Omni-DNA: A Genomic Model Supporting Sequence Understanding, Long-context, and Textual Annotation

Zehui Li[1,2,*], **Vallijah Subasri**[3,4], **Yifei Shen**[2], **Dongsheng Li**[2], **Wentao Gu**[2], **Guy-Bart Stan**[1,†], **Yiren Zhao**[1,†], **Caihua Shan**[2,*]

[1]Imperial College London [2]Microsoft Research
[3]Vector Institute [4]University Health Network

[*]Correspondence to: `{zl6222@ic.ac.uk, caihua.shan@microsoft.com}`
[†]Co-corresponding authors: `{g.stan@imperial.ac.uk, a.zhao@imperial.ac.uk}`

Abstract

The interpretation of genomic sequences is crucial for understanding biological processes. To handle the growing volume of DNA sequence data, Genomic Foundation Models (GFMs) have been developed by adapting architectures and training paradigms from Large Language Models (LLMs). Despite their remarkable performance in DNA sequence classification tasks, there remains a lack of systematic understanding regarding the pre-training and task-adaptation processes of GFMs. Moreover, existing GFMs cannot achieve state-of-the-art performance on both short and long-context tasks and lack multimodal abilities.

By revisiting pre-training architectures and post-training techniques, we propose OMNI-DNA, a family of models spanning 20M to 1.1B parameters that supports sequence understanding, long-context genomic reasoning, and natural-language annotation. Omni-DNA establishes new state-of-the-art results on 18 of 26 evaluations drawn from Nucleotide Transformer and Genomic Benchmarks. When jointly finetuning on biologically related tasks, Omni-DNA consistently outperforms existing models and demonstrates multi-tasking abilities. Furthermore, we introduce SEQPACK, an adaptive compression mechanism that enables efficient long-context modeling by summarizing historical tokens through position-aware learnable sampling. This allows transformer-based models to process ultra-long genomic sequences with minimal memory and computational overhead. Leveraging SEQPACK, Omni-DNA excels at enhancer–target interaction prediction, capturing distal regulatory effects over 450kbp. Finally, we present SEQ2FUNC, a newly constructed dataset that empowers Omni-DNA to generate accurate and functionally meaningful interpretations of DNA sequences, opening new avenues for genomic analysis and discovery.

1 Introduction

As genomic data has been expanding exponentially over recent decades [19], Genomic Foundation Models (GFMs) have emerged as an essential paradigm to model genomic sequences and complete various tasks, including variant effect prediction [21, 44], sequence generation [39], and functional classification [10]. Most GFMs follow a "pretrain + task-specific finetune" paradigm: first pretraining on unlabeled DNA sequences, and finetuning for each specific task.

While GFMs have demonstrated utility on specific genomic tasks [13, 47], they still exhibit important limitations: **(i) Limited optimization of architectural components for genomic tasks:** Prior work has explored design choices including tokenization (DNABERT [17] vs. DNABERT-2 [49]), positional encodings (NT vs. NTv2 [10]), and genomic-specific modules (GPN-MSA [5], Caduceus [30]), but a more comprehensive, task-specific search of architectures and pretraining strategies tailored to genomic inductive biases is still lacking. **(ii) Limited exploration of multi-task synergies:** Many approaches fine-tune on a single downstream task, missing the opportunity to exploit shared biological signals and achieve cross-task performance gains, despite evidence of such benefits in genomic assay prediction (e.g., AlphaGenome [3], Enformer [2]). **(iii) Insufficient multimodal capability:** Most existing GFMs restrict vocabularies to nucleotide base pairs, predefined labels, or learned subwords via BPE tokenization, limiting their ability to handle tasks that require mapping DNA sequences to functional annotations in natural language. While ChatNT can process joint (DNA, text) inputs, it employs separate tokenizers and a two-stage architecture consisting of NTv2 [10] as the DNA encoder and Vicuna-7B [38] as the decoder, resulting in less unified multimodal integration. **(iv) Context-length adaptivity challenges:** Transformer-based models remain constrained by the quadratic cost of self-attention, limiting their scalability to long-context genomic tasks such as enhancer–target prediction.

To bring DNA sequence modeling closer to the breakthroughs that large language models have enabled in natural language processing, we comprehensively explored the design space of GFMs, from pre-training architectures and objectives to post-training techniques, and developed several targeted improvements. To this end, we propose Omni-DNA (Figure 1), a family of GFMs ranging from 20M to 1.1B parameters that enables multitask learning, long-context genomic reasoning, and natural-language annotation. We then outline how Omni-DNA addresses these challenges.

Optimized backbone (Section 2). Early GFMs such as DNABERT [17], DNABERT-2 [49] and Nucleotide Transformer [10] rely on masked language modeling within a BERT framework; recent models, including HyenaDNA [24], BioxLSTM [32], DNAGPT [46], and MegaDNA [36], have shifted to causal language modeling. For the scaling of data and models, Omni-DNA is also a decoder-only Transformer trained with causal language modeling on 300 billion nucleotides from RefSeq [33]. Performing an extensive architecture sweep, we further analyze and ablate normalization (LayerNorm vs. RMSNorm) and positional encoding (RoPE vs. ALiBi) to select the most robust configuration.

Unified multi-task finetuning (Section 3.1). In the post-training phase, the standard approach is to append a classification head (an MLP) and fine-tune the model independently on each downstream dataset. Using this conventional setup, Omni-DNA achieves state-of-the-art performance on 18 of 26 tasks in both the Nucleotide Transformer [9] and GB benchmarks [13]. We further extend this approach by grouping biologically related tasks and performing joint fine-tuning. The multitask fine-tuning achieves substantial improvements across multiple tasks and obtains the highest overall average accuracy. It demonstrates that Omni-DNA captures transferable biological knowledge during pre-training and exhibits the synergistic effect in the fine-tuning.

Since Omni-DNA is auto-regressive, it can naturally accommodate new tokens as instructions or labels. This enables the use of instruction-following techniques, such as supervised fine-tuning, to enrich the model's vocabulary and enrich the model's vocabulary for more flexible genomic reasoning.

Effective vocabulary expansion (Section 3.2). Because most GFMs are pre-trained on unlabeled genomes, their vocabularies are typically restricted to four canonical nucleotides (A, T, C, G) plus a handful of special tokens [9]. In contrast, LLMs routinely operate with vocabularies exceeding 10^4 sub-word units [25, 41]. This restricts GFM's capacity to both comprehend and generate sequence-level semantics, and conduct the multi-model tasks. To address this, we analyze how to mitigate the distribution shift of vocabulary when adding unseen tokens during post-training. We successfully create a large-scale dataset of text-function mapping, which enables Omni-DNA to directly generate natural functional descriptions of given DNA sequences.

Scalable long-context reasoning (Section 3.3). Our final improvement focuses on extending the model's effective context length. Prior approaches such as Caduceus [30], HyenaDNA [24], Evo [22] and MegaDNA [36] replace the standard Transformer with state-space architectures, hierarchical compression, or convolution-based operators to achieve sub-quadratic complexity. However, the whole genome includes long repetitive regions and short meaningful elements (e.g., genes and regulatory regions). These methods do not explicitly exploit such sparsity, limiting their effectiveness at the whole-genome scale. To address this gap, we introduce SEQPACK, a post-training context-extension operator that learns to compress historical tokens into a compact synopsis. Specifically, SeqPack adaptively compresses a DNA sequence of length N into an embedding of length L ($L \ll N$), reducing the complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(L^2)$. By learning to ignore "junk" regions [27] and retain only informative segments, SeqPack enables efficient ultra-long genomic reasoning while preserving critical positional information.

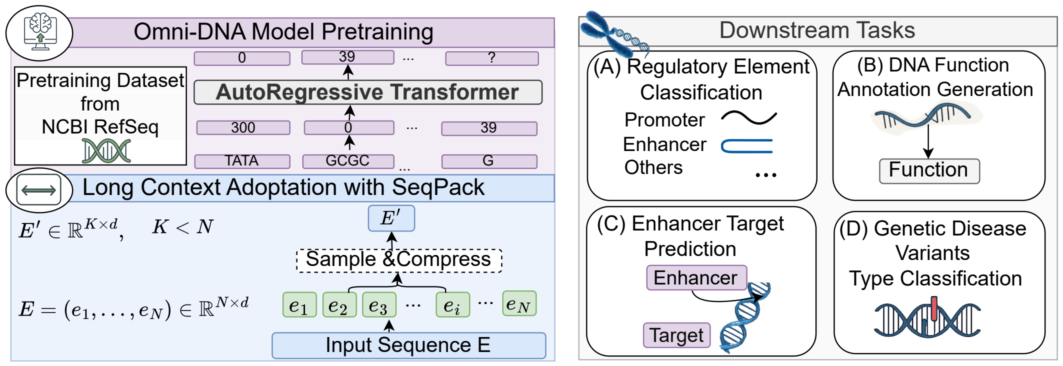

Omni-DNA Model Pretraining
Pretraining Dataset from NCBI RefSeq
AutoRegressive Transformer
0
39
?
300
0
39
TATA
GCGC
G
Long Context Adaptation with SeqPack
$E' \in \mathbb{R}^{K\times d}, \quad K < N$
E'
Sample &Compress
$E = (e_1, \dots, e_N) \in \mathbb{R}^{N\times d}$
e_1 e_2 e_3 ⋯ e_i ⋯ e_N
Input Sequence E
Downstream Tasks
(A) Regulatory Element Classification
Promoter
Enhancer
Others
(B) DNA Function Annotation Generation
Function
(C) Enhancer Target Prediction
Enhancer
Target
(D) Genetic Disease Variants Type Classification

Figure 1: **Overview of the Omni-DNA framework.** Omni-DNA combines large-scale autoregressive pretraining on 300 billion nucleotides with post-training compression via SeqPack to enable efficient long-context DNA modeling. The pretrained model is evaluated separately across diverse downstream tasks, including (A) regulatory element classification, (B) DNA function annotation, (C) enhancer–target interaction prediction, and (D) disease type classification based on genetic variants.

2 Revisiting DNA Sequence Pretraining Paradigms

2.1 Pretraining Setup

Omni-DNA is pre-trained with causal next-token prediction objective on a 300-billion–nucleotide corpus drawn from NCBI's multi-species assemblies [33]. Data deduplication procedures are described in Appendix C. Following Zhou et al. [48], we adopt Byte-Pair Encoding (BPE) [35] with an initial vocabulary of 4096. Each model processes 250-token contexts ($\approx$1kbp) in batches of 384 sequences across 8 × A100-40 GB GPUs for 800k steps. The optimization is performed by AdamW ($\beta_1 = 0.9,\ \beta_2 = 0.95$, weight-decay=0.1) with a linear warm-up of 5k steps to a peak learning rate of 3×10^{-4}, followed by linear decay to 3×10^{-5}. We also employ mixed precision (bf16 matmuls, fp32 norm stats), global gradient clipping at 1.0, and PyTorch FSDP layer sharding.

2.2 Model Architecture

Following Table 1, Omni-DNA adopts a decoder-only Transformer backbone [14], but is *not* a naive transplant of NLP models. We conduct three kinds of ablation studies: normalization layer (bias-free LayerNorm [4] vs. RMSNorm [45]), positional encoding (ALiBi [29] vs. RoPE [40]), and parameter budget. The optimizer, learning-rate scheduler, and tokenizer are kept identical.

Table 1: Architecture of pretrained Omni-DNA models.

# Params	Layers	Heads	d_{model}	Layer Norm	Pos. Emb.
20M	8	8	256	RMSNorm	RoPE
60M	8	8	512	RMSNorm	RoPE
116M	12	12	768	Bias-Free LN	RoPE
116M	12	12	768	RMSNorm	RoPE
116M	12	12	768	Bias-Free LN	ALiBi
116M	12	12	768	RMSNorm	ALiBi
300M	16	16	1024	RMSNorm	RoPE
700M	16	16	1536	RMSNorm	RoPE
1B	16	16	2048	Bias-Free LN	RoPE

2.3 Empirical Insights

Finding 1: The combination of RoPE and Bias-Free LayerNorm deliver the best trade-off between compute efficiency and downstream accuracy, whereas ALiBi consistently converges $\sim$15% slower and lags on both promoter and enhancer tasks.

Explanation. As shown in Figure 2 (*Left*), all variants exhibit lower training loss with increasing FLOPs, but those using RoPE (green, red) descend more steeply than their ALiBi counterparts (blue, orange). When transferred to the performance of downstream tasks in Figure 2 (*Middle* and *Right*), such as TATA-box recognition and enhancer-activity prediction, RoPE+Bias-Free LayerNorm attains 0.967 ± 0.002 and 0.594 ± 0.006 accuracy, surpassing all other models. These results suggest that relative phase encoding (RoPE) captures long-range DNA dependencies more effectively than the linear bias of ALiBi, especially when coupled with the unbiased normalizer.

Finding 2: With the same compute larger models tend to obtain lower test loss; however, lower test loss *does not* guarantee higher downstream accuracy

Explanation. Figure 3 (*Left*) follows the similar compute–loss scaling law [18] extended from LLMs: with the same compute budget, larger models attain lower test loss (cross-entropy). Yet Figure 3 (*Right*) reveals a decoupling between losses and downstream performance: histone-modification and enhancer tasks saturate once the loss reaches ≈ 4.1, while promoter and splice tasks remain near-ceiling even for the smallest models (The downstream tasks refer to 18 tasks from Dalla-Torre et al. [9]; finetuning details are in Section 4.1.) This mismatch cautions against relying solely on the loss as a proxy for downstream biological performance.

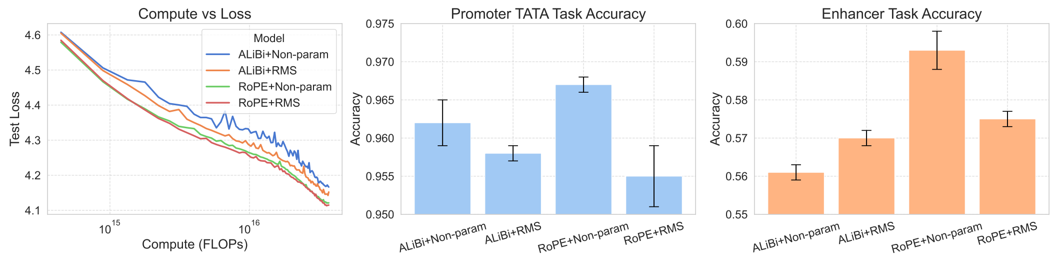


Figure 2: **The ablation of pretraining architectures. Left:** Test loss as a function of cumulative pre-training compute (FLOPs, log–scale) for four 116M-parameter variants that combine ALiBi or RoPE positional biases with bias-free LayerNorm (*Non-param*) or RMSNorm. **Middle:** Promoter TATA site classification accuracy **Right:** Enhancer activity prediction accuracy. Bars show mean ± std over three fine-tuning runs.

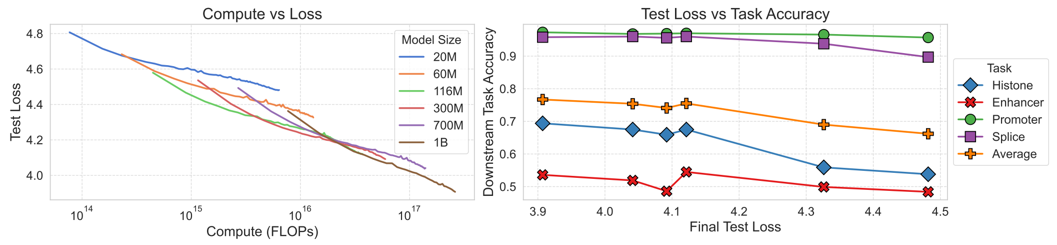


Figure 3: **Compute–Loss Scaling and Its Decoupling from Downstream Accuracy. Left:** Pre-training loss decreases predictably with total compute as model size increases from 20M to 1B parameters, following an approximate power law. **Right:** Final test loss poorly predicts the fine-tuned performance on four representative tasks (Histone, Enhancer, Promoter, Splice).

3 Adaptive Post-Training Strategies for GFMs

Here we first explore the new multi-task setting in Section 3.1. Subsequently, we propose novel approaches for vocabulary expansion and long-context compression in Section 3.2 and Section 3.3.

3.1 Single-task vs. Multi-task Setting

Current fine-tuning in GFMs for a specific task typically is attaching a linear head (an MLP) to the hidden state of the last token from the final layer, which is then used to predict the class. In the single-task setting, the model is fine-tuned separately for each individual task. In contrast, for the multi-task setting, we first group k related tasks, and jointly fine-tune these k linear heads.

Moreover, since Omni-DNA is auto-regressive, an alternative fine-tuning approach is to introduce new tokens instructions and labels and apply instruction-following techniques, such as supervised fine-tuning (SFT). For multiple tasks, we add task-specific prompt tokens and class label tokens, enabling the model to learn k tasks simultaneously.

3.2 Vocabulary Expansion

To enable the model to handle prompts, answer tokens, or label words that are *absent* from the pre-training alphabet during SFT, the original vocabulary V_o must be expanded to $V = V_o \cup V_z$, where V_z represents the newly introduced symbols (e.g., task tags, response delimiters, class labels).

Vocabulary distribution shift problem. Let $h_{t-1} \in \mathbb{R}^d$ be the decoder state at step $t-1$ and $E = [\,e_v\,]_{v \in V} \in \mathbb{R}^{d \times |V|}$ the output-embedding matrix whose *columns* are the token vectors e_v. With the expanded vocabulary, the softmax becomes

$$p_\theta(x_t \mid x_{1:t-1}) \;=\; \frac{\exp\bigl(h_{t-1}^\top e_{x_t}\bigr)}{\underbrace{\sum_{m \in V_o} \exp\bigl(h_{t-1}^\top e_m\bigr)}_{Z} \;+\; \underbrace{\sum_{n \in V_z} \exp\bigl(h_{t-1}^\top e_n\bigr)}_{Z'}}. \tag{1}$$

For a pre-existing token $x_t \in V_o$, its probability under the original softmax would have been $p_\theta^{\text{old}}(x_t \mid x_{1:t-1}) = \exp(h_{t-1}^\top e_{x_t})/Z$. Comparing with Equation 1 yields the shrinkage factor:

$$p_\theta(x_t \mid x_{1:t-1}) \;=\; p_\theta^{\text{old}}(x_t \mid x_{1:t-1}) \times \frac{Z}{Z + Z'} \;=\; p_\theta^{\text{old}}(x_t \mid x_{1:t-1}) \times \frac{1}{1 + Z'/Z}. \tag{2}$$

Thus every *pre-existing* token probability is decayed by $1/(1 + Z'/Z)$—a shift that can slow convergence and, if unaddressed, cause catastrophic forgetting.

To mitigate distribution shift, we integrate two techniques: **(1) NEFTune** [16] : isotropic Gaussian noise is added to all token embeddings during SFT, acting as dropout in representation space and discouraging the model from over-relying on a few high-frequency additions. **(2) Key-token replication**: for classification tasks, we duplicate each label token α times (we use $\alpha = 5$) within the target string. This boosts its relative frequency, ensuring that gradients from the label position dominate those from task prompts. In practice, we also keep $|V_z|$ to the minimum required.

3.3 SeqPack: Long-Context Sequence Compression

Let $x_{1:N} = (x_1, \dots, x_N)$ denote a DNA sequence of length $N \gg L_{\text{nat}}$, where L_{nat} is the *native context length* of a pretrained backbone f_θ (typically 1–16k tokens). We tokenize DNA sequences into k-mers and apply the embedding table $\Phi : \mathcal{V} \to \mathbb{R}^d$ to obtain the embedding matrix $E = \Phi(x_{1:N}) \in \mathbb{R}^{N \times d}$. The objective of SeqPack is to learn a differentiable compression operator

$$\mathcal{C}_L : \mathbb{R}^{N \times d} \longrightarrow \mathbb{R}^{K \times d}, K \le L_{\text{nat}} \tag{3}$$

which can identify and preserve the most informative K tokens based on their positions and content before the computation-heavy layers.

3.3.1 Position-Aware Learnable Sampling

To realize the compression operator $\mathcal{C}_L$, we estimate the token importance by a trainable *temperature vector* $\theta \in \mathbb{R}^N$. In detail, the importance weight of each position i is defined by

$$\pi_i = \frac{\exp(\theta_i)}{\sum_{j=1}^N \exp(\theta_j)}. \tag{4}$$

Because hard sampling is non-differentiable, we employ the Gumbel-Softmax trick [15]:

$$\text{logit}_i \;=\; \log \pi_i \;+\; g_i, \qquad g_i \sim \text{Gumbel}(0, 1), \tag{5}$$

followed by a top-K operator that keeps the K highest-scoring indices. Concatenating the selected rows of E with a fixed suffix of the most recent K tokens yields the compressed representation $E' = \mathcal{C}_L(E)$, guaranteeing that the downstream backbone receives at most K tokens.

3.3.2 Geometric Splitting

Applying a global top-K over N tokens costs $O(N \log K)$ time, which is untenable for ultra-long sequences. To avoid this bottleneck, we introduce a *geometric splitting* scheduler that partitions the sequence into exponentially growing segments and performs local sampling within each segment.

Algorithm 1 SEQPACK compression layer (inference mode)

```
Require: Embeddings $E \in \mathbb{R}^{N\times d}$, the number of preserved tail tokens $K_{\text{tail}}$, the total preserved tokens $K$, the learned temperature vector $\theta$
Ensure: Compressed sequence $E' \in \mathbb{R}^{K\times d}$
 1: $H \leftarrow N - K_{\text{tail}} - 1$        ▷ history length (excluding $e_1$ and tailed tokens)
 2: $B \leftarrow K - K_{\text{tail}} - 1$        ▷ history budget
 3: if $H \le B$ then
 4:     return $E$        ▷ no compression needed
 5: end if
 6: Segment $e_{2:N-K_{\text{tail}}}$ into geometric blocks $\{S_i\}_{i=1}^{M}$
 7: for all $S_i$ do
 8:     Compute the sampling budget $b_i$ via Equation 7
 9:     Compute the importance weight $\pi_j^i$ via Equation 8
10:     Sample to obtain compressed output $E_i'$ based on $b_i$ and $\pi_j^i$
11: end for
12: $E' \leftarrow [\, e_1 \,;\, E_1' \,;\, \dots \,;\, E_M' \,;\, e_{N-K_{\text{tail}}:N} \,]$
13: return $E'$
```

Given input embeddings $E \in \mathbb{R}^{N\times d}$ and compression target K, we divide the sequence into $M = \lceil \log_2(N) \rceil$ segments with exponentially increasing sizes. The starting point of these segments $i \in [1, M-1]$ is

$$p_{i+1} = p_i + \min(2^{i-1}, N - p_i) \tag{6}$$

where the boundary conditions are $p_1 = 1$ and $p_{M+1} = N+1$. For each segment $S_i = E_{[p_i:p_{i+1}-1]}$, we allocate a sampling budget based on the length:

$$b_i = \max\left(1, |S_i| \cdot \frac{K}{N}\right). \tag{7}$$

We then sample b_i tokens from each segment according to the segment-normalized importance weights following Equation 4:

$$\pi_j^i = \frac{\exp(\theta_j)}{\sum_{k=p_i}^{p_{i+1}-1} \exp(\theta_k)} \qquad j \in [p_i, p_{i+1}-1], \tag{8}$$

From each segment, we obtain sampled E_i' and the final compressed output is $E' = [E_1'; \dots; E_M']$. To further analyze the efficiency of geometric splitting, we present Proposition 1 with the proof in Appendix B, which establishes the linear-time complexity of the sampling process.

Proposition 1 (Linear-time compression under geometric splitting)**.** *Let $E \in \mathbb{R}^{N\times d}$ be token embeddings and let $K \le L_{nat}$ be the target context length. Using the geometric splitting scheduler defined above, the expected time to obtain the compressed sequence $E' \in \mathbb{R}^{K\times d}$ is $\Theta(N)$.*

3.3.3 Compression with Positional Bias Preservation

We implement SEQPACK as a lightweight PyTorch layer that is inserted *once* before the first attention block during post-training. To preserve positional inductive biases, we exclude the first token and the final K_{tail} tokens from compression. This design is motivated by the attention sink phenomenon [42], where attention to the first token can dominate attention scores. In DNA-pretrained transformers, we observe similar sink effects at both the beginning and end of the sequence (Appendix G). The overall procedure is described in Algorithm 1.

4 Experiments

We benchmark Omni-DNA integrated with SEQPACK against leading genomic foundation models across four types of representative tasks. (i) Gene regulatory element classification: identify functional regions that control gene expression; (ii) Pathogenic variant effect prediction: assess the functional impact of genetic mutations linked to disease; (iii) Enhancer–target gene interaction mapping: resolve the complex spatial relationships between regulatory elements and their target genes; and (iv) Sequence-conditioned functional interpretation generation: a novel task introduced in this work, supported by a carefully curated dataset derived from high-quality NCBI genome assemblies.

Table 2: **Comparison of Omni-DNA with existing GFMs on 18 NT downstream tasks.** The family of Omni-DNA obtains the best average result. In splice tasks, Omni-DNA performs lower than NT models but still surpasses other models.

Metric	CADUCEUS-PH (1.9 M)	CADUCEUS-PS (1.9 M)	DNABERT2 (117 M)	NTv2 (500 M)	NT2.5B (2.5 B)	HyenaDNA (1.6 M)	Omni-DNA (116 M)	Omni-DNA (1 B)
H3	0.815 ± 0.048	0.799 ± 0.029	0.785 ± 0.033	0.784 ± 0.047	0.814	0.779 ± 0.037	0.818 ± 0.005	$\mathbf{0.824} \pm 0.032$
H3K14AC	0.631 ± 0.026	0.541 ± 0.212	0.516 ± 0.028	0.551 ± 0.021	0.550	0.612 ± 0.065	0.685 ± 0.014	$\mathbf{0.697} \pm 0.077$
H3K36ME3	0.601 ± 0.129	0.609 ± 0.109	0.591 ± 0.020	0.625 ± 0.013	0.632	0.613 ± 0.041	0.661 ± 0.013	$\mathbf{0.686} \pm 0.002$
H3K4ME1	0.523 ± 0.039	0.488 ± 0.102	0.511 ± 0.028	0.550 ± 0.021	0.559	0.512 ± 0.024	0.577 ± 0.083	$\mathbf{0.617} \pm 0.000$
H3K4ME2	0.487 ± 0.170	0.388 ± 0.101	0.336 ± 0.040	0.319 ± 0.045	0.326	0.455 ± 0.095	$\mathbf{0.576} \pm 0.003$	0.547 ± 0.006
H3K4ME3	0.544 ± 0.045	0.440 ± 0.202	0.352 ± 0.077	0.410 ± 0.033	0.421	0.549 ± 0.056	0.587 ± 0.222	$\mathbf{0.642} \pm 0.001$
H3K79ME3	0.697 ± 0.077	0.676 ± 0.026	0.613 ± 0.030	0.626 ± 0.026	0.642	0.672 ± 0.048	0.718 ± 0.027	$\mathbf{0.752} \pm 0.007$
H3K9AC	0.622 ± 0.030	0.604 ± 0.048	0.542 ± 0.029	0.562 ± 0.040	0.575	0.581 ± 0.061	0.658 ± 0.029	$\mathbf{0.701} \pm 0.002$
H4	0.811 ± 0.022	0.789 ± 0.020	0.796 ± 0.027	0.799 ± 0.025	0.822	0.763 ± 0.044	0.802 ± 0.002	$\mathbf{0.822} \pm 0.005$
H4AC	0.621 ± 0.054	0.525 ± 0.240	0.463 ± 0.041	0.495 ± 0.032	0.501	0.564 ± 0.038	$\mathbf{0.663} \pm 0.029$	0.652 ± 0.001
Enhancer	0.546 ± 0.073	0.491 ± 0.066	0.516 ± 0.098	0.548 ± 0.144	0.580	0.517 ± 0.117	$\mathbf{0.593} \pm 0.005$	0.580 ± 0.018
Enhancer Types	0.439 ± 0.054	0.416 ± 0.095	0.423 ± 0.051	0.424 ± 0.132	0.474	0.386 ± 0.185	0.498 ± 0.001	$\mathbf{0.492} \pm 0.023$
Promoter: ALL	0.970 ± 0.004	0.967 ± 0.004	0.971 ± 0.006	0.976 ± 0.006	$\mathbf{0.974}$	0.960 ± 0.005	0.973 ± 0.002	0.973 ± 0.001
Promoter: NONTATA	0.969 ± 0.011	0.968 ± 0.006	0.972 ± 0.005	0.976 ± 0.005	$\mathbf{0.977}$	0.959 ± 0.008	0.972 ± 0.016	0.975 ± 0.001
Promoter: TATA	0.953 ± 0.016	0.957 ± 0.015	0.955 ± 0.021	0.966 ± 0.013	0.964	0.944 ± 0.040	0.967 ± 0.001	$\mathbf{0.973} \pm 0.002$
Splice All	0.940 ± 0.027	0.927 ± 0.021	0.939 ± 0.009	0.983 ± 0.008	$\mathbf{0.983}$	0.956 ± 0.011	0.927 ± 0.025	0.941 ± 0.003
Splice Acceptor	0.937 ± 0.033	0.936 ± 0.077	0.975 ± 0.006	0.981 ± 0.011	$\mathbf{0.990}$	0.958 ± 0.010	0.968 ± 0.002	0.972 ± 0.001
Splice Donor	0.948 ± 0.025	0.874 ± 0.289	0.963 ± 0.006	$\mathbf{0.985} \pm 0.022$	0.984	0.949 ± 0.024	0.951 ± 0.007	0.963 ± 0.001
Average	0.715	0.689	0.679	0.698	0.709	0.707	0.755	$\mathbf{0.767}$

Table 3: **Performance of Omni-DNA across 8 tasks in the genomic benchmark.** Omni-DNA (116M) achieves the highest average.

Task Name	**CNN (264K)**	**HYENADNA (436K)**	**CADUCEUS -PH (470K)**	**DNABERT2 (117M)**	**Omni-DNA (116M)**
MOUSE ENHANCERS	0.715 ± 0.087	0.780 ± 0.025	0.754 ± 0.074	0.792 ± 0.031	$\mathbf{0.799} \pm 0.004$
CODING VS. INTERGENOMIC	0.892 ± 0.008	0.904 ± 0.005	0.915 ± 0.003	$\mathbf{0.949} \pm 0.002$	0.942 ± 0.010
HUMAN VS. WORM	0.942 ± 0.002	0.964 ± 0.002	0.973 ± 0.001	0.975 ± 0.002	$\mathbf{0.976} \pm 0.001$
HUMAN ENHANCERS COHN	0.702 ± 0.021	0.729 ± 0.014	$\mathbf{0.747} \pm 0.004$	0.714 ± 0.025	0.738 ± 0.002
HUMAN ENHANCER ENSEMBL	0.744 ± 0.122	0.849 ± 0.006	0.893 ± 0.008	0.891 ± 0.051	$\mathbf{0.919} \pm 0.021$
HUMAN REGULATORY	0.872 ± 0.005	0.869 ± 0.012	0.872 ± 0.011	0.852 ± 0.024	$\mathbf{0.895} \pm 0.012$
HUMAN OCR ENSEMBL	0.698 ± 0.013	0.783 ± 0.007	$\mathbf{0.828} \pm 0.006$	0.789 ± 0.012	0.791 ± 0.001
HUMAN NONTATA PROMOTERS	0.861 ± 0.009	0.944 ± 0.002	0.946 ± 0.007	0.912 ± 0.013	$\mathbf{0.968} \pm 0.013$
Average	0.803	0.853	0.866	0.859	$\mathbf{0.879}$

4.1 Gene Regulatory Element Classification

Nucleotide Transformer Downstream Nucleotide Transformer Downstream [9] includes 18 tasks both binary and multi-class classification tasks for epigenetic marker [28], promoter [26], enhancer and splice site prediction. Closely following the evaluation metrics from Caduceus [31], Matthews Correlation Coefficient (MCC) is used for histone marker and enhancer, while F1-score is used for splice and promoter classification. To compare Omni-DNA against Nucleotide Transformer V2(NTv2) [9], DNABERT2 [48], HyenaDNA [23], Caduceus [31]. We perform 10-fold cross-validation to select the hyperparameter for each model as detailed in Appendix D, and the detailed task-specific results for NT-Downstream tasks in Table 2. Omni-DNA (1B) and Omni-DNA (116M) achieve the best average performance of 0.767 and the second of 0.755, and they achieve superior performance on 13 out of 18 benchmark tasks, surpassing competing methods. For the remaining five tasks, Omni-DNA (1B) ranks second and third in Promoter:ALL and Promoter:NonTATA. In the three splice site classification tasks, Omni-DNA models perform lower than NT models but still surpass other models.

Genomic Benchmark Genomic Benchmark (GB) [13] contains eight DNA regulatory element classification tasks, similar to NT downstream tasks, with additional tasks on species classification and gene regulatory element classification on mouse. We compare model performance of Omni-DNA (116M) with DNABERT2, CADUCEUS-PH, and HynnaDNA, and show the results in Table 3. Omni-DNA (116M) is ranked first in five out of eight tasks and second in the remaining three, while achieving the highest average score. Compared to DNABERT-2, which has a similar model size, Omni-DNA (116M) outperformed DNABERT-2 in seven out of eight tasks.

Multi-tasking. We extend pretrained Omni-DNA (116M) and NTv2 (500M) models to a multi-task setting and compare their performance with their single tasks counter-parts. In particular, 10 related tasks from NT Downstream [9]—H3, H3K14AC, H3K36ME3, H3K4ME1, H3K4ME2, H3K4ME3, H3K79ME3, H3K9AC, H4, and H4AC—are considered. These tasks are interconnected due to the biological correlation between acetylation and methylation effects [28]. The objective here is to train a model capable of addressing all 10 tasks with a single fine-tuning.

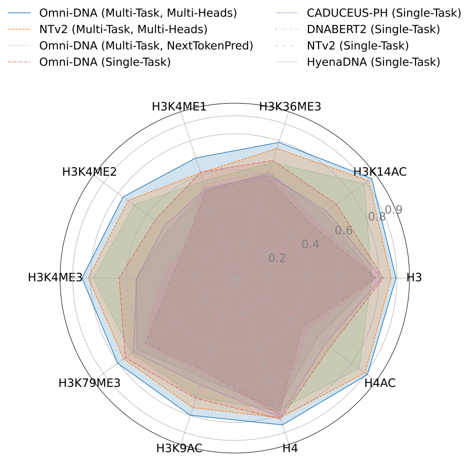


Figure 4: **Multi-task fine-tuning boosts accuracy.** Classification MCC on ten nucleosome-target (NT) tasks. Multi-head outperform NTP multi-tasking, but both surpass single-task baselines.

Two approaches are compared: **Multi-head fine-tuning** attaches task-specific linear heads to a shared Transformer backbone. **Next-Token prediction** augments the tokenizer with three label tokens $\{0, 1, 2\}$ and ten task identifiers $\{\texttt{task}_1 \ldots \texttt{task}_{10}\}$; the model autoregressively emits the label token, casting classification as language modelling.

Figure 4 shows *Omni-DNA (multitask,multi-head)* and *NTv2 (multitask,multi-head)* strictly outperform their single-task counterparts on all tasks, while *Omni-DNA (multi-task,NTP)* also achieves higher accuracy in half of the tasks. We hereby summarize the results as following finding.

Finding 3: Jointly fine-tuning on biologically related tasks consistently outperforms independent single-task fine-tuning.

4.2 Enhancer–Target Interaction Prediction

Task Enhancers are distal regulatory elements that can reside *kilobases* to *megabases* from the genes they activate, in contrast to promoters that typically lie immediately upstream of a transcription start site. We adopt the curated benchmark from Cheng et al. [7] originated from [11, 12, 34], which contains 2,602 experimentally validated enhancer–target pairs plus matched negative controls. Each sample is a 450 kb genomic window, and the data are split 8:1:1 into training, validation, and test sets. The task is binary classification: does the window contain an *effective* enhancer–target interaction? *Because accurate prediction requires modelling dependencies across hundreds of kilobases, this task is intrinsically more challenging than the shorter-context tasks assessed in the previous section.*

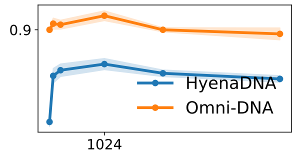
Models Because 450kb exceeds the context limits of NTv2 and DNABERT2, both models ran out of memory (OOM) and are therefore omitted. For Omni-DNA (116M) we raise the *native context length* to 2048 tokens to accommodate longer sequences. Additional long-range baselines include HyenaDNA (medium-450K) and CADUCEUS (PH/PS). We also report an expert "upper-bound" model, Activity-By-Contact (ABC), which leverages DNase-seq, ChIP-seq, and Hi-C matrices, and a lightweight CNN baseline from Cheng et al. [7]. The performance is measured by AUROC.

Results For HyenaDNA, CADUCEUS, and Omni-DNA, we append a single linear classification head and mean-pool the final hidden states over the sequence length. Models are fine-tuned for at most 20 epochs; learning rate and batch size are selected via a small grid search (see Table 8). As summarized in Table 4, Omni-DNA attains an AUROC of **0.895**, outperforming all neural baselines and closing much of the gap to the assay-rich ABC upper bound. In Appendix H, we provide additional ablation studies on the computational overhead and compression ratio of SeqPack.

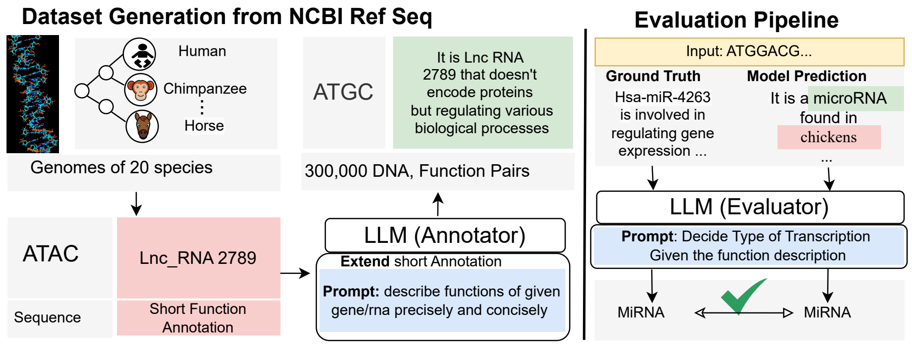
Table 4: **Enhancer–Target Sequence Classification**

Model	**AUROC**
ABC (expert)	0.926
CNN	0.797
HyenaDNA	0.828
CADUCEUS-PH	0.826
CADUCEUS-PS	0.821
Omni-DNA	**0.895**

Table 5: **Context-length sensitivity (*left*) and variant-type classification (*right*).** *Left*: F1 score of Omni-DNA on LC as a function of input length. *Right*: F1 score (↑) on LC and ICCs.



	HyenaDNA	DNABERT2	NTv2	Omni-DNA
LC	0.83±0.02	0.71± 0.03	0.75± 0.04	**0.93±0.01**
ICCs	0.78±0.08	0.49±0.02	0.75±0.00	**0.89±0.01**



Dataset Generation from NCBI Ref Seq
Human
Chimpanzee
Horse
Genomes of 20 species
ATGC
It is Lnc RNA 2789 that doesn't encode proteins but regulating various biological processes
300,000 DNA, Function Pairs
ATAC
Lnc_RNA 2789
Sequence
Short Function Annotation
LLM (Annotator)
Extend short Annotation
Prompt: describe functions of given gene/rna precisely and concisely
Evaluation Pipeline
Input: ATGGACG...
Ground Truth
Hsa-miR-4263 is involved in regulating gene expression ...
Model Prediction
It is a microRNA found in chickens ...
LLM (Evaluator)
Prompt: Decide Type of Transcription Given the function description
MiRNA
MiRNA

Figure 5: **SEQ2FUNC construction and evaluation pipeline.** Genomic windows extracted from 20 NCBI species are clustered by transcriptional evidence and passed to a LLM that produces concise, expert-style function descriptions. After agreement filtering and manual vetting the process yields 300k sequence–function pairs covering 10 functional classes.

4.3 Genetic Disease Variant–Type Classification

Moving beyond binary pathogenicity prediction [21], we address the more challenging task of directly inferring the disease category from a single–nucleotide variant (SNV), encoded as the triple $(\text{reference sequence}, \text{mutated sequence}, \text{label})$. We consider two clinically salient groups: **(i) Lung Cancer (LC)** with 11 disease types, and **(ii) Inherited Cardiac Conditions (ICCs)** with 1108 disease types.

Starting from the full ClinVar release (14810 unique diseases and $\sim$3 M variants), variants annotated with any of the **LC** or **ICCs** labels are marked positive; negatives are drawn at random from the remaining diseases. The resulting training sets contain 7,876 LC variants and 532,621 ICC variants. Each example is therefore a triple (r, m, y), where r is the reference sequence, m the mutated sequence, and $y \in \{0, 1\}$ denotes the absence or presence of the target disease class.

The left of Table 5 shows that extending the context length improves performance up to 1024bp, with negligible gains thereafter. We fine-tune Omni-DNA (116M), DNABERT2, NTv2 (500M), and HyenaDNA-medium (450K) based on the procedure in Appendix E. The F1 scores are summarized in the right of Table 5, where Omni-DNA delivers the strongest performance on both disease groups.

4.4 Sequence-to-Function Generation

Dataset. Publicly available functional labels are sparse, making automatic interpretation of raw DNA sequences challenging. To bridge this gap, we release **SEQ2FUNC**, a corpus of 300k high-quality *(sequence, function)* pairs (Figure 5). Starting from coarse annotations in the NCBI multi-species assemblies [6], we prompt a state-of-the-art LLM to generate refined, natural-language descriptions. Each entry is validated through agreement between two independent LLM passes, with low-confidence cases discarded. The final dataset spans 20 species and 10 broad functional categories, with detailed statistics provided in Appendix F. We follow an $80/10/10$ train/validation/test split.

Task and training protocol. Given a DNA sequence, the objective is to generate a textual description for its functional annotation. We first extend the Omni-DNA vocabulary from 4096 DNA tokens to 8173 tokens (adding 4077 wordpieces) and then fine-tune with next-token prediction on

Model Input: What is the function of this sequence?GGCTG...TTTTCTGA

Ground Truth: Olfactory receptor 5A2-like (OR5A2L) is a member of the olfactory receptor gene family, which is involved in the detection of volatile substances, contributing to smell sense.

Omni-DNA: The mRNA for olfactory receptor family 5 subfamily B member 109 (OR5B109) encodes a protein that is part of the olfactory receptor family. These receptors are involved in the detection of odorant molecules and and play a crucial role in the sense of smell.

Figure 6: **Demonstration of Omni-DNA's functional description generation.** Given a DNA sequence, Omni-DNA could generate a natural language description for functional annotations.

the **SEQ2FUNC** training set, consisting of concatenated `[DNA]` || `[description]` sequences. We use AdamW and NEFTune to train for up to 10 epochs with early stopping on validation perplexity. An LLM-based evaluator maps each generated description to one of 10 functional classes, allowing quantitative comparison with ground truth via weighted F1 score and Matthews correlation coefficient (MCC). Since existing genomic models cannot produce free-form text, we benchmark against GPT-4o in zero-shot mode (used prompts are shown in Table 9), with a random guesser as a lower bound.

Results. The qualitative examples in Figure 6 and Appendix F show that Omni-DNA@FT produces fluent and biologically plausible explanations and correctly identifies an unseen olfactory-receptor locus. Moreover, the quantitative results in Table 6 show that Omni-DNA outperforms GPT-4o in zero-shot generation by 7.1 % in weighted F1 and 38.2 % in MCC, substantially exceeding the random baseline.

Table 6: Performance on the SEQ2FUNC test set.

Model	**Weighted F1**	**MCC**
Omni-DNA @FT	**0.730**	**0.367**
GPT-4o (zero-shot)	0.659	−0.015
Random Guess	0.483	0.008

5 Conclusion

In this paper, we introduced Omni-DNA, a framework for exploring optimal strategies in pre-training and task adaptation for genomic modeling. Empirical analysis identified that the combination of non-parametric bias-free layer normalization with RoPE positional embeddings delivers the best trade-off between computational efficiency and downstream accuracy. We further demonstrate that multi-task fine-tuning on biologically related tasks substantially enhances performance. To extend the applicability of Omni-DNA, we develop SEQPACK, enabling efficient processing of ultra-long DNA sequences, and presentSEQ2FUNC, which empowers Omni-DNA to annotate DNA sequence functions using natural language. Across a range of downstream tasks, including regulatory element classification, enhancer–target interaction prediction, genetic disease variant classification, and DNA function annotation generation, Omni-DNA achieves state-of-the-art performance. We envision Omni-DNA as a step for scaling and generalizing genomic language models toward broader biological tasks.

Acknowledgments and Disclosure of Funding

Zehui Li is grateful to his family—Jinghai Li, Huiqing Wang, and Yiqiu Sun for their support.

References

[1] Sotiris Anagnostidis, Dario Pavllo, Luca Biggio, Lorenzo Noci, Aurelien Lucchi, and Thomas Hofmann. Dynamic context pruning for efficient and interpretable autoregressive transformers. *Advances in Neural Information Processing Systems*, 36:65202–65223, 2023.

[2] 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.

[3] Žiga Avsec, Natasha Latysheva, Jun Cheng, Guido Novati, Kyle R Taylor, Tom Ward, Clare Bycroft, Lauren Nicolaisen, Eirini Arvaniti, Joshua Pan, et al. Alphagenome: advancing regulatory variant effect prediction with a unified dna sequence model. *bioRxiv*, pages 2025–06, 2025.

[4] Jimmy Lei Ba. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[5] Gonzalo Benegas, Carlos Albors, Alan J Aw, Chengzhong Ye, and Yun S Song. Gpn-msa: an alignment-based dna language model for genome-wide variant effect prediction. *bioRxiv*, pages 2023–10, 2024.

[6] Richa Tanya Jeff Dennis A Colleen Evan Devon J Rodney St Agarwala Barrett Beck Benson Bollin Bolton Bourexi, Richa Agarwala, Tanya Barrett, Jeff Beck, Dennis A. Benson, et al. Database resources of the national center for biotechnology information. *Nucleic Acids Research*, 46:D8 – D13, 2017.

[7] Wenduo Cheng, Zhenqiao Song, Yang Zhang, Shike Wang, Danqing Wang, Muyu Yang, Lei Li, and Jian Ma. Dnalongbench: A benchmark suite for long-range dna prediction tasks. *bioRxiv*, pages 2025–01, 2025.

[8] Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.

[9] Hugo Dalla-Torre, Liam Gonzalez, Javier Mendoza-Revilla, Nicolas Lopez Carranza, Adam Henryk Grzywaczewski, Francesco Oteri, Christian Dallago, Evan Trop, Bernardo P de Almeida, Hassan Sirelkhatim, et al. Nucleotide transformer: building and evaluating robust foundation models for human genomics. *Nature Methods*, pages 1–11, 2024.

[10] Hugo Dalla-torre, Liam Gonzalez, Javier Mendoza Revilla, Nicolás López Carranza, Adam Henryk Grzywaczewski, Francesco Oteri, Christian Dallago, Evan Trop, Hassan Sirelkhatim, Guillaume Richard, Marcin J. Skwark, Karim Beguir, Marie Lopez, and Thomas Pierrot. The nucleotide transformer: Building and evaluating robust foundation models for human genomics. *bioRxiv*, 2024.

[11] Charles P Fulco, Joseph Nasser, Thouis R Jones, Glen Munson, Drew T Bergman, Vidya Subramanian, Sharon R Grossman, Rockwell Anyoha, Benjamin R Doughty, Tejal A Patwardhan, et al. Activity-by-contact model of enhancer–promoter regulation from thousands of crispr perturbations. *Nature genetics*, 51(12):1664–1669, 2019.

[12] Molly Gasperini, Andrew J Hill, José L McFaline-Figueroa, Beth Martin, Seungsoo Kim, Melissa D Zhang, Dana Jackson, Anh Leith, Jacob Schreiber, William S Noble, et al. A genome-wide framework for mapping gene regulation via cellular genetic screens. *Cell*, 176(1):377–390, 2019.

[13] Katarína Grešová, Vlastimil Martinek, David Čechák, Petr Šimeček, and Panagiotis Alexiou. Genomic benchmarks: a collection of datasets for genomic sequence classification. *BMC Genomic Data*, 24(1):25, 2023.

[14] Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, et al. Olmo: Accelerating the science of language models. *arXiv preprint arXiv:2402.00838*, 2024.

[15] Iris AM Huijben, Wouter Kool, Max B Paulus, and Ruud JG Van Sloun. A review of the gumbel-max trick and its extensions for discrete stochasticity in machine learning. *IEEE transactions on pattern analysis and machine intelligence*, 45(2):1353–1371, 2022.

[16] Neel Jain, Ping-yeh Chiang, Yuxin Wen, John Kirchenbauer, Hong-Min Chu, Gowthami Somepalli, Brian R Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Aniruddha Saha, et al. Neftune: Noisy embeddings improve instruction finetuning. *arXiv preprint arXiv:2310.05914*, 2023.

[17] Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V. Davuluri. Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *bioRxiv*, 2020.

[18] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[19] W Lathe, J Williams, M Mangan, and D Karolchik. Genomic data resources: challenges and promises. *Nature Education*, 1(3):2, 2008.

[20] Guihong Li, Duc Hoang, Kartikeya Bhardwaj, Ming Lin, Zhangyang Wang, and Radu Marculescu. Zero-shot neural architecture search: Challenges, solutions, and opportunities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[21] Zehui Li, Vallijah Subasri, Guy-Bart Stan, Yiren Zhao, and Bo Wang. Gv-rep: A large-scale dataset for genetic variant representation learning. *Advances in Neural Information Processing Systems*, 37:134701–134720, 2024.

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

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

[24] Eric D Nguyen, Michael Poli, Marjan Faizi, Armin W. Thomas, Callum Birch-Sykes, Michael Wornow, Aman Patel, Clayton M. Rabideau, Stefano Massaroli, Yoshua Bengio, Stefano Ermon, Stephen A. Baccus, and Christopher Ré. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. *ArXiv*, 2023.

[25] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, et al. Gpt-4 technical report. 2023.

[26] Mhaned Oubounyt, Zakaria Louadi, Hilal Tayara, and Kil To Chong. Deepromoter: robust promoter predictor using deep learning. *Frontiers in genetics*, 10:286, 2019.

[27] Alexander F Palazzo and T Ryan Gregory. The case for junk dna. *PLoS genetics*, 10(5):e1004351, 2014.

[28] Dmitry K Pokholok, Christopher T Harbison, Stuart Levine, Megan Cole, Nancy M Hannett, Tong Ihn Lee, George W Bell, Kimberly Walker, P Alex Rolfe, Elizabeth Herbolsheimer, et al. Genome-wide map of nucleosome acetylation and methylation in yeast. *Cell*, 122(4):517–527, 2005.

[29] Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021.

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

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

[32] Niklas Schmidinger, Lisa Schneckenreiter, Philipp Seidl, Johannes Schimunek, Pieter-Jan Hoedt, Johannes Brandstetter, Andreas Mayr, Sohvi Luukkonen, Sepp Hochreiter, and Günter Klambauer. Bio-xlstm: Generative modeling, representation and in-context learning of biological and chemical sequences. *arXiv preprint arXiv:2411.04165*, 2024.

[33] Conrad L Schoch, Stacy Ciufo, Mikhail Domrachev, Carol L Hotton, Sivakumar Kannan, Rogneda Khovanskaya, Detlef Leipe, Richard Mcveigh, Kathleen O'Neill, Barbara Robbertse, et al. Ncbi taxonomy: a comprehensive update on curation, resources and tools. *Database*, 2020:baaa062, 2020.

[34] Daniel Schraivogel, Andreas R Gschwind, Jennifer H Milbank, Daniel R Leonce, Petra Jakob, Lukas Mathur, Jan O Korbel, Christoph A Merten, Lars Velten, and Lars M Steinmetz. Targeted perturb-seq enables genome-scale genetic screens in single cells. *Nature methods*, 17(6):629–635, 2020.

[35] Rico Sennrich. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[36] Bin Shao. A long-context language model for deciphering and generating bacteriophage genomes. *bioRxiv*, 2024.

[37] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[38] Tianyuan Shi, Fanqi Wan, Canbin Huang, Xiaojun Quan, Chenliang Li, Ming Yan, and Ji Zhang. Profuser: Progressive fusion of large language models. *arXiv preprint arXiv:2408.04998*, 2024.

[39] 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.

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

[41] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[42] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

[43] Zhao Yang, Bing Su, Chuan Cao, and Ji-Rong Wen. Regulatory dna sequence design with reinforcement learning. *arXiv preprint arXiv:2503.07981*, 2025.

[44] Huixin Zhan, Jason H Moore, and Zijun Zhang. A disease-specific language model for variant pathogenicity in cardiac and regulatory genomics. *Nature Machine Intelligence*, pages 1–11, 2025.

[45] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

[46] Daoan Zhang, Weitong Zhang, Yu Zhao, Jianguo Zhang, Bing He, Chenchen Qin, and Jianhua Yao. Dnagpt: A generalized pre-trained tool for versatile dna sequence analysis tasks. *arXiv preprint arXiv:2307.05628*, 2023.

[47] Jian Zhou and Olga G Troyanskaya. Predicting effects of noncoding variants with deep learning–based sequence model. *Nature methods*, 12(10):931–934, 2015.

[48] 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 preprint arXiv:2306.15006*, 2023.

[49] Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana V. Davuluri, and Han Liu. Dnabert-2: Efficient foundation model and benchmark for multi-species genome. *ArXiv*, abs/2306.15006, 2023.

A Limitations

Despite these advances, limitations exists. Naive approach to explore the architecture choices through pretraining could be costly. Future research could streamline architecture optimization using automated techniques such as Zero-Shot NAS [20]. Additionally, integrating reinforcement learning for post-training methods could enhance efficiency and address the generalizability issue with supervised fune-tuning [8], although effective reward design remains challenging [37, 43].

B Proofs of Propositions

Proof. We bound the cost of constructing the compressed sequence E' under the geometric splitting scheduler.

(a) **Segment construction.** The sequence is partitioned into $M = \lceil \log_2 N \rceil$ segments whose lengths grow geometrically as $|S_i| \le 2^{i-1}$. Forming the segment boundaries therefore requires $O(M) = O(\log N)$ operations.

(b) **Local sampling.** Let b_i $(1 \le i \le M)$ be the sampling budget of segment S_i. Selecting b_i indices with a heap-based top-b_i (or an equivalent QuickSelect routine) costs $O(|S_i| \log b_i)$ time. Because the scheduler guarantees $b_i \le |S_i|$ and, in practice, $b_i \ll |S_i|$ for long histories, we may upper-bound the per-segment cost by $O(|S_i| \log|S_i|)$.

(c) **Total cost.** Summing over all segments,

$$\sum_{i=1}^{M} O\big(|S_i| \log|S_i|\big) \;=\; \sum_{i=1}^{M} O\big(2^{i-1}(i-1)\big).$$

Since $\sum_{i=1}^{M} 2^{\,i-1} = N$ and $(i-1) \le \log_2|S_i|$, the series is dominated by the linear term:

$$\sum_{i=1}^{M} 2^{i-1}(i-1) \;=\; O\big(N\big).$$

Hence the overall expected running time is $O(N)+O(\log N) = \Theta(N)$, in contrast to the $O(N \log K)$ required by a global top-K selection. □

Table 7: Comparison of pretraining setup between Omni-DNA, DNABERT2, and NT-transformer.

	Omni-DNA (six sizes)	**DNABERT2 (116M)**	**NT-transformer (5 sizes)**
Layer norm type	non-parametric/RMSNorm	parametric	parametric
Positional embeddings	RoPE	ALiBi	RoPE
Attention variant	full	full	full
Biases	none	in both LN and Attention	in both LN and Attention
Block type	sequential	sequential	sequential
Activation	SwiGLU	GEGLU	SwiGLU
Sequence length (In Tokens)	250	128	1000
Batch size (Across GPUs)	384	4096	512
Total Steps	800000	150000	$16,000$
Warmup steps	5000	-	16,000
Peak LR	3.0E-04	5.0E-04	1.0E-04
Minimum LR	3.0E-05	0	5.0E-05
Weight decay	0.1	1e-5	0.1
Beta1	0.9	0.9	0.9
Beta2	0.95	0.98	0.999
Epsilon	1.0E-05	1.0E-05	1.0E-08
LR schedule	linear	linear	linear
Gradient clipping	global 1.0	-	-
Training Precision	Mxied (bf16 for most operations)	-	-
Training Framework	Pytorch Built-in FSDP	Pytorch HuggingFace trainer	-
Training Duration (min)	8 GPUs* 1 days	8GPUs * 14 days	128 GPUs*1days
Training Duration (max)	8 GPUs* 7 days	8GPUs * 14 days	128 GPUs*28days
GPU Types	A100 Nvidia GPU 40GB	Nvidia RTX 2080Ti GPUs.	A100 Nvidia GPU

C Detailed Deduplication Process

C.1 Handling Non-{A, T, G, C} Characters

The raw genome data from the reference genome includes additional characters beyond {A, T, G, C}, which represent ambiguities or gaps. These characters are defined as follows:

- **N**: Represents any nucleotide (A, T, G, or C).
- **R**: Represents purines (A or G).
- **Y**: Represents pyrimidines (C or T).
- Other characters (e.g., **W**, **S**, **K**, **M**, etc.) represent specific nucleotide subsets or unknown bases.

These characters were removed during preprocessing to retain only the core nucleotide sequences ({A, T, G, C}), ensuring consistency and facilitating the deduplication process.

C.2 Chunk-Based Deduplication

The deduplication procedure was performed as follows:

1. **Chunking**: The genome was divided into non-overlapping chunks of 1024 base pairs.
2. **Exact Matching**: Identical sequences across the dataset were identified and removed.
3. **Efficiency**: This step utilized hashing techniques and optimized string comparison algorithms to handle the large dataset efficiently.

D Finetuning with Classification Head

Basic Setup We closely follow the setup from Caduceus [31] for evaluating Omni-DNA. The following models are compared: Omni-DNA, DNABERT-2 [48], NT-Transformer [9], HyenaDNA [23], and the Caduceus models [31].

For NT Downstream tasks, we use a maximum fine-tuning epoch of 20, while for the Genomic Benchmark (GB) tasks, we use a maximum of 10 epochs. Both NT Downstream and GB tasks include a training set and a test set. For hyperparameter search, 10% of the training set is reserved as a validation set.

Hardware & Framework All fine-tuning is conducted on a single NVIDIA A100 40GB GPU. We utilize the Hugging Face `Trainer` API for full-size fine-tuning of the pretrained checkpoints. Models are loaded using the `AutoModelForSequenceClassification` class, which automatically adds a linear layer on top for sequence classification.

Our experimental results align with those reported in Caduceus [31]. For consistency, we report statistics from [31] for two Caduceus models, DNABERT-2, NT-Transformer 500M, and HyenaDNA in NT Downstream tasks, as well as CNN, HyenaDNA, and CADUCEUS-PH in GB tasks. For NT2.5B, we use the results from its original paper [9].

We replicate experiments for the following models: NT50M, NT100M, NT250M, along with six Omni-DNA models on 18 NT Downstream tasks. Additionally, we fine-tune DNABERT-2 and Omni-DNA 116M on the eight Genomic Benchmark tasks.

Hyperparameters 10-fold cross validation is performed for each model to decide the best hyperparameters provided in table 8.

Additional Notes We observe a performance gap in HyenaDNA between our replication results, which align with those of [31], and the results reported in the original paper [23], both on NT downstream and GB tasks. Our replication follows our own setup, yielding results consistent with [31]. Upon reviewing the code within the container image provided by [23], we identified additional techniques such as Exponential Moving Average (EMA) and data augmentation that were employed, which may account for the discrepancy. In this work, we use HyenaDNA results obtained without these techniques as the baseline.

Table 8: Finetuning hyperparameters.

Hyperparameter	Value
Peak learning rate	$[3 \times 10^{-4}, 1 \times 10^{-5}, 5 \times 10^{-6}]$
Per device train batch size	[8,16,32]
Max gradient norm	1.0
Weight decay	0.1
Adam β_1	0.9
Adam β_2	0.999
Adam ϵ	1×10^{-8}
Optimizer	ADAMW

E Genetic Disease Variant Type Classification

Finetuning procedure. For every backbone (NTv2, Omni-DNA, DNABERT2, HyenaDNA) we wrap the pretrained model with a lightweight classification head that follows the implementation in algorithm 2. Each sample is encoded as a pair of sequences—reference (`ref`) and alternate (`alt`). Both sequences are passed independently through the frozen backbone to obtain the last–layer hidden states.

- **Encoder backbones** : we take the first-token representation (`[CLS]`).
- **Decoder backbones** : we take the representation of the last non-padding token in each sequence.

The two vectors are optionally transformed by a learnable pooler (a linear layer followed by $\tanh$). We then compute their difference, $\Delta h = h_{\text{alt}} - h_{\text{ref}}$, and feed it to a single fully connected layer that produces class logits. Cross-entropy loss is minimised for at most 100 epochs; a coarse grid search (summarised in table 8) is used to select learning rate, batch size, and weight decay.

F Functional Annotation Generation

Dataset Construction Process The Seq2Func dataset is constructed using genomic sequences sourced from the **NCBI Reference Sequence (RefSeq)** database, incorporating sequences from **20 different species**. The selected species and their genome assemblies include:

- GCF-000001405.40, human
- GCF-000001635.27, mouse
- GCF-036323735.1, rat
- GCF-028858775.2, Chimpanzee
- GCF-018350175.1, cat
- GCF-011100685.1, dog
- GCF-000696695.1, burgud
- GCF-000003025.6, pig
- GCF-016772045.2, sheep
- GCF-016699485.2, chicken
- GCF-025200985.1, fly
- GCF-000002035.6, Zebrafish
- GCF-029289425.2, Pygmy Chimpanzee
- GCF-029281585.2, Western gorilla
- GCF-028885655.2, Pongo abelii
- GCF-028885625.2, Bornean orangutan
- GCF-003339765.1, Rhesus monkey
- GCF-037993035.1, Macaca fascicularis
- GCF-003668045.3, Chinese hamster

Algorithm 2 Differential Classification Wrapper

```
1: procedure InitializeWrapper($\mathcal{M}_{\text{base}}$, $C$, decoder, usePooler)
2:     $d \leftarrow \mathcal{M}_{\text{base}}$.hidden_size                    ▷ fallback to $d_{\text{model}}$ if absent
3:     if usePooler then
4:         Pooler $\leftarrow$ Linear$(d, d) \circ \tanh$
5:     else
6:         Pooler $\leftarrow$ identity
7:     end if
8:     Head $\leftarrow$ Linear$(d, C)$
9:     return $\mathcal{W} = \{\mathcal{M}_{\text{base}}, \text{Pooler}, \text{Head}, \text{decoder}\}$
10: end procedure
11:
12: procedure Forward($\mathcal{W}$, $\mathbf{x}^{\text{ref}}$, $\mathbf{m}^{\text{ref}}$, $\mathbf{x}^{\text{alt}}$, $\mathbf{m}^{\text{alt}}$, $y$)
13:     $\mathbf{H}^{\text{ref}} \leftarrow \mathcal{M}_{\text{base}}(\mathbf{x}^{\text{ref}}, \mathbf{m}^{\text{ref}})$.lastHidden
14:     $\mathbf{H}^{\text{alt}} \leftarrow \mathcal{M}_{\text{base}}(\mathbf{x}^{\text{alt}}, \mathbf{m}^{\text{alt}})$.lastHidden
15:     if $\mathcal{W}$.decoder then
16:         $\mathbf{h}^{\text{ref}} \leftarrow$ LastToken$(\mathbf{H}^{\text{ref}}, \mathbf{m}^{\text{ref}})$
17:         $\mathbf{h}^{\text{alt}} \leftarrow$ LastToken$(\mathbf{H}^{\text{alt}}, \mathbf{m}^{\text{alt}})$
18:     else
19:         $\mathbf{h}^{\text{ref}} \leftarrow \mathbf{H}^{\text{ref}}[:, 0]$                    ▷ [CLS] token
20:         $\mathbf{h}^{\text{alt}} \leftarrow \mathbf{H}^{\text{alt}}[:, 0]$
21:     end if
22:     $\mathbf{h}^{\text{ref}} \leftarrow$ Pooler$(\mathbf{h}^{\text{ref}})$
23:     $\mathbf{h}^{\text{alt}} \leftarrow$ Pooler$(\mathbf{h}^{\text{alt}})$
24:     $\Delta\mathbf{h} \leftarrow \mathbf{h}^{\text{alt}} - \mathbf{h}^{\text{ref}}$
25:     $\hat{y} \leftarrow$ Head$(\Delta\mathbf{h})$
26:     if $y \neq \varnothing$ then
27:         $\mathcal{L} \leftarrow$ CrossEntropy$(\hat{y}, y)$
28:         return $\{\mathcal{L}, \hat{y}\}$
29:     else
30:         return $\hat{y}$
31:     end if
32: end procedure
```

- GCF-041296265.1, horse

The construction process follows these key steps:

Sequence Extraction and Functional Annotation

- The raw dataset includes a diverse set of RNA sequences with different functional annotations.
- The original dataset consists of the following RNA types and their counts:
  - mRNA: 220397
  - tRNA: 20674
  - snRNA: 18996
  - snoRNA: 15577
  - lncRNA: 14291
  - miRNA: 12238
  - primary-transcript: 8644
  - rRNA: 6225
  - transcript: 3860
  - ncRNA: 477
  - guide-RNA: 143
  - antisense-RNA: 28

  - RNase-P-RNA: 9
  - V-gene-segment: 7
  - scRNA: 7
  - Y-RNA: 6
  - telomerase-RNA: 5
  - vault-RNA: 4
  - SRP-RNA: 4
  - RNase-MRP-RNA: 3
  - C-gene-segment: 2
  - D-gene-segment: 1
- To refine the dataset, only **seven functional RNA types** are retained:
  - mRNA
  - tRNA
  - snRNA
  - snoRNA
  - lncRNA
  - miRNA
  - rRNA

Annotation Enhancement Using a Large Language Model (LLM)

- A **LLM Annotator** is employed to extend and refine functional annotations.
- The model is prompted with: *Describe the functions of the given gene/RNA precisely and concisely."*
- For example, an initial annotation like Lnc RNA 2789" is expanded into:

  "It is Lnc RNA 2789 that doesn't encode proteins but regulates various biological processes."

Final Dataset Composition

- The final dataset consists of **300,000 DNA-function pairs**, where each DNA sequence is paired with an enhanced functional description.
- These functionally annotated sequences serve as high-quality input for machine learning models.

Usage	**Prompt**
Used by **LLM (Annotator)** for extending the short annotation to detailed function annotation	**(System Prompt)** You are a helpful assistant that answers functions of given gene/rna precisely and concisely.
Used by **LLM (Evaluator)** for deciding the type of DNA given its function description	**(System Prompt)** You are a helpful assistant that determines the type of RNA based on the given function description. When deciding the functoin. Your answer should to only be one of the ['mRNA', 'tRNA', 'snRNA', 'snoRNA', 'lncRNA', 'miRNA', 'rRNA']
Given to **GPT4o@zeroshot** for performing zeroshot DNA function prediction	**(System Prompt)** You are a helpful assistant that determines the type of RNA based on the given DNA sequence. Your answer should only be one of the mRNA, tRNA, snRNA, snoRNA, lncRNA, miRNA, rRNA

Table 9: Prompt Used during dataset construction, evaluation and zeroshot prediction

Ground Truth	**GPT4o**	**OmniDNA@ft**
(Input) TGGCAGAGATATG...AAGAAAAAAGAAAATGAACTTGGTGCAGGCA (DNA of length 1000)		
mRNA:CD209, also known as DC-SIGN (Dendritic Cell Specific Intercellular adhesion molecule-3-grabbing Nonintegrin), is a type I transmembrane protein primarily expressed on dendritic cells and some other immune cells. It functions as a receptor that binds and internalizes pathogens, including viruses and bacteria, thereby facilitating their presentation to T cells and playing a critical role in the immune response. CD209 is involved in the recognition of mannose-rich ligands, mediating cellular adhesion, and influencing various aspects of immune activation and tolerance. Additionally, it can modulate inflammatory responses and has been implicated in HIV-1 and other pathogen infections.	The provided DNA sequence likely represents a genomic region involved in regulatory or coding functions. Based on its composition, it may contain promoter elements, coding regions, untranslated regions (UTRs), or repetitive sequences.	C-C motif chemokine ligand 22 (CCL22) is a chemokine involved in immune responses . It primarily attracts and activates T cells , particularly regulatory T cells (Tregs), and plays a role in modulating inflammation and immune responses. CCL22 is produced by various cell types , including macrophages and dendritic cells , and is implicated in various diseases , including autoimmune disorders and cancer.

Table 10: Comparison of Ground Truth, GPT4o, and OmniDNA@ft in RNA type determination (Example 1)

G Attention Sink Visualization

To investigate the emergence of attention sinks in our model, we visualize the self-attention scores across several transformer layers during inference. Specifically, we load the `Omni-DNA-116M` model and generate a random DNA sequence of length 100. This sequence is tokenized and passed through the model without applying a causal attention mask, as many downstream genomic tasks require bidirectional context.

We attach an attention probe to each transformer block, which records the softmax-normalized attention weights during the forward pass. In Figure 7, we show the attention matrices for four layers (`L0`, `L1`, `L-2`, `L-1`), each across four attention heads.

The visualization reveals distinct patterns: in `Layer 1`, the attention is diffuse without a clear sink; in `Layer 2`, we observe a strong attention sink at the `START` token; and in the last two layers, both the `START` and `END` tokens receive significant attention, suggesting the model learns to aggregate global information around these special positions in deeper layers.

H Additional Ablation Study On SeqPack Algorithm

Additional Ablation on the SeqPack Parameter with Different Compression Ratios To ablate the parameter of **SeqPack** compression ratios ($c = \frac{L_{original}}{L_{pruned}}$) and assess SeqPack's effectiveness against established methods, we adapted *Dynamic Context Pruning (DCP)* [1] to work with **Omni-DNA 116M** on the *Enhancer–Target Prediction* task. The results demonstrate SeqPack's substantial advantages.

Ground Truth	**GPT4o**	**OmniDNA@ft**
(Input) GAGGTAGTAGATTGTATAG		
miR -3596d is a microRNA found in rats (Rattus norvegicus). It is involved in the regulation of gene expression at the post-transcriptional level, primarily by binding to complementary sequences in messenger RNAs (mRNAs), leading to their degradation or inhibition of translation . While specific target genes may vary, miRNAs like rno-miR-3596d generally play roles in various biological processes such as development, cell differentiation, proliferation, and apoptosis. Specific roles or targets for rno-miR-3596d would require experimental validation or further studies to elucidate its functions in rat biology.	The DNA sequence GAGGTAGTAGATTGTATAG could serve various functions depending on its genomic context. Generally, it could be: 1. A regulatory element – If found in promoter or enhancer regions, it may influence gene expression. 2. A binding site – It could be a recognition site for transcription factors, such as ETS-family proteins (which recognize GGAA/T motifs). 3. A coding or non-coding region – If within a gene, it might encode a peptide or function as a non-coding RNA. 4. A structural or repeat element – It could contribute to chromatin organization or genome stability.For precise functional annotation, its location within a genome and experimental validation would be needed.	gga - miR - 65 77 - 5p is a microRNA (miRNA) found in chickens (Gallus gallus). MicroRNAs are small , non - coding RNA molecules that play a crucial role in regulating gene expression by binding to complementary sequences on target messenger RNAs (mRNAs), leading to their degradation or inhibition of translation . Specifically , gga - miR - 65 77 - 5p may be involved in various biological processes , including development , differentiation , and response to stress , although specific target genes and functions may require further study

Table 11: Comparison of Ground Truth, GPT4o, and OmniDNA@ft in RNA type determination (Example 2)

At extreme compression ratios (220×, 150×), DCP performance degrades severely, barely outperforming random baselines, while SeqPack maintains strong performance (AUROC > 0.89). Even at moderate compression (50×), SeqPack outperforms DCP (0.907 vs. 0.880 AUROC).

The Computational Overhead of SeqPack Compressing inputs (25,600–819,200 tokens) to a fixed 4,096 with SeqPack, we report (i) SeqPack operator time and (ii) Omni-DNA-1B forward time in seconds on the compressed sequence. As input length grows 32×, SeqPack time rises only $3.56\times$ (inference: 0.00898 to 0.03197 s) and $5.95\times$ (training: 0.015 to 0.091 s). Forward time remains effectively flat. SeqPack's share of end-to-end time remains low and increases gently with the compression ratio (6.25× to 200×): 1.2% to 4.0% (inference) and 4.3% to 21.6% (training).

Ground Truth	**GPT4o**	**OmniDNA@ft**
(Input) TGCTCACTTCAGCAGCACATGTGCTAAAAT TGGAATGACACAAAGAAGATTAGCATGGCCGC TGAGCAAGGATGACACGCACATTTGTGAATTGTTCCATAAAAAA		
nRNA:U6 spliceosomal RNA is a small nuclear RNA (snRNA) that plays an essential role in the splicing of pre-mRNA in eukaryotic cells. It is a core component of the spliceosome, the complex responsible for the removal of introns from pre-mRNA transcripts. U6 engages in base-pairing interactions with other spliceosomal RNAs and pre-mRNA, facilitating the assembly and function of the spliceosome. Its primary roles include catalyzing the cleavage of introns and the ligation of exons during the splicing process. U6 is also involved in the activation of the catalytic core of the spliceosome, making it crucial for proper gene expression.	The given DNA sequence likely encodes a functional genetic element, such as a protein-coding region or a regulatory sequence. To determine its precise function, I will need to analyze it for potential coding regions, motifs, or regulatory elements.	U6 spliceosomal RNA is a small nuclear RNA (snRNA) that plays a crucial role in the splicing of pre - mRNA in eukaryotic cells . It is a component of the spliceosome , the complex responsible for removing introns from pre - mRNA and joining exons together . U6 RNA is involved in the catalytic activity of the spliceosome and is essential for the recognition of splice sites and the formation of the active site for splicing . It also interacts with other snRNAs.

Table 12: Comparison of Ground Truth, GPT4o, and OmniDNA@ft in RNA type determination (Example 3)

Table 13: Performance comparison between SeqPack and DCP under various compression ratios on the Enhancer–Target Prediction task.

Model	AUROC ($\pm$ SE)	Compression Ratio
ABC (expert)	0.926	—
CNN	0.803 ± 0.022	—
SeqPack	0.894 ± 0.015	220
SeqPack	0.899 ± 0.009	150
SeqPack	0.907 ± 0.011	50
DCP	0.502 ± 0.089	220
DCP	0.632 ± 0.010	150
DCP	0.880 ± 0.012	50

Table 14: Inference-time and Training-time analysis of SeqPack compression across varying input lengths. For inference, SeqPack operator time grows sublinearly with sequence length, while forward time remains stable. At training time, SeqPack accounts for an increasing but still moderate fraction of total time as sequence length increases.

Length	SeqPack (s)	Forward (s)	SeqPack %
25,600	0.00898 ($\pm$0.000)	0.77257 ($\pm$0.003)	1.1
51,200	0.01032 ($\pm$0.001)	0.76120 ($\pm$0.001)	1.3
102,400	0.01225 ($\pm$0.001)	0.77986 ($\pm$0.003)	1.5
204,800	0.01508 ($\pm$0.001)	0.76666 ($\pm$0.001)	1.9
409,600	0.02050 ($\pm$0.001)	0.76665 ($\pm$0.001)	2.6
819,200	0.03197 ($\pm$0.001)	0.76514 ($\pm$0.003)	4.0

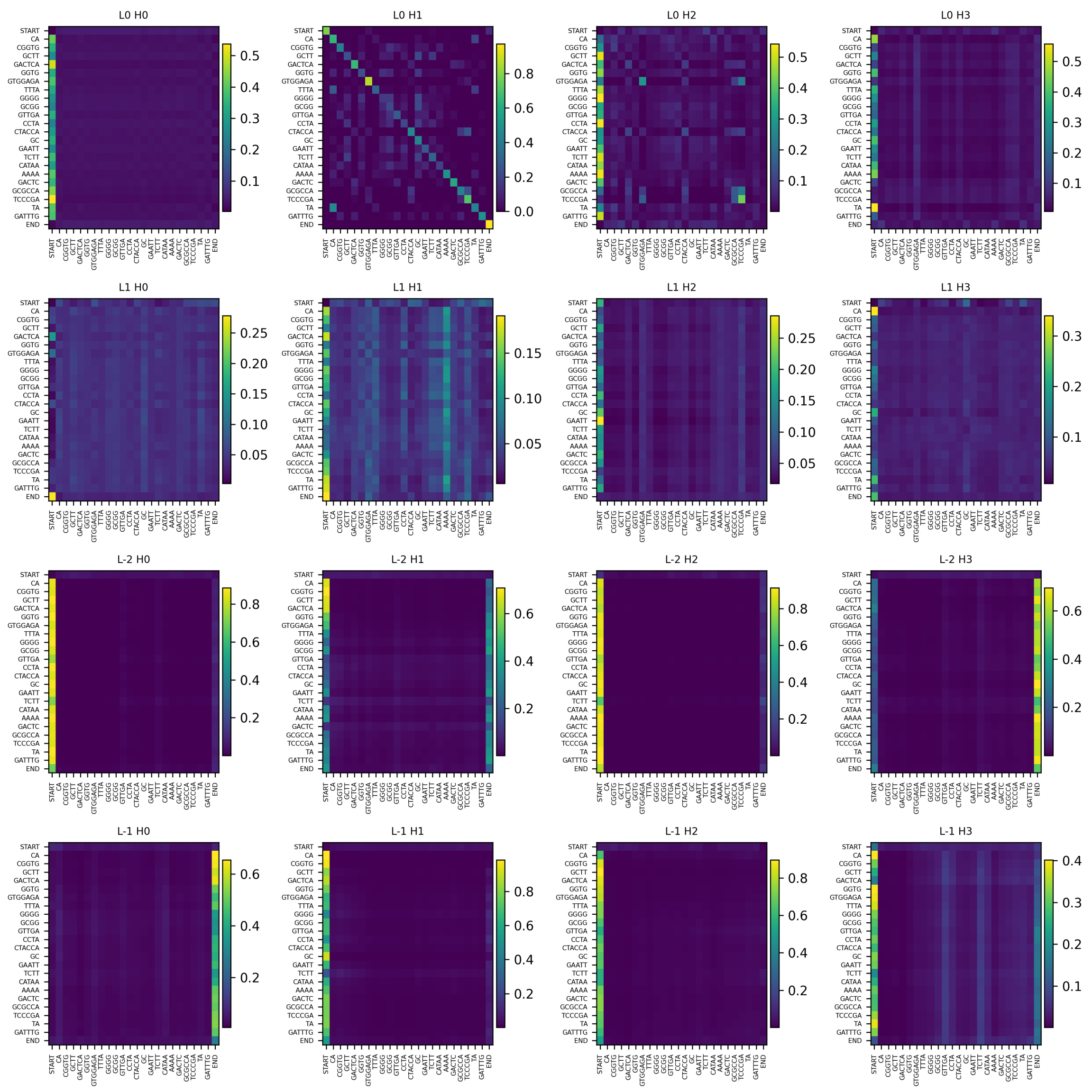


Figure 7: **Attention patterns across layers and heads in Omni-DNA.** Each row corresponds to a different transformer layer (L0, L1, L-2, L-1), and each column to a different attention head (H0–H3). For layer 1, attention is relatively diffused with no clear sink. At layer 2, we observe a strong attention sink at position 0 (START). In the last two layers, both the START and END tokens exhibit sink-like behavior, suggesting global aggregation mechanisms emerging in deeper layers.

Length	SeqPack (s)	Forward (s)	SeqPack %
25,600	0.01523 (±0.000)	0.33756 (±0.002)	4.3
51,200	0.01793 (±0.000)	0.32607 (±0.002)	5.2
102,400	0.02339 (±0.001)	0.34088 (±0.001)	6.4
204,800	0.03389 (±0.001)	0.32727 (±0.002)	9.4
409,600	0.05290 (±0.001)	0.32809 (±0.002)	13.9
819,200	0.09067 (±0.001)	0.32850 (±0.002)	21.6

NeurIPS Paper Checklist

1. **Claims**

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: We clearly list the section to return for each claim made in the introduction (Section 1).

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please see appendix A for the limitations.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: For each theoretical results, we either provide the proof inline section 3.3 or in the appendix section 3.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We fully describe the pretraining settings in table 7 and the additional details about finetuning for each tasks are included in appendix D and algorithm 2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the full code we use for performing the downstream tasks in the supplementary. Some of the data and pretrained model could be too large to share but we provide instructions on how to obtain the data in the supplementary.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Please see the justification for question 4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We conduct each experiment for multiple times to obtain the statistical significance.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We declare the use of computational resource for each experiment. Mostly on A100 Nvidia GPU 40GB*8 for pretraining, and A100 Nvidia GPU 80GB*1 for finetuning.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We confirm we conform all the items in NeurIPS code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We briefly discuss the positive contributions of our research to the broader community. To the best of our knowledge, it does not pose any negative societal impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: NA

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We confirm we have properly cited the work, on which we perform the benchmarking and used in any other form.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We include a structure description file for the code in the README along with the code itself in the supplementary.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: NA

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: We do not use LLM as important, original, or non-standard components for core method development.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.