JanusDNA: A Powerful Bi-directional Hybrid DNA Foundation Model

Qihao Duan[1,4,5], Bingding Huang[2], Zhenqiao Song[3],
Irina Lehmann[1], Lei Gu[4†], Roland Eils[1,5,6,7†], Benjamin Wild[1†]
[1]Berlin Institute of Health, Charité – Universitätsmedizin Berlin
[2]College of Big Data and Internet, Shenzhen Technology University
[3]Language Technologies Institute, Carnegie Mellon University
[4]Epigenetics Laboratory, Max Planck Institute for Heart and Lung Research
[5]Department of Mathematics and Computer Science, Freie Universität Berlin
[6]Health Data Science Unit, Heidelberg University Hospital and BioQuant
[7]Intelligent Medicine Institute, Fudan University
[†]Corresponding authors
{qihao.duan, irina.lehmann, roland.eils, benjamin.wild}@bih-charite.de
huangbingding@sztu.edu.cn zhenqias@andrew.cmu.edu
lei.gu@mpi-bn.mpg.de

Abstract

Large language models (LLMs) have revolutionized natural language processing and are increasingly applied to other sequential data types, including genetic sequences. However, adapting LLMs to genetics presents significant challenges. Capturing complex genomic interactions requires modeling long-range global dependencies within DNA sequences, where interactions often span over 10,000 base pairs, even within a single gene. This poses substantial computational demands under conventional model architectures and training paradigms. Additionally, traditional LLM training approaches are suboptimal for DNA sequences: autoregressive training, while efficient for training, only supports unidirectional sequence understanding. However, DNA is inherently bidirectional. For instance, bidirectional promoters regulate gene expression in both directions and govern approximately 11% of human gene expression. Masked language models (MLMs) enable bidirectional understanding. However, they are inefficient since only masked tokens contribute to loss calculations at each training step. To address these limitations, we introduce **JanusDNA, the first bidirectional DNA foundation model built upon a novel pretraining paradigm**, integrating the optimization efficiency of autoregressive modeling with the bidirectional comprehension capability of masked modeling. JanusDNA's architecture leverages **a Mamba-Attention Mixture-of-Experts (MoE) design**, combining the global, high-resolution context awareness of attention mechanisms with the efficient sequential representation learning capabilities of Mamba. The MoE layers further enhance the model's capacity through sparse parameter scaling, while maintaining manageable computational costs. Notably, **JanusDNA can process up to 1 million base pairs at single-nucleotide resolution on a single 80GB GPU using its hybrid architecture**. Extensive experiments and ablation studies demonstrate that JanusDNA achieves new state-of-the-art performance on three genomic representation benchmarks. Remarkably, JanusDNA surpasses models with **250x** more activated parameters, underscoring its efficiency and effectiveness. Code available at https://github.com/Qihao-Duan/JanusDNA.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

1 Introduction

Modeling the *language* of DNA is crucial for investigating its biological function, offering potential advancements in understanding genotype-phenotype associations, disease diagnosis, and new drug development [1]. Large language models (LLMs) have demonstrated remarkable success in large-scale nonlinear representation learning for natural language processing (NLP) tasks, such as text generation, translation, and summarization [2, 3, 4]. This success has inspired researchers to explore LLM applications in other domains [5, 6], including bioinformatics [7]. However, applying general LLMs directly to DNA sequence data is challenging. DNA sequences are represented as series of nucleotides, lacking clear semantic meaning and posing challenges for interpretation by LLMs. Additionally, non-coding regions in DNA can be remotely related to coding regions both upstream and downstream, necessitating models capable of processing long-range dependencies while maintaining bidirectional understanding. These factors make it challenging to apply LLMs directly to DNA sequence data without significant modifications or adaptations [8]. Some models have been developed specifically for DNA sequence representation [9, 10, 11]. However, these models still face some limitations.

Current limitations (1) **Limited Sequence Length and Low Resolution:** Capturing complex genomic interactions requires modeling long-range dependencies within DNA sequences, where interactions can span over 10,000 base pairs even within a single gene [12]. This necessitates a model that can effectively handle long-range dependencies and relationships within the sequence. However, many current models solely rely on global attention mechanisms inspired by their superior success in natural language applications [13, 14], yet they often struggle to effectively process long genomic sequences and uncover meaningful long-range interactions [11]. K-mer tokenization is frequently used to expand the context window of DNA sequence models [9, 10]. However, this method introduces a trade-off between sequence length and resolution [15], potentially leading to the loss of crucial information, especially in cases where single nucleotide polymorphisms (SNPs) are essential for understanding gene function. (2) **Unidirectional Understanding:** Many genomic processes are influenced by bidirectional interactions, with essential regulatory elements located both upstream and downstream of key genomic regions. For example, bidirectional promoters initiate transcription in both orientations [16, 17]. Additionally, intergenic enhancers transcribed predominantly bidirectionally often function as weak promoters in both directions. Conversely, for elements with unidirectional transcription (both enhancers and promoters), transcription direction typically correlates with the orientation in which the element functions as a promoter in vivo [18]. Accurately modeling these bidirectional interactions is crucial for understanding genomic function and making precise predictions, imposing significant demands on model architecture and training strategies [8]. However, some existing decoder-only models based on State Space Models (SSMs) [15, 19, 20] are primarily unidirectional or limited in their capacity to effectively understand bidirectional context, thus constraining their ability to comprehensively capture these complex interactions. (3) **Low Training Efficiency:** Long-range modeling and bidirectional understanding of DNA sequences both require substantial computational resources and memory, especially for those requiring attention for a more global representation. Therefore, training efficiency significantly impacts the model's performance, especially under limited computational resources. Most bidirectional models are trained using masked training paradigms, such as BERT [21], which utilize only a small fraction of tokens (typically 15%) for loss calculation at each step. This limitation can hinder the model's ability to learn effectively from the entire sequence within limited training steps, thereby requiring more training epochs to adequately cover the full training data.

Additionally, the masking process itself introduces extra computational overhead. In contrast, autoregressive (next-token prediction) training is more efficient, as nearly all tokens contribute to the loss at each training step, allowing the model to learn more effectively within a fixed number of steps as sequence length increases [22]. However, it's important to note that autoregressive models are inherently unidirectional, limiting their ability to incorporate bidirectional context.

In response to the aforementioned issues, we introduce **JanusDNA**[1], the first bidirectional DNA foundation model built upon a novel pretraining paradigm. Our architecture employs two principal strategies: (1) **Hybrid Architecture:** To achieve powerful global understanding while maintaining

[1]Janus, the ancient Roman god of transitions and duality, is symbolized by two faces gazing in opposite directions.

computational efficiency for long contexts, we integrate the strengths of state-space models (SSMs) [23] and Mixture-of-Experts (MoE) designs [24, 25] into attention mechanisms [26], enabling the model to effectively capture long-range global dependencies and complex interactions within DNA sequences. (2) **Bidirectional Efficient Training:** While preserving the bidirectional understanding of DNA sequences typically achieved through masked training, we significantly improve learning efficiency by computing the loss over all tokens in each training, same as in autoregressive modeling. Notably, **JanusDNA is capable of processing up to 1 million base pairs at single-nucleotide resolution with global attention on a single 80GB GPU**, making it suitable for large-scale understanding in genomic research. We evaluated JanusDNA on 35 diverse genomic tasks to showcase its superior global understanding as well as long-range representation ability.

In summary, our contributions are as follows:

- We propose **JanusDNA**, a novel bidirectional DNA foundation model capable of capturing global long-range dependencies and interactions at single-nucleotide resolution.
- We introduce an efficient **Hybrid Mamba-Attention-MoE architecture** designed for processing ultra-long genomic sequences within practical computational budgets.
- We present **Janus Modeling**, a novel and efficient pretraining paradigm that effectively combines the strengths of autoregressive and masked modeling, facilitating effective global bidirectional learning.
- We demonstrate **state-of-the-art performance** across diverse genomic benchmarks, outperforming significantly larger models. In particular, JanusDNA **significantly surpasses the expert model Enformer on eQTL prediction tasks**, despite having far fewer parameters.

2 Preliminary and Related Work

2.1 Large Language Model Pretraining Paradigms

Autoregressive Language Modeling (ALM) is a generative pretraining paradigm in which the model predicts the next token in a sequence given all previous tokens. Trained on large corpora, the model learns statistical properties to generate coherent text. The training objective is:

$$\mathcal{L}_{\text{CLM}} = -\sum_{t=1}^{T} \log P(x_t | x_1, x_2, \ldots, x_{t-1}), \tag{1}$$

where T is the sequence length, and x_t denotes the token at position t. The model generates text by sampling from the learned probability distribution over the vocabulary at each time step [15, 19, 20, 4, 27]. Each token contributes to the overall loss, and the model minimizes the average loss across all tokens. However, to maximize generative performance [28], ALM is unidirectional, causing a limited ability to model bidirectional contexts, which is crucial for DNA sequence understanding [1, 29].

Masked Language Modeling (MLM) is a non-causal pretraining paradigm where the model predicts masked tokens in a sequence using surrounding context. The training objective is:

$$\mathcal{L}_{\text{MLM}} = -\sum_{i=1}^{N} \log P(x_i | x_{j_1}, x_{j_2}, \ldots, x_{j_k}), \tag{2}$$

where N is the total number of tokens, x_i is the masked token, and $x_{j_1}, x_{j_2}, \ldots, x_{j_k}$ are the unmasked tokens. This approach enables the model to learn bidirectional representations, capturing dependencies in both directions [1, 28, 29, 9, 10, 30, 11, 31]. However, MLMs mostly follow the BERT-style training paradigm, which masks a fixed percentage of tokens in the input sequence, e.g., 15% [8, 9, 10]. This can lead to inefficiencies, as only a small fraction of the data is used for loss computation during each iteration. In contrast, autoregressive training paradigms take advantage of nearly the entire data, significantly improving training efficiency and overall performance.

A detailed review of related work on DNA language models is provided in Appendix A.

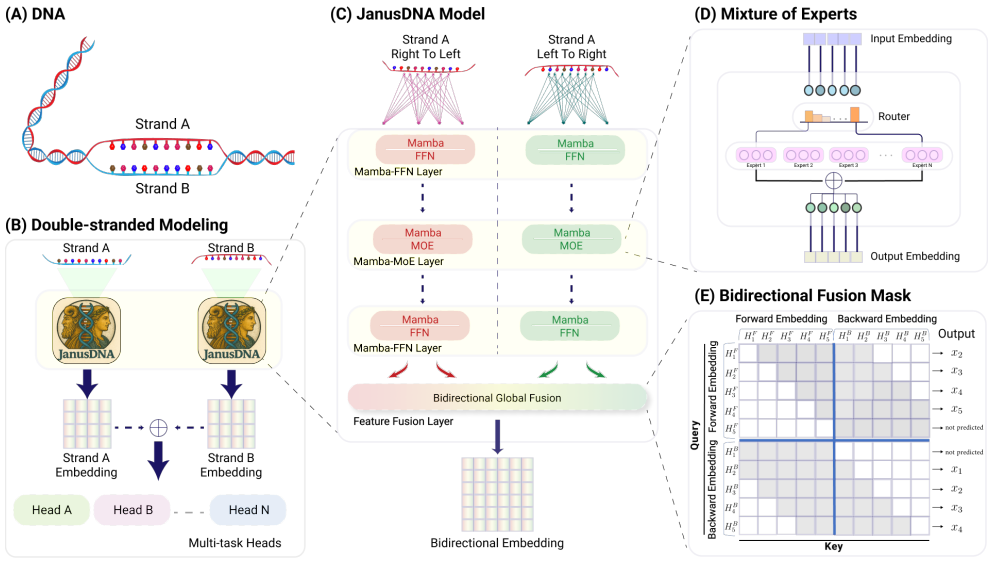

Figure 1: **The JanusDNA Architecture for Bidirectional DNA Modeling.** JanusDNA employs a hierarchical bidirectional strategy to comprehensively model DNA sequences. **(A)** DNA, with its inherent double-stranded nature. **(B)** The model processes both the forward and reverse complement strands independently in parallel to capture complete biological context, with their embeddings subsequently combined for downstream tasks. **(C)** The core JanusDNA model architecture processes a single input strand using parallel left-to-right and right-to-left pathways. Each pathway consists of Mamba-FFN and Mamba-MoE layers for effective and efficient sequential encoding. **(D)** The MoE architecture enhances model capacity and specialization by dynamically and sparsely routing inputs to a subset of expert networks, enabling efficient computation and improved representation learning. **(E)** The Bidirectional Global Fusion mechanism, utilizing a specific attention mask, integrates the forward and backward representations from (C) to ensure that each nucleotide's embedding is informed by its complete sequence context.

3 JanusDNA

We propose Janus modeling, an efficient bidirectional training method with global attention, and JanusDNA, a powerful hybrid DNA foundation model.

As illustrated in Figure 1, JanusDNA processes bidirectional DNA sequences from both left-to-right and right-to-left using two independent stacks of Mamba and Mixture-of-Experts (MoE) layers. These stacks generate forward and backward representations independently, ensuring no information leakage. The two representations are then fused to create a unified representation that encapsulates bidirectional information. Each token position is predicted based on all upstream and downstream tokens. The following sections detail the efficient bidirectional training method with global attention – **Janus Modeling**, and the hybrid Mixture-of-Experts (MoE) architecture – **JanusDNA**.

3.1 Bidirectional Efficient Training

As discussed in Section 2, conventional pretraining paradigms face a trade-off: Masked Language Models (MLMs) offer bidirectional understanding but suffer from low training efficiency due to sparse loss signals, especially for those requiring global attention, while Autoregressive Models are efficient in training but inherently unidirectional. To overcome this, we introduce *Janus modeling*, a novel pretraining objective designed to achieve efficient, fully bidirectional sequence understanding with global attention, as illustrated in Figure 2.

The core idea of Janus modeling is to predict *every* token x_t in a sequence of length T using its complete bidirectional context, i.e., all tokens preceding x_t $(x_1, \ldots, x_{t-1})$ and all tokens succeeding x_t $(x_{t+1}, \ldots, x_T)$. The training objective is therefore:

$$\mathcal{L}_{\text{bidirectional}} = -\sum_{t=1}^{T} \log P(x_t | x_1, \ldots, x_{t-1}, x_{t+1}, \ldots, x_T) \tag{3}$$

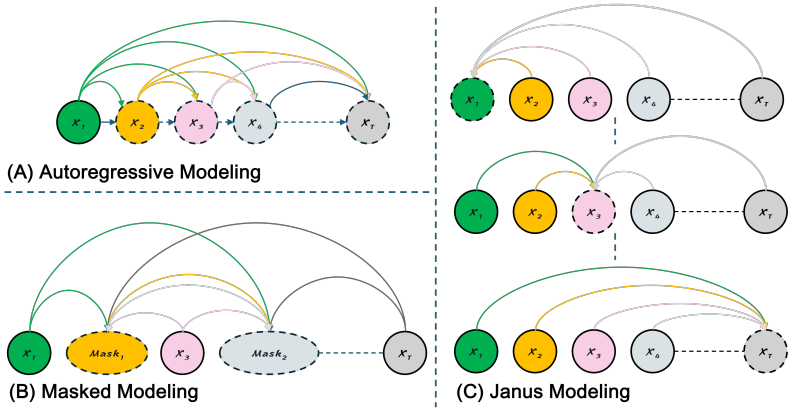

Figure 2: **Modeling Interpretation.** Janus modeling treats each token as a target for loss calculation, enabling higher training efficiency compared to masked modeling by full sequence learning while keeping bidirectional context understanding.

This objective ensures that every token contributes to the loss, maximizing training efficiency, while simultaneously demanding bidirectional comprehension.

To realize this objective, Janus modeling employs a two-stage process: independent context encoding followed by a global fusion mechanism:

Independent Context Encoding: The input sequence $X = (x_1, \ldots, x_T)$ is processed by two parallel and independent stacks of layers (detailed in Section 3.2):

- A **forward pass** processes the sequence from left-to-right, generating a sequence of hidden states $\mathcal{R}_{\text{fwd}} = (H_1^F, H_2^F, \ldots, H_T^F)$. Each hidden state H_t^F is a function of the input tokens $(x_1, \ldots, x_t)$ and primarily captures information about the left context of x_t.

$$H_t^F = \text{ForwardEncoder}(x_1, \ldots, x_t) \tag{4}$$

- A **backward pass** processes the sequence from right-to-left, generating a sequence of hidden states $\mathcal{R}_{\text{bwd}} = (H_1^B, H_2^B, \ldots, H_T^B)$. Each hidden state H_t^B is a function of the input tokens $(x_T, \ldots, x_t)$ and primarily captures information about the right context of x_t.

$$H_t^B = \text{BackwardEncoder}(x_T, \ldots, x_t) \tag{5}$$

These two sets of representations, $\mathcal{R}_{\text{fwd}}$ and $\mathcal{R}_{\text{bwd}}$, are generated independently, ensuring no premature information leakage between past and future contexts before the explicit fusion step. The entire model is trained end-to-end using $\mathcal{L}_{\text{bidirectional}}$ from Equation 3.

Bidirectional Global Fusion: To compute $P(x_t|x_1, \ldots, x_{t-1}, x_{t+1}, \ldots, x_T)$ for each x_t as per Equation 3, the left-context information captured in $\mathcal{R}_{\text{fwd}}$ and the right-context information captured in $\mathcal{R}_{\text{bwd}}$ must be integrated. This is organically achieved via a global attention mechanism, specifically implemented with FlexAttention [26] for efficiency. The representations from both passes, $\mathcal{R}_{\text{fwd}} = (H_1^F, \ldots, H_T^F)$ and $\mathcal{R}_{\text{bwd}} = (H_1^B, \ldots, H_T^B)$, are concatenated to form a combined input sequence for the attention layer: $R_{\text{input}} = [H_1^F, \ldots, H_T^F, H_1^B, \ldots, H_T^B]$. This R_{input} sequence has a length of $2T$. The core of the fusion lies in a carefully designed attention mask, $\mathcal{M}_{ij}$, which dictates how tokens in R_{input} can attend to each other. This mask ensures that the prediction for x_t is based only on H_k^F for $k < t$ and H_j^B for $j > t$, preventing information leakage. The mask, also illustrated in Figure 1(E), is defined as:

$$\mathcal{M}_{ij} = \begin{cases} q_{\text{idx}} \geq kv_{\text{idx}}, & \text{if } kv_{\text{idx}} < T \text{ and } q_{\text{idx}} < T, \\ q_{\text{idx}} \leq kv_{\text{idx}}, & \text{if } kv_{\text{idx}} \geq T \text{ and } q_{\text{idx}} \geq T, \\ kv_{\text{idx}} \geq T + q_{\text{idx}} + 2, & \text{if } kv_{\text{idx}} \geq T \text{ and } q_{\text{idx}} < T, \\ q_{\text{idx}} \geq kv_{\text{idx}} + T + 2, & \text{if } kv_{\text{idx}} < T \text{ and } q_{\text{idx}} \geq T. \end{cases} \tag{6}$$

Here, q_{idx} and kv_{idx} are the 0-indexed indices of the query and key-value pairs within the $2T$-length R_{input}, respectively. T is the original sequence length. The mask $\mathcal{M}_{ij}$ is a binary matrix where allowed

attentions are 1 and disallowed are 0 (or $-\infty$ after softmax). The first two cases handle causal attention within the $\mathcal{R}_{\text{fwd}}$ and $\mathcal{R}_{\text{bwd}}$ segments, respectively. The third and fourth cases manage the cross-attention between $\mathcal{R}_{\text{fwd}}$ and $\mathcal{R}_{\text{bwd}}$ segments, precisely controlling information flow to maintain the integrity of bidirectional prediction without information leakage relative to the token being predicted.

The output of this attention mechanism provides a fused representation H_t^{final} for each token x_t, which is then used to make the final prediction following a repositioning step, where the representations of the same token are summed, except for the first and last tokens, due to their representations containing information from only one direction, while optimizing $\mathcal{L}_{\text{bidirectional}}$.

This Janus modeling approach, as conceptually depicted in Figure 2 (C), enables each token to be a learning target informed by its full bidirectional context, thereby enhancing training efficiency compared to traditional MLMs (Figure 2 (B)) and overcoming the limitations of unidirectional models (Figure 2 (A)).

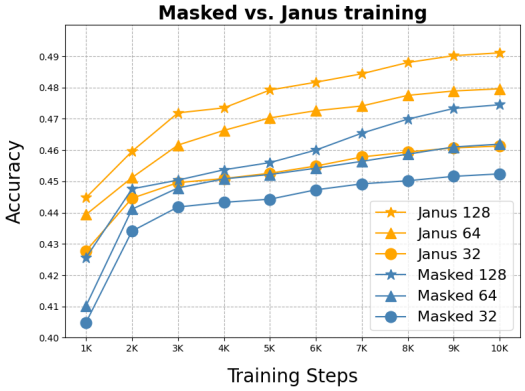

Figure 3: **Superior Learning Efficiency of Janus Modeling.** Comparison of last-token prediction accuracy between Janus modeling and conventional masked modeling over 10k training steps. Janus modeling consistently achieves higher accuracy for the same model architecture and training duration, demonstrating its enhanced efficiency in learning from sequence data. The number in the legend indicate hidden dimention.

Empirical Validation of Training Efficiency To empirically assess the learning efficiency of Janus modeling against conventional bidirectional approaches, masked modeling, we conducted a comparative experiment focused on last-token prediction. This task was chosen as it allows a direct comparison: both a standard masked language model and our Janus modeling approach can predict the final token x_T given the preceding context $x_1, \ldots, x_{T-1}$, ensuring a fair basis for evaluation.

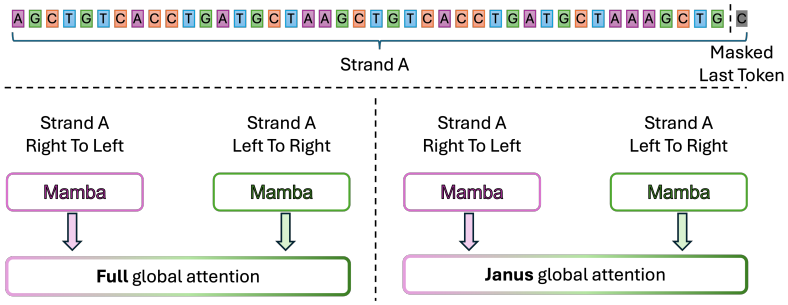

Figure 4: **Janus and Masked Modeling Efficiency Validation.** Both models are pre-trained from scratch using identical hyperparameter settings, with the only difference being the masking strategy applied in the final fusion attention layer. Last-token prediction is used to enable a fair comparison of learning efficiency between the two models.

For this demonstration, we configured two model variants as Figure 4: **Janus Model**: A single-layer bidirectional Mamba architecture equipped with a FlexAttention layer [26] for bidirectional fusion, utilizing the Janus-specific attention mask $\mathcal{M}_{ij}$ (Equation 6). This model predicts x_t based on the fused bidirectional context of $x_1, \ldots, x_{t-1}, x_{t+1}, \ldots, x_T$ as per the Janus modeling objective. **Masked Language Model**: As a baseline, we construct a comparable single-layer Mamba architecture followed by the same FlexAttention layer without an attention mask, trained using the conventional masked language modeling (MLM) objective, where a fraction of tokens (typically 15%) contribute to the loss.

Both models were trained on the human reference genome (HG38) [32] for 10,000 steps. Each training sample had a context length of 131,072 tokens, processed with a batch size of 1. We evaluated performance across three hidden dimensions: 32, 64, and 128, keeping other hyperparameters

consistent. Pre-training for Janus models, benefited from their sparse attention masks, takes around 27 minutes per 1,000 steps, nearly twice as fast as masked models.

For evaluation, both the masked and Janus models are given the full input sequence with the last token x_T masked. This evaluation is conducted on a test set containing 1,920 sequences.

The results, presented in Figure 3, clearly demonstrate that models trained with the Janus modeling method significantly outperformed those trained with the masked modeling approach in prediction accuracy, given the same number of training steps. This finding substantiates that Janus modeling is more effective at leveraging the DNA sequences for learning, thereby achieving superior training efficiency while maintaining robust bidirectional understanding.

3.2 Hybrid Mixture-of-Experts (MoE) Model

After introducing the bidirectional efficient training method, we developed a bidirectional backbone model to enhance sequence representation. Leveraging the efficient bidirectional fusion method, we propose JanusDNA, a hybrid model that integrates the strengths of Mamba, Attention, and MoE.

Architecture As illustrated in Fig. 1, JanusDNA incorporates three primary components for sequence representation: Mamba, MoE, and Attention. Mamba, a State Space Model (SSM) [23], efficiently encodes input sequences using high-dimensional parameters. Compared to traditional Transformer architectures, Mamba is more memory- and computationally efficient, making it particularly suitable for processing long DNA sequences.

The Mixture-of-Experts (MoE) architecture provides a scalable approach to significantly increase model capacity without proportionally increasing computational costs during training and inference [33]. In JanusDNA, MoE layers replace the feedforward network (FFN) layers in the Mamba blocks at a specific ratio following [24], achieving a balance between performance and efficiency. To ensure balanced utilization of the experts, an auxiliary loss is introduced [34], encouraging the model to distribute input data evenly across all experts. This auxiliary loss is computed based on the router logits, which represent the probabilities of selecting each expert for a given input, as shown in Eq. 7. Given N experts indexed by $i = 1$ to N and a batch $\mathcal{B}$ with T tokens, the auxiliary loss is computed as the scaled dot-product between vectors f and P:

$$\mathcal{L}_{\text{total}} = \alpha \cdot N \cdot \sum_{i=1}^{N} f_i \cdot P_i \tag{7}$$

where f_i is the fraction of tokens dispatched to expert i,

$$f_i = \frac{1}{T} \sum_{x \in \mathcal{B}} 1\{\text{argmax } p(x) = i\}, \tag{8}$$

and P_i is the fraction of the router probability allocated to expert i,

$$P_i = \frac{1}{T} \sum_{x \in \mathcal{B}} p_i(x). \tag{9}$$

Here, $p_i(x)$ represents the probability of routing token x to expert i, while T is the total number of tokens in the batch $\mathcal{B}$. This auxiliary loss encourages balanced utilization of all experts, improving overall model performance.

Bidirectional sequences are processed through independent stacks of Mamba and MoE layers, each designed to enhance the model's representational capacity. The Mamba layers efficiently capture local contextual dependencies within the sequence, leveraging their memory-efficient state-space modeling, while the MoE layers provide sparse scaling to enhance the model's representational capacity without incurring proportional computational overhead.

The forward and backward representations generated by these layers are fused using FlexAttention [26], an optimized attention mechanism that supports sparse masks with reduced memory consumption. This fusion enables the model to integrate both local and long-range global information streams, resulting in a comprehensive bidirectional representation for improved performance.

Reverse Complement In the double-helix DNA structure, each strand contains semantically equivalent information, with the *reverse complement* (RC) of a strand oriented in the opposite direction and its bases complemented relative to the *forward* strand (A paired with T, and C paired with G) [8]. However, recognizing both the forward and RC versions of non-palindromic motifs, such as *GATA* and *TATC*, poses a significant challenge, as it is akin to learning two distinct motifs [35]. To address this, we adopt a post-hoc reverse complement representation strategy [36]. Specifically, the DNA sequence and its reverse complement are processed in parallel using the identical model. The resulting representation vectors are then pooled to form a unified, enriched representation as shown in Figure 1 (B). This approach enables the model to effectively learn from both the original and reverse complement sequences, improving its ability to capture intricate patterns and relationships within DNA sequences, thereby enhancing performance across various tasks. We further conduct ablation experiments on reverse complement design in Appendix B.1.3.

4 Experiments

4.1 Pre-training on Human Reference Genome

To ensure a fair comparison with prior work, we pre-train our model on only the human reference genome (HG38 [32]) following the training setup described in [8]. Specifically, we adopt single nucleotide-level tokenization to capture high-resolution input sequences and avoid overlooking critical DNA information that may be lost when using k-mer tokenization, commonly used in attention-based models [15]. Additionally, single nucleotide-level tokenization is employed to facilitate downstream research on Single Nucleotide Polymorphisms (SNPs). Please note that performance on downstream tasks depends on the model architecture, as well as the composition and diversity of the pretraining data [11, 19, 20, 10]. Here, we specifically focus only on the architecture, and thus use only the human reference genome for pretraining to ensure a fair comparison.

4.2 Downstream Tasks

We evaluate our model on three different benchmarks: Genomic Benchmark [37], Nucleotide Transformer Benchmark [11], and DNALONGBENCH [38]. We follow all benchmark settings of Genomic Benchmark and Nucleotide Transformer Benchmark as described in [8]. Accordingly, we adopt the reported results from [8] as our reference. Considering the practical computational cost of sparse MoEs, we adjust the model's hidden size to match or slightly reduce the number of activated parameters compared to the baseline [8], ensuring a fair comparison. As we introduce the model with a middle attention layer in the ablation experiments (Section B.1.1), we present results for models both with and without mid-attention on the Genomic Benchmark and Nucleotide Transformer Benchmark.

4.2.1 Genomic Benchmark

Table 1: Genomic Benchmarks. Top-1 accuracy (↑) across 5-fold cross-validation (CV) for a supervised CNN baseline, pretrained HyenaDNA, Caduceus models, ConvNova and JanusDNA models. Best values per task are **bolded**, second best are underlined. Error bars indicate the difference between the maximum and minimum values across 5 random seeds used for CV.

Models activated param	CNN (264K)	HyenaDNA (436K)	Caduceus Ph (470K)	Caduceus PS (470K)	ConvNova (386K)	JanusDNA mlp w/ Mid-Attn (426K)	JanusDNA mlp w/o Mid-Attn (431K)
Mouse Enhancers	0.715±0.087	0.780±0.025	0.754±0.074	**0.793**±0.058	0.784±0.009	0.770±0.048	0.769±0.029
Coding vs. Intergenomic	0.892±0.008	0.904±0.005	0.915±0.003	0.910±0.003	**0.943**±0.001	0.912±0.003	0.911±0.001
Human vs. Worm	0.942±0.002	0.964±0.002	**0.973**±0.001	0.968±0.002	0.967±0.002	0.971±0.001	0.971±0.001
Human Enhancers Cohn	0.702±0.021	0.729±0.014	**0.747**±0.004	0.745±0.007	0.743±0.005	0.741±0.005	0.742±0.006
Human Enhancer Ensembl	0.744±0.122	0.849±0.006	0.893±0.008	**0.900**±0.006	**0.900**±0.004	0.897±0.004	0.899±0.004
Human Regulatory	0.872±0.005	0.869±0.012	0.872±0.011	0.873±0.007	0.873±0.002	**0.877**±0.005	0.868±0.008
Human OCR Ensembl	0.698±0.013	0.783±0.007	**0.828**±0.006	0.818±0.006	0.793±0.004	0.822±0.003	0.824±0.001
Human NonTATA Promoters	0.861±0.009	0.944±0.002	0.946±0.007	0.945±0.010	0.951±0.003	**0.957**±0.004	0.954±0.010

We start with the Genomic Benchmark, which is a collection of 8 regulatory element classification tasks with sequence lengths mostly ranging from 200 to 500, and one up to 4,776. We take the hidden

state embedding of the final layer and apply a pooling layer on sequences to obtain a fixed-length representation. We then apply a linear layer to map the representation to the number of classes for each task. We perform 5-fold cross-validation for each task using the same seed as [8]. As shown in Table 1, our model achieves state-of-the-art performance on 3 out of 8 tasks, outperforming the previous best model, while the remaining tasks are close to the best model. Please note that this benchmark is already quite saturated, as shown in Table 1, and we do not expect improvements in pretraining to meaningfully improve benchmark performance further.

Table 2: Nucleotide Transformer Tasks. Performance (↑) across 10-fold CV for Enformer, DNABERT-2, Nucleotide Transformer v2, HyenaDNA, Caduceus-PH, ConvNova, and $\text{JanusDNA}_{\text{mlp}}$. Metrics vary by task: MCC for histone markers and enhancer annotation, F1-score for promoter annotation and splice site acceptor/donor, and accuracy for splice site "all". Best values per task are **bolded**, second best are *italicized*. Given the disparity in model size, we also underline the best value among models with fewer than 2M activated parameters. Error bars indicate the difference between the maximum and minimum values across 10 random seeds used for CV.

	> 100M activated Param. Models			< 2M activated Param. Models				
	Enformer (252M)	DNABERT-2 (117M)	NT-v2 (500M)	HyenaDNA (1.6M)	Caduceus-Ph (1.9M)	ConvNova (1.7M)	JanusDNA mlp w/ midattn (2.001M)	JanusDNA mlp w/o midattn (2.009M)
Histone Markers								
H3	0.719±0.048	0.785±0.033	0.784±0.047	0.779±0.037	0.815±0.048	0.812±0.017	**0.835**±0.009	0.831±0.023
H3k14ac	0.288±0.077	0.516±0.028	0.551±0.021	0.612±0.065	0.631±0.026	0.644±0.009	**0.729**±0.022	0.718±0.026
H3k36me3	0.344±0.055	0.591±0.020	0.625±0.013	0.613±0.041	0.601±0.129	0.661±0.019	**0.702**±0.015	0.699±0.025
H3k4me1	0.291±0.061	0.511±0.028	0.550±0.021	0.512±0.024	0.523±0.039	0.554±0.023	0.615±0.035	**0.616**±0.018
H3k4me2	0.211±0.069	0.336±0.040	0.319±0.045	0.455±0.095	0.487±0.170	0.485±0.032	**0.589**±0.023	0.586±0.019
H3k4me3	0.158±0.072	0.352±0.077	0.410±0.033	0.549±0.056	0.544±0.045	0.566±0.027	**0.688**±0.026	0.675±0.014
H3k79me3	0.496±0.042	0.613±0.030	0.626±0.026	0.672±0.048	0.697±0.077	0.700±0.007	**0.747**±0.013	0.743±0.009
H3K9ac	0.420±0.063	0.542±0.029	0.562±0.040	0.581±0.061	0.622±0.030	0.658±0.011	**0.673**±0.014	0.661±0.027
H4	0.732±0.076	0.796±0.027	0.799±0.025	0.763±0.044	0.811±0.022	0.808±0.008	0.812±0.011	**0.813**±0.013
H4ac	0.273±0.063	0.463±0.041	0.495±0.032	0.564±0.038	0.621±0.054	0.636±0.011	0.698±0.013	**0.705**±0.023
Regulatory Annotation								
Enhancer	0.451±0.108	0.516±0.098	0.548±0.144	0.517±0.117	0.546±0.073	**0.586**±0.038	0.559±0.042	0.542±0.044
Enhancer types	0.309±0.134	0.423±0.051	0.424±0.132	0.386±0.185	0.439±0.054	0.500±0.018	**0.503**±0.038	0.492±0.096
Promoter: All	0.954±0.006	0.971±0.006	**0.976**±0.006	0.960±0.005	0.970±0.004	0.967±0.001	0.970±0.002	0.970±0.003
NonTATA	0.955±0.010	0.972±0.005	**0.976**±0.005	0.959±0.008	0.969±0.011	0.968±0.003	0.971±0.004	0.971±0.003
TATA	0.960±0.023	0.955±0.021	0.966±0.013	0.944±0.040	0.953±0.016	**0.969**±0.003	0.958±0.007	0.960±0.008
Splice Site Annotation								
All	0.848±0.019	0.939±0.009	**0.983**±0.008	0.956±0.011	0.940±0.027	0.965±0.004	0.967±0.005	0.943±0.020
Acceptor	0.914±0.028	0.975±0.006	**0.981**±0.011	0.958±0.010	0.937±0.033	0.971±0.003	0.957±0.012	0.961±0.009
Donor	0.906±0.027	0.963±0.006	**0.985**±0.022	0.949±0.024	0.948±0.025	0.965±0.003	0.948±0.008	0.935±0.016

4.2.2 Nucleotide Transformer Tasks

Next, we evaluate our model on the Nucleotide Transformer tasks, which include 18 datasets covering histone marker prediction, regulatory annotation prediction, and splice site annotation prediction. Following the evaluation metrics outlined in [11], we perform 10-fold cross-validation for each task, adhering to the same experimental settings as [8].

As shown in Table 2, our model achieves state-of-the-art performance on 12 out of 18 tasks, surpassing previous models, including those with 250 times more activated parameters. While the promoter and splice site annotation tasks exhibit slightly weaker performance compared to the best larger model, this underscores the potential importance of training data scale and diversity for these specific tasks. For clarity, we present only the results of Caduceus-PH in the table due to space constraints, as Caduceus-PS performs slightly worse than Caduceus-PH.

4.2.3 DNA Long Range Benchmark

To further assess our model's ability to capture long-range dependencies in DNA sequences, we evaluate it on the expression Quantitative Trait Loci (eQTL) prediction task from DNALONGBENCH [38] with the sequence length of 450,000. The eQTL task measures whether a nucleotide variant can influence the expression of a target gene based on the sequences of the gene and its surrounding regions.

Table 3: DNALongBench eQTL Tasks. The AUROC for expert model - Enformer, Caduceus-PH, and JanusDNA. The best results are **bolded**.

Models activated param	expert model (252M)	Caduceus-Ph (7.7M)	JanusDNA w/o mid-Attn (7.662M)	JanusDNA mlp w/o mid-Attn (7.745M)
Artery Tibial	0.741	0.690	0.803	**0.852**
Adipose Subcutaneous	0.736	0.759	0.741	**0.769**
Cells Cultured fibroblasts	0.639	0.690	0.771	**0.802**
Muscle Skeletal	0.621	0.789	0.803	**0.864**
Nerve Tibial	0.683	0.842	0.877	**0.914**
Skin Not Sun Exposed Suprapubic	0.710	0.812	0.875	**0.903**
Skin Sun Exposed Lower leg	0.700	0.692	0.706	**0.846**
Thyroid	0.612	0.703	0.752	**0.793**
Whole Blood	0.689	0.769	0.794	**0.821**

Due to limited computational resources, we compare our model against the state-of-the-art DNA language model, Caduceus-PH [8], which is also trained with the same data scale for fair comparison, and the expert model for this task, Enformer [39]. The results for Enformer are taken directly from DNALONGBENCH [38]. For Caduceus-PH, we entirely fine-tune the official HuggingFace-released weights, which are pretrained on sequences of length 131k. Our model is pretrained and entirely fine-tuned using the same setup and sequence length as Caduceus-PH to ensure a fair comparison. As shown in Table 3, JanusDNA achieves the best performance on 8 out of 9 datasets, significantly outperforming the expert model and Caduceus-PH despite using fewer computational parameters.

5 Conclusion

Summary In this work, we introduced a novel global modeling paradigm for bidirectional DNA sequence representation, combining the bidirectional capability of masked language modeling with the speed and optimization benefits of autoregressive approaches. We proposed JanusDNA, a Mamba MoE-based DNA foundation model with global attention that enhances genomic sequence understanding while maintaining low memory complexity, supporting the processing of up to 1 million base pairs (1 Mbp) on a single 80GB GPU. Experimental results demonstrate that JanusDNA outperforms HyenaDNA, Caduceus, and other Transformer-based models across a range of benchmark tasks and the expert model on eQTL tasks. By leveraging global attention mechanisms and efficient long-range sequence processing, JanusDNA offers a powerful framework for advancing research on long-range genomic interactions.

Limitations and Future Work Although JanusDNA demonstrates high learning efficiency on a fixed data scale, current training is restricted to the human reference genome for fair architectural comparison. Expanding the corpus to include human genomic variants (e.g., 1000 Genomes Project) and non-human species (e.g., primates) could further boost modeling capacity and biological insight. JanusDNA also lacks integration of multimodal data such as epigenetic states (e.g., chromatin accessibility, histone modifications) and single-cell transcriptomic profiles, which are vital for resolving cell-type-specific regulation and predicting chromatin-influenced phenotypes. Future work will incorporate these modalities and explore functional roles of key genomic features (e.g., CTCF-mediated chromatin loops, enhancer RNAs, non-coding risk variants). Experimental validation (e.g., CRISPR, organoids) will prioritize therapeutic targets, while clinical collaborations will evaluate JanusDNA's utility in personalized diagnostics and drug discovery. These efforts aim to evolve JanusDNA into a unified framework linking genome structure, epigenetic regulation, and disease mechanisms.

6 Acknowledgments

The authors acknowledge the Scientific Computing of the IT Division at the Charité - Universitätsmedizin Berlin and the Science Computing at Shenzhen Technology University for providing computational resources that have contributed to the research results reported in this paper. B.W. and R.E. acknowledge support by the Collaborative Research Center (SFB 1470) funded by the German Research Council (DFG).

References

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

[2] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[3] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[4] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[5] Yuxiao Hu, Qian Li, Dongxiao Zhang, Jinyue Yan, and Yuntian Chen. Context-alignment: Activating and enhancing llm capabilities in time series. *arXiv preprint arXiv:2501.03747*, 2025.

[6] Yanhao Jia, Ji Xie, S Jivaganesh, Hao Li, Xu Wu, and Mengmi Zhang. Seeing sound, hearing sight: Uncovering modality bias and conflict of ai models in sound localization. *arXiv preprint arXiv:2505.11217*, 2025.

[7] Wei Liu, Jun Li, Yitao Tang, Yining Zhao, Chaozhong Liu, Meiyi Song, Zhenlin Ju, Shwetha V Kumar, Yiling Lu, Rehan Akbani, et al. Drbioright 2.0: an llm-powered bioinformatics chatbot for large-scale cancer functional proteomics analysis. *Nature communications*, 16(1):2256, 2025.

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

[9] Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V Davuluri. Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *Bioinformatics*, 37(15):2112–2120, 2021.

[10] 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.

[11] 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*, 22(2):287–297, 2025.

[12] Sarah Nemsick and Anders S Hansen. Molecular models of bidirectional promoter regulation. *Current opinion in structural biology*, 87:102865, 2024.

[13] Samy Jelassi, David Brandfonbrener, Sham M Kakade, and Eran Malach. Repeat after me: Transformers are better than state space models at copying. *arXiv preprint arXiv:2402.01032*, 2024.

[14] Georgios Pantazopoulos, Malvina Nikandrou, Alessandro Suglia, Oliver Lemon, and Arash Eshghi. Shaking up vlms: Comparing transformers and structured state space models for vision & language modeling. *arXiv preprint arXiv:2409.05395*, 2024.

[15] 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*, 36:43177–43201, 2023.

[16] Daniel Schulz, Bjoern Schwalb, Anja Kiesel, Carlo Baejen, Phillipp Torkler, Julien Gagneur, Johannes Soeding, and Patrick Cramer. Transcriptome surveillance by selective termination of noncoding rna synthesis. *Cell*, 155(5):1075–1087, 2013.

[17] Wu Wei, Vicent Pelechano, Aino I Järvelin, and Lars M Steinmetz. Functional consequences of bidirectional promoters. *Trends in Genetics*, 27(7):267–276, 2011.

[18] Olga Mikhaylichenko, Vladyslav Bondarenko, Dermot Harnett, Ignacio E Schor, Matilda Males, Rebecca R Viales, and Eileen EM Furlong. The degree of enhancer or promoter activity is reflected by the levels and directionality of erna transcription. *Genes & development*, 32(1):42–57, 2018.

[19] Eric Nguyen, Michael Poli, Matthew G Durrant, Brian Kang, Dhruva Katrekar, David B Li, Liam J Bartie, Armin W Thomas, Samuel H King, Garyk Brixi, et al. Sequence modeling and design from molecular to genome scale with evo. *Science*, 386(6723):eado9336, 2024.

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

[21] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[22] Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. Llm2vec: Large language models are secretly powerful text encoders. *arXiv preprint arXiv:2404.05961*, 2024.

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

[24] Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, et al. Jamba: A hybrid transformer-mamba language model. *arXiv preprint arXiv:2403.19887*, 2024.

[25] Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.

[26] PyTorch Team. Flexattention: A new way to simplify and enhance attention mechanisms. `https://pytorch.org/blog/flexattention/`, 2024. Accessed: 2025-05-07.

[27] Haotian Cui, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang. scgpt: toward building a foundation model for single-cell multi-omics using generative ai. *Nature Methods*, 21(8):1470–1480, 2024.

[28] Xingyi Cheng, Bo Chen, Pan Li, Jing Gong, Jie Tang, and Le Song. Training compute-optimal protein language models. biorxiv. 2024.

[29] Micaela E Consens, Cameron Dufault, Michael Wainberg, Duncan Forster, Mehran Karimzadeh, Hani Goodarzi, Fabian J Theis, Alan Moses, and Bo Wang. To transformers and beyond: large language models for the genome. *arXiv preprint arXiv:2311.07621*, 2023.

[30] Christina V Theodoris, Ling Xiao, Anant Chopra, Mark D Chaffin, Zeina R Al Sayed, Matthew C Hill, Helene Mantineo, Elizabeth M Brydon, Zexian Zeng, X Shirley Liu, et al. Transfer learning enables predictions in network biology. *Nature*, 618(7965):616–624, 2023.

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

[32] Genome Reference Consortium et al. Genome reference consortium human build 37 (grch37). *Database (GenBank or RefSeq)*, 2009.

[33] Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. A survey on mixture of experts in large language models. *IEEE Transactions on Knowledge and Data Engineering*, page 1–20, 2025.

[34] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.*, 23(1), January 2022.

[35] Hannah Zhou, Avanti Shrikumar, and Anshul Kundaje. Towards a better understanding of reverse-complement equivariance for deep learning models in genomics. In *Machine Learning in Computational Biology*, pages 1–33. PMLR, 2022.

[36] Babak Alipanahi, Andrew Delong, Matthew T Weirauch, and Brendan J Frey. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 33(8):831–838, 2015.

[37] 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.

[38] 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.

[39] 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.

[40] 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.

[41] Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning*, pages 28043–28078. PMLR, 2023.

[42] Badri Narayana Patro and Vijay Srinivas Agneeswaran. Mamba-360: Survey of state space models as transformer alternative for long sequence modelling: Methods, applications, and challenges. *arXiv preprint arXiv:2404.16112*, 2024.

[43] Mingqian Ma, Guoqing Liu, Chuan Cao, Pan Deng, Tri Dao, Albert Gu, Peiran Jin, Zhao Yang, Yingce Xia, Renqian Luo, et al. Hybridna: A hybrid transformer-mamba2 long-range dna language model. *arXiv preprint arXiv:2502.10807*, 2025.

[44] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

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

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: Our main claims are summarized in Section 3. Also see Section 4 and Appendix B.2 for more experimental evidence.

   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: We include the limitations of our work in Section 5.

   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: We detail the assumption and proof of theoretical result on training efficiency in 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 use publicly-accessible human reference genome [32] and benchmarks [37, 11, 40]. We explain our setting in Section 4 and Appendix B.2. We upload the codes and instructions to recover the results.

   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 use publicly-accessible dataset HG38 [32]. We upload the codes and instructions to recover the results.

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: Details are summarized in Section 4 and Appendix B.2.

   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 show the standard error in most training curves. We calculate standard error across 10 random seeds for nucleotide transform tasks, and 5 random seeds for genomic benchmark in Section 4. We use 3 random seeds for ablation in Appendix B.1.1. But no error bars for DNALongBench [40] due to limited computational resources.

   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 have the Appendix B.3 on this.

   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 have read and understood the code of ethics and have done our best to conform.

   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: [No]

    Justification: This work focuses on biological information understanding for the academic use. This work is not related to any private or personal data, and there are no explicit negative social impacts.

    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: [No]

    Justification: We do not foresee any high risk for misuse of this work.

    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: Yes, we credited them in appropriate ways.

    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 will release our code base with included readme files.

    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: The paper does not involve crowdsourcing nor research with human subjects.

    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: [No]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    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 did not involve LLMs as any important, original or non-standard components.

    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.

A Related Work

A.1 DNA Language Models

Attention-based Models Attention-based DNA language models, such as DNABERT [9], DNABERT2 [10], and Nucleotide Transformer [11], have demonstrated significant success in modeling DNA sequences. These models employ k-mer encoders to group consecutive nucleotide base pairs into single tokens, enabling efficient sequence representation. However, their global attention mechanisms restrict scalability to sequences of approximately 12,000 base pairs (bps), limiting their ability to capture long-range dependencies. Furthermore, the use of k-mer tokenizers reduces modeling resolution, posing challenges for tasks like Single Nucleotide Polymorphism (SNP) analysis.

State Space Models State Space Models (SSMs) have been applied to genomic tasks, with HyenaDNA [15] utilizing the Hyena operator [41] to process sequences up to 1 million base pairs (bps). Despite this capability, its unidirectional design limits the model's ability to capture bidirectional genomic contexts [29]. To overcome this, Caduceus [8] introduces a bidirectional Mamba architecture, which aggregates information from both upstream and downstream sequences, enhancing genomic context comprehension. However, SSM-based models often face challenges in memory recall tasks when compared to transformer-based approaches [42]. Additionally, their masking-based training paradigm is less efficient, as it uses only 15% of the data for loss computation per iteration. In contrast, autoregressive training paradigms take advantage of nearly 99% of the data, significantly improving training efficiency and overall performance. Latest SSM-based DNA models, such as Evo [19] and Evo2 [20], introduce stripedHyena and StripedHyena2, showing better scaling rate and improved throughput compared to HyenaDNA. However, to gain better generative performance, they still rely on the autoregressive training paradigm, facing the same limitation as HyenaDNA for bidirectional understanding.

Hybrid Models Hybrid models incorporate multiple encoding mechanisms such as convolutional neural networks (CNNs), Mamba, and Attention to leverage the complementary strengths of each architecture. Enformer [39] combines CNNs with Transformers, enabling the model to capture long-range genomic interactions spanning up to 100 kilobases. HybriDNA [43] integrates Mamba and Transformer components, extending its receptive field to 131 kilobases.

B Experimental Details

B.1 Ablation Experiments

B.1.1 JanusDNA Hybrid Architecture

To determine the optimal hybrid architecture for DNA sequence modeling that effectively balances local and global attention, we perform ablation experiments on various configurations of the unidirectional encoder (i.e., modules preceding the bidirectional fusion layer). Referring to [24], we also explore the value of the additional mid-attention layer. Specifically, we evaluate the following configurations: 1) mamba and FFN blocks only, 2) mamba and FFN blocks with mid-attention, 3) mamba and FFN blocks with MoE, and 4) mamba and FFN blocks with both mid-attention and MoE. The ratio of MoE to replace FFNs is set to 0.5. In models with mid-attention, the two independent bidirectional Mamba blocks at the 4th layer are replaced with two independent causal Attention blocks implemented using FlashAttention2 [44].

The models are pre-trained on sequences of lengths 1024 and 131072, with batch sizes of 128 and 1, respectively, using a single GPU. All other hyperparameters and training settings are consistent with the pre-training setup described earlier. To ensure a consistent number of activated parameters across different models, we adjust the model configurations as Table 4. The training perplexity results for these configurations are shown in Figure 5.

The training perplexity results reveal that the model with mamba, FFN, and mid-attention exhibits higher perplexity compared to the model with only mamba and FFN, while the model with mamba, FFN, and MoE achieves lower perplexity. This suggests that mid-attention may not enhance training efficiency, whereas MoE contributes positively. Notably, the model combining mamba, FFN, mid-

Table 4: Hyperparameter settings for JanusDNA ablation experiments.

	JanusDNA (all with Mamba)			
	FFN	MidAttn+FFN	MoE+FFN	MidAttn+MoE+FFN
Layers	8	8	8	8
Width	148	148	128	128
activated Params (M)	5.973	5.973	6.084	6.080
total Params (M)	5.973	5.973	28.104	28.100
global steps	10k	10k	10k	10k
Expert number of MoE	0	0	16	16
head number of Attention	4	4	4	4
multiple number of FFN width	4	4	4	4
Optimizer	AdamW			
Optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.95$			
Learning rate	$8e^{-3}$			
LR Scheduler	Cosine decay			
Weight decay (model)	0.1			

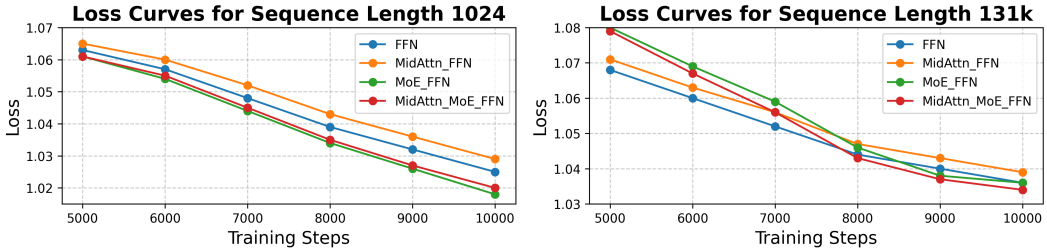

Figure 5: Training perplexity of Mid-attention and MoE ablation models on 1024-length and 131k-length sequences.

attention, and MoE achieves the lowest perplexity, indicating that mid-attention and MoE can synergistically improve training performance.

To further investigate the role of mid-attention, we conduct additional ablation experiments comparing models with and without mid-attention (all incorporating MoE due to its demonstrated benefits) on the Nucleotide Transformer Benchmark with 3-fold cross-validation since Genomic Benchmark is too saturated to show apparent differences. We pretrain and fine-tune models with hidden dimensions of 32, 72, and 128. The results, summarized in Figure

6, indicate that models with mid-attention perform better at a hidden dimension of 32. However, as the hidden dimension increases, models without mid-attention outperform those with mid-attention at a dimension of 72. At a dimension of 128, the performance of models with and without mid-attention becomes comparable. These findings suggest that mid-attention provides diminishing benefits as the hidden dimension grows larger, eventually becoming negligible at higher dimensions.

Given that larger hidden dimensions are generally preferred for large-scale DNA sequence modeling, and considering that mid-attention introduces additional computational overhead, we prefer to use models with mamba and FFN blocks without mid-attention for downstream tasks. Nonetheless, we include the experiments of the model with mid-attention in our formal evaluations on Genomic Benchmark and Nucleotide Transformer Benchmark to ensure completeness and provide a comprehensive analysis.

B.1.2 Reverse Complement (RC)

DNA follows the complementary base-pairing principle, meaning each DNA strand has a reverse complement strand with equivalent genetic information. However, despite this theoretical equivalence, many biologically important motifs (e.g., transcription factor binding sites) are non-palindromic. Recognizing both the forward and RC versions of such motifs (for instance, motifs like GATA and TATC) is challenging, as it effectively requires the model to learn two distinct representations. Therefore, explicitly integrating RC information allows the model to more robustly and comprehensively capture DNA sequence patterns. Nonetheless, the utility of RC depends heavily on the specific biological

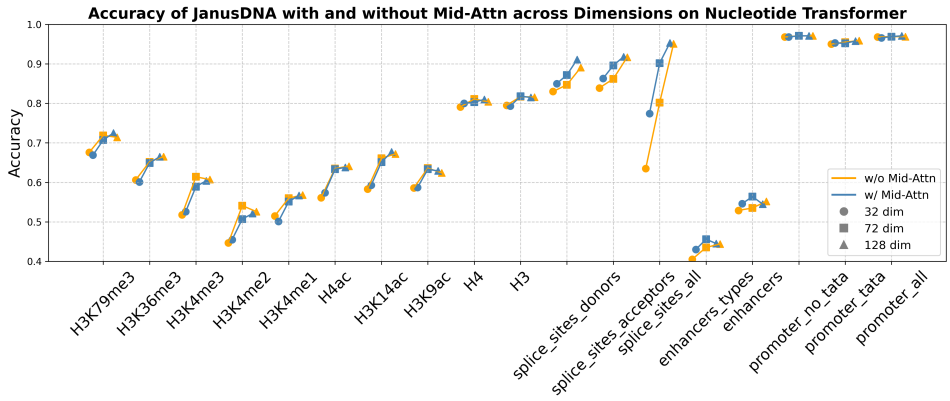

Figure 6: Mid attention ablation results on Nucleotide Transformer.

context. For instance, genomic elements such as splice sites are strictly defined by a single DNA strand, making RC inclusion potentially suboptimal or misleading.

Regarding computational overhead, RC is introduced only during inference by averaging the embedding representations of a strand and its RC prior to decoding. Thus, the additional computational cost is minimal, primarily involving an extra forward pass through the model for the RC strand.

To quantitatively assess the impact of incorporating RC, we conducted ablation experiments using the NT benchmark. We fine-tuned JanusDNA models under consistent experimental conditions (learning rate 1e-3, batch size 256), with and without RC during prediction. The results clearly indicate that models utilizing RC generally outperform those without RC across most tasks. Notably, the exception to this pattern was observed in splice site prediction tasks, where RC inclusion led to inferior performance, consistent with the biological reality that splice sites are inherently strand-specific.

Table 5: Performance of JanusDNA with and without middle attention, with and without Reverse Complement on Nucleotide Transformer Benchmark. Top-1 accuracy (↑) across 5-fold cross-validation (CV) for different model variants. Best values per task within each group (left two columns are JanusDNA without middle attention, right two columns are JanusDNA with middle attention) are **bolded**. Error bars indicate the standard deviation across 5 random seeds used for CV.

Tasks	JanusDNA			
	w/ mid-attn		w/o mid-attn	
	w/o RC	w/ RC	w/o RC	w/ RC
H3	0.789±0.028	**0.828**±0.020	0.795±0.026	**0.830**±0.015
H3K14ac	0.689±0.029	**0.729**±0.022	0.662±0.015	**0.700**±0.015
H3K36me3	0.661±0.021	**0.701**±0.022	0.658±0.016	**0.688**±0.012
H3K4me1	0.574±0.025	**0.609**±0.022	0.555±0.030	**0.605**±0.028
H3K4me2	0.546±0.026	**0.588**±0.020	0.532±0.020	**0.581**±0.024
H3K4me3	0.640±0.013	**0.681**±0.016	0.625±0.015	**0.675**±0.014
H3K79me3	0.723±0.025	**0.747**±0.013	0.710±0.020	**0.743**±0.009
H3K9ac	0.638±0.023	**0.673**±0.014	0.631±0.016	**0.658**±0.020
H4	0.781±0.020	**0.810**±0.022	0.775±0.019	**0.813**±0.011
H4ac	0.653±0.023	**0.696**±0.019	0.629±0.017	**0.684**±0.020
enhancers	0.382±0.035	**0.396**±0.033	0.379±0.041	**0.397**±0.065
EnhancersTypes	0.475±0.053	**0.488**±0.066	0.490±0.046	**0.492**±0.096
PromoterAll	0.964±0.003	**0.969**±0.002	0.962±0.003	**0.970**±0.002
PromoterNoTata	0.961±0.004	**0.969**±0.004	0.961±0.004	**0.970**±0.005
PromoterTata	0.946±0.012	**0.954**±0.010	0.947±0.010	**0.953**±0.019
SpliceSitesAll	**0.961**±0.003	0.948±0.008	**0.946**±0.007	0.922±0.019
SpliceSitesAcceptors	**0.928**±0.010	0.906±0.016	**0.932**±0.014	0.904±0.008
SpliceSitesDonors	**0.915**±0.008	0.893±0.006	**0.921**±0.009	0.874±0.011

B.1.3 Janus and Casual Modeling

We conducted an ablation comparing Masked and Janus modeling using the same architecture as in Figure 4. In order to further demonstrate the value of bidirectional modeling, we here conduct the ablation experiments between Janus modeling and Casual modeling.

Table 6: Comparasion of Janus models and Casual models on Last-Token prediction.

	Janus			Casual		
size(M)	0.056	0.207	0.795	0.061	0.209	0.814
step(k)\dim	32	64	128	40	76	152
1	0.428	0.439	0.445	0.421	0.426	0.434
2	0.445	0.451	0.460	0.454	0.441	0.453
3	0.450	0.462	0.472	0.443	0.447	0.471
4	0.451	0.466	0.473	0.443	0.451	0.468
5	0.453	0.470	0.479	0.454	0.457	0.478
6	0.455	0.473	0.482	0.451	0.453	0.481
7	0.458	0.474	0.484	0.449	0.457	0.482
8	0.459	0.477	0.488	0.459	0.465	0.479
9	0.461	0.479	0.490	0.453	0.467	0.479
10	0.461	0.480	0.491	0.459	0.465	0.484

Table 7: Comparasion of Janus models and Casual models on Middle-Token prediction.

	Janus			Casual		
size(M)	0.056	0.207	0.795	0.061	0.209	0.814
step(k)\dim	32	64	128	40	76	152
1	0.428	0.436	0.440	0.384	0.400	0.408
2	0.438	0.454	0.465	0.429	0.425	0.443
3	0.437	0.449	0.467	0.436	0.446	0.454
4	0.443	0.465	0.471	0.434	0.451	0.454
5	0.444	0.477	0.477	0.434	0.449	0.451
6	0.456	0.474	0.482	0.428	0.442	0.446
7	0.458	0.471	0.477	0.438	0.457	0.461
8	0.453	0.478	0.478	0.434	0.455	0.461
9	0.458	0.481	0.485	0.447	0.458	0.465
10	0.456	0.482	0.488	0.443	0.456	0.465

Conducting a direct ablation with same architecture between Casual modeling and Janus modeling is challenging due to inherent architectural differences. The core strength of Janus modeling lies in its bidirectional context understanding, requiring a bidirectional architecture. Casual modeling, by definition, is strictly unidirectional, and adapting it to a bidirectional architecture would conflict with its fundamental principles.

However, we can still maintaint the most faireness by keeping models with same parameters. We constructed Casual models with one layer of a single mamba encoder and a FlashAttention2 layer with a causal attention mask, ensuring comparable model sizes to the Janus models. Following same pre-training settings, we compared the two kinds of models in last-token prediction and middle-token prediction. we expected similar performance for the prediction of the last token (where there is no bidirectional information, so both models have exactly the same information to predict the last token), whereas we expected Janus models to outperform Casual models for the prediction of a token in the center of the sequence (where Janus benefits from the bidirectional context).

The results are presented in table 6 and table 7. We confirm that the Janus model consistently outperforms the Casual models for tokens in the center of the sequence across all evaluated model sizes. To our surprise, Janus models also slightly outperform the Casual models even on the last token prediction task, suggesting Janus models indeed learn richer DNA representations through bidirectional training.

B.1.4 Multilayer Perceptron as feature enhancer

The features fused by attention can be further enhanced through the addition of a Multi-Layer Perceptron (MLP). To verify this, we conduct ablation experiments and name the JanusDNA models equipped with an MLP as JanusDNA-MLP, distinguishing them from the original JanusDNA models without the MLP attached after the feature fusion attention module. We perform the ablation

experiments on both the Nucleotide Transformer Benchmark and the DNALONGBENCH Benchmark, as shown in Table 8 and Table 3. The inclusion of the MLP leads to notable performance improvements across both benchmarks.

Table 8: Performance of JanusDNA with and without MLP on Nucleotide Transformer Tasks. Performance (↑) across 10-fold CV for janusDNA and JanusDNA mlp variants. Metrics vary by task: MCC for histone markers and enhancer annotation, F1-score for promoter annotation and splice site acceptor/donor, and accuracy for splice site "all". Best values per task are **bolded**, second best are *italicized*. Since all models are approximately fewer than 2M activated parameters, we underline the best value(s) among them. Error bars indicate the difference between the maximum and minimum values across 10 random seeds used for CV.

	JANUSDNA W/ MID-ATTN (1.980M)	JANUSDNA W/O MIDATTN (1.988M)	JANUSDNA MLP W/ MIDATTN (2.001M)	JANUSDNA MLP W/O MIDATTN (2.009M)
Histone Markers				
H3	0.821±0.021	0.824±0.012	**0.835**±0.009	0.831±0.023
H3K14AC	0.665±0.034	0.685±0.016	**0.729**±0.022	0.718±0.026
H3K36ME3	0.658±0.024	0.670±0.012	**0.702**±0.015	0.699±0.025
H3K4ME1	0.563±0.041	0.571±0.018	0.615±0.035	**0.616**±0.018
H3K4ME2	0.509±0.056	0.548±0.022	**0.589**±0.023	0.586±0.019
H3K4ME3	0.605±0.030	0.629±0.022	**0.688**±0.026	0.675±0.014
H3K79ME3	0.716±0.017	0.727±0.023	**0.747**±0.013	0.743±0.009
H3K9AC	0.641±0.024	0.639±0.019	**0.673**±0.014	0.661±0.027
H4	0.809±0.021	**0.816**±0.008	0.812±0.011	0.813±0.013
H4AC	0.637±0.060	0.653±0.034	0.698±0.013	**0.705**±0.023
Regulatory Annotation				
ENHANCER	**0.564**±0.022	0.535±0.036	0.559±0.042	0.542±0.044
ENHANCER TYPES	0.462±0.049	0.470±0.025	**0.503**±0.038	0.492±0.096
PROMOTER: ALL	0.969±0.002	**0.971**±0.002	0.970±0.002	0.970±0.003
NONTATA	0.971±0.003	0.971±0.002	0.971±0.004	0.971±0.003
TATA	0.956±0.010	0.958±0.008	0.958±0.007	**0.960**±0.008
Splice Site Annotation				
ALL	0.963±0.022	0.960±0.009	**0.967**±0.005	0.943±0.020
ACCEPTOR	0.949±0.020	0.939±0.022	0.957±0.012	**0.961**±0.009
DONOR	0.947±0.015	0.936±0.014	**0.948**±0.008	0.935±0.016

B.2 Formal Experiment

B.2.1 Pre-training

Architecture configuration We utilize Mamba, FFN, MoE blocks as the primary building blocks for unidirectional representation, which are then followed by a bidirectional fusion layer. There are 8 layers in each unidirectional encoder and each layer consists of one Mamba block and one FFN block. The MoE block is to replace the FFN block at a certain ratio, which is set to 0.5 in our experiments. The number of experts is set to 16 and the dimension of FFN is set to 4 times the hidden dimension. The bidirectional fusion layer is achieved by a FlexAttention layer [26] with 4 attention heads.

Meanwhile, we also implement a version of the model with mid-attention, which replaces the Mamba block at the fifth layer with a mid-attention layer. The mid-attention layer is implemented with FlexAttention with 4 attention heads. The attention mask is set to half triangle to allow the model to only attend to the tokens ahead of the current token to keep causality.

Pre-training setup To ensure a fair comparison with prior work, we pre-train our model on the human reference genome (HG38 [32]) following the training setup described in [8]. We use cross-entropy loss for pre-training. The model is trained with a learning rate of 8×10^{-3}, maintaining a constant token count of 2^{20} tokens per batch. Two sequence lengths are used: 1024 and 131072, with corresponding batch sizes of 128 and 1, respectively, across 8 GPUs. Optimization is performed using AdamW [45] with a weight decay of 0.1, $\beta_1 = 0.9$, and $\beta_2 = 0.95$. A cosine learning rate scheduler is applied, incorporating a warmup phase for 10% of the training steps. The learning rate starts at 1×10^{-6} and peaks at 1×10^{-4}. The coefficient for the MoE auxiliary loss is set to 0.2. The gradient clipping threshold is set to 1.0.

For the 1024-length model, training is conducted for 10,000 steps, while the 131072-length model is trained for 50,000 steps.

We pre-trained three scales of the model: 32, 72, and 144 hidden dimensions to keep the same activated parameters for fair comparison with baseline models on different benchmarks. The hidden dimensions of 32 and 72 are used for the 1024-length model, while the 144 hidden dimension is used for the 131072-length model. The feedforward dimension is set to 4 times the hidden dimension, and the number of attention heads is set to 8. The coefficient for the MoE auxiliary loss is set to 0.2, the number of experts is set to 16. All hyperparameter settings are listed as Table 9.

Table 9: Hyperparameter settings for JanusDNA pretraining (select models).

	JanusDNA				
	w/ MidAttn		w/o midattn		
Layers	8	8	8	8	8
Width	32	72	32	72	144
activated Params (M)	0.422	1.980	0.428	1.989	7.664
total Params (M)	1.798	8.947	1.804	8.956	35.533
Max seq. len.	1024	1024	1024	1024	131072
Batch size	1024	1024	1024	1024	8
global steps	10k	10k	10k	10k	50k
Expert number of MoE	16	16	16	16	16
head number of Attention	4	4	4	4	4
multiple number of FFN width	4	4	4	4	4
coefficient of Auxiliary MoE loss	0.2	0.2	0.2	0.2	0.2
runtime(H800 with evaluation every 2k steps)	3h3m	3h7m	3h2m	3h8m	9h17m
Optimizer	AdamW				
Optimizer momentum	β_1, β_2 = 0.9, 0.95				
Learning rate	$8e^{-3}$				
LR Scheduler	Cosine decay				
Weight decay (model)	0.1				

B.2.2 Benchmarks

Genomic Benchmark For the Genomic Benchmark tasks, we follow the experimental setup of [8] and report their results for comparison. To ensure a fair evaluation, we fine-tune 32-dimensional models to match the activated parameter count of the baselines.

We apply 5-fold cross-validation, splitting the training set into 90/10 train/validation splits and using early stopping based on validation performance with seeds of $\{1, 2, 3, 4, 5\}$. Models are fine-tuned for 10 epochs with a batch size of 256.

For learning rate selection, we perform hyperparameter tuning over $1 \times 10^{-3}, 2 \times 10^{-3}$ as [8], and report the best-performing configuration across cross-validation, as summarized in Table 10. We use cross-entropy loss for fine-tuning. For JanusDNA, the coefficient for the MoE auxiliary loss is set to 0.2.

Table 10: JanusDNA models with and without mid-attention hyperparameter selection for learning rate on genomic benchmarks for top-1 accuracy averaged over 5-fold cross-validation.

Dataset	w/ midattn	w/o midattn
Mouse Enhancers	$1e^{-3}$	$2e^{-3}$
Coding vs. Intergenomic	$1e^{-3}$	$1e^{-3}$
Human vs. Worm	$1e^{-3}$	$1e^{-3}$
Human Enhancers Cohn	$1e^{-3}$	$1e^{-3}$
Human Enhancer Ensembl	$1e^{-3}$	$1e^{-3}$
Human Regulatory	$1e^{-3}$	$1e^{-3}$
Human OCR Ensembl	$1e^{-3}$	$2e^{-3}$
Human NonTATA Promoters	$1e^{-3}$	$1e^{-3}$

Nucleotide Transformer Tasks For the Nucleotide Transformer tasks, we adopt the experimental setup from [8] and report their results for comparison. To ensure a fair comparison, we fine-tune 72-dimensional models to match the activated parameter count of the baseline models.

We use 10-fold cross-validation with seeds of $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$, splitting the dataset into 90/10 train/validation subsets and applying early stopping based on validation performance. Each model is fine-tuned for 20 epochs.

We use cross-entropy loss for fine-tuning. We conduct hyperparameter tuning over the following search space, the same as [8]: learning rates in $1 \times 10^{-3}, 2 \times 10^{-3}$ and batch sizes in $128, 256, 512$. For each task, we report the best-performing configuration across cross-validation, as summarized in Table 11. For JanusDNA, the coefficient for the MoE auxiliary loss is set to 0.2.

Table 11: JanusDNA Hyperparameter Selection for Nucleotide Transformer Tasks. Model with Mid-Attention and Model without Mid-Attention fine-tuning hyperparameters chosen based on best performance averaged over 10-fold cross-validation.

		JANUSDNA			
		W/ MIDATTN		W/O MIDATTN	
		LR	BATCH SIZE	LR	BATCH SIZE
HISTONE MARKERS	H3	$1e^{-3}$	256	$2e^{-3}$	128
	H3K14AC	$1e^{-3}$	256	$1e^{-3}$	256
	H3K36ME3	$1e^{-3}$	256	$2e^{-3}$	512
	H3K4ME1	$1e^{-3}$	256	$1e^{-3}$	256
	H3K4ME2	$1e^{-3}$	256	$1e^{-3}$	256
	H3K4ME3	$1e^{-3}$	256	$1e^{-3}$	256
	H3K79ME3	$1e^{-3}$	256	$1e^{-3}$	256
	H3K9AC	$1e^{-3}$	256	$1e^{-3}$	128
	H4	$1e^{-3}$	256	$1e^{-3}$	256
	H4AC	$1e^{-3}$	256	$1e^{-3}$	256
REGULATORY ANNOTATION	ENHANCERS	$1e^{-3}$	512	$1e^{-3}$	512
	ENHANCERS TYPES	$2e^{-3}$	256	$2e^{-3}$	256
	PROMOTER ALL	$1e^{-3}$	512	$1e^{-3}$	128
	PROMOTER NO TATA	$1e^{-3}$	256	$1e^{-3}$	128
	PROMOTER TATA	$2e^{-3}$	256	$1e^{-3}$	128
SPLICE SITE ANNOTATION	SPLICE SITES ACCEPTORS	$2e^{-3}$	128	$2e^{-3}$	128
	SPLICE SITES ALL	$2e^{-3}$	128	$2e^{-3}$	128
	SPLICE SITES DONORS	$2e^{-3}$	128	$2e^{-3}$	128

DNALONGBENCH For the DNALONGBENCH eQTL tasks, we compare JanusDNA with both the expert model Enformer and the state-of-the-art architecture Caduceus-PH. We obtain the pre-trained weights of Caduceus-PH from Hugging Face: `https://huggingface.co/kuleshov-group/caduceus-ph_seqlen-1k_d_model-256_n_layer-4_lr-8e-3`.

For all models, we extract the hidden state embeddings from the final layer and apply a pooling layer to obtain a fixed-length representation for each input sequence. A linear classification head is then used to map these representations to the target number of classes for each cell type.

To ensure comparability, we fine-tune the JanusDNA model with 144-dimensional embeddings, matching the activated parameter count of Caduceus-PH. Fine-tuning is conducted for 3 epochs using a learning rate of 4×10^{-4} and a batch size of 8. We use cross-entropy loss for fine-tuning, and for JanusDNA, the coefficient for the MoE auxiliary loss is set to 0.02. Training is distributed across eight 80GB GPUs, with each GPU processing one batch. All models are fine-tuned and evaluated using float32 precision to ensure stability and fairness in comparison. Due to limited computational resources, we conduct a single run per sub-dataset using the same set of hyperparameters across all experiments.

Transcription Factor Prediction (Mouse) We conducted experiments on non-human Genome Understanding Evaluation (GUE) tasks [10], focusing on transcription factor binding site prediction in mouse genomes to evaluate its generalization capability.

We followed DNABERT2's experimental setup, performing standard fine-tuning with a batch size of 32. Unlike DNABERT2's 1000-epoch fine-tuning at a learning rate of 3e-5, we fine-tuned the JanusDNA model for only 10 epochs at a learning rate of 1e-3.

Results, shown in Table 12, it is very interesting to see that JanusDNA, with fewer active parameters, can perform similarly to DNABERT2 on these tasks, despite being pre-trained only on the human reference genome. In future work, we plan to explore further how more diverse pre-training data affects the model performance.

Table 12: Transcription Factor Prediction (Mouse). Performance (↑) across various models. Metrics are MCC for different categories (0-4). Best values per category are **bolded**. Given the disparity in model size, we also underline the best value among models with fewer than 100M activated parameters.

		TRANSCRIPTION FACTOR PREDICTION (MOUSE)				
MODEL	ACTIVATED PARAMS	0	1	2	3	4
> 100M activated Param. Models						
DNABERT-2 (PRE-TRAINED ON GUE)	117M	**0.642**	**0.863**	0.813	0.735	0.508
DNABERT-2 (NOT PRE-TRAINED ON GUE)	117M	0.568	0.848	0.793	0.665	**0.527**
NT-500M-HUMAN	480M	0.310	0.750	0.617	0.292	0.293
NT-500M-1000G	480M	0.393	0.755	0.647	0.331	0.340
NT-2500M-1000G	2537M	0.483	0.800	0.701	0.423	0.434
NT-2500M-MULTI	2537M	0.633	0.838	0.715	0.694	0.471
< 100M activated Param. Models						
DNABERT (3-MER)	86M	0.423	0.791	0.699	0.554	0.420
DNABERT (4-MER)	86M	0.494	0.800	0.726	0.518	0.441
DNABERT (5-MER)	87M	0.425	0.793	0.622	0.499	0.403
DNABERT (6-MER)	89M	0.444	0.789	0.714	0.449	0.425
JANUSDNA-72DIM	2.009M	0.619	0.850	**0.875**	**0.843**	0.502

B.3 Details of resources used

We use 80GB NVIDIA H100, A100, A800 GPUs for pre-training and fine-tuning.