucleotides, and the output comprises two scalar values indicating the activity of developmental and housekeeping enhancers, experimentally measured using STARR-seq. Sequences containing N characters were removed, which had a minimal impact on the dataset size (reducing it by approximately 0.005%). The dataset [57] was analyzed using the original DeepSTARR model architecture, described as follows:

- Input
- 1D convolution (256 filters, size 7, stride 1)
- BatchNorm + ReLU activation
- Max-pooling (size 2, stride 2)
- 1D convolution (60 filters, size 3, stride 1)
- BatchNorm + ReLU activation
- Max-pooling (size 2, stride 2)
- 1D convolution (60 filters, size 5, stride 1)
- BatchNorm + ReLU activation
- Max-pooling (size 2, stride 2)
- 1D convolution (120 filters, size 3, stride 1)
- BatchNorm + ReLU activation
- Max-pooling (size 2, stride 2)
- Fully-connected layer (256 units)
- BatchNorm + ReLU activation
- Dropout (0.4)
- Fully-connected layer (256 units)
- BatchNorm + ReLU activation
- Dropout (0.4)
- Fully-connected output layer (2 units, linear activation)

The DeepSTARR model was trained using EvoAug, an evolution-inspired data augmentations that was shown to lead to SOTA performance. We acquired weights from `DeepSTARR_combo1_finetune_2.ckpt` located on `https://zenodo.org/records/7767325`. The augmentation hyperparameters for this model were set to:

- mutation: mutate_frac = 0.05

- translocation: shift_min = 0, shift_max = 20
- insertion: insert_min = 0, insert_max = 20
- deletion: delete_min = 0, delete_max = 30
- reverse-complement: rc_prob = 0
- noise: noise_mean = 0, noise_std = 0.3

with a maximum number of augmentations set to 2. Dataset was acquired from `https://zenodo.org/records/7265991`.

Cell-type-specific cis-regulatory sequences. The MPRA dataset was curated by [16], which is a compilation of several MPRA experiments for K562, HepG2, and SK-N-SH cell lines from ENCODE [75]. We acquired and processed the data from Supplementary Table 2 of [16]. The input data are 200 nt genomic sequences with reference or alternate alleles for genomic variants, and the target is a scalar value (for each cell line) representing experimentally measured activities. Sequences from chromosomes 7 and 13 were used for testing (63,958 total), chromosomes 19, 21, and X were for validation (59,697 total), and all of the other sequences were used for training (640,029 total).

Our custom model is a variation of ResidualBind [63, 64], consisting of convolutional layers, residual blocks, and dense layers. The model takes an input with a specified shape and applies the following layers:

- Input layer
- Conv1D(196, 15) - BatchNorm - Activation('exponential') - Dropout(0.1)
- Residual Block(196, 3, 'relu', num_dilation_factors=4)
- Dropout(0.2) - MaxPooling1D(4)
- Conv1D(256, 5) - BatchNorm - Activation('relu') - Dropout(0.1)
- Residual Block(256, 3, 'relu', num_dilation_factors=3)
- MaxPooling1D(4) - Dropout(0.2)
- Conv1D(256, 3) - BatchNorm - Activation('relu') - Dropout(0.1)
- Residual Block(256, 3, 'relu', num_dilation_factors=1)
- MaxPooling1D(2) - Dropout(0.2)
- Flatten()
- Dense(512) - BatchNorm - Activation('relu') - Dropout(0.5)
- Dense(256) - BatchNorm - Activation('relu') - Dropout(0.5)
- Dense(1, activation='linear')

The model uses exponential activations in the first layer as it has been shown to lead to more interpretable convolutional kernels and improved efficacy with attribution analysis [76]. The residual block consists of a series of convolutional layers with varying dilation rates, batch normalization, and activation functions, according to:

- Input layer
- Conv1D(num_filters, filter_size, dilation_rate=1) - BatchNorm - Activation('relu')
- Dropout(0.1)
- For each dilation factor(1, 2, 4, 8)[:num_dilation_factor]:
  - Conv1D(num_filters, filter_size, dilation_rate=factor) - BatchNorm - Activation('relu')
- Add input layer and the output of the last convolutional layer
- Activation('relu')

where the number of dilation factors subsamples the dilation factor list, the output of the residual block is then added to the input layer, creating a residual connection.

ResidualBind was trained for 100 epochs using the Adam optimizer [77] with an initial learning rate of 1×10^{-3}. The learning rate was decayed by a factor of 0.2 when the validation loss did not improve for five epochs. Early stopping with a patience of 10 epochs was used. The model

checkpoint with the lowest validation loss was chosen for downstream analysis. A separate model was trained for each cell line. Notably, the performance of ResidualBind for K562 (Pearson's r = 0.839), HepG2 (Pearson's r = 0.839), and SK-N-SH (Pearson's r = 0.827) was higher than what was achieved by the models explored in [78]. Nevertheless, nuances in data processing and training make direct comparisons not straightforward (even with the same data splits).

D Fly enhancer design: additional analysis

Table 4: Conditional sampling diversity for fly enhancers. 500 sequences were generated using a fixed conditional value of $[7.045,\ 1.973]$ for [developmental task, housekeeping task] (from test sequence index=21,106) and assessed according to sequence diversity and mechanistic diversity.

Model	Sequence Diversity (percent identity)	Mechanistic diversity (Euclidean distance)
DFM-Conv	23.9	0.428
DFM-Tran	25.0	0.425
3D-Conv	26.6	0.456
3D-Tran	26.5	0.472
Shuffle	29.5	0.436

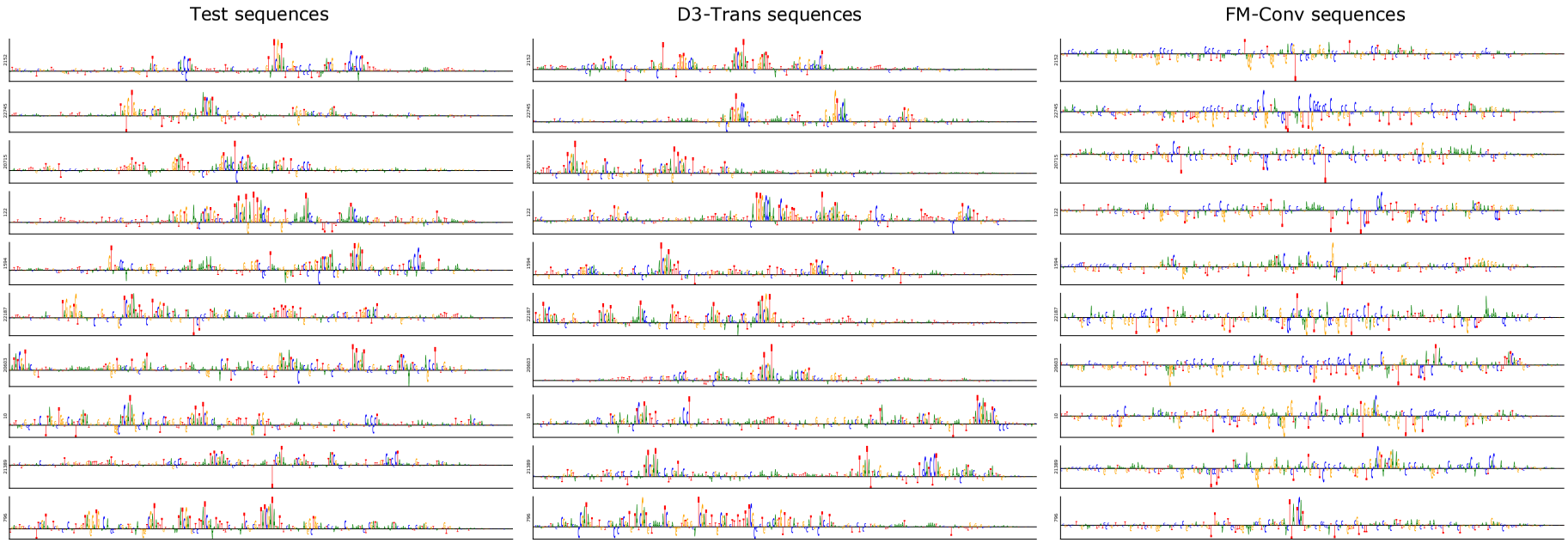

Figure 3: Attribution maps for test sequences (left), D3-Tran-generated sequences (middle), and DFM-Conv-generated sequences (right), with each row representing matched conditional activities based on the test sequence. Test and D3-Tran-generated sequences show contiguous motif-like patterns, while DFM-Conv-generated sequences have more spurious attribution noise. Attribution maps were calculated using GradientSHAP from Captum with an EvoAug-trained DeepSTARR oracle and visualized using Logomaker.

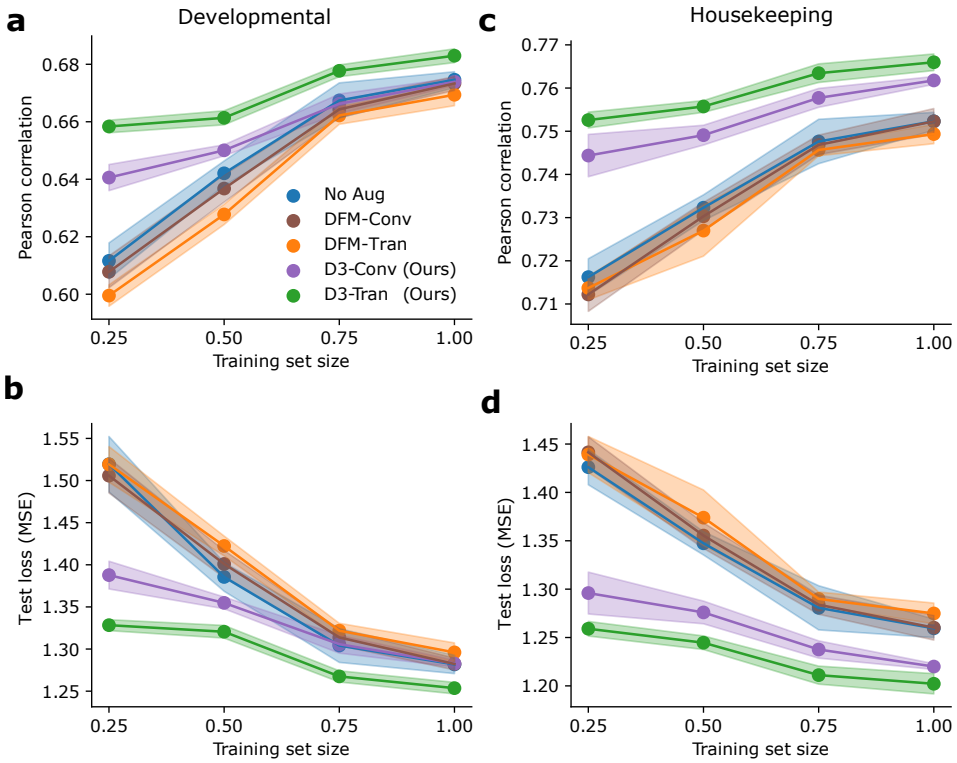

Figure 4: Augmentation analysis for synthetic fly enhancers. The performance of training a DeepSTARR model using different downsampled sets of training sequences (No Aug) and with the addition of synthetic sequences generated conditionally by different methods to match the activity of the full training set (shown in a different color) according to the Pearson correlation for developmental (**a**) and housekeeping (**c**) contexts and the test loss for developmental (**b**) and housekeeping (**d**) contexts. Plots display average values; shaded regions represent the standard deviation across five trials, each with a random initialization.

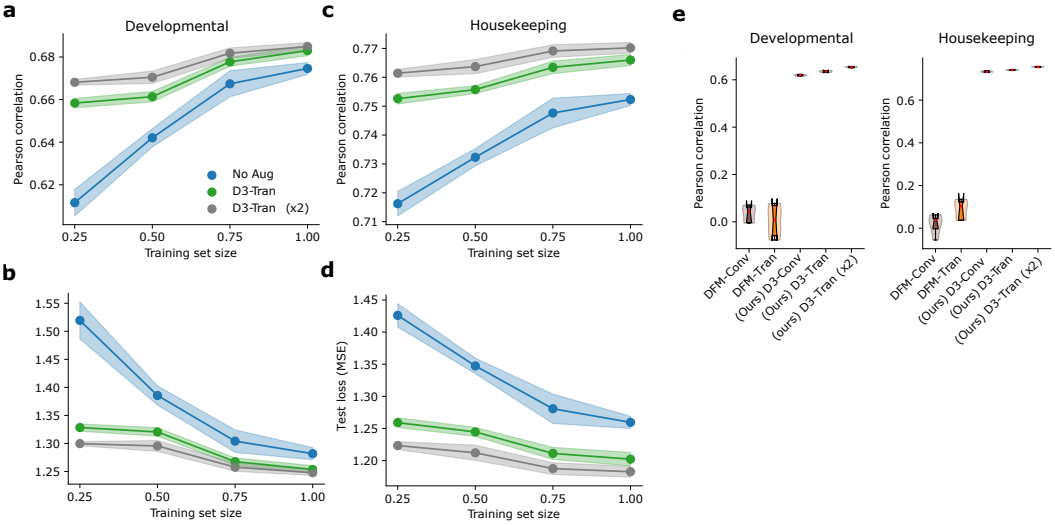

Figure 5: Augmentation analysis for synthetic fly enhancers. The performance of training a DeepSTARR model using different downsampled sets of training sequences (No Aug) and with the addition of synthetic sequences generated conditionally to match the activity of the full training set by D3-Tran with different multiples of the generated training set (shown in a different color) according to the Pearson correlation for developmental (**a**) and housekeeping (**c**) contexts and the test loss for developmental (**b**) and housekeeping (**d**) contexts. Plots display average values; shaded regions represent the standard deviation across five trials, each with a random initialization. **e** Boxplot of Pearson correlation for DeepSTARR models trained using only synthetic sequences from various methods, evaluated on test sequences.

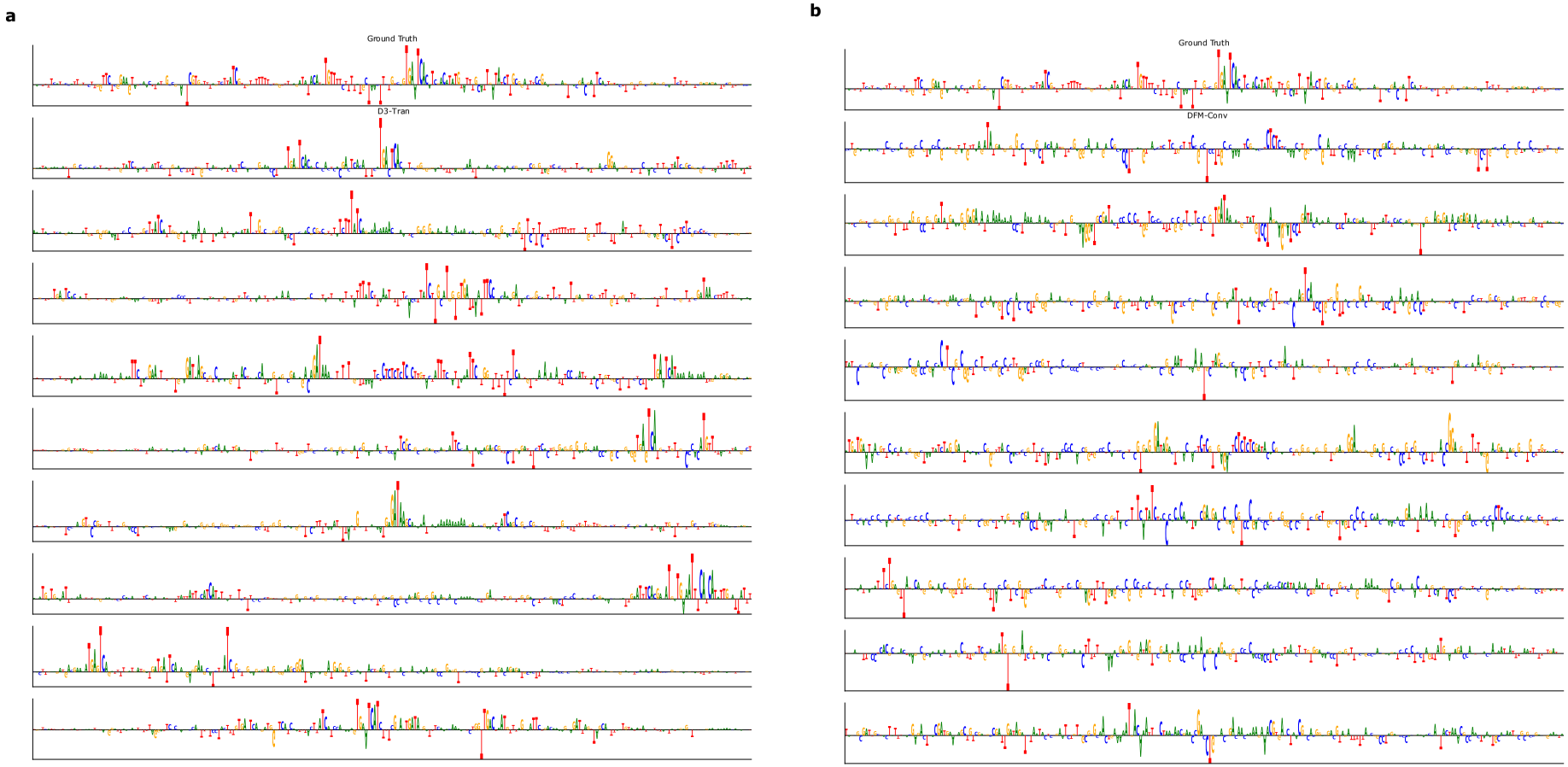

Figure 6: Attribution maps of generated sequences using the same conditional values for D3-Tran (**a**) and DFM-Conv (**b**). D3-Tran-generated sequences show contiguous motif-like patterns, while DFM-Conv-generated sequences have more spurious attribution noise. An attribution map for the wild type sequence is shown at the top for reference. Attribution maps were calculated using GradientSHAP from Captum with an EvoAug-trained DeepSTARR oracle and visualized using Logomaker. Importantly, the attribution maps in each row show very different underlying motif patterns, suggesting a different cis-regulatory mechanism that drives comparable activity levels.

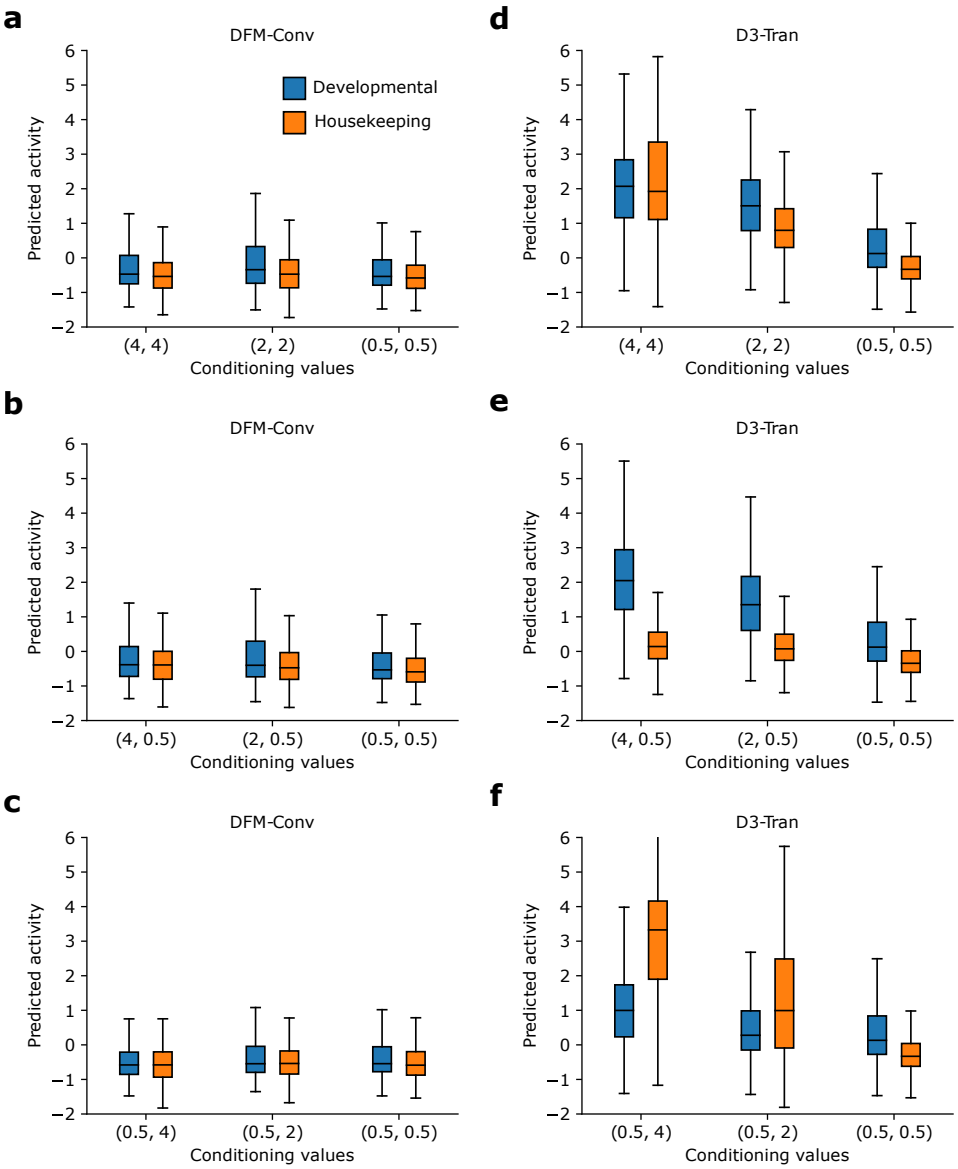

Figure 7: Task-specific design for fly enhancers. Comparison of predicted activities by EvoAug-trained DeepSTARR model for generated sequences conditioned on task-specific activities for DFM-Conv (left) and D3-Tran (right). Task activity values used for conditioning are shown on the x-ticks as (activity for developmental tasks and activity for housekeeping tasks). Each boxplot represents the task-specific predictions for 500 generated sequences. (**a**,**d**) Results when conditional activities are similar across cell types. (**b**,**c**,**e**,**f**) Results with differential conditional activities. D3-Tran generates sequences whose predicted activity more closely tracks the conditioning value trends, albeit the magnitude of the high activities is not as precise.

E Cell-type-specific Regulatory Sequence Design

Data. The dataset, created by [16], consists of cis-regulatory sequences with activity values measured via an MPRA in three human cell lines: K562 (erythroid precursors), HepG2 (hepatocytes), and SK-N-SH (neuroblastoma). It contains 640,029 training, 59,697 validation, and 63,958 test sequences. Each 200 nt sequence has three scalar activity values for each cell type. We used this dataset to train and evaluate generative models for cell-type-specific sequence activity.

Baselines. We used the same models as in the fly enhancer analysis (see Appendix C for details).

Evaluation. We evaluated the conditional generation fidelity from Sec. A and explored the ability to generate sequences with cell-type-specific activity levels. The oracle was a custom ResidualBind model [63, 64] with dilated convolutional blocks and residual connections (details in Appendix C.2).

Results. Similarly, D3-based models led to lower MSE in *conditional generation fidelity* (Table 5). For *task-specific design*, D3-Tran generated sequences with appropriate activities when the conditioning activity values across cell types were the same (Fig. 8a). However, D3-Tran yielded lower efficacy when generating sequences were conditioned with highly differential activities across cell types (Fig. 8b-d). Further investigation revealed that task labels in the MPRA training set were highly correlated, with high activity examples being extremely rare (Fig. 9). In contrast, the fly enhancer dataset exhibited lower activity correlation across tasks and contained more high-activity examples. This suggests that while D3-Tran has learned a useful distribution capturing many observed properties of sequence-function space, its ability to generate unseen combinations of task-specific activities is limited. However, this issue could be a limitation of the supervised oracle's inability to generalize under covariate shifts. Experimental evidence is needed to resolve this discrepancy, but it remains out of scope for this study.

Table 5: Results for cell-type-specific generation from MPRA data.

Model	Conditional generation fidelity (MSE)
DFM-Conv	2.243
DFM-Tran	1.620
D3-Conv	0.937
D3-Tran	0.929
Shuffle	1.824

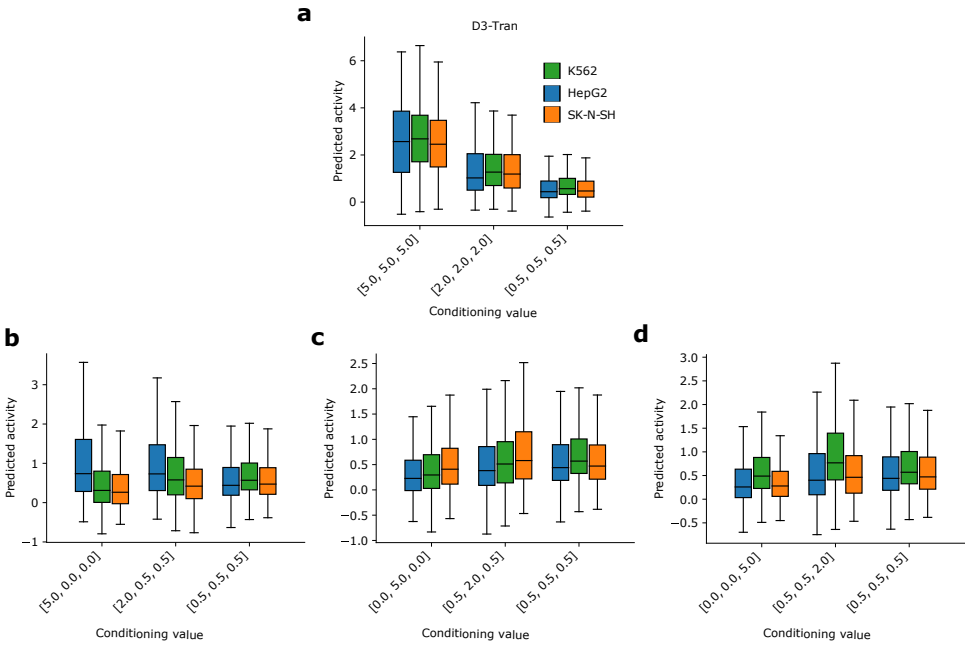

Figure 8: Task-specific design for cell-type-specific cis-regulatory sequences from MPRA data. Boxplots show oracle-predicted activities for sequences generated by D3-Tran, conditioned on values indicated by (K562 activity, HepG2 activity, SK-N-SH activity) on the x-axis. Each boxplot represents the corresponding cell type activities from 500 generated sequences. (**a**) Results when conditional activities are similar across cell types. (**b**-**d**) Results with differential conditional activities.

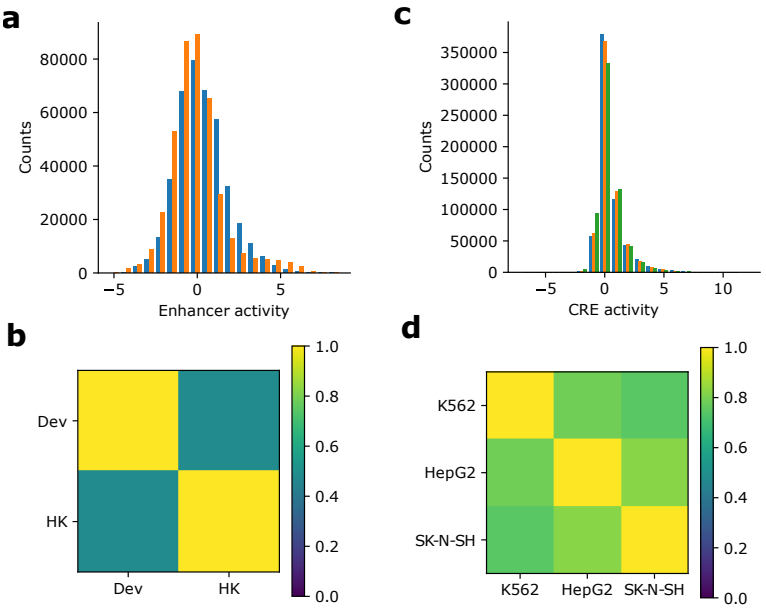

Figure 9: Label statistics for fly enhancer and cell-type-specific MPRA data. Histogram of label activities for each task within fly enhancers dataset (**a**) and cell-type-specific regulatory sequences (**c**). Correlation matrix of target activities of the training data for fly enhancers (**b**) and cell-type-specific regulatory sequences (**d**).

F D3-generated synthetic sequences improve supervised modeling

Motivation. A key question in evaluating generative models is whether the generated sequences capture meaningful biological mechanisms. If D3 effectively learns underlying cis-regulatory grammars, its synthetic sequences should provide complementary value when integrated with real data.

Experiment. Using the transformer-based D3 model trained on the 25% downsampled training set, we generated synthetic sequences conditioned on training set activities. These sequences were used to augment the training set by factors of 2x, 3x, and 4x. We then trained a DeepSTARR model from scratch on these augmented datasets and evaluated its performance using the Pearson correlation coefficient for both enhancer tasks on the original test set.

Methods. The transformer-based D3 trained on 25% downsampled training set was used as a baseline. To enhance performance in small-data settings, we explored transfer learning and data augmentation, both of which expose the model to broader cis-regulatory sequence patterns that may overlap with the fly enhancer dataset.

Transfer learning. We leveraged an ATAC-seq dataset from Drosophila embryos—previously used for pretraining supervised models for fly enhancer design[59]—as a pretraining dataset for D3. The dataset consists of 1,001-bp sequences with binary accessibility labels across 18 tissue pseudo-bulk ATAC-seq profiles. For training, we used the fold09 dataset, with the training set (471,536 sequences) drawn from all chromosomes except chrX, while validation and test sets were from chrX. During pretraining, D3 used the binary labels as conditional inputs. Fine-tuning was then performed on the downsampled STARR-seq dataset, with the learning rate reduced from 1e-4 to 1e-5.

Data augmentations. We applied EvoAug, a 2-stage evolution-inspired augmentation strategy that has demonstrated significant performance gains in supervised learning. During pretraining, EvoAug introduces random translocations, insertions, deletions, inversions, and mutations (using default parameters) to increase sequence diversity. While these transformations may alter sequence function, the assumption—aligned with the billboard model of gene regulation—is that key regulatory features are largely preserved. In the fine-tuning stage, the model was trained on the original dataset, refining its understanding of cis-regulatory grammar and motif interactions. This two-step process—first learning generalizable motifs through augmentation, then fine-tuning on unperturbed data—has been shown to improve performance in supervised tasks. Here, we assess whether EvoAug similarly enhances D3 generated synthetic sequences to improve supervised modeling in small-data regimes. A learning rate of 3e-4 was used during pretraining, followed by a reduction to 1e-5 during fine-tuning.

Results. Supervised models like DeepSTARR struggle in small-data regimes, as they rely on large datasets to learn complex cis-regulatory mechanisms. However, augmenting the training set with D3-generated sequences consistently improved DeepSTARR's predictive performance on both enhancer tasks (Fig. 10(b,c)). These results suggest that D3 has learned meaningful cis-regulatory features, enabling it to generate sequences that provide complementary information to real data, ultimately benefiting supervised models. Performance gains were observed across all augmentation levels (2x, 3x, and 4x the original dataset size), demonstrating that D3-generated sequences enhance regulatory signal learning in data-limited settings. Among the different D3 training strategies, EvoAug-trained D3 models produced the most substantial and consistent improvements, suggesting that EvoAug enhances the diversity and informativeness of generated sequences. In contrast, sequences generated by D3 models trained solely on 25% of the real data yielded more modest improvements, while ATAC-seq pretraining provided only limited additional benefit.

These findings indicate that D3 can generate functionally relevant cis-regulatory sequences that help supervised models overcome data limitations, improving their ability to capture meaningful sequence-function relationships. They also highlight the potential of data augmentation and transfer learning in enhancing the sequence quality generated by D3 in the small data regime. A major advantage of EvoAug is that it eliminates the need for a carefully curated pretraining dataset that closely matches the target task, which is often difficult or impossible to obtain. This makes EvoAug a promising approach for further exploration, as it could provide performance improvements even in cases where suitable pretraining data is unavailable. These methods underscore D3's potential as a synthetic data generation tool for advancing predictive modeling in regulatory genomics.

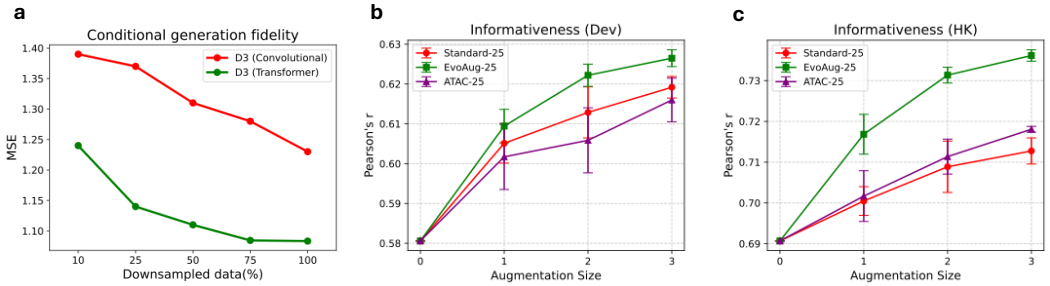

Figure 10: (**a**) Performance evaluation of D3 models in small data regime. We train with both convolution and transformer architecture, for each downsampled version, and evaluate on conditional generation fidelity averaged over developmental (Dev) and housekeeping (HK) enhancer task activities. (**b,c**) Informativeness analysis of D3 generated sequences with different training strategies. Downsampled STARR-seq dataset (25%) are used to train D3 models with (EvoAug and ATAC-seq) and without fine-tuning. Each of these trained models are then tasked to generate 3 sets of sequences conditioned on the enhancer activities of that downsampled 25% dataset. These generated sequences are augmented progressively to the original 25% dataset to create different sized dataset versions for each D3 model. Performance of all these trained prediction models through Pearson correlation coefficient for Dev and HK enhancer task activities are presented. Augmentation size refers to the sets of additional generated sequences. Each model training is repeated 5 times leading to error bars for every result in both these plots.

G Additional Figures

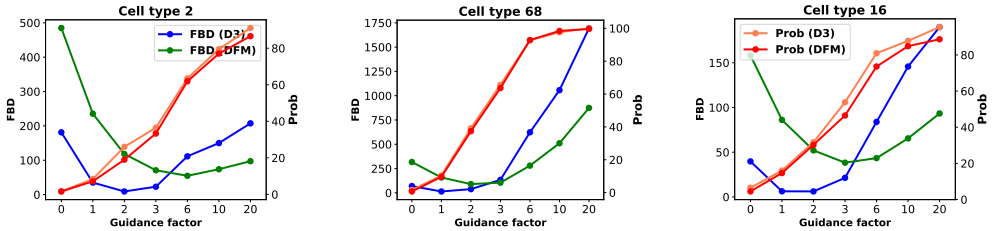

Figure 11: Classifier free guidance for cell type conditional enhancer design. Sequences are generated specifically for a cell type with DFM and D3 using different CFG levels. FBD and target class probabilities of the generated sequences for three different cell types i.e. 2, 68 and 16 are plotted here following [43].

Table 6: Performance of embeddings on fly enhancer activity prediction task for developmental (Dev) and Housekeeping (Hk) conditions from STARR-seq data using different probing strategies. DeepSTARR and a baseline CNN from Tang et al. [3] was trained using the full embeddings, while an MLP and ridge regression was trained using the mean embeddings. As a control, DeepSTARR and the CNN were also trained on one-hot encoded sequences.

Fly Enhancers (STARR-seq)

Probing method	**Embedding**	**Pearson's r (↑)**		**Spearman's ρ (↑)**	
		Dev	HK	Dev	HK
DeepSTARR	One-hot	0.664	0.741	0.635	0.560
	Evo2(7B)	0.526	0.568	0.519	0.455
	Evo2(40B)	0.578	0.723	0.562	0.530
	D3	**0.687**	**0.785**	**0.656**	**0.603**
CNN(Tang et al.)[3]	One-hot	0.652	0.742	0.629	0.565
	Evo2(7B)	0.611	0.721	0.590	0.537
	Evo2(40B)	0.582	0.726	0.566	0.537
	D3	**0.684**	**0.785**	**0.659**	**0.609**
MLP(Mean embedding)	Evo2(7B)	0.354	0.442	0.341	0.326
	D3	**0.578**	**0.763**	**0.562**	**0.561**
Ridge(Mean embedding)	Evo2(7B)	0.449	0.571	0.444	0.416
	D3	**0.563**	**0.755**	**0.555**	**0.545**

Table 7: Performance of embeddings on cell-type specific regulatory activity prediction task from lentiMPRA data using different probing strategies. A baseline CNN from Tang et al. [3] was trained using the full embeddings, while an MLP and ridge regression was trained using the mean embeddings. As a control, the CNN were also trained on one-hot encoded sequences.

LentiMPRA (Pearson's r)

Probing method	Embedding	K562	HepG2
CNN(Tang et al.)[3]	One-hot	0.691	0.638
	DNABERT2	0.568	0.434
	HyenaDNA	0.650	0.556
	GPN	0.678	0.637
	Evo2(7B)	0.666	0.587
	Evo2(40B)	0.696	0.616
	D3	**0.766**	**0.705**
MLP(Mean embedding)	DNABERT2	0.498	0.445
	HyenaDNA	0.459	0.397
	GPN	0.526	0.459
	Evo2(7B)	0.491	0.417
	Evo2(40B)	0.647	0.548
	D3	**0.659**	**0.627**
Ridge(Mean embedding)	DNABERT2	0.423	0.402
	HyenaDNA	0.380	0.336
	GPN	0.454	0.403
	Evo2(7B)	0.563	0.474
	Evo2(40B)	0.627	0.540
	D3	**0.639**	**0.615**