MergeDNA: Context-aware Genome Modeling with Dynamic Tokenization through Token Merging

Siyuan Li[1,2,3], Kai Yu[2], Anna Wang[2], Zicheng Liu[1,2], Chang Yu[2*], Jingbo Zhou[1,2], Qirong Yang[3], Yucheng Guo[3], Xiaoming Zhang[3], Stan Z. Li[2*]

[1]Zhejiang University, Hangzhou, China
[2]AI Lab, Research Center for Industries of the Future, Westlake University, China
[3]BioMap Research, Beijing, China

Abstract

Modeling genomic sequences faces two unsolved challenges: the information density varies widely across different regions, while there is no clearly defined minimum vocabulary unit. Relying on either four primitive bases or independently designed DNA tokenizers, existing approaches with naive masked language modeling pre-training often fail to adapt to the varying complexities of genomic sequences. Leveraging Token Merging techniques, this paper introduces a hierarchical architecture that jointly optimizes a dynamic genomic tokenizer and latent Transformers with context-aware pre-training tasks. As for network structures, the tokenization module automatically chunks adjacent bases into words by stacking multiple layers of the differentiable token merging blocks with local-window constraints, then a Latent Encoder captures the global context of these merged words by full-attention blocks. Symmetrically employing a Latent Decoder and a Local Decoder, MergeDNA learns with two pre-training tasks: Merged Token Reconstruction simultaneously trains the dynamic tokenization module and adaptively filters important tokens, while Adaptive Masked Token Modeling learns to predict these filtered tokens to capture informative contents. Extensive experiments show that MergeDNA achieves superior performance on three popular DNA benchmarks and several multi-omics tasks with fine-tuning or zero-shot evaluation, outperforming typical tokenization methods and large-scale DNA foundation models.

1 Introduction

Modeling genomic DNA sequences with foundation models (Avsec et al. 2021; Ji et al. 2021) is an emerging frontier that promises to advance bioinformatics and precision medicine. DNA is often likened to a natural language carrying the "code of life" (Cooper 1981), yet it poses unique modeling challenges far beyond ordinary text. Firstly, genomic information is distributed unevenly. Only around 2% of the human genome consists of coding sequences (CDS), densely packed with functional information, whereas the vast majority is non-coding sequence (nCDS) with regulatory or unknown functions, which contains repetitive or less informative content (Nguyen et al. 2024a). Secondly, unlike natural languages with semantic words (Kudo and Richardson 2018), DNA has no inherent word boundaries or predefined vocabulary units (Zhou et al. 2023). The meaningful

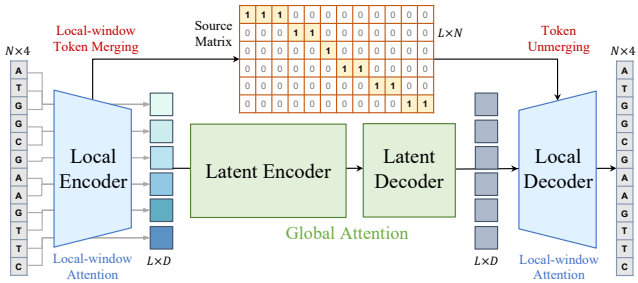

Figure 1: **Overview of MergeDNA architecture**.

"units" of DNA vary by context: a biologically relevant motif might be 3 bases (a codon in CDS) (Liu et al. 2025b) or 6–10 bases (a transcription factor binding site), or even longer sequences (Dalla-Torre et al. 2023). This makes fixed tokenization schemes inadequate (Qiao et al. 2024). Third, DNA sequences are extremely long (Nguyen et al. 2024b), often spanning tens of thousands to millions of bases, requiring models that can capture both short-range motifs and long-range dependencies efficiently. And naive pre-training objectives (Radford et al. 2018; Devlin et al. 2019) may fail to focus on the truly important parts of these vast sequences. These factors collectively make DNA fundamentally distinct from human language and call for a new class of sequence modeling architectures.

Recent studies have explored various facets of DNA foundation modeling. Long-sequence architectures such as linear-time state-space models (SSMs) (Nguyen et al. 2024b; Schiff et al. 2024), hierarchical Transformers (Shao and Yan 2024), and hybrid networks (Nguyen et al. 2024a; Ma et al. 2025) improve context length scalability with efficiency. Meanwhile, DNA tokenization strategies range from base-level encodings to k-mers (Ji et al. 2021; Wu et al. 2025) and learned vocabularies via BPE (Zhou et al. 2023) or vector quantization (van den Oord, Vinyals, and Kavukcuoglu 2017; Li et al. 2024a). Pre-training objectives also vary, including masked modeling (Ji et al. 2021), autoregressive loss (Zhang et al. 2023), and advanced masking (Roy et al. 2023). However, most works optimize these dimensions in isolation and lack a unified mechanism to address all three DNA modeling challenges. For example, the latest long-range models (Brixi et al. 2025) that still operate on single-base tokens may waste capacity on repetitive intergenic regions, while a learned tokenizer without a matching long-context encoder could miss global dependencies (Qiao et al. 2024). In this work, we argue that effective genome-

*Corresponding authors.

scale modeling requires two core capabilities: (i) a context-sensitive tokenizer that learns to segment DNA into variable-length units based on local structure and semantics, and (ii) adaptive pre-training objectives that prioritize information-dense regions for representation learning. We try to address these jointly by leveraging token merging techniques (Bolya et al. 2023; Lee and Hong 2024) for end-to-end learnable token granularity and contextual abstraction.

This work presents **MergeDNA**, a context-aware genome modeling framework that dynamically adapts tokenization and pre-training to genomic context, as shown in Figure 1. The core idea of MergeDNA is a hierarchical autoencoder-style Transformer that learns to compress and reconstruct DNA sequences with a differentiable tokenizer and a long-range context model. Specifically, we design a Local Encoder composed of stacked local-window attention blocks with differentiable token merging, enabling the model to chunk adjacent bases into variable-length tokens based on local similarity. These merged tokens are then processed by a global-context Latent Encoder using full attention. On the decoder side, a symmetric Latent Decoder and Local Decoder reconstruct the input sequence. Two pre-training objectives jointly supervise the model: (i) Merged Token Reconstruction trains the tokenizer and encoder to preserve key information while filtering redundancies; and (ii) Adaptive Masked Token Modeling selectively masks and predicts important tokens identified through token merging, encouraging context-aware learning of functionally relevant patterns. Together, these components form a unified and scalable genome modeling pipeline that adapts both token resolution and attention allocation based on input complexity.

Our contributions are summarized as follows:

- **Unified Architectural Design:** We propose a novel hierarchical framework that tightly integrates a learnable DNA tokenizer with long-range sequence modeling. Leveraging differentiable token merging within local attention blocks, the Local Encoder captures irregular genomic patterns and determines where to merge as words.
- **Adaptive Context Modeling:** We propose context-aware pre-training tasks that adapt to varying information density in genomic sequences. Using token merging to select informative positions, the proposed Merged Token Reconstruction and Adaptive Masked Token Modeling allow the model to capture both local motif-level information and global long-range dependencies.
- **Strong Empirical Results:** MergeDNA achieves state-of-the-art performance across three major DNA benchmarks and shows excellent generalization to several RNA and protein downstream tasks, outperforming prior DNA tokenization methods and foundation models in both short- and long-context settings.

2 Related Work

DNA Foundation Models. Adapting sequence modeling architectures to genomics has demonstrated impressive transfer capacities to genomic applications, where a family of DNA foundation models (Ji et al. 2021; Benegas, Batra, and Song 2023) has merged with four lines of research. **(a) Long sequence modeling** is the most crucial technique for long DNA sequences. State-space models (SSMs) like HyenaDNA (Nguyen et al. 2024b; Thoutam and Ellsworth 2024) and Caduceus (Schiff et al. 2024) deliver linear complexity, while hierarchical attention (Shao and Yan 2024) or hybrid SSM-attention designs (Brixi et al. 2025; Ma et al. 2025) capture both motifs and chromosome-level structure with moderate memory footprints. **(b) DNA tokenization** remains discussion with byte-level (Nguyen et al. 2024a), k-mers (Dalla-Torre et al. 2023), or learnable vocabularies (Ji et al. 2023; Qiao et al. 2024). **(c) Pre-training objectives** can be the BERT-style (Devlin et al. 2019; Zhou et al. 2023; Li et al. 2025a) or auto-regressive-like masked token modeling (Zhang et al. 2023; Zhu et al. 2024) for the encoder or decoder architectures, where some loss reweighing (Brixi et al. 2025) or tailored masking curricula (Roy et al. 2023; Roy, Sural, and Ganguly 2024) could be further beneficial. Only minor methods utilize contrastive learning (Zhou et al. 2025b) or cross-modality alignment tasks (Liu et al. 2025b) to integrate multi-omic cues. **(d) Domains of pre-training and applications** are usually bound. While most models are pre-trained on the human reference (Nguyen et al. 2024b) or multiple species corpora (Zhou et al. 2023), specialized datasets confer niche expertise, *e.g.*, prokaryotic (Nguyen et al. 2024a), plant genomic domains (Mendoza-Revilla et al. 2023; Zhai et al. 2025), and metagenomes (Zhou et al. 2025a). Extending beyond monomodal DNA, multi-omics models aim to simulate the central dogma (Cooper 1981) within a unified architecture (Yang et al. 2024) with a gene-to-expression pipeline (Avsec et al. 2021; Yang, Zhu, and Su 2025) or the genome-to-protein pipeline (Song, Segal, and Xing 2024; Liu et al. 2025b), leveraging shared structure across DNA, RNA, protein, and epigenome.

Byte-level Architectures. In NLP, early subword approaches like BPE (Sennrich, Haddow, and Birch 2015) and SentencePiece (Kudo and Richardson 2018) remain the default module in LLMs (Touvron et al. 2023), and dynamic schemes like Dynamic Pooling (Nawrot et al. 2022; Liu et al. 2024a) partially relax fixed vocabularies, yet they still require external pre-processing. Leveraging SSMs for linear-time attentions (Gu and Dao 2023; Yang et al. 2025), MegaByte (Yu et al. 2023) and MambaByte (Wang et al. 2024; Slagle 2024) demonstrated that multi-scale or SSM-based architectures without the subword tokenizer can model million-byte inputs end-to-end with great scale abilities (Ge et al. 2025) on text and other modalities (Wu et al. 2024). More recently, BLT (Pagnoni et al. 2024) introduces learned chunking with entropy-balanced patches, and HNet (Hwang, Wang, and Gu 2025) designs differentiable segmentation with jointly optimization. Similarly, DNA models with raw nucleotides as the input are also the byte-level architectures (Nguyen et al. 2024a), while the classical tokenization strategies like BPE (Zhou et al. 2023) and k-mers (Dalla-Torre et al. 2023) and learnable dictionaries (Li et al. 2024a) have also been explored.

3 Methodology

3.1 Preliminary

A DNA sequence can be seen as a string in the nucleotide alphabet $\mathcal{D} = \{\mathtt{A}, \mathtt{T}, \mathtt{C}, \mathtt{G}\}$. We denote a sequence of length N as $X = (x_1, x_2, \ldots, x_N) \in \mathcal{D}^N$, where each $x_i \in \mathcal{D}$. A DNA tokenizer $\mathcal{T} : \mathcal{D}^N \to \mathcal{V}^L$, segments X into a sequence of L tokens $Z_L = (z_1, \ldots, z_L)$ and maps to a vocabulary $\mathcal{V}$

with $N \geq L$. Given DNA sequences with a causal mask $M \in \{0,1\}^N$, a model f_θ with Attention blocks can be trained with an objective of masked token modeling (MTM):

$$\mathcal{L}_{MLM}(\theta) = -\frac{1}{L}\sum_{i=1}^{L} \log P(x_i \mid X * M;\, \theta), \qquad (1)$$

which encourages f_θ to infer each masked token x_i from its surrounding context to model the DNA context.

3.2 Architectural Overview

Adopting an autoencoder style, MergeDNA consists of four main components, which merge the fixed tokenizer and the sequence model into a hierarchical network in Figure 1.

Local Encoder for Tokenization. The Local Encoder $\mathcal{E}\phi$ serves as a learnable DNA tokenizer with local contexts, producing a tokenized sequence $Z_L \in \mathbb{R}^{L\times D}$ in the embedding dimension of D with a binary source matrix $\mathcal{S} \in \{0,1\}^{L\times N}$:

$$Z_L,\ \mathcal{S} \ =\ \mathcal{E}_\phi(X). \qquad (2)$$

Intuitively, Z_L is a context-dependent segmentation of X, and each row of $\mathcal{S}$ indicates which original positions in X were merged to form the corresponding token in Z_L. To adaptively chunk adjacent bases into informative tokens with local context, we implement the Local Encoder as a stack of local-window self-attention layers interleaved with differentiable token merging operations (described in Sec. 3.3), where the fixed local window size can also ensure linear-time computational complexity despite the long input. This learned tokenizer can be trained jointly with the rest of the model, allowing it to optimize token boundaries for the pre-training objective. Rather than using a fixed k-mer or requiring a byte-pair scheme, the Local Encoder can allocate shorter tokens (finer granularity) to dense information regions and longer tokens to repetitive regions, thereby addressing the varying information density of genomes.

Latent Context Modeling. Based on the tokenized sequence, the Latent Encoder $\mathcal{E}\psi$ is the main network for capturing long-range dependencies across the entire input, which can be a Transformer encoder with full attention (implemented with Flash Attention). During the standard inference, the Latent Encoder produces an output of the same length \$L\$:

$$Z'_L = \mathcal{E}_\psi(Z_L), \qquad (3)$$

where $Z'_L \in \mathbb{R}^{L\times D}$ are contextually enriched token embeddings. On top of the encoder, we include a lightweight Latent Decoder $\mathcal{E}_\omega$, which transforms Z'_L back toward the token space of $\hat{Z}_L$. The Latent Decoder has a symmetric architecture to $\mathcal{E}_\psi$, and outputs $\hat{Z}_L = \mathcal{E}_\omega(Z'_L)$, where $\hat{Z}_L$ can be seen as a reconstructed version of the Local Encoder's token sequence, containing the information needed to recover the original input. Together, $\mathcal{E}_\psi$ and $\mathcal{E}_\omega$ form an autoencoder on the token level. This design enables us to apply reconstruction-based training at the token level, providing learning signals to both the tokenizer (via the decoder) and the context encoder. We emphasize that the Latent Decoder is used only during pre-training (to assist the encoder and tokenizer); for many downstream tasks, the Latent Decoder may be omitted. This follows the common practice of using an encoder's learned representations for downstream prediction, while the decoder is specialized for generative reconstruction.

Local Decoder for Reconstruction. The final stage is the Local Decoder $\mathcal{E}_\zeta$, which maps the Latent Decoder's output $\hat{Z}_L$ back to the original base space and plays the role of "detokenizer". We first apply a token unmerging operation $\mathcal{U}(\cdot,\cdot)$ using the source matrix $\mathcal{S}$ to upsampling the L-length decoded tokens to length N, $\bar{Z}_N = \mathcal{U}(\hat{Z}_L, \mathcal{S})$. $\bar{Z}_N \in \mathbb{R}^{N\times D}$ denotes an unmerged sequence, where each position corresponds to an original base in X. In matrix form, if $\mathcal{S}_{ij} = 1$ indicates the position i covers original position j, then $\bar{Z}_N = \mathcal{S}^\top \hat{Z}_L$. After unmerging, $\mathcal{E}_\zeta$ applies a stack of local attention (as the reverse of the Local Encoder) to refine local details and output the reconstructed sequence $\hat{X} = (\hat{x}_1, \ldots, \hat{x}_N)$:

$$\hat{X} \ =\ \mathcal{E}_\zeta(\bar{Z}_N). \qquad (4)$$

The Local Decoder thus completes the autoencoder by learning to fill in base-level information that may have been abstracted away by the Local Encoder. Meanwhile, the Local Encoder can be encouraged to produce merge groupings that are easy to invert, as the source matrix preserves positional information that enables accurate reconstruction.

Training *vs.* Inference. During pre-training, we can optimize all modules $\theta = \{\phi, \psi, \omega, \zeta\}$ end-to-end by applying learning objectives of reconstruction and prediction tasks (detailed in Sec. 3.4) between the final output $\hat{X}$ and the original input. As for inference, MergeDNA can be truncated or reconfigured depending on the task types, which can function as a typical encoder-only model for representation learning, or as an encoder–decoder model for generative purposes. For generative tasks or any task requiring output at the nucleotide level (*e.g.*, sequence reconstruction or base pair prediction), one can use the entire autoencoder, or fine-tune the Local Decoder for the specific output prediction. For classification or regression tasks at the sample level (*e.g.*, the goal is to produce a label or embedding for the sequence), one can discard both decoders and use the Latent Encoder output directly with a fine-tuned head.

3.3 MergeDNA Tokenization

Local-window Token Merging. At the heart of the Local Encoder is a differentiable token merging mechanism that learns to segment the sequence. We build upon ToMe (Bolya et al. 2023), which progressively fuses similar tokens to reduce sequence length, but adapt it for local, fine-grained chunking. Each Local Encoder layer consists of a standard local self-attention followed by a token merging module. Given the l-th layer, supposing the input sequence length is N_{l-1}, the merging module will select r_l pairs of tokens within each window to compute the average, reducing the sequence by r_l tokens. We denote this operation as:

$$\mathcal{S}^{(l)}, Z^{(l)}_{N_l} = \text{LocalToMeAttn}^{(l)}\Big(Z^{(l-1)}_{N_{l-1}}, \mathcal{S}^{(l-1)}, r_l\Big), \quad (5)$$

where $\mathcal{S}^{(l-1)}$ is the source matrix carried from the previous layer (with $\mathcal{S}^{(0)} = I_{N_0}$ as an identity matrix at input),

and $\mathcal{S}^{(l)}$ is updated to reflect the new merges at the l-th layer. In implementation, we compute a similarity score for each pair of tokens in a local window (using a lightweight *grouping* embedding as in DTEM (Lee and Hong 2024)). The top-r_l most similar token pairs in each window are selected to merge. We then perform a *soft merging*: one token ("keeper") absorbs the other ("merger") by adding their representations, or a weighted average, and we mark this in $\mathcal{S}^{(l)}$, where the merger token's source positions are assigned to the keeper. Tokens not selected for merging pass through unchanged to the next layer. This continuous relaxation of token merging ensures the operation is differentiable, allowing gradients to tune both the token embeddings and the merging criteria.

Token Unmerging and Reconstruction. To optimize the end-to-end tokenization capacity, we introduce a *Merged Token Reconstruction (MTR)* objective $\mathcal{L}_{MTR}$ that forces the network to reconstruct the original sequence from compressed tokens, which can be computed as the cross-entropy between $\hat{X}$ and X:

$$\mathcal{L}_{MTR}(\theta) = -\frac{1}{N}\sum_{i=1}^{N} \log P(\hat{X}_i \mid X_i;\ \theta), \tag{6}$$

During training, we use a compression ratio sampling strategy, which randomly chooses the number of tokens to retain each iteration. For example, if the average goal is $L \approx \frac{N}{2}$, we might sample L from a Gaussian distribution centered at $\frac{N}{2}$ (with the variance to ensure $L \in [0.4N, 0.6N]$). This strategy exposes $\mathcal{E}\phi$ to a wider range of segmentations during training, improving its generalization ability and ensuring that the Local Encoder does not overfit to a particular compression rate.

3.4 Adaptive Context Modeling

As discussed in Sec. 3.2, the Latent Encoder $\mathcal{E}\psi$ processes L tokens uniformly during inference. However, genomic sequences often contain long stretches of low-information content (*e.g.*, repetitive DNA) where modeling every token is unnecessary. We aim to help the model find out and focus most informative tokens and design two steps to improve the naive MLM object.

Selection and Reconstruction. During pre-training, we modify the latent encoder $\mathcal{E}_\psi$ to forward an additional round and select a smaller number K of salient tokens. Technically, we replace the standard attention in $\mathcal{E}\psi$ with a ToMe-style Attention that merges tokens at the global scale (as opposed to local windows). Formally, we obtain $(Z'_K, \mathcal{S}') = \mathcal{E}\psi(Z_L, \mathcal{S})$, where $Z'_K \in \mathbb{R}^{K\times D}$ with $K < L$. Note that $\mathcal{S}'$ identifies the K most essential tokens among the Z_L: the merging algorithm preferentially fuses tokens that appear redundant or less salient, while preserving distinct tokens that carry unique information. We then feed Z'_K into the Latent Decoder to to produce $\hat{Z}_L$, but first we upsample it back to length L with the unmerge operation, $\bar{Z}_L = \mathcal{U}(Z'_K, \mathcal{S}')$. This distributes each of the K latent tokens back to its original token positions. Finally, $\hat{Z}_L$ is passed through the Local Decoder to produce $\hat{X}$, and we compute a reconstruction loss as $\mathcal{L}_{MTR}$. We refer to this loss as the latent MTR loss, $\mathcal{L}_{MTR}(\theta \setminus \{\phi\})$, since it trains the latent models to recover context from the selected tokens while the Local Encoder is held fixed (ϕ is not updated in this step). Intuitively, this task forces the latent transformer to not rely on having every token available – it must learn to encode the sequence in such a way that even if nearly $L - K$ tokens worth of information are dropped, the remaining K still capture the essential context to rebuild the sequence. This pushes $\mathcal{E}_\psi$ to generate a more compact, salient representation.

Adaptive Masked Token Modeling Beyond reconstruction, we also devise a masking strategy to predict the informative tokens. We leverage the latent merging outcome $\mathcal{S}'$ to decide which tokens to mask. The key idea is to assign a higher masking probability to tokens deemed important (those not merged heavily) and lower probability to tokens that were aggressively merged (low information). Given $\mathcal{S}' \in 0, 1^{K\times L}$ from the Latent Encoder, we compute an importance probability for each of the L local tokens. Let $g_i = \sum_{j=1}^{L} S' * i, j$ be the number of original tokens (out of L) that were grouped into the i-th latent token. We assign each latent group i a weight inversely proportional to its size, *e.g.*, $w_i = \frac{1}{g_i}$. For each token j that belongs to group i, we set $P_L(j) \propto \frac{w_i}{g_i}$, and choose the normalizing constant such that $\sum_{j=1}^{L} P_L(j) = 1$. This yields a probability vector $P_L \in \mathbb{R}^L$ over the local tokens, where tokens in large merged groups (large g_i) receive low probability and tokens in singleton or small groups receive higher probability. We then sample exactly K tokens without replacement according to P_L to mask. Letting $M_L \in 0, 1^L$ be the mask indicator, we map this mask back to the input space via the source matrix, $M_N = \mathcal{U}(M_L, \mathcal{S}) \in 0, 1^N$. In other words, if a merged token is selected to be masked, all of its constituent base positions in X will be masked out. Finally, we feed the masked sequence $X * M_N$ through the entire network without latent token merging, and get an output $\tilde{X}$. We define an Adaptive Masked Token Modeling (AMTM) loss as:

$$\mathcal{L}_{AMTM}(\theta) = -\frac{1}{K}\sum_{i:\, M_N(i)=1} \log P(\hat{X}_i \mid X * M_N; \theta). \tag{7}$$

This is essentially a masked language modeling loss focused on the K high-information tokens (and their base positions), ignoring the easy/redundant tokens. Overall, our full pre-training objectives involve three losses computed in three forward passes, which can be computed as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{MTR}(\theta) + \lambda\mathcal{L}_{MTR}(\theta \setminus \{\phi\}) + \mathcal{L}_{AMTM}(\theta), \tag{8}$$

where λ denotes a down-weighting factor, we set $\lambda = 0.25$ in practice, which ensures that the model learns to recover dropped information without overweighting the low-information content in its training signal.

4 Experiments

We first evaluate MergeDNA by comparison experiments on DNA benchmarks and multi-omics tasks with a wide range of sequence lengths using the supervised fine-tuning (SFT) or zero-shot evaluation protocols. Then, we empirically analyze the learned vocabularies and the proposed techniques. All experiments are conducted with PyTorch and NVIDIA A100-80G GPUs for three trials.

Method	HyenaDNA	Caduceus-16	DNABERT	DNABERT2	GENA-LM	NT-500M	VQDNA	MxDNA	ConvNova	GENERator	**MergeDNA**
Date	NeurIPS'23	ICML'24	Bioinfo'21	ICLR'24	NAR'23	NM'24	ICML'23	NeurIPS'24	ICLR'25	arXiv'25	**Ours**
# Params	6.6M	7.9M	86M	117M	113M	500M	93M	100M	1.7M	1.3B	380M
Architecture Type	byte+SSM	byte+SSM	6-mer+A	BPE+A	BPE+A	6-mer+A	VQ+A	DC+A	byte+CNN	6-mer+A	byte+A
Pre-training Task	AR	AR	BERT	BERT	BERT	BERT	BERT	BERT	BERT	AR	MTR+AMTM
Enhancers (3 tasks)	80.88	79.96	80.14	82.81	83.22	84.56	82.37	82.79	80.90	84.87	**85.11**
Species Classification (2 tasks)	93.61	94.65	94.74	95.49	95.11	96.64	95.79	96.46	95.50	**96.95**	96.84
Regulatory Elements (3 tasks)	88.89	85.97	83.42	86.33	87.89	89.05	87.62	90.57	87.30	90.30	**90.66**
Average (8 tasks)	87.07	85.89	85.02	87.30	87.94	89.26	87.69	89.12	86.95	90.71	**90.87**

Table 1: **Comparison on Genomic Benchmarks**. Top-1 accuracy (%) averaged over several similar tasks is reported for popular DNA foundation models with SFT evaluation. The best and the second best results are marked as the **bold** and underlined types.

Method	HyenaDNA	Caduceus-PS	DNABERT	GROVER	DNABERT2	NTv2-500M	MxDNA	ConvNova	GENERator	**MergeDNA**
Date	NeurIPS'23	ICML'24	Bioinfo'21	bioRxiv'23	ICLR'24	NM'24	NeurIPS'24	ICLR'25	arXiv'25	**Ours**
# Params (M)	6.6M	1.9M	86M	87M	117M	500M	100M	1.7M	1.2B	380M
H3	78.14	80.48	77.41	76.80	79.31	78.17	82.78	81.50	80.60	**82.95**
H3K4me1	44.52	52.83	43.83	46.10	48.34	51.64	56.15	**56.60**	55.30	56.24
H3K4me2	42.68	49.88	32.38	40.30	43.02	37.24	55.59	**57.45**	42.40	55.67
H3K4me3	50.41	56.72	31.49	45.80	45.43	50.30	63.68	**67.15**	51.20	64.10
H3K9ac	58.50	63.27	52.55	62.60	60.04	61.05	64.78	**68.10**	61.20	ul65.01
H3K14ac	56.71	60.84	46.51	54.80	54.49	57.22	68.27	**70.71**	60.50	68.51
H3K36me3	59.92	61.12	50.98	56.30	57.58	60.50	67.05	68.31	65.70	**68.19**
H3K79me3	66.25	67.17	60.48	58.10	64.38	65.78	**74.29**	72.08	67.00	74.23
H4	78.15	80.10	79.60	76.90	78.18	79.87	81.18	81.12	**81.50**	81.06
H4ac	54.15	59.26	41.53	53.00	51.80	55.22	**67.65**	66.10	59.20	67.26
Enhancer	53.13	55.20	79.13	51.60	52.50	54.51	**79.90**	57.60	58.00	79.84
Enhancer Types	48.16	47.17	54.73	43.30	44.32	43.36	60.50	49.75	47.70	**60.62**
Promoter All	95.57	96.65	97.05	92.60	96.23	96.82	97.16	96.82	96.20	**97.40**
Promoter Non-TATA	95.86	96.31	97.02	92.50	97.17	**97.45**	97.24	96.76	96.20	97.35
Promoter TATA	95.88	96.21	96.22	89.10	**96.99**	96.53	96.01	96.34	94.80	96.70
All	94.05	92.87	97.83	91.90	93.75	98.15	98.14	96.33	97.80	**98.35**
Accpetor	96.98	94.21	97.81	91.20	97.49	97.99	98.01	96.23	98.10	**98.67**
Donor	95.27	94.69	98.43	88.80	94.33	98.50	98.10	96.62	97.80	**98.93**
Average (18 tasks)	70.24	72.50	68.61	67.32	69.74	71.13	78.14	76.42	72.84	**78.39**

Table 2: **Comparison on NT Benchmark**. Matthews Correlation Coefficient (MCC) (%) or F1 score (%) is reported for subtasks with SFT evaluation. The best and the second best results are marked as the **bold** and underlined types.

4.1 Experimental Setup

Implementation of MergeDNA. Following the Transformer architecture in LLaMA (Touvron et al. 2023), MergeDNA adopts an embed dimension of $D = 1024$ and the local window size of 16. The Local Encoder and Decoder stack 4 and 2 Local ToMeAttention blocks, while the Latent Encoder and Latent Decoder use 20 and 4 Transformer blocks, totaling 380M parameters. Following DNABERT-2 (Zhou et al. 2023), we pre-train MergeDNA on the `Multi-Species Genomes` corpus using the AdamW optimizer (Loshchilov and Hutter 2019) for 100K iterations with a base learning rate of 1×10^{-4} and maximum sequence length $N = 4096$. The hierarchical compression yields a local encoder output length $L{=}\frac{N}{2}$ and a latent encoder length $K{=}\frac{L}{2}$, effectively modeling long-range context with reduced complexity. For downstream tasks, we adhere to the benchmark-specific SFT protocols. On sequence-level classification tasks, we discard both decoders and fine-tune a classification head on the latent encoder's output. For token-level (base-resolution) tasks, we retain the Local Decoder to recover sequence resolution and fine-tune a new token-level prediction head. View more details in the Appendix.

Comparison Baselines. We compare MergeDNA with state-of-the-art genomics models across four architecture paradigms: (1) sequence modeling architectures with SSMs have HyenaDNA (Nguyen et al. 2024b) and Caduceus (Schiff et al. 2024), (2) Standard Transformers have DNABERT (Ji et al. 2021), DNABERT-2 (Zhou et al. 2023), NTv1/v2 (Dalla-Torre et al. 2023), GROVER (Sanabria, Hirsch, and Poetsch 2023), and GenSLM (Zvyagin et al. 2022)), (3) Hybrid models have Evo (Nguyen et al. 2024a), Evo2 (Brixi et al. 2025), and HyriDNA (Ma et al. 2025)), and (4) CNN is ConvNova (Bo et al. 2025). As for the tokenizer, four popular types are compared in Table 1: (1) Byte-level like Evo, (2) k-mer like NTv2, (3) BPE like DNABERT2, (4) DNA dynamic tokenizer methods have VQDNA (Li et al. 2024a) and MxDNA (Qiao et al. 2024). All baseline foundation models were pre-trained with standard masked language modeling (BERT) or autoregressive (AR) objectives. As for downstream tasks, typical specialist models are also included.

4.2 Comparison Results on Genomic Benchmarks

Genomic Benchmarks. We first evaluate on eight representative tasks from the Genomic Benchmark suite (Grešová et al. 2023), covering enhancer identification, species classification, and regulatory element prediction. All models are fine-tuned on each task, and we report top-1 accuracy following the GenBench protocol. As shown in Table 1, MergeDNA achieves the highest overall accuracy (90.87%), outperforming all prior DNA foundation models. Notably, it yields state-of-the-art results on the enhancer tasks (85.11% *vs* 84.87% by the second best) and regulatory element tasks, while maintaining competitive performance on species classification (second only to a larger model).

Method	HyenaDNA	Caduceus-PS	DNABERT	NT-multi	DNABERT2	VQDNA	MxDNA	ConvNova	HybriDNA-7B	**MergeDNA**
Date	NeurIPS'23	ICML'24	Bioinfo'21	NM'24	ICLR'24	ICML'24	NeurIPS'24	ICLR'25	arXiv'25	**Ours**
# Params (M)	6.6M	1.9M	86M	2.5B	117M	93M	100M	1.7M	7B	380M
Epigenetic Marks Prediction (10)	58.94	58.39	49.08	58.06	55.98	57.95	67.29	68.91	63.05	**68.82**
Human TF Detection (3)	61.74	–	64.17	63.34	70.11	70.56	–	–	**72.89**	72.24
Mouse TF Detection (3)	64.37	–	56.43	67.02	67.99	69.80	–	–	**78.02**	73.21
Core Promoter Detection (3)	69.22	–	71.81	71.63	70.53	73.37	–	–	71.37	**73.41**
Promoter Detection (3)	80.14	–	81.69	**88.15**	84.21	86.58	–	–	85.53	87.73
Splice Site Reconstructed (1)	77.76	–	84.07	89.35	84.99	89.53	–	–	**90.09**	89.95
Virus Covid Classification (1)	25.88	–	55.50	73.04	71.02	74.32	–	–	74.02	**74.41**
Average (24 tasks)	62.58	58.39	60.53	67.23	66.43	68.51	67.29	68.91	76.42	**77.11**

Table 3: **Comparison on GUE Benchmark**. Matthews Correlation Coefficient (MCC) (%) or F1 score (%) averaged across sub-tasks is reported with SFT evaluation. The best and the second best results are marked as the **bold** and underlined types.

Method	SpliceAI	DNABERT2	NT-500M	Caduceus	Evo2-7B	**MergeDNA**
# Params	3.5M	117M	500M	7.9M	7B	380M
Donor	57.4	63.5	55.7	64.2	**64.5**	64.4
Acceptor	69.1	70.7	72.2	74.0	74.3	**74.5**
Mean	63.2	67.1	63.9	69.1	69.2	**69.8**

Table 4: **Comparison on Splicing Prediction** on the SpliceAI dataset, where the AUROC score is reported.

Method	GenSLM-2.5B	NT-2500M	ESM2-650M	Evo-7B	Evo2-7B	**MergeDNA**
# Params	2.5B	2.5B	650M	7B	7B	380M
Bacteria	24.7	9.4	51.2	45.30	**45.85**	42.72
Human	6.9	4.7	37.5	11.10	**36.9**	20.58

Table 5: **Comparison on Protein Fitness Prediction**. Zero-shot SRCC (%) is reported on DMS datasets.

Nucleotide Transformer Benchmarks. We further compare on the comprehensive Nucleotide Transformer (NT) benchmark (18 tasks) (Dalla-Torre et al. 2023) as summarized in Table 2. This benchmark includes a diverse mix of epigenomic signal classification (various histone marks and enhancer states, measured by MCC or F1) and core promoter/splice site detection tasks. MergeDNA again attains the best overall performance with an average score of 78.39, slightly surpassing the previous dynamic tokenizer method MxDNA (78.14) and substantially higher than other baselines. In particular, our model consistently ranks at or near the top on most individual tasks (*e.g.*, achieving the highest MCC on 10 out of 18 tasks).

GUE Benchmarks. We also evaluate on the Genome Understanding Evaluation (GUE) suite introduced by DNABERT2 (Zhou et al. 2023), which aggregates 24 short-range subtasks grouped into seven practical genomic applications: Epigenetic Mark Prediction (Yeast), Transcription Factor (TF) binding site detection (Human and Mouse), Promoter and Core Promoter Detection, Splice Site Prediction, and Virus Genomic Classification. We use Matthews Correlation Coefficient (MCC) or F1 score, as in prior work, and baseline results are taken from DNABERT-2 or the original papers for consistency. As summarized in Table 3, MergeDNA delivers the highest mean performance (77.11%), edging out the much larger HybriDNA-7B (76.42%) and outperforming all other foundation models. These results highlight that MergeDNA's dynamic tokenization and dual-context pre-training yield broad improvements across heterogeneous genomic prediction tasks.

4.3 Multi-omics Downstream Tasks

We further assess MergeDNA's generalization to long-range and cross-omics scenarios, including RNA splicing, expression prediction, and protein tasks.

RNA Splicing Site Prediction. Pre-mRNA splicing is a crucial step in gene expression, and we evaluate our model on the SpliceAI dataset (Jaganathan et al. 2019), which provides long pre-mRNA sequences labeled with donor and acceptor splice sites. We treat this as a binary sequence classification (site vs. non-site) and report the area under the ROC curve (AUROC). In Table 4, MergeDNA achieves a mean AUROC of 69.8, substantially outperforming the classic SpliceAI model (63.2) and all prior DNA foundation models. MergeDNA nearly matches the 7B-parameter Evo2 on donor site prediction and exceeds it on acceptor sites.

Long-range Expression Prediction. We next consider two challenging expression quantitative trait tasks from the Genomics Long-Range Benchmark (LRB)(Trop et al. 2024) that demand modeling of kilobase-scale contexts. (i) Causal eQTL Effect Prediction: given a genomic locus and a candidate variant, predict if the variant alters gene expression (evaluated by AUROC). (ii) Bulk RNA Expression Prediction: predict gene expression levels from surrounding DNA sequence (evaluated by R^2 correlation). As shown in Table 6, MergeDNA attains new state-of-the-art results on both tasks: an AUROC of 0.75 for eQTL (*vs* 0.74 by the best baseline) and an R^2 of 0.62 for bulk expression (*vs* 0.60).

Protein Fitness Prediction. Finally, we further evaluate MergeDNA in a strict zero-shot setting on protein fitness prediction tasks using Deep Mutational Scanning (DMS) data (Notin et al. 2022). Here, models must predict the functional fitness of protein variants (amino acid mutations) directly from the DNA coding sequence, without any fine-tuning on protein data. Table 5 reports Spearman's rank correlation coefficient (SRCC) between predicted and actual fitness on two representative DMS datasets (one bacterial protein and one human protein). A specialized protein language model (ESM2 (Lin et al. 2022), gray in the table) achieves the highest scores as expected. Among DNA-based models, MergeDNA shows strong cross-omics generalization: for the bacterial protein, it obtains 42.7% SRCC—on par with the 7B Evo model and only slightly behind the multi-omics Evo2 (45.9%). On the human protein, MergeDNA (20.6% SRCC) substantially outperforms earlier DNA models like GenSLM (6.9%) and the original Evo (11.1%), though it trails Evo2, which leverages direct protein training.

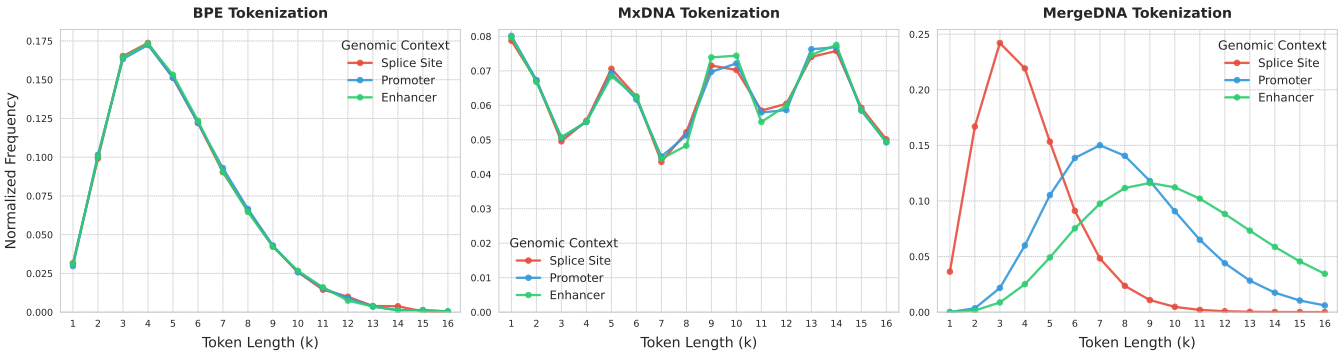

Figure 2: **Visualization of Token Length Distributions** for (a) BPE (Zhou et al. 2023), (b) MxDNA (Qiao et al. 2024), and (c) MergeDNA across different genomic contexts. Baseline tokenizers show a static, context-agnostic distribution, while MergeDNA adaptively changes its tokenization strategy based on the sequence type, demonstrating strong context-awareness.

Method	DNABERT2	DNABERT-S	NTv2-500M	HyenaDNA-160K	Caduceus-131K	HybridDNA-131K	Evo2-7B	**MergeDNA**
# Params	117M	117M	500M	12.9M	7.7M	300M	7B	380M
Causal eQTL (AUROC)	0.72	0.73	0.72	0.71	0.68	0.74	0.74	**0.75**
Bulk RNA (R^2)	0.51	0.52	0.60	0.46	0.52	0.52	0.60	**0.62**

Table 6: **Comparison on LRB Benchmark** with Causal eQTL Variant Effect Prediction and Bulk RNA Expression Prediction, where AUROC and R^2 scores are reported.

4.4 Empirical Analysis of Tokenization

To understand how MergeDNA learns to parse genomic sequences, we analyze the vocabularies learned by different tokenization strategies. We compare MergeDNA with two representative baseline tokenizers, **BPE** and **MxDNA**, by visualizing the normalized frequency of token lengths (from 1-mer to 16-mer) across different genomic contexts: promoters, enhancers, and splice sites. As shown in Figure 2, the tokenizers exhibit distinct behaviors. The BPE tokenizer in Figure 2(a) produces a fixed, long-tailed distribution that peaks around a token length of 6, regardless of the underlying sequence type. Similarly, the vector-quantized MxDNA tokenizer in Figure 2(b) yields a relatively uniform token length distribution that also shows minimal variation across different genomic contexts. This context-agnostic behavior limits their ability to adaptively capture functionally relevant motifs of varying lengths. In sharp contrast, MergeDNA's local encoder, as shown in Figure 2(c), demonstrates strong context-awareness. It learns to produce different token length distributions tailored to the biological properties of the input sequence, for splice sites, which are characterized by short, conserved motifs, the distribution peaks at a shorter token length (k=4). For longer and more complex regulatory regions like promoters and enhancers, the distributions shift towards longer tokens (peaking at k=7 and k=9, respectively). This adaptive, data-driven tokenization allows MergeDNA to dynamically capture genomic motifs at their relevant biological scales, providing a more expressive input representation for the downstream encoder and contributing to its superior performance.

4.5 Ablation Study

We conduct ablation experiments on the Genomic Benchmark tasks to quantify the contribution of each component in MergeDNA. Table 7 summarizes the average top-1 accuracy on the 8-task Genomic Benchmark under various configurations. First, we examine the impact of the hierarchical architecture. Replacing the first 4 Transformer layers with our Local Encoder (merging tokens in windows) improves performance by +0.39 with the same parameter budget, confirming the benefit of local token merging. Next, we ablate the pre-training objectives. Training with only the naive masked token modeling (MTM) objective results in suboptimal performance; adding the Merged Token Reconstruction ($\mathcal{L}MTR$) objectives targeting the tokenizer's output provides a large gain (+1.03), and further introducing the Adaptive MTM on filtered tokens pushes the improvement to +1.57 over baseline. We also find that scaling down the loss weight λ for the latent $\mathcal{L}MTR$ (*i.e.*, not directly updating tokenizer parameters) to 0.25 is crucial for better generalization, yielding the best result. Finally, we vary the depth of the Local Encoder: using only 2 local merging layers (with correspondingly more latent decoder layers) degrades performance, indicating that a deeper tokenizer (4 merging blocks) is important for capturing rich subword representations.

Tokenizer	Latent Enc.	Local Dec.	Pre-training	Acc.
Byte	24 layers	2 layers	$\mathcal{L}_{MTM}$	89.30
Local Enc. (4)	20 layers	2 layers	$\mathcal{L}^{\theta}_{MTR} + \mathcal{L}_{MTM}$	+0.39
Local Enc. (4)	20 layers	2 layers	$\mathcal{L}^{\theta}_{MTR} + \mathcal{L}^{\theta\setminus\{\phi\}}_{MTR} + \mathcal{L}_{AMTM}$	+1.03
Local Enc. (4)	20 layers	2 layers	$\mathcal{L}^{\theta}_{MTR} + \lambda\mathcal{L}^{\theta\setminus\{\phi\}}_{MTR} + \mathcal{L}_{AMTM}$	**+1.57**
Local Enc. (2)	20 layers	4 layers	$\mathcal{L}^{\theta}_{MTR} + \lambda\mathcal{L}^{\theta\setminus\{\phi\}}_{MTR} + \mathcal{L}_{AMTM}$	+1.21

Table 7: **Ablation Study** of Life-Code and pre-training tasks with DNA, protein, and central dogma (CD) tasks. Note that the blue background denotes the selected setups.

5 Conclusions

This paper introduces MergeDNA, a context-aware DNA foundation model that addresses fundamental challenges in genome modeling: heterogeneous information density, ambiguous sequence tokenization, and long-range dependencies. MergeDNA unifies a differentiable local tokenizer and a global latent Transformer through a hierarchical architecture and two complementary pre-training tasks, *i.e.*, Merged Token Reconstruction and Adaptive Masked Token Modeling. Extensive experiments on three standard DNA benchmarks and several multi-omics tasks demon-

strate that MergeDNA achieves state-of-the-art performance with strong generalization across species and modalities, offering a scalable and principled approach to genome-scale representation learning.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Project No. 624B2115, 623B2086, and U21A20427), the Science & Technology Innovation 2030 Major Program (Project No. 2021ZD0150100), the Center of Synthetic Biology and Integrated Bioengineering at Westlake University (Project No. WU2022A009), and the Westlake University Industries of the Future Research Program (Project No. WU2023C019). This work was done when Siyuan Li interned at BioMap Research. We thank the GPU support from BioMap Research and the AI station of Westlake University.

References

AlQuraishi, M. 2019. ProteinNet: a standardized data set for machine learning of protein structure. In *BMC Bioinformatics*, 95–104.

Avsec, Ž.; Agarwal, V.; Visentin, D.; Ledsam, J. R.; Grabska-Barwinska, A.; Taylor, K. R.; Assael, Y.; Jumper, J.; Kohli, P.; and Kelley, D. R. 2021. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature methods*, 18(10): 1196–1203.

Avsec, Ž.; Latysheva, N.; Cheng, J.; Novati, G.; Taylor, K. R.; Ward, T.; Bycroft, C.; Nicolaisen, L.; Arvaniti, E.; Pan, J.; et al. 2025. AlphaGenome: advancing regulatory variant effect prediction with a unified DNA sequence model. *bioRxiv*, 2025–06.

Bai, J.; Bai, S.; Chu, Y.; Cui, Z.; Dang, K.; Deng, X.; Fan, Y.; Ge, W.; Han, Y.; Huang, F.; Hui, B.; Ji, L.; Li, M.; Lin, J.; Lin, R.; Liu, D.; Liu, G.; Lu, C.; Lu, K.; Ma, J.; Men, R.; Ren, X.; Ren, X.; Tan, C.; Tan, S.; Tu, J.; Wang, P.; Wang, S.; Wang, W.; Wu, S.; Xu, B.; Xu, J.; Yang, A.; Yang, H.; Yang, J.; Yang, J.; Yang, S.; Yao, Y.; Yu, B.; Bowen, Y.; Yuan, H.; Yuan, Z.; Zhang, J.; Zhang, X.; Zhang, Y.; Zhang, Z.; Zhou, C.; Zhou, J.; Zhou, X.; and Zhu, T. 2023. Qwen Technical Report. *ArXiv*, abs/2309.16609.

Benegas, G.; Batra, S. S.; and Song, Y. S. 2023. DNA language models are powerful predictors of genome-wide variant effects. *Proceedings of the National Academy of Sciences*, 120(44): e2311219120.

Bo, Y.; Mao, W.; Shao, Y.; Bai, W.; Ye, P.; Ma, X.; Zhao, J.; Chen, H.; and Shen, C. 2025. Revisiting Convolution Architecture in the Realm of DNA Foundation Models. *arXiv preprint arXiv:2502.18538*.

Bolya, D.; Fu, C.-Y.; Dai, X.; Zhang, P.; Feichtenhofer, C.; and Hoffman, J. 2023. Token Merging: Your ViT But Faster. In *International Conference on Learning Representations (ICLR)*.

Brixi, G.; Durrant, M. G.; Ku, J.; Poli, M.; Brockman, G.; Chang, D.; Gonzalez, G. A.; King, S. H.; Li, D. B.; Merchant, A. T.; Naghipourfar, M.; Nguyen, E.; Ricci-Tam, C.; Romero, D. W.; Sun, G.; Taghibakshi, A.; Vorontsov, A.; Yang, B.; Deng, M.; Gorton, L.; Nguyen, N.; Wang, N. K.; Adams, E.; Baccus, S. A.; Dillmann, S.; Ermon, S.; Guo, D.; Ilango, R.; Janik, K.; Lu, A. X.; Mehta, R.; Mofrad, M. R.; Ng, M. Y.; Pannu, J.; Ré, C.; Schmok, J. C.; St. John, J.; Sullivan, J.; Zhu, K.; Zynda, G.; Balsam, D.; Collison, P.; Costa, A. B.; Hernandez-Boussard, T.; Ho, E.; Liu, M.-Y.; McGrath, T.; Powell, K.; Burke, D. P.; Goodarzi, H.; Hsu, P. D.; and Hie, B. L. 2025. Genome modeling and design across all domains of life with Evo 2. *Arc Institute Manuscripts*.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *International Conference on Machine Learning (ICML)*.

Cheng, W.; Song, Z.; Zhang, Y.; Wang, S.; Wang, D.; Yang, M.; Li, L.; and Ma, J. 2025. DNALongBench: A Benchmark Suite for Long-Range DNA Prediction Tasks. *bioRxiv*.

Cooper, S. 1981. The central dogma of cell biology. *Cell biology international reports*, 5(6): 539–549.

Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; and Wei, Y. 2017. Deformable Convolutional Networks. In *International Conference on Computer Vision (ICCV)*, 764–773.

Dalla-Torre, H.; Gonzalez, L.; Mendoza-Revilla, J.; Carranza, N. L.; Grzywaczewski, A. H.; Oteri, F.; Dallago, C.; Trop, E.; de Almeida, B. P.; Sirelkhatim, H.; et al. 2023. The nucleotide transformer: Building and evaluating robust foundation models for human genomics. *bioRxiv*, 2023–01.

Dao, T.; Fu, D. Y.; Ermon, S.; Rudra, A.; and R'e, C. 2022. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. *ArXiv*, abs/2205.14135.

de Almeida, B. P.; Dalla-Torre, H.; Richard, G.; Blum, C.; Hexemer, L.; Gélard, M.; Mendoza-Revilla, J.; Pandey, P.; Laurent, S.; Lopez, M.; et al. 2024. SegmentNT: annotating the genome at single-nucleotide resolution with DNA foundation models. *bioRxiv*, 2024–03.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 4171–4186.

Dreos, R.; Ambrosini, G.; Cavin Périer, R.; and Bucher, P. 2013. EPD and EPDnew, high-quality promoter resources in the next-generation sequencing era. *Nucleic acids research*, 41(D1): D157–D164.

Garau-Luis, J. J.; Bordes, P.; Gonzalez, L.; Roller, M.; de Almeida, B. P.; Hexemer, L.; Blum, C.; Laurent, S.; Grzegorzewski, J.; Lang, M.; Pierrot, T.; and Richard, G. 2024. Multi-modal Transfer Learning between Biological Foundation Models. In Globerson, A.; Mackey, L.; Belgrave, D.; Fan, A.; Paquet, U.; Tomczak, J.; and Zhang, C., eds., *Advances in Neural Information Processing Systems*, volume 37, 78431–78450. Curran Associates, Inc.

Ge, H.; Feng, J.; Huang, Q.; Fu, F.; Nie, X.; Zuo, L.; Lin, H.; Cui, B.; and Liu, X. 2025. ByteScale: Efficient Scaling of LLM Training with a 2048K Context Length on More Than 12,000 GPUs. *ArXiv*, abs/2502.21231.

Grešová, K.; Martinek, V.; Čechák, D.; Šimeček, P.; and Alexiou, P. 2023. Genomic benchmarks: a collection of datasets for genomic sequence classification. *BMC Genomic Data*, 24(1): 25.

Gu, A.; and Dao, T. 2023. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. *ArXiv*, abs/2312.00752.

Hayes, T.; Rao, R.; Akin, H.; Sofroniew, N. J.; Oktay, D.; Lin, Z.; Verkuil, R.; Tran, V. Q.; Deaton, J.; Wiggert, M.; et al. 2025. Simulating 500 million years of evolution with a language model. *Science*, 387(6736): 850–858.

He, Y.; Fang, P.; Shan, Y.; Pan, Y.; Wei, Y.; Chen, Y.; Chen, Y.; Liu, Y.; Zeng, Z.; Zhou, Z.; et al. 2024. LucaOne: Generalized Biological Foundation Model with Unified Nucleic Acid and Protein Language. *bioRxiv*, 2024–05.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Hwang, S.; Wang, B.; and Gu, A. 2025. Dynamic Chunking for End-to-End Hierarchical Sequence Modeling. *arXiv preprint arXiv:2507.07955*.

Jaegle, A.; Borgeaud, S.; Alayrac, J.-B.; Doersch, C.; Ionescu, C.; Ding, D.; Koppula, S.; Brock, A.; Shelhamer, E.; H'enaff, O. J.; Botvinick, M. M.; Zisserman, A.; Vinyals, O.; and Carreira, J. 2021a. Perceiver IO: A General Architecture for Structured Inputs & Outputs. *ArXiv*, abs/2107.14795.

Jaegle, A.; Gimeno, F.; Brock, A.; Zisserman, A.; Vinyals, O.; and Carreira, J. 2021b. Perceiver: General Perception with Iterative Attention. *ArXiv*, abs/2103.03206.

Jaganathan, K.; Kyriazopoulou Panagiotopoulou, S.; McRae, J. F.; Darbandi, S. F.; Knowles, D.; Li, Y. I.; Kosmicki, J. A.; Arbelaez, J.; Cui, W.; Schwartz, G. B.; Chow, E. D.; Kanterakis, E.; Gao, H.; Kia, A.; Batzoglou, S.; Sanders, S. J.; and Farh, K. K.-H. 2019. Predicting Splicing from Primary Sequence with Deep Learning. *Cell*, 176(3): 535–548.e24.

Ji, Y.; Zhou, Z.; Liu, H.; and Davuluri, R. V. 2021. DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome. *Bioinformatics*, 37(15): 2112–2120.

Ji, Z.; Zhang, H.; Huang, J.; et al. 2023. GENA-LM: Multimodal Pre-training for the Central Dogma. *bioRxiv*.

Jumper, J.; Evans, R.; Pritzel, A.; Green, T.; Figurnov, M.; Ronneberger, O.; Tunyasuvunakool, K.; Bates, R.; Žídek, A.; Potapenko, A.; et al. 2021. Highly accurate protein structure prediction with AlphaFold. *nature*, 596(7873): 583–589.

Khare, S.; Gurry, C.; Freitas, L.; Schultz, M. B.; Bach, G.; Diallo, A.; Akite, N.; Ho, J.; Lee, R. T.; Yeo, W.; et al. 2021. GISAID's role in pandemic response. *China CDC weekly*, 3(49): 1049.

Krishna, R.; Wang, J.; Ahern, W.; Sturmfels, P.; Venkatesh, P.; Kalvet, I.; Lee, G. R.; Morey-Burrows, F. S.; Anishchenko, I.; Humphreys, I. R.; et al. 2024. Generalized biomolecular modeling and design with RoseTTAFold All-Atom. *Science*, 384(6693): eadl2528.

Kudo, T.; and Richardson, J. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Conference on Empirical Methods in Natural Language Processing*.

Lee, D. H.; and Hong, S. 2024. Learning to Merge Tokens via Decoupled Embedding for Efficient Vision Transformers. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Li, Q.; Wu, W.; Zhu, Y.; Feng, F.; Ye, J.; and Wang, Z. 2025a. GENERanno: A Genomic Foundation Model for Metagenomic Annotation. *bioRxiv*.

Li, S.; Wang, Z.; Liu, Z.; Wu, D.; Tan, C.; Zheng, J.; Huang, Y.; and Li, S. Z. 2024a. VQDNA: Unleashing the Power of Vector Quantization for Multi-Species Genomic Sequence Modeling. *ArXiv*, abs/2405.10812.

Li, Z.; Cranganore, S. S.; Youngblut, N. D.; and Kilbertus, N. 2024b. Whole Genome Transformer for Gene Interaction Effects in Microbiome Habitat Specificity. *ArXiv*, abs/2405.05998.

Li, Z.; Ni, Y.; Beardall, W. A. V.; Xia, G.; Das, A.; Stan, G.-B. V.; and Zhao, Y. 2024c. DiscDiff: Latent Diffusion Model for DNA Sequence Generation. *ArXiv*, abs/2402.06079.

Li, Z.; Subasri, V.; Shen, Y.; Li, D.; Zhao, Y.; Stan, G.-B.; and Shan, C. 2025b. Omni-DNA: A Unified Genomic Foundation Model for Cross-Modal and Multi-Task Learning. *arXiv preprint arXiv:2502.03499*.

Lin, Z.; Akin, H.; Rao, R.; Hie, B.; Zhu, Z.; Lu, W.; Smetanin, N.; dos Santos Costa, A.; Fazel-Zarandi, M.; Sercu, T.; Candido, S.; et al. 2022. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *bioRxiv*.

Liu, G.; Chen, L.; Wu, Y.; Han, Y.; Bao, Y.; and Zhang, T. 2025a. PDLLMs: A group of tailored DNA large language models for analyzing plant genomes. *Molecular Plant*, 18(2): 175–178.

Liu, H.; and Wang, H. 2024. GenoTEX: A Benchmark for Evaluating LLM-Based Exploration of Gene Expression Data in Alignment with Bioinformaticians. *ArXiv*, abs/2406.15341.

Liu, Y.; Ji, T.; Sun, C.; Wu, Y.; and Wang, X. 2024a. Generation with Dynamic Vocabulary. In *Conference on Empirical Methods in Natural Language Processing*.

Liu, Z.; Li, J.; Li, S.; Zang, Z.; Tan, C.; Huang, Y.; Bai, Y.; and Li, S. Z. 2024b. GenBench: A Benchmarking Suite for Systematic Evaluation of Genomic Foundation Models. arXiv:2406.01627.

Liu, Z.; Li, S.; Chen, Z.; Xin, L.; Wu, F.; Yu, C.; Yang, Q.; Guo, Y.; Yang, Y.; and Li, S. Z. 2025b. Life-Code: Central Dogma Modeling with Multi-Omics Sequence Unification. *ArXiv*, abs/2502.07299.

Liu, Z.; Wang, L.; Li, S.; Wang, Z.; Lin, H.; and Li, S. Z. 2024c. LongVQ: Long Sequence Modeling with Vector Quantization on Structured Memory. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations (ICLR)*.

Ma, M.; Liu, G.; Cao, C.; Deng, P.; Dao, T.; Gu, A.; Jin, P.; Yang, Z.; Xia, Y.; Luo, R.; Hu, P.; Wang, Z.; Chen, Y.; Liu, H.; and Qin, T. 2025. HybriDNA: A Hybrid Transformer-Mamba2 Long-Range DNA Language Model. *ArXiv*, abs/2502.10807.

Mendoza-Revilla, J.; Trop, E.; Gonzalez, L.; Roller, M.; Dalla-Torre, H.; de Almeida, B. P.; Richard, G.; Caton, J.; Lopez Carranza, N.; Skwark, M.; et al. 2023. A Foundational Large Language Model for Edible Plant Genomes. *bioRxiv*, 2023–10.

MiniMax; Li, A.; Gong, B.; Yang, B.; Shan, B.; Liu, C.; Zhu, C.; Zhang, C.; Guo, C.; Chen, D.; Li, D.; Jiao, E.; Li, G.; Zhang, G.; Sun, H.; Dong, H.; Zhu, J.; Zhuang, J.; Song, J.; Zhu, J.; Han, J.-M.; Li, J.; Xie, J.; Xu, J.; Yan, J.; Zhang, K.; Xiao, K.; Kang, K.; Han, L.; Wang, L.; Yu, L.-C.; Feng, L.; Zheng, L.; Chai, L.; Xing, L.; Ju, M.; Chi, M.; Zhang, M.; Huang, P.; Niu, P.-X.; Li, P.; Zhao, P.; Yang, Q.; Xu, Q.; Wang, Q.; Wang, Q.; Li, Q.; Leng, R.; Shi, S.; Yu, S.; Li, S.-S.; Zhu, S. H.; Huang, T.; Liang, T.; Sun, W.; Sun, W.-B.; Cheng, W.; Li, W.; Song, X.; Su, X.; Han, X.; Zhang, X.; Hou, X.-Y.; Min, X.; Zou, X.; Shen, X.; Gong, Y.; Zhu, Y.-H.; Zhou, Y.; Zhong, Y.; Hu, Y.; Fan, Y.; Yu, Y.; Yang, Y.; Li, Y.; Huang, Y.; Li, Y.; Huang, Y.; Xu, Y.; Mao, Y.; Li, Z.; Li, Z.; Tao, Z.; Ying, Z.; Cong, Z.; Qin, Z.; Fan, Z.-Y.; Yu, Z.; Jiang, Z.; and Wu, Z. 2025. MiniMax-01: Scaling Foundation Models with Lightning Attention. *ArXiv*.

Nawrot, P.; Chorowski, J.; Łańcucki, A.; and Ponti, E. M. 2022. Efficient Transformers with Dynamic Token Pooling. arXiv:2211.09761.

Nguyen, E.; Poli, M.; Durrant, M. G.; Kang, B.; Katrekar, D.; Li, D. B.; Bartie, L. J.; Thomas, A. W.; King, S. H.; Brixi, G.; Sullivan, J.; Ng, M. Y.; Lewis, A.; Lou, A.; Ermon, S.; Baccus, S. A.; Hernandez-Boussard, T.; Ré, C.; Hsu, P. D.; and Hie, B. L. 2024a. Sequence modeling and design from molecular to genome scale with Evo. *Science*, 386(6723): eado9336.

Nguyen, E.; Poli, M.; Faizi, M.; Thomas, A.; Wornow, M.; Birch-Sykes, C.; Massaroli, S.; Patel, A.; Rabideau, C.; Bengio, Y.; et al. 2024b. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. *Advances in neural information processing systems*, 36.

Notin, P.; Dias, M.; Frazer, J.; Marchena-Hurtado, J.; Gomez, A. N.; Marks, D. S.; and Gal, Y. 2022. Tranception: protein fitness prediction with autoregressive transformers and inference-time retrieval. *ArXiv*, abs/2205.13760.

Outeiral, C.; and Deane, C. M. 2024. Codon language embeddings provide strong signals for use in protein engineering. *Nature Machine Intelligence*, 6(2): 170–179.

Pagnoni, A.; Pasunuru, R.; Rodriguez, P.; Nguyen, J.; Muller, B.; Li, M.; Zhou, C.; Yu, L.; Weston, J. E.; Zettlemoyer, L. S.; Ghosh, G.; Lewis, M.; Holtzman, A.; and Iyer, S. 2024. Byte Latent Transformer: Patches Scale Better Than Tokens. *ArXiv*, abs/2412.09871.

Qiao, L.; Ye, P.; Ren, Y.; Bai, W.; Liang, C.; Ma, X.; Dong, N.; and Ouyang, W. 2024. Model Decides How to Tokenize: Adaptive DNA Sequence Tokenization with MxDNA. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving Language Understanding by Generative Pre-Training.

Roy, S.; Sural, S.; and Ganguly, N. 2024. Unlocking Efficiency: Adaptive Masking for Gene Transformer Models. In *European Conference on Artificial Intelligence*.

Roy, S.; Wallat, J.; Sundaram, S. S.; Nejdl, W.; and Ganguly, N. 2023. GeneMask: Fast Pretraining of Gene Sequences to Enable Few-Shot Learning. In *European Conference on Artificial Intelligence*.

Sanabria, M.; Hirsch, J.; and Poetsch, A. R. 2023. The human genome's vocabulary as proposed by the DNA language model GROVER. *bioRxiv*, 2023–07.

Scalzitti, N.; Kress, A.; Orhand, R.; Weber, T.; Moulinier, L.; Jeannin-Girardon, A.; Collet, P.; Poch, O.; and Thompson, J. D. 2021. Spliceator: multi-species splice site prediction using convolutional neural networks. *BMC bioinformatics*, 22: 1–26.

Schiff, Y.; Kao, C.-H.; Gokaslan, A.; Dao, T.; Gu, A.; and Kuleshov, V. 2024. Caduceus: Bi-directional equivariant long-range dna sequence modeling. *arXiv preprint arXiv:2403.03234*.

Sennrich, R.; Haddow, B.; and Birch, A. 2015. Neural Machine Translation of Rare Words with Subword Units. *ArXiv*, abs/1508.07909.

Shao, B.; and Yan, J. 2024. A long-context language model for deciphering and generating bacteriophage genomes. *Nature Communications*, 15(1): 9392.

Slagle, K. 2024. SpaceByte: Towards Deleting Tokenization from Large Language Modeling. *ArXiv*, abs/2404.14408.

Song, L.; Segal, E.; and Xing, E. P. 2024. Toward AI-Driven Digital Organism: Multiscale Foundation Models for Predicting, Simulating and Programming Biology at All Levels. *ArXiv*, abs/2412.06993.

Theodoris, C. V.; Xiao, L.; Chopra, A.; Chaffin, M. D.; Al Sayed, Z. R.; Hill, M. C.; Mantineo, H.; Brydon, E. M.; Zeng, Z.; Liu, X. S.; et al. 2023. Transfer learning enables predictions in network biology. *Nature*, 618(7965): 616–624.

Thoutam, V.; and Ellsworth, D. 2024. MSAMamba: Adapting Subquadratic Sequence Models To Long-Context DNA MSA Analysis. In *Submitted to The 14th International Conference on Information Science and Technology*. Under review.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; Rodriguez, A.; Joulin, A.; Grave, E.; and Lample, G. 2023. LLaMA: Open and Efficient Foundation Language Models. *ArXiv*, abs/2302.13971.

Trop, E.; Schiff, Y.; Marroquin, E. M.; Kao, C. H.; Gokaslan, A.; Polen, M.; Shao, M.; de Almeida, B. P.; Pierrot, T.; Li, Y. I.; et al. 2024. The Genomics Long-Range Benchmark: Advancing DNA Language Models.

van den Oord, A.; Vinyals, O.; and Kavukcuoglu, K. 2017. Neural Discrete Representation Learning. In *ArXiv*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Conference on Neural Information Processing Systems (NeurIPS)*, volume 30.

Wang, J.; Gangavarapu, T.; Yan, J. N.; and Rush, A. M. 2024. Mambabyte: Token-free selective state space model. *arXiv preprint arXiv:2401.13660*.

Wu, S.; Tan, X.; Wang, Z.; Wang, R.; Li, X.; and Sun, M. 2024. Beyond Language Models: Byte Models are Digital World Simulators. arXiv:2402.19155.

Wu, W.; Li, Q.; Li, M.; Fu, K.; Feng, F.; Ye, J.; Xiong, H.; and Wang, Z. 2025. GENERator: A Long-Context Generative Genomic Foundation Model. *arXiv preprint*.

Yang, S.; Kautz, J.; and Hatamizadeh, A. 2024. Gated Delta Networks: Improving Mamba2 with Delta Rule. arXiv:2412.06464.

Yang, S.; Wang, B.; Zhang, Y.; Shen, Y.; and Kim, Y. 2025. Parallelizing Linear Transformers with the Delta Rule over Sequence Length. arXiv:2406.06484.

Yang, Z.; Fan, X.; Lan, M.; Li, X.; You, Y.; Tian, L.; Church, G.; Liu, X.; and Gu, F. 2024. Multiomic foundation model predicts epigenetic regulation by zero-shot. *bioRxiv*, 2024–12.

Yang, Z.; Zhu, J.; and Su, B. 2025. SPACE: Your Genomic Profile Predictor is a Powerful DNA Foundation Model. In *International Conference on Machine Learning (ICML)*.

Yu, L.; Simig, D.; Flaherty, C.; Aghajanyan, A.; Zettlemoyer, L.; and Lewis, M. 2023. MEGABYTE: Predicting Million-byte Sequences with Multiscale Transformers. *ArXiv*, abs/2305.07185.

Zhai, J.; Gokaslan, A.; Schiff, Y.; Berthel, A.; Liu, Z.-Y.; Lai, W.-Y.; Miller, Z. R.; Scheben, A.; Stitzer, M. C.; Romay, M. C.; et al. 2025. Cross-species modeling of plant genomes at single-nucleotide resolution using a pretrained DNA language model. *Proceedings of the National Academy of Sciences*, 122(24): e2421738122.

Zhang, D.; Zhang, W.; Zhao, Y.; Zhang, J.; He, B.; Qin, C.; and Yao, J. 2023. DNAGPT: A generalized pre-trained tool for versatile DNA sequence analysis tasks. *arXiv preprint arXiv:2307.05628*.

Zhang, X.; Yang, M.; Yin, X.; Qian, Y.; and Sun, F. 2024. Deepgene: An efficient foundation model for genomics based on pan-genome graph transformer. *bioRxiv*, 2024–04.

Zhou, Z.; Ji, Y.; Li, W.; Dutta, P.; Davuluri, R.; and Liu, H. 2023. Dnabert-2: Efficient foundation model and benchmark for multi-species genome. *arXiv preprint arXiv:2306.15006*.

Zhou, Z.; Riley, R.; Kautsar, S.; Wu, W.; Egan, R.; Hofmeyr, S.; Goldhaber-Gordon, S.; Yu, M.; Ho, H.; Liu, F.; et al. 2025a. GenomeOcean: An Efficient Genome Foundation Model Trained on Large-Scale Metagenomic Assemblies. *bioRxiv*, 2025–01.

Zhou, Z.; Wu, W.; Ho, H.; Wang, J.; Shi, L.; Davuluri, R. V.; Wang, Z.; and Liu, H. 2025b. DNABERT-S: Pioneering species differentiation with species-aware DNA embeddings. *Bioinformatics*, 41: i255–i264.

Zhu, X.; Qin, C.; Wang, F.; Yang, F.; He, B.; Zhao, Y.; and Yao, J. 2024. CD-GPT: a biological foundation model bridging the gap between molecular sequences through central dogma. *bioRxiv*, 2024–06.

Zvyagin, M. T.; Brace, A.; Hippe, K.; Deng, Y.; Zhang, B.; Bohorquez, C. O.; Clyde, A.; Kale, B.; Perez-Rivera, D.; Ma, H.; et al. 2022. GenSLMs: Genome-scale language models reveal SARS-CoV-2 evolutionary dynamics. *bioRxiv*.

Reproducibility Checklist

1. General Paper Structure

1.1. Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes/partial/no/NA) yes

1.2. Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes/no) yes

1.3. Provides well-marked pedagogical references for less-familiar readers to gain background necessary to replicate the paper (yes/no) yes

2. Theoretical Contributions

2.1. Does this paper make theoretical contributions? (yes/no) no

If yes, please address the following points:

2.2. All assumptions and restrictions are stated clearly and formally (yes/partial/no) yes

2.3. All novel claims are stated formally (e.g., in theorem statements) (yes/partial/no) yes

2.4. Proofs of all novel claims are included (yes/partial/no) yes

2.5. Proof sketches or intuitions are given for complex and/or novel results (yes/partial/no) yes

2.6. Appropriate citations to theoretical tools used are given (yes/partial/no) yes

2.7. All theoretical claims are demonstrated empirically to hold (yes/partial/no/NA) NA

2.8. All experimental code used to eliminate or disprove claims is included (yes/no/NA) NA

3. Dataset Usage

3.1. Does this paper rely on one or more datasets? (yes/no) yes

If yes, please address the following points:

3.2. A motivation is given for why the experiments are conducted on the selected datasets (yes/partial/no/NA) yes

3.3. All novel datasets introduced in this paper are included in a data appendix (yes/partial/no/NA) yes

3.4. All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no/NA) NA

3.5. All datasets drawn from the existing literature (potentially including authors' own previously pub-

lished work) are accompanied by appropriate citations (yes/no/NA) yes

3.6. All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available (yes/partial/no/NA) yes

3.7. All datasets that are not publicly available are described in detail, with an explanation of why publicly available alternatives are not scientifically satisfying (yes/partial/no/NA) yes

4. Computational Experiments

4.1. Does this paper include computational experiments? (yes/no) yes

If yes, please address the following points:

4.2. This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting (yes/partial/no/NA) yes

4.3. Any code required for pre-processing data is included in the appendix (yes/partial/no) no

4.4. All source code required for conducting and analyzing the experiments is included in a code appendix (yes/partial/no) no

4.5. All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no) yes

4.6. All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes/partial/no) yes

4.7. If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results (yes/partial/no/NA) yes

4.8. This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks (yes/partial/no) yes

4.9. This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics (yes/partial/no) yes

4.10. This paper states the number of algorithm runs used to compute each reported result (yes/no) yes

4.11. Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information (yes/no) yes

4.12. The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank) (yes/partial/no) partial

4.13. This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments (yes/partial/no/NA) partial

Implementation Details

Pre-training Settings

Following (Dalla-Torre et al. 2023; Zhou et al. 2023), we utilize a pure DNA dataset for pre-training, `Multi-species Genomes` [1], with reference sequences (RefSeq) of multiple species to ensure generalization abilities of multiple domains from the National Center for Biotechnology Information (NCBI) database at https://www.ncbi.nlm.nih.gov. We pre-train the MergeDNA model (including the latent decoder and the local decoder) by AdamW optimizer (Loshchilov and Hutter 2019) for 100,000 iterations (randomly sampled datasets) with a basic learning rate of 1×10^{-4} and a batch size of 256. With the Ubuntu workstation, the model is pre-trained by 8 Nvidia A100-80G GPUs with a per-GPU batch size of 8 and a gradient accumulation time of 16 for five days.

Configuration	Local Enc.	Latent Enc.	Latent Dec.	Local Dec.
Embedding dim	1024			
Block number	4	20	4	2
Block type	Local-Attn	Attention	Attention	Local-Attn
Permanent	✓	✓	✗	✗
Parameters	51M	253M	51M	25M
Optimizer	AdamW			
(β_1, β_2)	$(0.9, 0.95)$			
Training iterations	100,000			
Weight decay	1×10^{-8}			
Base learning rate	1×10^{-4}			
Batch size	256			
LR scheduler	Cosine Annealing			
Warmup iterations	10,000			
Gradient clipping	1.0			

Table A1: Configuration of the network architecture and pre-training for MergeDNA. The Local-Attn or Attention blocks denote the local-window or self-attention block with Flash-Attention implementation.

Evaluation Setups

In most cases, we apply Supervised Fine-tuning (SFT) to evaluate the transfer capacity of pre-trained models to genomic and multi-omics downstream tasks. Following (Nguyen et al. 2024b; Zhou et al. 2023), adding the decoder head (*e.g.*, an MLP head) to a specific downstream task, the linear attention (RNN) or self-attention blocks in the pre-trained encoder models are frozen, while Low-Rank Adaptation (LoRA) strategy (Hu et al. 2021) is employed to parameter-efficiently fine-tuning the models by AdamW optimizer with a batch size of 32. For each task, if the benchmark and models have provided hyper-parameters, we follow the official settings, or we choose the best combinations of the basic learning rate $\{1 \times 10^{-5}, 5 \times 10^{-5}, 1 \times 10^{-4}\}$, the weight decay $\{0, 0.01\}$, the LoRA rank $\{4, 8, 16, 24, 48\}$, the LoRA alpha $\{8, 16, 24, 48, 96\}$, and the total fine-tuning epoch $\{5, 10\}$ on the validation set following the GUE benchmark and GenBench (Liu et al. 2024b). As for Protein Fitness Prediction task, we conduct zero-shot evluation by learning a linear regression model upon the embedding of the Latent Encoder. Overall, we report the averaged results over three runs with the optimal settings.

[1]Multi-species Genomes are originally provided in https://huggingface.co/datasets/InstaDeepAI/multi_species_genomes, which is further extended by DNABERT2 in https://github.com/MAGICS-LAB/DNABERT_2

Downstream Task Benchmarks

DNA Tasks with Genomics Benchmark

As proposed by (Grešová et al. 2023), three groups of basic genomic tasks are collected as balanced binary classification with top-1 accuracy in the Genomics Benchmark. As for the enhancer prediction, three datasets are provided for identifying enhancer regions in the mouse or human genome. As for the species classification, two datasets are selected for identifying sequences as either coding (exonic) or intergenic (non-coding) and classifying sequences as originating from humans or worms (C. elegans). Regarding the classification of regulatory elements, three datasets are utilized to categorize sequences as regulatory regions based on Ensembl annotations, identify open chromatin regions, or pinpoint non-TATA promoter regions in the human genome. Each example is a 200bp central sequence extracted from the reference genome and labeled by high-throughput functional assays. We utilize the fully reproduced results of various DNA models in GenBench (Liu et al. 2024b). Table A2 provides full results of the Genomics Benchmark.

DNA Tasks with Nucleotide Transformer (NT) Benchmark

The NT benchmark (Dalla-Torre et al. 2023) spans 18 diverse tasks that probe both local and distal regulatory logic, including ten histone-mark predictions (*e.g.*, H3K4me3), four promoter/enhancer subtasks, and four canonical splice-site detection variants. We adopt the original version of the NT benchmark, where sequences of 1–4 kbp surrounding epigenetic peaks are sampled from the hg38 assembly with chromosome-wise train/valid/test splits to prevent leakage. We use **Matthews Correlation Coefficient (MCC)** for imbalanced or noisy labels (histone marks, splice sites) and **F1 score** for balanced promoter/enhancer tasks. Table 2 presents the full results of the NT benchmark with popular DNA models.

DNA Tasks with GUE Benchmark

As proposed by DNABERT2 (Zhou et al. 2023), the GUE benchmark contains 24 datasets of 7 practical biological genome analysis tasks for 4 different species. All sequences are provided as FASTA strings (or BED coordinates) with an official 70/15/15 train/valid/test split; inputs range from 70 bp to 1 kb so that both local and distal regulatory logic is probed. We follow the original evaluation protocol and report MCC for all binary tasks, while the multi-class Covid variant classifier is scored with macro-F1. A concise task-wise description is given below.

(A) Promoter Detection (Human). Identify proximal promoters (-249 +50 bp around TSS) in the GRCh38 assembly. Positive examples come from EPDnew (Dreos et al. 2013); negatives are random intergenic regions matched by GC content, stratified into *TATA*, *non-TATA*, and their union (*all*)—yielding **3 datasets**. **(B) Core-Promoter Detection (Human).** Predict the 70 bp core promoter window (-34 +35 bp) immediately flanking the TSS, a harder

Method	HyenaDNA	Caduceus-16	DNABERT	DNABERT2	GENA-LM	NT-500M	VQDNA	MxDNA	ConvNova	GENERator	**MergeDNA**
Date	NeurIPS'23	ICML'24	Bioinfo'21	ICLR'24	NAR'23	NM'24	ICML'23	NeurIPS'24	ICLR'25	arXiv'25	**Ours**
# Params	6.6M	7.9M	86M	117M	113M	500M	93M	100M	1.7M	1.3B	380M
Mouse Enhancers	79.34	81.63	80.99	81.82	82.97	85.12	81.06	80.57	78.40	**87.10**	85.62
Human Enhancers Cohn	72.96	73.76	70.23	75.87	75.63	76.12	75.63	74.67	74.30	76.30	**76.54**
Human Enhancers Ensembl	90.33	84.48	89.19	90.75	91.07	92.44	90.41	93.13	90.00	91.20	**93.18**
Coding vs Intergenomic	90.97	93.72	93.64	93.58	93.24	95.76	94.35	95.28	94.30	95.90	**96.02**
Human vs Worm	96.24	95.57	95.84	97.39	96.98	97.51	97.23	97.64	96.70	**98.00**	97.65
Human Regulatory	93.08	87.30	88.16	87.94	88.10	93.79	90.92	**94.11**	87.30	92.80	93.49
Human OCR Ensembl	79.14	81.76	74.96	75.82	78.98	80.42	76.58	81.05	79.30	**82.30**	82.16
Human NonTATA Promoters	94.45	88.85	87.13	95.24	**96.60**	92.95	95.37	96.56	95.30	95.80	96.34

Table A2: **Full Results on Genomic Benchmarks**. Top-1 accuracy (%) averaged across three trials is reported for the latest DNA foundation models, where the best and the second best results are marked as the **bold** and underlined types.

Method	HyenaDNA	Caduceus-PS	DNABERT	NT-2500M-multi	DNABERT2	VQDNA	MxDNA	ConvNova	HybriDNA-7B	**MergeDNA**
Date	NeurIPS'23	ICML'24	Bioinfo'21	NM'23	ICLR'24	ICML'24	NeurIPS'24	ICLR'25	arXiv'25	**Ours**
# Params	6.6M	1.9M	86M	2.5B	117M	93M	100M	1.7M	7B	380M
Human TF-0	64.47	–	66.84	66.64	71.99	**72.48**	–	–	70.00	72.36
Human TF-1	70.74	–	70.14	70.28	76.0	76.43	–	–	74.47	**76.50**
Human TF-2	60.44	–	61.03	58.72	66.52	66.80	–	–	70.42	**70.48**
Human TF-3	39.78	–	51.89	51.65	58.54	58.92	–	–	**64.52**	61.12
Human TF-4	73.27	–	70.97	69.43	77.43	78.10	–	–	**85.03**	80.76
Mouse TF-0	56.25	–	44.42	63.31	56.76	58.34	–	–	**71.68**	68.23
Mouse TF-1	80.46	–	78.94	83.76	84.77	85.81	–	–	**87.75**	86.69
Mouse TF-2	78.14	–	71.44	71.52	79.32	80.39	–	–	**86.59**	82.85
Mouse TF-3	60.83	–	44.89	69.44	66.47	69.72	–	–	**87.62**	73.46
Mouse TF-4	46.25	–	42.48	47.07	52.66	54.73	–	–	**56.47**	54.82
Core Promoter (all)	66.18	–	68.90	70.33	69.37	**71.02**	–	–	66.50	70.78
Core Promoter (no TATA)	67.41	–	70.47	**71.58**	68.04	70.58	–	–	70.66	**70.91**
Core Promoter (TATA)	74.07	–	76.06	72.97	74.17	78.50	–	–	76.94	**78.54**
Promoter (all)	83.04	–	90.48	91.01	86.77	90.75	–	–	88.28	**91.02**
Promoter (no TATA)	91.03	–	93.05	94.00	94.27	94.40	–	–	94.73	**94.90**
Promoter (TATA)	66.36	–	61.56	**79.43**	71.59	74.52	–	–	73.59	77.27
Splice Reconstructed	77.76	–	84.07	89.35	84.99	89.50	–	–	**90.09**	89.95
H3	78.14	77.90	73.10	78.77	78.27	79.21	**82.14**	81.50	–	81.63
H3K14ac	56.71	54.10	40.06	56.20	52.57	54.46	68.29	**70.71**	–	69.09
H3K36me3	59.92	60.90	47.25	61.99	56.88	61.75	65.46	**68.31**	–	68.24
H3K4me1	44.52	48.80	41.44	55.30	50.52	53.28	54.97	56.60	–	**57.10**
H3K4me2	42.68	38.80	32.27	36.49	31.13	34.05	55.30	**57.45**	–	55.87
H3K4me3	50.41	44.00	27.81	40.34	36.27	39.10	63.80	**67.15**	–	66.84
H3K79me3	66.25	67.60	61.17	64.70	67.39	68.47	73.74	72.08	–	**73.80**
H3K9ac	58.50	60.40	51.22	56.01	55.63	56.63	63.15	68.10	–	**68.36**
H4	78.15	78.90	79.26	81.67	80.71	81.84	80.89	81.12	–	**82.06**
H4ac	54.15	52.50	37.24	49.13	50.43	50.69	65.14	**66.10**	–	65.22
Virus Covid Classification	25.88	–	55.50	73.04	71.02	74.32	–	–	74.02	**74.41**

Table A3: **Full Results on GUE benchmark**. MCC (%) or F1 (%) is reported for Epigenetic Marks Prediction, Human Transcription Factor (TF) Prediction, Mouse Transcription Factor Prediction, Core Promoter Detection, Promoter Detection, Splice Site Reconstructed, and Covid Variants Classification (Virus Covid). The best and the second best results are marked as the **bold** and underlined types.

variant of (A) due to shorter context. Positive/negative selection mirrors (A), giving **3 additional datasets**. **(C) Transcription-Factor Binding Site Prediction (Human).** Classify 101 bp regions centered on ChIP-seq peaks from ENCODE (161 TFs, 91 cell lines). GUE retains **5 representative TFs** after filtering trivial or extremely imbalanced cases; negatives are GC-matched genomic windows outside peaks. **(D) Splice-Site Prediction (Human).** Detect splice *donor* and *acceptor* sites within 400 bp windows extracted from GRCh38. To avoid saturation, the original 10k-sample dataset (Scalzitti et al. 2021) is adversarially augmented with hard negatives until MCC stabilizes, producing **1 challenging dataset**. **(E) Transcription-Factor Binding Site Prediction (Mouse).** Analogous to (C) but using mouse ENCODE ChIP-seq (78 TFs). Five TFs are randomly chosen following the same GC-matched negative selection, resulting in **5 datasets**. **(F) Epigenetic-Mark Prediction (Yeast).** Predict presence of 10 histone marks or nucleosome occupancy tracks in *S. cerevisiae*. Each sequence is 147 bp centered on experimentally validated sites downloaded from the JAIST repository; random genomic positions serve as negatives, yielding **10 datasets**. **(G) Covid Variant Classification (Virus).** Multi-class identification of *SARS-CoV-2* lineages (*Alpha*, *Beta*, …, *Zeta*) from 1 kb genome snippets obtained via GISAID's EpiCoV (Khare et al. 2021).

Multi-omics Downstream Tasks

RNA Splicing Site Prediction SpliceAI (Jaganathan et al. 2019) pairs 10kb sequences centered on annotated splice junctions with binary labels (*donor*, *acceptor*). Following (Liu et al. 2024b), we use the "chromosome-held-out" split: chromosomes 1–19 for training, 20 for validation, and 21–22 for testing, guaranteeing isoform-level independence. Performance is evaluated by the Area Under the ROC Curve (AUROC) for each site type and averaged across donor/acceptor.

Long-Range Benchmarks (LRB) We adopt two long-range tasks collected in LRB (Trop et al. 2024). **(a) Causal eQTL Variant Effect.** Given a 20kb locus and a candidate SNV, the task is to classify whether the variant modulates gene expression; ground truth is derived from GTEx v8 fine-mapping. We report AUROC on the held-out tissue set proposed by (Trop et al. 2024). **(b) Bulk RNA Expression.** Predict log-transformed gene expression in 54 GTEx tissues from a 40 kb upstream sequence window. Following LRB, we fit a linear regressor on frozen sequence embeddings and compute the coefficient of determination (R^2) on an unseen chromosome split.

Zero-shot Protein Fitness Prediction Deep Mutational Scanning (DMS) (AlQuraishi 2019) assays exhaustively mutate a protein coding sequence and measure functional fitness. We follow Evo (Nguyen et al. 2024a) and evaluate two representative datasets: Escherichia coli TEM-1 β-lactamase (Bacteria) and human BRCA1 RING domain (Human). All single- and double-mutant variants are used. True fitness values remain hidden during inference to emulate zero-shot generalization. We report **Spearman's Rank Correlation Coefficient (SRCC)** between model scores (negative log-likelihoods of mutated codons) and experimental fitness.

Extended Related Work

DNA Foundation Models

Researchers have started to build large-scale sequence models for genomes since the 2020s, inspired by advances in natural language processing (Vaswani et al. 2017; Touvron et al. 2023). Early attempts like DNABERT (Ji et al. 2021) and DNAGPTd (Zhang et al. 2023) demonstrated that Transformer architectures can be adapted to DNA by treating nucleotide sequences as a "language". Subsequent models greatly scaled up pre-training (GPN (Benegas, Batra, and Song 2023), Nucleotide Transformer variants (Dalla-Torre et al. 2023), and DNABERT2 (Zhou et al. 2023)), achieved genome-scale pre-training on human or multi-species data, and showed broad utility across diverse downstream genomic tasks (Grešová et al. 2023; Cheng et al. 2025). The methodologies of these DNA language models can be discussed along four key dimensions: (a) *long sequence modeling*, (b) *DNA tokenization strategies*, (c) *pre-training objectives*, and (d) *pre-training domains and downstream tasks*.

Long Sequence Modeling. A core technical challenge is modeling extremely long sequences (*e.g.*, many gene regions span 8k–1M bases). Several works have adopted state-space models (SSMs) (Gu and Dao 2023; Yang, Kautz, and Hatamizadeh 2024) for efficient long-range modeling: HyenaDNA (Nguyen et al. 2024b), MSAMamba (Thoutam and Ellsworth 2024), and Caduceus (Schiff et al. 2024) were among the first to use linear-time SSM architectures for DNA, enabling context lengths beyond what vanilla Transformers (Dao et al. 2022) can handle. More recently, MegaDNA (Shao and Yan 2024) introduced a hierarchical Transformer for genome sequences, while models like Evo2 (Brixi et al. 2025), LifeCode (Liu et al. 2025b), and HybriDNA (2025) (Ma et al. 2025) combined SSMs with self-attention mechanisms to balance memory efficiency and modeling accuracy (MiniMax et al. 2025). Meanwhile, alternative sequence learners have been tried: ConvNova (Bo et al. 2025) applies convolutional neural networks to genomic sequences, and DeepGene (Zhang et al. 2024) leverages a graph neural network over a pan-genome graph to encode DNA context.

DNA Vocabulary and Tokenization. Unlike natural language, DNA has no inherent word boundaries or semantics, and coding *vs.* non-coding regions have different significance (coding regions translate in triplets as codons) (Cooper 1981; Outeiral and Deane 2024). Accordingly, most recent DNA models (Nguyen et al. 2024a,b) forego complex tokenization and simply use the four nucleotides (A, C, G, T) as the base vocabulary. Some approaches employ fixed-length k-mers or subwords: DNABERT (Ji et al. 2021) and Nucleotide Transformer (Dalla-Torre et al. 2023) represent sequences as k-mer tokens, and DNABERT2 (Zhou et al. 2023) and GENA-LM (Ji et al. 2023) build a byte-pair encoding (BPE) vocabulary for DNA. Beyond static schemes, researchers have proposed dynamic data-driven tokenization, *e.g.*, VQDNA (Li et al. 2024a) and MxDNA (Qiao et al. 2024) learn custom DNA tokens via Vector Quantization (VQ) (van den Oord, Vinyals, and Kavukcuoglu 2017) and deformable convolution (Dai et al. 2017), automatically discovering higher-level nucleotide motifs.

Pre-training Objectives. The training strategies for DNA foundation models mirror those in NLP (Devlin et al. 2019), mainly falling into masked language modeling (BERT-style) or autoregressive generation (Radford et al. 2018). Using nucleotide k-mers or subwords, encoder-based models like DNABERT2 and GENERanno (Li et al. 2025a) employ masked sequence modeling (predicting masked bases or k-mers from context), which has proven effective for many short-range genomic annotation tasks (Zhou et al. 2023). In contrast, many large parameter DNA models adopt autoregressive training to better model long-range dependencies (Ji et al. 2023; Benegas, Batra, and Song 2023). Evo2 (Brixi et al. 2025) further adjusts loss weights between coding and non-coding regions to emphasize functionally important context. Meanwhile, GeneMask (Roy et al. 2023) and CM-GEMS (Roy, Sural, and Ganguly 2024) introduce adaptive or focused masking schemes to accelerate pre-training and improve sequence representation. Beyond reconstruction-based objectives, some works pursue alternative pre-training tasks: DNABERT-S (Zhou et al. 2025b) uses contrastive learning (Chen et al. 2020) to learn species-aware embeddings, and LifeCode (Liu et al. 2025b) incorporates knowledge distillation and cross-modal alignment into pre-training to fuse information across the central dogma (DNA–RNA–protein).

Pre-training Domains and Downstream tasks. The choice of pre-training data strongly influences a model's generalization ability. Most DNA foundation models have been trained on the human genome (Nguyen et al. 2024b) or a mixture of species (Zhou et al. 2023), enabling them to generalize across common genomic tasks in eukaryotes. However, some models target specific taxonomic domains: Evo (Nguyen et al. 2024a), for example, was trained on prokaryotic genomes and excels at bacterial and viral

tasks such as designing CRISPR–Cas9 target sequences. Likewise, AgroNT (Mendoza-Revilla et al. 2023), PlantCaduceus (Zhai et al. 2025), and PDLLMs (Liu et al. 2025a) are specialized on plant genomic data, and ProKaformer (Li et al. 2024b) is tailored to model microbial community (microbiome) genomes. DNABERT-S (Zhou et al. 2025b) builds species-differentiated embeddings for multi-species data, and GenomeOcean (Zhou et al. 2025a) leverages large-scale metagenomic assemblies to learn generalizable genome representations. After pre-training, DNA foundation models are evaluated on a range of downstream tasks. Common tasks include token-level predictions and sequence-level classifications or regressions (*e.g.*, predicting regulatory function or phenotype from a sequence). For instance, SegmentNT (de Almeida et al. 2024) uses a foundation model to partition genomes into meaningful segments at single-nucleotide resolution. Meanwhile, GPT models (Nguyen et al. 2024a; Zhu et al. 2024) can generate realistic DNA sequences (even whole genomes) in an autoregressive manner, and diffusion-based approaches (Li et al. 2024c) produce novel DNA sequences via latent diffusion. Finally, researchers are exploring hybrid AI systems that integrate general-purpose LLMs (Bai et al. 2023) with genomic expertise (Liu and Wang 2024; Li et al. 2025b), which combine open-source ChatGPT-like models with DNA foundation models to handle complex multi-step genomic applications.

Multi-omics Modeling

Beyond DNA-alone modeling, researchers are extending foundation models to multi-omics (Krishna et al. 2024; He et al. 2024), aiming to connect DNA with other molecular modalities like RNA and protein within the unified framework (Hayes et al. 2025; Nguyen et al. 2024a). Protein-centric models have already achieved remarkable success in structure prediction and design (*e.g.*, AlphaFold-2 (Jumper et al. 2021)), which motivates developing DNA-centric multi-omics models that leverage genomic information. Recent research (Ji et al. 2023; Yang et al. 2024) uses DNA sequence models to predict downstream molecular phenotypes (*e.g.*, protein function or chromatin profiles) across species. Based on the central dogma principle (Cooper 1981), current DNA-centered multi-omics models generally fall into two broad categories.

The *first category* seeks to predict protein-level properties and functions directly from DNA sequences, effectively learning the end-to-end mapping from genome to proteome. Evo (Nguyen et al. 2024a) pioneered this approach by modeling sequence-to-function relationships at the genome scale, and its successor Evo2 (Brixi et al. 2025) further expanded to modeling genomes across all domains of life. AIDO (Song, Segal, and Xing 2024) and HybriDNA (Ma et al. 2025) are designed to handle ultra-long inputs (full genomes), CD-GPT (Zhu et al. 2024), LifeCode (Liu et al. 2025b), and GENERator (Wu et al. 2025) incorporate explicit transcription and translation tasks into their objectives, and frameworks like LucaOne (He et al. 2024) uses supervised multi-task learning to unify representations of nucleic acids and proteins for diverse predictive tasks. The *second category* allows gene-to-expression modeling, focusing on predicting cellular and molecular readouts from DNA sequence (Avsec et al. 2021, 2025) and bridging genotype to phenotype at the regulatory or cellular level. Enformer (Avsec et al. 2021) first demonstrated that a deep Transformer can accurately predict gene expression profiles from DNA by accounting for long-range genomic interactions. Geneformer (Theodoris et al. 2023) showed the benefit of transfer learning for network biology predictions, IsoFormer (Garau-Luis et al. 2024) and SPACE (Yang, Zhu, and Su 2025) introduced multi-modal alignment and mixture-of-experts to enhance regulatory sequence predictions, and AlphaGenome (Avsec et al. 2025) was recently proposed as a unified model that can predict a wide range of cell-level genomic assay results from the long DNA context.

Byte Architectures

Most traditional language models rely on a pre-computed tokenizer to convert raw data into tokens—commonly using subword algorithms like BPE (Sennrich, Haddow, and Birch 2015) or SentencePiece (Kudo and Richardson 2018) to segment the input byte sequence. Complex dynamic tokenization methods, such as Dynamic Pooling (Nawrot et al. 2022) and Dynamic Vocabulary (Liu et al. 2024a), still involve training separate tokenization modules and do not fundamentally escape the constraints of discrete token boundaries. This tokenization step might introduce information loss or segmentation artifacts, motivating a shift toward models that operate directly on byte sequences (Pagnoni et al. 2024; Hwang, Wang, and Gu 2025). With the advent of more efficient long-context sequence models (Gu and Dao 2023; Yang et al. 2025; Liu et al. 2024c), a new class of byte-level architectures has emerged to eliminate tokenization altogether (Wang et al. 2024). MegaByte (Yu et al. 2023) and SpaceByte (Slagle 2024) pioneered multiscale Transformers and SSMs that process sequences at the byte level, enabling context lengths on the order of millions of characters without arbitrary tokenization. Building on this, ByteScale (Ge et al. 2025) scaled up token-free models to unprecedented sizes (allowing training with 2-million token contexts on thousands of GPUs), while bGPT (Wu et al. 2024) extended the tokenizer-free approach to multimodal data. Very recently, BLT (Pagnoni et al. 2024) performs entropy-based segmentation of byte sequences using a latent Perceiver-style architecture (Jaegle et al. 2021b,a), whereas HNet (Hwang, Wang, and Gu 2025) provides a fully differentiable tokenization scheme, enabling end-to-end learning of how to chunk byte sequences during model training.