Gener*ator*: A Long-Context Generative Genomic Foundation Model

Wei Wu[*,1], **Qiuyi Li**[*,†,1], **Mingyang Li**[1], **Kun Fu**[1], **Fuli Feng**[4], **Jieping Ye**[1], **Hui Xiong**[†,2,3], **Zheng Wang**[†,1]

[*]Equal Contribution [†]Equal Senior Authorship

[1]Alibaba Cloud Computing, Beijing, China
[2]Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China
[3]Hong Kong University of Science and Technology , Hong Kong SAR, China
[4]Institute of Dataspace, Hefei, China

[†]Correspondence to: *qiuyi.li1993@gmail.com, xionghui@ust.hk, wz388779@alibaba-inc.com*

ABSTRACT

Advancements in DNA sequencing technologies have significantly improved our ability to decode genomic sequences. However, the prediction and interpretation of these sequences remain challenging due to the intricate nature of genetic material. Large language models (LLMs) have introduced new opportunities for biological sequence analysis. Recent developments in genomic language models have underscored the potential of LLMs in deciphering DNA sequences. Nonetheless, existing models often face limitations in robustness and application scope, primarily due to constraints in model structure and training data scale. To address these limitations, we present **Gener***ator*, a generative genomic foundation model featuring a context length of 98k base pairs (bp) and 1.2B parameters. Trained on an expansive dataset comprising 386B bp of eukaryotic DNA, the **Gener***ator* demonstrates state-of-the-art performance across both established and newly proposed benchmarks. The model adheres to the central dogma of molecular biology, accurately generating protein-coding sequences that translate into proteins structurally analogous to known families. It also shows significant promise in sequence optimization, particularly through the prompt-responsive generation of enhancer sequences with specific activity profiles. These capabilities position the **Gener***ator* as a pivotal tool for genomic research and biotechnological advancement, enhancing our ability to interpret and predict complex biological systems and enabling precise genomic interventions. Implementation details and supplementary resources are available at `https://github.com/GenerTeam/GENERator`.

Keywords Genomics, Genomic Foundation model, Large Language Models, Sequence Design

1 Introduction

Genomic sequences encapsulate vast biological information that drives complex processes, from gene regulation to protein synthesis, influencing phenotypic traits and disease states [51, 42]. Rapid advancements in DNA sequencing technologies [55] have significantly enhanced our ability to decode genomic sequences of various organisms on an unprecedented scale. However, the prediction and interpretation of genomic sequences remain a formidable challenge due to the intricate nature of genetic material and the scarcity of high-quality, task-specific datasets.

Recent progress in machine learning, particularly with large language models (LLMs) [66], has opened new avenues for understanding biological sequences. In natural language processing (NLP), LLMs such as BERT [18] and GPT [40] have shown remarkable success by leveraging large-scale pre-training data to generalize across diverse downstream tasks. This success extends to the realm of biological sequences, as demonstrated by models like AlphaFold [29, 2] and ESM [34, 25], which adeptly unravel the complexities of protein structures and functions. Within genomics, early efforts have primarily focused on masked language models such as DNABERT [28, 69], GROVER [50], Caduceus [52], and Nucleotide Transformer (NT) [13], which excel at understanding DNA semantics but lack generative capabilities and are generally restricted to relatively short sequence lengths.

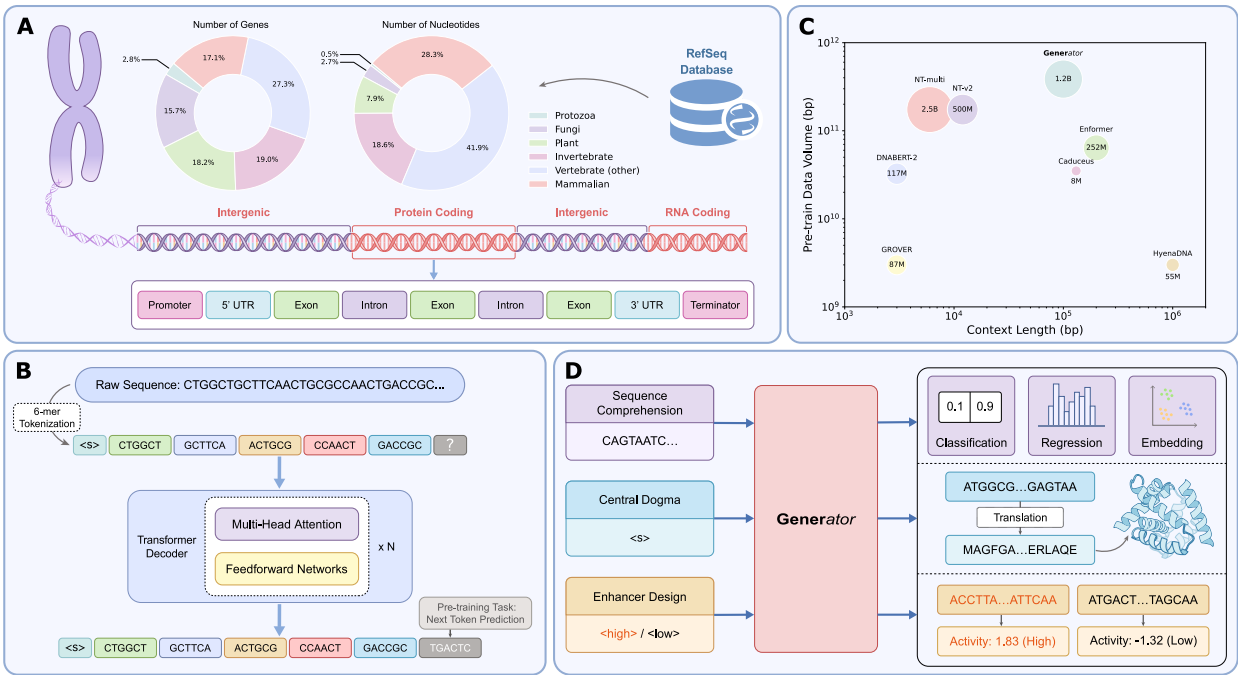

Figure 1: Overview of the **Gener***ator*. (A) The pre-training dataset of the **Gener***ator* encompasses a diverse range of eukaryotic organisms and gene types, totaling 386B nucleotides. (B) The pre-training employs the next token prediction (NTP) task, utilizing a 6-mer tokenizer. (C) Model comparison regarding the context length, pre-train data volume, and model parameters. (D) Downstream applications include sequence comprehension, interpretation of the central dogma by generating DNA sequences that encode functional proteins, and prompt-responsive enhancer design with targeted activity profiles.

Generative models offer the potential to overcome these limitations by combining semantic understanding with the capability to perform sequence design tasks. However, attempts to train large-scale generative DNA language models have been relatively underexplored. Models like megaDNA [54] focus solely on bacteriophage genomes, and HyenaDNA [37] is confined to human genomes—both are limited in scope regarding model parameter size and dataset scale. A notable exception is Evo [36], trained on a comprehensive dataset of bacterial and viral genomes, which demonstrates proficiency in assembling functional CRISPR-Cas molecular complexes. However, there remains a gap for a generative model in the broader eukaryotic domain, known for its biodiversity and complex gene structures.

In this study, we introduce **Gener***ator*, a generative genomic foundation model utilizing the transformer decoder architecture, trained on an expansive dataset comprising 386 billion base pairs (bp) of eukaryotic DNA derived from the RefSeq database [39]. The extensive and diverse pre-training data endow the **Gener***ator* with enhanced understanding and generation capabilities across various organisms. Our evaluations demonstrate that the **Gener***ator* consistently achieves state-of-the-art performance across a wide spectrum of benchmarks, including Genomic Benchmarks [22], NT tasks [13], and our newly proposed Gener tasks. An overview of the **Gener***ator* is provided in *Fig.* 1.

Beyond benchmark performance, the **Gener***ator* adheres to the central dogma of molecular biology [1], accurately generating protein-coding DNA sequences that produce proteins structurally analogous to known families. Moreover, the **Gener***ator* showcases significant promise in sequence optimization, particularly in the design of enhancer sequences that regulate gene expression during various biological stages, highlighting its potential for a series of biologically significant tasks. Our findings position the **Gener***ator* as a vital resource for genomic research and biotechnological advancement. By enhancing our capability to interpret and predict genomic sequences, the **Gener***ator* paves the way for profound improvements in our understanding of complex biological systems and the development of precise genomic interventions.

The primary contributions of this paper are as follows:

1. Introduction of **Gener***ator*, a generative genomic foundation model trained on 386B bp of eukaryotic DNA, featuring a context length of 98k bp and 1.2B parameters, consistently achieving state-of-the-art performance across various genomic benchmarks.

2. Validation of the model's alignment with the central dogma of molecular biology through its ability to generate proteins structurally analogous to known families.
3. Demonstration of the model's capability in enhancer design, highlighting its potential to advance biotechnological applications and genomic research.

2 Related Work

2.1 Large Language Models in Biosciences

Large language models (LLMs) have emerged as powerful tools for natural language comprehension and generation [66]. Beyond their application in traditional natural language tasks, there is a growing interest in leveraging LLMs to accelerate scientific research. Early studies revealed that general-purpose LLMs, owing to their rich pre-training data, exhibit promise across various research domains [4]. Subsequent efforts have focused on directly training LLMs using domain-specific data, aiming to extend the transfer learning paradigm from natural language processing (NLP) to biosciences. This body of work primarily falls into three categories: molecular LLMs, protein LLMs, and genomic LLMs.

For molecular modeling, extensive work has been conducted on training with various molecular string representations, such as SMILES [61, 53, 49], SELFIES [30, 11, 3], and InChI [26]. Additionally, several studies address the modeling of molecular 2D [48] and 3D structures [68] to capture more detailed molecular characteristics. In the realm of protein LLMs, related work [45, 34, 20] mainly concentrates on modeling the primary structure of proteins (amino acid sequences), providing a solid foundation for protein structure prediction [29, 2]. For genomic sequences, numerous studies have attempted to leverage the power of LLMs for improved genomic analysis and understanding. These efforts predominantly involve training models on DNA [6, 28, 5, 13, 69, 50, 21, 52, 64, 54, 37, 36] and RNA [60, 62, 41] sequences. In the following section, we delve deeper into genomic LLMs specifically designed for DNA sequence modeling.

2.2 DNA Language Models

In the early stages, Avsec et al. introduced the BPNet convolutional architecture to learn transcription factor binding patterns and their syntax in a supervised manner. Prior to the emergence of large-scale pre-training, BPNet was widely used in genomics for supervised learning on relatively small datasets. With the advent of BERT [18], DNABERT [28] pioneered the application of pre-training on the human genome using K-mer tokenizers. To effectively capture long-range interactions, Enformer [5] advanced human genome modeling by incorporating convolutional downsampling into transformer architectures.

Following these foundational works, numerous models based on the transformer encoder architecture have emerged. A notable example is the Nucleotide Transformer (NT) [13], which scales model parameters from 100 million to 2.5 billion and includes a diverse set of multispecies genomes. Recent studies, DNABERT-2 [69] and GROVER [50], have investigated optimal tokenizer settings for masked language modeling, concluding that Byte Pair Encoding (BPE) is better suited for masked DNA LLMs. The majority of these models face the limitation of insufficient context length, primarily due to the high computational cost associated with extending the context length in the transformer architecture. To address this limitation, GENA-LM [21] employs sparse attention, and Caduceus [52] uses the more lightweight BiMamba architecture [23], both trained on the human genome.

Although these masked DNA LLMs effectively understand and predict DNA sequences, they lack generative capabilities, and generative DNA LLMs remain in the early stages of development. An early preprint [64] introduced DNAGPT, which learns mammalian genomic structures through three pre-training tasks, including next token prediction. Recent works, such as HyenaDNA [37] and megaDNA [54], achieve longer context lengths by employing the Hyena [43] and multiscale transformer architectures respectively, though they are significantly limited by their data and model scales. A more recent influential study, Evo [36], trained on an extensive dataset of prokaryotic and viral genomes, has garnered widespread attention for its success in designing CRISPR-Cas molecular complexes, thus demonstrating the practical utility of generative DNA LLMs in the genomic field.

3 Method

3.1 Data Preparation

For training the **Gener***ator* model, we sourced raw DNA sequences from all eukaryotic organisms in the RefSeq database. We explored two data processing strategies:

1. **Functional Sequence Training**: Utilizing the rich annotation data available in RefSeq, we isolated the functional regions from genomic sequences. These regions encompass a wide array of functionalities, including transcription into various RNA molecules, translation into complex proteins, and regulatory functions such as promoters and enhancers that control gene expression. Defined broadly as gene regions, these functional DNA segments formed our training samples, totaling 386B nucleotides.
2. **All Sequence Training**: In this approach, we fed a mixture of functional and non-functional DNA sequences from all eukaryotic organisms in RefSeq directly into the language model for training. This dataset includes approximately 2T nucleotides. This strategy aligns with the conventional pre-training paradigm for general LLMs and forms the foundational training approach seen in existing DNA language models.

Generally, while Scheme 2 displays a lower pre-training loss (*Fig.* S5), Scheme 1 consistently outperforms Scheme 2 across a range of downstream tasks (even for tasks where non-functional context dominates, as demonstrated in *Sec.* 4). One possible explanation for this seemingly counterintuitive result is the inherent difference between DNA and human language. DNA is not a 'concise language' but rather filled with 'randomness' and 'redundancy'. If we hypothesize that, at the origin of life, DNA was nearly entirely random, then over billions of years of evolution, processes such as mutation, recombination, and natural selection have randomly given rise to a limited number of biologically functional segments [32, 33]. These specific nucleotide sequences, which have biological functions, represent the 'semantics' of DNA [39]. In contrast, the majority of the unannotated regions, while potentially holding undiscovered genes, largely consist of segments that have not yet acquired function (or semantic meaning). These regions often include highly repetitive and simple segments, such as long stretches of `AAAAAA` or `GCGCGC` [9], contributing to a lower pre-training loss.

Supporting this hypothesis, observed mutation rates in gene regions are significantly lower than in non-gene regions [63]. This difference arises because mutations in gene regions tend to have pronounced effects on biological functions, leading to negative selection pressures, whereas non-gene regions can accommodate more variability. Many genetic editing techniques [12] exploit this property by replacing redundant non-functional regions with meaningful gene sequences to endow new functions. Based on this assumption, incorporating non-functional regions does not merely function as 'increasing data volume' but could be seen as 'contaminating high-quality data'. This insight offers an explanation for the superior performance of Scheme 1, where training is concentrated on semantically rich regions, thereby enhancing model efficacy on downstream tasks.

3.2 Tokenization

In this section, we examine the process of tokenizing DNA sequences into suitable tokens for input into language models. The widely used tokenizers in this field include:

1. **Single Nucleotide Tokenizer**: This tokenizer uses individual nucleotides (`A`, `T`, `C`, `G`) as tokens, providing the finest granularity and serving as the most foundational tokenizer. It is employed by models like Evo, Caduceus, HyenaDNA, and megaDNA.
2. **K-mer Tokenizer**: This widely used tokenizer for DNA sequences treats all possible combinations of K consecutive nucleotides as tokens. Models like DNABERT and NT utilize this approach.
3. **Byte Pair Encoding (BPE) Tokenizer** [31]: Commonly used in NLP, this tokenizer learns a vocabulary by aggregating frequent subwords. Models such as DNABERT-2 and GROVER adopt this method.

Overall, the selection of an appropriate tokenizer involves a trade-off between sequence resolution and contextual coverage. The single nucleotide tokenizer provides the highest resolution, capturing details at the finest level. However, for a fixed number of tokens, this increased granularity results in a reduced context window compared to K-mer or BPE tokenizers. Additionally, the computational cost associated with attention calculation escalates quadratically as sequence length extends. To mitigate this challenge, models like HyenaDNA, Evo, and Caduceus adopt more streamlined state space models [24], such as Hyena [43], StripedHyena [44], and BiMamba [52], to achieve longer context lengths. However, as our experiments demonstrate, they still fall short in achieving equivalent long-text modeling capabilities. These constraints underscore the difficulty in training DNA language models. Unlike protein sequences, which typically consist of hundreds to a few thousand amino acids and can be effectively handled by transformer architectures in protein language models [34], gene segments often extend to tens of thousands of nucleotides, as illustrated in *Fig.* 1A. This emphasizes the importance of selecting an appropriate tokenizer that strikes a balance between granularity and context window size.

In separate investigations, DNABERT-2 [69] and GROVER [50] assessed how tokenizer choice impacts masked language models, both concluding that BPE is the optimal choice, albeit with varying vocabulary sizes tailored to their datasets. Prior to our research, the impact of DNA tokenizer selection for causal language models had not

been thoroughly explored. Although we initially hypothesized that BPE would be most effective in this context, our empirical findings indicate that the K-mer tokenizer significantly outperforms others in the next token prediction (NTP) pre-training. A detailed comparison of tokenizers is provided in *Sec.* 4.1, where the 6-mer model emerges as the most effective among our experiments. This may seem counterintuitive, as it goes against common experiences in NLP [31]. However, upon closer analysis of their pre-training paradigms, it becomes evident that NTP pre-training is not optimally aligned with BPE tokenization for genomic sequences. This misalignment stems from the hierarchical nature of the BPE vocabulary, where words frequently have nested relationships. For instance, with a target next token like `GCCT`, predictions such as `G`, `GC`, or `GCC` would be deemed incorrect, inadvertently complicating the training process.

In contrast, the masked language modeling (MLM) task does not encounter this issue. When a token in the sequence is masked, MLM's learning objective is to fill in the blank with precisely one token from the vocabulary, eliminating the possibility of multiple partially correct answers. This observation further underscores the intrinsic differences between DNA sequences and human language. Unlike human language, which has clear lexical boundaries, DNA sequences lack such explicit delimiters, suggesting that BPE's strong performance in human language may not necessarily translate to DNA sequences. This recognition is one of the critical objectives of our work. While the fundamental approach of LLMs can be adapted for DNA, our aim is to develop a foundation model specifically tailored to the unique characteristics of DNA data.

3.3 Pre-training

In this section, we describe the pre-training phase of the **Gener***ator*, designed to effectively and efficiently handle DNA sequence data using the functional sequence training strategy alongside a 6-mer tokenizer.

In terms of model architecture, we broadly follow the structure of Llama [57, 65], with the detailed configuration shown in Table 1. The pre-training process employs a batch size of 2 million tokens and utilizes the AdamW [35] optimizer, coupled with a cosine learning rate scheduler with a warm-up phase. The entire pre-training spans 6 epochs, processing a total of 386B tokens. To mitigate discrepancies arising from variable tokenization starting points, our K-mer tokenizer introduces a randomized starting position between 0 and 5 for each sample. This method enhances the model's robustness against variations in sequence alignment. To improve the efficiency of long-context pre-training, we utilize Flash Attention [16] and Zero Redundancy Optimizer [46]. More details regarding the pre-training process are provided in the appendix. Overall, our configuration effectively harnesses the potential of transformer architectures while being specifically tailored to the nuances of DNA sequence data, thereby advancing the model's capabilities in genomic modeling.

Table 1: Detailed architecture of the **Gener***ator*.

Parameter	**Value**
Layers	26
Hidden Size	2048
Intermediate Size	5632
Vocabulary Size	4128
Attention Heads	32 (8 KV heads)
Context Length	16,384 (98,304 bp)
Positional Encoding	RoPE [56]
Hidden Activation	SiLU [19]

3.4 Downstream Tasks

In this section, we provide an overview of the evaluation datasets and describe relevant downstream tasks. We draw on two established benchmarks: Genomic Benchmarks [22] and NT tasks [13], and introduce new tasks and datasets.

3.4.1 Sequence Classification

Both Genomic Benchmarks and NT tasks focus on sequence classification. Genomic Benchmarks emphasize human data but also include organisms like mouse, roundworm, and fruit fly. Its datasets are categorized into human regulatory elements (e.g., promoters, enhancers), demo datasets for species or transcript type classification, and dummy datasets for rapid prototyping. In contrast, NT tasks cover a broader species range and involve tasks such as promoter classification, enhancer classification, yeast-based epigenetic marks prediction, and splice site identification across over 100 organisms. Despite the advantages of these benchmarks, they share a significant limitation: the sequences they utilize are predominantly short, typically measuring in the hundreds of nucleotides. This limitation significantly

reduces their relevance to practical applications, where longer sequences are often encountered. To address this issue, we introduce two new sequence classification tasks: gene classification and taxonomic classification.

The gene classification task assesses the model ability to understand short to medium-length sequences, ranging from 100 to 5000 bp. It includes six different gene types and control samples drawn from unannotated regions, with balanced sampling from six distinct eukaryotic taxonomic groups in RefSeq. The classification goal is to predict the gene type. The taxonomic classification task is designed to assess the model's comprehension of longer sequences, which include both gene and predominantly non-gene regions, ranging in length from 10,000 to 100,000 bp. Samples are similarly balanced and sourced from RefSeq across the same six taxonomic groups, with the objective being to predict the taxonomic group of each sample.

3.4.2 Next K-mer Prediction

In addition to sequence classification tasks, we have designed a task to evaluate the generative capabilities of models: next K-mer prediction. Defining metrics to assess generative ability is notably challenging, as the generated sequences often lack a direct reference point, making quantitative evaluation difficult. In NLP, perplexity (PPL) is commonly used to measure generative capability. However, for DNA sequences, there is significant variation in PPL across different tokenizers, precluding direct comparison. Therefore, we resort to next K-mer prediction.

The concept of next K-mer prediction was first introduced by GROVER [50]. It involves inputting a sequence segment into the model and having it predict the next K base pairs. The predicted sequence is then compared to the actual sequence to assess accuracy. In GROVER, this evaluation occurs after fine-tuning the pre-trained model. In this paper, we present next K-mer prediction as a zero-shot task to universally evaluate the generative capabilities of various pre-trained models. For masked language models, we design a specific autoregressive generation strategy: a `<mask>` token is appended to the end of the input sequence at each step, and the prediction for this `<mask>` token is added to the input sequence for the subsequent step.

Importantly, the evaluation dataset is meticulously constructed to ensure that the region subjected to generation is confined to gene regions, as the primary objective of sequence generation is to produce functional DNA sequences. Conversely, we also ensure that the input segment contains a substantial (or even predominant) amount of non-gene content, enabling models utilizing the all sequence training to fully exploit the input context.

3.4.3 Central Dogma

In addition to the aforementioned tasks designed for evaluating and comparing model performance, we have developed two downstream tasks that align more closely with practical applications: central dogma and sequence design. A crucial criterion for assessing the practicality of generative DNA LLMs is their ability to create functional DNA sequences. The central dogma of molecular biology describes the process by which DNA is transcribed and translated into proteins, enabling life functions [1]. We specifically design experiments to test whether the fine-tuned **Gener***ator* model can generate protein-coding DNA sequences that translate into proteins structurally similar to those in a target protein family.

We sourced protein-coding DNA sequences from specific protein families and fine-tuned the **Gener***ator* on these sequences, directing it to generate similar protein-coding DNA sequences. The newly generated DNA sequences are translated into protein sequences using a codon table and compared against the target protein families. To further assess the generated proteins, we examine their conformity to the natural distribution using the perplexity metric from Progen2 [38], and verify their foldability with AlphaFold3 [2].

3.4.4 Sequence Design

As a central and promising field in biology, sequence design plays a crucial role in advancing our understanding and manipulation of biological systems [27, 47]. A notable example is Evo [36], which undertakes the design of CRISPR-Cas molecular complexes using a methodology akin to our 'central dogma' approach. By fine-tuning on target sequences, Evo generates millions of candidate sequences that mirror the distribution of these targets. Suitable samples are then selected through external evaluation.

In contrast, we employ a more efficient and flexible approach, refining supervised fine-tuning to accommodate prompt-responsive sequence generation. In our sequence design task, we utilize enhancer sequences and activity data from the DeepSTARR [17] dataset. We select enhancer sequences from the top and bottom quartiles of activity values. These samples are labeled with prompts `<high>` for high-activity and `<low>` for low-activity enhancers. This refined fine-tuning approach enables the design of enhancer sequences with desired activity profiles, illustrating the model's capability in biologically directed sequence optimization.

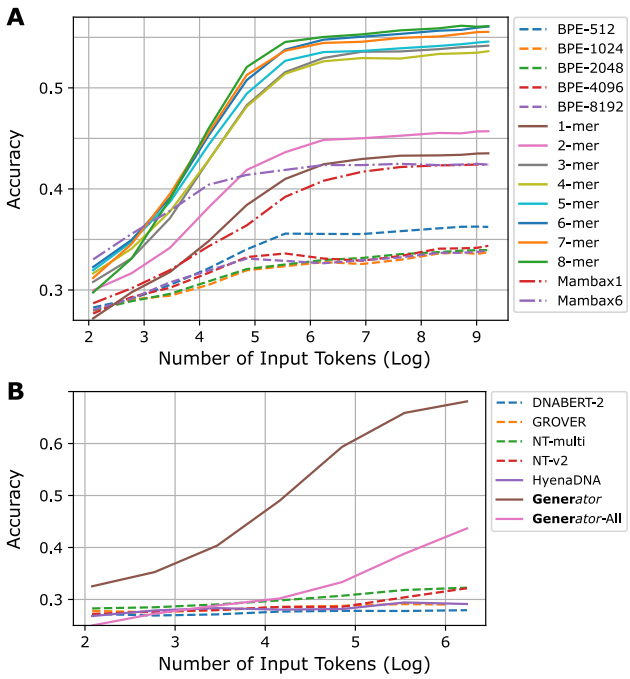

Figure 2: Evaluation of next K-mer prediction with a prediction length of 16 bp. (A) Accuracy of the next K-mer prediction task across various tokenizers and input token lengths. (B) Comparison of the **Gener***ator* against baseline models on a dataset comprised exclusively mammalian DNA.

4 Experiments

4.1 Next K-mer Prediction

As mentioned in *Sec.* 3.2, we conducted extensive experiments to explore the most suitable tokenizer for training causal DNA language models. This was achieved by training multiple models on identical datasets, each employing a different tokenizer. All models share the same architecture as the **Gener***ator* and are uniformly compared at 32,000 training steps. We employed the accuracy of the next K-mer prediction task as our evaluation metric. This zero-shot task facilitates a direct assessment of the pre-trained model quality, ensuring equitable comparisons across various tokenizers. As depicted in *Fig.* 2A, the tested tokenizers include BPE tokenizers with vocabulary sizes ranging from 512 to 8192, and K-mer tokenizers with K values from 1 to 8 (noting that the single nucleotide tokenizer corresponds to a K-mer tokenizer with K=1). Overall, K-mer tokenizers demonstrate superior performance compared to BPE tokenizers. Among the K-mer tokenizers, the 6-mer tokenizer is selected for its robust performance with limited input tokens and its ability to maintain top-tier performance as the number of input tokens increases.

Moreover, we evaluated the performance of Mamba [23, 15], recognized for its capacity in handling long-context pre-training. To adequately assess its capabilities, we configured a Mamba model utilizing the single nucleotide tokenizer with 1.2B parameters and a context length of 98k bp. The Mamba model is compared to the 1-mer and 6-mer models under varied configurations. The comparison with the 1-mer model is straightforward; the Mamba model (denoted as Mamba×1 in *Fig.* 2A) exhibits slightly better performance with fewer input tokens but underperforms as the token count increases. Despite Mamba's context length being six times that of the 1-mer model, this feature does not translate into improved performance. This might suggest that Mamba's renowned ability to handle long-context pre-training primarily refers to cost-effective training rather than enhanced model performance [59, 8]. To compare against the 6-mer model, we adjust the input token count for the Mamba model by a factor of six (denoted as Mamba×6) to compare the models on the same base-pair basis. In this context, Mamba×6 shows slightly better performance with fewer input tokens; however, it rapidly lags as the token count increases. These findings collectively indicate that a transformer decoder architecture paired with a 6-mer tokenizer provides the most effective approach for training causal DNA language models, aligning with the configuration of the **Gener***ator*.

We further compared the **Gener***ator* model with other baseline models to evaluate their generative capabilities. As illustrated in *Fig.* 2B, we assess model performance using a dataset composed exclusively of mammalian DNA, given

that HyenaDNA and GROVER are trained solely on human genomes. The **Gener***ator* significantly outperforms other baseline models, including its variant, **Gener***ator*-All, which incorporates pre-training on non-functional regions. This suggests that the functional sequence training strategy, which emphasizes semantically rich regions, provides a more effective training scheme compared to the conventional all sequence training. This effectiveness is likely due to the sparsity of functional segments in the whole genome (less than 10%) and the disproportionate importance of these segments. Among the other baseline models, NT-multi demonstrates the best performance, likely attributable to its extensive model scale (2.5B parameters), yet it still lags significantly behind the **Gener***ator*. This result aligns with expectations, as the MLM training paradigm is recognized for its limitations in generative capabilities. Meanwhile, HyenaDNA, despite utilizing the NTP training paradigm, does not show improved performance compared to other masked language models, likely due to its overly small model size (55M parameters), insufficient for exhibiting robust generative abilities. This comparison underscores the critical role of the **Gener***ator* in bridging the gap for large-scale generative DNA language models within the eukaryotic domain.

Due to space constraints, we have chosen only to demonstrate specific examples with mammalian DNA data and a fixed K-mer prediction length of 16 bp in *Fig.* 2. A more comprehensive analysis across various taxonomic groups and K-mer lengths is provided in the appendix.

4.2 Benchmark Evaluations

In this section, we compare the **Gener***ator* with state-of-the-art genomic foundation models: Enformer [5], DNABERT-2, HyenaDNA, Nucleotide Transformer, Caduceus, and GROVER, across various benchmark tasks. To ensure a fair comparison, we uniformly fine-tune each model and perform a 10-fold cross-validation on all datasets. For each model on each dataset, we conduct a hyperparameter search, exhaustively tuning learning rates in $\{1e^{-5}, 2e^{-5}, 5e^{-5}, \ldots, 1e^{-3}, 2e^{-3}, 5e^{-3}\}$ and batch sizes in $\{64, 128, 256, 512\}$. Detailed hyperparameter settings and implementation specifics are provided in the appendix.

Nucleotide Transformer Tasks Since the NT task dataset was revised recently [13], we conducted experiments on both the original and revised datasets. The results for the revised NT tasks are provided in Table 2, and the results for the original NT tasks are provided in Table 3. Overall, the **Gener***ator* outperforms other baseline models. However, the **Gener***ator*-All variant shows some performance decline. Notably, despite its earlier release, Enformer continues to deliver competitive results in chromatin profile and regulatory element tasks. This performance could be attributed to its original training in a supervised manner specifically for chromatin and gene expression tasks. The latest release of Nucleotide Transformer, NT-v2, although smaller in size (500M), demonstrates enhanced performance compared to NT-multi (2.5B). In contrast, DNABERT-2 and GROVER, which utilize BPE tokenizers, along with HyenaDNA and Caduceus, which employ the finer-grained single nucleotide tokenizer, do not show distinct performance advantages, likely due to the limited model scope and data scale.

Genomic Benchmarks We also conducted a comparative analysis on the Genomic Benchmarks [22], which primarily focus on the human genome. The evaluation results are provided in Table 4. Overall, the **Gener***ator* still outperforms other models. However, it is worth noting that the Caduceus models also exhibit comparable performance while being significantly smaller (8M). This is likely due to the fact that Caduceus models are trained exclusively on the human genome, making them efficient and compact. Nevertheless, this exclusivity may limit their generalizability to other genomic contexts.

Gener Tasks Lastly, we evaluated the newly proposed Gener tasks, which focus on assessing genomic context comprehension across various sequence lengths and organisms. As shown in Table 5, the **Gener***ator* achieves the best performance on both gene and taxonomic classification tasks, with NT-v2 also demonstrating similar results. Further details on the evaluation of Gener tasks, including visualizations of confusion matrices, are provided in the appendix. The superior performance of the **Gener***ator* and NT-v2 is likely due to their pre-training on multispecies datasets. In contrast, despite also being trained on multispecies data, DNABERT-2 exhibits noticeable performance degradation. This may be attributed to its limited model size (117M for DNABERT-2, 500M for NT-v2, and 1.2B for **Gener***ator*) and shorter context length (3k for DNABERT-2, 12k for NT-v2, and 98k for **Gener***ator*). Other models, such as HyenaDNA and Caduceus, although trained exclusively on the human genome, still exhibit relevant generalizability on both tasks after fine-tuning, attributable to their long-context capacity ($>$100k). GROVER, on the other hand, significantly lags behind in taxonomic classification due to its limited context length (3k).

Table 2: Evaluation of the revised Nucleotide Transformer tasks. The reported values represent the Matthews correlation coefficient (MCC) averaged over 10-fold cross-validation, with the standard error in parentheses.

	Enformer (252M)	DNABERT-2 (117M)	HyenaDNA (55M)	NT-multi (2.5B)	NT-v2 (500M)	Caduceus-Ph (8M)	Caduceus-PS (8M)	GROVER (87M)	**Gener*ator*** (1.2B)	**Gener*ator*-All** (1.2B)
H2AFZ	0.522 (0.019)	0.490 (0.013)	0.455 (0.015)	0.503 (0.010)	0.524 (0.008)	0.417 (0.016)	0.501 (0.013)	0.509 (0.013)	**0.529 (0.009)**	0.506 (0.019)
H3K27ac	0.520 (0.015)	0.491 (0.010)	0.423 (0.017)	0.481 (0.020)	0.488 (0.013)	0.464 (0.018)	0.464 (0.022)	0.489 (0.023)	**0.546 (0.015)**	0.496 (0.014)
H3K27me3	0.552 (0.007)	0.599 (0.010)	0.541 (0.018)	0.593 (0.016)	0.610 (0.006)	0.547 (0.010)	0.561 (0.036)	0.600 (0.008)	**0.619 (0.008)**	0.590 (0.014)
H3K36me3	0.567 (0.017)	0.637 (0.007)	0.543 (0.010)	0.635 (0.016)	0.633 (0.015)	0.543 (0.009)	0.602 (0.008)	0.585 (0.008)	**0.650 (0.006)**	0.621 (0.013)
H3K4me1	**0.504 (0.021)**	0.490 (0.008)	0.430 (0.014)	0.481 (0.012)	0.490 (0.017)	0.411 (0.012)	0.434 (0.030)	0.468 (0.011)	**0.504 (0.010)**	0.490 (0.016)
H3K4me2	**0.626 (0.015)**	0.558 (0.013)	0.521 (0.024)	0.552 (0.022)	0.552 (0.013)	0.480 (0.013)	0.526 (0.035)	0.558 (0.012)	0.607 (0.010)	0.569 (0.012)
H3K4me3	0.635 (0.019)	0.646 (0.008)	0.596 (0.015)	0.618 (0.015)	0.627 (0.020)	0.588 (0.020)	0.611 (0.015)	0.634 (0.011)	**0.653 (0.008)**	0.628 (0.018)
H3K9ac	**0.593 (0.020)**	0.564 (0.013)	0.484 (0.022)	0.527 (0.017)	0.551 (0.016)	0.514 (0.014)	0.518 (0.018)	0.531 (0.014)	0.570 (0.017)	0.556 (0.018)
H3K9me3	0.453 (0.016)	0.443 (0.025)	0.375 (0.026)	0.447 (0.018)	0.467 (0.044)	0.435 (0.019)	0.455 (0.019)	0.441 (0.017)	**0.509 (0.013)**	0.480 (0.037)
H4K20me1	0.606 (0.016)	0.655 (0.011)	0.580 (0.009)	0.650 (0.014)	0.654 (0.011)	0.572 (0.012)	0.590 (0.020)	0.634 (0.006)	**0.670 (0.006)**	0.652 (0.010)
Enhancer	**0.614 (0.010)**	0.517 (0.011)	0.475 (0.006)	0.527 (0.012)	0.575 (0.023)	0.480 (0.008)	0.490 (0.009)	0.519 (0.009)	0.594 (0.013)	0.553 (0.020)
Enhancer type	**0.573 (0.013)**	0.476 (0.009)	0.441 (0.010)	0.484 (0.012)	0.541 (0.013)	0.461 (0.009)	0.459 (0.011)	0.481 (0.009)	0.547 (0.017)	0.510 (0.022)
Promoter all	0.745 (0.012)	0.754 (0.009)	0.693 (0.016)	0.761 (0.009)	0.780 (0.012)	0.707 (0.017)	0.722 (0.014)	0.721 (0.011)	**0.795 (0.005)**	0.765 (0.009)
Promoter non-TATA	0.763 (0.012)	0.769 (0.009)	0.723 (0.013)	0.773 (0.010)	0.785 (0.009)	0.740 (0.012)	0.746 (0.009)	0.739 (0.018)	**0.801 (0.005)**	0.786 (0.007)
Promoter TATA	0.793 (0.026)	0.784 (0.036)	0.648 (0.044)	0.944 (0.016)	0.919 (0.028)	0.868 (0.023)	0.853 (0.034)	0.891 (0.041)	**0.950 (0.009)**	0.862 (0.024)
Splice acceptor	0.749 (0.007)	0.837 (0.006)	0.815 (0.049)	0.958 (0.003)	**0.965 (0.004)**	0.906 (0.015)	0.939 (0.012)	0.812 (0.012)	0.964 (0.003)	0.951 (0.006)
Splice site all	0.739 (0.011)	0.855 (0.005)	0.854 (0.053)	0.964 (0.003)	**0.968 (0.003)**	0.941 (0.006)	0.942 (0.012)	0.849 (0.015)	0.966 (0.003)	0.959 (0.003)
Splice donor	0.780 (0.007)	0.861 (0.004)	0.943 (0.024)	0.970 (0.002)	0.976 (0.003)	0.944 (0.026)	0.964 (0.010)	0.842 (0.009)	**0.977 (0.002)**	0.971 (0.002)

Table 3: Evaluation of the original Nucleotide Transformer tasks. The reported values represent the Matthews correlation coefficient (MCC) averaged over 10-fold cross-validation, with the standard error in parentheses.

	Enformer (252M)	DNABERT-2 (117M)	HyenaDNA (55M)	NT-multi (2.5B)	NT-v2 (500M)	Caduceus-Ph (8M)	Caduceus-PS (8M)	GROVER (87M)	**Gener*ator*** (1.2B)	**Gener*ator*-All** (1.2B)
H3	0.724 (0.018)	0.785 (0.012)	0.781 (0.015)	0.793 (0.013)	0.788 (0.010)	0.794 (0.012)	0.772 (0.022)	0.768 (0.008)	**0.806 (0.005)**	0.803 (0.007)
H3K14ac	0.284 (0.024)	0.515 (0.009)	**0.608 (0.020)**	0.538 (0.009)	0.538 (0.015)	0.564 (0.033)	0.596 (0.038)	0.548 (0.020)	0.605 (0.008)	0.580 (0.038)
H3K36me3	0.345 (0.019)	0.591 (0.005)	0.614 (0.014)	0.618 (0.011)	0.618 (0.015)	0.590 (0.018)	0.611 (0.048)	0.563 (0.017)	**0.657 (0.007)**	0.631 (0.013)
H3K4me1	0.291 (0.016)	0.512 (0.008)	0.512 (0.008)	0.541 (0.005)	0.544 (0.009)	0.468 (0.015)	0.487 (0.029)	0.461 (0.018)	**0.553 (0.009)**	0.549 (0.018)
H3K4me2	0.207 (0.021)	0.333 (0.013)	**0.455 (0.028)**	0.324 (0.014)	0.302 (0.020)	0.332 (0.034)	0.431 (0.016)	0.403 (0.042)	0.424 (0.013)	0.400 (0.015)
H3K4me3	0.156 (0.022)	0.353 (0.021)	**0.550 (0.015)**	0.408 (0.011)	0.437 (0.028)	0.490 (0.042)	0.528 (0.033)	0.458 (0.022)	0.512 (0.009)	0.473 (0.047)
H3K79me3	0.498 (0.013)	0.615 (0.010)	0.669 (0.014)	0.623 (0.010)	0.621 (0.012)	0.641 (0.028)	**0.682 (0.018)**	0.626 (0.026)	0.670 (0.011)	0.631 (0.021)
H3K9ac	0.415 (0.020)	0.545 (0.009)	0.586 (0.021)	0.547 (0.011)	0.567 (0.020)	0.575 (0.024)	0.564 (0.018)	0.581 (0.015)	**0.612 (0.006)**	0.603 (0.019)
H4	0.735 (0.023)	0.797 (0.008)	0.763 (0.012)	0.808 (0.007)	0.795 (0.008)	0.788 (0.010)	0.799 (0.010)	0.769 (0.017)	**0.815 (0.008)**	0.808 (0.010)
H4ac	0.275 (0.022)	0.465 (0.013)	0.564 (0.011)	0.492 (0.014)	0.502 (0.025)	0.548 (0.027)	0.585 (0.018)	0.530 (0.017)	**0.592 (0.015)**	0.565 (0.035)
Enhancer	0.454 (0.029)	0.525 (0.026)	0.520 (0.031)	0.545 (0.028)	0.561 (0.029)	0.522 (0.024)	0.511 (0.026)	0.516 (0.018)	**0.580 (0.015)**	0.540 (0.026)
Enhancer type	0.312 (0.043)	0.423 (0.018)	0.403 (0.056)	0.444 (0.022)	0.444 (0.036)	0.403 (0.028)	0.410 (0.026)	0.433 (0.029)	**0.477 (0.017)**	0.463 (0.023)
Promoter all	0.910 (0.004)	0.945 (0.003)	0.919 (0.003)	0.951 (0.004)	0.952 (0.002)	0.937 (0.002)	0.941 (0.003)	0.926 (0.004)	**0.962 (0.002)**	0.955 (0.002)
Promoter non-TATA	0.910 (0.006)	0.944 (0.003)	0.919 (0.004)	0.955 (0.003)	0.952 (0.003)	0.935 (0.007)	0.940 (0.002)	0.925 (0.006)	**0.962 (0.001)**	0.955 (0.002)
Promoter TATA	0.920 (0.012)	0.911 (0.011)	0.881 (0.020)	0.919 (0.008)	0.933 (0.009)	0.895 (0.010)	0.903 (0.010)	0.891 (0.009)	**0.948 (0.008)**	0.931 (0.007)
Splice acceptor	0.772 (0.007)	0.909 (0.004)	0.935 (0.005)	0.973 (0.002)	0.973 (0.004)	0.918 (0.017)	0.907 (0.015)	0.912 (0.010)	**0.981 (0.002)**	0.957 (0.009)
Splice site all	0.831 (0.012)	0.950 (0.003)	0.917 (0.006)	0.974 (0.004)	0.975 (0.002)	0.935 (0.011)	0.953 (0.005)	0.919 (0.005)	**0.978 (0.001)**	0.973 (0.002)
Splice donor	0.813 (0.015)	0.927 (0.003)	0.894 (0.013)	0.974 (0.002)	0.977 (0.007)	0.912 (0.009)	0.930 (0.010)	0.888 (0.012)	**0.978 (0.002)**	0.967 (0.005)

Table 4: Evaluation of the Genomic Benchmarks. The reported values represent the accuracy averaged over 10-fold cross-validation, with the standard error in parentheses.

	DNABERT-2 (117M)	HyenaDNA (55M)	NT-v2 (500M)	Caduceus-Ph (8M)	Caduceus-PS (8M)	GROVER (87M)	**Gener*ator*** (1.2B)	**Gener*ator*-All** (1.2B)
Coding vs. Intergenomic	0.951 (0.002)	0.902 (0.004)	0.955 (0.001)	0.933 (0.001)	0.944 (0.002)	0.919 (0.002)	**0.963 (0.000)**	0.959 (0.001)
Drosophila Enhancers Stark	0.774 (0.011)	0.770 (0.016)	0.797 (0.009)	**0.827 (0.010)**	0.816 (0.015)	0.761 (0.011)	0.821 (0.005)	0.768 (0.015)
Human Enhancers Cohn	0.758 (0.005)	0.725 (0.009)	0.756 (0.006)	0.747 (0.003)	0.749 (0.003)	0.738 (0.003)	**0.763 (0.002)**	0.754 (0.006)
Human Enhancers Ensembl	0.918 (0.003)	0.901 (0.003)	0.921 (0.004)	**0.924 (0.002)**	0.923 (0.002)	0.911 (0.004)	0.917 (0.002)	0.912 (0.002)
Human Ensembl Regulatory	0.874 (0.007)	0.932 (0.001)	**0.941 (0.001)**	0.938 (0.004)	**0.941 (0.002)**	0.897 (0.001)	0.928 (0.001)	0.926 (0.001)
Human non-TATA Promoters	0.957 (0.008)	0.894 (0.023)	0.932 (0.006)	**0.961 (0.003)**	**0.961 (0.002)**	0.950 (0.005)	0.958 (0.001)	0.955 (0.005)
Human OCR Ensembl	0.806 (0.003)	0.774 (0.004)	0.813 (0.001)	0.825 (0.004)	**0.826 (0.003)**	0.789 (0.002)	0.823 (0.002)	0.812 (0.003)
Human vs. Worm	0.977 (0.001)	0.958 (0.004)	0.976 (0.001)	0.975 (0.001)	0.976 (0.001)	0.966 (0.001)	**0.980 (0.000)**	0.978 (0.001)
Mouse Enhancers Ensembl	0.865 (0.014)	0.756 (0.030)	0.855 (0.018)	0.788 (0.028)	0.826 (0.021)	0.742 (0.025)	**0.871 (0.015)**	0.784 (0.027)

Table 5: Evaluation of the Gener tasks. The reported values represent the weighted F1 score averaged over 10-fold cross-validation, with the standard error in parentheses.

	DNABERT-2 (117M)	HyenaDNA (55M)	NT-v2 (500M)	Caduceus-Ph (8M)	Caduceus-PS (8M)	GROVER (87M)	**Gener*ator*** (1.2B)	**Gener*ator*-All** (1.2B)
Gene	0.660 (0.002)	0.610 (0.007)	0.692 (0.005)	0.629 (0.005)	0.644 (0.007)	0.630 (0.003)	**0.700 (0.002)**	0.687 (0.003)
Taxonomic	0.922 (0.003)	0.970 (0.024)	0.981 (0.001)	0.958 (0.021)	0.968 (0.006)	0.843 (0.006)	**0.999 (0.000)**	0.998 (0.001)

4.3 Central Dogma

In our experimental setup, we selected two target protein families from the UniProt [7] database: the Histone and Cytochrome P450 families. By cross-referencing gene IDs and protein IDs, we extracted the corresponding protein-coding DNA sequences from RefSeq [39]. These sequences served as training data for fine-tuning the **Gener***ator* model, directing it to generate analogous protein-coding DNA sequences.

To assess the quality of generation, we compared several summary statistics. The results for the Histone family are provided in *Fig.* 3, while the evaluation results for the Cytochrome P450 family are provided in *Fig.* 4. After deduplication, the lengths of the generated DNA sequences and their translated protein sequences, using a codon table, closely resemble the distributions observed in the target families. This preliminary validation suggests that our generated DNA sequences maintain stable codon structures that are translatable into proteins. We conducted a more in-depth structural and functional analysis of these translated protein sequences. First, we assessed whether protein language models 'recognize' these generated protein sequences by calculating their perplexity (PPL) using Progen2 [38]. The results show that the PPL distribution of generated sequences closely matches that of the natural families and significantly differs from the shuffled sequences.

Furthermore, we used AlphaFold3 to predict the folded structures of the generated protein sequences and employed Foldseek [58] to find analogous proteins in the Protein Data Bank (PDB) [10]. Remarkably, we identified numerous instances where the conformations of the generated sequences exhibited high similarity to established structures in the PDB (TM-score > 0.8). This structural congruence is observed despite substantial divergence in sequence composition, as indicated by sequence identities less than 0.3. This low sequence identity positively suggests that the model is not merely replicating existing protein sequences but has learned the underlying principles to design new molecules with similar structures. This highlights the capability of the **Gener***ator* in generating biologically relevant sequences.

4.4 Enhancer Design

We employed the enhancer activity data from DeepSTARR [17], following the dataset split initially proposed by DeepSTARR and later adopted by NT. Using this data, we developed an enhancer activity predictor by fine-tuning the **Gener***ator*. This predictor surpasses the accuracy of DeepSTARR and NT-multi (Table 6), establishing itself as the current state-of-the-art predictor. By applying our refined SFT approach as outlined in *Sec.* 3.4.4, we generated a collection of candidate enhancer sequences with specific activity profiles. As illustrated in *Fig.* 5, the predicted activities of these candidates exhibit significant differentiation between the generated high/low-activity enhancers and natural samples.

To our knowledge, this represents one of the first attempts to use LLMs for prompt-guided design of DNA sequences, highlighting the capability of the **Gener***ator* in this domain. These generated sequences, and more broadly, this sequence design paradigm using the **Gener***ator*, merit further exploration. Our approach underscores the potential of the **Gener***ator* model to transform DNA sequence design methodologies, providing a novel pathway for the conditional design of DNA sequences using LLMs. In our subsequent research, we plan to extend our evaluations through further validations in wet lab conditions to explore the real-world applicability of these designed sequences.

Table 6: Evaluation of the DeepSTARR dataset. The reported values represent the Pearson correlation coefficient.

	DeepSTARR	NT-multi	**Gener***ator*
Developmental	<u>0.68</u>	0.64	**0.70**
Housekeeping	0.74	<u>0.75</u>	**0.79**

5 Discussion & Future Development

In this study, we presented **Gener***ator*, a generative genomic foundation model based on the transformer decoder architecture, meticulously trained on an extensive dataset of eukaryotic DNA sequences. Our model demonstrates state-of-the-art performance across various genomic tasks. This work marks a significant advancement in merging generative capabilities with genomic data analysis, delivering a model that not only comprehends the semantics of DNA but also excels in sequence design and optimization.

A pivotal feature of the **Gener***ator* is its adherence to the central dogma of molecular biology, generating protein-coding DNA sequences that can produce proteins analogous to those in established families, such as Histones and Cytochrome P450. This capability is validated by the structural congruence of translated proteins with existing structures, as verified by AlphaFold and Foldseek analyses. Additionally, in the sequence design task, the **Gener***ator* demonstrates

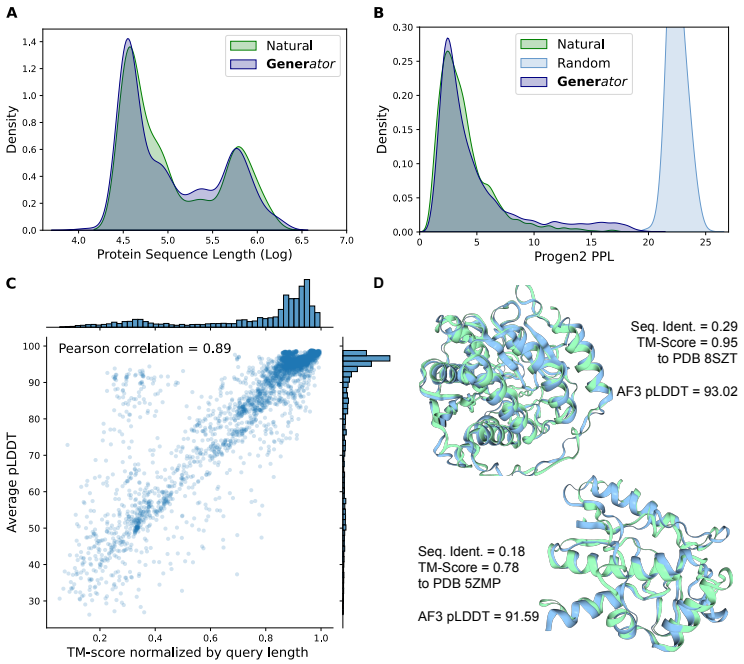

Figure 3: Histone generation. (A) Distribution densities of the protein sequence lengths for generated and natural samples. (B) Distribution densities of Progen2 PPL for generated and natural samples, along with randomly shuffled sequences. (C) Scatter plot of TM-score and AlphaFold3 prediction confidence (pLDDT) with marginal distributions. (D) Two folded structures of generated samples displaying structural congruence with natural samples.

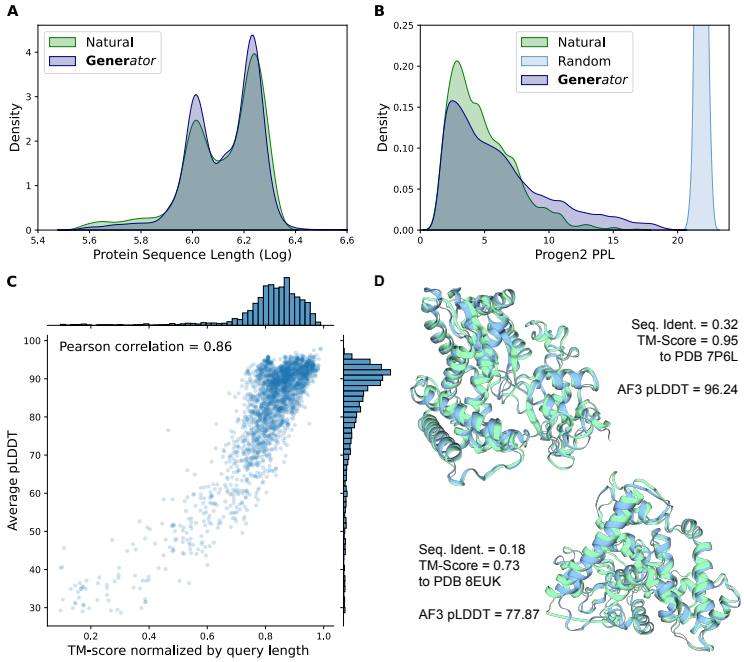

Figure 4: Cytochrome P450 generation. (A) Distribution densities of the protein sequence lengths for generated and natural samples. (B) Distribution densities of Progen2 PPL for generated and natural samples, along with randomly shuffled sequences. (C) Scatter plot of TM-score and AlphaFold3 prediction confidence (pLDDT) with marginal distributions. (D) Two folded structures of generated samples displaying structural congruence with natural samples.

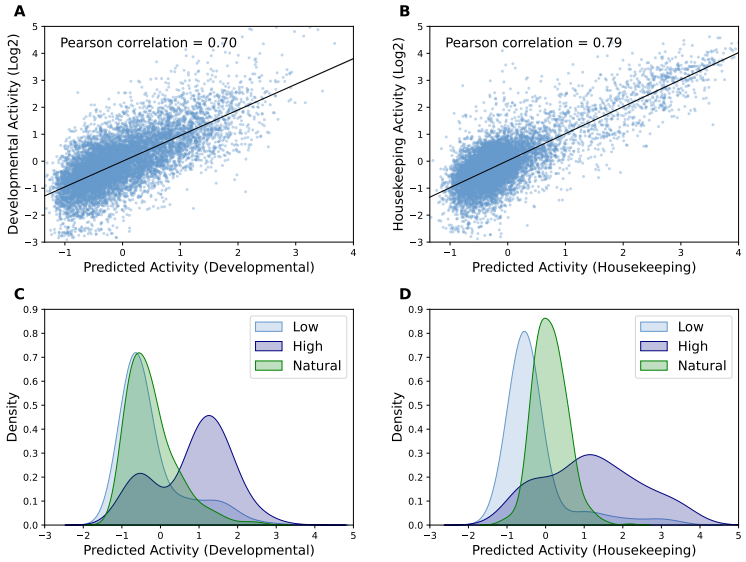

Figure 5: Enhancer design. (A-B) Pearson correlation between the predicted enhancer activity and the measured activity. (C-D) Distribution densities of the predicted activity of generated enhancer sequences with distinct activity profiles.

its capability in optimizing enhancer sequences, achieving specific activity profiles with precision and computational efficiency. This underscores the model's potential to advance the field of synthetic biology, facilitating precise sequence design and optimization and enhancing our understanding and manipulation of complex biological systems.

The remarkable performance of the **Gener***ator* can be primarily attributed to its training on a diverse and extensive dataset, which endows it with enhanced robustness and versatility in genomic data analysis. Through comprehensive investigation and careful selection of suitable tokenization and data filtering strategies, the **Gener***ator* efficiently captures the semantic richness of functional DNA regions, consequently achieving outstanding results in downstream tasks.

Nevertheless, this work also reveals certain challenges and limitations. A notable limitation of the **Gener***ator* is its exclusive focus on eukaryotic DNA, without considering prokaryotic and viral data. In this context, the **Gener***ator* serves as a complementary model to Evo, which is trained on prokaryotic and viral genomes. Consequently, we lack appropriate tasks to directly compare the **Gener***ator* with Evo [36]. This delineation reflects the inherent differences in cellular mechanisms between eukaryotes and prokaryotes, despite both utilizing DNA as genetic material. Variations in gene architecture and organelle types justify this specialized focus. Considering the computational costs of training LLMs, partitioning datasets into eukaryotic and prokaryotic-plus-viral categories emerges as a practical and effective strategy.

In future work, we plan to train a prokaryotic-plus-viral version of **Gener***ator*, facilitating a fair and comprehensive comparison with Evo. Additionally, the remarkable performance of the **Gener***ator* in the gene classification task suggests its potential for gene annotation tasks, which involve identifying functional regions within entire DNA sequences. We are currently developing a specialized model, **Gener***anno*, dedicated to this endeavor. This model is in the training phase and will be detailed in our subsequent work.

In summary, the **Gener***ator* exemplifies the potential of generative models in genomics, with its primary contribution lying in fostering innovation that enhances current biological understanding. Serving as a foundational tool in genomic research, the model holds promise for significant contributions to scientific and technological advances in genomics. The prospective integration of the **Gener***ator* into practical applications, such as clinical genomics, warrants further exploration. By aligning sequence generation capabilities with specific therapeutic objectives, the **Gener***ator* could play a pivotal role in advancing precision medicine and biotechnology interventions.

We are committed to promoting transparency and collaboration in research. To this end, all necessary materials to replicate this work—including data, code, and model weights, will be made fully open-source on the GenerTeam GitHub page. We hope that this contribution will help to advance the development of the DNA language model community.

References

[1] On protein synthesis. *Symposia of the Society for Experimental Biology*, 12:138–63, 1958.

[2] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 630:493 – 500, 2024.

[3] Walid Ahmad, Elana Simon, Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. Chemberta-2: Towards chemical foundation models. *arXiv preprint arXiv:2209.01712*, 2022.

[4] Microsoft Research AI4Science and Microsoft Azure Quantum. The impact of large language models on scientific discovery: a preliminary study using gpt-4. *arXiv preprint arXiv:2311.07361*, 2023.

[5] 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.

[6] Žiga Avsec, Melanie Weilert, Avanti Shrikumar, Sabrina Krueger, Amr Alexandari, Khyati Dalal, Robin Fropf, Charles McAnany, Julien Gagneur, Anshul Kundaje, et al. Base-resolution models of transcription-factor binding reveal soft motif syntax. *Nature genetics*, 53(3):354–366, 2021.

[7] Alex Bateman, Maria Jesus Martin, Sandra E. Orchard, Michele Magrane, Shadab Ahmad, Emanuele Alpi, Emily H. Bowler-Barnett, et al. Uniprot: the universal protein knowledgebase in 2023. *Nucleic Acids Research*, 51:D523 – D531, 2022.

[8] Assaf Ben-Kish, Itamar Zimerman, Shady Abu-Hussein, Nadav Cohen, Amir Globerson, Lior Wolf, and Raja Giryes. Decimamba: Exploring the length extrapolation potential of mamba. *arXiv*, 2024. URL `https://arxiv.org/abs/2406.14528`.

[9] Gonzalo Benegas, Carlos Albors, Alan J Aw, Chengzhong Ye, and Yun S Song. A dna language model based on multispecies alignment predicts the effects of genome-wide variants. *Nature Biotechnology*, pages 1–6, 2025.

[10] Stephen K. Burley, Charmi Bhikadiya, Chunxiao Bi, Sebastian Bittrich, Li Chen, Gregg V. Crichlow, Cole H. Christie, Kenneth Dalenberg, Luigi Di Costanzo, Jose M. Duarte, Shuchismita Dutta, Zukang Feng, Sai J. Ganesan, David S. Goodsell, Sutapa Ghosh, Rachel Kramer Green, Vladimir Guranovic, Dmytro Guzenko, Brian P. Hudson, Catherine L. Lawson, Yuhe Liang, Robert Lowe, Harry Namkoong, Ezra Peisach, Irina Persikova, Chris Randle, Alexander S. Rose, Yana Rose, Andrej Sali, Joan Segura, Monica Sekharan, Chenghua Shao, Yi-Ping Tao, Maria Voigt, John D. Westbrook, Jasmine Y. Young, Christine Zardecki, and Marina Zhuravleva. Rcsb protein data bank: powerful new tools for exploring 3d structures of biological macromolecules for basic and applied research and education in fundamental biology, biomedicine, biotechnology, bioengineering and energy sciences. *Nucleic Acids Research*, 49:D437 – D451, 2020.

[11] Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. Chemberta: large-scale self-supervised pretraining for molecular property prediction. *arXiv preprint arXiv:2010.09885*, 2020.

[12] Le Cong, Fei Ann Ran, David B. T. Cox, Shuailiang Lin, Robert Barretto, Naomi Habib, Patrick D. Hsu, Xuebing Wu, Wenyan Jiang, Luciano A. Marraffini, and Feng Zhang. Multiplex genome engineering using crispr/cas systems. *Science*, 339:819 – 823, 2013.

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

[14] Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*, 2024.

[15] Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv*, 2024. URL `https://arxiv.org/abs/2405.21060`.

[16] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

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

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

[19] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks*, 107:3–11, 2018.

[20] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, et al. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(10): 7112–7127, 2021.

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

[22] 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.

[23] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv*, 2024. URL `https://arxiv.org/abs/2312.00752`.

[24] Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022.

[25] Thomas Hayes, Roshan Rao, Halil Akin, Nicholas James Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q. Tran, Jonathan Deaton, Marius Wiggert, Rohil Badkundri, Irhum Shafkat, Jun Gong, Alexander Derry, Raul S. Molina, Neil Thomas, Yousuf Khan, Chetan Mishra, Carolyn Kim, Liam J. Bartie, Matthew Nemeth, Patrick D. Hsu, Tom Sercu, Salvatore Candido, and Alexander Rives. Simulating 500 million years of evolution with a language model. *bioRxiv*, 2024.

[26] Stephen Heller, Alan McNaught, Stephen Stein, Dmitrii Tchekhovskoi, and Igor Pletnev. Inchi-the worldwide chemical structure identifier standard. *Journal of cheminformatics*, 5:1–9, 2013.

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

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

[29] John M. Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596:583 – 589, 2021.

[30] Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. Self-referencing embedded strings (selfies): A 100% robust molecular string representation. *Machine Learning: Science and Technology*, 1(4):045024, 2020.

[31] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Conference on Empirical Methods in Natural Language Processing*, 2018.

[32] Qiuyi Li, Celine Scornavacca, Nicolas Galtier, and Yao-Ban Chan. The multilocus multispecies coalescent: A flexible new model of gene family evolution. *Systematic Biology*, 70(4):822–837, 11 2020. ISSN 1063-5157. doi:10.1093/sysbio/syaa084. URL `https://doi.org/10.1093/sysbio/syaa084`.

[33] Qiuyi Li, Yao-ban Chan, Nicolas Galtier, and Celine Scornavacca. The effect of copy number hemiplasy on gene family evolution. *Systematic Biology*, 73(2):355–374, 02 2024. ISSN 1063-5157. doi:10.1093/sysbio/syae007. URL `https://doi.org/10.1093/sysbio/syae007`.

[34] Zeming Lin, Halil Akin, Roshan Rao, Brian L. Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic level protein structure with a language model. *bioRxiv*, 2022.

[35] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

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

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

[38] Erik Nijkamp, Jeffrey A Ruffolo, Eli N Weinstein, Nikhil Naik, and Ali Madani. Progen2: exploring the boundaries of protein language models. *Cell systems*, 14(11):968–978, 2023.

[39] Nuala A. O'Leary, Mathew W. Wright, James Rodney Brister, Stacy Ciufo, Diana Haddad, Richard McVeigh, Bhanu Rajput, Barbara Robbertse, Brian Smith-White, et al. Reference sequence (refseq) database at ncbi: current status, taxonomic expansion, and functional annotation. *Nucleic Acids Research*, 44:D733 – D745, 2015.

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

[41] Rafael Josip Penić, Tin Vlašić, Roland G Huber, Yue Wan, and Mile Šikić. Rinalmo: General-purpose rna language models can generalize well on structure prediction tasks. *arXiv preprint arXiv:2403.00043*, 2024.

[42] Jonathan Pevsner. Bioinformatics and functional genomics. 2003.

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

[44] Michael Poli, Jue Wang, Stefano Massaroli, Jeffrey Quesnelle, Ryan Carlow, Eric Nguyen, and Armin Thomas. StripedHyena: Moving Beyond Transformers with Hybrid Signal Processing Models, 12 2023. URL `https://github.com/togethercomputer/stripedhyena`.

[45] Roshan Rao, Jason Liu, Robert Verkuil, Joshua Meier, John F. Canny, P. Abbeel, Tom Sercu, and Alexander Rives. Msa transformer. *bioRxiv*, 2021.

[46] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3505–3506, 2020.

[47] Milong Ren, Chungong Yu, Dongbo Bu, and Haicang Zhang. Accurate and robust protein sequence design with carbondesign. *Nature Machine Intelligence*, 6(5):536–547, 2024.

[48] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data. *Advances in neural information processing systems*, 33:12559–12571, 2020.

[49] Jerret Ross, Brian Belgodere, Vijil Chenthamarakshan, Inkit Padhi, Youssef Mroueh, and Payel Das. Large-scale chemical language representations capture molecular structure and properties. *Nature Machine Intelligence*, 4(12): 1256–1264, 2022.

[50] Melissa Sanabria, Jonas Hirsch, Pierre M. Joubert, and Anna R. Poetsch. Dna language model grover learns sequence context in the human genome. *Nat. Mac. Intell.*, 6:911–923, 2024.

[51] Frederick Sanger, Gillian M. Air, Bart Barrell, Nigel L. Brown, A. R. Coulson, John C. Fiddes, Clyde A. Hutchison, Patrick M. Slocombe, and M. Smith. Nucleotide sequence of bacteriophage ϕx174 dna. *Nature*, 265:687–695, 1977.

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

[53] Philippe Schwaller, Daniel Probst, Alain C Vaucher, Vishnu H Nair, David Kreutter, Teodoro Laino, and Jean-Louis Reymond. Mapping the space of chemical reactions using attention-based neural networks. *Nature machine intelligence*, 3(2):144–152, 2021.

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

[55] Jay A. Shendure and Hanlee P. Ji. Next-generation dna sequencing. *Nature Biotechnology*, 26:1135–1145, 2008.

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

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

[58] Michel Van Kempen, Stephanie S Kim, Charlotte Tumescheit, Milot Mirdita, Jeongjae Lee, Cameron LM Gilchrist, Johannes Söding, and Martin Steinegger. Fast and accurate protein structure search with foldseek. *Nature biotechnology*, 42(2):243–246, 2024.

[59] Roger Waleffe, Wonmin Byeon, Duncan Riach, Brandon Norick, Vijay Korthikanti, Tri Dao, Albert Gu, Ali Hatamizadeh, Sudhakar Singh, Deepak Narayanan, Garvit Kulshreshtha, Vartika Singh, Jared Casper, Jan Kautz, Mohammad Shoeybi, and Bryan Catanzaro. An empirical study of mamba-based language models. *arXiv*, 2024. URL `https://arxiv.org/abs/2406.07887`.

[60] Ning Wang, Jiang Bian, Yuchen Li, Xuhong Li, Shahid Mumtaz, Linghe Kong, and Haoyi Xiong. Multi-purpose rna language modelling with motif-aware pretraining and type-guided fine-tuning. *Nature Machine Intelligence*, pages 1–10, 2024.

[61] Sheng Wang, Yuzhi Guo, Yuhong Wang, Hongmao Sun, and Junzhou Huang. Smiles-bert: large scale unsupervised pre-training for molecular property prediction. In *Proceedings of the 10th ACM international conference on bioinformatics, computational biology and health informatics*, pages 429–436, 2019.

[62] Xi Wang, Ruichu Gu, Zhiyuan Chen, Yongge Li, Xiaohong Ji, Guolin Ke, and Han Wen. Uni-rna: universal pre-trained models revolutionize rna research. *bioRxiv*, pages 2023–07, 2023.

[63] Kenneth H Wolfe, Paul M Sharp, and Wen-Hsiung Li. Mutation rates differ among regions of the mammalian genome. *Nature*, 337(6204):283–285, 1989.

[64] Daoan Zhang, Weitong Zhang, Bing He, Jianguo Zhang, Chenchen Qin, and Jianhua Yao. Dnagpt: a generalized pretrained tool for multiple dna sequence analysis tasks. *bioRxiv*, pages 2023–07, 2023.

[65] Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model. *ArXiv*, abs/2401.02385, 2024.

[66] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, et al. A survey of large language models. *ArXiv*, abs/2303.18223, 2023.

[67] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

[68] Gengmo Zhou, Zhifeng Gao, Qiankun Ding, Hang Zheng, Hongteng Xu, Zhewei Wei, Linfeng Zhang, and Guolin Ke. Uni-mol: A universal 3d molecular representation learning framework. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=6K2RM6wVqKu`.

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

A Supplementary Experiment Results

A.1 Gener Tasks

The confusion matrices for the gene classification task are shown in *Fig.* S1, whereas *Fig.* S2 illustrates those for the taxonomic classification task. For both tasks, the **Gener***ator* not only excels in average performance but also maintains top performance across different categories of genes and taxonomic groups.

A.2 Next K-mer Prediction

The evaluations of next K-mer prediction are provided in *Fig.* S3 and *Fig.* S4. Specifically, *Fig.* S3 presents the evaluations for tokenizer comparison, where the 6-mer tokenizer outperforms others across different taxonomic groups. *Fig.* S4 presents the evaluations for model comparison, where the **Gener***ator* consistently outperforms others across various taxonomic groups.

B Details of Pre-training

In the pre-training phase, we employed the AdamW [35] optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and weight decay $= 0.1$. The learning rate schedule incorporated a linear warm-up followed by cosine decay: the learning rate increased linearly from 0 to its peak value over the first 2000 steps, then followed a cosine decay, decreasing until it reached 10% of the peak value by the end of the training process. The peak learning rate was established at $4e^{-4}$, and gradient clipping was applied with a norm threshold of 1.0. We adhered to standard practices in pre-training LLMs, using a batch size that encompasses 2 million tokens. Given the maximum sequence length of 16,384 tokens, this configuration resulted in each batch comprising 128 samples, where each sample was initialized with a random starting point between 0 and 5 at each epoch. The complete pre-training of the **Gener***ator* spanned 6 epochs, amounting to approximately 185,000 steps. To enhance computational efficiency, we employed optimization techniques like Flash Attention [16, 14] and the Zero Redundancy Optimizer [46, 67]. The pre-training process, utilizing 32 NVIDIA A100 GPUs, was completed in 368 hours. This configuration achieved a model FLOPs utilization (MFU) of approximately 46%.

The summary statistics of the pre-training data in terms of the number of nucleotides and number of genes are provided in Table S1 and Table S2. The pre-training losses for the **Gener***ator* and **Gener***ator*-All are presented in *Fig.* S5. Although the **Gener***ator*-All exhibits a lower pre-training loss compared to the **Gener***ator*, it consistently underperforms in almost all benchmark tests, as discussed in previous sections. This discrepancy is likely due to the inclusion of non-functional regions, which often contain highly repetitive and simple segments. Additionally, the pre-training losses for different tokenizers and the Mamba model are shown in *Fig.* S6. The significant variation in vocabulary sizes among the tokenizers results in considerable differences in pre-training losses, making direct comparisons challenging. This issue inspires the proposition of next K-mer prediction for a more effective model comparison.

C Details of Benchmark Experiments

C.1 Benchmarks

In this paper, we conducted benchmark evaluations on two widely used benchmarks, Nucleotide Transformer tasks [13] and Genomic Benchmarks [22], as well as on the Gener tasks we propose. For Nucleotide Transformer tasks, there are two different versions available: the original version can be accessed here, and the revised version is available here. For Genomic Benchmarks, the dataset can be downloaded from here.

C.2 Metrics

Regarding evaluation metrics, we follow the original settings of the benchmarks. For Nucleotide Transformer tasks, we employ the Matthews correlation coefficient (MCC):

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}},$$

where TP, TN, FP, FN represent true positives, true negatives, false positives, and false negatives, respectively. For Genomic Benchmarks, accuracy is used as the evaluation metric. For Gener tasks, we utilize the weighted F1 score for multi-classification, calculated as follows:

$$\text{Precision}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i}, \quad \text{Recall}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i},$$

$$\text{F1}_i = 2 \times \frac{\text{Precision}_i \times \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i}, \quad \text{F1}_{\text{weighted}} = \sum_{i=1}^{n} w_i \times \text{F1}_i,$$

where $w_i = n_i/N$ denotes the weight for class i, n_i is the number of samples in class i, and N is the total number of samples.

C.3 Baselines

To comprehensively assess the capability of the **Gener***ator* in sequence comprehension, we include several state-of-the-art models as baselines. These models differ in terms of training data, model size, model architecture, and other aspects. Table S3 provides a summary of these baselines. Where applicable, evaluation results for the original and revised NT tasks are sourced directly from the NT paper [13]. For the remaining baseline models and datasets, we conduct consistent and fair evaluation tasks ourselves to obtain the results. These self-evaluated baseline models can be accessed through the following HuggingFace[1] repositories:

- DNABERT-2: `zhihan1996/DNABERT-2-117M`
- HyenaDNA: `LongSafari/hyenadna-large-1m-seqlen`
- NT-v2: `InstaDeepAI/nucleotide-transformer-v2-500m-multi-species`
- Caduceus-Ph: `kuleshov-group/caduceus-ph_seqlen-131k_d_model-256_n_layer-16`
- Caduceus-PS: `kuleshov-group/caduceus-ps_seqlen-131k_d_model-256_n_layer-16`
- GROVER: `PoetschLab/GROVER`

C.4 Experimental Setups

In our benchmark experiments, we retained the optimizer configuration from the pre-training phase, with $\beta_1 = 0.9$, $\beta_2 = 0.95$ and weight decay $= 0.1$. We adopted the 'reduce on plateau' strategy for the learning rate scheduler and implemented early stopping based on validation metrics, with a patience of 5. Optimal learning rates and batch sizes for various models and datasets were determined through hyperparameter search, as detailed in Tables S4 to S7. For Nucleotide Transformer tasks and Genomic Benchmarks, all models had sufficient context length, allowing them to utilize the entire input sequences. In contrast, for Gener tasks, input sequences exceeding the available context length were truncated to fit within the model constraints. For all causal language models, we performed prediction through an additional linear layer using the embedding of the `<EOS>` token, while for masked language models, we used the `<BOS>` token or `<CLS>` token instead. Notably, for Caduceus, we followed its original configuration using mean pooling and applied the recommended reverse complement data augmentation for Caduceus-Ph. All models obtained embeddings from their final layer and underwent full fine-tuning. All evaluation metrics were obtained through 10-fold cross-validation.

D Details of Central Dogma

We sourced the protein sequences and the corresponding protein-coding DNA fragments for the Histone and Cytochrome P450 families from the UniProt [7] database. Each family had approximately 400,000 entries. The **Gener***ator* model was then fine-tuned on these datasets using the protein-coding sequences, with a learning rate of $5e^{-5}$ and a batch size of 1024. For the fine-tuned models, we generated 100,000 sequences for each combination of temperature $T \in \{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ and nucleus sampling $P \in \{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. Following translation into protein sequences, duplicates were removed by comparing the generated sequences with the training set, resulting in a unique selection of 10,000 samples per family. The hyperparameters were set to $T = 0.6$, $P = 0.6$ for the Cytochrome P450 family (*Fig.* 4), and $T = 1.0$, $P = 0.8$ for the Histone family (*Fig.* 3).

E Details of Enhancer Design

For the enhancer activity prediction, we adhered to the data partitioning of DeepSTARR [17]. We conducted the same hyperparameter search strategy as in our benchmark experiments and selected a learning rate of $2e^{-5}$ and a batch size of 64 for training the activity predictor. For enhancer generation, we selected enhancer sequences from the top and bottom

[1] `https://huggingface.co`

quartiles of activity values within the DeepSTARR training set. These sequences were used to perform supervised fine-tuning (SFT) on the **Gener***ator*, labeled with prompts `<high>` and `<low>` for high and low activity sequences, respectively. The hyperparameters for SFT were set to a learning rate of $1e^{-5}$ and a batch size of 2048. During the generation process, we applied the same hyperparameter search strategy detailed in *Sec.* D. The hyperparameters used for visualization in *Fig.* 5 were set to $T = 0.5$, $P = 0.7$ for the developmental activity, and $T = 0.8$, $P = 0.6$ for the housekeeping activity.

F Computational Resources

In this study, the primary computational resources utilized for model training and inference included NVIDIA V100 and A100 GPUs. Prior to pre-training, we trained 14 models for one epoch each to evaluate the DNA sequence modeling capabilities of various tokenizers. Our pre-training phase involved two models that varied based on the training data used. In the downstream tasks, we executed over 10,000 runs, which entailed an extensive hyperparameter search across 36 combinations and utilized 10-fold cross-validation for each benchmark task per model. Detailed statistics on the usage of main computational resources are presented in Table S8.

G Supplementary Figures & Tables

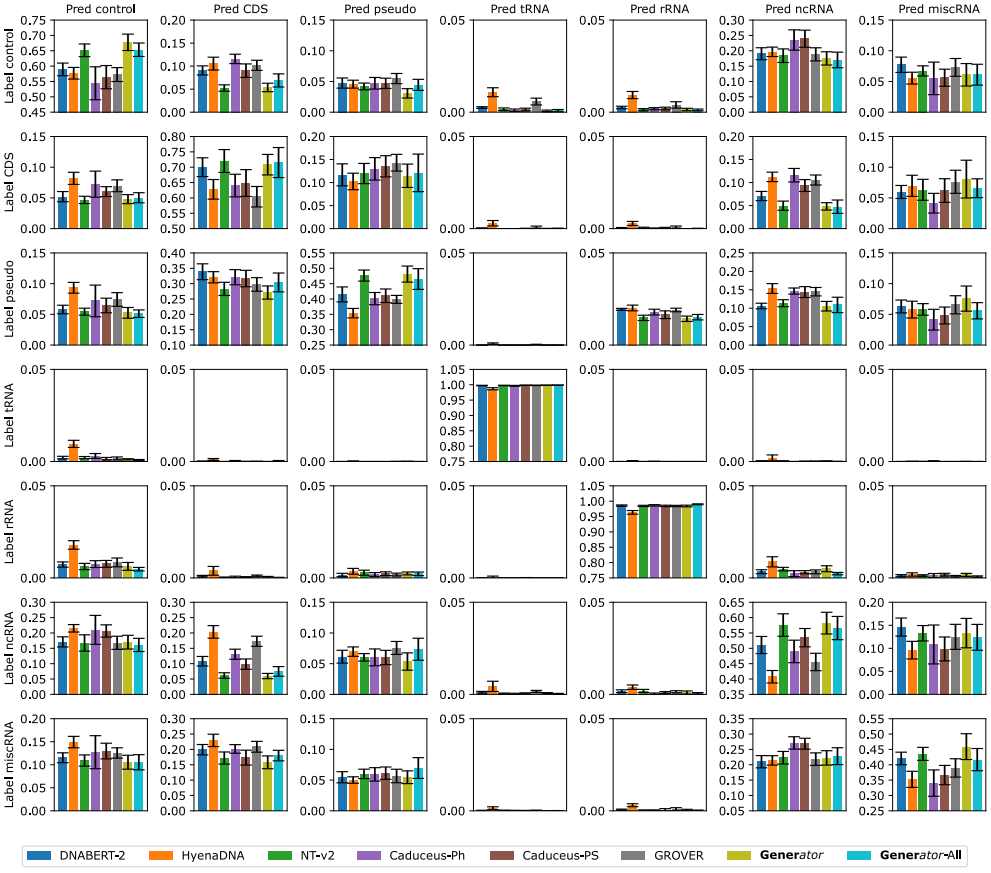

Figure S1: Evaluation of model performance on gene classification.

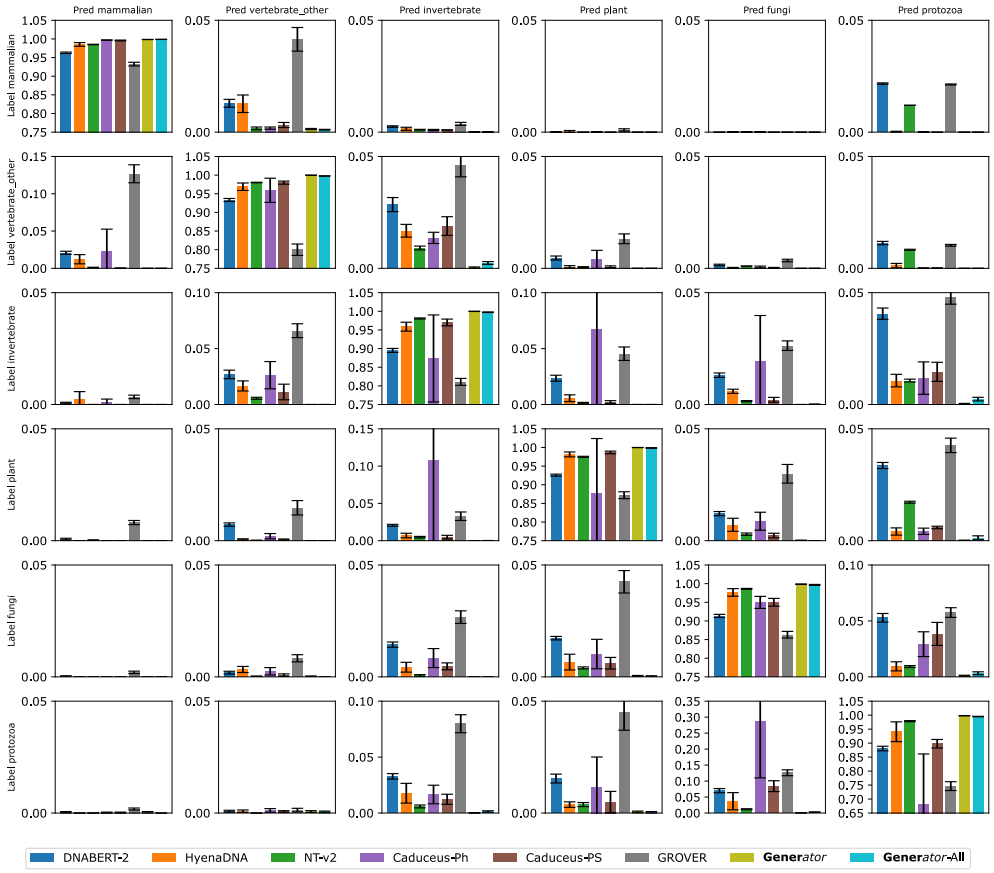

Figure S2: Evaluation of model performance on taxonomic classification.

Table S1: Pre-training data statistics (taxonomic group).

Taxonomic group	Number of genes	Number of nucleotides (bp)
Protozoa	1,107,499	2,034,179,213
Fungi	6,299,990	10,593,031,563
Plant	7,279,591	30,548,451,231
Invertebrate	7,602,349	71,979,748,208
Vertebrate (other)	10,915,710	161,686,774,317
Mammalian	6,837,553	109,406,723,311

Table S2: Pre-training data statistics (gene type).

Gene type	Number of genes	Number of nucleotides (bp)
Protein Coding	30,883,451	340,466,000,000
miscRNA	342,761	6,448,121,770
ncRNA	4,151,851	28,538,620,565
pseudo	2,322,278	10,307,467,170
rRNA	654,807	360,329,916
tRNA	1,687,523	128,397,507
tmRNA	21	9,793

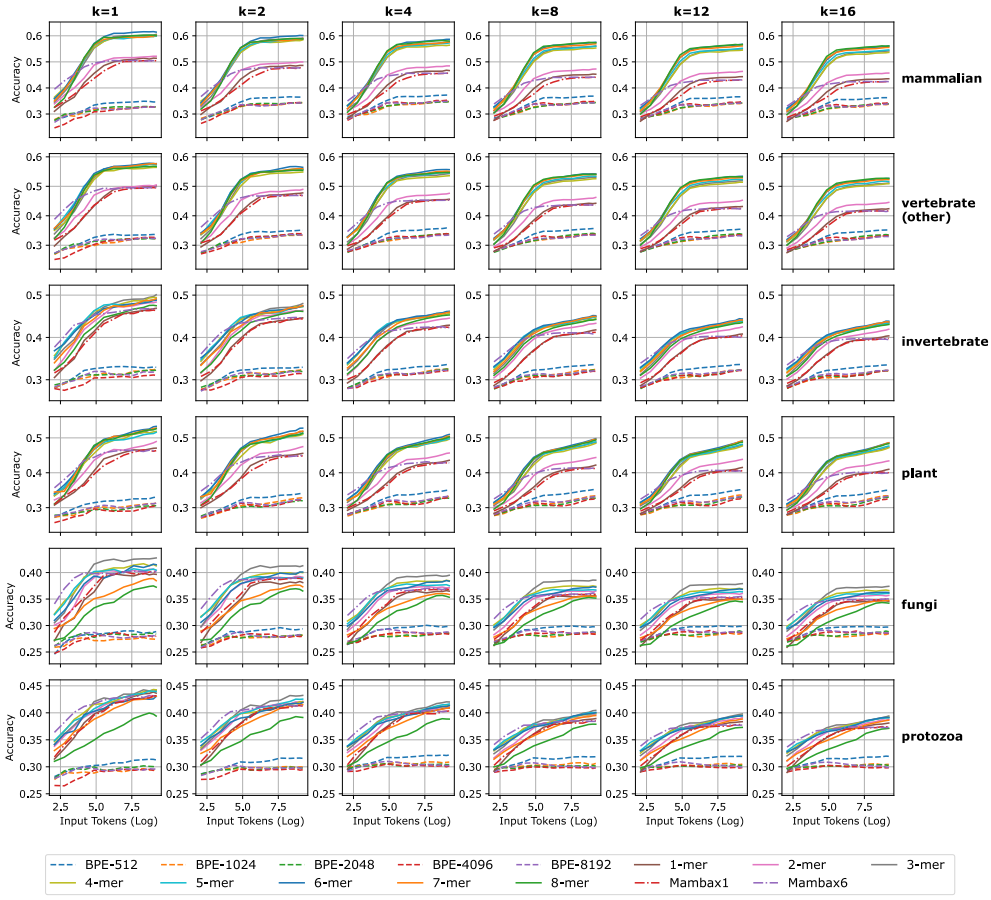

Figure S3: Evaluation of next K-mer prediction for tokenizer comparison.

Table S3: Summary of baseline models.

Model Name	Pre-train Task	Architecture	Tokenizer	Pre-train Data Scope	Pre-train Data Volume (bp)	Context Length (bp)	Parameter Size
Enformer	Supervised	Transformer	Single Nucleotide	Human, Mouse	64B	200K	252M
DNABERT-2	MLM	Transformer	BPE	Multispecies	32.5B	3K	117M
HyenaDNA	NTP	SSM	Single Nucleotide	Human	3B	1M	55M
NT-multi	MLM	Transformer	6-mer	Multispecies	174B	6K	2.5B
NT-v2	MLM	Transformer	6-mer	Multispecies	174B	12K	500M
Caduceus	MLM	SSM	Single Nucleotide	Human	35B	131K	8M
GROVER	MLM	Transformer	BPE	Human	3B	3K	87M
Gener*ator*	NTP	Transformer	6-mer	Multispecies (Gene Only)	386B	98K	1.2B
Gener*ator*-All	NTP	Transformer	6-mer	Multispecies	1.9T	98K	1.2B

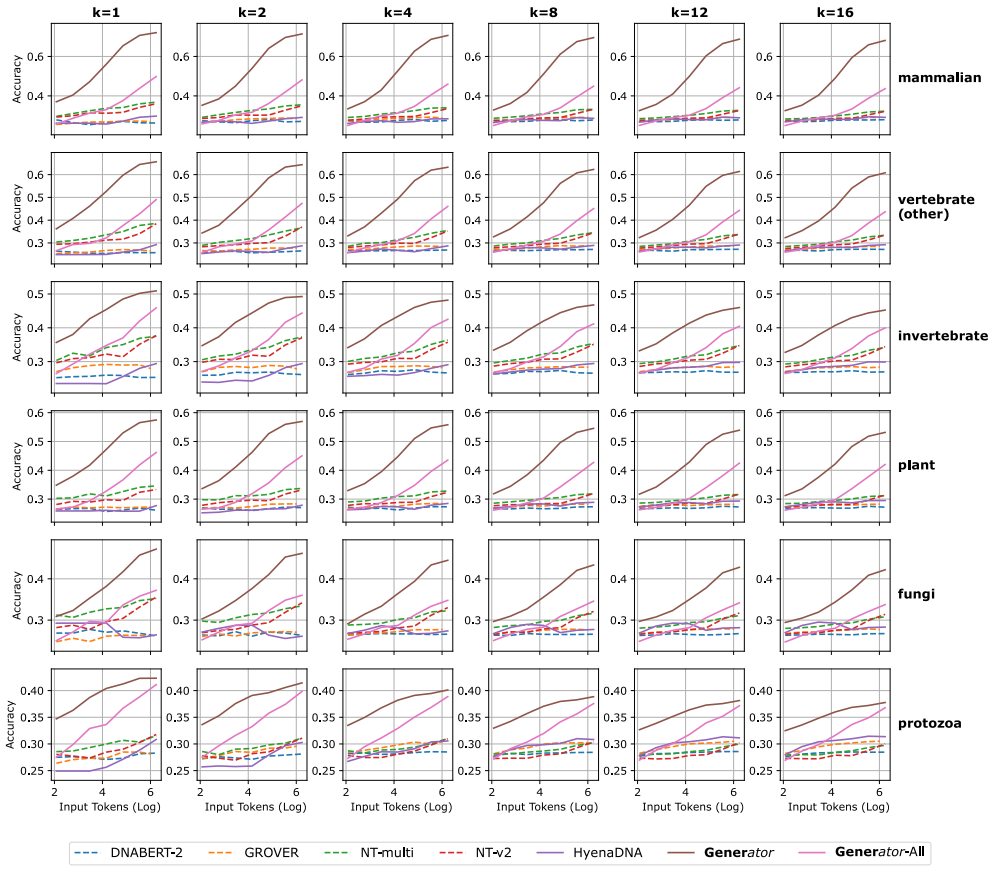

Figure S4: Evaluation of next K-mer prediction for model comparison.

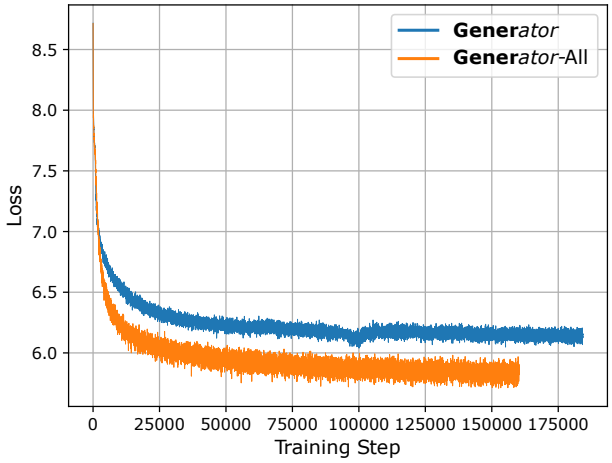

Figure S5: Pre-training losses of the **Gener***ator* and **Gener***ator*-All.

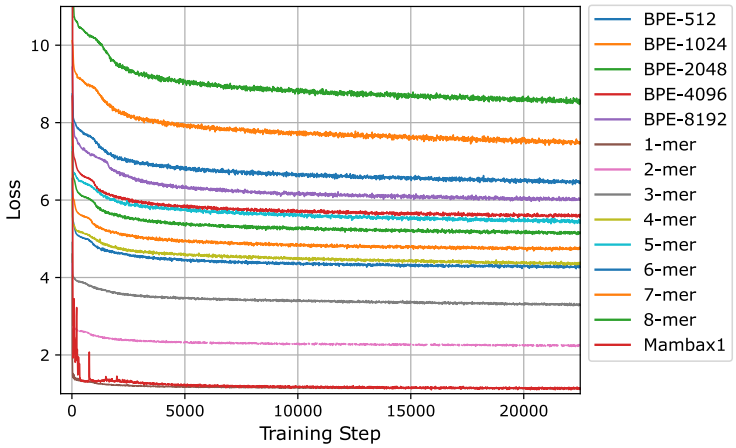

Figure S6: Pre-training losses of different tokenizers.

Table S4: Hyperparameter settings for the original Nucleotide Transformer tasks.

	NT-v2		Caduceus-Ph		Caduceus-PS		GROVER		**Gener***ator*		**Gener***ator*-All	
	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS
H3	$2e^{-5}$	256	$5e^{-4}$	64	$1e^{-3}$	128	$1e^{-4}$	256	$1e^{-4}$	64	$2e^{-5}$	64
H3K14ac	$1e^{-5}$	64	$2e^{-4}$	64	$5e^{-4}$	64	$5e^{-4}$	512	$2e^{-4}$	512	$5e^{-5}$	128
H3K36me3	$2e^{-5}$	64	$5e^{-4}$	128	$1e^{-3}$	256	$5e^{-5}$	64	$2e^{-4}$	512	$1e^{-5}$	64
H3K4me1	$5e^{-5}$	64	$2e^{-4}$	64	$1e^{-3}$	512	$5e^{-5}$	128	$1e^{-4}$	128	$2e^{-5}$	256
H3K4me2	$5e^{-5}$	256	$2e^{-4}$	128	$5e^{-4}$	512	$1e^{-4}$	256	$5e^{-5}$	256	$1e^{-4}$	512
H3K4me3	$5e^{-5}$	128	$5e^{-4}$	256	$5e^{-4}$	256	$1e^{-4}$	128	$5e^{-5}$	64	$5e^{-5}$	64
H3K79me3	$5e^{-5}$	128	$2e^{-4}$	64	$5e^{-4}$	256	$1e^{-4}$	512	$2e^{-4}$	128	$5e^{-5}$	128
H3K9ac	$5e^{-5}$	256	$5e^{-4}$	256	$5e^{-4}$	64	$1e^{-4}$	64	$1e^{-4}$	256	$1e^{-4}$	256
H4	$2e^{-5}$	128	$1e^{-4}$	128	$5e^{-4}$	128	$5e^{-5}$	512	$5e^{-5}$	512	$2e^{-5}$	128
H4ac	$5e^{-5}$	64	$5e^{-4}$	256	$2e^{-4}$	64	$5e^{-5}$	64	$1e^{-4}$	128	$5e^{-5}$	64
Enhancers	$2e^{-5}$	128	$2e^{-4}$	128	$5e^{-4}$	512	$5e^{-5}$	512	$1e^{-4}$	512	$2e^{-5}$	128
Enhancers types	$1e^{-4}$	128	$2e^{-4}$	256	$2e^{-4}$	64	$1e^{-4}$	512	$2e^{-4}$	512	$1e^{-4}$	128
Promoter all	$1e^{-5}$	64	$1e^{-4}$	64	$2e^{-4}$	64	$1e^{-4}$	64	$1e^{-5}$	64	$1e^{-5}$	64
Promoter non-TATA	$5e^{-5}$	256	$1e^{-3}$	512	$5e^{-4}$	512	$1e^{-4}$	256	$2e^{-5}$	64	$5e^{-5}$	256
Promoter TATA	$5e^{-5}$	128	$2e^{-3}$	512	$1e^{-3}$	256	$2e^{-4}$	512	$5e^{-5}$	512	$5e^{-5}$	128
Splice sites acceptors	$2e^{-5}$	64	$1e^{-3}$	64	$2e^{-3}$	512	$1e^{-4}$	64	$2e^{-5}$	128	$5e^{-5}$	64
Splice sites all	$5e^{-5}$	128	$2e^{-3}$	256	$1e^{-3}$	64	$1e^{-4}$	64	$2e^{-5}$	256	$5e^{-5}$	128
Splice sites donors	$5e^{-5}$	128	$1e^{-3}$	64	$1e^{-3}$	64	$5e^{-5}$	64	$1e^{-4}$	64	$5e^{-5}$	128

Table S5: Hyperparameter settings for the revised Nucleotide Transformer tasks.

	NT-v2		Caduceus-Ph		Caduceus-PS		GROVER		**Gener***ator*		**Gener***ator*-All	
	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS
H2AFZ	$2e^{-5}$	64	$1e^{-5}$	64	$1e^{-3}$	256	$1e^{-5}$	128	$2e^{-5}$	128	$1e^{-4}$	128
H3K27ac	$2e^{-5}$	128	$5e^{-4}$	64	$5e^{-4}$	64	$2e^{-5}$	128	$2e^{-5}$	256	$2e^{-5}$	128
H3K27me3	$5e^{-5}$	256	$1e^{-3}$	256	$1e^{-3}$	128	$1e^{-5}$	64	$5e^{-5}$	256	$5e^{-5}$	256
H3K36me3	$5e^{-5}$	128	$2e^{-4}$	64	$1e^{-3}$	256	$5e^{-5}$	256	$1e^{-5}$	256	$5e^{-5}$	128
H3K4me1	$5e^{-5}$	512	$5e^{-4}$	256	$1e^{-3}$	256	$5e^{-5}$	128	$5e^{-5}$	64	$1e^{-5}$	64
H3K4me2	$2e^{-5}$	256	$1e^{-3}$	256	$1e^{-3}$	256	$2e^{-5}$	128	$2e^{-5}$	64	$5e^{-5}$	512
H3K4me3	$2e^{-5}$	64	$1e^{-3}$	128	$5e^{-4}$	64	$1e^{-5}$	128	$2e^{-5}$	64	$5e^{-5}$	64
H3K9ac	$5e^{-5}$	512	$5e^{-4}$	128	$1e^{-4}$	64	$1e^{-5}$	64	$2e^{-5}$	128	$5e^{-5}$	256
H3K9me3	$1e^{-4}$	128	$2e^{-4}$	128	$1e^{-3}$	512	$1e^{-5}$	64	$1e^{-4}$	64	$1e^{-4}$	128
H4K20me1	$2e^{-5}$	128	$1e^{-3}$	256	$2e^{-4}$	64	$2e^{-5}$	128	$5e^{-5}$	256	$1e^{-5}$	64
Enhancer	$2e^{-5}$	256	$2e^{-4}$	128	$5e^{-4}$	512	$5e^{-5}$	256	$5e^{-5}$	64	$2e^{-5}$	64
Enhancer types	$5e^{-5}$	512	$1e^{-3}$	64	$2e^{-4}$	128	$2e^{-5}$	128	$2e^{-5}$	64	$5e^{-5}$	64
Promoter all	$1e^{-4}$	64	$5e^{-4}$	128	$5e^{-4}$	128	$1e^{-5}$	128	$1e^{-4}$	128	$5e^{-5}$	512
Promoter non-TATA	$2e^{-5}$	128	$5e^{-5}$	64	$5e^{-5}$	128	$2e^{-5}$	256	$2e^{-5}$	64	$1e^{-4}$	128
Promoter TATA	$2e^{-4}$	128	$5e^{-4}$	64	$1e^{-3}$	128	$5e^{-5}$	64	$5e^{-5}$	128	$1e^{-4}$	64
Splice acceptor	$5e^{-5}$	256	$1e^{-3}$	128	$5e^{-3}$	64	$1e^{-5}$	256	$5e^{-5}$	128	$5e^{-5}$	64
Splice site all	$5e^{-5}$	256	$1e^{-3}$	64	$2e^{-3}$	64	$1e^{-5}$	64	$1e^{-4}$	256	$1e^{-4}$	128
Splice site donors	$2e^{-5}$	128	$2e^{-3}$	64	$5e^{-3}$	64	$2e^{-5}$	128	$5e^{-5}$	128	$2e^{-5}$	64

Table S6: Hyperparameter settings for the Genomic Benchmarks.

	DNABERT-2		HyenaDNA		NT-v2		Caduceus-Ph		Caduceus-PS		GROVER		**Gener***ator*		**Gener***ator*-All	
	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS
Coding vs. Intergenomic	$1e^{-5}$	64	$1e^{-4}$	64	$5e^{-5}$	512	$2e^{-4}$	128	$2e^{-4}$	128	$5e^{-5}$	256	$2e^{-5}$	256	$1e^{-5}$	128
Drosophila Enhancers Stark	$5e^{-5}$	512	$1e^{-3}$	512	$5e^{-5}$	64	$5e^{-4}$	64	$1e^{-3}$	64	$1e^{-4}$	64	$2e^{-4}$	256	$1e^{-5}$	128
Human Enhancers Cohn	$5e^{-5}$	128	$5e^{-5}$	128	$5e^{-5}$	256	$2e^{-4}$	64	$2e^{-4}$	256	$2e^{-5}$	256	$1e^{-5}$	128	$2e^{-5}$	64
Human Enhancers Ensembl	$1e^{-4}$	256	$5e^{-4}$	512	$5e^{-5}$	128	$5e^{-4}$	256	$2e^{-4}$	64	$5e^{-5}$	512	$5e^{-5}$	128	$1e^{-4}$	512
Human Ensembl Regulatory	$5e^{-5}$	128	$1e^{-4}$	128	$2e^{-5}$	128	$5e^{-4}$	64	$5e^{-4}$	64	$1e^{-5}$	128	$1e^{-5}$	128	$2e^{-5}$	512
Human NonTATA Promoters	$2e^{-4}$	256	$1e^{-4}$	128	$5e^{-5}$	64	$5e^{-4}$	64	$5e^{-4}$	128	$5e^{-5}$	64	$5e^{-5}$	128	$1e^{-4}$	64
Human OCR Ensembl	$1e^{-4}$	256	$5e^{-5}$	256	$5e^{-5}$	128	$1e^{-3}$	256	$5e^{-4}$	128	$5e^{-5}$	64	$1e^{-5}$	64	$5e^{-5}$	128
Human vs. Worm	$2e^{-5}$	64	$1e^{-4}$	128	$2e^{-5}$	64	$2e^{-4}$	64	$2e^{-4}$	128	$2e^{-5}$	128	$2e^{-5}$	512	$2e^{-5}$	64
Mouse Enhancers Ensembl	$1e^{-5}$	64	$2e^{-4}$	128	$5e^{-5}$	64	$1e^{-3}$	64	$2e^{-4}$	128	$1e^{-5}$	128	$5e^{-4}$	512	$5e^{-5}$	64

Table S7: Hyperparameter settings for the Gener tasks.

	DNABERT-2		HyenaDNA		NT-v2		Caduceus-Ph		Caduceus-PS		GROVER		**Gener***ator*		**Gener***ator*-All	
	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS	LR	BS
Gene	$5e^{-5}$	64	$2e^{-4}$	128	$5e^{-5}$	128	$5e^{-4}$	128	$2e^{-4}$	64	$5e^{-5}$	128	$1e^{-5}$	64	$1e^{-4}$	64
Taxonomic	$5e^{-5}$	64	$2e^{-4}$	64	$1e^{-4}$	512	$2e^{-4}$	128	$5e^{-4}$	256	$1e^{-4}$	128	$1e^{-5}$	128	$5e^{-6}$	64

Table S8: Computational resource usage (gray color denotes that the model was trained for less than 1 epoch).

	Model	Computational Resources	Data-parallel Size
Development	BPE-512	11,551 V100 hours	32
	BPE-1024	10,562 V100 hours	32
	BPE-2048	8,431 V100 hours	32
	BPE-4096	11,155 V100 hours	64
	BPE-8192	8,118 V100 hours	32
	2-mer	28,380 V100 hours	64
	3-mer	15,270 V100 hours	32
	4-mer	11,635 V100 hours	32
	5-mer	9,202 V100 hours	32
	6-mer	9,494 V100 hours	64
	7-mer	8,133 V100 hours	64
	8-mer	7,883 V100 hours	64
	Single Nucleotide (llama)	1,350 A100 hours	8
	Single Nucleotide (mamba)	1,465 A100 hours	8
Pre-train	**Gener***ator*	11,793 A100 hours	32
	Gener*ator*-All	9,494 A100 hours	32
Downstream	46 runs per task per model	$>$ 64,000 V100 hours	8